

1. RoKiX IoT Platform

1.1. Overview

The *RoKiX IoT Platform* provides a powerful and easy to use environment to begin the evaluation of ROHM and Kionix products. Multiple hardware options are supported (*RoKiX IoT Platform HW*), as well as common, hardware independent SW tools (*RoKiX IoT Platform SW*). *RoKiX Windows GUI* consists of an easy to use graphical user interface for displaying and recording data, and a register map editor which allows the user to see and change the content of control registers. *RoKiX Python CLI*, in turn, provides reference implementation for driver SW and also a convenient way to access device low level features and versatile data logging functionality. Finally, *RoKiX Android App* provides an easy way to display and log the sensor data from a *RoKiX Sensor Node* over Bluetooth.

For a quick set up with:

- *RoKiX Windows GUI*, refer to [section 5.3](#)
- *RoKiX Python CLI*, refer to [section 6.1](#)

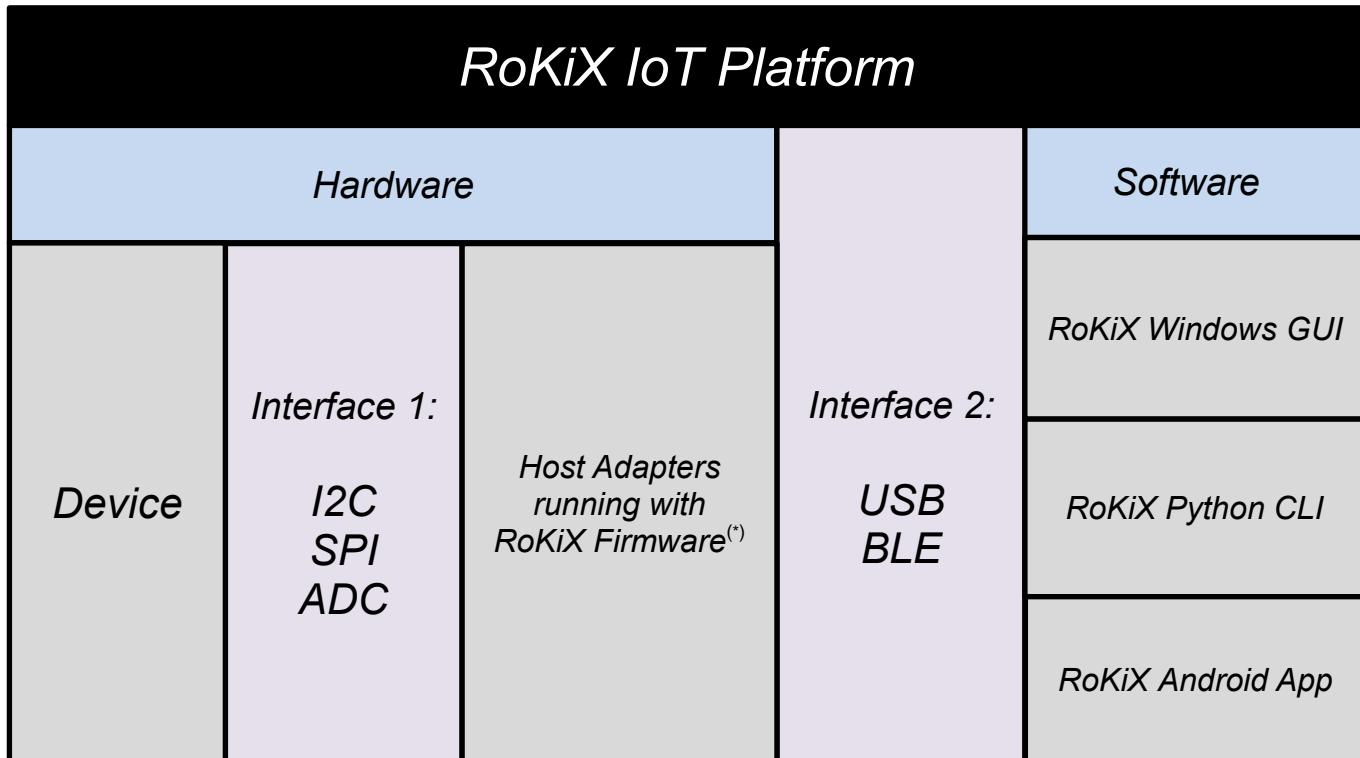


Figure 1. Structure of *RoKiX IoT Platform*.
(*) Additionally also Aardvark is supported in the RoKiX Python CLI.

1.2. Definitions

<i>RoKiX IoT Platform</i>	Complete offering of HW and SW for device evaluation purposes
<i>RoKiX Hardware</i>	PWB containing device and/or bus connection between device and host adapter
<i>RoKiX Sensor Node</i>	MCU + BLE + sensors + battery running <i>RoKiX Firmware</i>
<i>RoKiX Sensor Node Mounting Accessory</i>	Wrist band for <i>RoKiX Sensor Node</i>
<i>RoKiX Adapter Board A3</i>	Board specifically designed to easily interface with RoKiX devices and numerous development platforms
<i>Kionix Evaluation Board</i>	RoKiX sensor evaluation board with 14 pin header http://www.kionix.com/developer-tools
<i>RoKiX Windows GUI</i>	RoKiX device evaluation software with graphical user interface (GUI) running in Windows OS
<i>RoKiX Python CLI</i>	RoKiX device evaluation software with Python based command line interface for quickly testing device low level features
<i>RoKiX Android App</i>	RoKiX Android App is used to access and collect device data from Bluetooth enabled <i>RoKiX Hardware</i>
<i>RoKiX Firmware</i>	Proprietary firmware running on microcontroller based host adapters

1.3. Acronyms

ADB	Android Debug Bridge
ASIC	Application-Specific Integrated Circuit
BLE	Bluetooth Low Energy
DFU	Device Firmware Update
DLL	Dynamic Link Library
DRDY	Data Ready
FFT	Fast Fourier Transform
FTDI	Future Technology Devices International
GPIO	General-Purpose Input/Output
GUI	Graphical User Interface
I2C	Inter-Integrated Circuit
IC	Integrated Circuit
IoT	Internet of Things
LDO	Low-Dropout (regulator)
LED	Light-Emitting Diode
MAC	Media Access Control
MCU	Microcontroller Unit
ODR	Output Data Rate
OS	Operating System
OTA	Over The Air
PWB	Printed Wiring Board
QR	Quick Response (code)
SAD	Slave Address
SoC	System-on-Chip
SPI	Serial Peripheral Interface
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
USB	Universal Serial Bus
WLAN	Wireless Local Area Network

Table of Contents

1.	RoKiX IoT Platform.....	1
1.1.	Overview.....	1
1.2.	Definitions.....	2
1.3.	Acronyms.....	3
2.	Hardware.....	8
2.1.	Hardware architecture.....	8
2.1.1.	Supported hardware configurations	8
2.2.	RoKiX Sensor Node.....	14
2.2.1.	RoKiX Sensor Node Layout.....	16
2.2.2.	Adding new I2C/SPI slave devices	17
2.3.	Kionix IoT Sensor Node	20
2.4.	RoKiX Adapter Board A3	21
2.5.	Rohm Sensor Evaluation Kit	21
2.6.	Aardvark I2C/SPI Host Adapter	21
3.	Software.....	22
3.1.	RoKiX Firmware.....	22
3.2.	RoKiX IoT Platform Client SW	22
3.2.1.	RoKiX Windows GUI.....	22
3.2.2.	RoKiX Python CLI.....	22
3.2.3.	RoKiX Android App.....	23
3.2.3.1.	QR code.....	23
4.	Dependencies	24
4.1.	USB driver installations.....	24
4.1.1.	USB driver for RoKiX Sensor Node and nRF52840-DK	24
4.1.2.	Cypress CY8CKIT-059 USB driver	24
4.1.3.	Aardvark USB driver	28
4.1.4.	Arduino USB driver	28
4.1.5.	Kionix IoT Sensor Node USB driver	29
4.1.6.	Apple OS X and FT232RL device	31
4.1.6.1.	Finding a proper serial port if auto-detect fails	31
4.2.	Pairing Bluetooth enabled RoKiX IoT Platform HW with Windows	31
4.3.	RoKiX Firmware programming and updating	32
4.3.1.	RoKiX Sensor Node firmware	32
4.3.2.	Nordic Semiconductor nRF52840-DK firmware	36
4.3.3.	Arduino firmware.....	36
4.3.4.	Cypress firmware.....	38
5.	RoKiX Windows GUI	39

5.1.	Introduction.....	39
5.2.	Setup	39
5.2.1.	Installation	39
5.2.2.	Configuration	40
5.2.2.1.	Board configuration and connection type.....	40
5.3.	Getting Started.....	40
5.3.1.	Getting Started with Kionix IoT Sensor Node	42
5.4.	User Interface – Menu bar	42
5.4.1.	File – Menu.....	42
5.4.2.	Data – Menu	42
5.4.2.1.	Streaming.....	42
5.4.2.2.	Logging	43
5.4.2.3.	Offline	43
5.4.3.	Connection – Menu.....	43
5.4.3.1.	Windows BLE connection.....	44
5.4.4.	Registers - Menu	44
5.4.4.1.	Load.....	44
5.4.5.	Settings – Menu.....	44
5.4.5.1.	Auto Connect	44
5.4.5.2.	Auto config download	45
5.4.5.3.	Automatic streaming.....	45
5.4.5.4.	COM port	45
5.4.5.5.	Logging	45
5.4.5.6.	Reset connection	45
5.4.5.7.	Paired BLE devices	45
5.4.6.	Stream - Menu.....	46
5.4.6.1.	Sensor Fusion.....	46
5.4.6.2.	Streams with magnetometer.....	47
5.4.7.	Board – Menu	47
5.4.8.	View – Menu	48
5.4.8.1.	Sub-channel view.....	48
5.4.9.	Help – Menu	50
5.4.9.1.	About	50
5.4.9.2.	RoKiX IoT Platform Users Guide	50
5.4.9.3.	GitHub Issues.....	50
5.5.	User Interface - Tabs	51
5.5.1.	Plotter – Tab	51
5.5.1.1.	Raw data.....	52

5.5.1.2.	Zooming	52
5.5.1.3.	Pausing	52
5.5.1.4.	Moving	52
5.5.1.5.	Clearing	52
5.5.1.6.	Frequency analysis	53
5.5.1.7.	Advanced Data Path (ADP)	54
5.5.2.	Cube – Tab	55
5.5.3.	Air Mouse – Tab	56
5.5.4.	Registers – Tab	57
5.5.4.1.	Register sets	57
5.5.4.2.	Register polling function	58
5.5.4.3.	Stream modify mode	59
5.6.	User Interface - Status bar	62
5.7.	User Interface - Pop-up windows	62
5.7.1.	No data pop-up window	62
5.7.2.	Streaming pop-up window	62
5.7.3.	Orientation reset pop-up window	62
5.7.4.	Magnetometer calibration pop-up window	63
5.8.	Orientation reset	63
5.8.1.	Cube reset	63
5.8.1.1.	RokiX Sensor Node	63
5.8.1.2.	Kionix IoT Sensor Node	63
5.8.2.	Air Mouse reset	64
5.8.2.1.	RokiX Sensor Node	64
5.8.2.2.	Kionix IoT Sensor Node	64
5.9.	Shortcuts	65
6.	RoKiX Python CLI	66
6.1.	Installation for Windows OS	66
6.2.	Installation for Linux and OS X	66
6.3.	Python Set Up	66
6.4.	Configuration	68
6.4.1.	Connection to RoKiX IoT Platform HW	68
6.4.2.	Generic settings	70
6.5.	Getting Started	70
6.6.	File structure of the evaluation kit	72
6.7.	Running test applications	73
6.7.1.	Other provided tools	75
6.8.	Changing test application configuration	77

6.9.	Reference driver implementation	78
7.	Troubleshooting and known issues.....	79
7.1.	General.....	79
7.2.	Communications	79
7.3.	RoKiX Windows GUI.....	79
7.4.	RoKiX Python CLI.....	81
8.	Appendix 1	82

2. Hardware

RoKiX IoT Platform is designed to support multiple RoKiX hardware for accessing devices from the *RoKiX IoT Platform SW*.

2.1. Hardware architecture

Usually platform software runs on a client machine where there is no direct connectivity to the device. Because of this, a separate host adapter which transfers messages from the client machine to the device is needed. For the connectivity, two separate connection interfaces are needed.

- Interface 1 between the device and the host adapter. Currently I2C, SPI and ADC connectivity options are supported with dedicated adapter boards.
- Interface 2 between the host adapter and the client machine. Currently USB and BLE connections are supported.

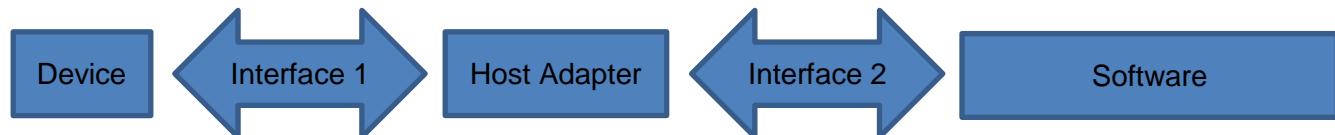


Figure 2: Hardware architecture of *RoKiX IoT Platform*

2.1.1. Supported hardware configurations

Both 3rd party adapters (e.g. Cypress, Aardvark, Arduino Uno, nRF52840-DK) and in-house adapters (e.g. *RoKiX Sensor Node*) are supported. Microcontroller based adapters use proprietary *RoKiX Firmware* for communication.

Adapter boards are offered for “Interface 1” connection between the host adapter and device (sections 2.4 and 2.5). For “Interface 2” connection both wired USB and wireless BLE connections are supported.

Currently supported main hardware configurations are listed in the following tables (Table 1 and Table 2) and output data rates for connection interface options are shown in Table 3, respectively. Appendix 1 introduces other supported HW configurations (Table 10 - Table 13).

Sensor	Interface to sensor HW	Host adapter	Interface to client SW	Client
	I2C SPI ADC	RoKiX Sensor Node	USB or BLE	
RoKiX Evaluation Board, Rohm sensor module, or Sensor Add-on board	I2C SPI ADC	RoKiX Sensor Node		RoKiX Windows GUI or RoKiX Python CLI
RoKiX Evaluation Board, Rohm sensor module, or Sensor Add-on board	I2C SPI ADC	RoKiX Adapter Board A3 Cypress CY8CKIT-059	USB	
	I2C	Kionix IoT Sensor Node		
RoKiX Evaluation Board, Rohm sensor module, or Sensor Add-on board	I2C	Kionix IoT Sensor Node	USB or BLE	

Table 1: Supported main HW configurations for the Windows and Python CLI environments

Sensor	Interface to sensor HW	Host adapter	Interface to client SW	Client
	I2C SPI ADC	RoKiX Sensor Node	Bluetooth 4.1, 4.2, 5.0	
Sensor Add-on board	I2C SPI ADC	RoKiX Sensor Node		
RoKiX Evaluation Board	I2C SPI	RoKiX Adapter Board A3	nRF52840-DK	RoKiX Android App
Rohm sensor module	I2C	Rohm Sensor Evaluation Kit 001/002/003	Bluetooth 4.1	
	I2C	Kionix IoT Sensor Node		
Sensor Add-on board	I2C	Kionix IoT Sensor Node		

Table 2: Supported HW configurations for the Android environment

Development board with interface to device HW		Interface to client SW		
		USB	BT4	BT5
RoKiX Sensor Node	I2C	3200 Hz	\leq 800 Hz	\leq 1600 Hz
	SPI	12.8 kHz	\leq 800 Hz	\leq 3200 Hz
	ADC	X	X	X
Cypress CY8CKIT-059	I2C	3200 Hz	X	X
	SPI	25.6 kHz	X	X
	ADC	10 kHz	X	X
Kionix IoT Sensor Node	I2C	800 Hz	\leq 200 Hz	X
Arduino	I2C	800 Hz	X	X
nRF52840-DK	I2C	3200 Hz	\leq 200 Hz	X
	SPI	6400 Hz	X	X

Table 3: Output data rates for different development boards and interface options

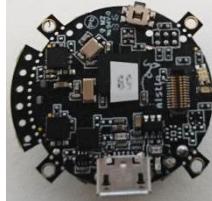
<u>RoKiX Sensor Node Board</u>	 A black square printed circuit board with various electronic components, including a central microcontroller, capacitors, and connectors.
<u>Kionix IoT Sensor Node Board</u>	 A circular printed circuit board with a central microcontroller, sensors, and a USB port.
<u>RoKiX Adapter Board A3</u>	 A green rectangular printed circuit board with a central microcontroller, expansion pins, and a small circular component.
<u>Sensor Add-on Board</u>	 A black rectangular printed circuit board with a central microcontroller, sensors, and a small screen.
<u>Kionix Evaluation Board</u>	 A green rectangular printed circuit board with a central microcontroller, expansion pins, and a small circular component.
<u>Rohm Sensor Module</u>	 A small black rectangular printed circuit board with a central microcontroller, sensors, and a small circular component.

Table 4: HW glossary, part 1.

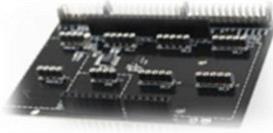
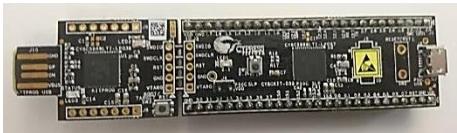
<u>Rohm Sensor Evaluation Kit</u>	
<u>Cypress CY8CKIT-059</u>	
<u>nRF52840-DK</u>	
<u>Raspberry Pi3</u>	
<u>Arduino Uno R3</u>	
<u>Aardvark I2C/SPI host adapter</u> and <u>a level shifter board</u>	

Table 5: HW glossary, part 2.

2.2. RoKiX Sensor Node



Figure 3: *RoKiX Sensor Node*

The *RoKiX Sensor Node* is a small sensor node based on the [Nordic Semiconductor nRF52840 Series](#) system-on-chip (SoC), and it supports both Bluetooth Low Energy (BLE 4.1, 4.2, 5.0) and USB interfaces. It consists of several sensors: accelerometers, magnetometers, gyroscope, and barometer.

The *RoKiX Sensor Node* has the *RoKiX Firmware* preinstalled. Firmware update instructions are described in section 4.3.1.

The hardware configuration of the *RoKiX Sensor Node* can be seen in Table 6. The serial number of the *RoKiX Sensor Node* is given in the form “AA – 12345”, see Figure 4. The Media Access Control address (MAC address) of the device is shown as “AA:BB:CC:DD:EE:FF”, respectively.

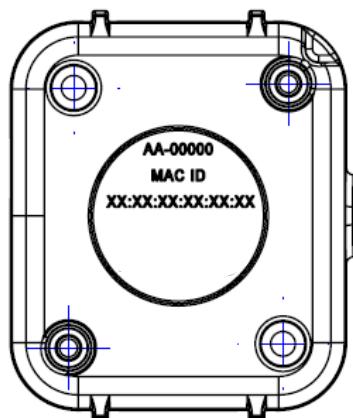


Figure 4: The serial number and the MAC address are shown on the backside of the *RoKiX Sensor Node*.

Product feature	Notes
Enclosure and mounting bracket	
nRF52840 SoC	
External antenna connector	See a guidance below this table
Charger for LiPo battery	
LiPo battery	With wire connector; Joy Battery Technology, type: 652025 Capacity: 250 mAh
On/Off switch	
Battery level measurement	Can be enabled / disabled to save power
Power on indicator	
USB connector	
External SPI memory	Size: 8 Mbit
System top connector	
System bottom connector	
Rohm 5+ connector	Combines the Rohm 5-pin and Kionix 14-pin connectors
RGY LED indicators	
On-board sensors: KX122 KMX62 KXG08 BM1383AGLV BM1422AGMV	

Table 6: Product description for *RoKiX Sensor Node (MAX variant)*

NOTE: Initially with the *RoKiX Sensor Node*, the Bluetooth signal is connected to the internal chip antenna (Figure 5). However, if the BLE connector is used for external antenna connection, please follow these instructions:

- Jumper pins 2 and 3 should be routed together,
- the connector for the external antenna is Murata MM4829-2702,
- for example: the antenna's side connector is IPEX MHF1 (u.fl compatible):

Pin	Name	Description
1	ANT	Antenna RF transmit/receive
2	GND	Ground
3	GND	Ground

2.2.1. RoKiX Sensor Node Layout

The main parts of layout and component placement are as follows (Figure 3, Figure 5 and Figure 6):

- A. Power switch
- B. Red power-on LED
- C. Red battery charging indication LED. When the battery is charging this LED is lit
- D. USB connector for communication (yellow LED)
- E. Blinking green status LED indicates BLE advertisement
- F. Green LED is lit when BLE is connected
- G. Red LED for indicating the bootloader mode

Remarks on the positioning of the onboard sensors (see Figure 5 and Figure 6):

- Sensor position 1 (LGA-12 footprint):
 - I2C and SPI busses
- Sensor position 2 (LGA16 footprint):
 - Suitable placement for a magnetometer
 - I2C and SPI busses
- Sensor position 3 (LGA-14 footprint):
 - I2C and SPI busses
- Sensor position 4 (LGA-10 footprint):
 - I2C bus
- Sensor position 5 (LGA-12 footprint):
 - Suitable placement for a magnetometer
 - I2C bus

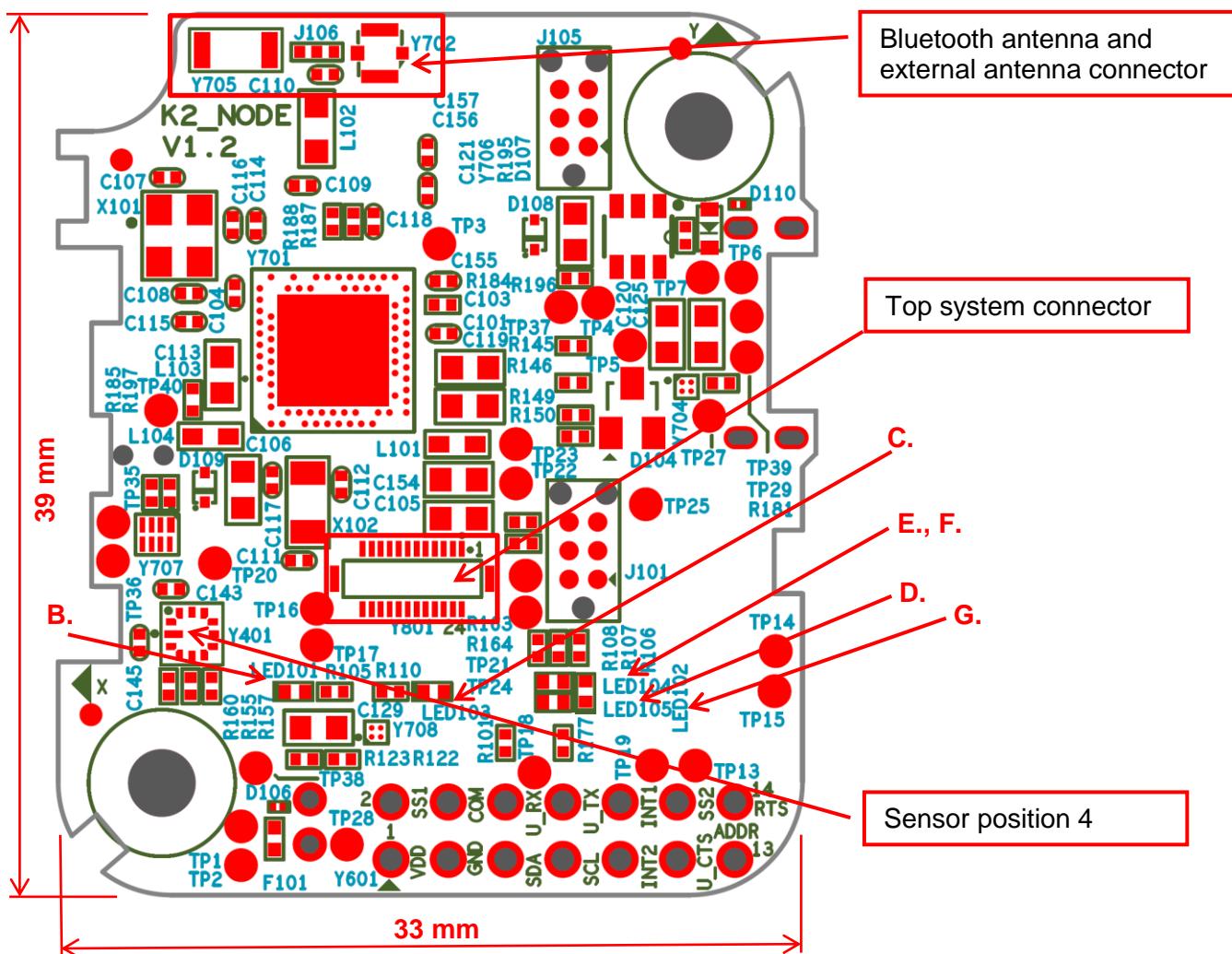


Figure 5: *RoKiX Sensor Node top component placement*

2.2.2. Adding new I2C/SPI slave devices

More devices can be connected to the *RoKiX Sensor Node* in two ways (Figure 5 and Figure 6):

- Sensor Add-on Board on the Top system connector: see Figure 7.
For assembling/disassembling the add-on board, please follow instructions given by the [connector manufacturer](#).
- Rohm 5+ connector that is a combination of the Rohm 5-pin and Kionix 14-pin connectors: with suitable SW and HW controls, the evaluation boards of Rohm and Kionix can be plugged-in directly to the connector. Connector specifications are presented in Table 7, and the correct placement of the sensor board on the bottom side of the board is shown in Figure 8. Pins marked with yellow color in Table 7 are for the Rohm sensor module, see Figure 8.

