A.N.Solutions
INDUSTRIAL GRADE WIRELESS

Getting Started

@ANY Design Development Kit Sub-1 GHz / 2.4 GHz IEEE 802.15.4 Standard

10 July 2020

A.N. Solutions GmbH
Am Brauhaus 5
01099 Dresden
Germany

List of Contents

List of Figures 3

List of Tables 3

1 Overview 4

1.1 Overview Development Kits 6

2 Connect the Individual Devices to the PC 7

3 First Step: A.N. Solutions ToolChain 8

3.1 A.N. Solutions ToolChain – SMS Monitor 8

3.2 Later Use of the SMS Monitor 10

4 Getting Started 11

4.1 Navigate the SMS 11

4.2 Reset All Devices to Factory Settings 11

4.3 Print Out Information 12

4.4 View, Edit and Save Profiles 13

4.5 Configure the SMS via Console 15

4.5.1 Configure the @ANY USB Dongle as Coordinator 15

4.5.2 Editing the IEEE Address 16

4.5.3 Configure the @ANY BRICK as an END Device 17

4.6 Building a Wireless Sensor Network (WSN) 17

4.6.1 Peer to Peer Network 17

4.6.2 Star Network 19

4.7 Mesh Network for SMS Pro User 21

4.7.1 Setting up the Network Participants 21

4.7.2 Setting Up and Commissioning the Test Environment 23

4.8 ADC Support for SMS Pro User 25

4.8.1 Basic Settings of the ADCs with the SMS Pro 25

4.8.2 Example of Battery Measurement with the ADC 26

5 Firmware Flash / Update 27

5.1 Structure of the @ANY Hardware 27

5.2 JTAG / SWD 29

5.3 Bootloader 29

5.4 Firmware Flash / Update with the SMS Updater 29

5.5 Firmware Flash / Update with Avrdude without JTAG / SWD 30

5.6 Firmware Flash / Update with Avrdude & JTAG / SWD 31

5.7 Firmware Flash / Update with JTAG / SWD & Atmel Studio 32

6 Software Tools 33
6.1 Sniffer Tool 33
6.2 Transparent UART – SMS PRO User 36
Reference Documents 38
Revision History 38
Disclaimer 39

List of Figures

Figure 1 Product overview 5
Figure 2 Device manager 7
Figure 3 Download Center 8
Figure 4 Tools 8
Figure 5 A.N. Solutions ToolChain 9
Figure 6 SMS Monitor 10
Figure 7 AT Command 11
Figure 8 ATI long 12
Figure 9 ATI small 12
Figure 10 Part one profile 13
Figure 11 Part two profile 13
Figure 12 Full profile overview 14
Figure 13 Full config dongle 16
Figure 14 Peer to Peer Network 18
Figure 15 Star - Network 19
Figure 16 Mesh peer to peer** 23
Figure 17 Dynamic Mesh** 24
Figure 18 Section @ANY BRICK X3 25
Figure 19 Circuit ADC0 26
Figure 20 @ANY900 Modules 27
Figure 21 @ANY900ARM-SC Modules 28
Figure 22 @ANY2400-SC Modules 28
Figure 23 @ANY2400-SC3 Modules 28
Figure 24 Atmel ICE Kit 29
Figure 25 ATANY-Updater 30
Figure 26 Tag Connect 32
Figure 27 Device programming head 33
Figure 28 Memories - Flash 33
Figure 29 Sniffer config 34

List of Tables

Table 1 Overview Development Kit 6
Table 2 ATI Command 12
Table 3 Config BRICK 17

1 Overview

We are pleased that you are interested in the @ANY Development Kit from AN Solutions. The @ANY DESIGN development kit allows you to set up an IEEE 802.15.4 / Zigbee wireless network using the provided tools, based on our industry-leading RF module platforms. You can easily learn how to use @ANY RF modules and develop unique wireless network applications for sensor data acquisition and control. Developers can try different network topologies to determine which one is best suited for the application. Sensors integrated into the BRICK board simplify the prototyping of a wireless sensor network.

The Smart MAC Suite (SMS) is a firmware based on the IEEE 802.15.4 standard, which can be configured with AT commands or by writing to the EEPROM. This enables the user to set up wireless networks without having to deal with difficult programming languages.

There are two types of firmware:

- Smart MAC Suite Base
- Smart MAC Suite Pro

The standard development kit includes the SMS Base. The purchased firmware package is flashed onto the hardware and is configured with the example SMS Monitor application. The SMS Pro can be purchased as a one-time-license on our website www.an-solutions.de. If you have purchased the development kit with the SMS Pro, the firmware is already on the hardware components. If you have subsequently purchased the SMS Pro, it must first be flashed onto the hardware components. To learn how to do this, jump to chapter 5.

Before you start, please make sure that all the parts are included. Also, you need an command line Tool to open a serial connection with the @ANY USB dongle or @ANY BRICK and your PC. We recommend the free tool HTerm.

The document describes the basic steps for using the SMS software and setting up a simple network between several participants. The Getting Started is firmware independent. For more complex applications, please read the "SMS Command Reference" document. The commands listed here, are used to get started with the SMS. Not all commands, options and arguments are presented. The document is available free of charge on our website.

This document describes the setup in Microsoft Windows version 8 and up. The latest Windows version that has been tested is 10.0.18362. The setup under Linux behaves somewhat differently.

We recommend reading the SMS Command Reference document in parallel with the Getting Started.

Section 5 describes an optional procedure. If it is followed, additional hardware and software is required that we recommend purchasing in advance:

- ATMEL-ICE-BASIC – Atmel ICE Basic Kit
- Atmel Studio 7 (As of 2019)
- Special JTAG Connector such as Tag Connect TC2050-IDC-NL-050 for the @ANY 2400-SC-Pro USB dongle
- Special JTAG Connector such as Tag Connect TC2030-CTX-NL 6-Pin for the @ANY900ARM-SC USB dongle

The Atmel ICE Basic Kit can be purchased from Microchip at www.microchipdirekt.com. Atmel Studio is also required to read the EEPROM. Atmel Studio can be downloaded free of charge from Microchip at www.microchip.com.

To flash the @ANY USB dongle over the JTAG connector, you need a special JTAG connector like the TC2050-IDC-NL-050 or the TC2030-CTX-NL 6-Pin. The @ANY USB dongle has no 6/10 Pin 50mil male connector on their PCB.

This Getting Started was created with the AT-ANY-DESIGN-2400-SC-3-2 DevKit. This DevKit works with the 2.4GHz frequency band. All settings made here refer to this frequency band and must be changed to the Sub-1 GHz frequency band in some places.

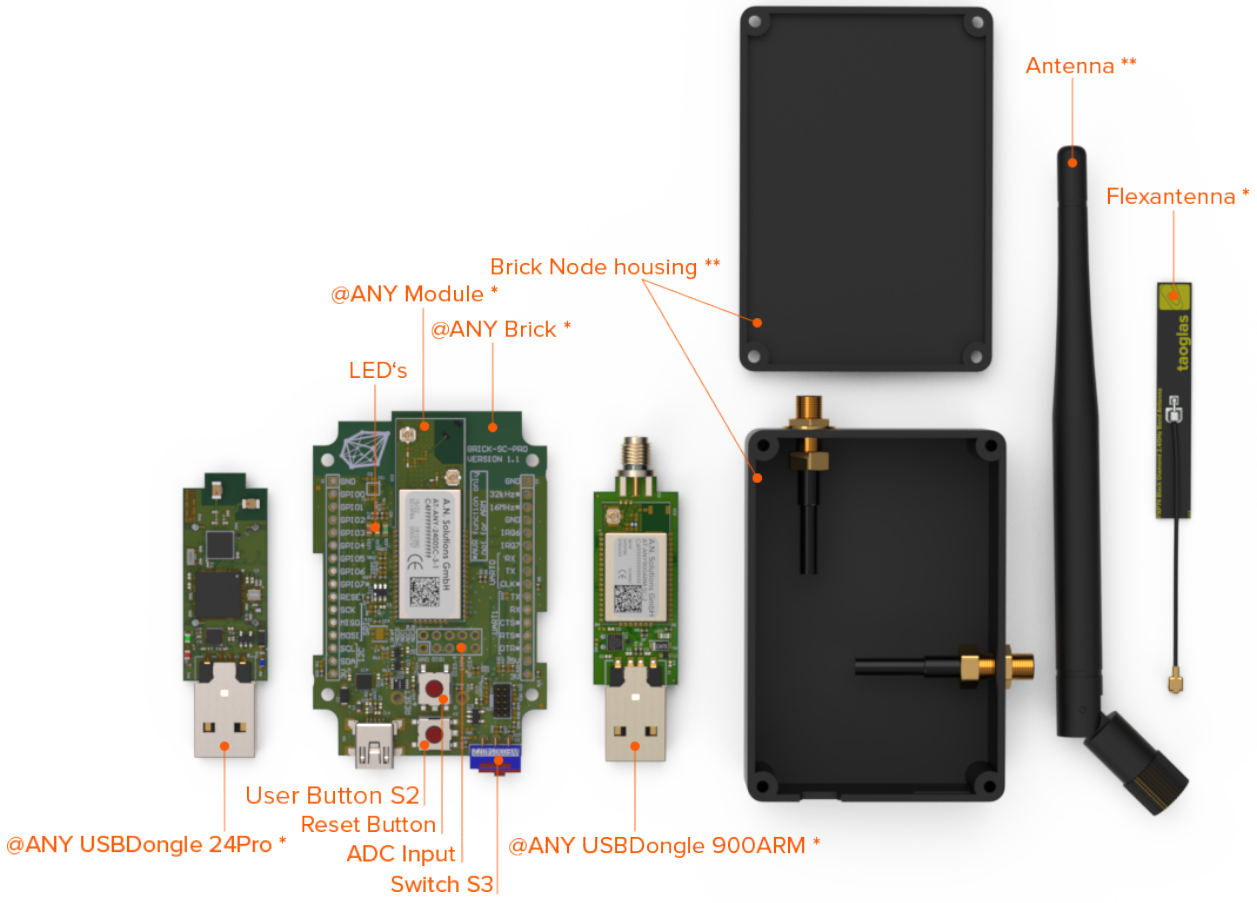


Figure 1 Product overview

* Development Kit dependent
** Optional

1.1 Overview Development Kits

BRICK Part.-No.:	Sensors & Features on @ANY BRICK Boards						Supported Features by @ANY Module		
	LM73 Temp. Sensor	SI7006 Humidity & Temp. Sensor	BMA222E Acceleration Sensor	Configurable LEDs	User Button	Reset Button	U.FL Connector	Integrated Antenna	On-Board Data-Flash
AT-ANY2400-SC-1 BRICK	yes	yes	yes	3	1	1	-	yes	-
AT-ANY2400-SC-2 BRICK	yes	yes	yes	3	1	1	yes	-	-
AT-ANY2400-SC-3-1 BRICK	yes	-	-	3	1	1	yes	yes	2 Mbit SST25VF0202
AT-ANY2400-SC-3-2 BRICK	yes	-	-	3	1	1	2	-	2 Mbit SST25VF0202
AT-ANY900-1 BRICK	yes	yes	yes	3	1	1	-	yes	2 Mbit SST25VF0202
AT-ANY900-2 BRICK	yes	yes	yes	3	1	1	yes	-	2 Mbit SST25VF0202
AT-ANY900ARM-SC BRICK	yes	yes	yes	3	1	1	yes	-	4 Mbit AT25XE04
Used @ANY USB Dongles:									
AT-ANY2400-SC-Pro USB	-	-	-	2	-	-	-	2	-
AT-ANY900ARM-SC USB	-	-	-	1	-	-	(SMA)	-	4 Mbit AT25XE04

Table 1 Overview Development Kit

2 Connect the Individual Devices to the PC

To be able to configure the SMS, a serial connection must be established between the hardware and your PC. To do this, plug the @ANY USB dongle into a free USB port on your PC. The @ANY BRICK boards are connected to the PC via a USB mini cable. The green LED lights up (standard firmware settings). As of Windows 8, the hardware setup wizard installs the necessary drivers for FT230X and UART.

After the installation, open the device manager with the keyboard-button-combination “Windows-Logo-Button + Pause”

In the opened windows choose the Device Manager. Another USB serial port (COM <n>) appears under the item “Ports (COM & LPT)”.

If this is not the case, the driver must be installed manually. To do this, go to the FTDI website at www.ftdichip.com, navigate to “drivers” on the left-hand website menu and download the latest VCP driver for your operating system.

