InvenSense

Software User Guide

For

ICM-20602 eMD

1 TABLE OF CONTENTS

Useful Links 3
Additional Documents 3
1 Overview 4
1.1 Introduction 4
1.2 ICM-206xx eMD Basics 4
1.2.1 ICM-20602 4
2 Hardware Platform 5
2.1 ST NUCLEO F411-RE setup 5
2.2 Nucleo Carrier Board setup 5
2.2.1 Pluging-in Sensor Daughter Boards 6
2.2.2 Powering the NUCLEO and Carrier board 7
2.2.3 Get traces on the NUCLEO and Carrier board 7
3 Software Environment 8
3.1 Prerequisite 8
3.2 eMD Developer Kit InvenSense packages 8
3.3 FW Package description 8
4 Building and running samples applications 9
4.1 Overview 9
4.2 Building example applications 9
4.2.1 Running the IDD-ICM20602 FULL example application 9
4.2.2 Running IDD-ICM20602 LITE example application 13
5 Document Information 16
5.1 Revision History 16

USEFUL LINKS

InvenSense website:

http://www.InvenSense.com/

ST website:

http://www.st.com/web/catalog/tools/FM116/SC959/SS1532/LN1847?sc=stm32nucleo

http://www.st.com/web/en/catalog/tools/PF258168

IAR website:

https://www.iar.com/iar-embedded-workbench/

ADDITIONAL DOCUMENTS

Tools User Guide for eMD

Software Metrics for ICM-20602 eMD

Application Note / Internal Development Guide for ICM-20602 eMD

1 OVERVIEW

The purpose of this document is to give an overview of the ICM-20602 eMD Developer Kit that will allow users to create an application based on motion sensors. This document may also serve as a quick start guide for the ICM-20602 package and its elements, including setup and how use the sample applications provided.

1.1 INTRODUCTION

The ICM-20602 eMD solution is compatible with the ST Nucleo board based on a STM32F411RE. The supported development tools are IAR Embedded Workbench. The purpose of this solution is to allow sensor management and algorithm processing by using a standalone microcontroller. The ICM-20602 eMD solution is an embedded sensors combo (accelerometer & gyroscope) on chip, easy to integrate for users developing in wearable and IoT space. The Developer's Kit's includes a full sensor software solution.

1.2 ICM-206XX EMD BASICS

1.2.1 ICM-20602

The ICM-20602 chip contains an Accelerometer and a Gyroscope accessible through SPI or I2C.

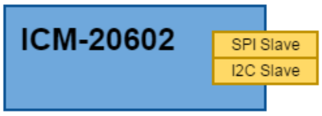


Figure 1 – ICM-20602 eMD Architecture

2 HARDWARE PLATFORM

The eMD platform for ICM-20602 consists in the following components:

- ST NUCLEO F411-RE board including a STM32 F411 MCU
- InvenSense Nucleo Carrier Board rev B or rev C
- InvenSense sensor Daughter Boards (aka DB) for ICM-20602 6-axis sensor
- Optionally, sensor Daughter Boards for AK09911,AK09912 or AK09915 magnetometer

2.1 ST NUCLEO F411-RE SETUP

The **ST Nucleo F411-RE** includes a STMF411 standalone microcontroller. For more information about the ST Nucleo board, please refer to ST website (see *Useful Links* section above.)

You will find the ST Link drivers to install on your PC here: http://www2.st.com/content/st_com/en/products/embedded-software/development-tool-software/stsw-link009.html

Required jumper configuration is for NUCLEO is as follow:

JP1	Open
JP5 (PWR)	(U5V)
JP6 (IDD)	Closed

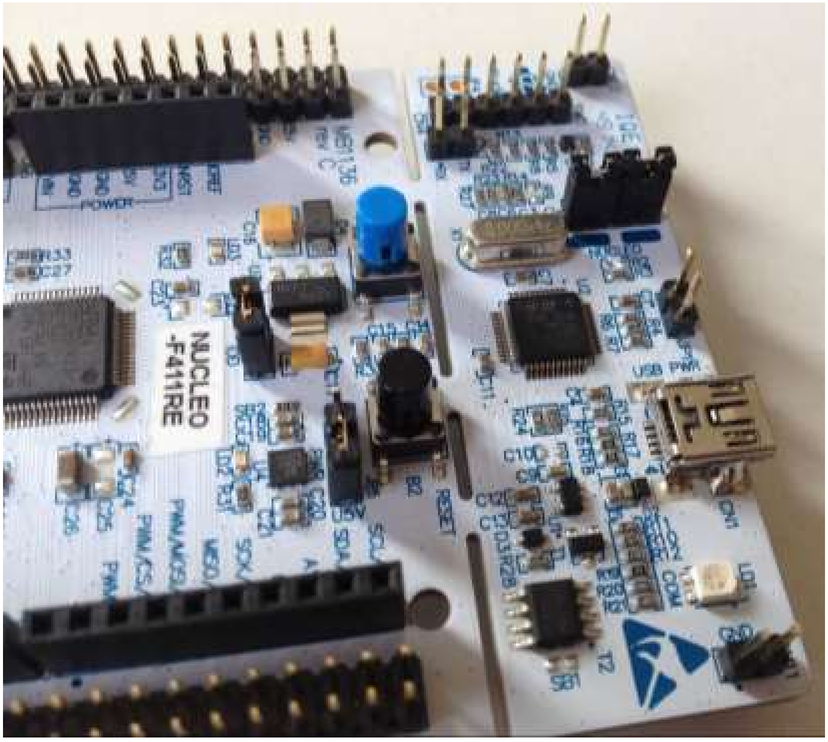


Figure 2 – ST Nucleo board Jumper configuration

2.2 NUCLEO CARRIER BOARD SETUP

The **Nucleo Carrier Board rev B or C** offers a more convenient setup to connect ST Nucleo and ICM-20602 sensor daughter boards.

Nucleo Carrier board jumper configurations:

JP1	(VDD=VDDIO)
JP2	(VDD=3V3)
JP8	(Nucleo)

Warning: The ST NUCLEO headers have to match the Nucleo Carrier board ones. Please verify to connect the boards are correctly connected (pin1 aligned).