NOTE: the Top system connector on the *RoKiX Sensor Node Board* is intended for connecting Sensor Add-on Boards (Figure 5). On the contrary, the Bottom system connector is intended only for “other use case” System Add-on Boards (Figure 6).

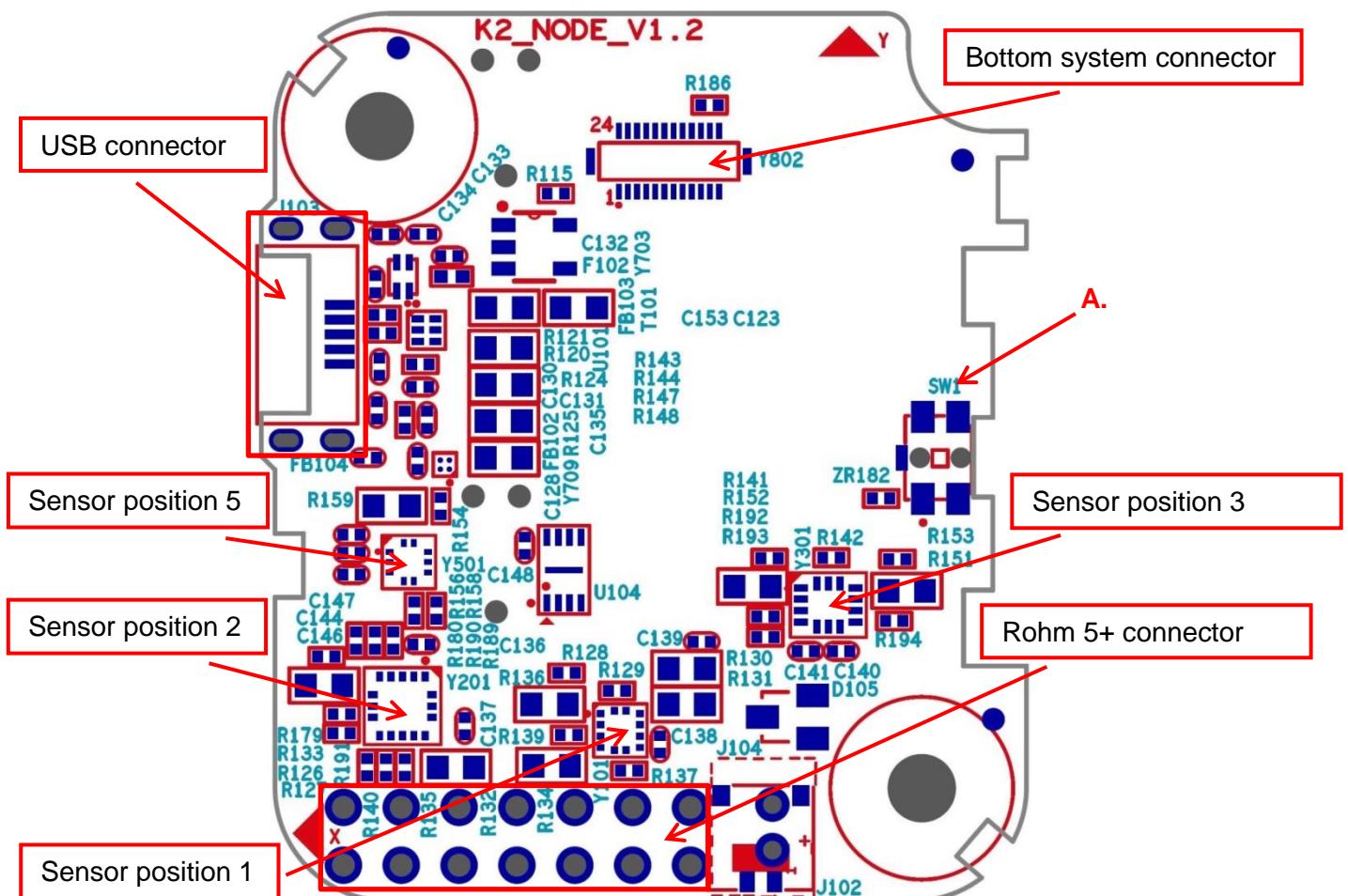


Figure 6: RoKiX Sensor Node bottom component placement

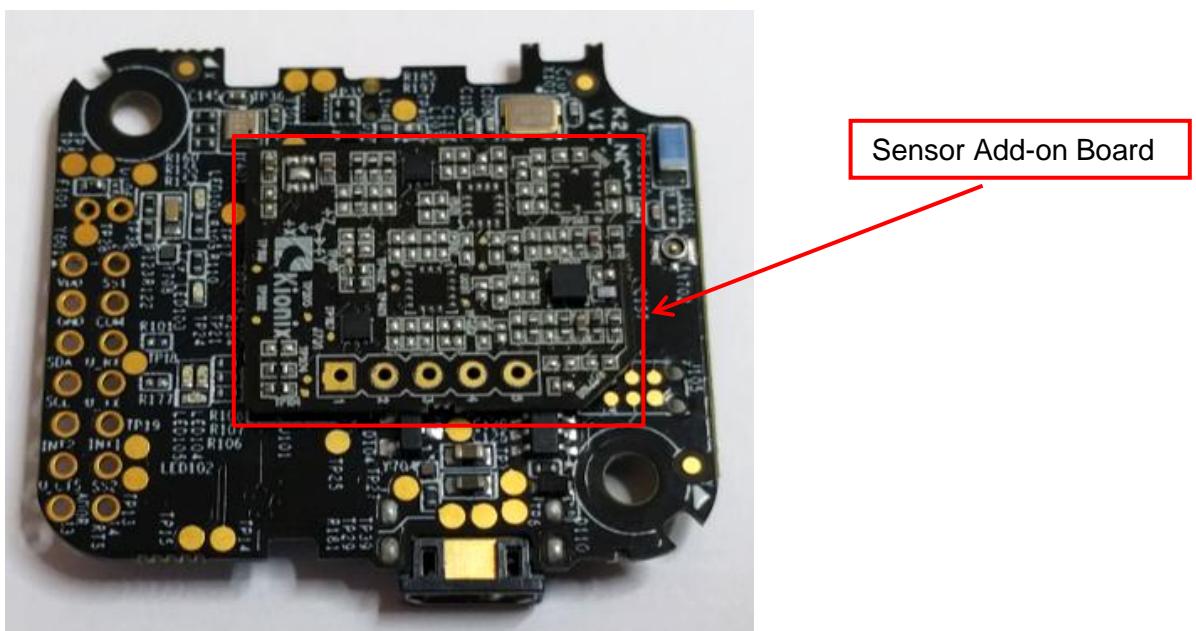
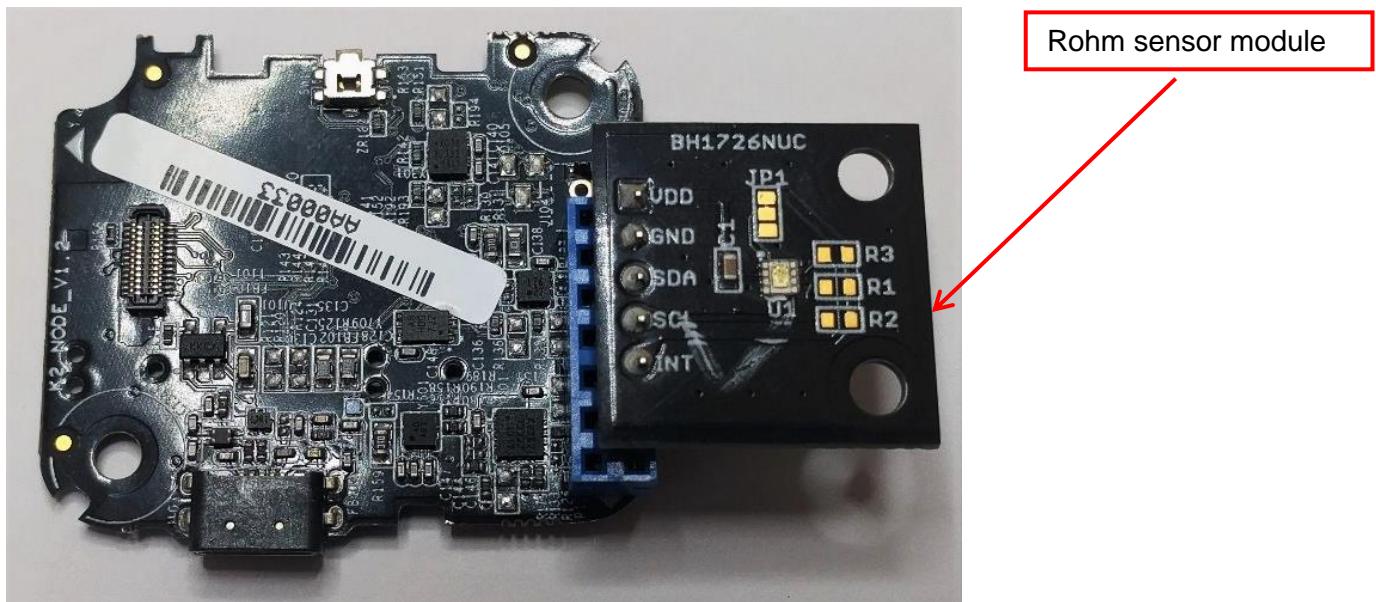


Figure 7: Correct placement of the Sensor Add-on Board on the Top system connector

SoC pin	SPI	I2C	Silk text	<u>CONNECTOR</u>	Silk text	I2C	SPI	SoC pin
			VDD	1	2	SS1		Chip select 1 P0.12
			GND	3	4	COM		AIN5
P0.19	MOSI	SDA	SDA	5	6	U_RX		P0.21
P0.24	SCLK	SCL	SCL	7	8	U_TX		P0.20
P1.01	Interrupt 2	Interrupt 2	INT2	9	10	INT1	Interrupt 1	P1.12
P0.22			U_CTS	11	12	SS2		Chip select 2 P0.14
P0.15	MISO	Slave address	ADDR	13	14	RTS		P0.23

Table 7: Specifications for the Rohm 5+ connector

Figure 8: Correct placement of the Rohm sensor module on the bottom side of the *RoKiX Sensor Node Board* using the Rohm 5+ connector.

2.3. Kionix IoT Sensor Node

Serial number	#100-899 White sticker	#900- Green sticker	#1111- Yellow sticker
Board name in <i>RoKiX Windows GUI</i>	kionix_iot_node_ _k1_build2	kionix_iot_node _k1_build3	kionix_iot_node _k1_build4
Gyroscope	KXG03	KXG03	KXG08
Accelerometer	KX122	KX126	KX122
Magnetometer	KMX62	KMX62	KMX62
Barometer	BM1383AGLV	BM1383AGLV	BM1383AGLV

Table 8: Sensors in *Kionix IoT Sensor Node*

The serial number can be found on the label located on the sticker of the *Kionix IoT Sensor Node*.



Figure 9: *Kionix IoT Sensor Node* serial number

Detailed information about the *Kionix IoT board* can be found at the [Kionix website](#).

Notes:

- Before using the *Kionix IoT Sensor Node* with a battery, please make sure the battery is fully charged. The battery is charged via the USB connector. The battery is fully charged when the battery charging LED turns off.
- If a battery is connected, the *Kionix IoT Sensor Node* does not automatically turn on the power when the USB is connected to the PC. The power switch must be pressed to power on the device before connecting the USB cable.

2.4. RoKiX Adapter Board A3

The *RoKiX Adapter Board A3* is a tool specifically designed to easily interface with ROHM/Kionix devices and numerous development platforms. It is a great platform for someone who needs an easy way to interface with Kionix's standard evaluation boards featuring a 14-pin male header, RoKiX 24-pin add-on boards, or [ROHM Sensor Shield Modules](#) 5-pin digital or 4-pin analog boards, with ready-to-use development platforms such as Cypress, Arduino, and Raspberry Pi.

The adapter board utilizes 3.3V/5V power from the host platform and converts to rotary switch selectable (1.7V/1.8V/2.5V/2.8V/3.0V/3.3V/3.6V) VDD voltage using a dedicated ROHM adjustable LDO. There is also an option to bypass the voltage regulator and provide the power from the host directly to devices that meet the voltage requirements. Furthermore, the IO_VDD voltage can be supplied from an independent adjustable LDO set to 3.3V or be tied to the selected VDD voltage using an onboard switch. The adapter board also features a pair of level shifters that can be used in I2C and SPI communication to interface 5V host systems with devices supporting lower VDD voltages. Current measurement is possible via removable jumpers on VDD rail and via in circuit removable zero-ohm jumper on IO_VDD rail. An LED is provided on the power rail to indicate the board is powered by the host.

2.5. Rohm Sensor Evaluation Kit

Rohm's [SensorShield-EVK-001/002/003](#) sensor evaluation kit is compatible with open platforms such as Arduino Uno and mbed. The sensor shield enables easy evaluation of the 8 onboard high-performance sensors, making it ideal for verifying sensor operation, initial set development, and as a learning and training tool. [SensorShield-EVK-001/002/003](#) can be used with the *RoKiX IoT Platform* by using Nordic Semiconductor's nRF52840-DK and Arduino Uno development boards through the *RoKiX Firmware*, ref. section 3.1.

Jumper settings for all the ROHM/Kionix 5-pin setups are as shown in Figure 10.

NOTE: BM1383AGLV should be connected to the position J7, all others to the position J5.

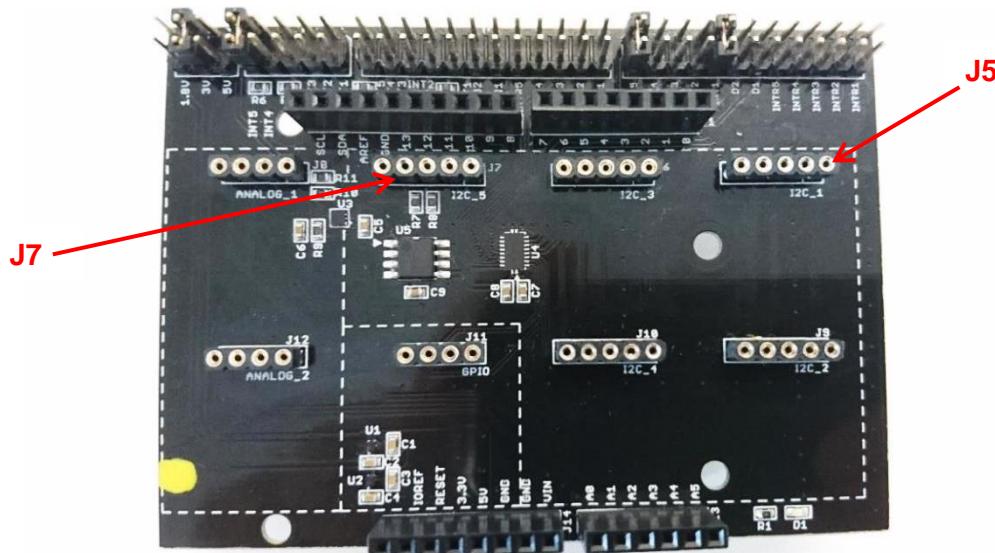


Figure 10: Rohm EVK-001 with proper jumper configuration

2.6. Aardvark I2C/SPI Host Adapter

Aardvark is a product of [Total Phase Inc.](#). *RoKiX Python CLI* running on a Windows PC supports both I2C and SPI connections on Aardvark. Since Aardvark uses 5 volts for logic voltage, level shifting may be required.

3. Software

RoKiX IoT Platform SW consists of both *RoKiX Firmware* for supported evaluation kit hardware, as well as, *RoKiX IoT Platform Client SW* for evaluating the array of devices ROHM/Kionix has to offer.

3.1. RoKiX Firmware

Firmware for supported microcontroller based *RoKiX IoT Platforms* is provided by ROHM. Programming instructions and updating instructions are described in section 4.3. *RoKiX Firmware* supports communication with *RoKiX IoT Platform Client SW*. ROHM currently offers firmware for:

- RoKiX Sensor Node
- Cypress CY8CKIT-059
- Kionix IoT Sensor Node
- Arduino Uno R3
- Nordic Semiconductor nRF52840-DK

3.2. RoKiX IoT Platform Client SW

The provided *RoKiX IoT Platform Client SW* is divided into three parts: *RoKiX Windows GUI*, *RoKiX Python CLI* and *RoKiX Android App*.

3.2.1. RoKiX Windows GUI

RoKiX Windows GUI provides an intuitive graphical user interface demonstrating high level device offerings and features. Some of the features included are:

- Visual display of real-time device data
- Ability to record device data to a file
- Device registry editor
- Demonstration of RoKiX Software offerings such as
 - Sensor Fusion Algorithm
 - Air mouse

Please see chapter 5 for usage instructions for the *RoKiX Windows GUI*.

3.2.2. RoKiX Python CLI

RoKiX Python CLI provides an advanced interface for demonstrating low level device features. Some of the features included are:

- Independent of Operating System
- Reference implementation for creating device driver software
- Flexible tool for recording device data
- Reference implementation for usage of device ASIC level features
- Framework for quick modification and testing of device functionality

The usage instructions for the *RoKiX Python CLI* can be found in chapter 6. Section 6.3 describes the installation processes of the necessary Python environment.

3.2.3. RoKiX Android App

RoKiX Android App's purpose is to collect device data over Bluetooth low energy connection. Currently this application is available for Android devices in the [Play store](#) or [GitHub](#). The application is also bundled with the *RoKiX IoT Platform* and located in the folder \Documents\RoKiX\RoKiX-Android-App .

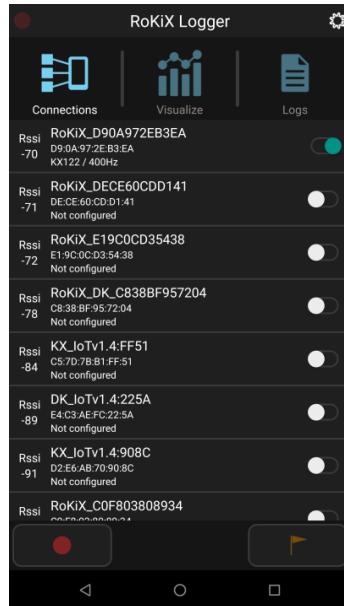


Figure 11: *RoKiX Android App*

3.2.3.1. QR code



4. Dependencies

This chapter lists dependencies needed by *RoKiX IoT Platform SW*.

4.1. USB driver installations

Sections 4.1.1 and 4.1.2 provide installation instructions for the USB driver for the *RoKiX Sensor Node*, nRF52840-DK, and Cypress CY8CKIT-059. However, if the *RoKiX IoT Platform* software installer is used, USB driver installations are not required on these development platforms.

4.1.1. USB driver for RoKiX Sensor Node and nRF52840-DK

With Windows 10, the operating system should automatically use the correct USB driver. Earlier Windows versions are not able to automatically find the CDC ACM driver and the user will need to install the signed release inf file from the USB drivers (\Documents\RoKiX\RoKiX-Firmware\Windows-dependencies\RoKiX-Sensor-Node\cdc_acm_driver).

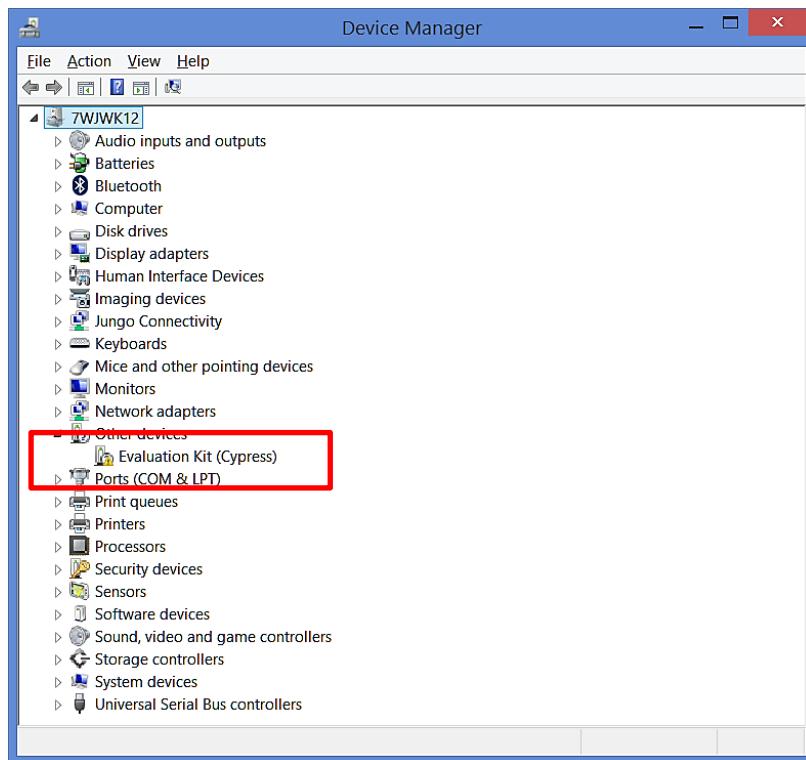
NOTE: For CDC ACM USB driver installation, please follow the instructions in the next section “*Cypress CY8CKIT-059 USB driver*” and select the inf file from \Documents\RoKiX\RoKiX-Firmware\Windows-dependencies\RoKiX-Sensor-Node\cdc_acm_driver .

4.1.2. Cypress CY8CKIT-059 USB driver

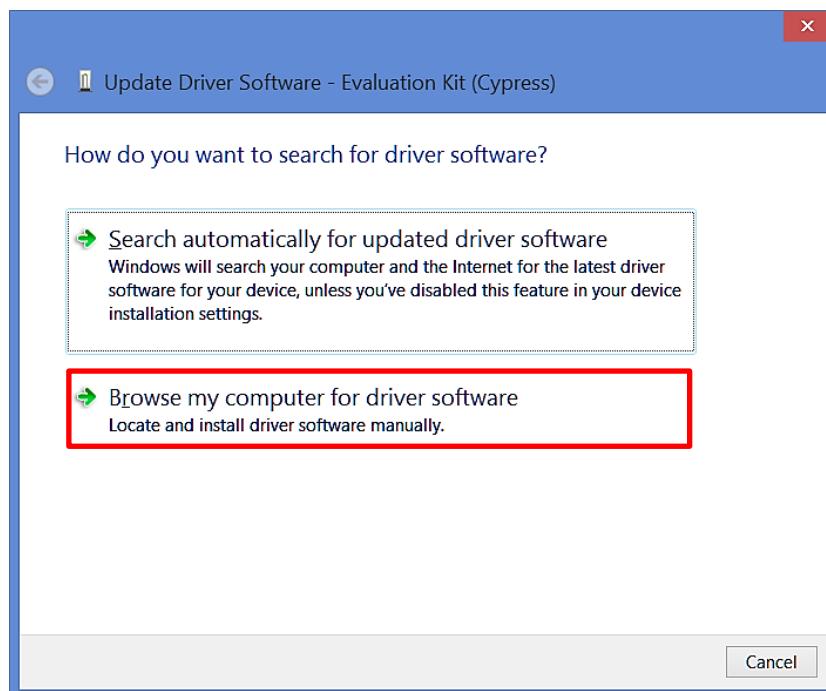
On Windows 10, the operating system should automatically use the correct driver. Earlier versions of Windows cannot automatically find the CDC driver and the user will need to install the signed driver.