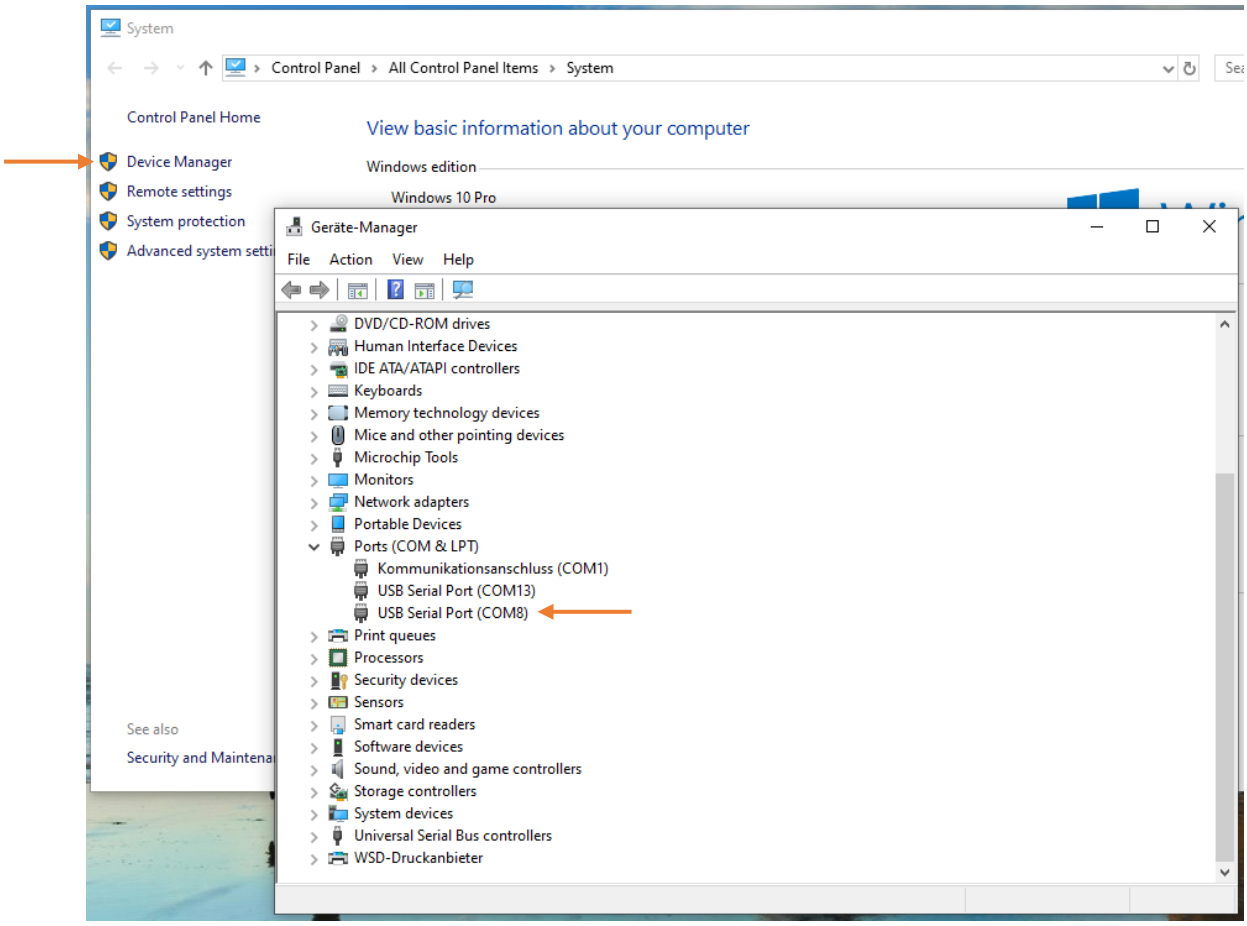


Figure 2 Device manager

Figure 2 shows the connected @ANY USB dongle under USB Serial Port (COM8).

3 First Step: A.N. Solutions ToolChain

3.1 A.N. Solutions ToolChain – SMS Monitor

The SMS Monitor offers you the option of using a wireless sensor network (WSN) and displaying it visually. It does not require any complex installation and is available free of charge as part of our A.N. Solutions ToolChain. The A.N. Solutions ToolChain can be downloaded from our website (currently available only for Microsoft Windows 10).

To do this, navigate to the download area. You can find the A.N. Solutions ToolChain.

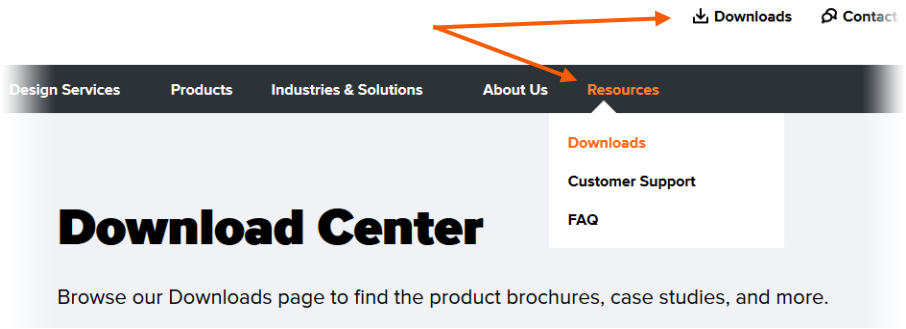


Figure 3 Download Center

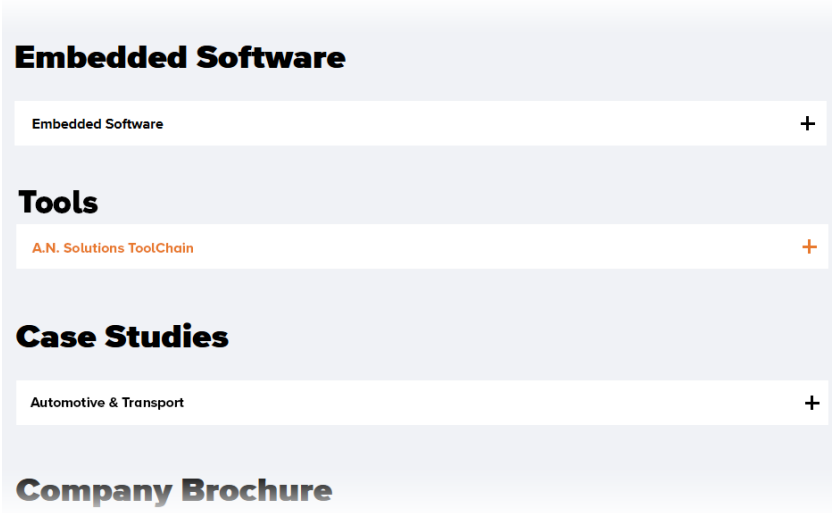


Figure 4 Tools

Download the A.N. Solutions ToolChain and unzip them to any location on your Windows PC. Launch the A.N. Solutions ToolChain VXXX.exe.

The tool starts and it is switched to the TAB SMS Monitor. You can now connect to the coordinator in the SMS monitor. @ANY USB dongle is the coordinator for your @ANY DESIGN development kit. All hardware components supplied are preconfigured with the example of SMS Monitor and tested for functionality.

Insert the @ANY USB dongle into a free USB port on your PC and make sure that it is correctly recognized (Chapter 2).

The COM port of the @ANY USB dongle is selected in the "Connection Parameters" area. The port can vary for different dongles and systems. The COM port list can be updated by pressing the "Refresh port list" button. The baud rate settings must be left at 38400bd.

Connect to the @ANY USB dongle by pressing "OK". The "Connection Parameters" window disappears and is replaced by a white background. There can only be one connection with one coordinator at a time. It is not possible to use the SMS Monitor in parallel with an command line tool and to establish a parallel connection with a device.

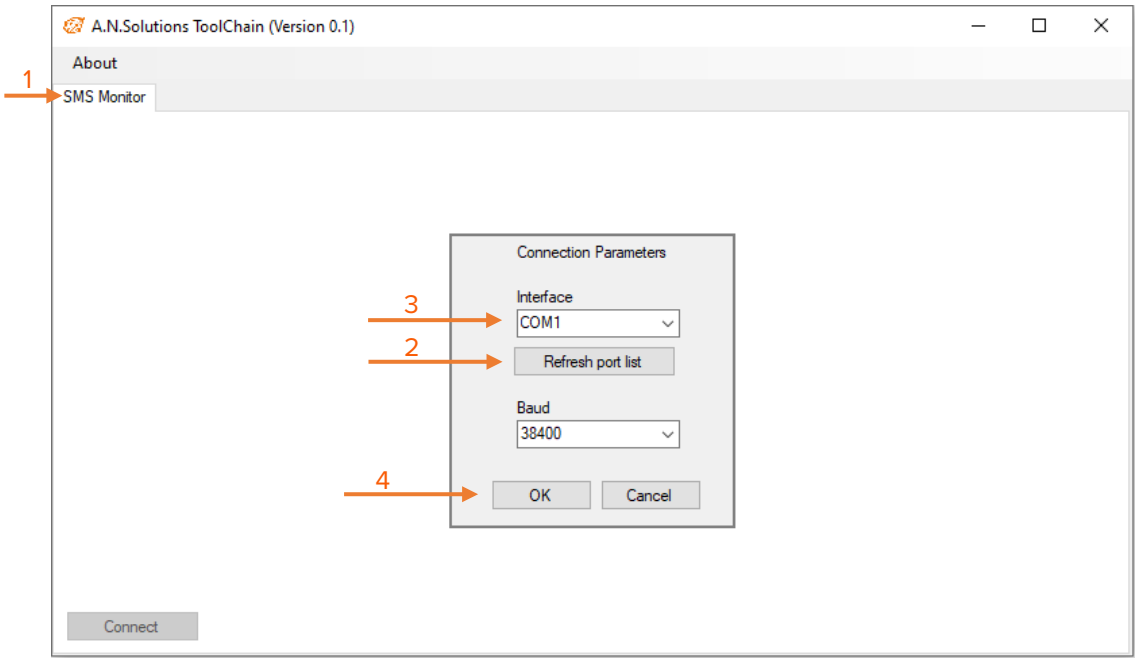


Figure 5 A.N. Solutions ToolChain

The supplied battery holder can now be attached to the @ANY BRICKs. The @ANY BRICKs must be powered by 2 x 1.5 V AAA batteries, which are not supplied with the kit.

With the S3 switch, each @ANY BRICK is switched on one after the other. The @ANY BRICKs that are switched on now appear in the SMS monitor and send a temperature value to the coordinator every second as illustrated by Figure 6 on the next page.

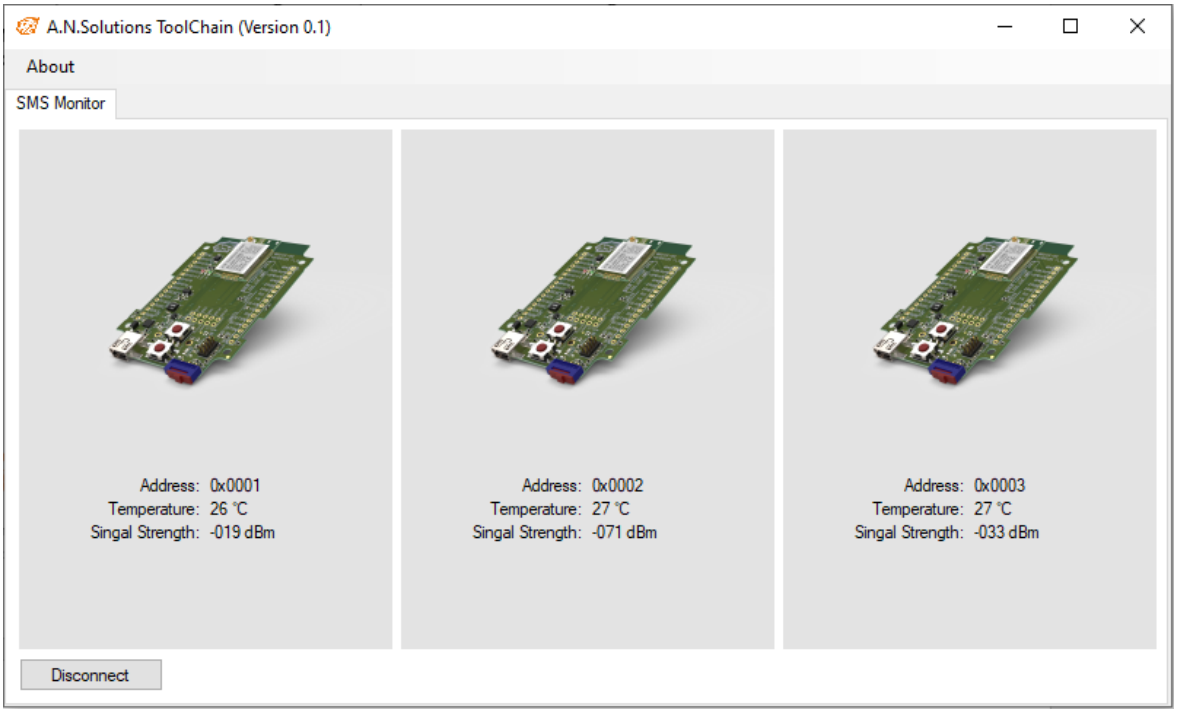


Figure 6 SMS Monitor

The short address, the temperature and the signal strength of each @ANY BRICK are listed in the SMS monitor. This is suitable, for example, for performing range measurements.