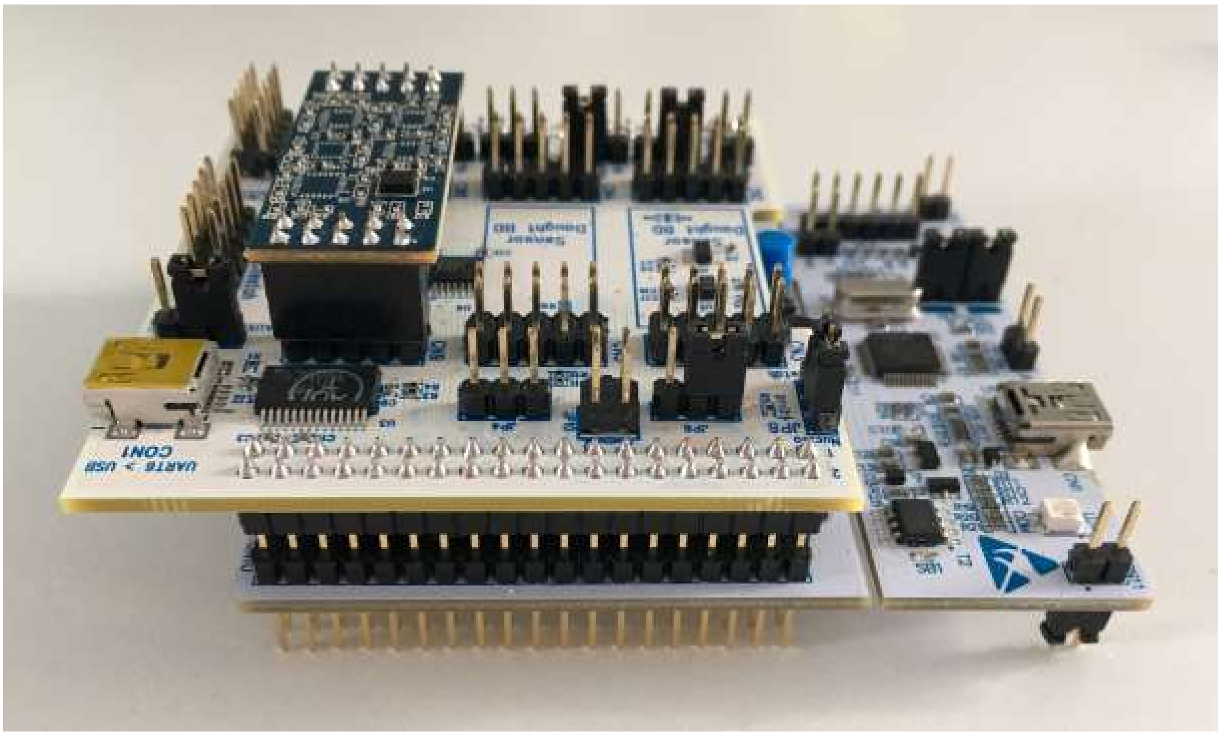


Figure 3- Nucleo Carrier Board rev B Setup

2.2.1 Pluging-in Sensor Daughter Boards

Using **SPI** or **I2C**, ICM-20602 daughter board must be connected on slot labelled 'INV Sensor BD'.

Optionally, you can connect an AK09911/AK09912/AK09915 daughter board into the slot labelled "Sensor Daught BD"

For **ICM-20602**, the ST Nucleo gets raw compass data from the AK0991x device through ICM-20602 **I2C primary interface**. The AK0991x board jumper configurations are the following:

J400	[1 :2] Open
	[3 :4] Open
	[5 : 6] Closed
	[7 : 8] Closed

The Nucleo Carrier board jumper configuration also needs to be updated to support interrupt handling from Ak0991x sensor daughter board.

JP4	(INTa2=PB4)
JP6	(INTb2=PB4)

To handle both SPI communication to ICM-20602 and I2C communication to AK0991x device, the JP10 configuration is the following.

JP10	Open (1!=3)
	Open (2!=4)

Otherwise for I2C only communication, the JP10 configuration is the following:

JP10	Closed (1==3)
	Closed (2==4)

2.2.2 Powering the NUCLEO and Carrier board

To power the platform, connect the NUCLEO to a PC using a mini-USB cable (on ST-NUCLEO CN1).

2.2.3 Get traces on the NUCLEO and Carrier board

According to the firmware options defined, to get FW traces:

- The on-board FTDI UART/USB converter can be used by connecting a USB cable to CON1 on the Carrier board. You will find FTDI drivers to install on your PC here: http://www.ftdichip.com/Drivers/VCP.htm

- The ST-LINK UART/USB connector can be used by connecting a USB cable to CN1 on ST Nucleo board.

3 SOFTWARE ENVIRONMENT

3.1 PREREQUISITE

To build and use samples application provided as part for the eMD Developer Kit packages, the following 3rd party software are required:

- IAR Workbench IDE: https://www.iar.com/iar-embedded-workbench/ (see Tools guide for installation steps and usage)
    - To build and debug provided FW application
- A RS232 terminal emulator (such as Putty: http://www.putty.org/)
    - To retrieve traces from provided FW application
- ST Link utilities: http://www.st.com/web/en/catalog/tools/PF258168
    - To load FW binaries and access to the USB VCOM port of the NUCLEO

3.2 EMD DEVELOPER KIT INVENSENSE PACKAGES

Several packages are available:

- *invn.firmware.emd-mcu.nucleo.icm20602_lite-iar-cm4-fpu-x.y.z.tar.gz*
  invn.firmware.emd-mcu.nucleo.icm20602_full-iar-cm4-fpu-x.y.z.tar.gz

Tools running on the PC for data display are available in the delivered package.

3.3 FW PACKAGE DESCRIPTION

The InvenSense ICM-20602 eMD packages include all the necessary files to create a custom application using ICM devices.