For USB driver installation, please follow the steps below.

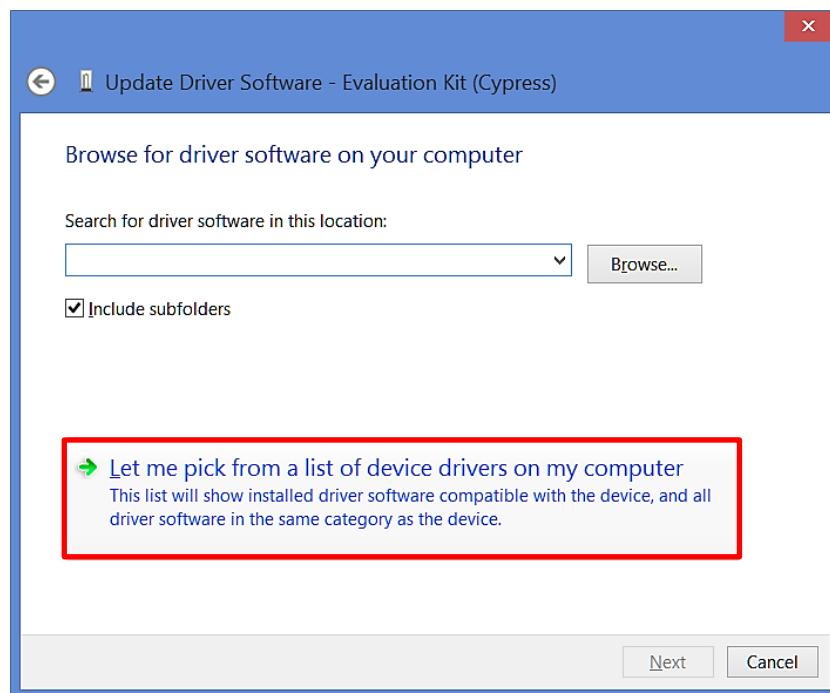
1. Acquire the inf and cat driver files:
 \Documents\RoKiX\RoKiX-Firmware\Windows-dependencies\RoKiX-Sensor-Node\cdc_acm_driver
2. Connect the Cypress board to a computer using the Cypress kit's Micro-USB connector.
3. Open the Device Manager where you should find the “Evaluation Kit (Cypress)” as shown in the figure below.



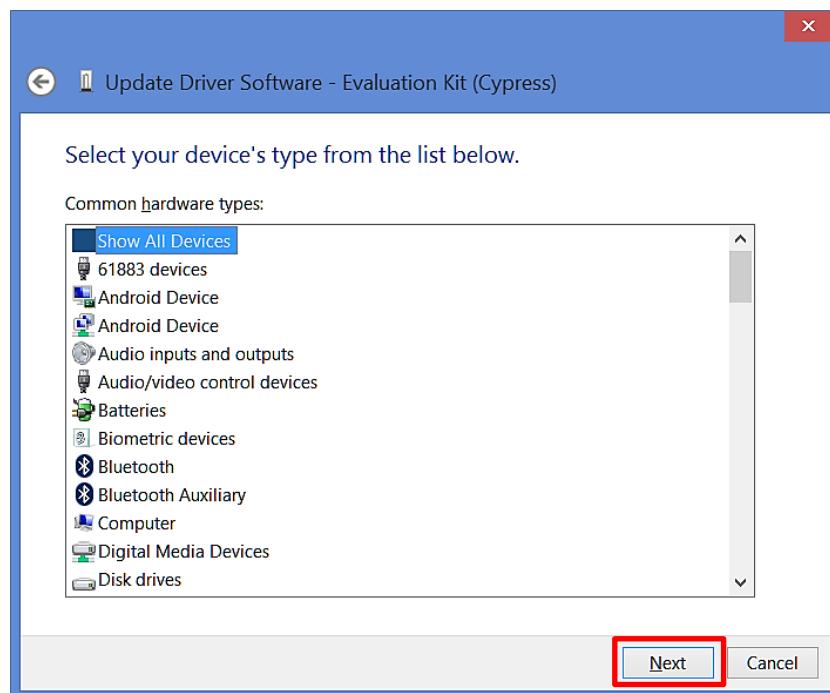
Right-click the “Evaluation Kit (Cypress)” item, and choose “Update Driver Software ...”. A new window should open. Select “Browse my computer for driver software” as shown below.



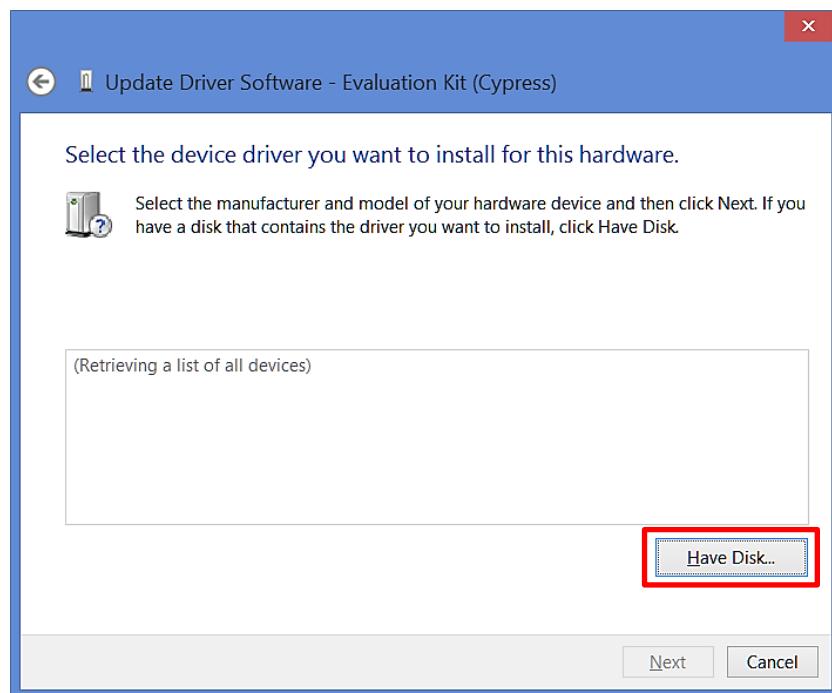
4. Select “Let me pick from a list of drivers on my computer”:



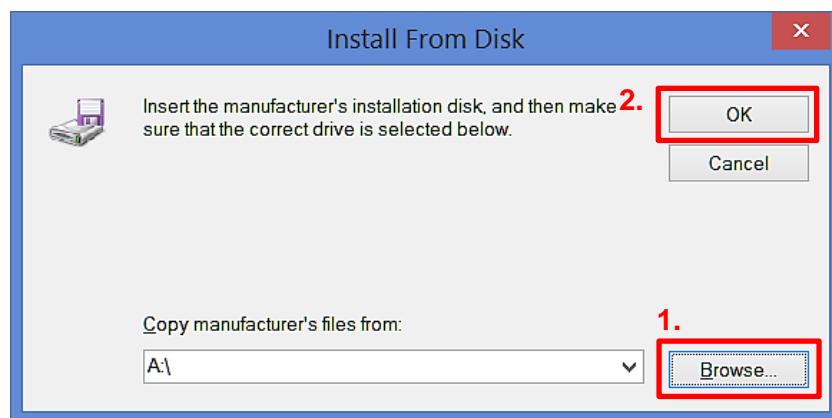
5. Choose “Next”, the selection in the list does not matter:

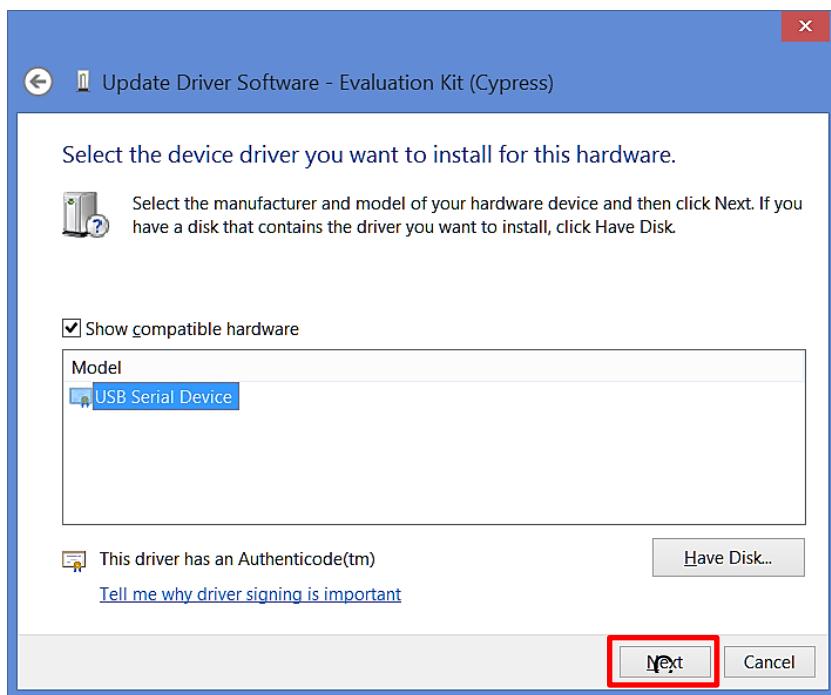


6. Choose “Have Disk” as shown below:

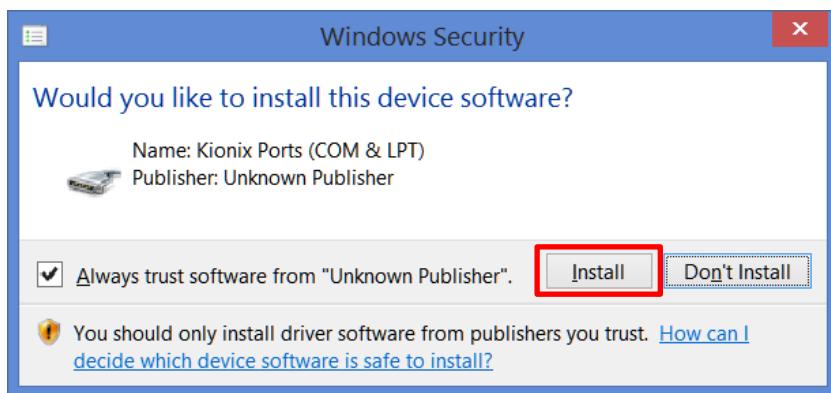


7. Then choose “Browse” and find the downloaded inf file and then select “OK”. You will be returned to an older pop-up window where you should select “Next”.





8. Windows will prompt you to install the driver, please select “Install”.



9. Finally, please wait until the driver installation is complete.

4.1.3. Aardvark USB driver

The Aardvark I2C/SPI host adapter is supported by *RoKiX Python CLI*.

The Total Phase USB driver for Aardvark is bundled with the *RoKiX IoT Platform* and located in the folder `\Documents\RoKiX\RoKiX-Firmware\Windows-dependencies\Aardvark`. The latest drivers are also available at <http://www.totalphase.com/products/usb-drivers-windows>

4.1.4. Arduino USB driver

For installation of the Arduino board driver, download the Arduino software package from [Github](#). This download includes the driver file named `arduino.inf`, which is located in the `\Arduino\build\windows\dist\drivers` folder of the Arduino software package, and it may be installed by following the step-by-step guide shown in [here](#).

4.1.5. Kionix IoT Sensor Node USB driver

Kionix IoT Sensor Node uses FTDI for USB serial communications. If Windows does not install the needed drivers automatically, they can be loaded from: <http://www.ftdichip.com/Drivers/VCP.htm>

Interoperability is tested with the following driver version
http://www.ftdichip.com/Drivers/CDM/CDM21216_Setup.exe

If drivers are not installed correctly, the *Kionix IoT Sensor Node* will be shown in the Device Manager in the following way:

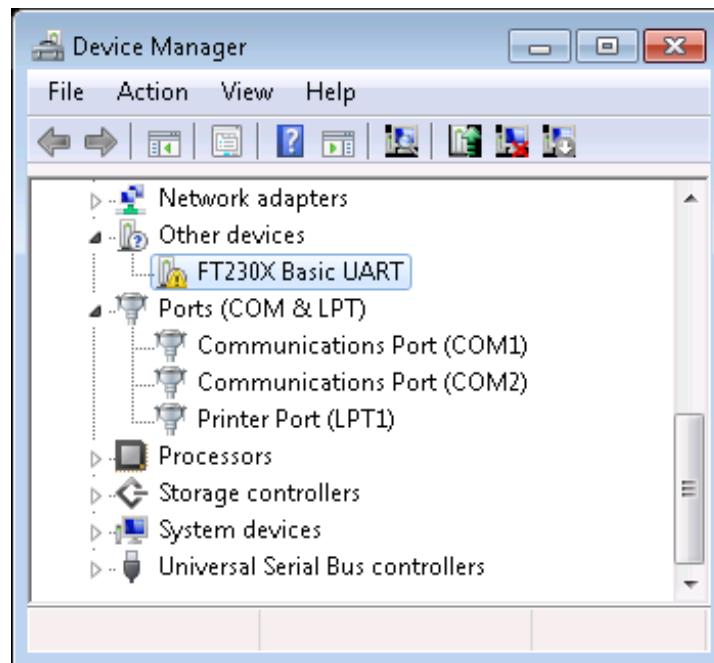


Figure 12: *Kionix IoT Sensor Node* USB driver not properly installed

After drivers are installed, *Kionix IoT Sensor Node* is listed as “USB Serial Port (COMx)” where ‘x’ is the COM port allocated for the *Kionix IoT Sensor Node*.

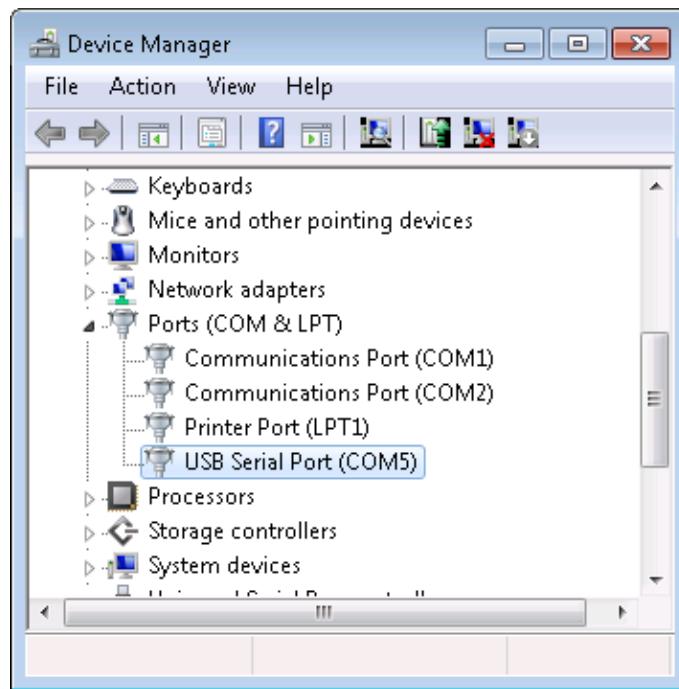


Figure 13: Kionix IoT Sensor Node USB driver properly installed

When using FTDI USB Serial connections with high ODR speeds (≥ 400 Hz), the COM port setting should be updated to ensure data is transferred as soon as it is available. Otherwise the data may come in bursts and time stamping values may not be correct.

The following configuration of a COM port was found to be suitable:

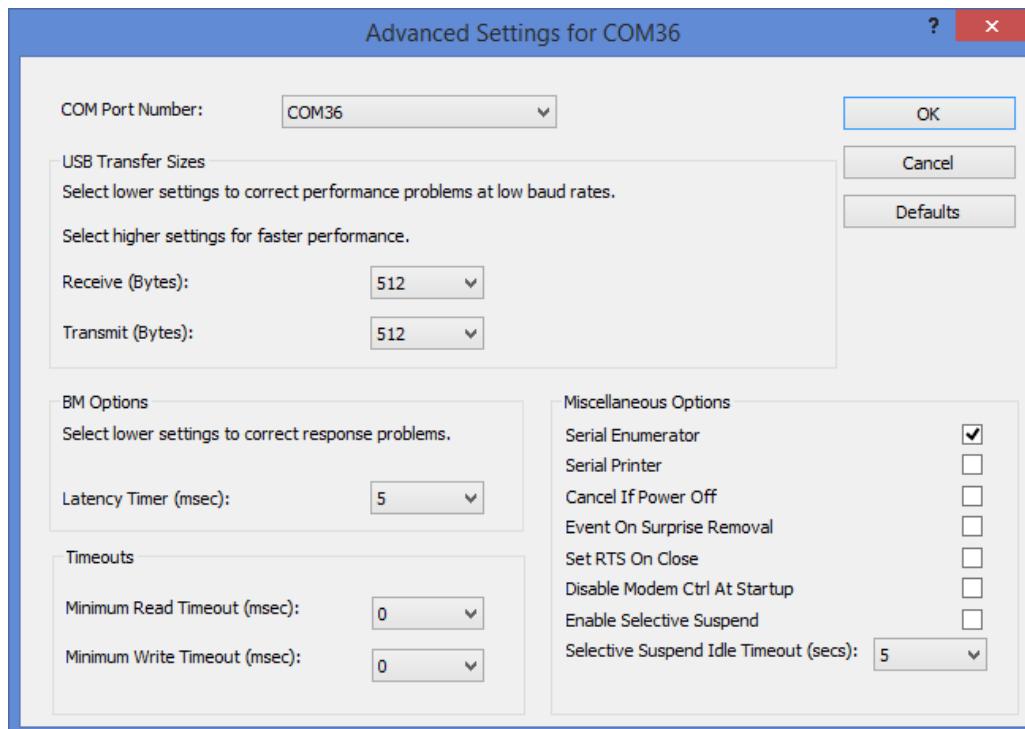


Figure 14: Kionix IoT Sensor Node USB settings

4.1.6. Apple OS X and FT232RL device

This section gives some notes on using the *RoKiX Sensor Node* or some versions of the Arduino Uno with an FTDI FT232RL device.

Apple OS X version 10.9 and later versions contain driver software for FTDI which is implemented by Apple. Usage scenarios with Apple FTDI driver(s) are limited; only drdy_function_mode = TIMER_POLL is working reliably.

For higher ODRs, the user needs to install the FTDI VCP driver. The installation instructions are available at:

http://www.ftdichip.com/Support/Documents/AppNotes/AN_134_FTDI_Drivers_Installation_Guide_for_MAC OSX.pdf

After an installation reboot of your machine, connect the evaluation board and check the driver installation using the terminal command

```
kextstat | grep FTDI
```

The output should contain only one line containing the text "com.FTDI.driver.FTDIUSBSerialDriver". If there are several lines, then the Apple FTDI driver is still loaded and may cause communication problems. In this case, follow the instructions in AN134 chapter 7 on disabling the Apple provided VCP driver for OS X.

4.1.6.1. Finding a proper serial port if auto-detect fails

If the COM-port is not automatically found/located the proper COM-port can be found (to be set) in *rokix_settings.cfg* with the terminal command

```
ls -l /dev/tty.*
```

FTDI based COM-ports are seen in the listing as "/dev/tty.usbserial-*", COM-ports using a generic USB CDC-device are seen as "/dev/tty.usbmodem*".

4.2. Pairing Bluetooth enabled RoKiX IoT Platform HW with Windows

If the established *RoKiX IoT Platform HW* supports Bluetooth connectivity, the pairing with Windows is done with the following steps:

1. Go to Control Panel\All Control Panel Items\Devices\Bluetooth & other devices
2. Add Bluetooth or other device
3. Choose Bluetooth
4. Choose "RoKiX_<MAC address>"
5. Next
 >> A LED light on the *RoKiX Sensor Node* should turn green during the device installation

If the pairing succeeded, the "RoKiX_<MAC address>" device is visible in the "Other devices" - section of "Bluetooth & other devices". Note that the device name "RoKiX_<MAC address>" is used there as an example: many other names exist as diverse HW and firmware versions are supported. For example, with the nRF52840-DK development board the device name will appear as "DK_IoTv1.4:90BC".

During Bluetooth pairing with Windows 10, the status LED in the node will change from red to green for a few seconds, but turn to red again until the *RoKiX Windows GUI* is started and an active connection is maintained.

When the connection is changed back to USB, the *RoKiX Sensor Node* will also start to use USB automatically. The color of the *RoKiX Sensor Node* status LEDs will change from green to red.

4.3. RoKiX Firmware programming and updating

This section describes the programming process of *RoKiX Firmware* to the supported *RoKiX IoT Platform HW*. These instructions are also suitable for programming the *Kionix IoT Node Firmware*.

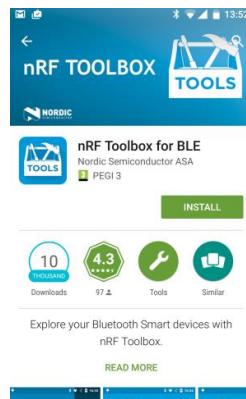
4.3.1. RoKiX Sensor Node firmware

This section provides instructions on how to use the DFU (Device Firmware Update) to update the bootloader and *RoKiX Firmware* to the *RoKiX Sensor Node* with OTA (Over The Air) using an Android or IOS phone.

The *RoKiX Sensor Node* is shipped with a pre-installed bootloader and *RoKiX Firmware*. Both can be updated with an iPhone or Android using the “DFU” functionality of the “nRF Toolbox”, which is available in the application store (for example in [Google Play](#)). The *RoKiX Firmware* is also bundled with the *RoKiX IoT Platform* and located in the folder \Documents\RoKiX\RoKiX-Firmware\RoKiX-Sensor-Node.

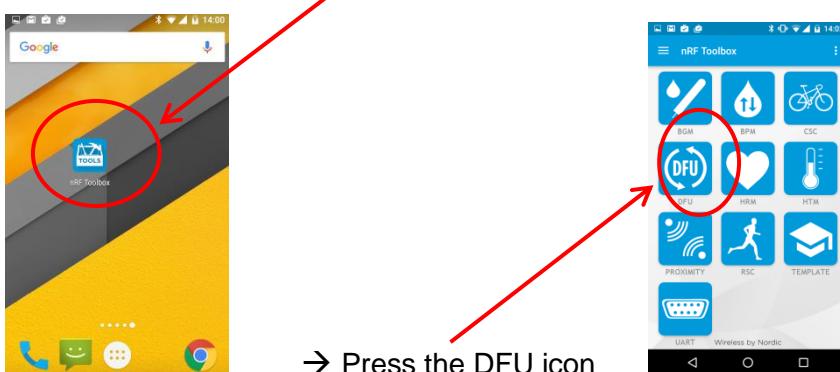
Update the *RoKiX Firmware* to the *RoKiX Sensor Node* with the following procedure:

- 1) Load the Nordic Toolbox App (nRF Toolbox App) for your Android –phone (Google Play store) or Apple Store for IPhone. Here is an example of the Android phone installation via the Google Play store;



Press the Install button

2) Start the application from the main icon

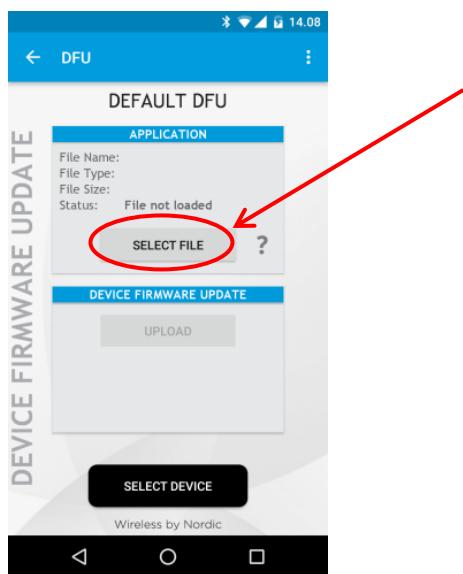


3) The DFU menu will open

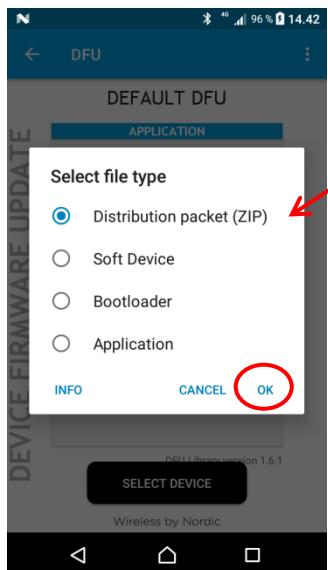
4) The *RoKiX Firmware* is bundled with the *RoKiX IoT Platform* and located in the directory
\\Documents\\RoKiX\\RoKiX-Firmware\\RoKiX-Sensor-Node .

