

## 1. Overview

The *Kionix IoT Evaluation Kit* provides a powerful and easy to use environment to begin evaluation of Rohm and Kionix products. Multiple hardware options are supported (*Kionix Evaluation Kit HW*), as well as common, hardware independent sw tools (*Kionix Evaluation Kit SW*). *Kionix Windows Sensor Evaluation Software* consists of an easy to use graphical user interface for displaying / recording sensor data and a graphical register map editor which allows the user to see and change the content of control registers. *Kionix Multi-OS Evaluation Software*, in turn, provides reference implementation for driver sw and also a convenient way to access sensor low level features and versatile data logging functionality.

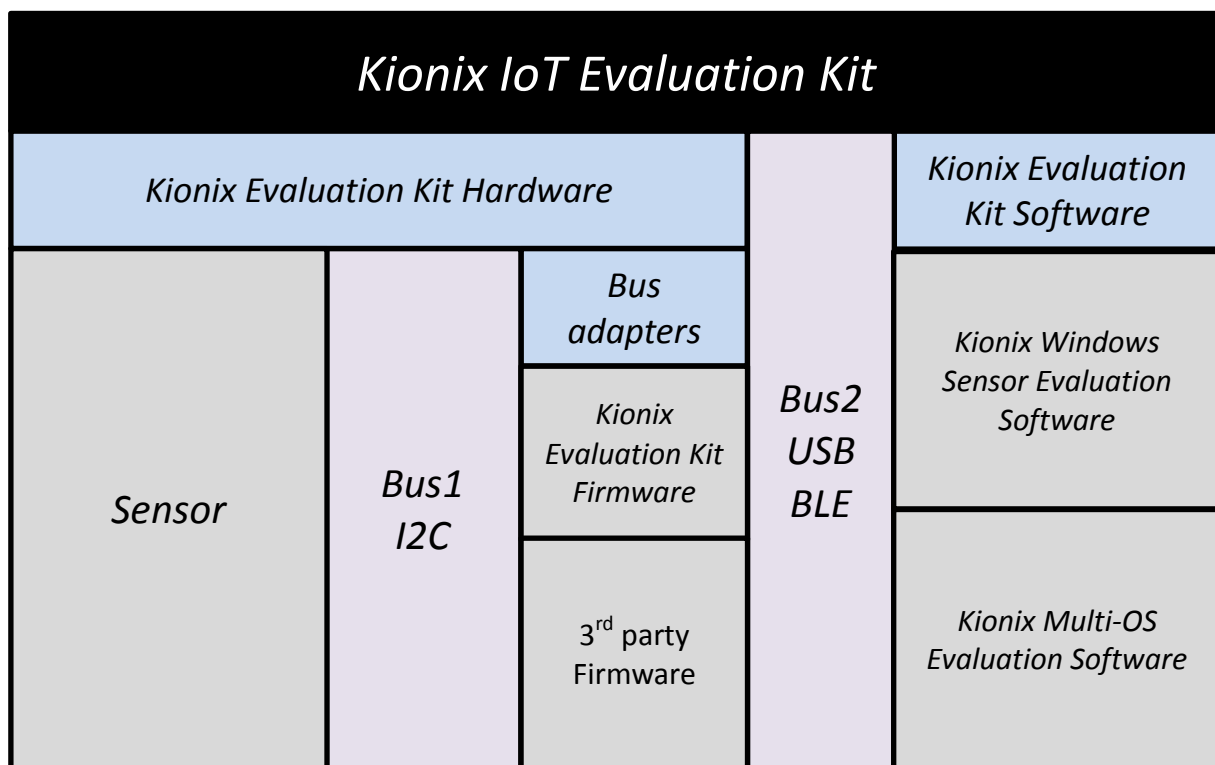


Figure 1: Structure of Kionix IoT Evaluation Kit

### 1.1. Definitions

<i>Kionix IoT Evaluation Kit</i>	<i>Complete offering of HW and SW for sensor evaluation purposes</i>
<i>Kionix Evaluation Kit HW</i>	PWB containing sensor component and/or bus connection between sensor and bus adapter.
<i>Kionix IoT Sensor Node</i>	MCU + BLE + sensors + battery running <i>Kionix Evaluation Kit Firmware</i>
<i>Kionix IoT Sensor Node Mounting Accessory</i>	Wrist band for <i>Kionix IoT Sensor Node</i>
<i>Kionix Multi-Platform Adapter Board</i>	Board specifically designed to easily interface with Kionix sensors and numerous development platforms. Example: A1 Adapter Board
<i>Kionix Evaluation Board</i>	Kionix sensor evaluation board with 14 pin header <a href="http://www.kionix.com/developer-tools">http://www.kionix.com/developer-tools</a>
<i>Kionix Evaluation Kit SW</i>	<i>Kionix Windows Sensor Evaluation Software</i> or <i>Kionix Multi-OS Evaluation Software</i>
<i>Kionix Windows Sensor Evaluation Software</i>	Kionix Sensor evaluation software with graphical user interface (GUI) running in Windows OS
<i>Kionix Multi-OS Evaluation Software</i>	Kionix sensor evaluation software with command line interface for quickly testing sensor low level features
<i>Kionix Evaluation Kit Firmware</i>	Proprietary firmware running on microcontroller based bus adapters
<i>Kionix BLE Router</i>	Android application which passes messages between <i>Kionix IoT Sensor Node</i> over BLE connection and PC over USB connection.

## Table of Contents

1.	Overview .....	1
1.1.	Definitions.....	2
Table of Contents.....		3
2.	Hardware.....	7
2.1.	Hardware architecture .....	7
2.1.1.	Supported hardware configurations.....	7
2.2.	Kionix IoT Sensor Node.....	9
2.2.1.	Kionix IoT Sensor Node Layout.....	10
2.2.2.	Adding new I2C slave devices .....	11
2.3.	<i>Kionix Multi-Platform Adapter Board</i> .....	11
2.3.1.	Supported development platforms .....	11
2.4.	Rohm Multi-Sensor Shield .....	11
2.4.1.	Stand alone example applications .....	12
2.5.	Aardvark I2C/SPI Host Adapter.....	12
2.6.	Embedded Linux .....	12
3.	Software.....	13
3.1.	Overview .....	13
3.1.1.	<i>Kionix Evaluation Kit Firmware</i> .....	13
3.1.2.	<i>Kionix Evaluation Kit SW</i> .....	13
3.1.2.1.	Windows Sensor Evaluation Kit Software .....	13
3.1.2.2.	<i>Kionix Multi-OS Evaluation Software</i> .....	13
3.1.3.	<i>Kionix BLE Router Android Software</i> .....	14
4.	Dependencies.....	14
4.1.	Installing required 3rd party software .....	14
4.1.1.	Python .....	14
4.1.2.	Aardvark .....	14
4.1.3.	FTDI USB Serial driver .....	15
4.1.4.	Android USB driver and ADB .....	17
4.1.5.	MICROSOFT VISUAL C++ REDISTRIBUTABLE.....	17
4.2.	<i>Kionix BLE Router setup to Android</i> .....	17
4.2.1.	Installation to PC.....	18
4.2.2.	Installation to Android devices .....	18
4.2.3.	Connecting Android to IoT Board .....	19
4.2.4.	Connecting Android to PC .....	21
4.3.	<i>Kionix Evaluation Kit Firmware programming / updating</i> .....	21

4.3.1.	Kionix IoT Sensor Node firmware .....	21
5.	Kionix Windows Sensor Evaluation Software .....	25
5.1.	Introduction .....	25
5.2.	Setup .....	25
5.2.1.	Installation .....	25
5.2.2.	Configuration .....	25
5.2.2.1.	Kionix Evaluation Kit HW and connection type selection .....	25
5.2.2.2.	Settings file.....	25
5.2.2.3.	Background image .....	26
5.3.	Getting Started .....	27
5.4.	User Interface – Menu bar.....	28
5.4.1.	File – Menu .....	28
5.4.1.1.	About.....	28
5.4.1.2.	Exit.....	28
5.4.2.	Data – Menu .....	28
5.4.2.1.	Streaming.....	28
5.4.2.2.	Channel.....	28
5.4.2.3.	Sensor Fusion.....	29
5.4.2.4.	Logging .....	30
5.4.2.5.	Raw data .....	30
5.4.3.	Connection – Menu .....	31
5.4.3.1.	Android BLE connection .....	31
5.4.3.2.	Windows BLE connection .....	32
5.4.3.3.	Connection reset.....	32
5.4.4.	Registers – Menu .....	32
5.4.4.1.	Load .....	32
5.4.4.2.	View.....	32
5.4.5.	Settings – Menu .....	32
5.4.5.1.	Auto Connect .....	33
5.4.5.2.	COM port .....	33
5.4.5.3.	Logging .....	33
5.4.6.	Device – Menu .....	33
5.5.	User Interface - Tabs.....	34
5.5.1.	Cube – Tab .....	34
5.5.2.	Plotter – Tab .....	35
5.5.2.1.	Zooming .....	36
5.5.2.2.	Pausing .....	36

5.5.2.3.	Moving .....	36
5.5.3.	Air Mouse – Tab .....	36
5.5.4.	Registers – Tab .....	37
5.5.4.1.	Register set and polling .....	37
5.6.	User Interface - Status bar.....	39
5.7.	User Interface - Pop-up windows.....	39
5.7.1.	Streaming pop-up window .....	39
5.7.2.	Orientation reset pop-up window .....	39
5.7.3.	Magnetometer calibration pop-up window .....	39
5.8.	Orientation reset.....	40
5.8.1.	Cube reset.....	40
5.8.2.	AirMouse reset .....	41
5.9.	Shortcuts .....	42
6.	Kionix Multi-OS Evaluation Software .....	43
6.1.	Introduction .....	43
6.2.	Set Up .....	43
6.2.1.	Installation .....	43
6.2.2.	Configuration .....	43
6.2.2.1.	Connection to <i>Kionix Evaluation Kit HW</i> .....	43
6.2.2.2.	Generic settings.....	44
6.2.2.3.	Data Ready operation settings.....	44
6.2.2.4.	Interrupt polarity .....	45
6.2.2.5.	Interrupt lines .....	45
6.3.	Getting Started .....	45
6.4.	File structure of the evaluation kit .....	46
6.5.	Running test applications .....	48
6.5.1.	High ODR logging .....	48
6.5.2.	Other provided tools .....	48
6.6.	Changing test application configuration.....	50
6.7.	Reference driver implementation.....	51
7.	Troubleshooting and known issues .....	53
7.1.	Communications .....	53
7.2.	<i>Kionix Windows Sensor Evaluation Software</i> .....	53
7.3.	<i>Kionix Multi-OS Evaluation Software</i> .....	55
8.	Appendix .....	56
8.1.	Connecting <i>Kionix Evaluation Board</i> to Aardvark I2C .....	56
8.2.	Connecting <i>Kionix Evaluation Board</i> to Beagle Bone Black .....	56



## 2. Hardware

*Kionix IoT Evaluation Kit* is designed to support multiple Bus adapters for accessing sensors from *Kionix Evaluation Kit SW*.

### 2.1. Hardware architecture

Usually evaluation kit software runs on client machine where there is no direct connectivity to sensor. Because of this, a separate bus adapter is needed which transfers messages from the client machine to the sensor. For that two separate bus connections are needed.

- Bus 2 between the Bus adapter and the client machine. Currently standard USB and BLE connections are supported
- Bus 1 between the Bus adapter and sensor. Currently I2C and SPI connectivity are supported with dedicated adapter boards



Figure 2: Hardware architecture of *Kionix IoT Evaluation Kit*

#### 2.1.1. Supported hardware configurations

Both “off the shelf” adapters (For example Aardvark) and microcontroller based adapters are supported (*Kionix IoT Sensor Node*). Microcontroller based adapters use proprietary *Kionix Evaluation Kit Firmware* for communication and provide a quick way to support new microcontroller based adapters in the future.

Adapter boards are offered for “Bus 1” connection between the bus adapter and sensor (2.3, 2.4 and 2.4).

For Bus 2 connection both wired USB and wireless BLE connections are supported.













Sensor	Bus 1	Bus Adapter	Bus 2	Client
 Kionix IoT node	 Kionix IoT node	 Kionix IoT node	 USB cable	PC or Embedded Linux
 Kionix IoT node	 Kionix IoT node	 Kionix IoT node	 BLE to Android	PC
 Kionix Ev.board	 <a href="#">Level shifter Board</a>	 <a href="#">Aardvark I2C/SPI host adapter</a>	 USB cable	PC

Table 1: Supported HW configurations



### 2.2. Kionix IoT Sensor Node



The *Kionix IoT Sensor Node* is a small sensor node based on the [Nordic Semiconductor nRF51822](#) Series system-on-chip (SoC), and it supports both Bluetooth Low Energy (BLE 4.1) and USB interfaces. It consists of four (4) sensors:

- [KXG03](#) – Digital Accelerometer and Gyroscope
- [KX112](#) – Digital Accelerometer
- [KMX62](#) – Digital Accelerometer and Magnetometer
- [BM1383GLV](#) – Digital Barometric Pressure Sensor (ROHM)

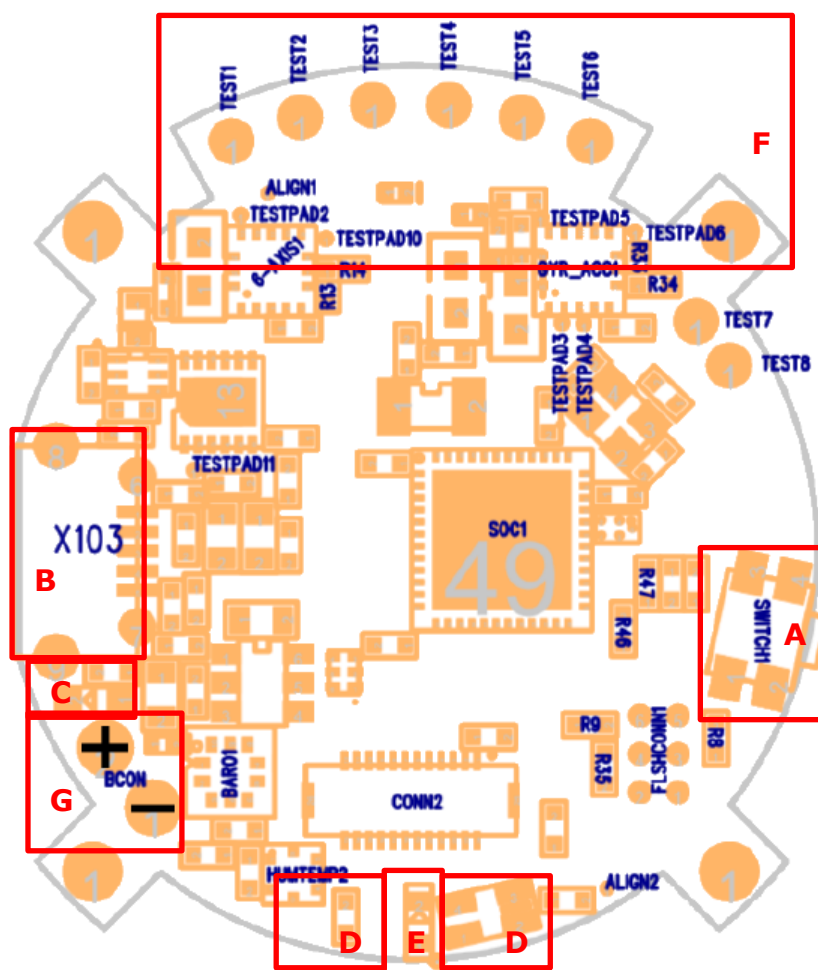
Detailed information about the Kionix IoT board in directory `Kionix-IoT-Evaluation-Kit\Kionix-Evaluation-Kit-HW\Kionix-IoT-Sensor-Node\`

Note: Before using the *Kionix IoT Sensor Node* with a battery, please charge the battery full. The battery is charged from the USB connector. The battery is fully charged when the battery charging LED turns off.