Two separates packages per devices are generated according to the examples provided;

- The FULL-example, targeting applications with the full feature set (FSR and mounting matrix settings, all the raw, calibrated, orientation, and gesture sensors supported).
- The LITE-example, targeting low performances microcontroller with a very simple use of the sensors

These packages are organized as follow

- ***release/bin*** **:** contains FW binaries of the samples application available in the package
- ***sources***:
    - ***examples***: contains sample codes applications for NUCLEO MCU that demonstrates how to control and retrieve data from ICM-20602 devices using InvenSense Device Driver library (libIDD).
    - ***Invn***: InvenSense libraries sources
    - ***board-hal***: contains the low level drivers for NUCLEO used in samples application
    - ***stm32f4x***: contains the ST libraries for STM32F4xx
- ***doc*****:** contains libIDD doxygen documentation
- ***prebuilt***: contains prebuilt InvenSense libraries (with IAR compiler for Cortex-M4 with FPU) for algorithms
- ***tools***: contains useful InvenSense tools to be used with the eMD Developer Kit, such as '*sensor-cli'* a command line application to control and display data from InvenSense device on a windows computer. Refer to Appendix B for more about *sensor-cli*

Note : Currently, folder hierarchy is very deep and IAR IDE may complain about filename being too long. To avoid this, copy/paste the relevant folders in an upper directory.

4 BUILDING AND RUNNING SAMPLES APPLICATIONS

4.1 OVERVIEW

The following applications are available in the package (in *…/sources/examples/*):

- ***example-idd-full-icm20602*** – this application demonstrates how to use *libIDD (InvenSense Device Driver library)* to control and retrieve data from ICM devices. It encodes sensor events in binary to send them over UART/USB to be displayed by *sensor-cli*. This same application can also be configured as standalone to print sensor events received from the device as traces on a secondary UART.

- ***example-idd-lite-icm20602* –** this application demonstrates how to use libIDD low level drivers to control and retrieve data from ICM devices. The application sends the sensor data to *sensor-cli* the same way it's done in the full example.

4.2 BUILDING EXAMPLE APPLICATIONS

A ready to use IAR project is available for each provided application. Look for the file with *.eww* extension.

Refer to Tools User Guide for eMD to load an IAR project, building the FW and loading it to the NUCLEO board.

Additional information is available in *README.md* to build application and use those applications in another environment.

A common configuration for all applications is available:

4.2.1 Running the IDD-ICM20602 FULL example application

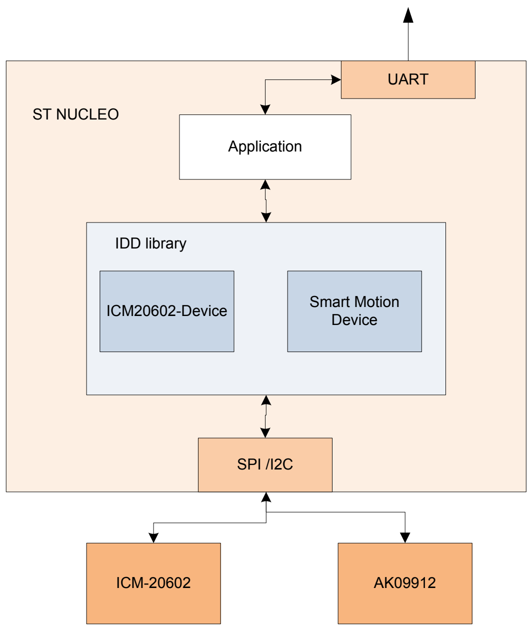
This configurable FW application shows how to use libIDD to setup, control and retrieve data from an ICM-20602. With this setup, the ICM driver runs on the STM32 MCU.

Refer to the source file *example.c* for all available configurations (through *#define*) and detailed explanation.