Congratulations, you've just launched your first WSN!

3.2 Later Use of the SMS Monitor

This chapter is not intended for getting started and only serves to reactivate the SMS Monitor later. For this, a basic knowledge of handling our SMS is required. If you are not yet familiar with our SMS, skip this chapter.

If you want to use the SMS Monitor at a later time, but its settings have been erased on the devices, follow the steps below for each device. Before you start, familiarize yourself with Chapters 2 and 4.1.

To reactivate the SMS Monitor, execute the following command chains for each device.

@ANY USB Dongle *Use the required command lines for the corresponding DevKits.*	
Complete command:	Response:
(2.4 GHz) ATH1&FS220=0SX202=12ABS208=20S200=0S0=1&W0&Y0Z↵	OK & Device restarts
(1 GHz EU) ATH1&FS220=0SX202=12ABS208=0S200=0S0=1&W0&Y0Z↵	OK & Device restarts
(1 GHz US) ATH1&FS220=0SX202=12ABS208=1S200=0S0=1&W0&Y0Z↵	OK & Device restarts

@ANY BRICKs *Use the required command lines for the corresponding DevKist.*	
Complete command:	Response:
ATH1&F&Y0&W0&W1Z ↵	OK & Device restarts
AT*AS234=2S238=1DSA245XS234=2 ↵	+NOT ASSOCIATED (EB) & ERROR
(2.4 GHz) ATS220=2SX202=12ABS208=20S200=0S236=4S18=1&W0&Y0Z ↵	OK & Device restarts
(1 GHz EU) ATS220=2SX202=12ABS208=0S200=0S236=4S18=1&W0&Y0Z ↵	OK & Device restarts
(1 GHz US) ATS220=2SX202=12ABS208=1S200=0S236=4S18=1&W0&Y0Z ↵	OK & Device restarts

4 Getting Started

4.1 Navigate the SMS

To communicate with the @ANY USB dongle and the @ANY BRICK, a command line tool is useful. Any command line tool can be used that enables a serial connection. Switch to serial connections in your command line tool. Connect to the COM port assigned to the @ANY USB dongle and set the baud rate to 38400bd. It also makes sense to set 8 data bits, no parity bits and one stop bit (8 N 1).

As soon as the connection between hardware and software is established, there is a bidirectional commutation channel between the @ANY USB dongle and the user. The user can now enter commands into the terminal and the @ANY USB dongle will respond.

CAUTION! There can only be one connection with one tool and one @ANY USB dongle/BRICK at a time.

The existing connection can be checked by entering the AT command. The @ANY USB dongle responds with an OK.

```
AT
OK
```

Figure 7 AT Command

Each command line starts with an AT command. No distinction is made between upper and lower case, except for a specific command to store the IEEE address (AT&Waddr). A command line can consist of several commands. These are processed one after the other. A faulty command is returned with an error message. Ambiguous command sequences can be explicitly separated by the "X". The maximum length of a command line is 114 characters. A device can be restarted using the ATZ command. A Device can be reset to factory defaults with the command AT&F.

4.2 Reset All Devices to Factory Settings

Before you can begin with the Getting Started, the SMS Monitor example must be deleted from the @ANY USB dongle and the @ANY BRICKs.

Deleting the SMS Monitor example must be done individually on each device. Therefore, follow the steps below for each device.

To reset the @ANY USB dongle or @ANY BRICK to the factory settings, enter the following command line in the command line tool and confirm the entry with Enter ↵.

Complete command:
ATH1&F&Y0&W0&W1Z ↵
Response:
OK & & Device restarts

4.3 Print Out Information

As an example, we want to pull out information about the connected device. The syntax looks as follows: ATI <n>
The parameter <n> stands for the arguments in Table 1.

Argument	Result
0	<ISM band>
1	<firmware rev.>
2	<Build-ID>
3	ANS - Smart MAC Suite V-<firmware rev.>
4	<board type>
5	<Program memory CRC>
6	<radio chip type>
7	<license owner> (Pro version only>
any	OK (argument in 1 ... 6 (7)) or ERROR

Table 2 ATI Command

Example:

single:
ATI0 ↵
2400
OK
ATI3 ↵
ANS – Smart Mac Suite V-1.47
OK
ATI4 ↵
@ANY2400SC-3
OK
ATI6 ↵
ATMEGA128RFA1
OK

```
ATI0
2400
OK
ATI3
ANS - Smart MAC Suite V-1.47
OK
ATI4
@ANY2400SC-3
OK
ATI6
ATMEGA128RFA1
OK
```

Figure 8 ATI long

combinate:
ATI0I3I4I6 ↵
2400
ANS – Smart MAC Suite V-1.47
@ANY2400SC-3
ATEMEGA128RFA1
OK

```
ATI0I3I4I6
2400
ANS - Smart MAC Suite V-1.47
@ANY2400SC-3
ATMEGA128RFA1
OK
```

Figure 9 ATI small

4.4 View, Edit and Save Profiles

There are two possible ways to create profiles with the SMS. A profile stores several configurations. A profile is an image of all S-Register. Only the most important S-Registers are output as the actual profile with the AT&V command. The first part of the profile describes the settings of the UART and various communication options, which are listed as S-Registers and addresses.

E1 Command Echo | F1 Online Echo | Q0 Result Code Opt. | V1 Result Code Format | &C0 DCD Opt.

&D0 DTR Opt. | &K0 Local Flow Control | &Q0 Communication Mode | &R1 RTS/CTS Opt. | &S0 DSR Opt.

```
E1 F1 Q0 V1 &C0 &D0 &K0 &Q0 &R1 &S0
S00:000 S02:043 S03:013 S04:010 S05:008 S17:016 S18:000
```

Figure 10 Part one profile

S00 Auto Mode ATA | S02 Esc character | S03 CR character | S04 LF character | S05 BS character

S17 Timer Prescaler | S18 Timer

The second part contains the option for a wireless sensor network (WSN). A profile is therefore only an orderly overview of the most important configurations.

```
IEEE  ADDRESS:  FE.95.3C.83.87.97.4F.FF
DEVICE ROLE:    END DEVICE
PAN ID:  FFFF   SHORT ADDRESS:  0000
CHANNEL: 20 (0) CHANNEL MASK: 07FFF800
```

Figure 11 Part two profile

IEEE ADDRESS: *EEPROM*
The IEEE address is a hardware address. It is only shown in the active profile. The IEEE address is not saved in a profile, but saved in the EEPROM. When booting, the SMS first searches for the IEEE address in the EEPROM, if none is detected there, the hardware chip DS2411 is searched, if it's still not available, the SMS generates a random IEEE.

DEVICE ROLE: *Register S220 8bit 0=Coordinator, 1=FF device (SMS_Pro), 2=END device*
The role of the device is important for the later task in the network. A coordinator creates a network and is waiting for end devices and FF devices to connect to it. This is comparable to a network host that takes over the coordination in the network. An end device can only connect to a coordinator and only send or receive data from a coordinator. An FF device (SMS Pro) is coordinator independent and can exchange data with other FF devices. It offers the possibility to create small and large mesh/tree networks.

PAN ID: *Register S202 16bit*
The PANID is the network address in the physical network. It must be the same for all participants in the same network and is output as a hexadecimal number or a decimal number.

SHORT ADDRESS: *Register S200 16bit*
This is the address of the individual participants in the network and serves as an identification. The coordinator assigns it to the individual participants. The coordinator must always be assigned the SHORT ADDRESS 0x0000 manually. Addressing the individual participants via the IEEE address is also possible.

CHANNEL: *Register S208 8bit*
This is the physical channel to exchange data with other devices in the network. It must be the same for all participants in the same network with the same PANID. It is important to ensure that the device with a frequency of 868,3 MHz EU is assigned to channel 0, with the frequency 915MHz 1 - 10 USA and for Japan and China please contact the A.N.S. Customer Support. Devices with a frequency of 2.4 GHz are assigned a channel in the range of 11 - 26.

CHANNEL MASK: *Register S204 32bit*

The IEEE 802.15.4 standard has a 32 bit channel mask. Each individual bit, in the channel mask, maps a channel. The SMS uses a total of 26 channels. A range of channels can be defined with the channel mask. This is used for scanning channels, for example, to query several channels.

For exact information about each S-Register, read the "SMS Command Reference" document. For information about network parameters, read the standard IEEE 802.15.4-2006(revison of IEEE std 802.15.4-2003).

```
AT&V
ACTIVE PROFILE:
E1 F1 Q0 V1 &C0 &D0 &K0 &Q0 &R1 &S0
S00:000 S01:000 S02:043 S03:013 S04:010 S05:008
S10:000 S12:000 S17:016 S18:000 S25:005 S26:001 S38:020
  IEEE  ADDRESS:  FE.95.3C.83.87.97.4F.FF
  DEVICE ROLE:    END DEVICE
  PAN ID:  FFFF   SHORT ADDRESS:  0000
  CHANNEL: 20 (0) CHANNEL MASK: 07FFF800

STORED PROFILE 0:
E1 F1 Q0 V1 &C0 &D0 &K0 &Q0 &R1 &S0
S00:000 S02:043 S03:013 S04:010 S05:008 S17:016 S18:000
  DEVICE ROLE:    END DEVICE
  PAN ID:  FFFF   SHORT ADDRESS:  0000
  CHANNEL: 20 (0) CHANNEL MASK: 07FFF800

STORED PROFILE 1 (DEFAULT):
E1 F1 Q0 V1 &C0 &D0 &K0 &Q0 &R1 &S0
S00:000 S02:043 S03:013 S04:010 S05:008 S17:016 S18:000
  DEVICE ROLE:    END DEVICE
  PAN ID:  FFFF   SHORT ADDRESS:  0000
  CHANNEL: 20 (0) CHANNEL MASK: 07FFF800

OK
```

Figure 12 Full profile overview

You can choose between STORED PROFILE 0 and 1, but one of them must be the standard profile. The standard profile can switch with the command AT&Y<n>. The parameter <n> stands for the profile 0 or 1. Which of these two profiles was selected as the standard is indicated by a (DEFAULT), next to the profile number.

If a profile is edited, it only happens in the active profile. If the changes are not saved in one of the STORED PROFILE, the changes will be lost when you restart the device. A change in a profile can be saved with the command AT&W<n>. (0 or 1)

You change the profile parameters with the command ATS<Reg.ID> for designated S-Registers. S-Registers can be written with a decimal number or a hexadecimal number. The size of the decimal-/hexadecimal number depends on the respective register.

Example: Write the number 27 in the S-Register 240

Decimal: ATS240=27 Hex: ATSX240=1B

The X after the ATS command is important for writing hexadecimal numbers in S-Register. With the command ATS<Reg.ID>? you can read S-Registers. For a hexadecimal response, use the X between the ATS command and the S-Register. ATSX240?

4.5 Configure the SMS via Console

4.5.1 Configure the @ANY USB Dongle as Coordinator

Later in the Getting Started, you will see how the @ANY USB dongle can serve as the coordinator. It is always advantageous to set up the @ANY USB dongle as a coordinator. Because the @ANY USB dongle has a higher performance and is bound to the USB port.