### 2.2.1. Kionix IoT Sensor Node Layout

The main parts of layout and component placement are as follows

- A. Power switch
- B. USB connector for communication and battery charging
- C. Battery charging indication LED
- D. Red + Green status LEDs
- E. Blue status LED
- F. Test pad area
- G. Battery connection pads (recommended to use [PRT-13112](#) battery)



### 2.2.2. Adding new I2C slave devices

More sensors can be connected to the *Kionix IoT Sensor Node* in two ways:

- AistinBus24 add-on board connector (Image 1 on Kionix-Evaluation-Kit-HW\Kionix-IoT-Sensor-Node\BTL3K3\_Application\_note\_rev102.pdf).
- Test points

TEST1	GENIO1/INT1 KMX62-1031 GPIO1 INPUT/OUTPUT
TEST2	I2C_SDA I2C_SDA LINE (SENSORS)
TEST3	I2C_SCL I2C_SCL LINE (SENSORS)
TEST4	SENSOR INTERRUPT KXG03/INT2, KX122/INT2, BM1383/INT1
TEST5	SENSOR INTERRUPT KXG03/INT1, KX122/INT1
TEST6	VSEN SENSOR VOLTAGE SUPPLY (VCC SWITCHED ON BY SEN)
TEST7	VCC REGULATED VOLTAGE SUPPLY (2.8V)
TEST8	GND COMMON GROUND

### 2.3. Kionix Multi-Platform Adapter Board

The *Kionix Multi-Platform Adapter Board* is an inexpensive board specifically designed to easily interface with Kionix sensors and numerous development platforms.

It is a great platform for someone who needs an easy way to interface with *Kionix Evaluation Boards* featuring a 14-pin male header or additional sensors using ready to use development platforms such as Cypress, Arduino, and Raspberry Pi.

The board utilizes 5V power from the host platform and converts to jumper-selectable (1.8V/2.5V/3.3V) VDD voltage using an on-board ROHM adjustable LDO. IO\_VDD is selected with a jumper to be either 3.3V (provide from another dedicated on-board ROHM adjustable LDO) or tied directly to VDD. Current measurement is possible via removable jumpers on IO\_VDD, and VDD rails. A LED is provided on the power rail to indicate the board is powered.

#### 2.3.1. Supported development platforms

- [Cypress](#) Low-Cost Prototype Kits: CY8CKIT-059 (PSoC 5LP / Cortex M3), CY8CKIT-043 (PSoC 4M / Cortex M0), CY8CKIT-049 (PSoC 4 / Cortex M0). These are low cost, yet very powerful Mixed Digital & Analog platforms with free development tools. The PSoC 5LP Cortex M3 platform also comes with built in USB 2.0.
- [Arduino Uno](#): The adapter board acts like a shield that can be plugged on top of compatible Arduino PCBs.
- Raspberry Pi: [Pi 1 ModelA+](#), [Pi 1 Model B+](#), [Pi 2 Model B](#), [Pi 3 Model B](#). The Raspberry Pi is a low cost, small form factor computer with endless capabilities.

### 2.4. Rohm Multi-Sensor Shield

ROHM Semiconductor's Multi-Sensor Shield, the [SENSORSHLD1-EVK-101](#), is a shield evaluation platform that integrates multiple ROHM sensor products on a

single board. The shield uses standard Arduino UNO R3 shield interface pins, making it possible to connect to any evaluation kit with a shield interface header.

The following sensors are included:

- Analog Temperature Sensor ([BDE0600G](#))
- Digital Barometric Pressure Sensor ([BM1383AGLV](#))
- Hall Switch Sensor (Omnipolar with Polarity Discrimination) ([BU52014HFV](#))
- Geomagnetic Sensor (BM1422GMV)
- Digital Color Sensors ([BH1745](#))
- Optical Proximity Sensors and Ambient Light Sensors ([RPR-0521](#))
- Analog UV Sensor ([ML8511](#))
- Digital Accelerometer ([KX122-1037/KX122-1048](#))
- Digital Magnetometer and Accelerometer ([KMX62](#))
- Digital Gyroscope and Accelerometer ([KXG03](#))

NOTE: Before using this shield with Arduino, please refer to [Usage Manual document](#) for needed HW rework.

### 2.4.1. Stand alone example applications

Stand alone example applications available for listed HW are also [available](#) for the following platforms:

- Arduino UNO, Firmware Example and Documentation
- LPCXpresso54102, Firmware Example Only
- Nordic Semiconductor nRF51-DK, Firmware Example and Documentation: Recommended for BTLE Low Power Sensor Applications

## 2.5. Aardvark I2C/SPI Host Adapter

Aardvark is a product of Total Phase Inc. *Kionix Multi-OS Evaluation Software* running on PC windows supports both I2C and SPI connection on Aardvark.

Since Aardvark uses 5 volts for logic voltage, level shifting may be required. Exemplary documentation on how to connect KXG03 *Kionix Evaluation Board* to Aardvark I2C bus is on Appendix 8.1.

## 2.6. Embedded Linux

*Kionix Multi-OS Evaluation Software* can also be run on embedded Linux system.

An example connection of the Beagle Bone Black and *Kionix Evaluation Board* is illustrated in Appendix 8.2. Please note the possible need of a level shifter. The Beagle Bone Black is a 3.3V system so evaluation board can be connected to it directly.

### 3. Software

*Kionix IoT Evaluation Kit* consist of both *Kionix Evaluation Kit Firmware* for supported evaluation kit hardware, as well as, *Kionix Evaluation Kit SW* for evaluating the array of sensors Kionix/ROHM has to offer.

#### 3.1. Overview

##### 3.1.1. *Kionix Evaluation Kit Firmware*

Firmware for supported microcontroller based *Kionix IoT Evaluation Kits* is provided by Kionix. Programming instructions and updating instructions are described in chapter 4.3. *Kionix Evaluation Kit Firmware* supports communication with *Kionix Evaluation Kit SW*. Kionix currently offers firmware for:

- Kionix IoT Sensor Node

##### 3.1.2. *Kionix Evaluation Kit SW*

The provided *Kionix Evaluation Kit SW* is divided into two parts: *Kionix Windows Sensor Evaluation Software* and *Kionix Multi-OS Evaluation Software*.

###### 3.1.2.1. **Windows Sensor Evaluation Kit Software**

Windows Sensor Evaluation Kit Software provides an intuitive graphical user interface demonstrating high level sensor offerings and features. Some of the features included are:

- Demonstration of Kionix Software offering like the Sensor Fusion Algorithm
- Visual display of real time sensor data
- Ability to record sensor data to a file
- Sensor registry editor
- Air mouse

Please see chapter 6 for usage instructions for *Kionix Windows Sensor Evaluation Software*.

###### 3.1.2.2. **Kionix Multi-OS Evaluation Software**

*Kionix Multi-OS Evaluation Software* provides an advanced interface for demonstrating low level sensor features. Some of the features included are:

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

Usage instructions for *Kionix Multi-OS Evaluation Software* are in chapter 6.

### 3.1.3. Kionix BLE Router Android Software

*Kionix BLE Router* application is provided for the Android Operating System. The application's purpose is to provide a Bluetooth Low Energy (BLE) connection required for using the offered software where no BLE connection is present. An example of this is if a desktop machine does not support Bluetooth or there are performance issues related to the built-in BLE functionality.

## 4. Dependencies

This chapter lists dependencies needed by *Kionix Evaluation Kit SW*.

### 4.1. Installing required 3rd party software

#### 4.1.1. Python

[Python](#) (version 2.7.x) interpreter is needed to run *Kionix Multi-OS Evaluation Software*.

##### Windows install

Download python interpreter (32 bit version) install package.

<https://www.python.org/downloads/> or  
ActivePython-2.7 <http://www.activestate.com/activepython/downloads>

The following python modules are needed and can be installed with the python package manager after python is installed:

For USB serial communication the following packages are needed:

```
pip install pyserial==3.0.1
```

IF automatic USB serial port detection is used (this feature is on by default)

```
pip install wmi  
pip install pypiwin32
```

If `plot.py` log viewer tool is used then matplotlib is required  
<http://matplotlib.org/>

##### Embedded Linux install

With Linux package manager (for example apt-get) install the following packages:

- Python 2.7.x
- i2c-dev
- i2c-tools
- python-smbus

#### 4.1.2. Aardvark

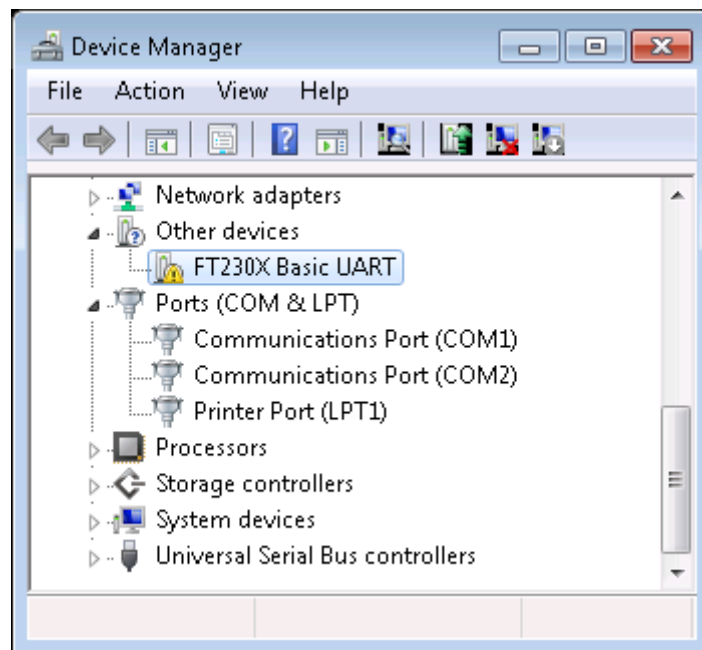
Aardvark I2C/SPI host adapter is supported by *Kionix Multi-OS Evaluation Software*.

Totalphase USB driver for Aardvark is bundled with *Kionix IoT Evaluation Kit* and located in folder `Kionix-Evaluation-Kit-Firmware\dependencies\Aardvark`. Latest drivers also available at <http://www.totalphase.com/products/usb-drivers-windows>

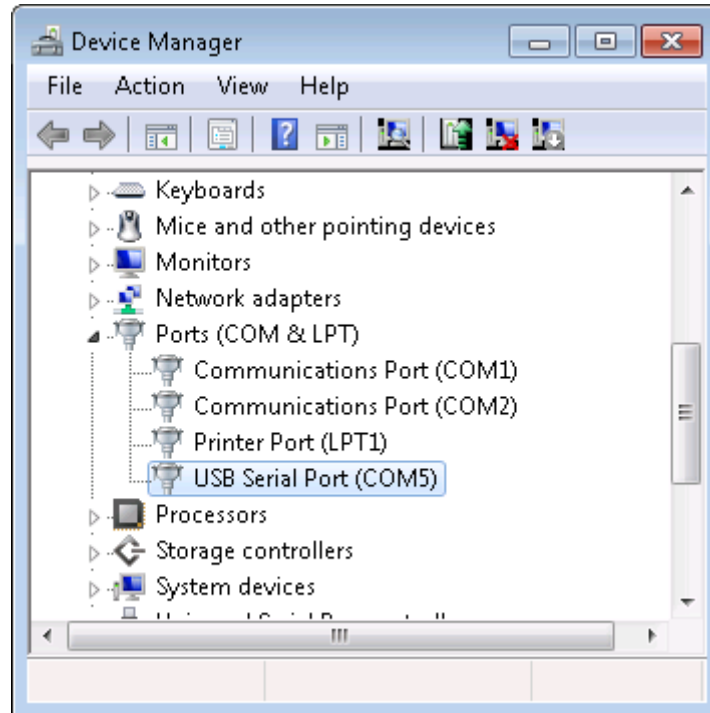
### 4.1.3.FTDI USB Serial driver

*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>  
[http://www.ftdichip.com/Drivers/CDM/CDM21216\\_Setup.exe](http://www.ftdichip.com/Drivers/CDM/CDM21216_Setup.exe)

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

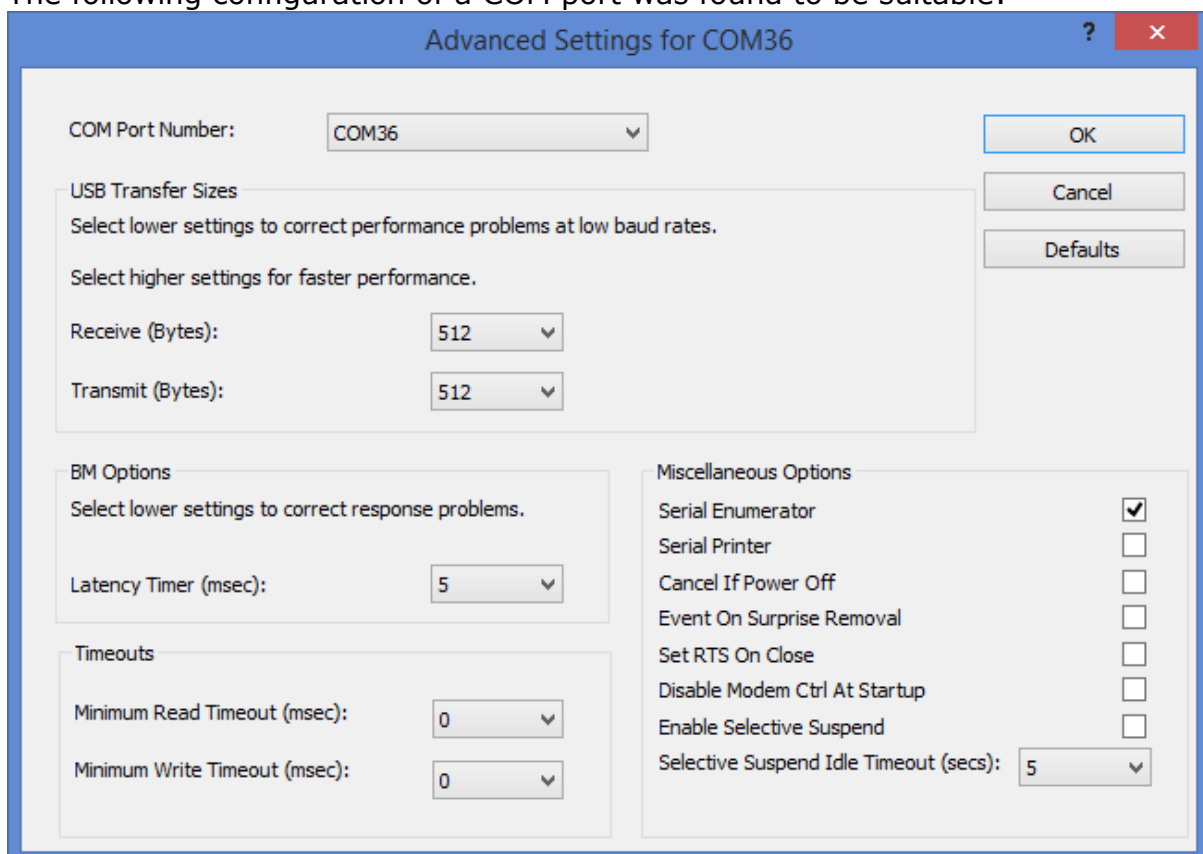


After drivers are installed, *Kionix IoT Sensor Node* is listed as "USB Serial Port (COMx)" where 'x' is the COM port allocated for *Kionix IoT Sensor Node*.