Limitation:

- The minimum frequency allowed for sensor data reporting is 4Hz. The 8-bit HW Gyroscope rate divider register causes this limitation.
- Enable all sensors at start results in FIFO overflow

Architecture diagrams:



4.2.1.1 Default application behavior

At init, the application will:

- Initialize NUCLEO peripherals (irq, timer, SPI)
- Configure the UART to get traces from the ST NUCLEO com port and connect the IDD Wrapper application to Nucleo Carrier board UART/FTDI connector.
- Initialize the libIDD for the selected ICM device
- Setup and initialize the ICM device
- Run the IDD Wrapper application (cf 4.2.1.8)

4.2.1.2 Receiving sensor events

For the standalone application, you have to set ***#define USE_IDDWRAPPER*** to ***0***. Select the sensors to start by configuring defines, by default the raw accelerometer and raw gyroscope are enabled (set to 1). After the default behavior, this application will:

- Check if the selected sensors are available (ping)
- Start selected sensors
- Display sensor event traces over *UART_LOG*.

To receive the sensor events, you need to use a terminal emulator application (such as Putty or Teraterm) with the following configuration:

- 921600 bauds
- 8bits data
- 1 Stop bit
- No parity
- No HW Flow Control

4.2.1.3 Choosing between SPI and I2C

By default, SPI is used to communicate between NUCLEO and ICM device. This can be changed by setting #***define SERIF_TYPE_SPI*** define to **0** and ***#define SERIF_TYPE_I2C*** to **1**.

Refer to bring-up section for proper wiring depending on your setup.

4.2.1.4 Configuring the device

Full Scale Range (FSR), Mounting Matrix, can be changed by updating the value of the corresponding variables in *example.c*.

Default FSR value are **+/- 4g** for accelerometer and **+/-2000 dps** for gyroscope.

Default mounting matrix is set to identity with corresponds to the following reference frame :

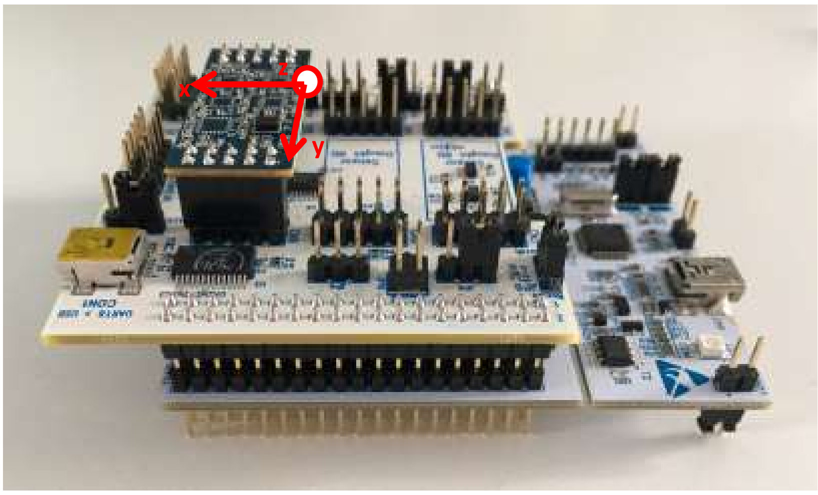


Figure 4 – board reference frame

Supported FSR values are :

- Gyroscope : 250dps, 500dps, 1000dps and 2000dps
- Accelerometer : 2g, 4g, 8g and 16g

4.2.1.5 Magnetometer sensor support

The default compass supported is the AK9912.

If you want to use the AK9911 instead, set the ***#define COMPASS_IS_AK9911*** *value to* ***1*** *and* ***#define COMPASS_IS_AK9912*** value to **0**.

Alternatively, if you want to use the AK9915, set ***#define COMPASS_IS_AK9915*** value to ***1*** and ***#define COMPASS_IS_AK9912*** value to ***0***.

4.2.1.6 Primary magnetometer sensor

The ICM-20602 supports Primary magnetometer (magnetometer directly connected to the I2C of the ST Nucleo).

To enable primary sensor in the example application you must set ***#define USE_PRIMARY_COMPASS*** value to **1**. If an AK09912 daughterboard is connected, starting RAW MAGNETOMETER sensor will report and display compass data.