The @ANY dongle is given the role of a coordinator. It transmits on the frequency 2.4 GHz and gets channel 20. For the Sub-1 GHz frequency band, other channels must be set according to the country. See chapter 4.4 Description of Channels. The PANID is freely selectable and is set to 0x12AB. The SHORT ADDRESS is set to 0x0000 and the channel mask can be neglected. The settings are saved in profile 0 and this is also the standard profile. After setting up the S-Registers, the device restarts.

command:	response:	explanation:
ATS220=0 ↵	OK	Set S-Register 220 to 0 dec., Coordinator
ATSX202=12AB ↵	OK	Set the PANID in the S-Register 202 as a hex number to 12AB
ATS208=20 ↵	OK	Set the channel in S-Register 208 to 20 dec.
ATS200=0 ↵	OK	Set the entire S-Register 200 to 0, short address
AT&W0 ↵	OK	Save the settings in profile 0
AT&Y0 ↵	OK	Define the profile 0 as the standard profile
ATZ ↵	.. OK	Reboot the device

combinate: ATS220=0SX202=12ABS208=20S200=0&W0&Y0Z

```
AT&V
ACTIVE PROFILE:
E1 F1 Q0 V1 &C0 &D0 &K0 &Q0 &R1 &S0
S00:000 S01:000 S02:043 S03:013 S04:010 S05:008
S10:000 S12:000 S17:016 S18:000 S25:005 S26:001 S38:020
  IEEE  ADDRESS:  FE.95.3C.83.87.97.4F.FF
  DEVICE ROLE:    COORDINATOR
  PAN ID:  12AB   SHORT ADDRESS:  0000
  CHANNEL: 20 (0) CHANNEL MASK: 07FFF800

STORED PROFILE 0 (DEFAULT):
E1 F1 Q0 V1 &C0 &D0 &K0 &Q0 &R1 &S0
S00:000 S02:043 S03:013 S04:010 S05:008 S17:016 S18:000
  DEVICE ROLE:    COORDINATOR
  PAN ID:  12AB   SHORT ADDRESS:  0000
  CHANNEL: 20 (0) CHANNEL MASK: 07FFF800

STORED PROFILE 1:
E1 F1 Q0 V1 &C0 &D0 &K0 &Q0 &R1 &S0
S00:000 S02:043 S03:013 S04:010 S05:008 S17:016 S18:000
  DEVICE ROLE:    END DEVICE
  PAN ID:  FFFF   SHORT ADDRESS:  0000
  CHANNEL: 20 (0) CHANNEL MASK: 07FFF800

OK
```

Figure 13 Full config dongle

4.5.2 Editing the IEEE Address

The IEEE address is a hardware address. This is represented as a hexadecimal number and is stored in the first eight bytes in the EEPROM. You can see whether an IEEE address is preconfigured using the AT+CF0? Command. The IEEE address noted there always begins with a C47DFE. If this is not the case, it can be assumed that there is no IEEE address defined by ANS in the EEPROM.

AT+CF0=<*xxxxxxxxxxxxxxxx*> OK	The IEEE address is specified in HEX.
AT&Waddr OK	Save the IEEE in the EEPROM. The upper and lower case must be considered.

Complete command:
AT+CF0=00000000000000FF&WaddrZ
Response:
OK & Device restarts

4.5.3 Configure the @ANY BRICK as an END Device

Later in the Getting Started, you will see how the @ANY BRICK can serve as an end-device powered by two 1,5 V AAA batteries. First connect the @ANY BRICK to a free USB port on your PC. Check switch S3 for correct switch position and the green LED lights up (standard firmware settings). The @ANY BRICK will be displayed in the device manager as a COM port and can be configured using an command line tool.

The @ANY BRICK must in the same network as the @ANY USB dongle, so they can communicate with each other. This results in the following settings for the @ANY BRICK:

Device Roll	PAN ID	Channel*	Short address
End device	12AB	20*	Set by coordinator

Table 3 Config BRICK

* For the Sub-1 GHz frequency band, other channels must be set according to the country. See chapter 4.4 Description of Channels. 208=Channel

```
Complete command:
ATS220=2SX202=12ABS208=20S200=0&W0&Y0Z
response:
OK & Device restarts
```

It makes sense to assign a defined IEEE address to the end devices because the coordinator remembers them during the session and assigns a short address. The end devices themselves are addressed via the short address in the network. If the same device reports to the coordinator with a different IEEE address, it will be assigned a new short address and other program routines that work with the old short address will no longer reach the end device.

```
Complete command:
AT+CF0=00000000000000EE&WaddrZ
response:
OK & Device restarts
```

4.6 Building a Wireless Sensor Network (WSN)

To build a working network with the @ANY USB dongle and @ANY BRICK, they must be set up as described in the chapters above. A coordinator creates with the ATA command a working network and the end device connects \with the ATA command. The coordinator saves the end devices dialled in using the IEEE addresses and distributes a short address to the end devices. The data exchange in the network takes place via the short addresses, but can also be implemented via the IEEE addresses. If a coordinator is switched off or restarted, it will lose this list. This list can be called up using command AT&V2. The short addresses are assigned in the order in which the end devices dial in. If an end device is known to a coordinator but not connected, its IEEE appears in this list, but has a short address of 0000. An existing connection can be terminated using the ATH command. Networks are searched by using the AT+S command.

4.6.1 Peer to Peer Network

Now we want to build a simple peer to peer network between the @ANY USB dongle and one @ANY BRICK. In addition, data must be exchanged between the two participants.

The previously configured settings are checked using the AT&V command. The ATA command is executed on the coordinator. The same command is then carried out on the end device. The participants list in the coordinator can be checked with the AT&V2 command.

Coordinator:	End-device:
ATA ↵ OK	
	ATA ↵
+DEVICE ASSOCIATED: 0001 (80)	+ASSOCIATED: 0001 OK
AT&V2 ↵ 00: 0001 / 00.00.00.00.00.00.00.EE (80)	

OK

The coordinator has set up a WSN in which the end device has dialled into. When dialling into a WSN successfully, the end device reports +ASSOCIATED ant the short address which was assigned to it. The coordinator also reports that an end device has dialled in and which short address has been assigned to it.

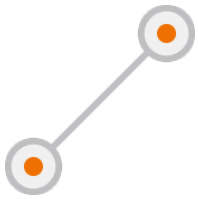


Figure 14 Peer to Peer Network

It is possible to send and poll data between the participants with the ATD command. The coordinator always specifies the destination, whereby the end device can only send data to the coordinator. An example on the next page make this clear.

Example:

Coordinator:	End-device:
	ATD;↵
	HELLO↵
	ID 001: 5 BYTES to 0000
	OK
+ DATA: 5 BYTES FROM 0001 (074)	+ SENT: ID 001
HELLO	
ATD1;↵	
HELLO BACK↵	
ID 002: 10 BYTES TO 0001	
OK	
	ATD?↵
	OK
+SENT: ID 002	+DATA: 10 BYTES FROM 0000 (077)
	HELLO BACK

The end device sends an ASCII Code to the coordinator. A “ ; “ must to be set next to the ATD command, otherwise the SMS waits for the length of the data to be send (see SMS Command Reference Command ATD). The sent data is displayed directly by the coordinator. If the coordinator sends data to the end device, the coordinator must specify the short address of the end device. But the +SEND notification is missing. The coordinator stores the sent data for about a minute. The end device must ask for data with the ATD? Command. It makes sense to have the end device ask for data in intervals. The number in brackets at the receiver indicates the reception power in db.

Congratulations! Your first peer to peer network has been set up.

4.6.2 Star Network

The star network is based on the peer to peer network. In this network, the data exchange is based on several end devices, which communicate with a coordinator. The end devices cannot communicate with each other. For this network, the remaining two @ANY BRICK boards are configured for the existing network. The existing peer to peer network is dissolved by restarting the @ANY USB dongle and the @ANY BRICK.

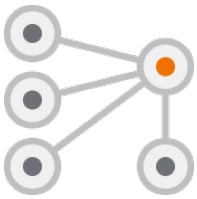


Figure 15 Star - Network

Configuration of the @ANY USB Dongle

The participants should automatically connect to the coordinator. To do this, the coordinator must set up the network when starting. The S-Register S0 must be set to 1 for this purpose.

Configuration of the @ANY BRICKs

The @ANY BRICKs should send the temperature values of the LM73 to the coordinator every 10 seconds or at the push of the button S2. The transmission is signalled by the blinking LED 2.

To register the temperature on the @ANY BRICK of the LM73, the S-Register 238 must be toggled with a 1. The temperature values obtained are stored in the S-Register 245 as a decimal number. A timer interrupt is required to send the temperature every 10 seconds. This is made possible via the S-Register 18. The desired seconds are entered as a decimal number. If you want to achieve smaller intervals, such as 1 millisecond, the S-Register 17 must also be used as a divider register, see SMS Command Reference chapter S-Register.

Triggering the S238 register and sending the content from register 245 should be automated. This is done via the Shadow Command Register (SCR). It is possible to store entire command lines in the SCR. The SCR responds to interrupts and then executes its saved command line. A command line is written with the command AT*, which replaces the normal AT. The command line can be saved in profiles.

First, the complete chain of commands is written to the SCR:

```
Complete command:
AT*AS234=2S238=1DSA245XS234=2
Response:
OK
```

AT*	Writes the following command line to the SCR
A	Connects to the network if it has not already done so
S234=2	Toggles LED2 (Information on the S-Register below).
S238=1	The LM73 measures the temperature once and stores it in the S-Register 245
DS	Sends the following S-Register to the coordinator
A	as ASCII
245	The S-Register which is sent
X	Sending is cancelled here and the following command line is not sent
S234=2	Toggles LED 2

So that the button can execute the SCR, the 4 must be written in the S-Register 236 (information on the S-Register below). Sending should also be signalled here by LED 2 flashing.

First, write the 4 in the S-Register 236. Next, the timer in the S-Register 18 is set to 10 seconds. The configuration is saved in profile 0 and the @ANY BRICK is restarted.

Complete command:
ATS236=4S18=10&W0Z
Response:
OK & Device restarts

The @ANY USB dongle is connected via a USB port. It can be addressed via a command line tool and responds to an AT with OK. By entering the ATA command, the @ANY USB dongle establishes a network.

The battery holders can be plugged into the @ANY BRICKs, the + 1.5V AAA batteries inserted and the TAOGLAS antennas connected to the respective @ANY modules. To start the @ANY BRICKs, flip S3 switch. LED 1 green lights up.

After the first 10 seconds, an @ANY BRICK reports to the coordinator with: + ASSOCIATED: 00*X* and after another 10 seconds, the first temperature message appears with: + DATA: 3 BYTES FROM 000*X* (0*XX*) 0*XX*. When the temperature is sent, LED 2 flashes briefly in yellow. By pressing the S2 button, a direct temperature report is sent to the coordinator.

You can restart an @ANY BRICK by pressing the S1 button, whereupon the @ANY BRICK reconnects to the network and sends temperatures to the coordinator.

Congratulations, you have just configured and commissioned your second network!

About S-Register 234, 236

S-Register 234:

Bit	7	6	5	4	3	2	1	0
	-	-	-	-	LED all	LED red	LED yellow	LED green
GPIO						2	1	0

S-Register 236:

Bit	7-4	3	2	1	0
	-	Interruptpin 7	Interruptpin 6 (Taster S2)	Interruptpin 7	Interruptpin 6 (Taster S2)
	-	Enable command execution		Enable Wake up	

You can also display the WSN you just set up using the SMS monitor. To do this, open the A.N. Solutions ToolChain and navigate to the "SMS Monitor" tab. Disconnect the command line tool and your coordinator. You can now connect to the coordinator in the SMS monitor. To do this, use the "Connection Parameters" window. After a few seconds, the @ANY BRICKs appear in the SMS monitor. See chapter 2

4.7 Mesh Network for SMS Pro User

4.7.1 Setting up the Network Participants

This chapter builds on the entire Getting Started and requires knowledge of how to set up a network with our Smart Mac Suite (SMS). Mesh networks can only be realized with the SMS Pro. All participants in this network must be assigned the same channel and the same PANID. If only the commands listed here are used, the mesh network will not work. In this case, read chapters 3.3.1 and 3.4.1 for full functionality.

For applications where the above approaches are not suitable, a complete routing layer based on the IEEE 802.15.4 standard is integrated in the SMS Pro. It offers the opportunity to build self-learning and self-healing mesh networks.

The routing algorithm itself is taken from the LwMesh stack. A "native routing" mechanism with multicast support was selected from the various options and implemented with the standard parameter settings of LwMesh. For details, see the LwMesh stack and documentation.

Our existing star network must be dissolved. All devices are connected to the PC in order to set them up using the command line tool. First, the timer in the S-Register 18 is set to 0 for all @ANY BRICKs and the connection to the coordinator is released with ATH.

End-Device:	Coordinator:
ATS18=0 ↵	
OK	
ATH ↵	+DISASSOCIATED: 000*X*
OK	

In order to enable dynamic routing, the S-Register 223 must be written with the 5. When using a routing layer, all devices in the network must have the same S223 settings. With this routing method, a statically predefined short address is required. In addition, all devices must be assigned the role of the FF device. A combination of coordinator and FF device is also possible. However, only a pure FF device mesh is described here.

The following settings are made for all devices, @ANY USB dongle and @ANY BRICKs. Make sure that each device gets a different short address!

Command:	Response:	Explanation:
ATS223=5 ↵	OK	Activate the dynamic routing function
ATS220=1 ↵	OK	Set the device roll to FF-Device
ATS200=000*X* ↵	OK	Set the Short Address, different for each device
AT&W0 ↵	OK	Save the settings in profile 0
ATZ ↵	OK	Reboot the device
combinate:	*ATS223=5S220=1S200=0001&W0Z*	

To test the settings, a connection is established between the @ANY USB dongle and an @ANY BRICK with the command ATA. There is no + DEVICE ASSOCIATED: 000X (80) message.