When using FTDI USB Serial connections with high ODR speeds ( $\geq 400\text{Hz}$ ), 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:





### 4.1.4. Android USB driver and ADB

The *Kionix Evaluation Kit SW* can use an Android device as BLE-USB router. When running the *Evaluation Kit SW* in Windows and if Windows does not install the needed drivers automatically, they can be loaded from: <http://developer.android.com/sdk/win-usb.html#download>

ADB.EXE can be found for example:

- Android Studio <https://developer.android.com/studio/index.html>
- Android platform-tools <http://stackoverflow.com/tags/adb/info>

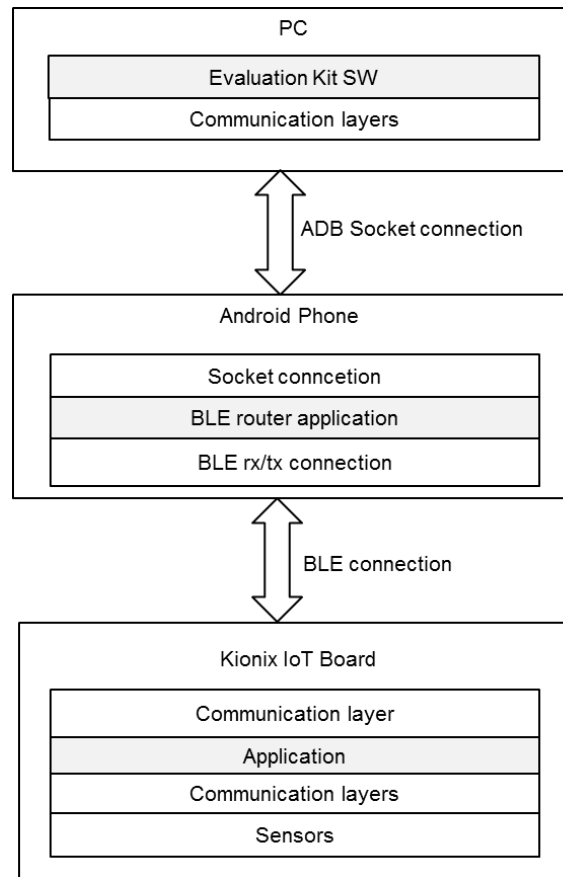
*Kionix BLE Router* set-up is described in chapter 4.2.

### 4.1.5. MICROSOFT VISUAL C++ REDISTRIBUTABLE

`vc_redist_x86.exe` is needed to run the *Kionix Windows Sensor Evaluation Software*. If it is not already installed on the PC, then locate the file in the install package which is bundled with the *Kionix IoT Evaluation Kit* and located in folder `Kionix-IoT-Evaluation-Kit\Kionix-Windows-Sensor-Evaluation-Software\dependency_install`.

## 4.2. Kionix BLE Router setup to Android

This chapter provides instructions on how to install the *Kionix BLE Router* for Android devices. The router application passes the BLE message between the phone and the *Kionix Evaluation Kit SW* in PC. The figure below illustrates how the connection is implemented.



### 4.2.1. Installation to PC

Windows requires the Google USB driver and the `ADB.EXE` tool. (ref. chapter 4.1.4)

### 4.2.2. Installation to Android devices

#### Enable developer mode from the Android device

On Android 5.0 and higher, the Developer options screen is hidden by default. To make it visible, go to **Settings > About phone** and tap **Build number** seven times. Return to the previous screen to find **Developer options** at the bottom. On some devices, the Developer options screen may be located or named differently.

Connect the Android device to the PC via USB and test connection. Test that the Android device is found:

```
C:\Windows\system32\cmd.exe
C:\>adb devices
List of devices attached
0602cdd700faa815    device
```

The *Kionix BLE Router* is located in folder `Kionix-IoT-Evaluation-Kit\Kionix-BLE-Router\Android`

Install the *Kionix BLE Router* to the Android device with the following command:  
`install_to_phone.bat`

Or:

```
adb install evkit-ble-router-v1.apk
```

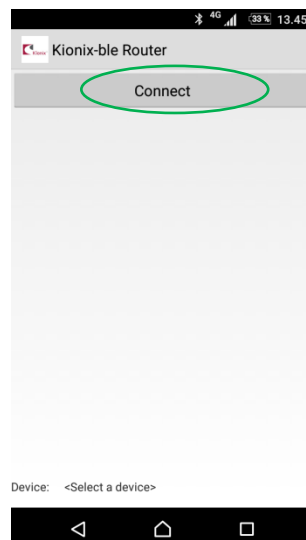
### 4.2.3. Connecting Android to IoT Board

On the *Kionix IoT Sensor Node* do the following steps:

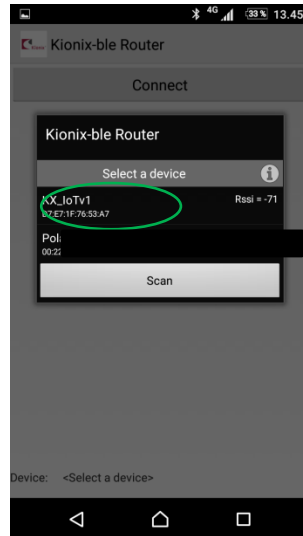
- Connect USB cable to power the device or use the battery.
- Turn on the *Kionix IoT Sensor Node* on by short pressing the power key. Red LED should be alight, when the device is entered to application mode.

On the Android device do the following steps:

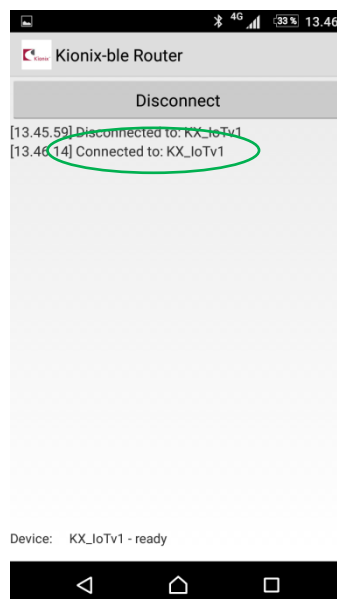
- Start *Kionix BLE Router* application.
- Press the "connect" button.



- The *Kionix IoT Sensor Node* should be automatically scanned and will be seen on the devices list.
- Select the device to connect.

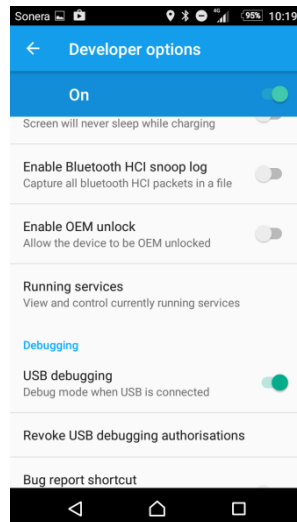


If connection was successful, you should see the connected message in the list view.



### 4.2.4. Connecting Android to PC

The *Kionix BLE Router* uses a USB cable connection to the PC. On top of the USB connection the ADB socket connection is used. To enable ADB socket connection, **Developer mode** and **USB debugging** must be enabled from within the device system settings, under **Developer options**.



### 4.3. Kionix Evaluation Kit Firmware programming / updating

This chapter describes the programming process of *Kionix Evaluation Kit Firmware* to supported microcontrollers.

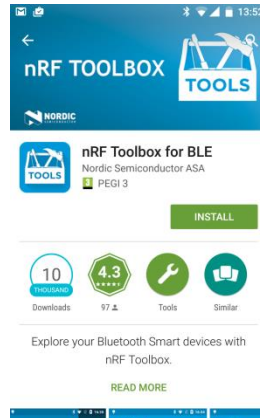
#### 4.3.1. Kionix IoT Sensor Node firmware

The *Kionix IoT Sensor Node* is shipped with pre-installed bootloader and *Kionix Evaluation Kit Firmware*. Both can be updated using iPhone or Android using "DFU" functionality of "nRF Toolbox" which is available in the application store (for example in [Google Play](#))

This chapter provides instructions on how to use DFU (Device Firmware Update) to update boot loader and *Kionix Evaluation Kit Firmware* to IOT Sensor Node with OTA (Over The Air) using Android or IOS phone.

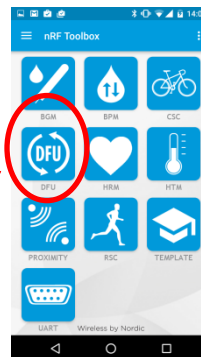
Update the *Kionix Evaluation Kit Firmware* to the *Kionix IoT Sensor Node* with the following procedure:

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



Press Install button

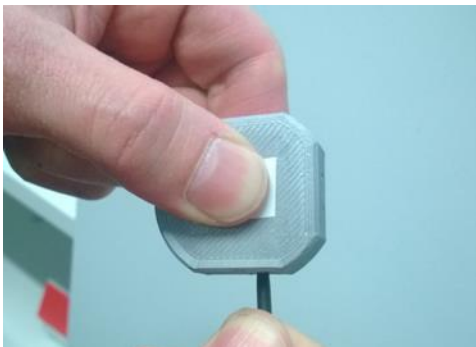
2) Start application from main icon



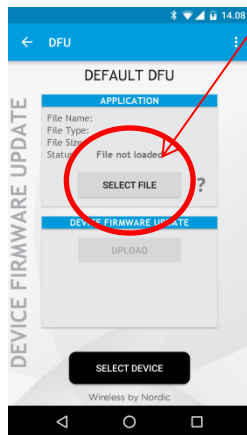
-> Press DFU icon

3) DFU menu will open

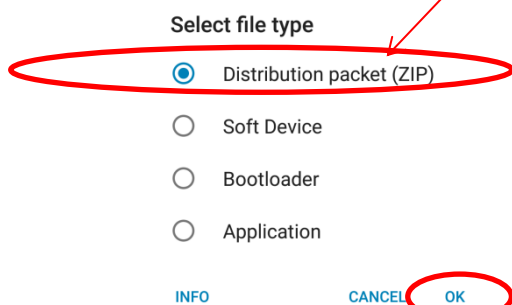
4) Before updating the application over-the-air, press *Kionix IoT Sensor Node* power button for 3 seconds. Check the device name using the nRF Toolbox. Press 'select device' button and see available devices. **Note! If device name in the list is different than 'KX\_Bv1' (or same prefix with greater number) then bootloader must be updated before updating firmware. Otherwise DFU will stop working and Bootloader and firmware update need to be done using ARM JTAG interface.**



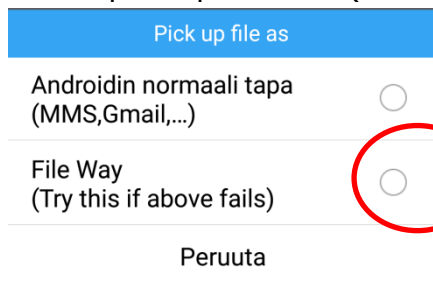
- 5) The *Kionix Evaluation Kit Firmware* is bundled with *Kionix IoT Evaluation Kit* and located in directory `Kionix-IoT-Evaluation-Kit\Kionix-Evaluation-Kit-Firmware\Kionix-IoT-Sensor-Node`
- Load the firmware update file to Android phone into a directory of your choice. Example `kionix-iot-firmware-v1.zip` to `/data` - directory. Alternatively
  - `copy_fwzip_to_phone.bat` can be used for copying (`adb.exe` [ref 4.2.1] must be found from PATH)
- 6) Press "Select File"-button



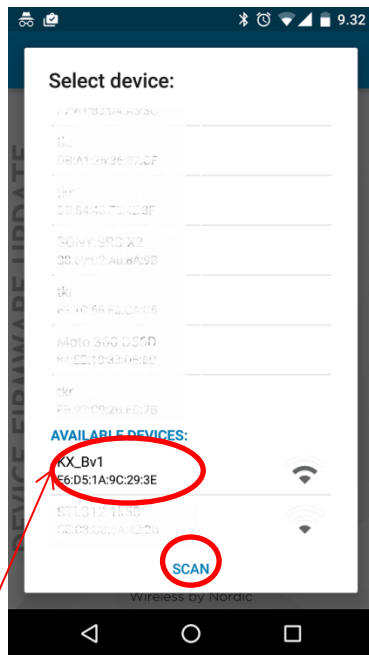
Select "Distribution packet (ZIP)"



- Press Ok button
- 7) Select you favorite file manager program
- Select the file
  - Select pick up method (File Way) button if asked



- 8) Press "Select Device"-button
- Scan to update MAC number info and choose your device



- b. Select your *Kionix IoT Sensor Node*'s MAC number from list

9) Press "Upload"-button

- a. Wait until uploading starts (may take up to 2 minutes)
  - i. Upload status bar informs the percentage level of the current upload
  - ii. If uploading does not start, please check "Troubleshooting hints" below.
- b. DFU application should inform that update was successful

Troubleshooting hints

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

References:

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



### 5. Kionix Windows Sensor Evaluation Software

#### 5.1. Introduction

The *Kionix Windows Sensor Evaluation Software* provides an easy to use graphical user interface demonstrating high level sensor offerings and features. Some of the features include:

- Demonstration of Kionix Software offering like the Sensor Fusion Algorithm
- Visual display of real time sensor data
- Ability to record sensor data to a file
- Sensor registry editor