1. Load the firmware update file to the Android phone into the directory of your choice. For example RoKiX-IoT-firmware.zip to /sdcard –directory.
2. The *RoKiX Evaluation Kit Firmware* can be downloaded to the device from GitHub:
[RoKiX Firmware](#)

5) Press the “Select File”-button



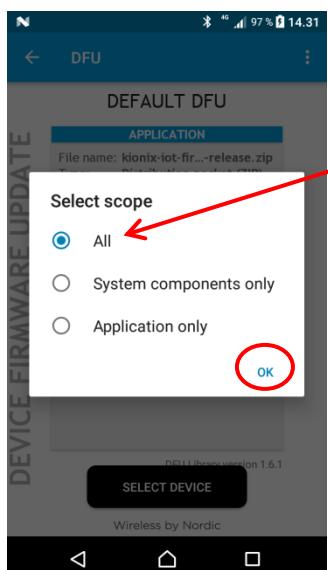
Select “Distribution packet (ZIP)”



Press the OK button

6) Select your favorite file manager program

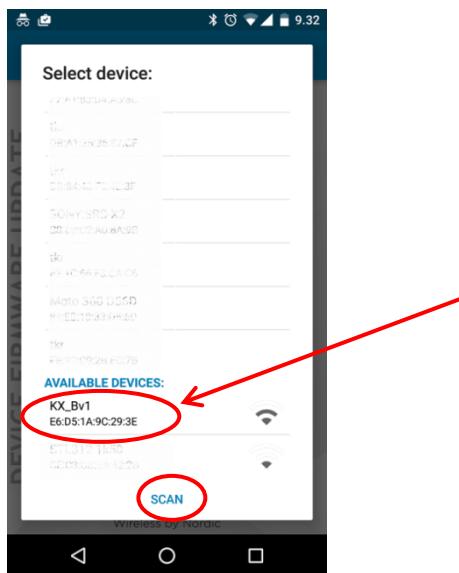
1. Select the file
2. Select scope (All)



3. Press OK button

7) Press the “Select Device”-button

1. Scan to update MAC number info and choose your device



2. Select your *RoKiX Sensor Node*'s MAC number from the list

8) Press the “Upload”-button

1. Wait until the uploading starts (may take up to 2 minutes)
 - i. The upload status bar informs the percentage level of the current upload
 - ii. If uploading does not start, please check “Troubleshooting hints” below.
2. The DFU application should notify the user that the update was successful

Troubleshooting hints:

- a. Make sure that the battery of the *RoKiX Sensor Node* is fully charged or plug the USB cable to the *RoKiX Sensor Node* during the DFU operation.
- b. Reboot the device in which the nRF toolbox is running
- c. Disable WLAN from the device in which the nRF toolbox is running
- d. Check the package size setting parameters in the DFU Settings

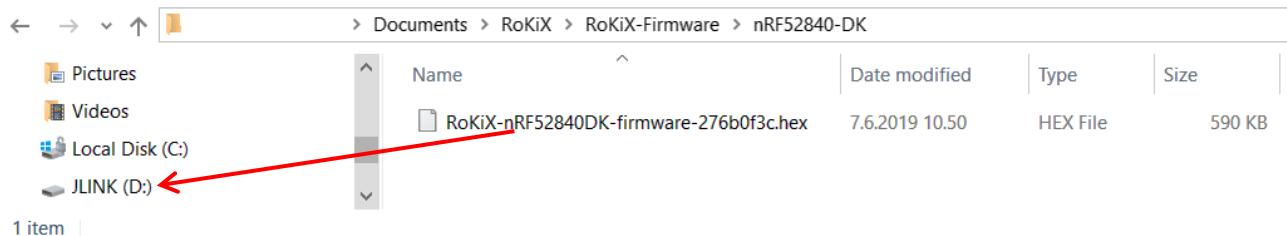
References:

- [nRF Toolbox manual](#)
- [Nordic Semiconductor Infocenter](#)
- [Nordic Semiconductor Developer Forum](#)

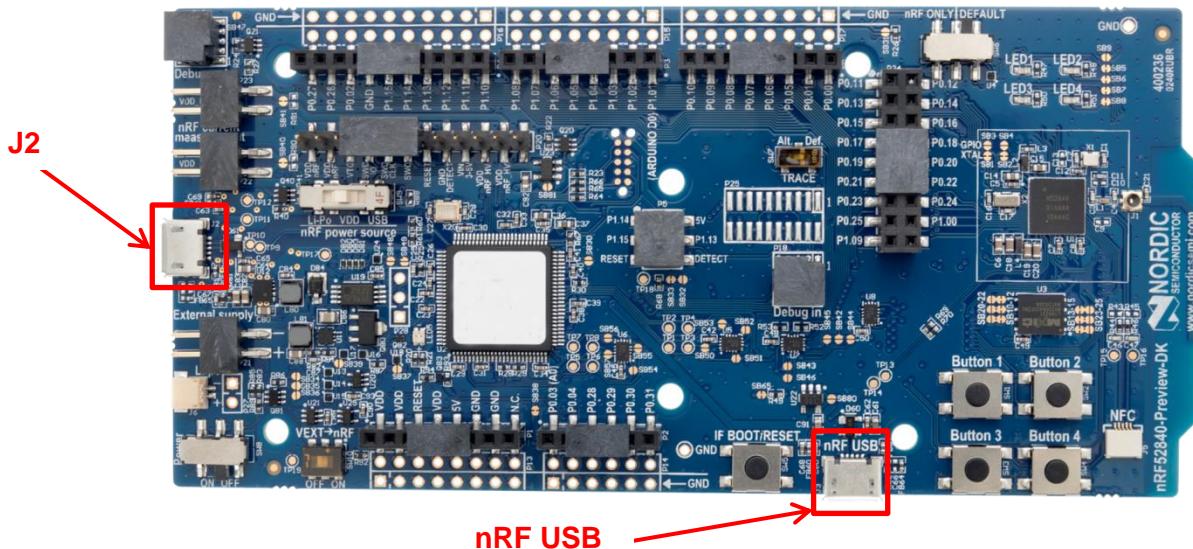
4.3.2. Nordic Semiconductor nRF52840-DK firmware

The [nRF52840-DK development kit](#) has a pre-installed mbed boot loader. The *RoKiX Firmware* can be programmed to the development board with the following steps:

- Connect a USB cable to the USB port J2 of the nRF52840-DK, and the other end to the PC. The nRF52840-DK is listed as a hard drive in the Windows File Explorer.
- Copy & Paste the firmware to nRF52840-DK. The firmware is located in \Documents\RoKiX\RoKiX-Firmware\nRF52840-DK or it can be loaded from [GitHub](#).



NOTE: The nRF52840-DK board contains an nRF USB port that is used with the *RoKiX Windows GUI* and *RoKiX Python CLI*.



4.3.3. Arduino firmware

Since the Arduino has a pre-installed bootloader, the *RoKiX Firmware* can be programmed through a USB connection. If the Arduino board is shipped with the *RoKiX IoT Platform* then the *RoKiX Firmware* has already been pre-installed.

- The Arduino USB driver must be installed as a prerequisite for installing the *RoKiX Firmware*, see section 4.1.4. The *RoKiX Firmware* and `flash.bat` for the Arduino is bundled with the *RoKiX IoT Platform*, and they are located in the folder \Documents\RoKiX\RoKiX-Firmware\Arduino .

If the Arduino IDE is already installed then the *RoKiX Firmware* programming can be done from the Windows command line using `flash.bat`. Otherwise, please use the XLoader tool.

The XLoader tool can be downloaded from

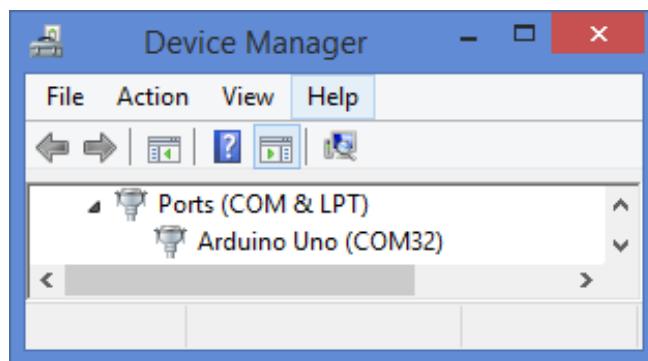
<http://xloader.russemotto.com/>,

Place this software package in a separate directory (see also [GitHub](#)):

\Documents\RoKiX\RoKiX-Firmware\Arduino

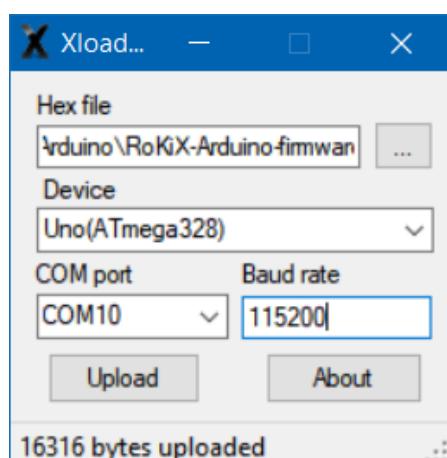
Update the *RoKiX Firmware* to the Arduino with the following procedure:

- 1) Open the Device Manager from Control Panel.
- 2) Look under ports (COM & LPT). You should see “Arduino Uno”,
- 3) Check the port number.



- 4) Navigate to the tools folder and open XLoader.
- 5) With the XLoader interface, select:
 - o The Arduino firmware hex file
 - o Device as Uno (ATmega328)
 - o COM port as the comport Arduino is connected
 - o Baud rate as 115200
- 6) Press Upload

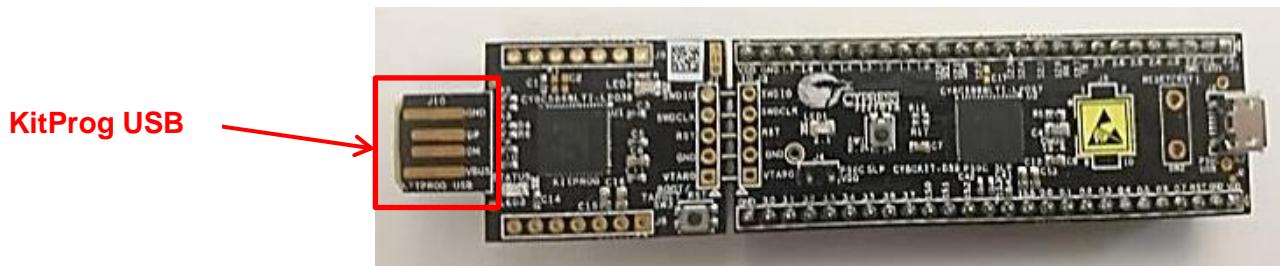
After a successful update, the console printout should look like this:



4.3.4. Cypress firmware

The Cypress device can be programmed with the stand-alone [PSoC Programmer](#), please follow the steps below.

1. Install and launch the PSoC Programmer.
2. Connect the KitProg USB to the PC:



3. If the indicator at the lower-right corner of the programmer is red and displays "Not Connected", select the KitProg from the Port Selection box.
4. Use "File -> File Load" to select the Cypress firmware hex file. The firmware is located in `\Documents\RoKiX\RoKiX-Firmware\Cypress-PSoC`. See also [GitHub](#).
5. Use "File -> Program" to program the device.

5. RoKiX Windows GUI

5.1. Introduction

The *RoKiX Windows GUI* provides an easy to use graphical user interface demonstrating high level device offerings and features. Some of the features include:

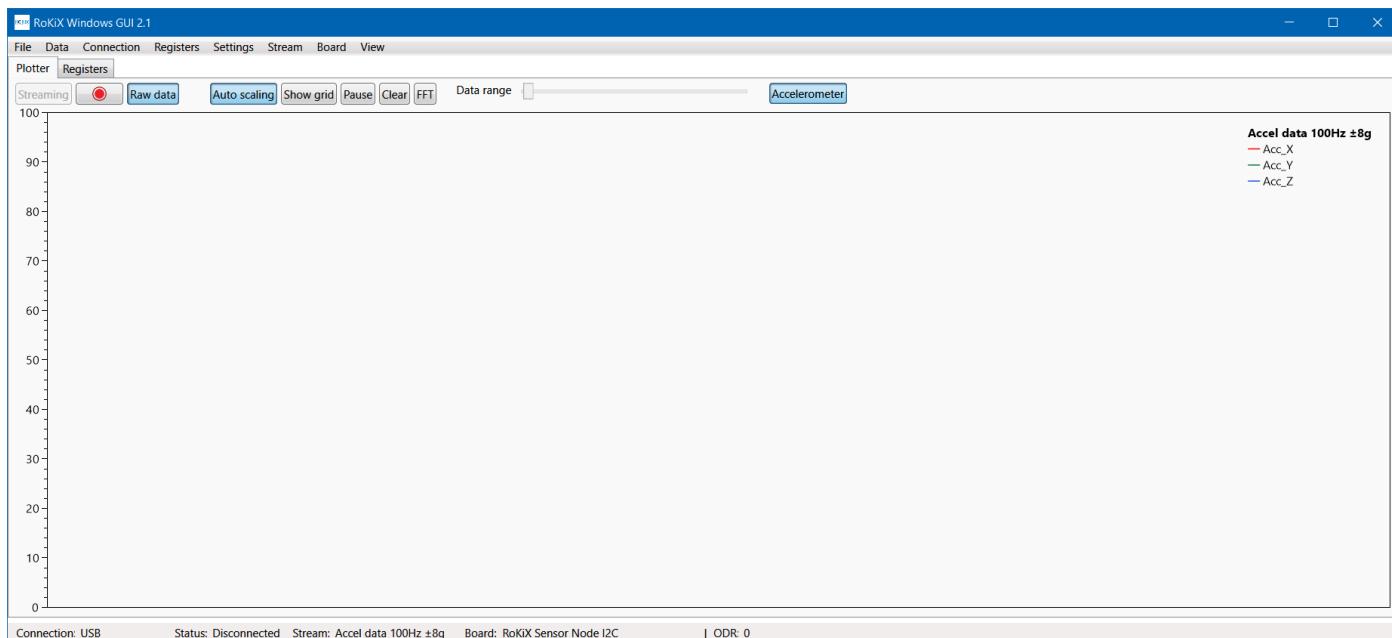
- A visual display of real-time device data
- Ability to record device data onto a file
- Device registry editor
- A demonstration of Rohm and Kionix software offering which includes the Sensor Fusion Algorithm

RoKiX Windows GUI is compatible with Windows OS versions 7, 8 and 10.

5.2. Setup

5.2.1. Installation

The *RoKiX Windows GUI* is located in \Documents\RoKiX\RoKiX-Windows-GUI\RoKiX.exe



Before running the software, the following installation steps may be necessary:

1. When using the *RoKiX Sensor Node* with a USB connection, necessary USB serial drivers (4.1.5) must be installed (if Windows does not install these drivers automatically)
2. When using the *RoKiX Sensor Node* with a Windows Bluetooth connection, pairing is needed (4.2)

The download link to latest the release of *RoKiX Windows GUI*: [Latest release](#)

5.2.2. Configuration

5.2.2.1. Board configuration and connection type

The following settings must be done from the application menu to get the device connected:

- Board configuration from Board-menu 5.4.7
- Connection type from Connection-menu 5.4.3

If these settings are not done properly, the “No data received” pop-up will appear (see section 5.7.1).

5.3. Getting Started

Follow these steps to get started with the *RoKiX Windows GUI* using USB connection and the *RoKiX Sensor Node*:

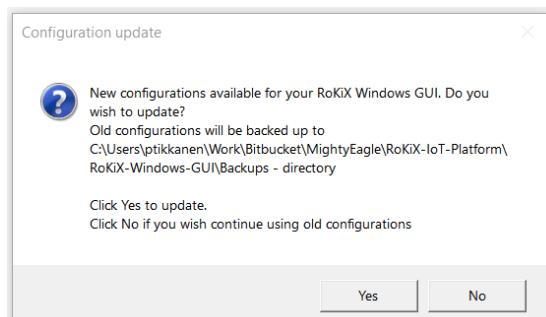
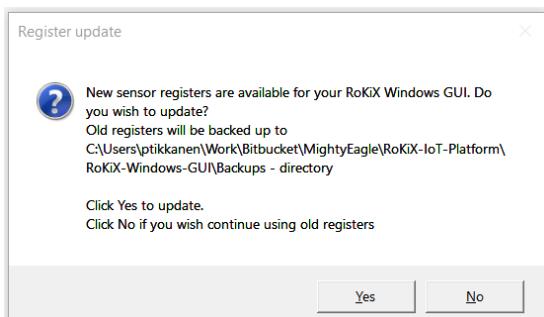
1. Plug in a USB cable between the PC and the *RoKiX Sensor Node*
2. Power on the *RoKiX Sensor Node* by pressing the blue button located on the left side of the node



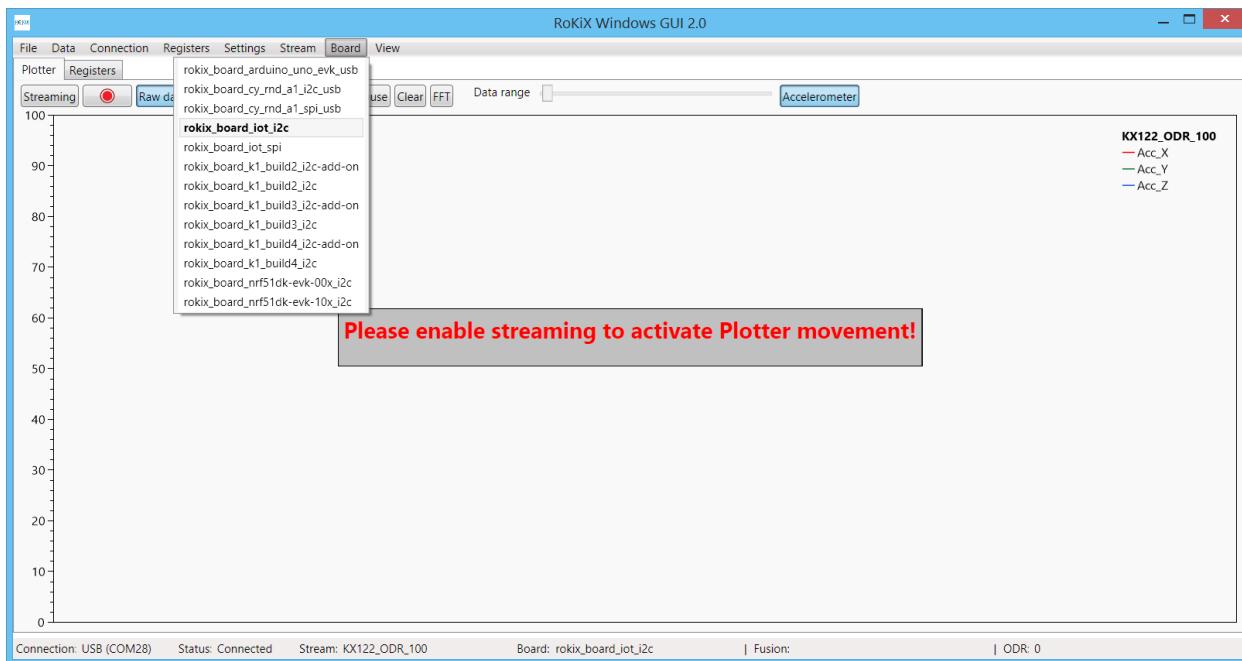
3. Start *RoKiX Windows GUI*



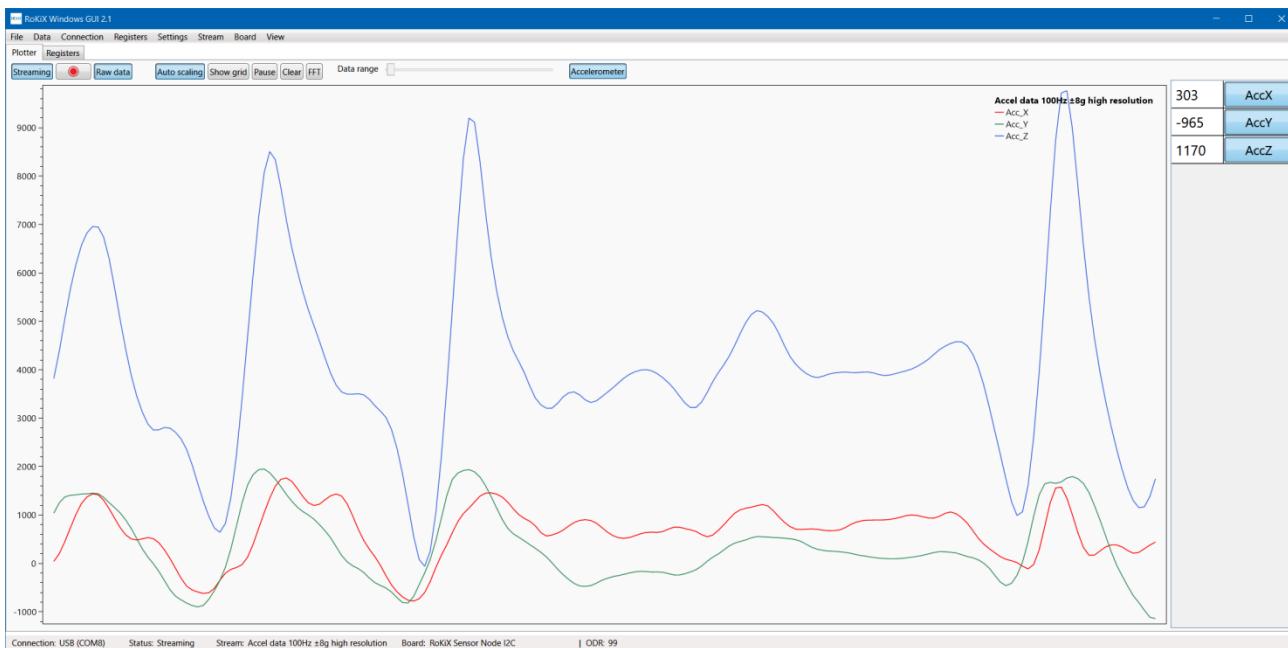
NOTE: At startup, *RoKiX Windows GUI* will suggest downloading the new register and device configuration files whenever they have been changed. The following pop-up menus will appear which also inform where the old register and configuration files will be stored as zipped files:



4. Check that either “RoKiX Sensor Node / I2C” or “RoKiX Sensor Node / SPI” board configuration is selected



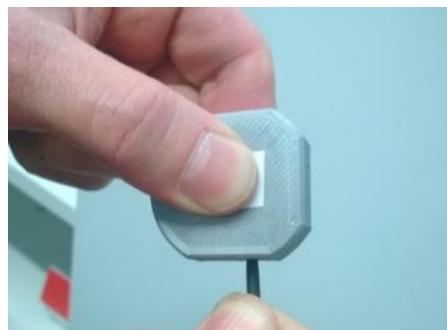
5. When the “Please enable streaming to activate Plotter movement!” – Pop-up window appears on the screen, enable data streaming with “Streaming” button.
6. The Plotter now shows graphs according to your *RoKiX Sensor Node* movements



5.3.1. Getting Started with Kionix IoT Sensor Node

Follow these simple steps to get started with the *RoKiX Windows GUI* using USB connection and the *Kionix IoT Sensor Node*:

If a battery powered board is used, power on the *Kionix IoT Sensor Node* by pressing the power switch.



Plug in a USB cable between the PC and the *Kionix IoT Sensor Node*

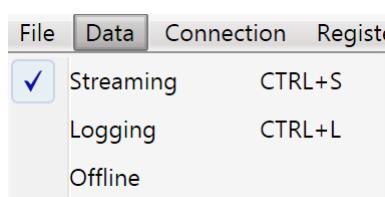
NOTE: When using the *Kionix IoT Sensor Node* on your PC for the first time, Windows will install a USB serial driver for it. It may take a while, please be patient. If the installation does not occur automatically, please refer to section 4.1.5 for installation instructions.