@ANY USB dongle *Short Address: 0000*	@ANY BRICK *Short Address: 0001*
ATA ↵	ATA ↵
OK	OK
	ATD0000; ↵
	TEST1 ↵
	ID 002: 5 BYTES to 0000
+DATA: 5 BYTES FROM 0002 (011)	OK
TEST1	+SENT: ID 002
ATD0001; ↵	
TEST2 ↵	
ID 002: 5 BYTES TO0001	
OK	+DATA: 5 BYTES FROM 0002 (011)
+SENT: ID 002	TEST2

If the transfer was successful, the most important steps have been taken for a mesh network. If the transmission is interrupted with an error message, check whether all devices have the same PANID, the same channel and a different short address. The ATA command must also be executed beforehand.

In order to better understand the dynamic routing, one @ANY BRICK will act as nodes and the second one will send temperature data to the @ANY USB dongle permanently.

The one @ANY BRICK which should act as a node, require further settings. S-Register 0 must be set to 1. The automatic connection to the configured network is thus active. After setting the S-Register, everything is saved in profile 0 with AT & W0.

```
Complete command:
ATS0=1&W0↵
response:
OK
```

With the @ANY BRICK which acts as a data logger uses the shadow register again, which is triggered with a 3 second timer. The LM73 is triggered once via the S-Register 238 and stores the temperature in the S-Register 245. With @ANY USB dongle this should be output as ASCII.

```
Complete command:
AT*AS238=1D0000SA245 ↵
response:
OK
```

AT*	Writes the following command line to the SCR
A	Connects to the network if it has not already done so
S238=1	The LM73 measures the temperature once and stores it in the S-Register 245
D0000	Send to address 0000 (@ANY USB dongle)
SA	The following S-Register, which is to be sent, as ASCI
245	The S-Register which is sent

Now the timer in the S-Register 18 is set to 3 seconds and the whole is saved in profile 0.

```
Complete command:
ATS18=3&W0↵
response:
OK
```

4.7.2 Setting Up and Commissioning the Test Environment

First, all @ANY BRICKs are disconnected from the PC. It makes sense to mark the second BRICK as a permanent transmitter in order to distinguish it from other BRICKs. The @ANY USB dongle remains plugged into the PC and is accessible via a command line tool. The BRICKs are powered by 1.5 V AAA batteries.

The @ANY USB dongle is connected via a command line tool and responds to an AT with OK. The BRICK, which transmits permanently, is switched on via switch S3 on the right side and placed next to the @ANY USB dongle. The green LED of the @ANY BRICK lights up green (standard firmware settings). The other @ANY BRICK remain switched off.

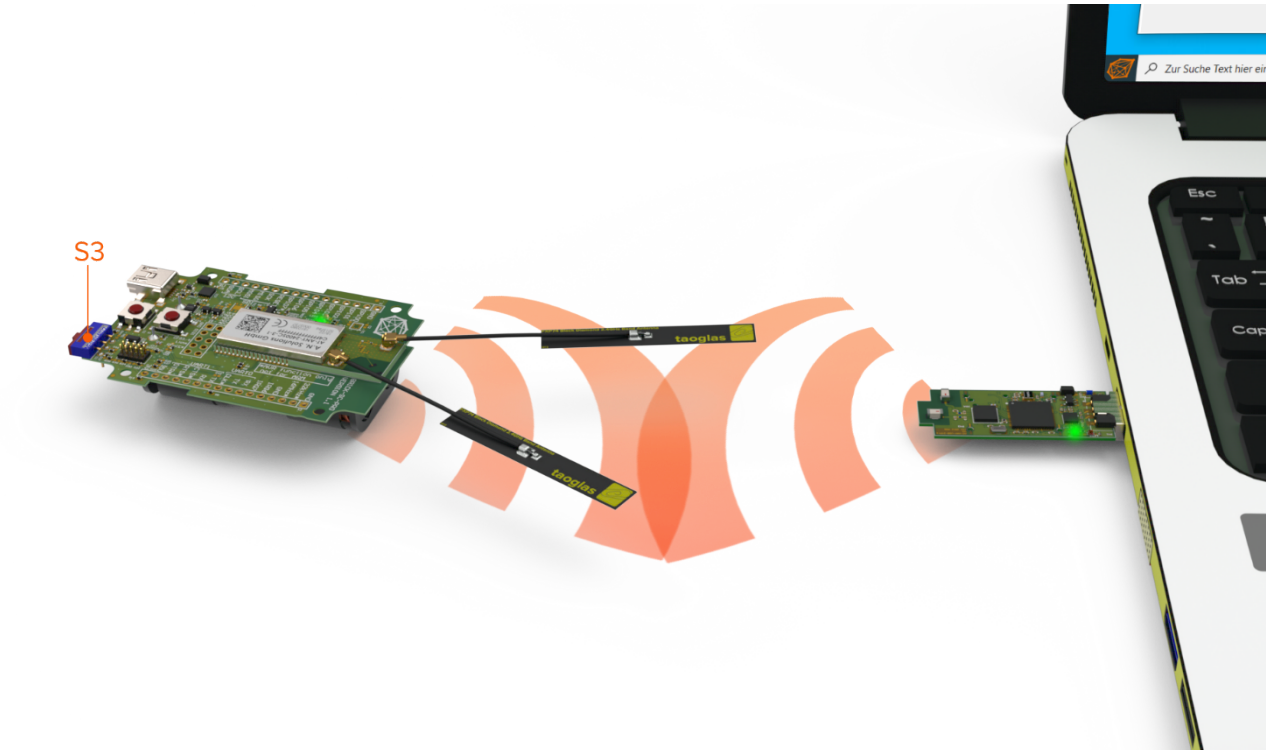


*Figure 16 Mesh peer to peer***

** distances shown do not correspond to reality

The @ANY USB dongle receives temperature values from the @ANY BRICK. Now the @ANY BRICK is removed from the @ANY USB dongle until the @ANY USB dongle no longer receives temperature values. If so, an @ANY BRICK is added as a node. Take the @ANY BRICK, switch it on using switch S3 and place it between the @ANY USB dongle and the @ANY BRICK, which transmits permanently.

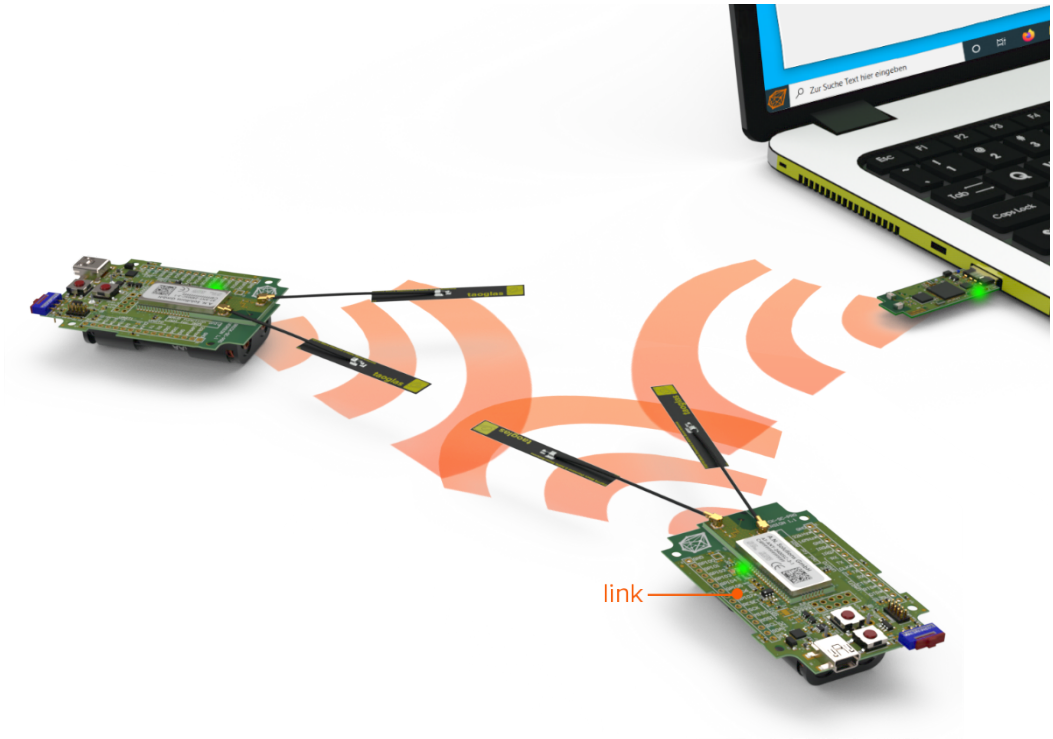


*Figure 17 Dynamic Mesh***

** distances shown do not correspond to reality

After a few seconds, the @ANY USB dongle should receive data from the @ANY BRICK again. This is no longer the direct route, but via the node which is referred to as "link" in the image above. Many different network arrangements are possible. It is also possible that all @ANY BRICKs and the @ANY USB dongle are data loggers and nodes. If one of the node fails, a new route to the destination is automatically searched for and noted. In the next chapter, the third BRICK is used with the ADC function.

Congratulations, you have just set up and started using your first mesh network!

4.8 ADC Support for SMS Pro User

The use of the Analog Digital Converter is described here only for the ATmega128RFA1 and is only possible with the Smart Mac Suite (SMS) Pro. For more detailed information than listed here, please refer to the data sheet of the ATmega128RFA1 [5]. This chapter builds on chapter 3.6. If you work through the example described here on your own, this may not lead to success. If this is the case, read chapters 3.3.1, 3.4.1 and 3.6 for full functionality.

4.8.1 Basic Settings of the ADCs with the SMS Pro

The last @ANY BRICK is to be integrated into the existing mesh network. It is given the task of forwarding data from the ADC to the @ANY USB dongle, which can be evaluated there with the help of a user interface.

The ATmega128RFA1 has a 10 Bit ADC with a Vref of 1.8 V. The Vref and the prescaler are specified by the SMS and cannot be changed. The range that can be displayed is between 0x0000 and 0x03FF. The ADC is triggered via the S-Register 238 and takes exactly one measurement. The ADC values obtained are stored in the S-Register 240 - 247 depending on the input used. The ADC has four input pins ADC0 to ADC3.

Information on S-Register 240 to 247 when using S-Register 238=3:

S-Register	247	246	245	244	243	242	241	240
	ADCL	ADCH	ADCL	ADCH	ADCL	ADCH	ADCL	ADCH
ADC Pin	3	3	2	2	1	1	0	0

ADC pin 0 is not suitable for direct ADC measurement because, pin 0 is followed by a voltage divider for Vcc measurement. A small example in the next chapter should clarify this.

If you want to use the ADC to measure external voltages, you need pins 1 to 3. But note that only voltages from 0 V to 1.8 V (0x0000 to 0x03FF) can be mapped. The measurement of these ADC pins is done in the same way as described in the next chapter , except that the S-Registers 240 and 241 are not output here, but only the S-Registers which are used for the respective ADC pin are needed. (See "Information on S-Register 240 to 247 when using S-Register 238 = 3")

Pin stripe X3 on the @ANY BRICK:

PD7	PD6	PG0	PG1	GND
ADC3	ADC2	ADC1	ADC0	Vref

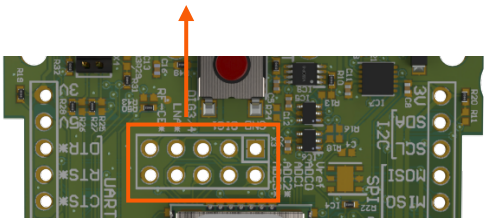
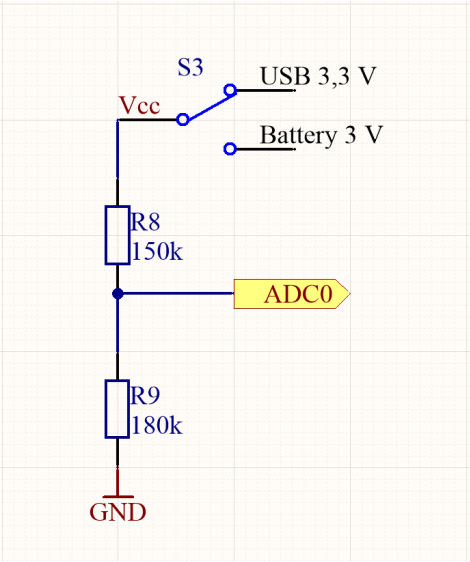


Figure 18 Section @ANY BRICK X3

4.8.2 Example of Battery Measurement with the ADC

Circuit:



The ADC0 pin is connected to switch S3 via a voltage divider of 180 KΩ and 150 KΩ. The Vcc can be set using switch S3. It is possible to supply the @ANY BRICK with a USB connection or with batteries. The battery supply is 3 V. The USB supply is reduced from 5 V to 3.3 V.