#### 5.2. Setup

##### 5.2.1. Installation

The *Kionix Windows Sensor Evaluation Software* is located at `Kionix-IoT-Evaluation-Kit\Kionix-Windows-Sensor-Evaluation-Software\KionixEvaluationKit1_0.exe`

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

1. Refer to chapter 4 for installing needed additional software
  - install c++ runtime environment (4.1.5)
2. For *Kionix IoT Sensor Node* using *Kionix BLE Router*, Android USB devices drivers are needed (4.1.4)
3. Needed USB serial drivers (4.1.3, **Error! Reference source not found.**) if Windows does not install automatically)

##### 5.2.2. Configuration

###### 5.2.2.1. Kionix Evaluation Kit HW and connection type selection

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

- Evaluation Kit HW from Device-menu 5.4.6
- Connection type from Connection-menu 5.4.2.4

###### 5.2.2.2. Settings file

The *Kionix Windows Sensor Evaluation Software* default settings are in the "app.config" – file. By changing these values before application startup, *Kionix Windows Sensor Evaluation Software* will use them as default values. In normal use there is no need to change these values.

```
<setting name="Socket_port" serializeAs="String">
  <value>8100</value>
</setting>
<setting name="COM_port" serializeAs="String">
  <value>COM1</value>
```

```
</setting>
<setting name="Baud_rate" serializeAs="String">
  <value>460800</value>
</setting>
<setting name="Auto_connect" serializeAs="String">
  <value>True</value>
</setting>
<setting name="Log_quaternion_data" serializeAs="String">
  <value>True</value>
</setting>
<setting name="Log_sensor_data" serializeAs="String">
  <value>False</value>
</setting>
<setting name="Log_time_stamps" serializeAs="String">
  <value>False</value>
</setting>
<setting name="DefaultSensorFilePath" serializeAs="String">
  <value>SensorRegisters\kxg03.xml</value>
</setting>
<setting name="SensorView" serializeAs="String">
  <value>1</value>
</setting>
<setting name="DefaultSetFileFolder" serializeAs="String">
  <value>SensorSet</value>
</setting>
<setting name="DefaultPollingInterval" serializeAs="String">
  <value>1000</value>
</setting>
<setting name="HW_device" serializeAs="String">
  <value>Kionix__IoT</value>
</setting>
<setting name="Arduino_Baud_rate" serializeAs="String">
  <value>460800</value>
</setting>
<setting name="Auto_COM_port" serializeAs="String">
  <value>COM0</value>
</setting>
```

### 5.2.2.3. Background image

The *Kionix Windows Sensor Evaluation Software*'s background image can be changed if needed. Copy your wanted background image file to `Kionix-IoT-Evaluation-Kit\Kionix-Windows-Sensor-Evaluation-Software\Background.png`.

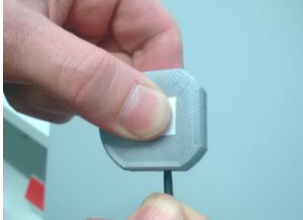
The *Kionix Windows Sensor Evaluation Software* package contains a couple of background images in `Backgrounds` – folder.

By default the application is using `Backgrounds\White_Kionix IoT.png`

### 5.3. Getting Started

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

1. Power on the *Kionix IoT Sensor Node*



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

NOTE: When using *Kionix IoT Sensor Node* in your PC for the first time, Windows will install a USB serial driver for it. It may take a while, please be patient.

2. Start *Kionix Windows Sensor Evaluation Software* with `KionixEvaluationKit1_0.exe`
3. When "Please enable data streaming to activate Cube movement!" – pop-up window appears on the screen, enable data streaming with "Streaming" - button or by pressing "CTRL + S".
4. Reset Cube orientation (ref. chapter 5.8.1)
5. The cube is now rotating according to your *Kionix IoT Sensor Node* movement

## 5.4. User Interface – Menu bar

### 5.4.1. File – Menu

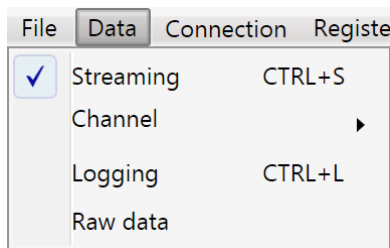
#### 5.4.1.1. About

Shows the *Kionix Windows Sensor Evaluation Software's* version and from which Git commit and when it has been built.

#### 5.4.1.2. Exit

Exits from application.

### 5.4.2. Data – Menu



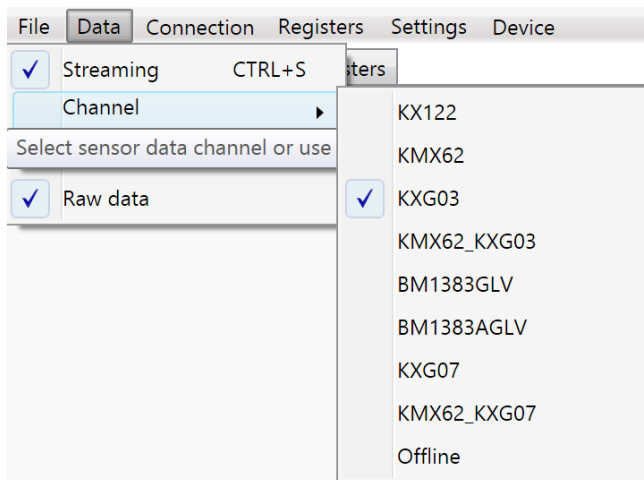
Data menu contains all main options for sensor data usage

#### 5.4.2.1. Streaming

Streaming menu item is used for enabling / disabling sensor data streaming. Shortcut: CTRL + S

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

#### 5.4.2.2. Channel



Channel menu is used for selecting the sensor data channel which the application is using to receive sensor data.

When selecting "Offline" file dialog will open for selecting the offline sensor data file to be used. Offline data is mainly used in development and for demo purposes. The selected offline data file must have accelerometer + magnetometer + gyro data in each line separated with space.

**NOTE:** Both the KXG03/KXG07 and the BM1383GLV/BM1383AGLV have the same I2C slave addresses if the corresponding part is assembled on *Kionix IoT Sensor Node*. For example: this means that the application allows the use of KXG07 channel even if KXG03 is present in device. Please make sure that the correct channel is selected to avoid misleading sensor data.

### 5.4.2.3. Sensor Fusion

The Sensor Fusion algorithm to be used is selected automatically according to used channel:

- when KXG03/KXG07 channel is used, Accelerometer + Gyro fusion is selected.
- when KMX62\_KXG03/KMX62\_KXG07 channel is used, Accelerometer + Magnetometer + Gyro fusion is selected.
- when KMX62 channel is used, Accelerometer + Magnetometer fusion is selected.

This selection affects operation of Cube 5.5.1, Air mouse 5.5.3, and logging 5.4.5.3.

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

CTRL + F1: Accelerometer + Gyro

CTRL + F2: Accelerometer + Magnetometer + Gyro

CTRL + F3: Accelerometer + Magnetometer

When 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 sensor.

### 5.4.2.4. Logging

Logging menu item is used for enabling / disabling sensor data logging. Status bar will show the log file name.

Logging to loki3.txt

More logging settings defined in 5.4.5.3.

Shortcut: CTRL + L

### 5.4.2.5. Raw data

If Raw data is enabled, raw sensor data is used by plotter and logger. When it's disabled SI units are used and plotter will show current SI units like this:

#### KXG03

- Acc\_X (m/s<sup>2</sup>)
- Acc\_Y (m/s<sup>2</sup>)
- Acc\_Z (m/s<sup>2</sup>)
- Gyro\_X (degrees/s)
- Gyro\_Y (degrees/s)
- Gyro\_Z (degrees/s)

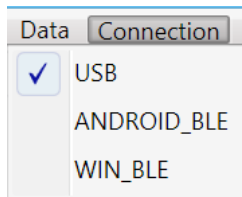
SI unit definition and data scaling settings are defined in data stream xml files located in "\RegisterConf\stream\_config\". KXG03 example:

```
<channels>
  <channel_1 name="Gyroscope"
    streams="Gyro_X (degrees/s),Gyro_Y (degrees/s),Gyro_Z (degrees/s)"
    SI_multiplier="2048 / 32768">
  </channel_1>
  <channel_2 name="Accelerometer"
    streams="Acc_X (m/s^2),Acc_Y (m/s^2),Acc_Z (m/s^2)"
    SI_multiplier="9.8 / 16384"> </channel_2>
</channels>
```

### 5.4.3. Connection – Menu

The *Kionix Windows Sensor Evaluation Software* connects to *Kionix IoT Sensor Node* either via USB COM port or via BLE.

Connection menu is used to select desired connection type. *Kionix Windows Sensor Evaluation Software* uses USB COM connection by default. When auto connect is enabled, USB connection is established automatically when the *Kionix Evaluation Kit HW* is connected.



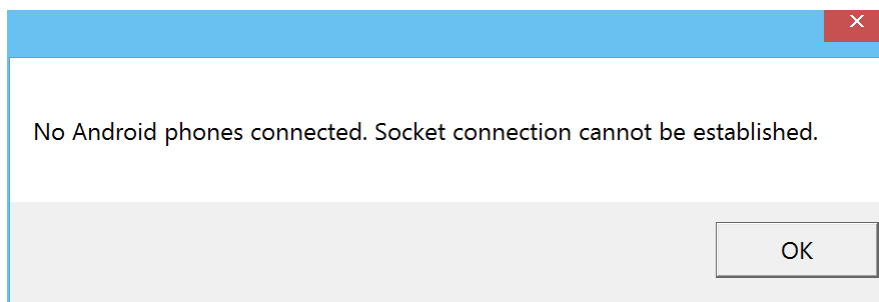
A BLE connection can be established by using an Android device (`ANDROID_BLE`) or with direct Windows BLE connection (`WIN_BLE`).

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

#### 5.4.3.1. Android BLE connection

With an Android BLE connection an Android phone is needed for receiving BLE data from the *Kionix IoT Sensor Node*. The *Kionix Windows Sensor Evaluation Software* reads the data via TCP socket connected to Android phone. The application will start and use the `adb.exe` which is used for communicating with the Android device via socket.

Before selecting BLE connection, please make sure that your Android device's USB drivers are installed and working per chapter 0. If you see this dialog, there's some problem with the Android phone connection:



**NOTE:** When having connection problems, please restart both the Android application *Kionix BLE Router* and *Kionix Windows Sensor Evaluation Software*. This will then clean the whole socket data transfer pipe.

### 5.4.3.2. Windows BLE connection

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

*Kionix IoT Sensor Node* must be paired with Windows before BLE connection can be created. Follow these steps to do the pairing:

1. Go to Control Panel\All Control Panel Items\Devices and Printers
2. Add a device
3. Choose "KX\_IoTv1"
4. Next  
» *Kionix IoT Sensor Node* led light should turn green during device installation

If pairing succeeded, "KX\_IoTv1" device is visible in "Unspecified" - section of "Devices and Printers"

In order to disconnect Windows BLE connection, *Kionix IoT Sensor Node* device must be turned off.

### 5.4.3.3. Connection reset

If having connection problems, "CTRL + R" can be used for refreshing current connection.

### 5.4.4. Registers – Menu

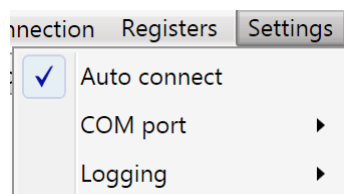
#### 5.4.4.1. Load

Load new sensor register XML file for "Registers" – tab. Please refer to chapter 5.5.4.

#### 5.4.4.2. View

Change "Registers" – tab view type. "Default" is descriptive register view and "Advanced" is line per register type of approach. "Advanced" – view is used by default.

### 5.4.5. Settings – Menu





### 5.4.5.1. Auto Connect

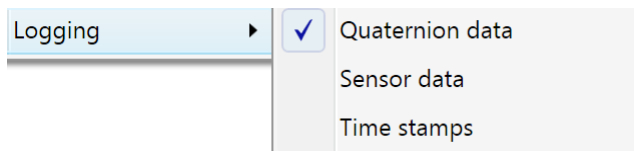
When "Auto connect" is enabled, the *Kionix Windows Sensor Evaluation Software* will automatically select the USB COM port for connected device.

### 5.4.5.2. COM port

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

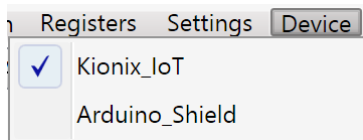
### 5.4.5.3. Logging

Under logging menu you can define what data is stored to file when logging sensor data.



The *Kionix Windows Sensor Evaluation Software* will store selected settings to user settings on the computer and will restore them the next time the application is started.

### 5.4.6. Device – Menu



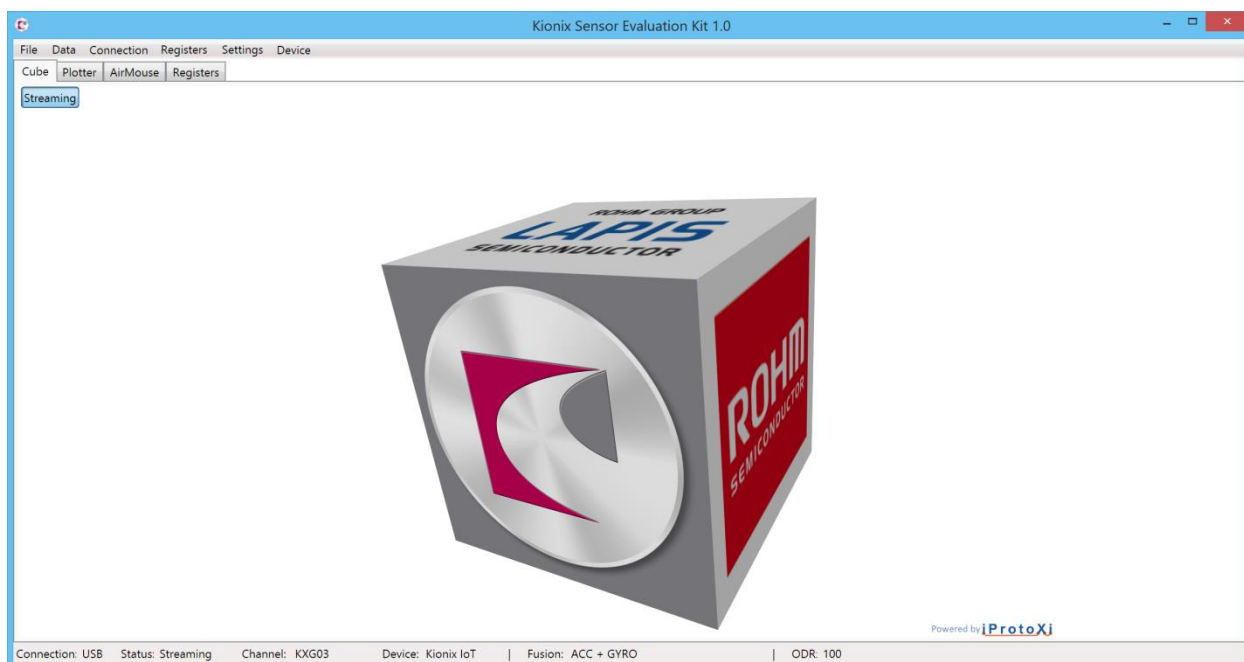
The *Kionix Windows Sensor Evaluation Software* supports *Kionix IoT Sensor Node* device. The type of connected device can be selected from this menu. The default device is `Kionix_IoT`.