5.4. User Interface – Menu bar

5.4.1. File – Menu

Selecting “Exit” will exit from the application.

5.4.2. Data – Menu



The data menu contains all main options for the device data usage.

5.4.2.1. Streaming

The streaming menu is used for enabling / disabling device data streaming. Shortcut: CTRL + S

NOTE: Data stream enabling / disabling may take a while, please be patient.

5.4.2.2. Logging

The logging menu is used for enabling / disabling device data logging. The status bar will show the log file name.

Logging to log_file.txt

More logging settings are defined in 5.4.5.5.

Shortcut: CTRL + L

NOTE: Offline data is not supported in logging.

5.4.2.3. Offline

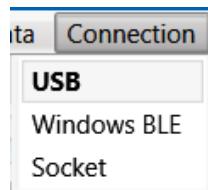
When “Offline” is selected a file dialog will open for choosing the offline device data file to be used. Offline data is mainly used in sensor fusion development and demo purposes. The selected offline data file must have accelerometer + magnetometer + gyro data to be used by the sensor fusion algorithm.

NOTE: Offline mode is not supported in Plotter and logging.

5.4.3. Connection – Menu

The *RoKiX Windows GUI* connects to the *RoKiX Sensor Node* either via USB COM port or via Bluetooth LE.

The connection menu is used to select the desired connection type. *RoKiX Windows GUI* uses USB COM connection by default. When auto-connect is enabled, the USB connection is established automatically when the *RoKiX IoT Platform HW* is connected.



NOTE: “Windows BLE” can be used when the system is running on Windows OS version 8.1 or later.

NOTE: Changing connection may take a while, please be patient.

NOTE: If you are having connection problems, “CTRL + R” can be used to refresh current connection.

5.4.3.1. Windows BLE connection

The Windows BLE connection works only with Windows 8.1 or later versions. Please note that when selecting Windows BLE, the ODR can sometimes be a bit slow. The FAQ section has information about this (section 7.3).

The *RoKiX Sensor Node* must be paired with Windows before BLE connection can be created. This procedure is described in section 4.2.

NOTE: When multiple Bluetooth devices are paired with Windows, you can select the used device in the “Paired BLE devices” – list in the Settings menu



NOTE: In order to disconnect the Windows BLE connection manually, the *RoKiX Sensor Node* device must be turned off.

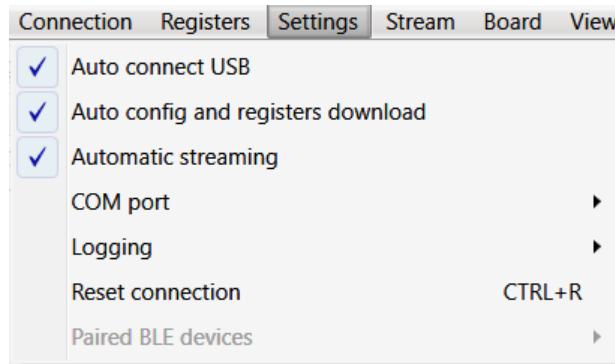
5.4.4. Registers - Menu



5.4.4.1. Load

Selecting “Load” will allow the user to load the device’s register definition file from the “Registers” – tab. Please refer to section 5.5.4.

5.4.5. Settings – Menu



5.4.5.1. Auto Connect

When “Auto connect” is enabled, the *RoKiX Windows GUI* will automatically select the USB COM port for the connected device and connect to it.

5.4.5.2. Auto config download

When “Auto config download” is enabled, the *RoKiX Windows GUI* will automatically check and download the latest board and stream configurations. The user will be notified when there are new configurations available for download.

5.4.5.3. Automatic streaming

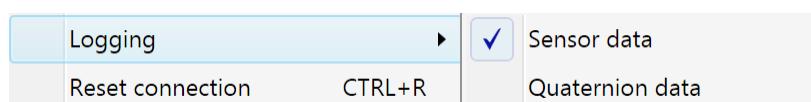
When “Automatic streaming” is enabled, the *RoKiX Windows GUI* will automatically start data streaming when the device stream is changed.

5.4.5.4. COM port

When there are multiple devices connected or there is some problem with the USB COM port selection, the COM port can be selected from the dropdown list. Before doing this the “Auto-connect” feature must be disabled.

5.4.5.5. Logging

Under the logging menu you can define what data is stored to file during device data logging.



NOTE: Logging will always use raw device data instead of SI units.

5.4.5.6. Reset connection

If you are having connection problems, “Reset connection” can be used for refreshing the current connection. It also initializes the current data stream again.

Shortcut: CTRL + R

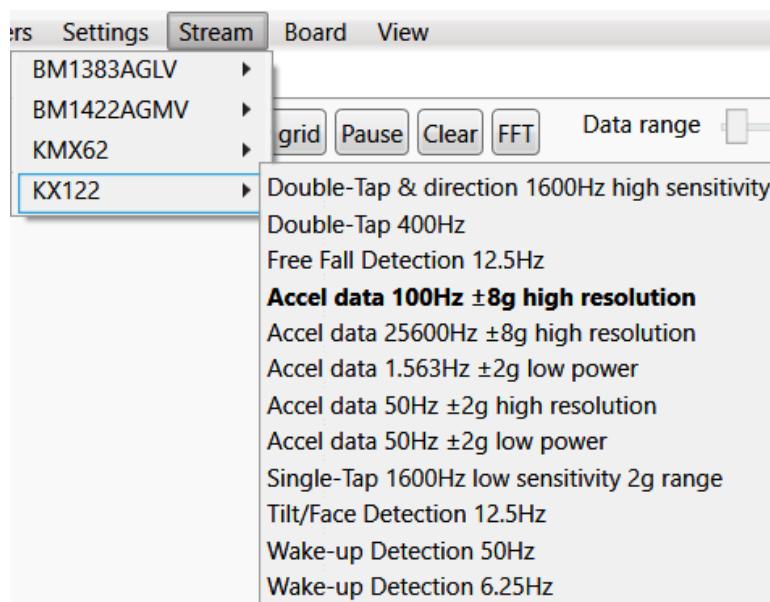
5.4.5.7. Paired BLE devices

This menu will be enabled automatically when “Windows BLE” – connection is taken into use. All paired devices and their MAC addresses are on this list. When multiple devices are paired with Windows, you can select the used device here.

NOTE: When you pair a new device while *RoKiX Windows GUI* is on, please reset the connection in order to refresh the paired devices list.

The *RoKiX Windows GUI* will store the selected settings to the user settings on the computer and will restore them the next time the application is started.

5.4.6. Stream - Menu



The stream menu is used for selecting the device data stream which the application uses to receive data. The list of streams is dynamic and will change according to the chosen board configurations (section 5.4.7).

NOTE: KX122/KX126 have the same I2C slave addresses if the corresponding part is assembled on the *Kionix IoT Sensor Node*.

For example: this means that the application allows the use of KX122 stream even if KX126 is present in the device. Please make sure that the correct board configuration (ref. section 2.2) and stream are selected to avoid misleading device data.

NOTE: The *RoKiX Windows GUI* will store the last used stream configuration and it will be loaded in the next application startup.

5.4.6.1. Sensor Fusion

The Sensor Fusion algorithm to be used is selected automatically according to the used sensor data stream:

- Stream contains accelerometer and gyroscope data, but is missing the magnetometer data
 - Accelerometer + Gyro fusion is selected
- Stream contains accelerometer, magnetometer and gyroscope data
 - Accelerometer + Magnetometer + Gyro fusion is selected
- Stream contains accelerometer and magnetometer data, but is missing the gyroscope data
 - Accelerometer + Magnetometer fusion is selected

This selection affects the operation of Cube 5.5.2, Air mouse 5.5.3, and logging 5.4.5.5.

You can also use key shortcuts to override the sensor fusion mode:

CTRL + F1:	Accelerometer + Gyro
CTRL + F2:	Accelerometer + Magnetometer + Gyro
CTRL + F3:	Accelerometer + Magnetometer

5.4.6.2. Streams with magnetometer

When the magnetometer is in use and the dialog box shown below appears, it is time to calibrate the magnetometer:

Please calibrate the magnetometer!

Calibration is done by performing “8” – like movement with the sensor.

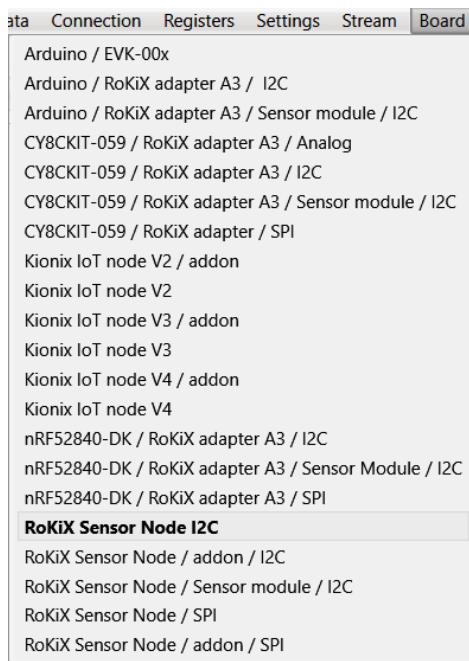
The following levels of the calibration quality are defined:

- Level 0: Not OK, calibration is requested
- Level 1: Satisfactory
- Level 2: Good
- Level 3: Ideal

NOTE: The need for the magnetometer calibration and the stability of the calibration level may vary a lot according to the prevailing magnetic conditions around the sensor.

5.4.7. Board – Menu

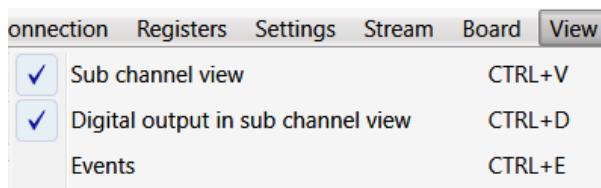
The *RoKiX Windows GUI* supports multiple board configurations:



The type of board that is used can be selected from this menu. The default board configuration is “RoKiX Sensor Node I2C”.

When the board is changed, the connection will be changed to USB automatically.

5.4.8. View – Menu

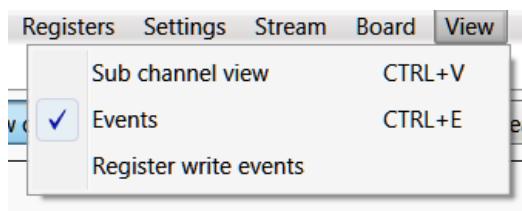


From View – Menu the “Events” – view can be enabled. It will appear on the bottom of the *RoKiX Windows GUI* window.

The “Events” – view will show “Double tap” and “Free fall” events.

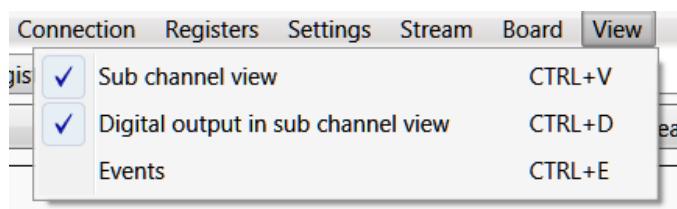


From View – Menu you can enable “Register write events” if “Events” has also been enabled.



When the “Register write events” is enabled, you will see all register write actions which the *RoKiX Windows GUI* is executing in the background.

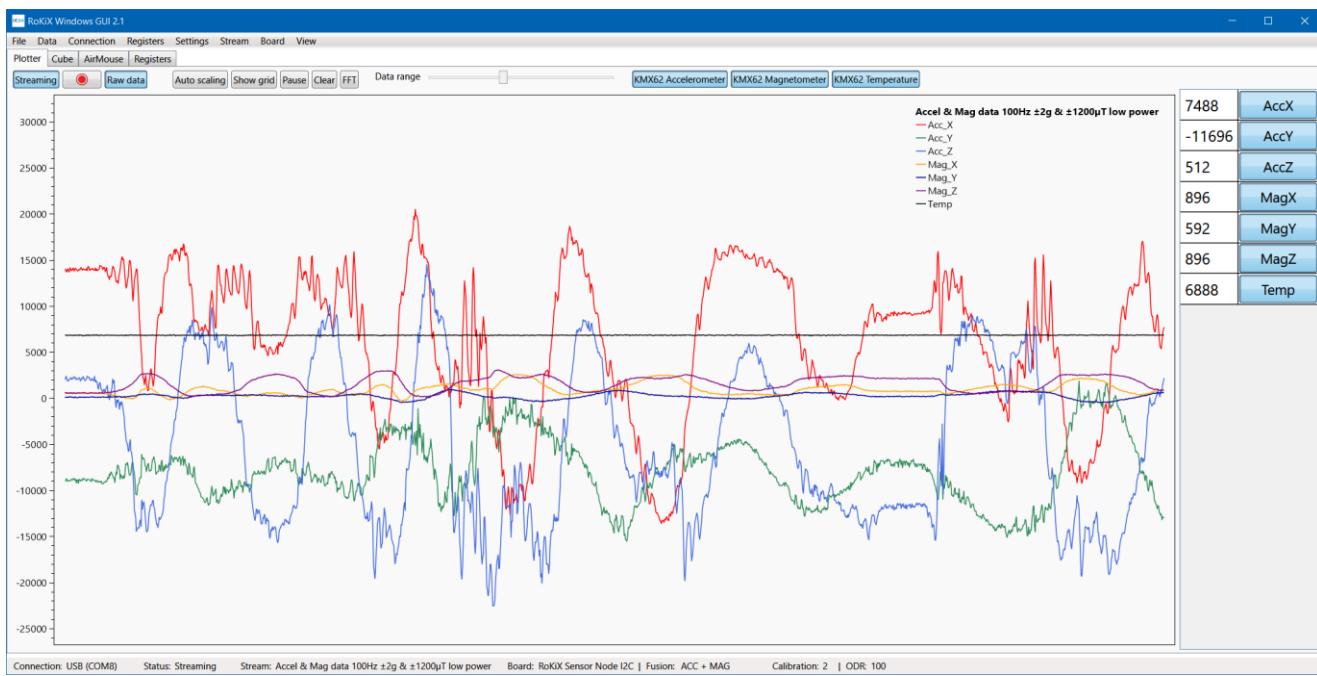
5.4.8.1. Sub-channel view



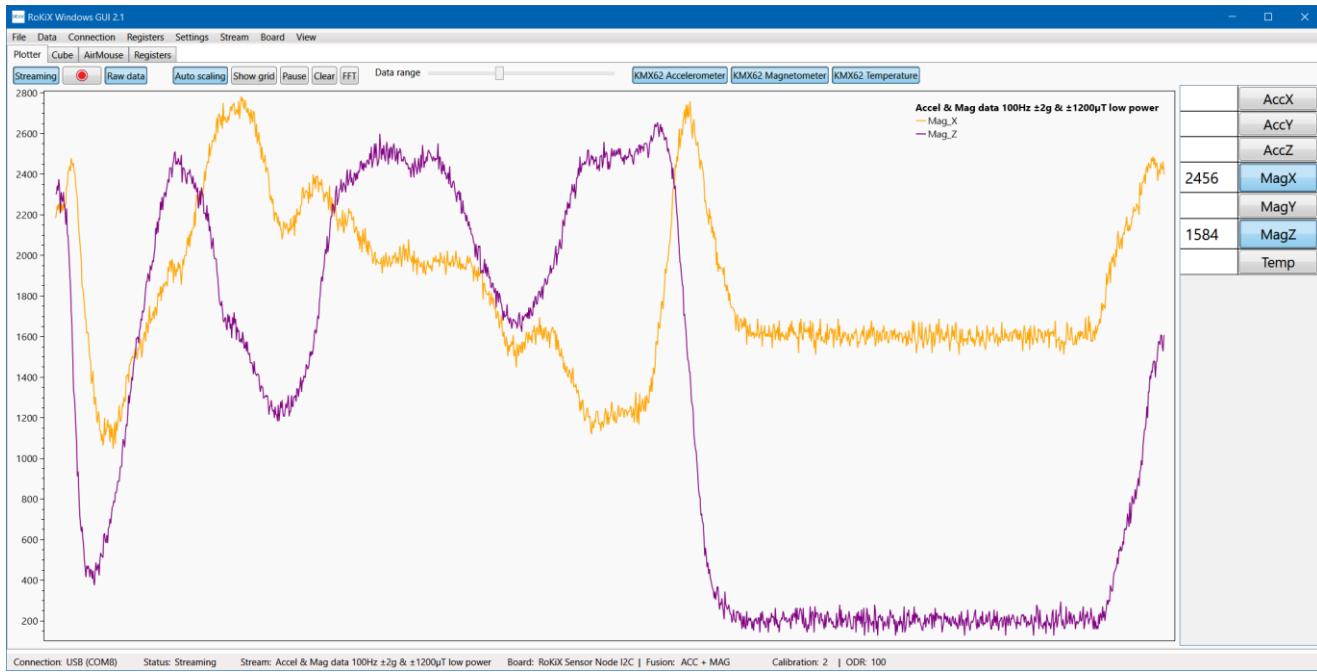
When the “Sub channel view” is enabled, the plotter shows an additional side panel on the right which can be used to show/hide the sub channel view. It also has a column for the digital output of each sub channel which can be enabled with “Digital output in sub channel view”.

NOTE: Available sub channel views are always related to the used device stream and the data channels inside of it.

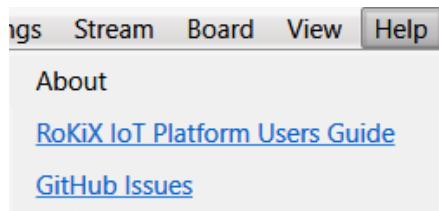
NOTE: When you have a very high ODR (12.8 kHz or 25.6 kHz), the digital output can slow down the plotter’s performance.



For example in this picture only “MagX” and “MagZ” sub-channels are shown:

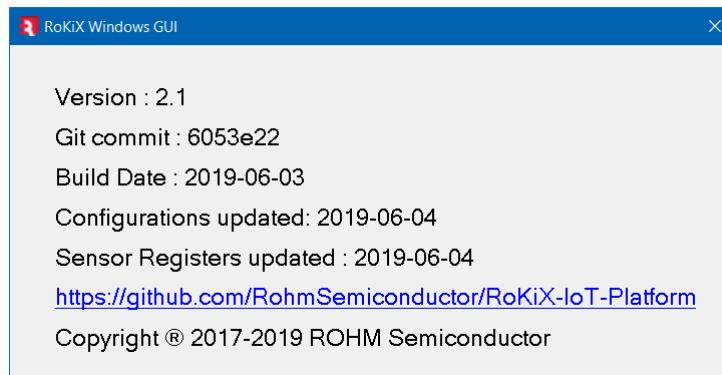


5.4.9. Help – Menu



5.4.9.1. About

The menu shows the *RoKiX Windows GUI*'s version and which Git commit it is from and when it has been built.



5.4.9.2. RoKiX IoT Platform Users Guide

A direct link to the *RoKiX IoT Platform Users Guide* is available for providing guidance.

5.4.9.3. GitHub Issues

A link to the *GitHub Issues* shows information on todos, feature requests, and bugs on the Windows GUI software as well as other RoKiX firmware and software.

5.5. User Interface - Tabs

The functionalities of the *RoKiX Windows GUI* are divided between separate tabs.

5.5.1. Plotter – Tab



The Plotter shows device data from the current stream. The Plotter has its own Streaming and Raw data buttons in order to change them quickly.

Streaming **Raw data** **Auto scaling** **Show grid** **Pause** **Clear** **FFT** **Data range** **KMX62 Accelerometer** **KMX62 Magnetometer** **KMX62 Temperature**

Data logging can be enabled/disabled easily with the button with the red circle icon.

NOTE: When logging is enabled the red circle icon starts to blink.

When Auto scaling is enabled, plotter will auto scale the minimum and maximum values in the y-axis according to the device data.

Show grid – enables data grid lines. The shortcut "G" can also be used for this.

NOTE: This may slow down the plotter's performance.

Pause – pauses the plotter. The shortcut "P" can also be used for this.

Clear – clears all data points from the plotter. The shortcut "C" can also be used for this.

FFT – turns on the "FFT" functionality of the plotter. More information can be found in section 5.5.1.6.

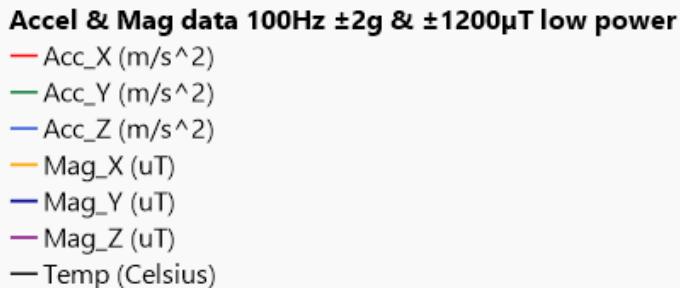
Data range – adjusts the amount of data points shown in the plotter.

NOTE: At high data rates, the slider area of Data Range may flash red, because data is averaged so that the plotter is able to draw the points.

The Plotter has dynamic buttons in order to show/hide data channels within the used device stream. For example in the KMX62_ODR_100 stream there are “Accelerometer”, “Magnetometer” and “Temperature” buttons that can be toggled.

5.5.1.1. Raw data

When Raw data is disabled, SI units for data streams are visible:



5.5.1.2. Zooming

You can zoom in and out using the mouse scroll button or right mouse button+CTRL.

NOTE: When zooming and “Auto scaling” is enabled, the Plotter will no longer perform auto scaling. In order to re-enable “Auto scaling” after zooming, it must be disabled and enabled again.

5.5.1.3. Pausing

You can pause with the “Pause” – button or with the shortcut “P”. The Plotter can also be paused to a certain position using the left mouse button.

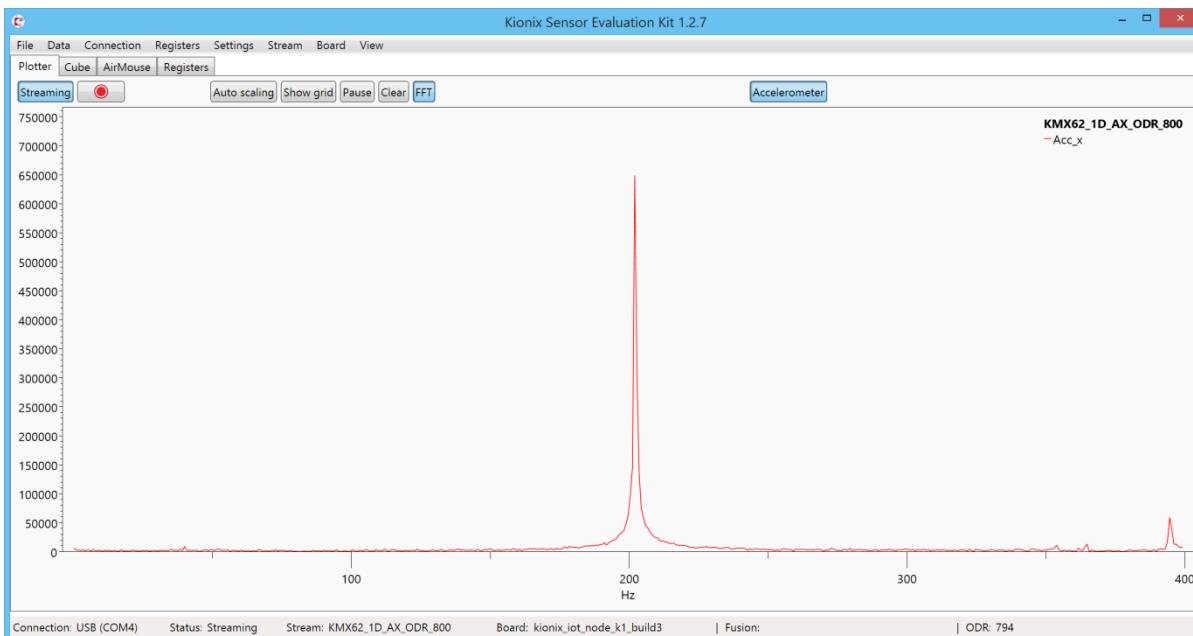
5.5.1.4. Moving

The position of the Data axis (y-axis) can be moved up and down using the right mouse button.