To determine the battery voltage, the S3 is switched to battery supply (switch position on the right). The ADC is triggered with the S-Register 238 = 3 and this stores the raw values in the S-Register 240-247, which can then be calculated back to the voltage.

Figure 19 Circuit ADC0

Because access to the SMS via command line tool is only possible via USB, the @ANY BRICK is configured so that it sends the read value to our mesh network every few seconds. The @ANY BRICK can thus be powered via the batteries.

The @ANY BRICK from the previous chapter which has no function yet, is connected to USB on the PC and addressed via a command line tool. In this case, it is the @ANY BRICK with the short address 0003. Make sure that the @ANY BRICK automatically dials into the mesh network (S-Register 0 = 1). This can be checked by briefly sending data packets with the ATD command.

The Shadow Command Register (SCR) is used to send the ADC values to the @ANY USB dongle. The following is written in the SCR:

```
Complete command:
AT*AS238=3D0000S240S241↵
Response:
OK
```

AT*	Writes the following command line to the SCR
A	Connects to the network if it has not already done so
S238=3	The ADC measures once and stores this in the S-Register 240 to 247
D0000	Sends to address 0000 (@ANY USB dongle)
S240	Sends the S-Register 240 ADCH of ADC pin 0
S241	Sends the S-Register 241 ADCL of ADC Pin 0

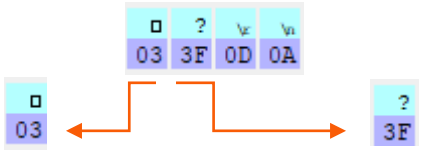
The timer interrupt is set to 1 second and everything is saved.

```
Complete command:
ATS18=1&W0↵
response:
OK
```

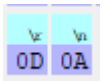
The @ANY USB dongle should now receive values. To process these values better, the command line tool must be switched to HEX output. The batteries are inserted and S3 switch is switched to the right position. The green

LED lights up (standard firmware settings). After a few seconds, the @ANY USB dongle receives raw ADC data again.

The incoming ADC raw data look as follows (depending on the command line tool):



Command line Tool depended:



ADCH of ADC Pin 0 ADCL of ADC Pin 0 Ausgabe des command line Tools

This results in a decimal number of 831. This can be calculated back into a voltage using the formula below. The hexadecimal number given above gives a voltage of around 2.7 V.

$$Vcc = \frac{ADCWert * Vref * (R8 + \mathrm{R9})}{ADCmaxWert * R9}$$

The whole thing can be checked by switching the Vcc to USB (S3 switch position on the left) and calculating the ADC value obtained again using the formula from above. A voltage of around 3.3 V should be calculated.

Congratulations, your ADC is working!

5 Firmware Flash / Update

5.1 Structure of the @ANY Hardware

The @ANY modules are available in different versions, either as separate modules or soldered onto a finished development board. The @ANY modules mainly consist of two parts, the microcontroller and a wireless transceiver. Several interfaces provided by the microcontroller are available on the module pins.

On the microcontroller side, depending on the module, the Atmel ATmega1281, the Atmel ATmega128RFA1 or the SAMR30G18 is used. These microcontrollers are equipped with a user-replaceable software part, the so-called firmware, which implements the desired functionality of the device. The software can be programmed into the microcontroller in two different ways, which are described below.

@ANY900 Modules:

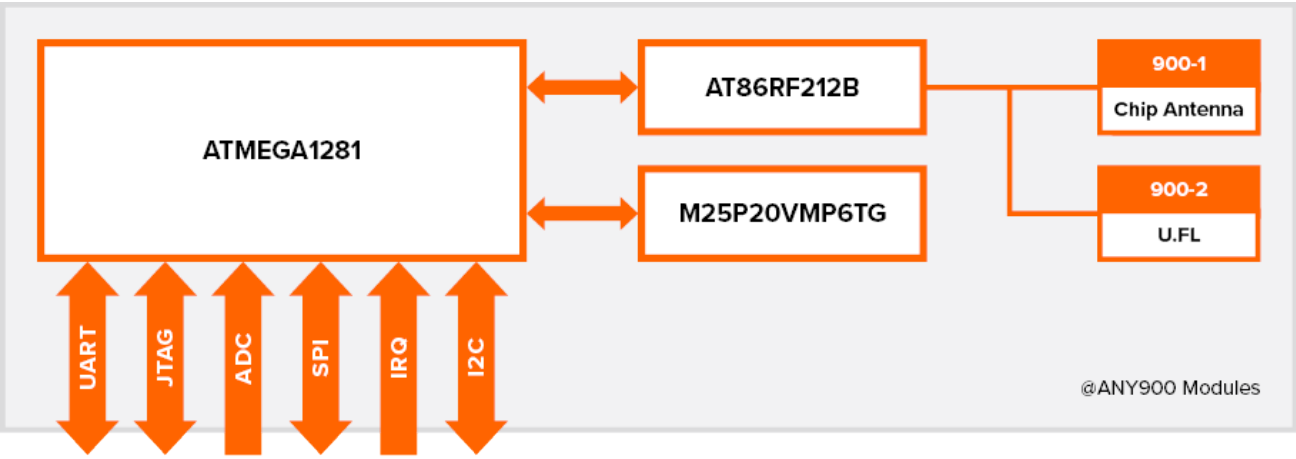


Figure 20 @ANY900 Modules

@ANY900ARM-SC Modules:

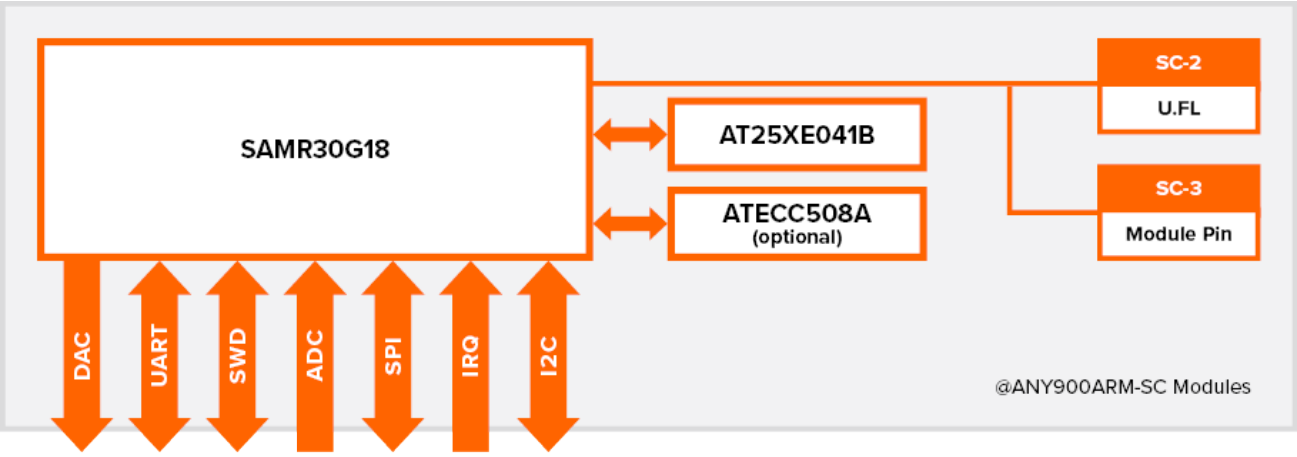


Figure 21 @ANY900ARM-SC Modules

@ANY2400-SC Modules:

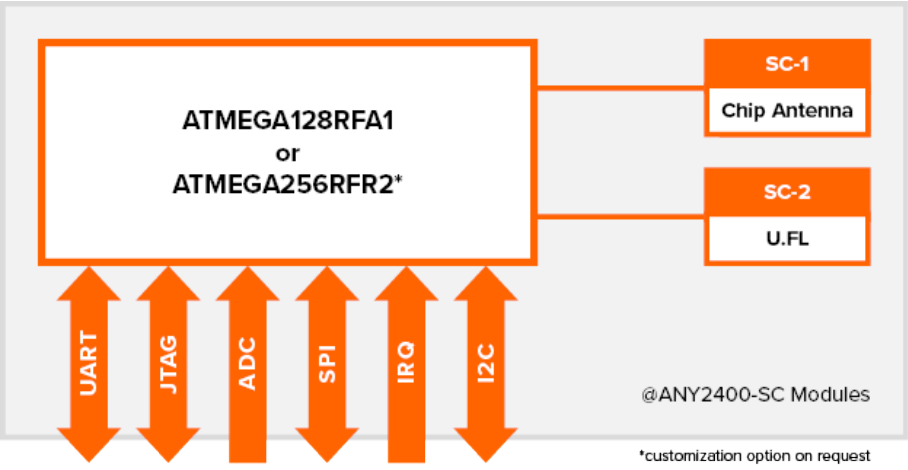


Figure 22 @ANY2400-SC Modules

@ANY2400-SC-3 Modules:

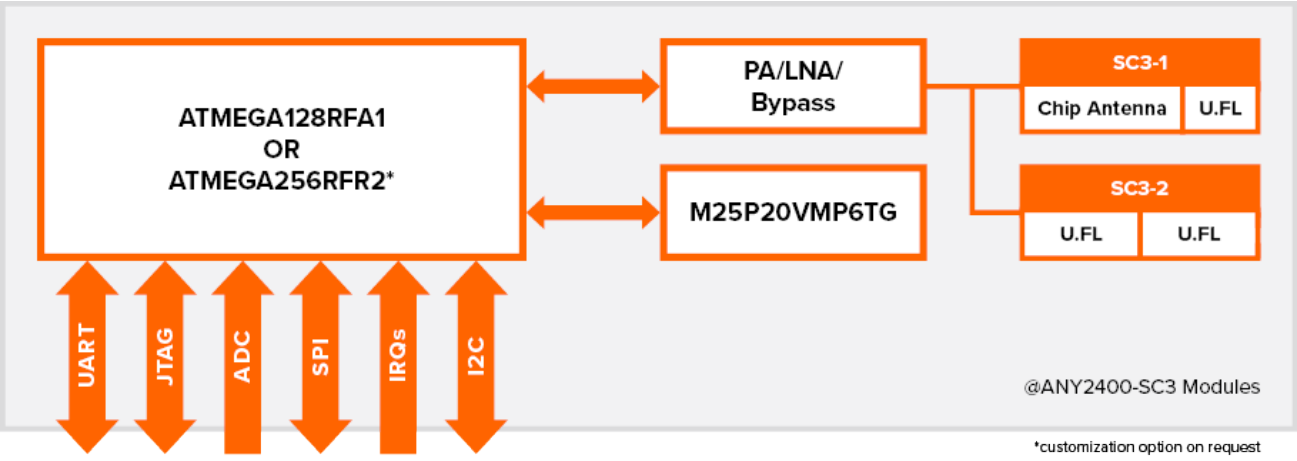


Figure 23 @ANY2400-SC-3 Modules

5.2 JTAG / SWD

A common way of programming a microcontroller is to use a provided JTAG / SWD interface. All devices mentioned in this document are equipped with such an interface. In order to use the JTAG / SWD interface, an additional programming hardware interface is required. The selection of possible hardware is large and ranges from self-made devices to professional hardware, which supports functions far beyond device programming. An example of a professional solution is the Atmel ICE (Figure 24) which can be purchased from Microchip.

Figure 24 Atmel ICE Kit

With the Atmel ICE it is possible to program all our devices that have a 50mil 10Pin JTAG / SWD connector. To program the @ANY USB dongle, an adapter is required (see chapter 1 Overview). The use of the JTAG / SWD interface offers the greatest possible flexibility and is recommended for power users and embedded developers, although additional costs may be required. A more cost-effective approach is presented in the next subsection.

5.3 Bootloader

Another option to update the firmware of the @ ANY module is to use a bootloader. Two requirements must be met for its use. First, a code with bootloader functionality must be in a specific area of the microcontroller's flash memory. Secondly, the application software must offer a way to jump into the bootloader code, or the fuse bite of the microcontroller must be set so that the bootloader code is executed each time it is switched on. The freely available Smart MAC Suit (SMS) software from A.N. Solutions contains the required bootloader code and provides a way to jump into the code from the application by sending a specific set of AT commands. The SMS software with bootloader function is preprogramed in the hardware components of the development kit.

For user-defined solutions, reference must be made to the @ANY BRICK data sheet, which contains the pin assignment and connection assignment.

5.4 Firmware Flash / Update with the SMS Updater

The SMS Updater is a free tool from A.N. Solutions, which makes it possible to upgrade or flash the firmware of a @ANY module with pre-installed bootloader SMS. It enables flashing and updating of the SMS without JTAG/SWD interface, only via USB. A prerequisite for this is the correct detection of the @ANY modules on the PC used (see chapter 2).