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

### 5.5. User Interface - Tabs

Functionalities of the *Kionix Windows Sensor Evaluation Software* are divided between separate tabs.

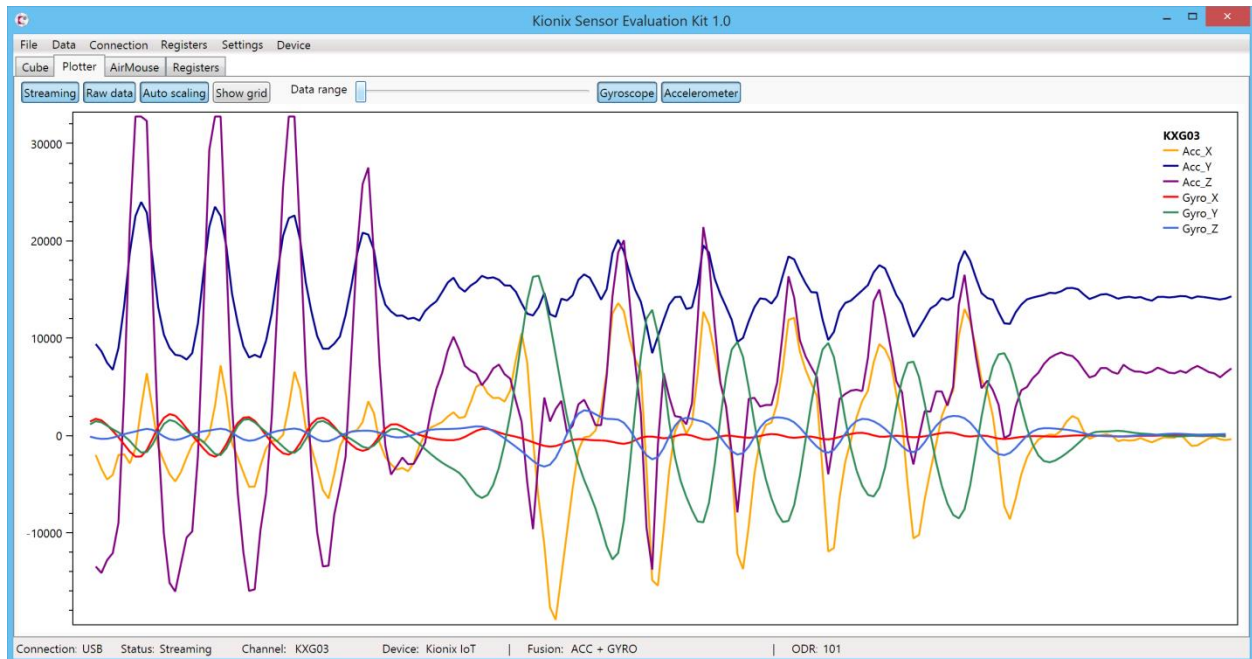
#### 5.5.1. Cube – Tab



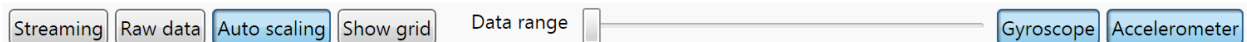
The 3D rotating cube is used for demonstrating sensor fusion algorithm's performance. Sensor fusion can be started by pressing the "Streaming" button.

Please refer to chapter 5.8.1 to get rotation operating correctly.

## 5.5.2. Plotter – Tab



Plotter shows sensor data from current channel.



Plotter has its own `Streaming` and `Raw data` buttons in order to change them quickly.

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

### KXG03

- Acc\_X (m/s<sup>2</sup>)
- Acc\_Y (m/s<sup>2</sup>)
- Acc\_Z (m/s<sup>2</sup>)
- Gyro\_X (degrees/s)
- Gyro\_Y (degrees/s)
- Gyro\_Z (degrees/s)

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

`Show grid` – enables data grid lines. **NOTE:** This will slow down the plotter performance.

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

Plotter has dynamic buttons in order to show/hide data streams inside channel. For example in KXG03 channel there are “Gyroscope” and “Accelerometer” buttons that can be toggled.

### 5.5.2.1. Zooming

You can zoom in and out using mouse scroll button.

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

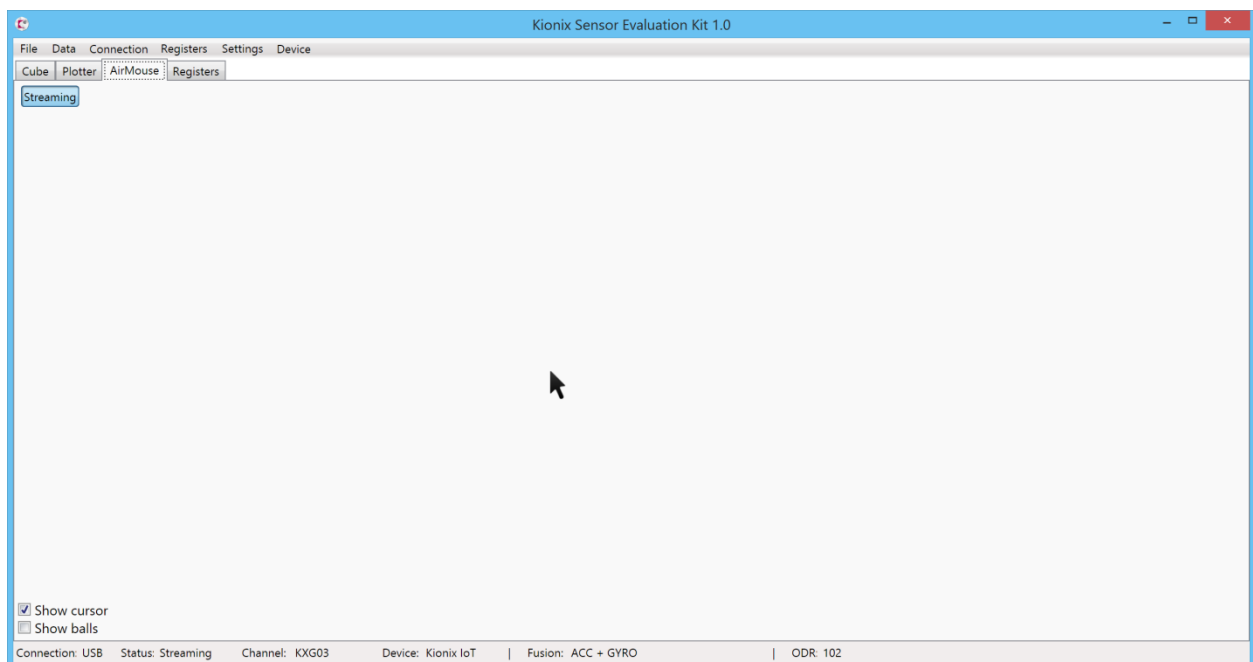
### 5.5.2.2. Pausing

Plotter can be paused to certain a position using the left mouse button.

### 5.5.2.3. Moving

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

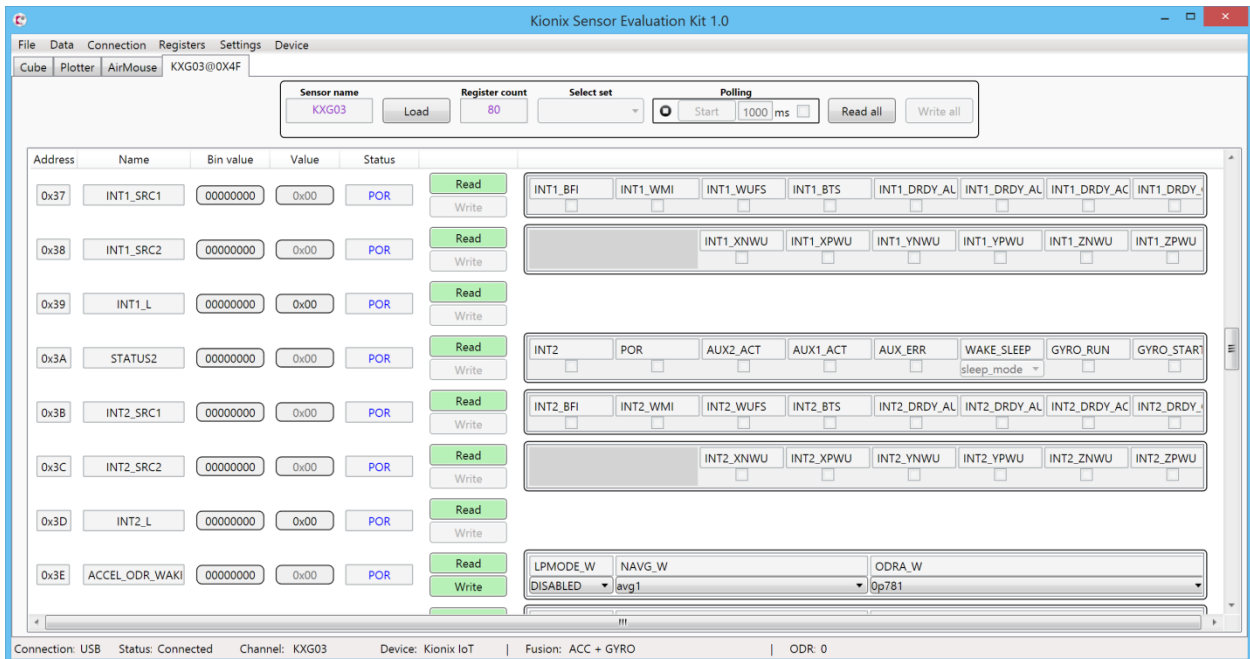
### 5.5.3. Air Mouse – Tab



Air mouse uses sensor fusion for moving the cursor. "Show balls" can be activated to show movement of x, y and z axis. AirMouse can be started by pressing "streaming" button.

Please refer to chapter 5.8.2 to get movement operating correctly.

## 5.5.4. Registers – Tab



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

When opening a new sensor in the register tab, the register content is not updated automatically. Instead, register values can be updated with "Read All" button or "Read" of individual register.

**NOTE:** Streaming is automatically disabled once Register editor tab is selected.

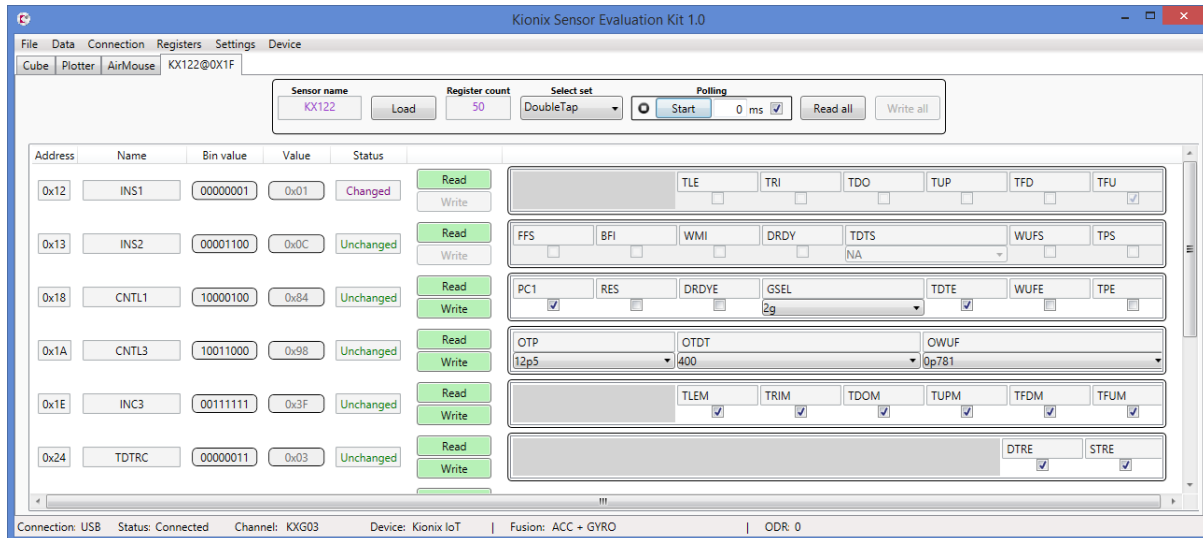
**NOTE:** When changing register values and returning back to Cube or AirMouse, it is possible that changes have been made to the register values that had an effect on data streaming. Please change the data channel to any other channel and then change back to the original channel in order to reset data streaming register values.

### 5.5.4.1. Register set and polling

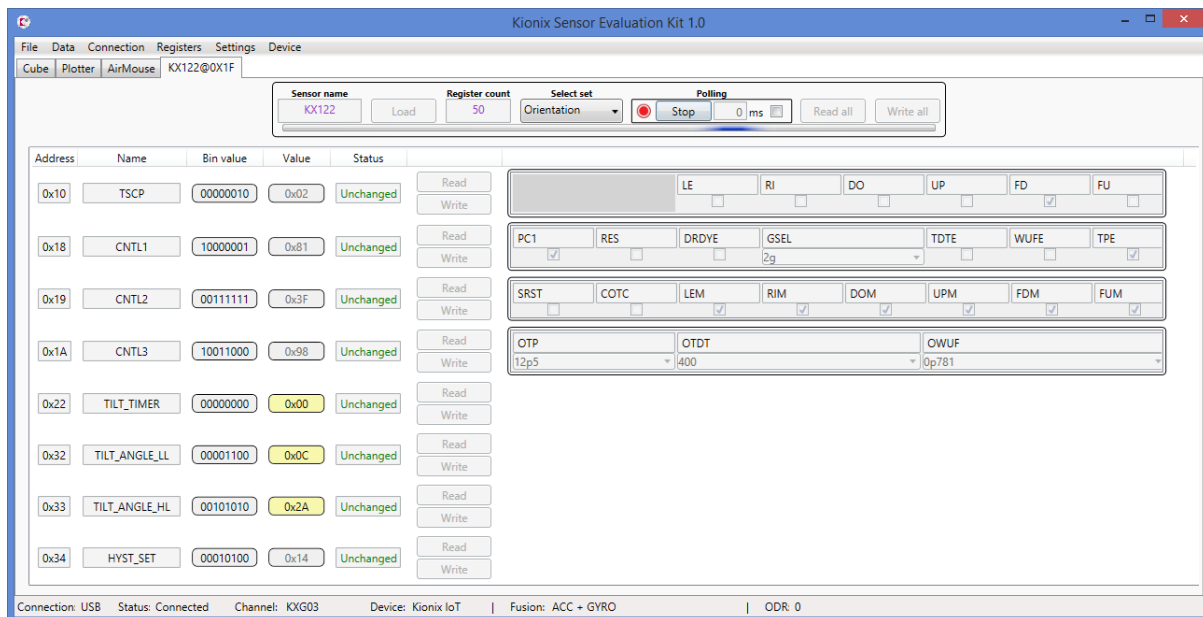
There is also a possibility to create application specific register sets. Using register set, polling can be activated with a specific interval (default interval is 1000ms). The polling feature has an "auto stop" – checkbox, which stops the polling to the first register value change. The below example shows how to monitor double tap detection:

- Select sensor KX122 by pressing "Load" button.
- Select set "Double Tap" from pull down menu
- Check Double tap detection enabled (TDTE) and Power control (PC1) from CNTL1 and press write.
- Tick Check box from "Polling"

- Press "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 is automatically disabled when register polling is started.