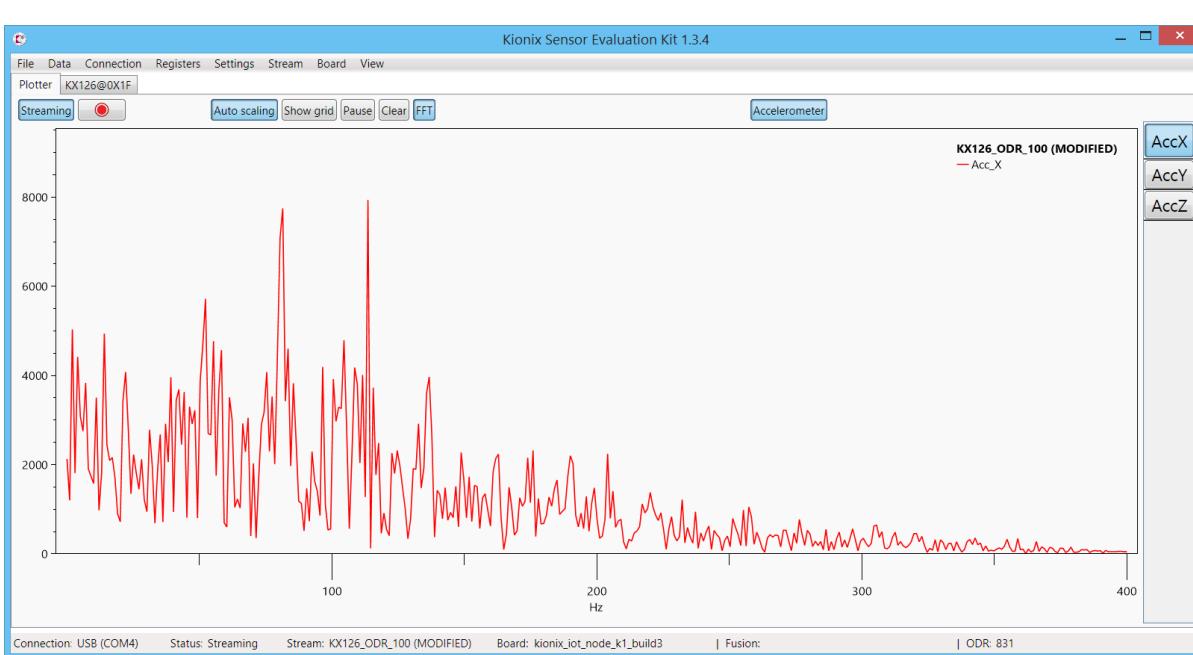
5.5.1.5. Clearing

You can clear all visible data points from the plotter with the “Clear” – button.

5.5.1.6. Frequency analysis



The sub-channel view can be used to show only the wanted sub-channel frequency graph(s).



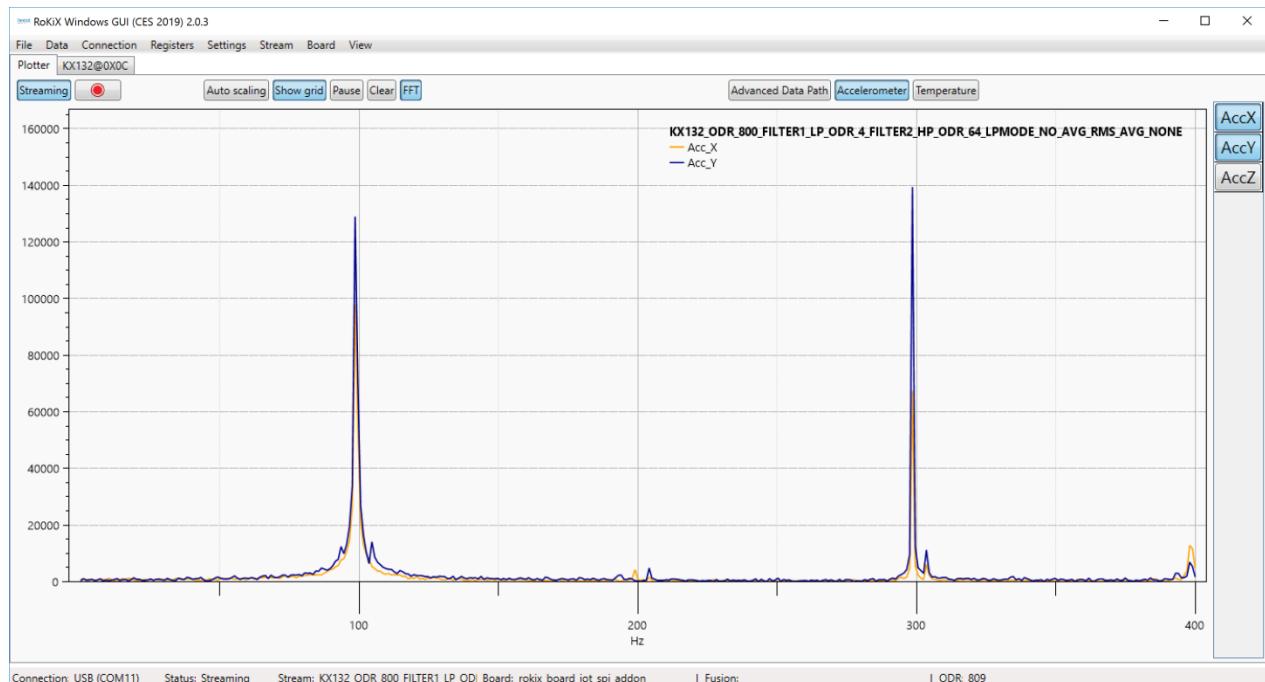
NOTE: The x-axis range starts from 0 and ends in ODR/2. It will be automatically adjusted if the ODR is modified.

5.5.1.7. Advanced Data Path (ADP)

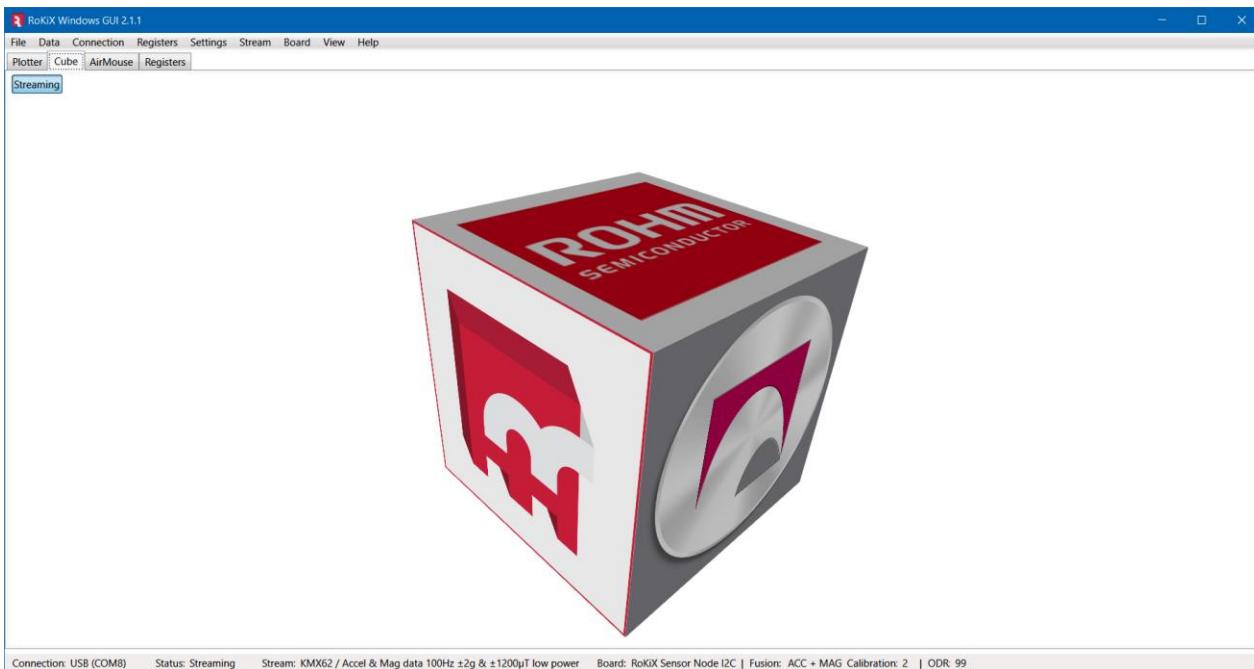


Advanced Data Path (ADP) is a special ASIC level functionality only for KX13x products, which consists of three blocks including a low-pass filter, low-pass/high-pass filter, and an RMS calculation engine. A user can configure each stage of the ADP individually or they can be bypassed with the corresponding register settings using the register editor.

Output of the ADP engine can be monitored with the Plotter by showing device stream data (above) or applying the frequency analysis (FFT) function (below).



5.5.2. Cube – Tab



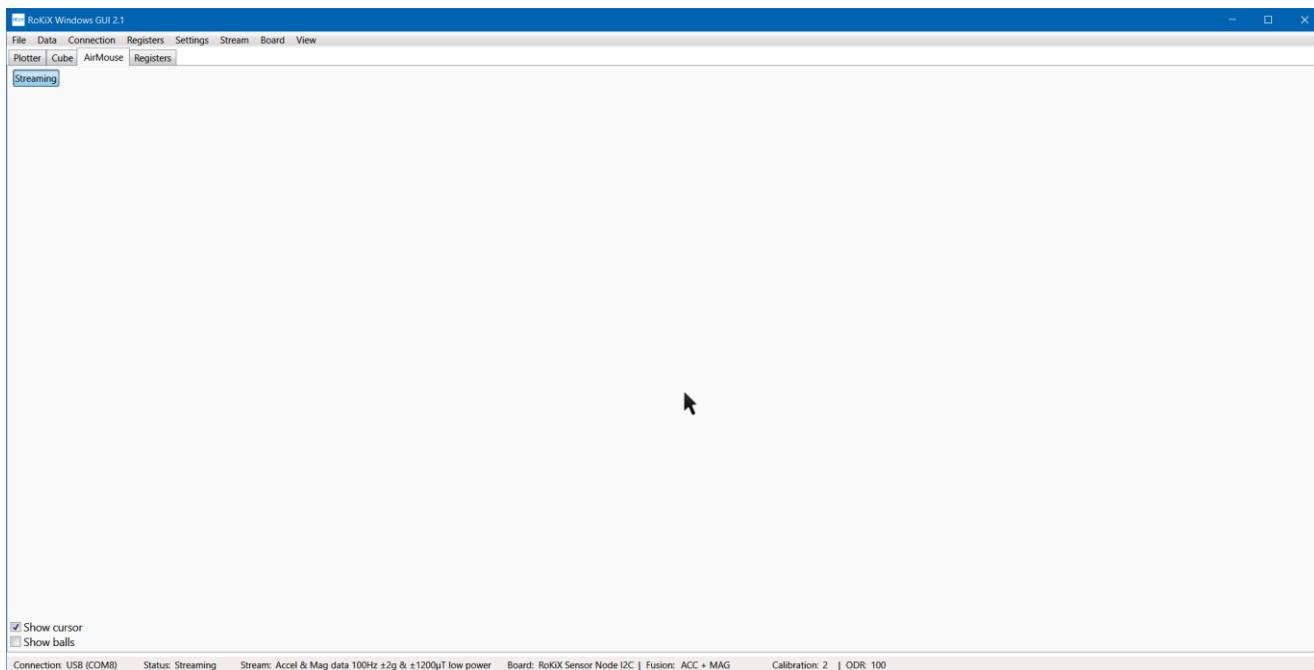
The 3D rotating cube is used for demonstrating the sensor fusion algorithm's performance. Sensor fusion is performed automatically in the background when the data streaming is started by pressing the "Streaming" button.

The Cube tab is visible only when the used data stream supports sensor fusion. It must include one of these channel combinations:

- Accelerometer + Gyroscope
- Accelerometer + Magnetometer + Gyroscope
- Accelerometer + Magnetometer

Please refer to section 5.8.1 to get the rotation operating correctly.

5.5.3. Air Mouse – Tab



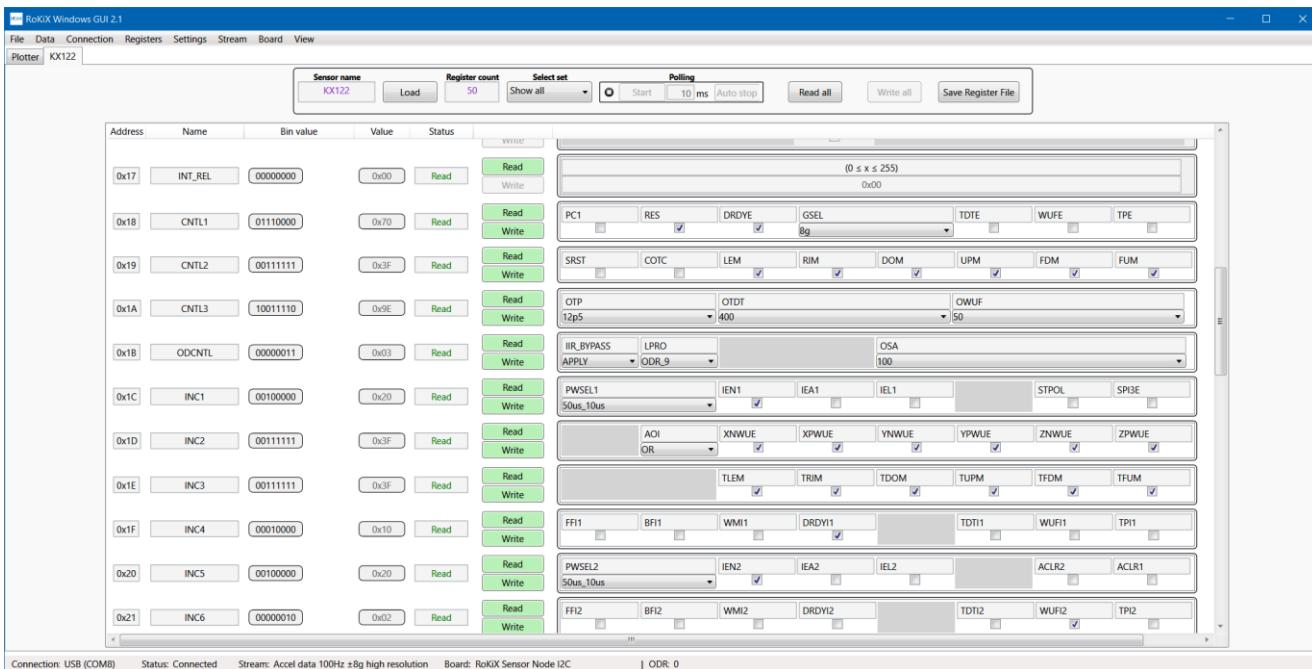
The Air mouse uses sensor fusion for moving the cursor. “Show balls” can be activated to show movement of the x, y and z axis. The Air Mouse can be started by pressing the “Streaming” button.

The Air mouse tab is visible only when the used data stream supports sensor fusion. It must include one of these channel combinations:

- Accelerometer + Gyroscope
- Accelerometer + Magnetometer + Gyroscope
- Accelerometer + Magnetometer

Please refer to section 5.8.2 to get the movement operating correctly.

5.5.4. Registers – Tab



The register editor tab can be used for reading and writing device register values. When you have loaded the device register XML file, the tab name will change to <device name>@<SAD>. If the selected device is not supported in the currently connected board, then “NOT FOUND” – text will be displayed instead of the SAD hex address.

When a new device in the register tab is opened, the register’s content is **not updated automatically**. Instead, register values can be updated with the “Read All” button or “Read” of individual register.

NOTE: Streaming and logging is automatically paused once the Register editor tab is selected.

NOTE: When register values are changed and you return back to the Cube or Air Mouse, it is possible that changes have been made to the register values that had an effect on data streaming. Please change the data channel to any other channel and then change back to the original channel from the Stream-menu (5.4.6) in order to reset the data streaming register values.

5.5.4.1. Register sets

It is also possible to create application specific register sets. The Register set will view only a subset of sensor registers. This helps in using a particular feature of the sensor. These are example settings related to:

- Changing data streaming related parameters: g-range, ODR, etc.
- The testing of ASIC level functionality: Motion wake-up, double tap detection, etc.

5.5.4.2. Register polling function

Register polling is a simple way to monitor register values when the register editor-tab is active. Polling can be activated with a specific interval (the default interval is 10ms). The polling feature has an “auto stop” – checkbox, which stops the polling to the first register value change. The below example shows how to monitor double tap detection:

- Select sensor KX122 by pressing the “Load” button.
- Select set “Double Tap Settings” from the “Select set” pull down menu
- Press the “Read all” button
- Uncheck Power Control (PC1) from register CNTL1 (0x18) and press write
- Check Double Tap Detection Enabled (TDTE) and press write
- Check Power Control (PC1) and press write
- Activate “Auto stop” from “Polling”
- Select set “Interrupts” from the “Select set” pull down menu
- Press the “Start” button from “Polling” (optionally change polling interval to 0)

Once double tap has been detected, the updated values are highlighted in red.

NOTE: Streaming and logging are automatically paused when register polling is started.

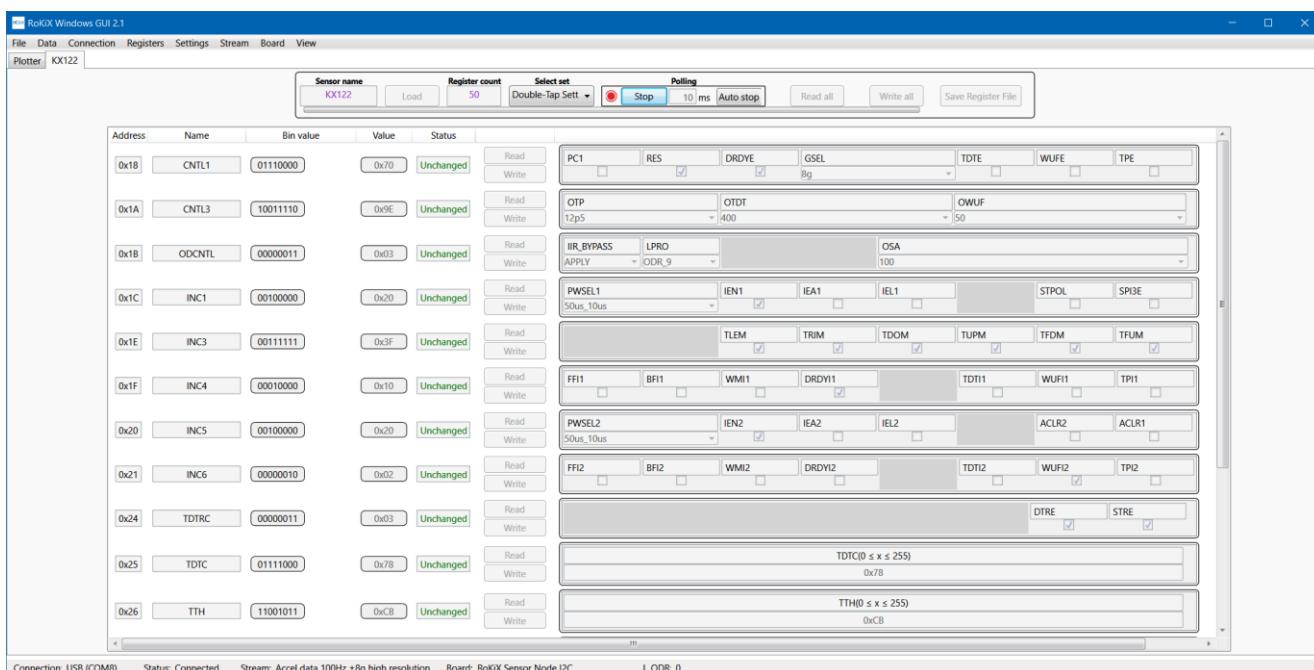


Figure 15: Double tap detection register set.

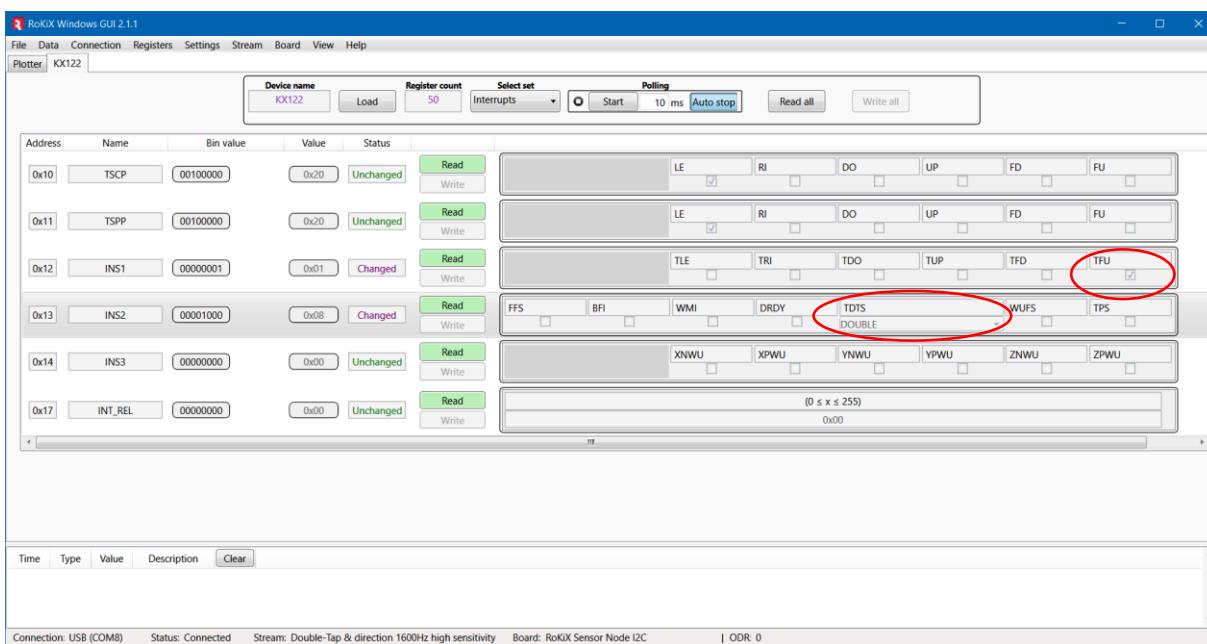


Figure 16: Interrupt registers for Double Tap detection.

5.5.4.3. Stream modify mode

When the Register editor tab is selected while Streaming is enabled, the register editor will enter into a data stream modify mode. This mode offers the user a way to change the registers which will have an effect on the data stream itself. When wanted register changes have been made and the user selects the Plotter, Cube or Air Mouse tab, Streaming will be re-enabled with the changed register values. For example this makes it easy to change the ODR or data range.

NOTE:

- Some devices require that the power control bit (PC) is set to 0 before changing the register values. Otherwise register value changes are not applied. After registers are edited, the PC bit must be set back to 1 to enable the device again.
- In order to restore default register values of the data stream, “Streaming” must be disabled and enabled again.
- Many data streams have their own register sets called “Data stream”. They contain only registers which are directly linked to the used data stream. Thus, this makes it easier to modify data registers related to data reading.