The @ANY USB dongle or the @ANY BRICK is connected to the PC via USB. A new COM port appears in the device manager. The ATANY updater opens. Make sure that the tool is up to date. More information on this is available on our website.

In the Interface area, the COM is selected which has been assigned to the @ANY USB dongle or @ANY BRICK. If this does not appear, the list can be updated with the "**Refresh port list**" button. If the COM still does not appear, go to chapter 2. The baud must be set to 38400bd and the field "**AT-ANY module type**" must be left auto. Now the desired firmware can be selected in the "**Flash file**" area by opening it. It is important that the firmware to be flashed does not have a bootloader area that is identified without _BL in the name. If a firmware is flashed with a bootloader,

it will overwrite the existing bootloader. If the current bootloader has been overwritten, the firmware can only be flashed back as described in chapter 5.6 or 5.7.

If the correct firmware has been selected, the flash process can be started with "**Start Update**". It runs through the middle part of the ATANY Updater and can take a few seconds. The flashing process is successfully completed with "**programming done**". The sequence is shown in the next Figure.

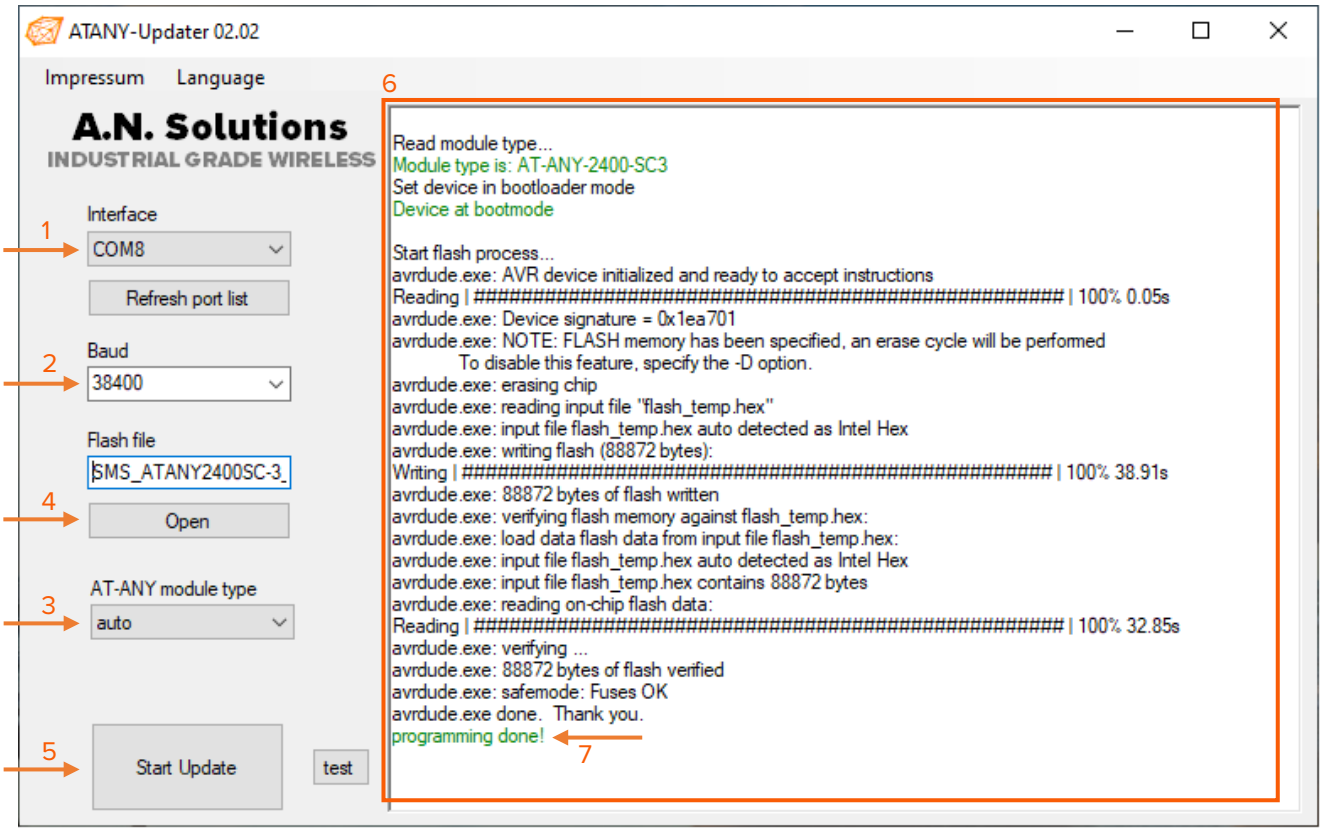


Figure 25 ATANY-Updater

5.5 Firmware Flash / Update with Avrdude without JTAG / SWD

Avrdude is free software which enables program code to be written into the flash memory of the Atmel AVR controller using command line parameters.

To flash the @ANY USB dongle or @ANY BRICK, they must be connected to a free USB port. The device manager initializes the devices and assigns them a COM. If this is not the case, go to chapter 2.

In order to flash the firmware via avrdude, there must be a bootloader on the @ANY USB dongle and @ANY BRICK. Two consoles are required for flashing. First, a connection to the @ANY USB dongle or @ANY BRICK is established using a command line tool. The existing connection is checked by an AT and the @ANY USB dongle or the @ANY BRICK responds with OK. The command line tool connection remains.

Now another console is opened. In Windows this is done by pressing the key combination "**Windows logo key + R**". An input field opens in which "**cmd**" is entered and confirmed with ENTER. The Windows console is now open. With the help of "**cd**" you navigate to the folder in which the "**avrdude.exe**" is located. By entering **"avrdude.exe"** avrdude is executed. The file must remain in the folder.To flash the firmware, the @ANY USB dongle or @ANY BRICK must jump into the bootloader. This is implemented with the AT+U command. Once this has been done, 20 seconds remain to execute the required command line parameters in avrdude. When the 20 seconds have passed, the @ANY USB dongle or @ANY BRICK jumps back to the normal program code. It makes sense to lay out the required command line in the avrdude. The process is explained below.

Command line Tool	
AT ↵	Connection check
OK	
ATS18=0 ↵	Set timer S-Register to 0
OK	
ATH1 ↵	Disconnect all existing connections
OK	
ATS94=224 ↵	Unlock bootloader in S-Register 94
OK	
AT+U ↵ → 20 sec timer is running	jump into the bootloader
Disconnect Tool connection!	

Avrdude.exe in the Windows console	
avrdude -p m128RFA1 -c stk500v1 -P\\.\COM8 -b 38400 -U fl:w:SMS_ATANY2400-3_131b_0BB9.hex ↵	
-p	Part number / Required. Specify AVR device.
-c	Programmer / Specify programmer type.
-P	Port / Specify connection port.
-b	Baud rate / Override RS-232 baud rate.
-U	<mem. type>:write:<filename> / Memory operation specification. Multiple -U options are allowed, each request is performed in the order specified.

The command line listed above should be set up in avrdude before executing the AT+U command so that it can be executed within 20 seconds. After triggering the command line in avrdude, it writes the program code into the flash-memory of the @ANY USB dongle or the @ANY BRICK. The flashing is done with "**avrdude done. Thank you Successfully completed**".

So, it is possible to keep the firmware on the @ANY USB dongle and on the @ANY BRICK up to date without any additional hardware. This is made much easier using the ATANY Updater as described in Chapter 4.4.

5.6 Firmware Flash / Update with Avrdude & JTAG / SWD

If an Atmel ICE kit or similar programming hardware is available, it can be used with avrdude. However, it must be noted that the Atmel ICE is only supported by version 6.3 of avrdude. In addition, for flashing the @ANY USB dongle the Special JTAG Connector is required because it does not have a 10 pin 50 mil connector.

With this variant there is no need for a bootloader on the @ANY USB dongle or the @ANY BRICK.

The @ANY BRICK and the @ANY USB dongle are connected to the Atmel ICE and the PC in two different ways.

The @ANY BRICK is plugged into a free USB port on the PC via USB. This is for the power supply. Alternatively, the @ANY BRICK can also be powered by two 1.5 V AAA batteries. The Atmel ICE Kit is also plugged into a free USB port on the PC via USB. The included 10 pin 50 mil connector is plugged into the Atmel ICE on the AVR side or SAM side for SAMR30G18 Modules and on the @ANY BRICK onto the existing 10 pin 50 mil connector. Make sure that the connector is aligned correctly!

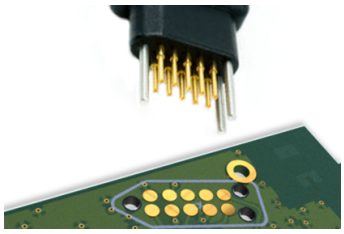


Figure 26 Tag Connect

With @ANY USB dongle, the connection is quite cumbersome. The @ANY USB dongle must be connected to the PC via USB for power supply. It is advantageous to use USB extension, as Tag Connect (Figure 26) is located on the back of @ANY USB dongle. Next, the Tag Connect cable is connected to the Tag Connect on the back of the dongle. Make sure that the Tag Connect is always in contact with the cable. Depending on the used DevKit the other end of the Tag Connect cable must be connected to the AVR slot or for ARM to the SAM slot of the Atmel ICE.

Now a console is opened. In Windows this is done by pressing the key combination "**Windows logo key + R**". An input field opens in which "**cmd**" is entered and confirmed with ENTER. The Windows console is now open. With the help of "**cd**" you navigate to the folder in which the "**avrdude.exe**" is located. By entering **"avrdude.exe"** avrdude is executed. It must remain in the folder.

The firmware is not flashed as described in chapter 5.5, because no bootloader is required here. However, the programmer and the port must be changed in the avrdude. The flashing of the dongle and the BRICK are the same, only the connection types differ.

The following command line is executed using avrdude.

Avrdude.exe in the Windows console	
avrdude -p m128RFA1 -c atmelice -P usb -b 38400 -U fl:w:SMS_ATANY2400-3_131b_0BB9.hex ↵	
-p	Part number / Required. Specify AVR device.
-c	Programmer / Specify programmer type.
-P	Port / Specify connection port.
-b	Baud rate / Override RS-232 baud rate.
-U	<mem. type>:write:<filename> / Memory operation specification. Multiple -U options are allowed, each request is performed in the order specified.

After triggering the command line in avrdude, he writes the program code into the flash memory of the @ANY USB dongle or the @ANY BRICK. The flashing is done with "**avrdude done. Thank you. Successfully completed"**. With the Atmel ICE, the flashing is much faster.

When using this method, the entire flash memory is overwritten and it is no longer possible to work with the bootloader later.

5.7 Firmware Flash / Update with JTAG / SWD & Atmel Studio

Another way to flash or update the firmware is by using the Microchip Atmel Studio. This also requires the Atmel ICE Kit and the Special JTAG Connector for the @ANY USB dongle.

With this variant there is no need for a bootloader on the @ANY USB dongle or the @ANY BRICK

The @ANY BRICK is plugged into a free USB port on the PC via USB. This is for the power supply. Alternatively, the @ANY BRICK can also be powered by two 1.5 V AAA batteries. The Atmel ICE Kit is also plugged into a free USB port on the PC via USB. The included 10 pin 50 mil connector is plugged into the Atmel ICE on the AVR side and on the @ANY BRICK onto the existing 10 pin 50 mil connector. Make sure that the connector is aligned correctly!

With @ANY USB dongle, the connection is quite cumbersome. The @ANY USB dongle must be connected to the PC via USB for power supply. It is advantageous to use USB extension, as Tag Connect (Figure 26) is located on the back of @ANY USB dongle. Next, the Tag Connect cable is connected to the Tag Connect on the back of the dongle. Make sure that the Tag Connect is always in contact with the cable. Depending on the used DevKit the other end of the Tag Connect cable must be connected to the AVR slot or for ARM to the SAM slot of the Atmel ICE.

The Atmel Studio is downloaded and installed from Microchip. After the start-up, open a new device programming window by pressing the device programming button (Crtl+Shift+P). The head of this window must look like this:

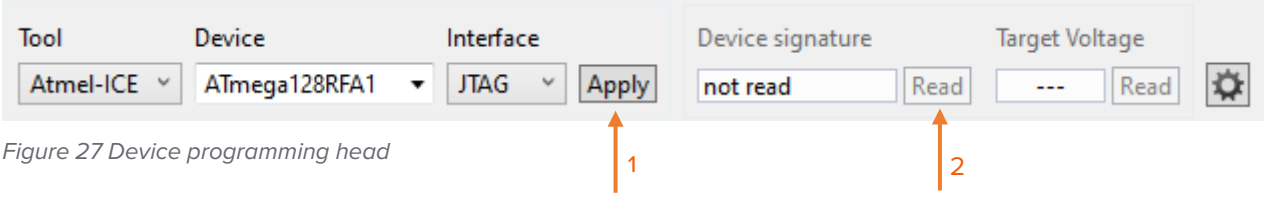


Figure 27 Device programming head

The settings are confirmed with Apply. By pressing the "**Read**" button, Atmel Studio reads the "**Device signature**" and the connection to the device has been checked. On the left side of the window, you switch to the "**Memories**" tab. Firmware can be loaded in the Flash area by pressing the button with the three dots. The firmware is written to the flash memory with the "**Program**" button.