4.2.1.7 Using SmartMotion libraries

ICM-20602 only reports RAW data. However, it is possible to get enhanced motion features using the virtual SmartMotion device that embeds InvenSense algorithm libraries, running on the MCU, to compute calibrated, orientation and gestures based on RAW data.

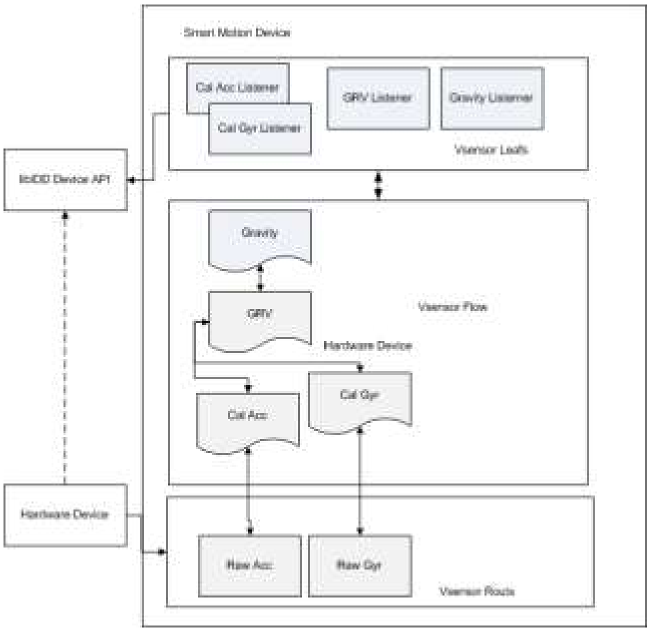
SmartMotion libraries are integrated in such fashion that it is transparent to the user as the same libIDD interface is exposed.

To use SmartMotion in the example application, you have to set ***#define USE_SMART_MOTION*** to ***1***.

SmartMotion is enabled by default in FULL example, you can then benefit from the following virtual sensors:

- Calibrated accelerometer
- Calibrated magnetometer (when primary compass is connected)
- Calibrated gyroscope
- Game Rotation Vector
- Rotation Vector (when primary compass is connected)
- Geomag Rotation Vector (when primary compass is connected)
- Step Counter
- Step Detector
- SMD
- BAC
- Tilt gesture
- PickUp gesture

Architecture diagram for basic features:



4.2.1.8 Running the IDD Wrapper application

This allows to control sensors using sensor-cli on Windows PC.

To do so, you have to set ***#define USE_IDDWRAPPER*** to ***1*** (enabled by default).

In this configuration, the ICM drivers run on the STM32 MCU, and sensor data send over USB through IDD wrapper protocol. The main UART (ST-MCU CN1) is reserved for this protocol communication. Additional FW traces are available on secondary UART.

To display data, run ***sensor-cli*** on the Windows PC from a console with the following argument, please refer to Tools Guide for eMD for further information:

```
sensor-cli --target=emdwrapper,port=\\.\COMXX --adapter=dummy
```

Limitations:

- Available sensors are :
  - Raw Accelerometer / Gyroscope / Magnetometer / Temperature
  - Calibrated Accelerometer / Gyroscope / Magnetometer
  - Linear Acceleration
  - Gravity
  - Uncalibrated Gyroscope / Magnetometer
  - Game Rotation Vector
  - Rotation Vector
  - Geomag Rotation Vector
  - Orientation (Euler angles)
  - B2S / Shake / Double Tap / SMD / Step detector / Tilt detector / Wake gesture / Glance gesture
  - WOM / BAC
  - Step counter

- Available commands are :
  - Ping a sensor to check if it is supported by the device
  - Start a sensor
  - Stop a sensor
  - Change sensor output data rate
  - Self-test on hardware sensors
  - Get / Set bias for calibrated accelerometer and gyroscope
  - Set a mounting matrix for the based sensors: raw accelerometer, accelerometer, raw gyroscope, uncalibrated gyroscope, gyroscope, raw magnetometer, uncalibrated magnetometer, and magnetometer. Orientation sensors not supported.