Example of double tap detection



Example of orientation detection

### 5.6. User Interface - Status bar

Connection: USB   Status: Streaming   Channel: 9D fusion   Device: Kionix IoT   |   Fusion: ACC + GYRO   |   ODR: 101

Status bar shows current connection and its status. It also keeps the user updated on selected device, sensor fusion and magnetometer calibration status. ODR is visible in the right corner. It can be hidden with "O" – key shortcut if needed.

### 5.7. User Interface - Pop-up windows

Application makes use of pop-up windows to notify about important actions. This sections provides detailed information about pop-up windows.

#### 5.7.1. Streaming pop-up window

Streaming pop-up window is showed in Cube, Plotter, and AirMouse to notify the user about data stream enabling.

Streaming can be enabled with specific "Streaming" – button, from Data->Streaming – menu or with shortcut "CTRL + S"

**Please enable streaming to activate Cube movement!**

#### 5.7.2. Orientation reset pop-up window

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

**Please reset the orientation with space button!**

#### 5.7.3. Magnetometer calibration pop-up window

Magnetometer calibration pop-up is showed in Cube and AirMouse when channel including magnetometer is used and calibration status is 0.

Calibration is done by performing "8" – like movement with sensor.

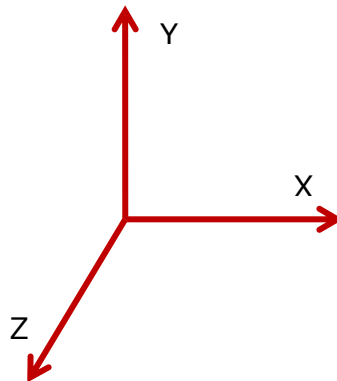
**Please calibrate the magnetometer!**

### 5.8. Orientation reset

#### 5.8.1. Cube reset

In order to reset 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 point to the right
2. Press "Space" – button
3. Cube is now reset to *Kionix IoT Sensor Nodes* position and Kionix logo is the front side





### 5.8.2. AirMouse reset

In order to reset AirMouse 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 "Space" – button
3. AirMouse is now reset to *Kionix IoT Sensor Nodes* position



### 5.9. Shortcuts

The *Kionix Windows Sensor Evaluation Software* 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
CTRL + A	=	Activate KX122 accelerometer stream and enable streaming
CTRL + M	=	Activate KMX62 magnetometer stream and enable streaming
CTRL + G	=	Activate KXG03 gyro stream and enable streaming
CTRL + 9	=	Activate KMX62_KXG03 (9D fusion) stream and enable streaming
CTRL + B	=	Activate bm1383glv barometer stream and enable streaming
CTRL + R	=	Reset used connection (disconnect & connect when having connection problem)

## 6. Kionix Multi-OS Evaluation Software

### 6.1. Introduction

The *Kionix Multi-OS Evaluation Software* offers quick access to sensor register level functionality. Simple and minimalistic demonstrator applications are provided for testing sensor features. These will help to test and evaluate sensor functionality and provide needed information on how to implement sensor driver SW.

### 6.2. Set Up

#### 6.2.1. Installation

The *Kionix Multi-OS Evaluation Software* is located in the following directory:  
Kionix-IoT-Evaluation-Kit\Kionix-Multi-OS-Evaluation-Software

*Kionix Multi-OS Evaluation Software* requires:

- Python interpreter installation
- Drivers for utilized bus adapters, e.g.:
  - Aardvark I2C/SPI adapter
  - FTDI – USB serial
  - BLE
  - Embedded Linux

Chapter 4 describes the install processes.

#### 6.2.2. Configuration

The *Kionix Multi-OS Evaluation Software* settings are in the `settings.cfg` file located in the root directory of *Kionix Multi-OS Evaluation Software*. The settings file uses windows `.ini` file syntax. Semicolon ";" at the beginning of the line indicate a comment.

There are settings for both:

- *Kionix Multi-OS Evaluation Software* framework and
- Test applications included in *Kionix Multi-OS Evaluation Software*

##### 6.2.2.1. Connection to Kionix Evaluation Kit HW

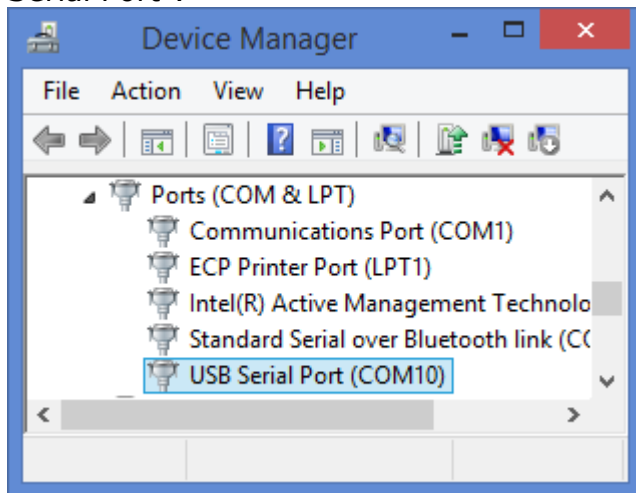
The most important framework level setting is connection. It defines the bus adapter which is used. The definition is in section `[connection]`.

```
[connection]
; Bus index selection
; 1 Aardvark with I2C
; 2 Aardvark with SPI
; 3 BLE connection using "Kionix BLE Router"
; 4 direct i2c connection through Embedded Linux GPIO
```

```
;bus_index          = 3
bus_index           = serial_com_kx_iot
```

Key `bus_index` defines the bus adapter to be used. Default connection is `serial_com_kx_iot` e.g. *Kionix IoT Sensor Node* connected to PC with USB cable.

Bus adapters with names starting "`serial_com_`" connect to sensor through windows USB serial port. For example *Kionix IoT Sensor Node* is listed as "USB Serial Port".



Corresponding COM port needs to be written in `settings.cfg` file:

```
[serial_com_kx_iot]
com_port           = COM10
```

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

### 6.2.2.2. Generic settings

Section `[generic]` contains connection independent settings. These include debug logging settings and interrupt related settings.

Debug logging level is defined with `logging_level` and default level is `INFO`:

```
logging_level = INFO
```

Other alternatives are: `DEBUG`, `INFO`, `WARNING`, `ERROR` and `CRITICAL`. Interrupt related configurations are explained in following chapters.

### 6.2.2.3. Data Ready operation settings

There are four options for data ready operations.

```
;drdy_operation     = ADAPTER_GPIO1_INT
;drdy_operation     = ADAPTER_GPIO2_INT
```

```
;drdy_operation      = DRDY_REG_POLL
drdy_operation       = INTERVAL_READ
```

The default setting used for `drdy_operation` is `INTERVAL_READ`. Interval is defined in `drdy_poll_interval` and the default value is set to 40ms. If using `INTERVAL_READ` or `DRDY_REG_POLL`, then it is not mandatory to connect interrupt lines between the sensor and Bus adapter and there is no need to configure the GPIO numbers for interrupt lines as described in chapter 6.2.2.5.

When using the `DRDY_REG_POLL` option then the data ready bit is monitored from the status register to trigger sensor data reading. This will introduce more traffic on the bus and thus reduce ODR but also verify that all data is read from the sensor.

### 6.2.2.4. Interrupt polarity

Interrupts can be configured either active LOW or active HIGH. Active LOW is used by default since Kionix IoT board requires active low interrupts.

```
;use active low as default
int1_active_high      = FALSE                      ; TRUE / FALSE
int2_active_high      = FALSE                      ; TRUE / FALSE
```

### 6.2.2.5. Interrupt lines

If the test application uses sensor's interrupt lines then also interrupt line mapping to HW board GPIO lines must be configured properly. Default options are listed in the `settings.cfg` for example:

```
pin1_index = 16 ; PAD TEST 5, int1 of kxg03, int 1 of kx122
pin2_index = 13 ; PAD TEST 4, int2 of kxg03 and int2 of kx122
```

Example: If INT2 signal of KX122 is used in test application for signaling data ready interrupt then corresponding data ready operation setting is:

```
drdy_operation      = ADAPTER_GPIO2_INT
```

Sometimes pin index must be changed depending on how sensor interrupt lines are connected to the microcontroller. Please refer to chapter 2 for details.

## 6.3. Getting Started

The quickest way to test *Kionix Multi-OS Evaluation Software* is to run the `hello_sensor.py` test application located at `Kionix-IoT-Evaluation-Kit\Kionix-Multi-OS-Evaluation-Software\hello_sensor.py`

The default connection is `serial_com_kx_iot` e.g. The *Kionix IoT Sensor Node* is connected to the PC with a USB cable (USB driver installation described in 4.1.3).

Executing `hello_sensor.py`. The *Kionix Multi-OS Evaluation Software* will find the right USB serial port, looks for all supported sensors from bus I2C and prints 10 data samples from each. An example output is shown below. NOTE: if running on Linux please refer to chapter 6.2.2.1.

```
Kionix-IoT-Evaluation-Kit\Kionix-Multi-OS-Evaluation-Software>python hello_sensor.py
INFO : Bus serial_com_kx_iot selected
INFO : Automatic port choice: COM36
INFO : Looking for sensor with "bh1745_driver".
INFO : Looking for sensor with "bml383glv_driver".
INFO : Looking for sensor with "bh1730_driver".
INFO : Looking for sensor with "kx022_driver".
INFO : KX112/KX122/KX123/KX124 found
INFO : Sensor found at slave address 0x1f
Reading 10 data samples with kx022_driver.
0 (168, 30, -16420)
1 (174, 39, -16419)
2 (178, 26, -16409)
3 (185, 27, -16410)
4 (182, 35, -16410)
5 (171, 36, -16408)
6 (169, 29, -16407)
7 (177, 27, -16408)
8 (181, 37, -16407)
9 (173, 43, -16410)
INFO : Looking for sensor with "kxg03_driver".
INFO : kxg03 found
INFO : Sensor found at slave address 0x4f
Reading 10 data samples with kxg03_driver.
0 (-4824, -16463, 32767, -735, -724, 15261)
1 (-745, -952, 8023, -772, -610, 15164)
2 (-374, 1284, 4276, -771, -667, 15216)
3 (-282, 1560, 3136, -827, -667, 15230)
4 (-320, 1003, 2240, -768, -737, 15283)
5 (-340, 937, 2051, -856, -689, 15216)
6 (-332, 870, 1962, -809, -585, 15153)
7 (-352, 856, 1878, -724, -596, 15123)
8 (-336, 850, 1818, -862, -691, 15204)
9 (-355, 845, 1808, -769, -629, 15098)
INFO : Looking for sensor with "kxg08_driver".
INFO : Looking for sensor with "kxm62_driver".
INFO : KMX62 found
INFO : Sensor found at slave address 0x0e
Reading 10 data samples with kxm62_driver.
0 (197, 625, 16209, 977, -1099, -635)
1 (204, 665, 16225, 990, -1080, -642)
2 (203, 646, 16233, 990, -1074, -637)
3 (187, 631, 16212, 990, -1077, -634)
4 (213, 647, 16226, 980, -1082, -637)
5 (209, 658, 16222, 980, -1087, -638)
6 (202, 655, 16207, 980, -1081, -637)
7 (210, 650, 16222, 975, -1085, -638)
8 (215, 660, 16229, 968, -1096, -651)
9 (203, 658, 16196, 977, -1083, -642)
INFO : Looking for sensor with "kxtj2_driver".
INFO : Looking for sensor with "kxtj3_driver".
INFO : Looking for sensor with "kxcnl_driver".
```

### 6.4. File structure of the evaluation kit

The overall structure of the *Kionix Multi-OS Evaluation Software* is the following:

```
Kionix-Multi-OS-Evaluation-Software/
settings.cfg
hello_sensor.py
plot.py
validate.py
connection_setup.py
settings.cfg
```

```
|—bh1730
|—bh1745
|—bm1383aglv
|—bm1383glv
|—kmx62
|—kx022_kx122
|—kxcn1
|—kxg03
|—kxg08
|—kxtj2
|—kxtj3
|—lib
|—license
```

Each sensor has its own directory containing:

- Register definitions (for example `kx022_registers.py`)
- Reference driver implementation (for example `kx022_driver.py`)
- Test application for reading sensor data (for example `kx022_data_logger.py`)
- Possible other test applications

Lib directory contains *Kionix Multi-OS Evaluation Software* middleware.

### 6.5. Running test applications

Test applications are sensor specific and located in corresponding their folders. For example: data logging application for KX022 is located at `Kionix-Multi-OS-Evaluation-Software\kx022_kx122` application and it is executed with command:

```
python kx022_data_logger.py.
```

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

```
INFO : Bus serial_com_kx_iot selected
INFO : Automatic port choice: COM7
INFO : KX112/KX122/KX123/KX124 found
INFO : Sensor found at slave address 0x1f
INFO : Kionix IoT node init start
INFO : kxg03 found
INFO : Sensor found at slave address 0x4f
INFO : KX112/KX122/KX123/KX124 found
INFO : Sensor found at slave address 0x1f
INFO : Sensor found at slave address 0x5d
INFO : Kionix IoT board init done
0.040684      847      315      -16345
0.084989      874      298      -16336
0.130157      851      296      -16339
0.175024      823      297      -16335
0.220869      823      294      -16324
0.266605      835      289      -16323
0.310884      845      295      -16324
0.356687      846      306      -16314
0.400454      847      312      -16324
0.445519      863      324      -16322
0.491399      874      324      -16317
```

Sensor data can be directed also to file:

```
python kx022_data_logger.py > kx022.txt
```

#### 6.5.1. High ODR logging

By default the *Kionix Multi-OS Evaluation Software* uses “polling” monitoring DRDY bit or corresponding interrupt line. When using the *Kionix IoT Evaluation Kit* with a bus adapter which uses *Kionix Evaluation Kit Firmware* then it is possible to achieve higher ODRs. This functionality requires that interrupt line(s) from the sensor are connected to the Bus adapter and interrupt lines are configured properly. Ref chapter 6.2.2.