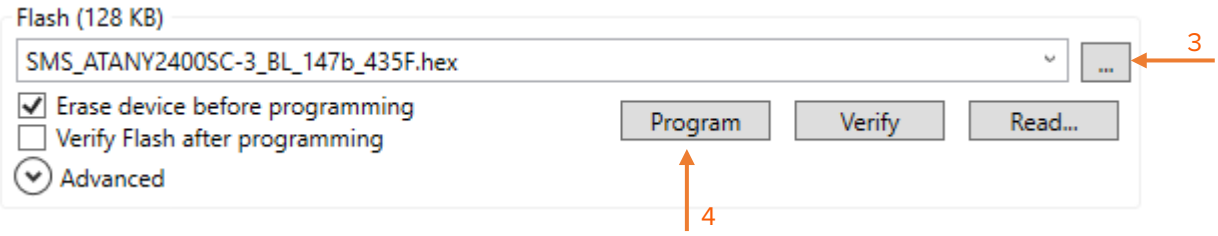


Figure 28 Memories - Flash

The successful flashing is returned with a "Programming Flash ... Ok".

When using this method, the entire flash memory is overwritten and it is no longer possible to work with the bootloader later.

6 Software Tools

6.1 Sniffer Tool

In the development phase of a project, it can be very useful to analyse and track every data packet sent. With the help of the sniffer firmware, it is possible to carry out precise network analyses and targeted debugging.

A powerful device with very good antennas should be selected as a sniffer. In this example, a peer-to-peer network is observed between a @ANY USB dongle and a @ANY BRICK. A second @ANY BRICK is used as a sniffer.

The first step is to set up a functional peer-to-peer network between the @ANY USB dongle and one of the two @ANY BRICKs. In this network it must be possible to exchange data using the ATD command. See chapter 4 "**Getting Started**".

Next, the other @ANY BRICK with the HalCtrl*XXXX*_v*XX*.hex firmware is flashed using one of the methods described in Chapter 5. When selecting the firmware, ensure compatibility with the individual hardware, which is specified in the name of the firmware after "HalCtrl".

With a suitable command line tool, a connection is established with the @ANY BRICK, which contains the sniffer firmware. Please note that the baud rate is changed to 57600. (COM*X* 57600bd 8 N 1)

With the command AT? a command overview can be called up. The overview is not as extensive as with the SMS. Next, it is changed to the channel of the peer to peer network with the command AT+C =*XX*. By entering the ATA command, the @ANY BRICK starts sniffing.

```
AT?
HalCtrl2400SC commands:
--------------------
AT? ............. this
ATB ............. jump to bootloader at 0x1F000
ATZ ............. reset Transceiver
ATA ............. enable RX
ATH ............. disable RX
AT+A ............ enable Transceiver
AT+H1 ........... disable Transceiver
AT+H0 ........... MCU sleep
ATS<r>(?/=<v>) .. read/write Transceiver register
ATD=<data> ...... send ascii
ATDX ............ endless TX (toggle)
AT+C(?/=<ch>) ... read/write channel
ATL<n>=(0/1) .... LED ctrl
AT+F(?/=<f>) .... data output format (RB/RH/FB/FH)
received frame format: +timestamp: ctl sq data crc (rssi)
OK

AT+C=20
OK

ATA
RX enabled
```

Figure 29 Sniffer config

By sending via ATD command, in the peer to peer network, the sniffer listens to the "conversation" and receives various information. It can look as follows:

Coordinator:	End-device
	ATD;↵ HELLO↵ ID 001: 5 BYTES to 0000 OK
+ DATA: 5 BYTES FROM 0001 (074) HELLO	+ SENT: ID 001

Sniffer:
+003A.DB47: 8861 9C AB 12 00 00 01 00 48 61 6C 6C 6F 0984 (0A) +003A.DB4C: 0002 9C EB5D (0B)

The information breaks down as follows:

Transmitter:	
+003A.DB47:	The time from when it was sent
8861	Configuration bits of the IEEE 802.15.4 standard can be broken down there
9C	Sequence number for assignment for possible answers
AB 12	PANID (Little-Endian-Format*)
00 00	receiver (Little-Endian-Format*)
01 00	transmitter (Little-Endian-Format*)
48 61 6C 6C 6F	Data sent Hello
0984	CRC checksum (! = Error in the checksum, transmission failed)
(0A)	Signal strength indicator (RSSI) **
Reciver:	
+003A.DB4C:	The time from when it was received
0002	Configuration bits as acknowledgment, sent as broadcast
9C	Sequence number for assignment at the sender
EB5D	CRC checksum (! = Error in the checksum, transmission failed)
(0B)	Signal strength indicator (RSSI) **

* The Little Endian Format is a format for the transmission or storage of binary data, in which the least significant byte (LSB) comes first and is stored in the lowest memory address

** Indicator SMS ED=3*(RSSI-1)

A sensible application example is the measurement of the packet error rate (PER) on the level of the hardware abstraction layer (HAL). In contrast to the software-based Smart MAC Suite applications, the HAL-based approach does not contain CSMA-Ca, acknowledgements and retransmissions for a more reliable connection. This makes this application ideal for distance measurements and robustness tests under different conditions. The application supports point-to-point transmission, with one device acting as a transmitter and the other as a receiver. Continuous packet transmission can be started and stopped with the ATDX command. Reception can be started with the ATA command and is automatically stopped after a defined time without packet reception or after the ATH command. The firmware with the file name "**autoTX**" automatically executes the command ATDX after switching on and thus sends packets permanently. They are therefore well suited for autonomous senders without a UART interface, e.g. some BRICK boards.

Two @ANY BRICKs are used for the test setup. The first is flashed with the firmware HalCtrl*XXXX*_PER_test.hex, as described in Chapter 5. The second is flashed with the firmware HalCtrl*XXXX*_PER_test_autoTX.hex. It makes sense to power the second BRICK with 1.5V AAA batteries for increased mobility. A connection is established via a command line tool with the first @ANY BRICK and the second one is switched on. The second @ANY BRICK starts transmitting immediately and the first receives the data.

6.2 Transparent UART – SMS PRO User

This chapter refers to procedures from Chapter 5, Flashing / Updating Firmware. Advanced knowledge is assumed. These procedures serve as a selection of possibilities and cannot be mapped 1 to 1 to the following. In addition, additional hardware and software may be required.

The transparent UART allows you to exchange data between two devices without much effort. This creates a simple bidirectional point-to-point data channel using our AT-ANY modules. The software has been specially designed to facilitate the establishment of a point-to-point connection between two wireless modules. The following example illustrates this.

Currently the following AVR modules are supported:

- AT-ANY 900-1
- AT-ANY 900-2
- AT-ANY 2400-SC-1
- AT-ANY 2400-SC-2
- AT-ANY 2400-SC-3-1
- AT-ANY 2400-SC-3-2

First of all, the tuned firmware is required. After purchasing @ANY Smart Mac Suite (SMS) Pro, you have received a software package from us (https://an-solutions.de/de/any-sms-pro.html). Next to SMS Pro, there is a folder "Transparent UART", which contains all important files for the transparent UART. Unzip this folder at a place of your choice on your PC.

In this folder, there are several different HEX files. Files beginning with "eeprom" are for the setup of the EEPROM and its settings. Files starting with "transparent_uart" are tuned firmware packages for the AT-ANY modules. The following designations refer to the BRICKs and Dongles used.

Select two of your existing devices to be used with the transparent UART.

In this example, two AT-ANY 2400-SC-3-2 BRICKs are used. The setup and procedure are identical on each A.N. Solutions device. Atmel Studio 7.0 from Microchip with the Atmel ICE (page 29, Figure 24) as programming interface was used to flash the firmware and the EEPROM.

Based on your selection, the correct EEPROM hex files are required. Make sure that the name of the hex files matches the name of your selected devices. Two files are required for setting up the EEPROM, "... 1 to 2 ..." and "... 2 to 1 ...".

The files needed here for the example are the following: *AT-ANY 2400-SC-3-2 BRICKs*

- eeprom_2400_ff_0001_to_0002_*XX*_*YY*.hex*
- eeprom_2400_ff_0002_to_0001_*XX*_*YY*.hex*

** XX – Version, YY – Channel*

The selection of the firmware is similar. Pay attention to the designation "transparent_uart" and the used device type.

The firmware required in this example is the following: *AT-ANY 2400-SC-3-2 BRICKs*

- transparent_uart_2400sc-3.hex

Next, the devices are flashed using one of the procedures listed in chapter 5. The procedures listed there serve as a selection of possibilities and cannot be mapped 1 to 1 to the following.

Proceed as follows:

1. connect one of the two devices to the PC via mini USB (chapter 2)
2. 2.4 GHz - change the fuses to hfuse: 0x19, lfuse: 0xde and efuse: 0xfe
   Sub-1 GHz - change the fuses to hfuse: 0x19, lfuse: 0xe2 and efuse: 0xfe
3. in Atmel Studio a manual erase of the chip has to be done first
4. flash the required EEPROM file to the device "eeprom_..._0001_to_0002_...hex
5. flash the correct firmware "transparent_uart_...hex" onto the device
6. 2.4 GHz - change the fuses to hfuse: 0x91, lfuse: 0x46, efuse: 0xfe
   Sub-1 GHz - change the fuses to hfuse: 0x91, lfuse: 0xe2, efuse: 0xfe
7. Connecting the second device via mini USB (Chapter 2)
8. 2.4 GHz - change the fuses to hfuse: 0x19, lfuse: 0xde and efuse: 0xfe
   Sub-1 GHz - change the fuses to hfuse: 0x19, lfuse: 0xe2 and efuse: 0xfe
9. in Atmel Studio a manual erase of the chip has to be done first
10. flash the required EEPROM file to the device "eeprom_..._0002_to_0001_...hex
11. flash the correct firmware "transparent_uart_...hex" onto the device
12. 2.4 GHz - change the fuses to hfuse: 0x19, lfuse: 0xde and efuse: 0xfe
    Sub-1 GHz - change the fuses to hfuse: 0x19, lfuse: 0xe2 and efuse: 0xfe

After successfully flashing the devices are ready for use. Connect both devices via mini USB to your PC and open two command line tools, we recommend using HTerm. Ensure unobstructed radio reception. The connected AT-ANY BRICKS / dongles are listed in the device manager as serial port.

In your command line tool, you can now connect to the assigned serial ports, a baud rate of 9600 baud, 8 data bits, one stop bit and no parity.

As soon as both devices are connected to one command line tool each, lines can be exchanged via the command line. These lines are received and output on the other device immediately after sending. This is possible in both directions. This way, existing devices can be extended and complicated cable communication can be avoided.

Congratulations, your transparent UART is ready for use.

Reference Documents

[1] SMS Command Reference www.an-solutions.de/SMS_Command_Reference.html

[2] IEEE Std 802.15.4™-2006 (Revision of IEEE Std 802.15.4-2003):
Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs)

[3] @ANY900 / @ANY900 / @ANY2400 Module Product data sheets (https://an-solutions.de/de/produkte.html)

[4] ITU-T V.250: Serial asynchronous automatic dialing and control

Revision History

Revision	Changes	Date
1	create the document	20.02.20
2	Review Engineering area and additions	24.04.20
3	release for publication	25.02.20
4	added chapter 6.2 Transparent UART	15.06.20

Disclaimer

A.N. Solutions believes that all information is correct and accurate at the time of issue. A.N. Solutions reserves the right to make changes to this product without prior notice. Please visit A.N. Solutions website for the latest available version.

A.N. Solutions does not assume any responsibility for the use of the described product or convey any license under its patent rights.

A.N. Solutions warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with A.N. Solutions standard warranty. Testing and other quality control techniques are used to the extent A.N. Solutions deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

Trademarks

AT-ANY, @ANY and related naming as well as A.N. Solutions logo are trademarks of A.N. Solutions GmbH. All other product names, trade names, trademarks, logos or service names are the property of their respective owners.

Technical Support

Technical support is provided by A.N. Solutions GmbH on demand and in accordance to service conditions agreed.
E-Mail: support@an-solutions.de
Please refer to Support Terms and Conditions for full details.

Contact Information

A.N. Solutions GmbH

Am Brauhaus 5
01099 Dresden
Germany

Tel: +49 351 - 30 900 199
Fax: +49 351 - 30 900 189

Office hours: 8:00am - 5:00pm (Central European Time)