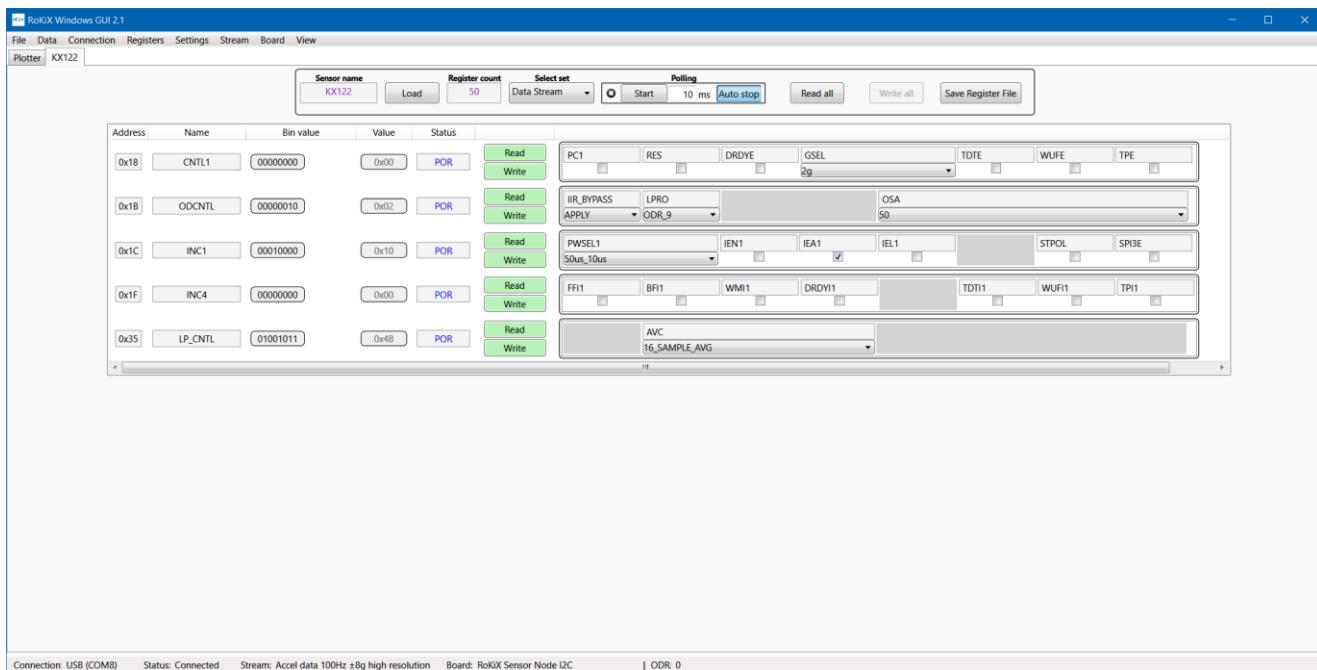


Figure 17: Register editor view.

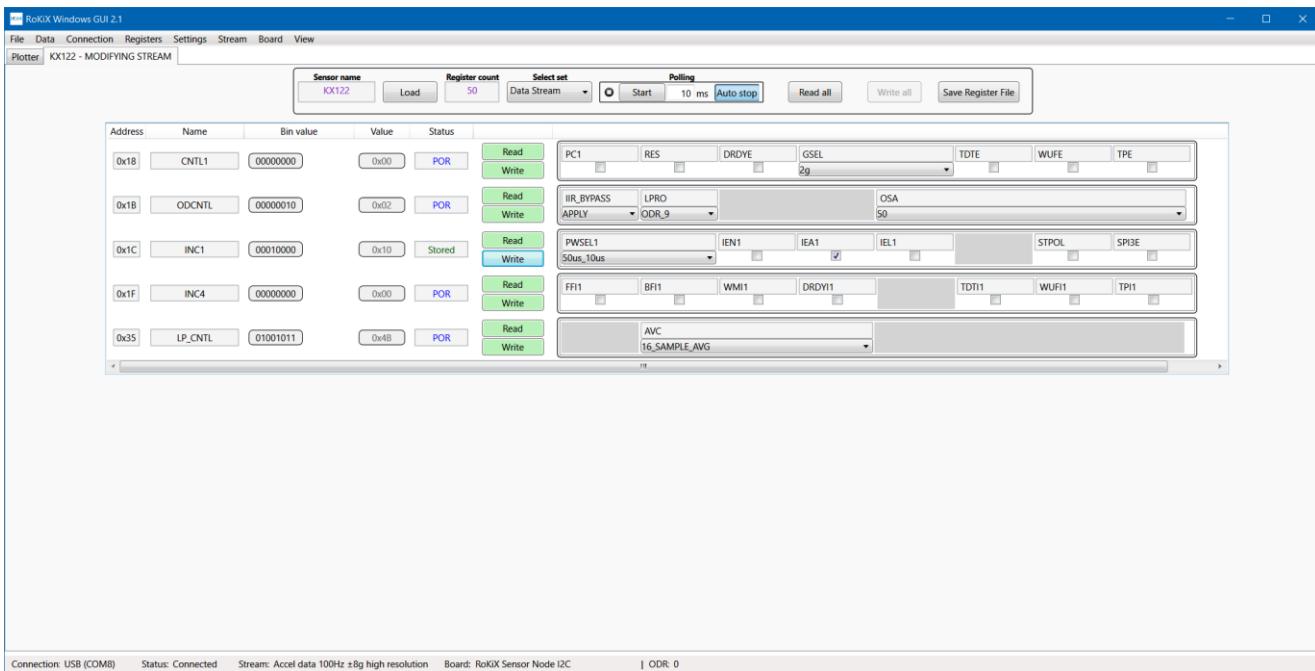


Figure 18: Stream modify mode in the register editor.

When registers have been changed, (MODIFIED) text will be shown after the stream name (Figure 19).

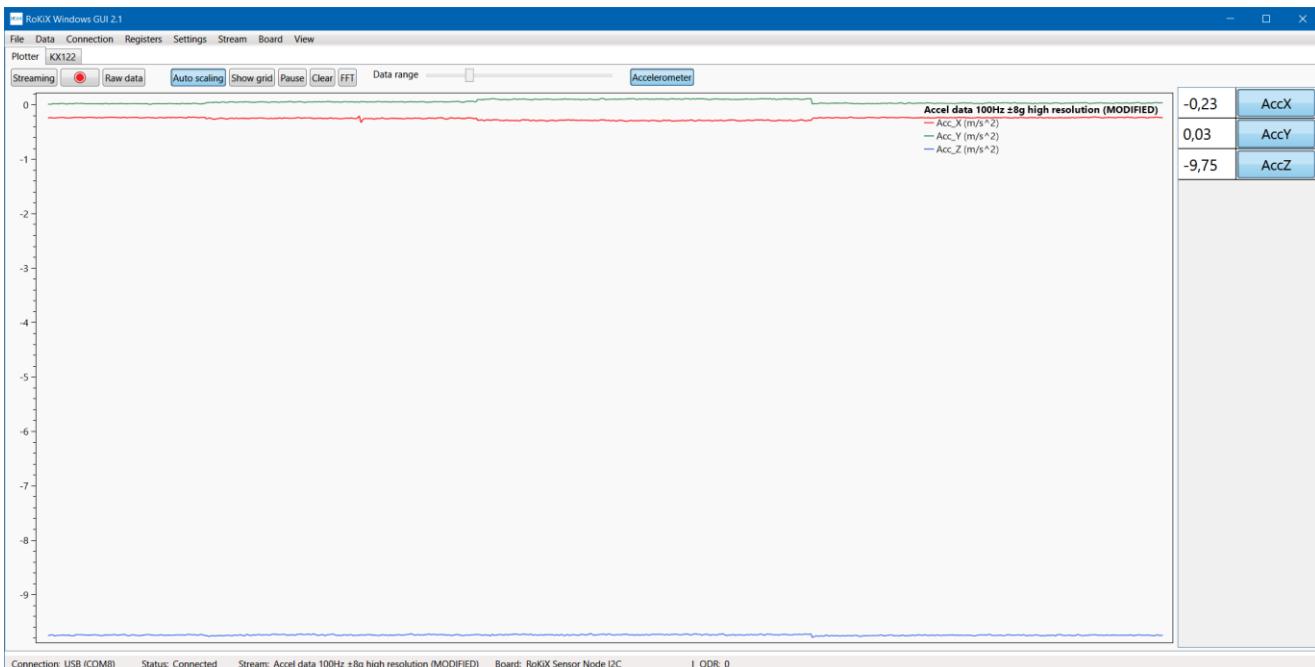


Figure 19: The plotter view after modifying the register values.

5.6. User Interface - Status bar

Connection: USB (COM8) Status: Streaming Stream: Accel data 100Hz ±8g high resolution Board: RoKiX Sensor Node I2C | ODR: 100

The status bar shows the current connection and its status. It also keeps the user updated on the selected stream and board configuration, sensor fusion and magnetometer calibration status.

ODR is visible in the right corner. It can be hidden with the “O” – key shortcut if needed.

NOTE: It is normal to see a slight variation in the ODR value. Data is received at varying intervals and the ODR is calculated when *RoKiX Windows GUI* receives the data from the used connection layer.

5.7. User Interface - Pop-up windows

The application makes use of pop-up windows to notify the user about important actions. This section provides detailed information about pop-up windows.

5.7.1. No data pop-up window

“No data received” pop-up is showed when streaming has been started, but no data is received. The problem could be invalid board configuration selection or some connection problem.

No data received! Please check your board configuration and device functionality.

5.7.2. Streaming pop-up window

Streaming pop-up window is shown in Cube, Plotter, and Air Mouse to notify the user about data stream enabling. Streaming can be enabled with the specific “Streaming” – button, from Data/Streaming – menu or with the shortcut “CTRL + S”.

Please enable streaming to activate Cube movement!

5.7.3. Orientation reset pop-up window

Orientation reset pop-up is shown once per created device connection. It reminds the user to perform device orientation reset in Cube and Air Mouse.

Please reset the orientation with space button!

5.7.4. Magnetometer calibration pop-up window

Magnetometer calibration pop-up is shown in Cube and Air Mouse when a channel including magnetometer is used and the calibration status is 0, see section 5.4.6.2 for more details. The calibration is done by performing “8” – like movement with the sensor.

Please calibrate the magnetometer!

5.8. Orientation reset

5.8.1. Cube reset

5.8.1.1. RokiX Sensor Node

In order to reset the rotating cube position and calibrate it with the position of the *RokiX Sensor Node*, please follow these steps:

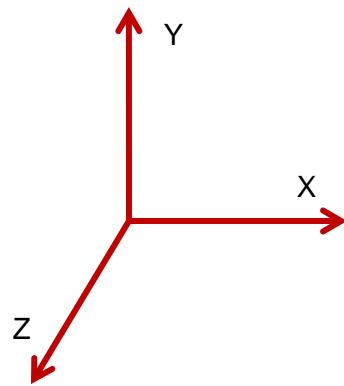
1. Place the *RokiX Sensor Node* in the position where the USB cable connector is pointing upward and the *RokiX Sensor Node*'s front side is pointing towards the Cube
2. Press the “Space” – button on your keyboard
3. The Cube is now reset to the *RokiX Sensor Node*'s position and the *RokiX Windows GUI* logo is on the front side



5.8.1.2. Kionix IoT Sensor Node

In order to reset the rotating cube position and calibrate it with the position of the *Kionix IoT Sensor Node*, please follow these steps:

1. Place the *Kionix IoT Sensor Node* in the position where USB cable connector is pointing downward and status LEDs points to the right
2. Press the “Space” – button on your keyboard
3. The Cube is now reset to the *Kionix IoT Sensor Node*'s position and the Kionix logo is on the front side

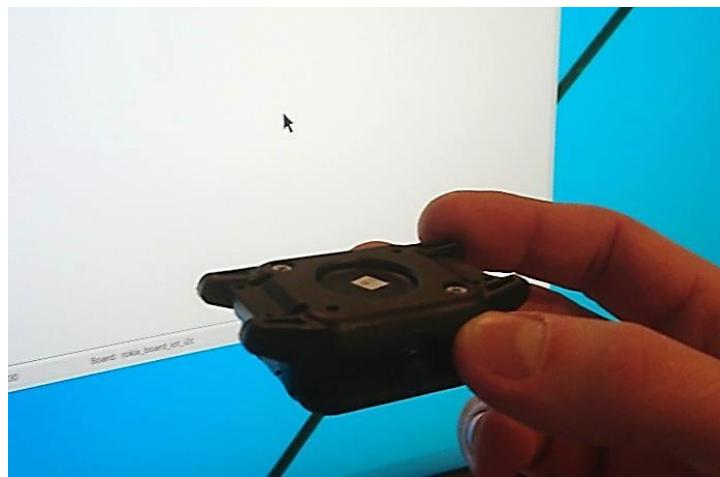


5.8.2. Air Mouse reset

5.8.2.1. RoKiX Sensor Node

In order to reset the Air Mouse position and calibrate it with the position of the *RoKiX Sensor Node*, please follow these steps:

1. Place the *RoKiX Sensor Node* in a position where the USB cable connector points towards the screen, the power button points away from the screen, and the *RoKiX Sensor Node*'s front side is pointed downwards.
2. Press the “Space” – button on your keyboard
3. The Air Mouse is now reset to the *RoKiX Sensor Node*'s position



5.8.2.2. Kionix IoT Sensor Node

In order to reset the Air Mouse position and calibrate it with the position of the *Kionix IoT Sensor Node*, please follow these steps:

1. Place the *Kionix IoT Sensor Node* in a position where the USB cable connector points away from the screen, the power button points towards the screen, and status LEDs point to the right

2. Press the “Space” – button on your keyboard
3. The Air Mouse is now reset to the *Kionix IoT Sensor Node*'s position



5.9. Shortcuts

The *RoKiX Windows GUI* has many keyboard shortcuts:

CTRL + F1:	Activate Accelerometer + Gyro sensor fusion mode
CTRL + F2:	Activate Accelerometer + Magnetometer + Gyro sensor fusion mode
CTRL + F3:	Activate Accelerometer + Magnetometer sensor fusion mode
CTRL + L:	Enable/disable logging
CTRL + S:	Enable/disable streaming
ALT + R + L:	Load sensor register XML file
CTRL + R:	Reset used connection and data streaming (disconnect and connect when having connection problem)
CTRL + E:	Show events view
CTRL + V:	Show sub channel view
CTRL + D:	Show digital output in sub channel view
+ C:	Clears the current points in plotter view
+ G:	Shows the grid in the plotter
+ P:	Pause plotter
+ O:	Hide/show ODR in status bar

6. RoKiX Python CLI

The *RoKiX Python CLI* offers versatile access to the device register level functionality. Simple and minimalist demonstrator applications are provided for testing a variety of device features. These will help to test and evaluate device functionality and provide needed information on how to implement device driver SW.

6.1. Installation for Windows OS

The *RoKiX IoT Platform SW* Windows installer includes the *RoKiX Python CLI* and it will be installed to the following directory: \Documents\RoKiX\RoKiX-Python-CLI. Additionally, the installer creates start menu items for the *RoKiX Python CLI*:



Section 6.3 describes the installation processes of the necessary Python environment. The following URL will always point to the latest version of the *RoKiX Python CLI*:

<https://github.com/RohmSemiconductor/RoKiX-IoT-Platform/releases/latest>

6.2. Installation for Linux and OS X

On the Linux system, the latest version of the *RoKiX Python CLI* can be downloaded with the following command:

```
wget https://github.com/RohmSemiconductor/RoKiX-IoT-Platform/releases/latest
```

Currently, Bluetooth usage with the *RoKiX Python CLI* is not supported when running on Apple OS X. Connecting via USB is the default.

See section 6.3 below for installation of the Python environment.

6.3. Python Set Up

A [Python](#) interpreter is needed to run the *RoKiX Python CLI*. Currently, Python versions above 3.0 are recommended for use (e.g., 3.7.1). As usual, a separate Python environment is recommended to bring into use, the instructions below will guide you through the required installation procedure.

Windows installation

For your Windows installation, please download the Anaconda distribution of Python:

<https://www.anaconda.com/download/>

and follow the instructions:

<http://docs.anaconda.com/anaconda/install/windows/>

After the installation is complete, please create a new virtual environment:

```
conda create -n rokix_env python
```

Take the rokix_env environment into use,

```
activate rokix_env
```

With Windows OS; if automatic USB serial port detection is used in the *RoKiX Python CLI* (by default this feature is active in `rokix_settings.cfg`, refer to section 6.4.1), the following modules (dependencies) are also needed:

```
python -m pip install -r requirements_windows.txt
```

To start using the virtual environment do the following:

```
activate rokix_env
cd kx022
python kx022_data_logger.py
```

If the *RoKiX Sensor Node* is used with a USB connection and CDC ACM USB drivers are not already installed, refer to section 4.1.1.

Linux installation

On Ubuntu the Python virtual environment is installed as follows:

```
sudo apt-get install python-virtualenv
```

Next, create a virtual environment,

```
virtualenv rokix_env
```

A new virtual environment is activated with the command

```
source rokix_env/bin/activate
```

and the required Python dependencies are installed with

```
python -m pip install -r requirements_linux.txt
```

To start using the virtual environment do the following:

```
source rokix_env/bin/activate
cd kx022
python kx022_data_logger.py
```

On some Linux system's (e.g. Ubuntu 18.04) serial ports may not be accessible for the user by default. If you encounter the problem "Permission denied: '/dev/ttyACM0'" while running data loggers, please refer to section 7.4 for details.

If automatic USB serial port detection is not used, the port settings to the `rokix_settings.cfg` file are used as an example:

```
serial_port = /dev/ttyUSB0
```

Ports of the connected devices can be obtained with the following command

```
ls -la /dev/serial/by-id
```

OS X installation

By default Apple OS X is missing the Python package manager "pip". It is necessary to install the required python packages. The easiest way to install `pip` is by opening the terminal and using the command

```
sudo easy_install pip
```

After `pip` is installed, please follow the instructions on the previous section for Linux install.

Optional installations

For both Windows OS and Linux, dependencies required by `plot.py` are installed with

```
python -m pip install -r requirements_plot.txt
```

In addition, dependencies related to the cloud based services are obtained by running

```
python -m pip install -r requirements_cloud.txt
```

Hereafter, to get or update the board configurations please run:

```
python get_configs.py
```

Board configurations will be placed in the directory `\Documents\RoKiX\RoKiX-Python-CLI\cfg`.

6.4. Configuration

The *RoKiX Python CLI* settings are in the `rokix_settings.cfg` file located in the root directory of *RoKiX Python CLI*. The settings file uses windows `.ini` file syntax. Semicolon ";" at the beginning of the line indicates a comment.

There are settings for both:

- *RoKiX Python CLI* framework and
- Test applications included in the *RoKiX Python CLI*

6.4.1. Connection to RoKiX IoT Platform HW

The most important framework level setting is board selection. It defines the board configuration which is used. The definition can be found in the section [board].

```
[board]
; Board selection

; RoKiX IoT Board, main board only
;board=rokix_board_rokix_sensor_node_i2c.json
;board=rokix_board_rokix_sensor_node_spi.json

; RoKiX IoT Board, with addon board
board=rokix_board_rokix_sensor_node_i2c_addon.json
;board=rokix_board_rokix_sensor_node_rohm5.json
;board=rokix_board_rokix_sensor_node_spi_addon.json

...
; Cypress CY8KIT059 development board with a RoKiX Adapter Board A3.
;board=rokix_board_cy8ckit059_analog_a3.json
;board=rokix_board_cy8ckit059_i2c_a3.json
;board=rokix_board_cy8ckit059_i2c_a3_rohm5.json
;board=rokix_board_cy8ckit059_spi_a3.json

...
; Other
; 8MHz spi example config
;board=rokix_board_nrf52840dk_example_spi.json

;board=linux

...
```

Key bus2 settings defines the connection type for the interface 2. Default is USB.

```
[bus2]
; bus2 selection for board
; NOTE: bus2=USB setting works for both USB_SERIAL and USB_AARDVARK if only one of
those is connected
bus2=USB
;bus2=BLE_PYGATT

;bus2=USB_SERIAL
;bus2=USB_AARDVARK

;bus2=BLE
```

The corresponding COM port can be written in the `rokix_settings.cfg` file:

```
[bus2]
; USB serial COM port number or 'auto' for autodetection
serial_port = COM10
```

The default connection is set to automatic port search:

```
[bus2]
serial_port = auto
```

On Linux the latest connected USB device can also be seen with the command dmesg.

When using BLE connection with Windows 8/10 please apply the pairing instructions given in section 4.2.

6.4.2. Generic settings

The [generic] section contains connection independent settings. These include debug logging settings and interrupt related settings.

The debug logging level is defined with the `logging_level` and the default level is `INFO`:

```
logging_level = INFO
```

Other alternatives are: `DEBUG`, `INFO`, `WARNING`, `ERROR` and `CRITICAL`.

Interrupt related configurations are explained in the following, there are four options for data ready operations:

<code>drdy_function_mode</code>	= <code>ADAPTER_GPIO1_INT</code>
<code>;drdy_function_mode</code>	= <code>ADAPTER_GPIO2_INT</code>
<code>;drdy_function_mode</code>	= <code>REG_POLL</code>
<code>;drdy_function_mode</code>	= <code>TIMER_POLL</code>

The default setting used for `drdy_function_mode` is `ADAPTER_GPIO1_INT`.

The interval time for `TIMER_POLL` is defined in `drdy_timer_interval` and the default value is set to 0.04 s.

When the `REG_POLL` option is used the data ready bit is monitored from the status register to trigger device data reading. This will introduce more traffic on the bus and thus reduce ODR. This will also verify that all data is read from the device. Physical interrupt lines are not used in this option. Table 9 summarizes the available modes to acquire data with the *RoKiX Python CLI*.

Data ready operations for ASIC level feature events are defined by the following setting:

```
; This setting defines how asic feature event works in applications where the features
is enabled
; REG_POLL / TIMER_POLL / ADAPTER_GPIO1_INT / ADAPTER_GPIO2_INT
other_function_mode = ADAPTER_GPIO2_INT

; if other_function_mode is TIMER_POLL or polling_mode is TRUE, use this value as in-
terval to poll
other_timer_interval = 0.04
```

6.5. Getting Started

The default configuration is `rokix_board_iot_i2c` with USB connection, e.g. the *RoKiX Sensor Node* is connected to the PC with a USB cable (USB driver installation described in section 4.1.1).

Setting for stream_mode	Setting for drdy_function_mode	Physical interrupt lines	Measurement mode with data ready operation	Notes
TRUE	TIMER_POLL	Connections are not used.	Asynchronous reading	Firmware polls a data register with interval. Use drdy_timer_interval to configure the interval period.
TRUE	ADAPTER_GPIO1_INT, ADAPTER_GPIO2_INT	Use device's interrupt lines.	Synchronous reading	Uses GPIO line 1 or 2 for the firmware interrupt. The fastest option and suitable for logging with higher ODR's. Default set to ADAPTER_GPIO1_INT
FALSE	REG_POLL	Connections are not used.	Synchronous reading	A client monitors the DRDY bit, increases traffic in the bus. The slowest mode.
FALSE	TIMER_POLL	Connections are not used.	Asynchronous reading	A client polls a data register with interval. Use drdy_timer_interval to configure the interval period.
FALSE	ADAPTER_GPIO1_INT, ADAPTER_GPIO2_INT	Use device's interrupt lines.	Synchronous reading	Needs to configure GPIO numbers. A client polls either GPIO line 1 or 2.

Table 9: Data acquisition modes within the *RoKiX Python CLI*.

6.6. File structure of the evaluation kit

The overall structure of the *RoKiX Python CLI* is the following:

```
RoKiX-Python-CLI/
__init__.py
get_configs.py
LICENSE.txt
plot.py
requirements_cloud.txt
requirements_linux.txt
requirements_plot.txt
requirements_windows.txt
rokix_settings.cfg
stream_logger.py
├── bh1383aglv
├── bh1422agmv
├── bh1749nuc
├── bh1792glc
├── bm1383aglv
├── bm1422agmv
├── cfg
├── examples
├── kmx62
├── kx_lib
├── kx022
├── kx126
├── kx132
├── kx224
├── kxr94
└── kxtj3
└── license
```

Each device has its own directory containing:

- Register definitions (for example `kx022_registers.py`)
- Reference driver implementation (for example `kx022_driver.py`)
- Test application for reading device data (for example `kx022_data_logger.py`)

`kx_lib` directory contains *RoKiX Python CLI* middleware.