The selected data logger test scripts are supporting this feature. The feature is enabled with `-s` switch.

```
python kx022_data_logger.py -s -l 4000 > kx022.txt
```

#### 6.5.2. Other provided tools

It is possible to verify the log validity:

```
kx022_kx122>..\validate.py kx022.txt
start time 0.00990 (s)
stop time 9.99000 (s)
duration 9.98010 (s)
samples 4000
max delta 5.16900 (ms) / 193 (Hz) at sample 2148
```

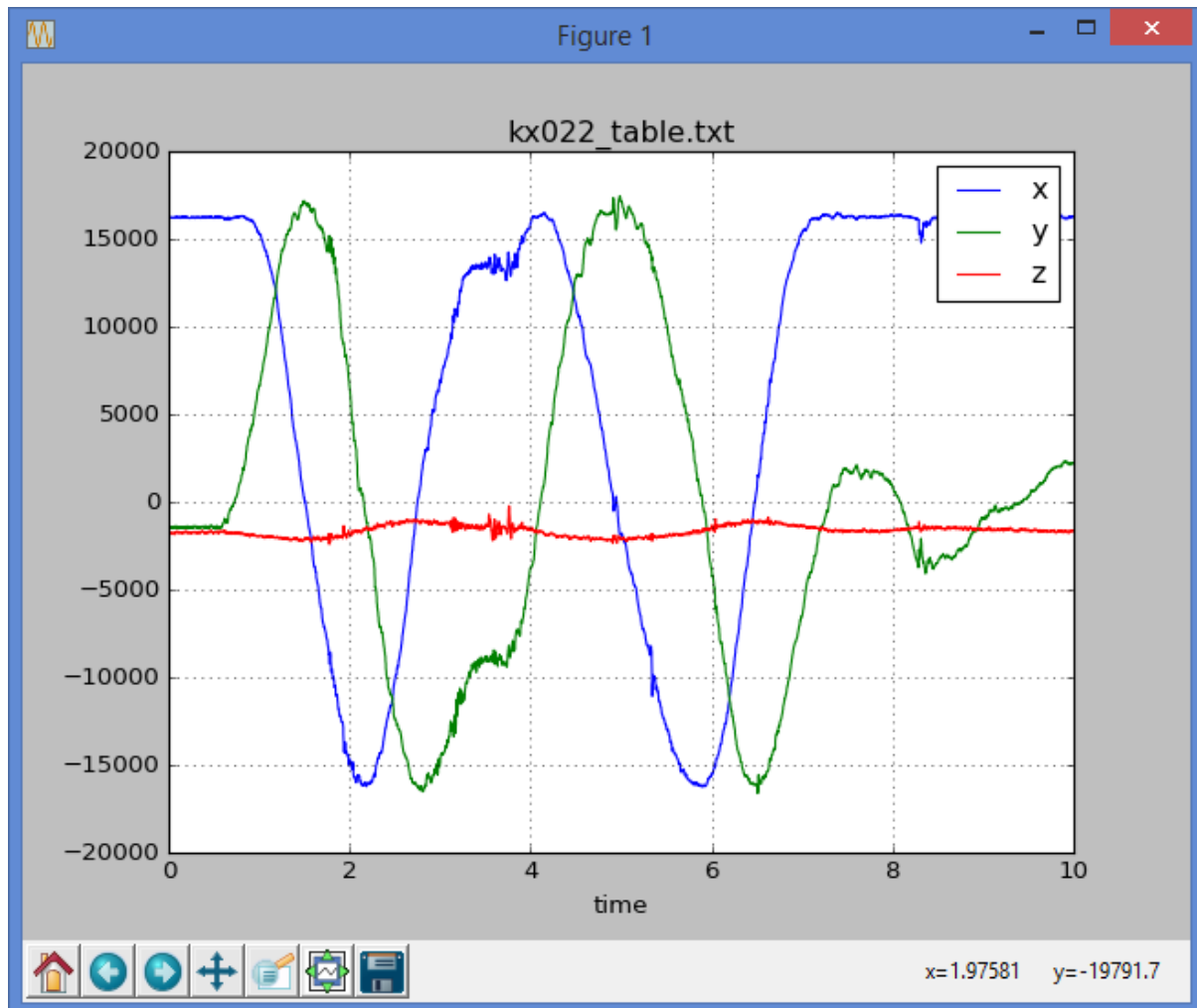


```
min delta  0.03900 (ms) / 25641 (Hz) at sample 692
avg delta   2.49502 (ms)
ODR avg     400.80 (Hz)
```

NOTE: timestamping is done on PC and it is not accurate with high ODRs. This causes an impact to delta time statistics.

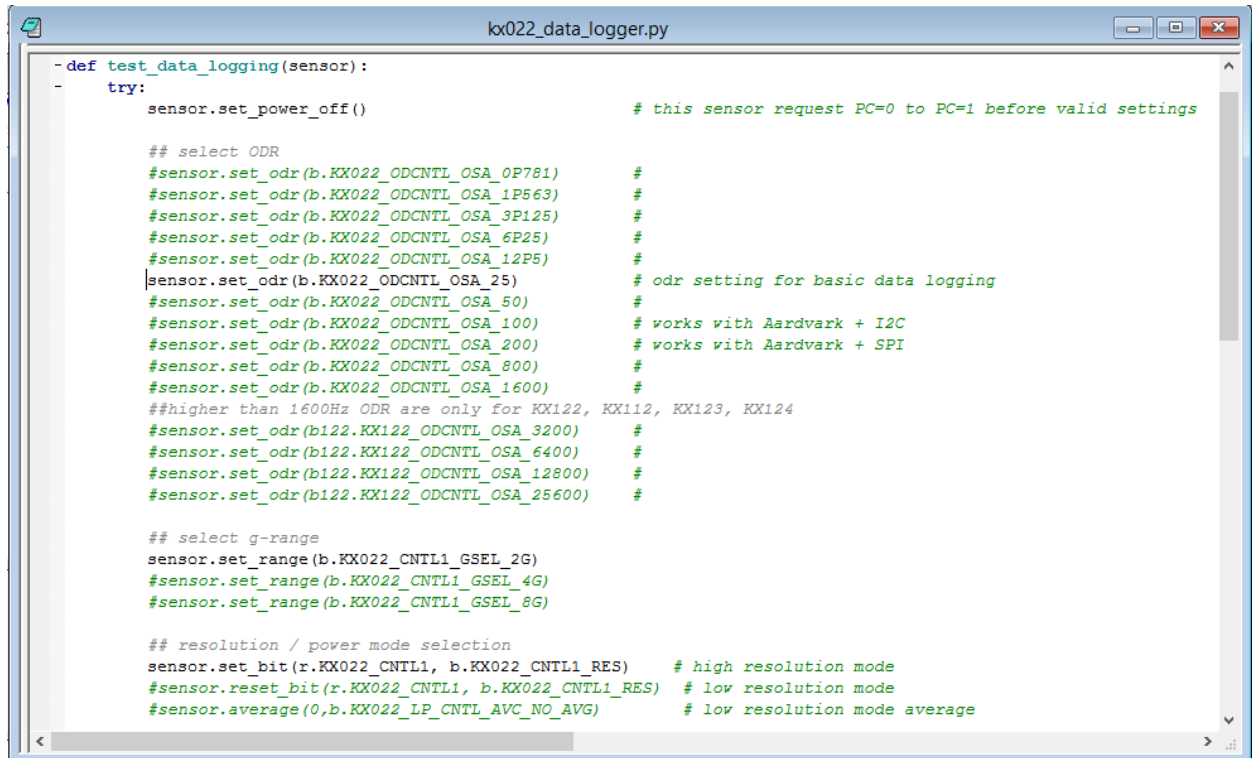
It is possible to view the recorded log with the plot.py application:

```
kx022_kx122>..\plot.py kx022.txt -t -l x y z
```



## 6.6. Changing test application configuration

Test applications list most common sensor configuration alternatives, including g-range and ODR settings. The required setting can be selected by commenting / un-commenting the corresponding lines of code:



```

kx022_data_logger.py

def test_data_logging(sensor):
    try:
        sensor.set_power_off()                # this sensor request PC=0 to PC=1 before valid settings

        ## select ODR
        #sensor.set_odr(b.KX022_ODCNTL_OSA_0P781)    #
        #sensor.set_odr(b.KX022_ODCNTL_OSA_1P563)    #
        #sensor.set_odr(b.KX022_ODCNTL_OSA_3P125)    #
        #sensor.set_odr(b.KX022_ODCNTL_OSA_6P25)     #
        #sensor.set_odr(b.KX022_ODCNTL_OSA_12P5)     #
        sensor.set_odr(b.KX022_ODCNTL_OSA_25)        # odr setting for basic data logging
        #sensor.set_odr(b.KX022_ODCNTL_OSA_50)        #
        #sensor.set_odr(b.KX022_ODCNTL_OSA_100)       # works with Aardvark + I2C
        #sensor.set_odr(b.KX022_ODCNTL_OSA_200)       # works with Aardvark + SPI
        #sensor.set_odr(b.KX022_ODCNTL_OSA_800)       #
        #sensor.set_odr(b.KX022_ODCNTL_OSA_1600)      #
        ##higher than 1600Hz ODR are only for KX122, KX112, KX123, KX124
        #sensor.set_odr(b122.KX122_ODCNTL_OSA_3200)   #
        #sensor.set_odr(b122.KX122_ODCNTL_OSA_6400)   #
        #sensor.set_odr(b122.KX122_ODCNTL_OSA_12800)  #
        #sensor.set_odr(b122.KX122_ODCNTL_OSA_25600)  #

        ## select g-range
        sensor.set_range(b.KX022_CNTL1_GSEL_2G)
        #sensor.set_range(b.KX022_CNTL1_GSEL_4G)
        #sensor.set_range(b.KX022_CNTL1_GSEL_8G)

        ## resolution / power mode selection
        sensor.set_bit(r.KX022_CNTL1, b.KX022_CNTL1_RES) # high resolution mode
        #sensor.reset_bit(r.KX022_CNTL1, b.KX022_CNTL1_RES) # low resolution mode
        #sensor.average(0,b.KX022_LP_CNTL_AVC_NO_AVG)     # low resolution mode average
  
```

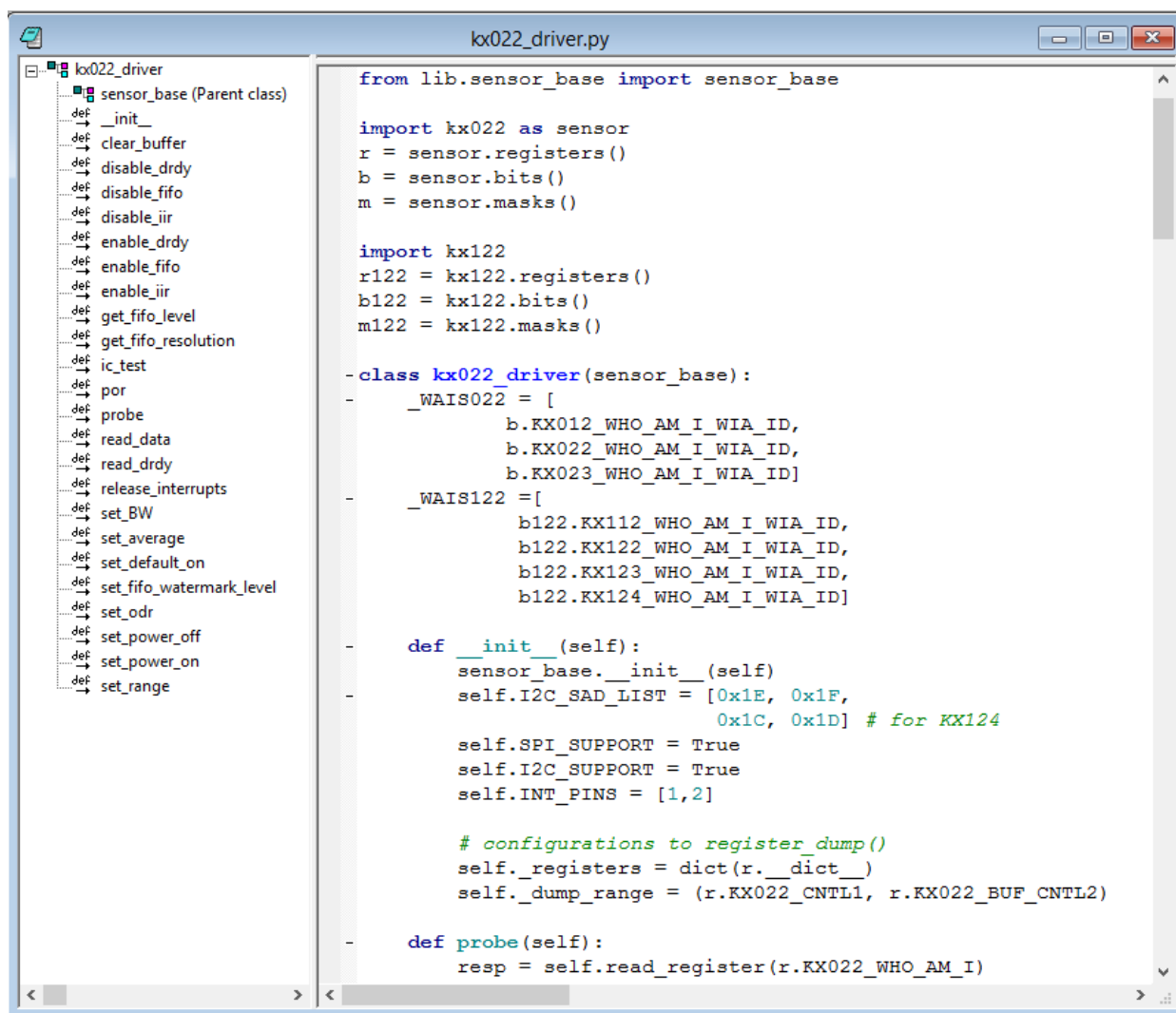
If the needed option is not listed, it can be easily added by referencing the corresponding register definition header file.

### 6.7. Reference driver implementation

The *Kionix Multi-OS Evaluation Software* offers platform independent reference sensor driver software. Since only sensor related configurations are in the driver and platform specific dependencies are not visible, it gives a good starting point for porting the driver software to a desired target platform.

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

```
sensor_base
├── object (Parent class)
│   ├── def __init__
│   ├── def _poll_delay
│   ├── def _poll_drdy_register
│   ├── def _poll_gpio_line1
│   ├── def _poll_gpio_line2
│   ├── def assign_bus
│   ├── def bus_poll_gpio
│   ├── def ic_test
│   ├── def is_use_adapter_int_pins_enabled
│   ├── def por
│   ├── def probe
│   ├── def read_data
│   ├── def read_drdy
│   ├── def read_register
│   ├── def register_dump
│   ├── def register_dump_listed
│   ├── def register_dump_range
│   ├── def reset_bit
│   ├── def set_bit
│   ├── def set_bit_pattern
│   ├── def set_default_on
│   └── def write_register
```



```

kx022_driver
├── sensor_base (Parent class)
│   ├── def __init__
│   ├── def clear_buffer
│   ├── def disable_drdy
│   ├── def disable_fifo
│   ├── def disable_iir
│   ├── def enable_drdy
│   ├── def enable_fifo
│   ├── def enable_iir
│   ├── def get_fifo_level
│   ├── def get_fifo_resolution
│   ├── def ic_test
│   ├── def por
│   ├── def probe
│   ├── def read_data
│   ├── def read_drdy
│   ├── def release_interrupts
│   ├── def set_BW
│   ├── def set_average
│   ├── def set_default_on
│   ├── def set_fifo_watermark_level
│   ├── def set_odr
│   ├── def set_power_off
│   ├── def set_power_on
│   └── def set_range
└── kx022_driver
    ├── from lib.sensor_base import sensor_base
    ├── import kx022 as sensor
    ├── r = sensor.registers()
    ├── b = sensor.bits()
    ├── m = sensor.masks()
    ├── import kx122
    ├── r122 = kx122.registers()
    ├── b122 = kx122.bits()
    ├── m122 = kx122.masks()
    ├── - class kx022_driver(sensor_base):
    │   ├── - _WAIS022 = [
    │   │   ├── b.KX012_WHO_AM_I_WIA_ID,
    │   │   ├── b.KX022_WHO_AM_I_WIA_ID,
    │   │   └── b.KX023_WHO_AM_I_WIA_ID]
    │   ├── - _WAIS122 = [
    │   │   ├── b122.KX112_WHO_AM_I_WIA_ID,
    │   │   ├── b122.KX122_WHO_AM_I_WIA_ID,
    │   │   ├── b122.KX123_WHO_AM_I_WIA_ID,
    │   │   └── b122.KX124_WHO_AM_I_WIA_ID]
    │   ├── def __init__(self):
    │   │   sensor_base.__init__(self)
    │   │   self.I2C_SAD_LIST = [0x1E, 0x1F,
    │   │   │   0x1C, 0x1D] # for KX124
    │   │   self.SPI_SUPPORT = True
    │   │   self.I2C_SUPPORT = True
    │   │   self.INT_PINS = [1,2]
    │   │   # configurations to register_dump()
    │   │   self._registers = dict(r.__dict__)
    │   │   self._dump_range = (r.KX022_CNTL1, r.KX022_BUF_CNTL2)
    │   ├── def probe(self):
    │   │   resp = self.read_register(r.KX022_WHO_AM_I)
    
```

## 7. Troubleshooting and known issues

### 7.1. Communications

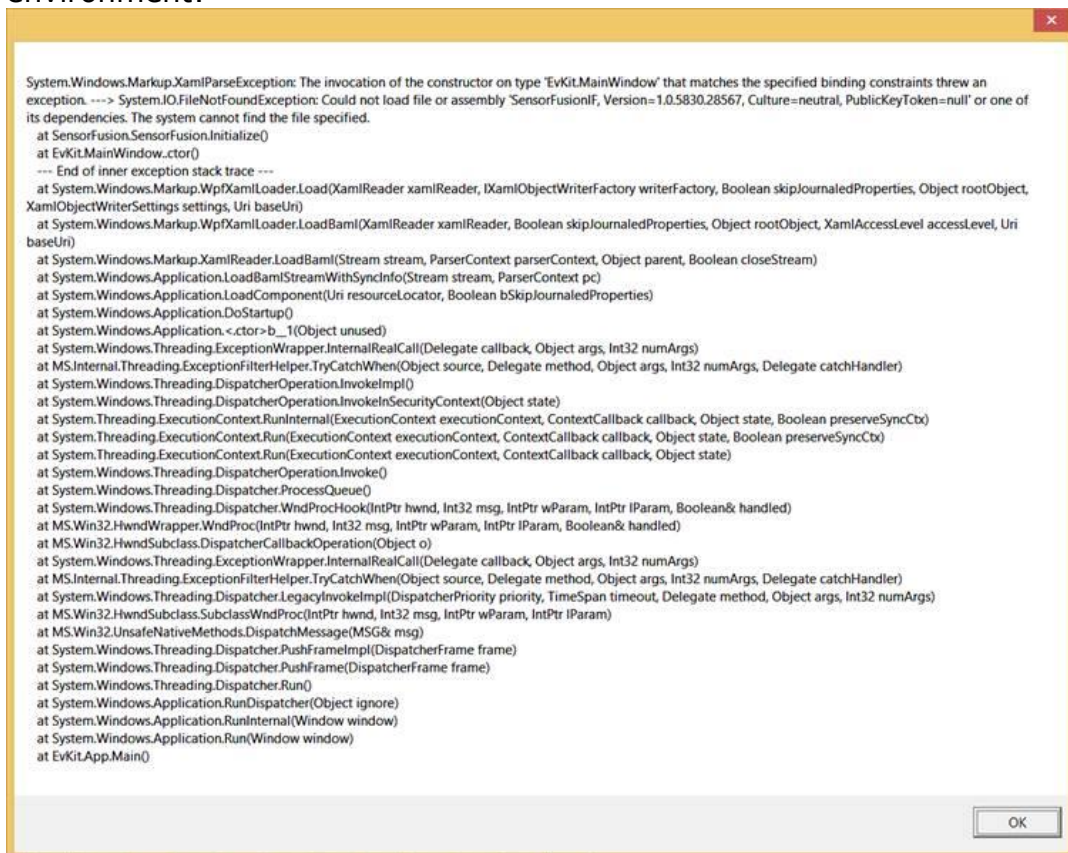
- USB communication may miss sensor data samples or the USB connection is lost randomly: Use good quality USB cables which are USB certified.



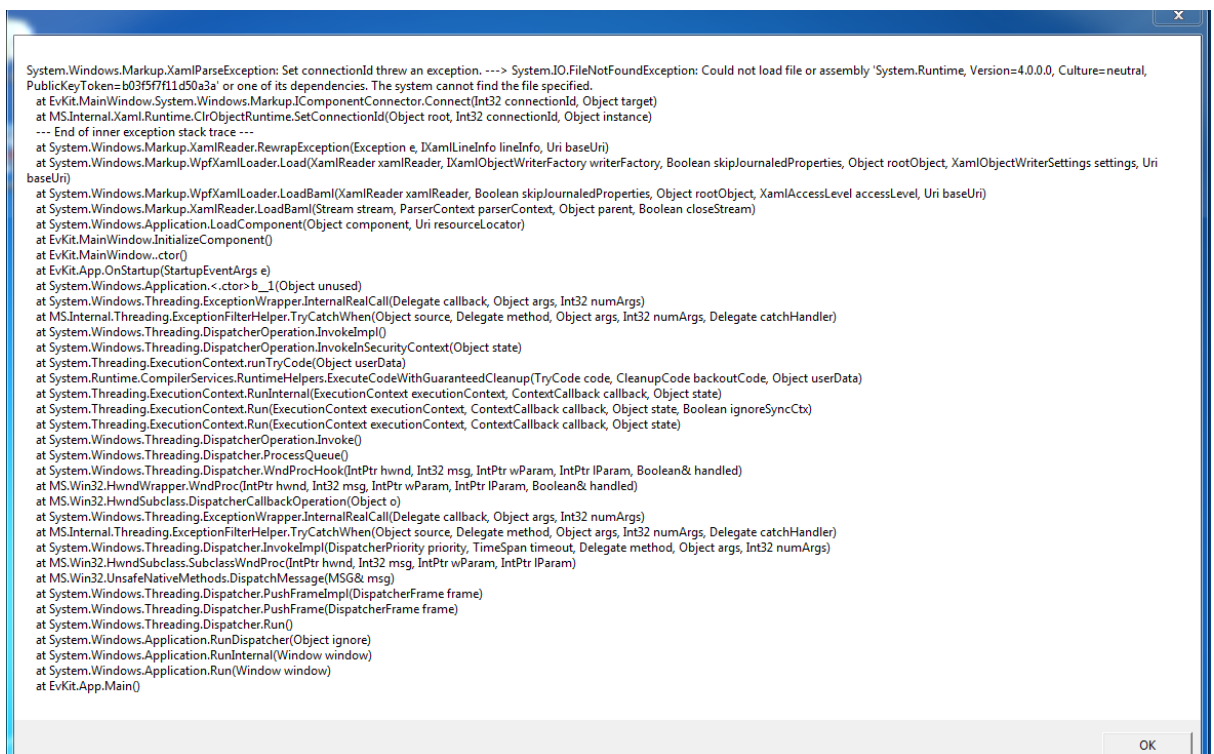
- The *Kionix Evaluation Kit SW* may not connect to the *Kionix IoT Sensor Node* when USB connection is used and the battery is connected to the *Kionix IoT Sensor Node*: the *Kionix IoT Sensor Node* must be powered on before connecting it to the PC with a USB cable. Otherwise the FTDI IC may get powered on slightly before nRF51822 SoC and thus may not detect nRF51822 SoC properly.
- Windows firewall may block TCP/IP port used by *Kionix BLE Router*.

### 7.2. Kionix Windows Sensor Evaluation Software

- If you get this kind of error message, please install c++ runtime environment:



- When running the application for the first time Windows may inform 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 selecting ANDROID\_BLE connection, a pop-up reading "please enable streaming..." is shown even when the *Kionix BLE Router* is not running in the Android device.
- If this kind of an error message appears, the Windows .NET installation is not up to date. Please run Windows update to in order to solve this.

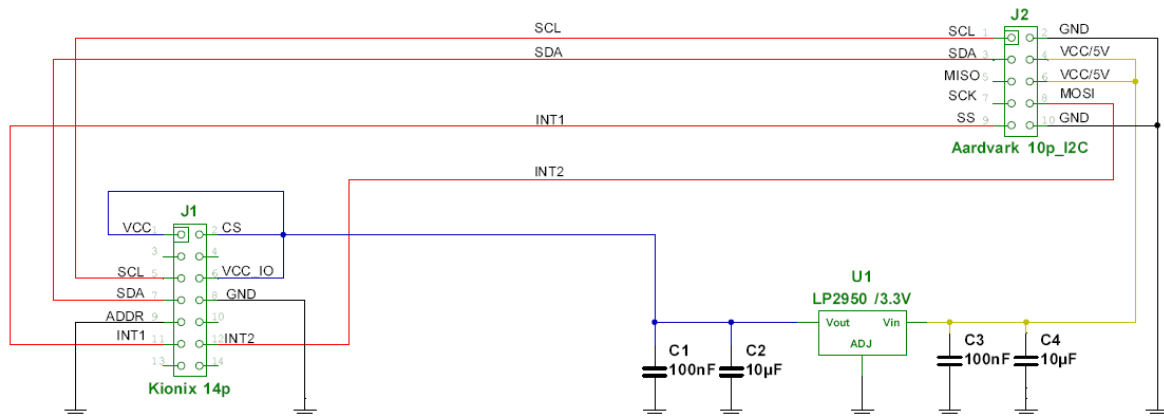


### 7.3. Kionix Multi-OS Evaluation Software

- In case there are issues with *Kionix Multi-OS Evaluation Software* operations, it is possible to change logging level from default ERROR for example to DEBUG.
- `logging_level = ERROR ; DEBUG / INFO / WARNING / ERROR / CRITICAL`
- Additionally logs can be saved to file by defining file name for key `log_file`, which is empty (e.g. no logging to file) by default.
- The *Kionix Multi-OS Evaluation Software* verifies that it can capture all interrupts (before starting to monitor interrupt line it is first verified that interrupt is not yet triggered). If `logging_level` is set to `WARNING` or higher then warning printout is given if interrupt speed (for example ODR) is too big.
- If the driver or lib code have been modified and execution fails to import error, delete all \*.pyc files.

## 8. Appendix

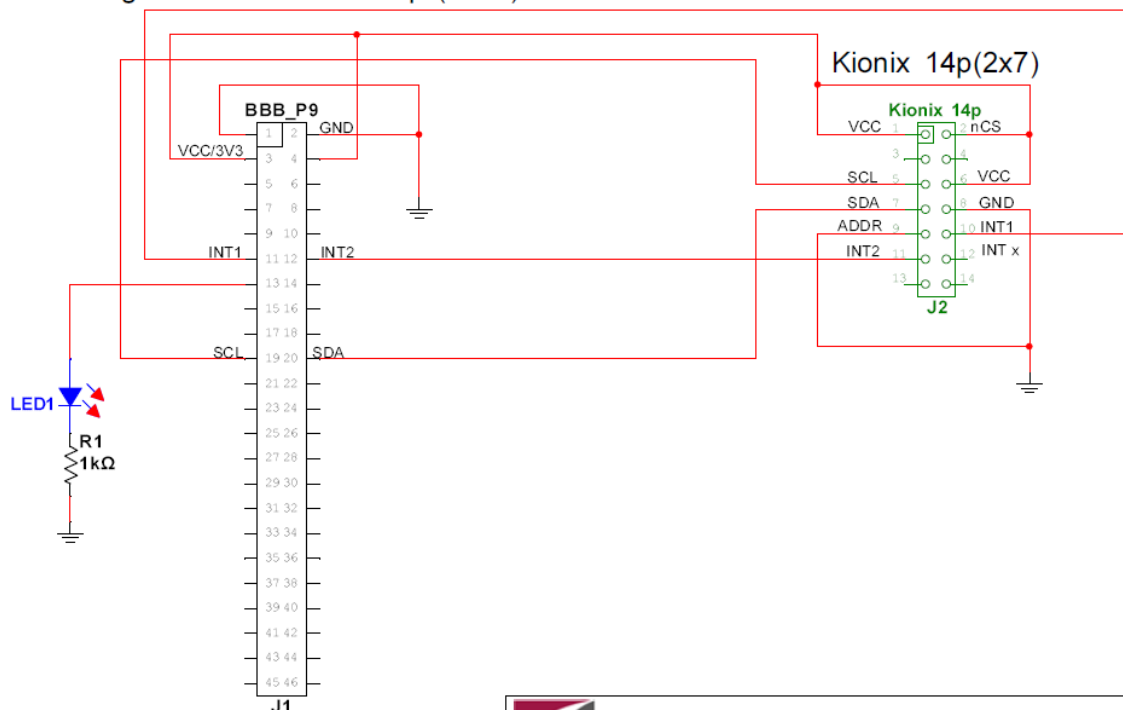
### 8.1. Connecting Kionix Evaluation Board to Aardvark I2C



Example: setup for KXG03		
Title: Simple Aardvark w Kionix	Desc.: Aardvark_I2C_Kionix_S	
Designed by: VKo	Document No:	Revision: A01
Checked by:	Date: 28.4.2016	Size: A4
Approved by:	Sheet 1 of 1	for I2C bus control

### 8.2. Connecting Kionix Evaluation Board to Beagle Bone Black

BeagleBone Black P9 /46p (2x23)



Kionix		
Title: BBB_Kionix	Desc.: Kionix14p_to_BBB	
Designed by: VKo	Document No:	Revision:
Checked by:	Date: 4.8.2015	Size: A4
Approved by:	Sheet 1 of 1	