6.7. Running test applications

Test applications are device specific and located in their corresponding folders. For example: the data logging application for KX022 is located in `\Documents\RoKiX\RoKiX-Python-CLI\kx022` and the following command shows the available arguments:

```
cd kx022
python kx022_data_logger.py --help
```

```
usage: kx022_data_logger.py [-h] [-b BOARD]
                            [--bus2 {USB,BLE_PYGATT,USB_SERIAL,USB_AARDVARK,BLE}]
                            [--serial_port SERIAL_PORT] [--ble_mac BLE_MAC]
                            [--drdy_function_mode {ADAPTER_GPIO1_INT,ADAPTER_GPIO2_INT,REG_POLL,TIMER_POLL}]
                            [--drdy_timer_interval DRDY_TIMER_INTERVAL]
                            [--other_function_mode {ADAPTER_GPIO1_INT,ADAPTER_GPIO2_INT,REG_POLL,TIMER_POLL}]
                            [--other_timer_interval OTHER_TIMER_INTERVAL]
                            [-s STREAM_MODE]
                            [--max_timeout_count MAX_TIMEOUT_COUNT] [-o ODR]
                            [-l LOOP] [--filename FILENAME]

Example: kx022_data_logger.py -s

optional arguments:
  -h, --help            show this help message and exit
  -b BOARD, --board BOARD
                        Board configuration file;
                        Examples:rokix_board_rokix_sensor_node_i2c.json,
                        rokix_board_rokix_sensor_node_i2c_addon.json;
                        Default:"rokix_board_rokix_sensor_node_i2c_addon.json"
  --bus2 {USB,BLE_PYGATT,USB_SERIAL,USB_AARDVARK,BLE}
                        Bus2 connection; Default:"USB"
  --serial_port SERIAL_PORT
                        Serial COM port to use; Default:"auto"
  --ble_mac BLE_MAC    Mac address for bluetooth connection. Full address (or
                        first bytes with windows); Default:"None"
  --drdy_function_mode {ADAPTER_GPIO1_INT,ADAPTER_GPIO2_INT,REG_POLL,TIMER_POLL}
                        This setting defines how data ready function works in
                        *_data_logger.py application;
                        Default:"ADAPTER_GPIO1_INT"
  --drdy_timer_interval DRDY_TIMER_INTERVAL
                        Dataready poll interval in seconds if
                        drdy_function_mode is TIMER_POLL; Default:"0.04"
  --other_function_mode {ADAPTER_GPIO1_INT,ADAPTER_GPIO2_INT,REG_POLL,TIMER_POLL}
                        This setting defines how asic feature event works in
                        applications where the features is enabled;
                        Default:"ADAPTER_GPIO2_INT"
  --other_timer_interval OTHER_TIMER_INTERVAL
                        Other function such as wakeup status poll interval in
                        seconds if other_function_mode is TIMER_POLL;
                        Default:"0.04"
  -s STREAM_MODE, --stream_mode STREAM_MODE
                        Streaming mode on/off, 1/0, True/False; Default:"True"
  --max_timeout_count MAX_TIMEOUT_COUNT
                        ; Default:"1"
  -o ODR, --odr ODR    Output data rate; Default:"25.0"
  -l LOOP, --loop LOOP Number of data samples; Default:"None"
  --filename FILENAME  File name with or without extension. If no extension
                        is given .csv is used; Default:"None"
```

The output of the application is the following (CTRL+C will stop the application):

```
python kx022_data_logger.py
```

```
# Log File Format Version = 1.0
# Stream Configuration File = kx022_data_logger.py
# Start time = 2019-05-17 13:30:54:000
# timestamp; ch; ax; ay; az
0.004858; 48; -4586; 17795; -22909
0.032338; 48; -8338; 14736; -28171
0.072752; 48; -5609; 16263; -26090
0.112976; 48; -1227; 16673; -16750
0.153179; 48; 1091; 7708; -7403
0.185394; 48; 1393; -1186; -2936
0.226848; 48; -6; 689; -5417
0.271864; 48; -2904; 6204; -13788
0.312083; 48; -5478; 8928; -22655
0.352290; 48; -4416; 10788; -23590
0.392667; 48; -1287; 7438; -12797
0.424968; 48; 31; -735; -3756
0.465120; 48; -1494; -3231; -8201
0.511567; 48; -3805; -903; -17036
0.551828; 48; -4459; 392; -20016
0.592192; 48; -3736; -640; -16824
0.632210; 48; -3372; -2846; -13388
0.664579; 48; -3918; -3465; -13776
# End time = 2019-05-17 13:30:55:000
```

Device data can also be directed to a file with a specified name:

```
python kx022_data_logger.py > kx022.csv
```

Another way to provide the data logger file name is to write in the `rokix_settings.cfg`:

...

```
# if not left empty then sensor data will be written to file
filename=kx022.csv
```

...

In addition, *RoKiX Python CLI* offers ASIC level features (e.g. WakeUp function) that can be used from the commandline:

```
python kx022_data_wu_logger.py
```

```
# Log File Format Version = 1.0
# Stream Configuration File = kx022_data_wu_logger.py
# Start time = 2019-05-17 13:21:06:000
# timestamp; ch; ax; ay; az; stat
0.007644; 48; -1648; 1862; -16234; 0
0.040362; 48; -1648; 1874; -16228; 0
0.120815; 48; -1640; 1880; -16216; 0
0.193049; 48; -1631; 1881; -16207; 0
0.273587; 48; -1596; 1865; -16362; 0
0.360170; 48; -2456; 1626; -19223; 0
0.433600; 48; -2420; 765; -23211; 0
0.513655; 48; -699; -412; -17271; 0
0.600064; 48; -1920; -3414; -7842; 0
0.673171; 48; -3099; -4618; -1526; 0
0.753193; 48; -3971; -4146; -6848; 0
0.839990; 48; -3326; -5818; -15704; 16
0.920519; 48; -27; -7401; -16467; 0
0.992910; 48; 2002; -8478; -14560; 0
1.079613; 48; 3626; -9317; -7077; 0
1.160118; 48; 4886; -10390; -951; 16
1.232754; 48; 3236; -9030; 2187; 0
1.312820; 48; 1726; -7230; -5595; 0
1.399455; 48; 1014; -6322; -13978; 0
1.472538; 48; -1566; -3680; -15756; 16
1.552648; 48; -2037; -2196; -18380; 0
1.639187; 48; -576; -2924; -20141; 0
1.720108; 48; 434; -3812; -19884; 0
1.792864; 48; 1671; -5217; -12915; 16
1.873120; 48; 4743; -7838; -4097; 0
1.959666; 48; 6262; -9314; 1108; 0
2.032730; 48; 4882; -8114; 1061; 0
2.112644; 48; 4333; -6837; -6165; 16
2.199352; 48; 4838; -6647; -17026; 0
2.279954; 48; 4343; -5619; -23019; 0
2.352569; 48; 876; -3696; -17307; 0
# End time = 2019-05-17 13:21:08:000
```

6.7.1. Other provided tools

With all the Python-based scripts, a command line option `--help` shows the available arguments.

The directory `RoKiX-Python-CLI/examples` provides a few tools and example software algorithms, for example:

- `kx022_register_dump.py`
- `rokix_board_version_info.py`
- `rokix_sensor_node_gpio.py`
- `tilt_algorithm.py`

In the root directory RoKiX-Python-CLI/ there are a few of software tools available:

- `get_configs.py` is a Python script for downloading and extracting new board configurations. The board configurations will be placed to RoKiX-Python-CLI/cfg.

```
python get_configs.py
```

- In addition to prebuilt data loggers, a general data logging tool using `stream_configuration` files is offered:

```
python stream_logger.py <stream_configuration_file_name>
```

This is a similar tool that is also available with RoKiX Windows GUI and RoKiX Android App.

- It is possible to view the recorded log with the `plot.py` application (Figure 20):

```
kx022_kx122>python ..\plot.py kx022.csv
```

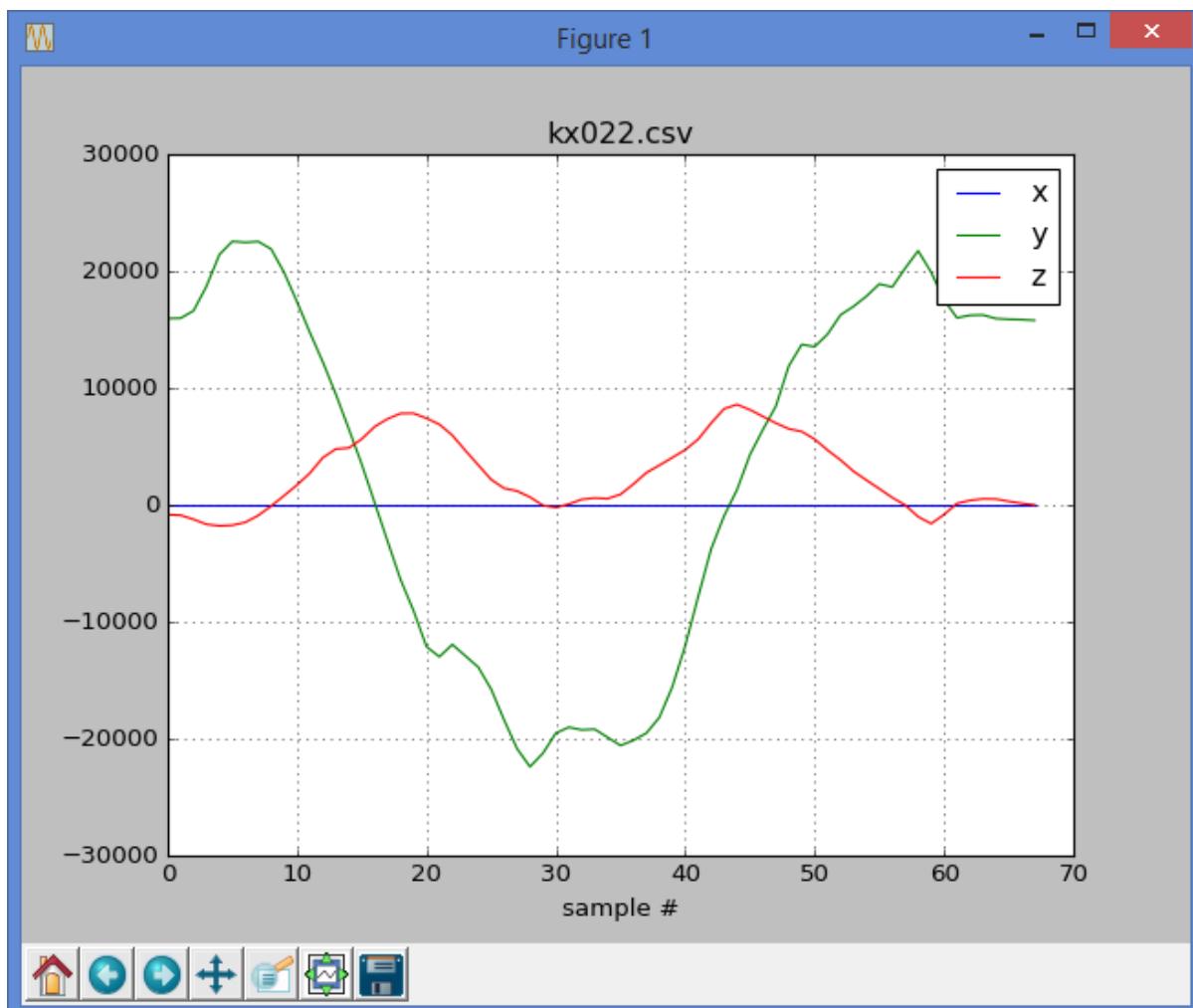


Figure 20: Recorded signals shown by the `plot.py` application.

- The frequency content of the recorded data can be analyzed and shown using the plot.py application with a special option -f as follows:

```
kx022_kx122>python ..\plot.py kx022.csv -f
```

6.8. Changing test application configuration

Test applications allow the user to modify the most common device configuration alternatives, for example, g-range, filtering and ODR settings (see Figure 21 below). It is recommended that the new configurations are given by the main program calling the test application and then overriding the default settings as required.

```

64
65
66
67 def enable_data_logging(sensor,
68         odr=25,
69         max_range='2G',
70         lp_mode=False,
71         low_pass_filter='ODR_9',
72         power_off_on=True,      # set to False if this function is part of other configuration
73         int_number=None):
74
75     LOGGER.info('enable_data_logging start')
76     #
77     # parameter validation
78     #
79     assert sensor.name in KX022Driver.supported_parts
80
81     if sensor.name == 'KX122':
82         valid_odrs = e122.KX122_ODCRTL_OSA.keys()
83     else:
84         valid_odrs = e.KX022_ODCRTL_OSA.keys()
85
86     assert convert_to_enumkey(odr) in valid_odrs, 'Invalid odr value "{}". Valid values are {}'.format(
87         odr, valid_odrs)
88
89     assert max_range in e.KX022_CNTL1_GSEL.keys(), 'Invalid max_range value "{}". Valid values are {}'.format(
90         max_range, e.KX022_CNTL1_GSEL.keys())
91
92     assert (lp_mode in list(e.KX022_LP_CNTL_AVC.keys()) +
93             [False]), 'Invalid lp_mode value "{}". Valid values are: False or {}'.format(
94         lp_mode, e.KX022_LP_CNTL_AVC.keys())
95
96     assert low_pass_filter in list(e.KX022_ODCRTL_LPRO.keys()) + \
97         ['BYPASS'], 'Invalid filter value "{}". Valid values are: BYPASS or {}'.format(
98         filter, e.KX022_ODCRTL_LPRO.keys())
99
100    # Set sensor to stand-by to enable setup change
101    if power_off_on:
102        sensor.set_power_off()
103
104    #
105    # Configure sensor
106    #
107
108    # NOTE CNTL1 registers are configured first

```

Figure 21: Python code of an example test application.

6.9. Reference driver implementation

The *RoKiX Python CLI* offers platform independent reference device driver software implemented by Python. Since only the device related configurations are in the driver, and the platform specific dependencies are not visible, it gives a good starting point for porting the driver software to a desired target platform (Figure 22).

All drivers follow the same interface defined in `kx_lib\sensor_base.py`

```

22 import struct
23 from . import imports # pylint: disable=unused-import
24 from kx_lib.kx_configuration_enum import BUSI_I2C, BUSI_SPI
25 from kx_lib.kx_sensor_base import SensorDriver, AxisMapper
26 from kx_lib import kx_logger
27 from kx_lib.kx_util import delay_seconds, bin2uint16, bin2uint8
28 from kx_lib.kx_configuration_enum import CH_ACC, ACTIVE_LOW, ACTIVE_HIGH
29 from kx022 import kx022_registers
30 from kx022 import kx122_registers
31
32 LOGGER = kx_logger.get_logger(__name__)
# LOGGER.setLevel(kx_logger.DEBUG)
34
35 r = kx022_registers.registers()
36 b = kx022_registers.bits()
37 m = kx022_registers.masks()
38 e = kx022_registers.enums()
39
40 r122 = kx122_registers.registers()
41 b122 = kx122_registers.bits()
42 m122 = kx122_registers.masks()
43 e122 = kx122_registers.enums()
44
45 # for PC1 start delay (acc) calculation
46 hz = [12.5, 25.0, 50.0, 100.0,
47       200.0, 400.0, 800.0, 1600.0,
48       0.781, 1.563, 3.125, 6.25,
49       3200.0, 6400.0, 12800.0, 25600.0]
50
51
52 class KX022Driver(SensorDriver):
53     _WAIS022 = [b.KX012_WHO_AM_I_WIA_ID,
54                 b.KX022_WHO_AM_I_WIA_ID,
55                 b.KX023_WHO_AM_I_WIA_ID]
56
57     _WAIS122 = [0x2C, # for KX222
58                 b122.KX112_WHO_AM_I_WIA_ID,
59                 b122.KX122_WHO_AM_I_WIA_ID,
60                 b122.KX123_WHO_AM_I_WIA_ID,
61                 b122.KX124_WHO_AM_I_WIA_ID]
62
63     def __init__(self):
64         SensorDriver.__init__(self)
65         self.i2c_sad_list = [0x1E, 0x1F, 0x1C, 0x1D]

```

Figure 22: Screen capture of an example driver software.

7. Troubleshooting and known issues

7.1. General

- Both with *RoKiX Windows GUI* and *RoKiX Python CLI*, timestamping is done on a PC and it is not accurate with high ODRs. This influences the delta time statistics.

7.2. Communications

- USB communication may miss device data samples or the USB connection is lost randomly: Use good quality USB cables which are USB certified.



- USB performance is not good on all Windows machines. The root cause is yet unknown.
- The *RoKiX IoT Platform SW* may not connect to the *Kionix IoT Sensor Node* when USB connection is used and the battery is connected to the *Kionix IoT Sensor Node*: the *Kionix IoT Sensor Node* must be powered on before connecting it to the PC with a USB cable. Otherwise the FTDI IC may get powered on slightly before nRF51822 SoC and thus may not detect nRF51822 SoC properly.

7.3. RoKiX Windows GUI

- When running the application for the first time Windows may inform the user that it is a security risk to run an unknown application. This can be solved by selecting *more info / run anyway*.
- When using Windows BLE, ODR can occasionally be slow. Turning off WLAN usually helps to improve the speed. Windows BLE connection speed is highly dependent on your PC load.
- When using Windows BLE, especially on Windows 10, there can be a pairing problem with the *RoKiX IoT Sensor Node*. Pairing issues are indicated by the LEDs flashing green and red colors. This problem can be solved just by releasing the pairing and doing it again.
- If an error message shown in Figure 23 appears, the Windows .NET installation is not up-to-date. Please run Windows update in order to solve this.
- Sometimes after installation the desktop shortcuts will not work. To overcome this, please uninstall and reinstall the application again to a different destination directory.
- Bluetooth streaming speed (bandwidth) varies widely over many HW/SW configurations. If you encounter issues, please try:
 1. Closing all other programs
 2. Setting the Power profile to High Performance
 3. Plugging the laptop into AC power
 4. Turning Wi-Fi off
 5. Changing the RoKiX Windows GUI window size
 6. Turning off grid lines

- In case of connection problems or application crash, please check the error log file of the *RoKiX Windows GUI*. The default path of this file is: \Documents\RoKiX\RoKiX-Windows-GUI\errorlog.txt.

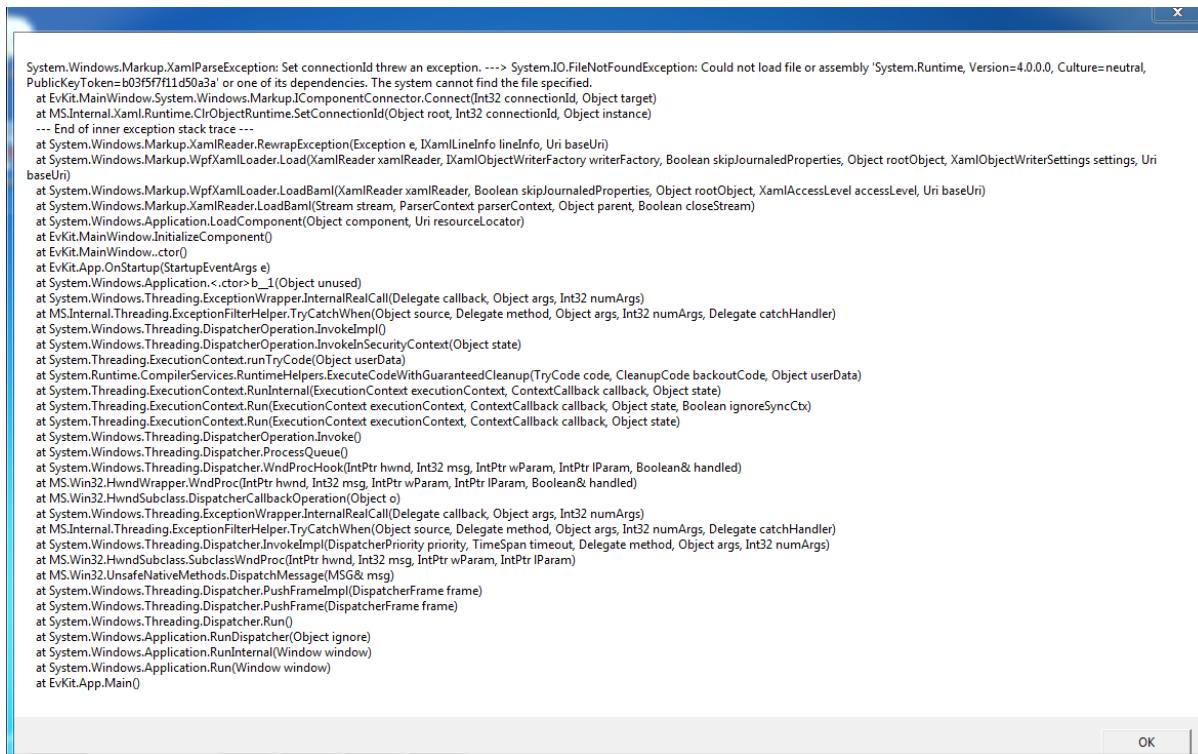


Figure 23: A Windows error message which indicates that Windows .NET installation is out-of-date.

7.4. RoKiX Python CLI

- In case there are issues with the *RoKiX Python CLI* operations, it is possible to change the logging level from default INFO to DEBUG for example.
`logging_level = ERROR ; DEBUG / INFO / WARNING / ERROR / CRITICAL`
- Additionally, logs can be saved to file by defining a file name for the key `log_file`, which is empty (e.g. no logging to file) by default.
- The *RoKiX Python CLI* verifies that it can capture all interrupts (before starting to monitor interrupt lines it is first verified that interrupt is not yet triggered). If the `logging_level` is set to WARNING or higher and interrupt speed (for example ODR) is too high a warning message will be displayed.
- If the driver or lib code has been modified and execution fails to import error, delete all *.pyc files.
- When having issues with accessing the serial ports on some Linux systems (e.g. Ubuntu 18.04) do the following: Create a file `/etc/udev/rules.d/50-rohm.rules` with the following content (as an administrator or use ‘`sudo`’):

```
# Kionix IoT Sensor Node:
```

```
ACTION=="add", SUBSYSTEM=="usb", ENV{DEVTYPE}=="usb_device",
ATTRS{idVendor}=="0403", ATTRS{idProduct}=="6015"
ACTION=="add", KERNEL=="ttyUSB[0-9]*", ATTRS{idVendor}=="0403",
ATTRS{idProduct}=="6015", MODE="0666"
```

```
# RoKiX Sensor Node:
```

```
ACTION=="add", SUBSYSTEM=="usb", ENV{DEVTYPE}=="usb_device",
ATTRS{idVendor}=="04b5", ATTRS{idProduct}=="0602", ENV{ID_MM_DEVICE_IGNORE}="1"
ACTION=="add", KERNEL=="ttyACM[0-9]*", ATTRS{idVendor}=="04b5",
ATTRS{idProduct}=="0602", MODE="0666"
```

```
# CY8CKIT-059:
```

```
ACTION=="add", SUBSYSTEM=="usb", ENV{DEVTYPE}=="usb_device",
ATTRS{idVendor}=="04b5", ATTRS{idProduct}=="0601", ENV{ID_MM_DEVICE_IGNORE}="1"
ACTION=="add", KERNEL=="ttyACM[0-9]*", ATTRS{idVendor}=="04b5",
ATTRS{idProduct}=="0601", MODE="0666"
```

```
Reload newly created rules:
```

```
sudo udevadm control --reload-rules
```

8. Appendix 1

Sensor	Interface to sensor HW	Host adapter	Interface to client SW	Client
RoKiX Evaluation board	I2C SPI RoKiX Adapter Board A3	nRF52840-DK	USB	RoKiX Windows GUI
	I2C ADC RoKiX Adapter Board A3	Arduino Uno		
Rohm sensor module	I2C Rohm Sensor Evaluation Kit 001/002/003	nRF52840-DK or Arduino Uno		

Table 10: Additional HW configurations with USB interface for the Windows environment

Sensor	Interface to sensor HW	Host adapter	Interface to client SW	Client
RoKiX Evaluation board	I2C SPI RoKiX Adapter Board A3	nRF52840-DK	BLE	RoKiX Windows GUI
	I2C Rohm Sensor Evaluation Kit 001/002/003			

Table 11: Additional HW configurations with BLE interface for the Windows environment

Sensor	Interface to sensor HW	Host adapter	Interface to client SW	Client
RoKiX Evaluation Board, Rohm sensor module, or Sensor Add-on Board	I2C SPI Level shifter board	Aardvark I2C/SPI host adapter		
RoKiX Evaluation board	I2C SPI RoKiX Adapter Board A3	nRF52840-DK	USB	RoKiX Python CLI
	I2C ADC RoKiX Adapter Board A3	Arduino Uno		
Rohm sensor module	I2C Rohm Sensor Evaluation Kit 001/002/003	nRF52840-DK or Arduino Uno		
RoKiX Evaluation Board	I2C RoKiX Adapter Board A3	Raspberry Pi3	Flat cable	

Table 12: Additional HW configurations with USB or flat cable interface for the Python environment

Sensor	Interface to sensor HW	Host adapter	Interface to client SW	Client
RoKiX Evaluation board	I2C SPI RoKiX Adapter Board A3	nRF52840-DK	BLE	RoKiX Python CLI
Rohm sensor module	I2C Rohm Sensor Evaluation Kit 001/002/003			

Table 13: Additional HW configurations with BLE interface for the Python environment