- Configuring sensor FSR for the base ICM sensors: raw accelerometer (mg), accelerometer (mg), raw gyroscope (dps), uncalibrated gyroscope (dps), gyroscope (dps)

- The frequency maximum supported is 2kHz
- The minimum frequency for orientation sensor is 50Hz
- The accelerometer and gyroscope sensor data rate output are tied together. So when both sensors are enabled with two different reporting period, the fastest will be applied.

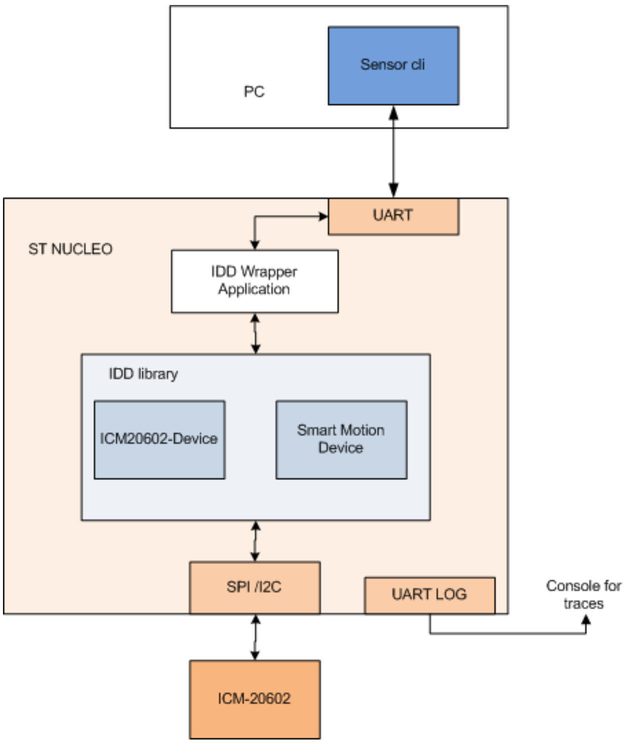
4.2.1.9 Storing self-test and algorithm bias in NV memory

An enhanced feature is **available using the IDD Wrapper application**; it is the self-test and algorithm bias storage in NV memory.

- At each **sensor-cli start-up** the self-tests and algorithms bias are applied if available.
- During the run, you may compute self-tests bias on hardware sensors and algorithm bias for accelerometer, gyroscope and magnetometer.
- These biases will be stored in NV memory when properly exiting sensor-cli meaning when receiving a proper **clean-up command**. This behavior will improve the sensor calibration speed.

This feature can be implementing independently of the IDD Wrapper application.

Diagram architecture:



4.2.2 Running IDD-ICM20602 LITE example application

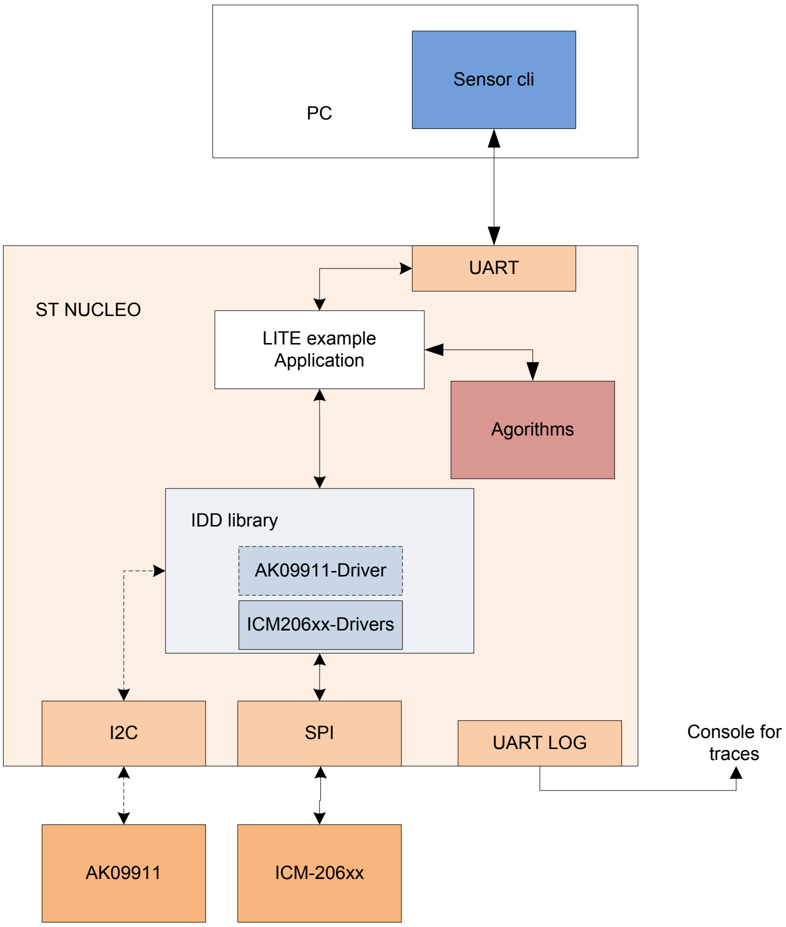
This application shows how to use the chip with lightest possible driver. It targets compatibility with low performances microcontroller (Cortex-M0, M3, …). The application instantiates directly the ICM driver and communicates through the low-level APIs. The algorithms are called in the application at the frequency requested. Data are reported through UART using the same IDD Wrapper than describes above.

The LITE example supports a large set of features without using a sensor framework, therefore there are some command simplifications to avoid a code being too complex:

- At initialization, all algorithms are initialized and executed continuously at the default rate (50Hz)
- When a *set_sensor_period* command is received, **it is changed for all sensors**
- When a *enable_sensor* command is received, the data report is enabled (through UART)

- When a *disable_sensor* command is received, the data report is disabled and algorithms continue to run

Architecture diagram:



Limitations:

Limited commands set:

- Enable / Disable sensor
- Set sensor period
- Self-tests
- Set a mounting matrix for base sensors

Sensor features supported:

- Raw accelerometer
- Raw Gyroscope
- Calibrated accelerometer
- Calibrated gyroscope
- Uncalibrated gyroscope
- Game rotation vector
- Predictive quaternion
- Gravity
- Linear Acceleration

Optional sensor features supported:

- Raw magnetometer (AK09911 only)
- Calibrated magnetometer
- Uncalibrated magnetometer
- Rotation vector

- Geomagnetic rotation vector

All sensors run at the same rate in the application.

The GRV requires having data at least at 50 Hz for optimal performances. Therefore, the **minimum frequency available for all sensors in the application will be 50 Hz.**

All sensors are turned on at start but none is reporting, still the data are received and algorithms processed.

Note: Per design at start-up, all sensors and algorithms are started. So the GRV orientation may drift until the gyroscope is calibrated. Once calibrated, the position is kept as the initial reference. The user may use this orientation as reference and only use the relative changes. To do so, you can refer to the following formula/code:

$$quat_{out} = quat_{grv} \, . conjugate(quat_{ref})$$

With:

- $quat_{out}$ the result quaternion
- $quat_{grv}$ the quaternion obtained with the GRV sensor
- $quat_{ref}$, the quaternion that represents the position of reference

```
static void applyReferenceQuat(const float qin[4], const float q0[4], float qout[4])
{
          float q0c[4];
          // Conjugate
          q0c[0] = q0[0];
          q0c[1] = -q0[1];
          q0c[2] = -q0[2];
          q0c[3] = -q0[3];
          // Apply Compensation
          qout[0] = q0c[0]*qin[0] - q0c[1]*qin[1] - q0c[2]*qin[2] - q0c[3]*qin[3];
          qout[1] = q0c[0]*qin[1] + q0c[1]*qin[0] + q0c[2]*qin[3] - q0c[3]*qin[2];
          qout[2] = q0c[0]*qin[2] + q0c[2]*qin[0] + q0c[3]*qin[1] - q0c[1]*qin[3];
          qout[3] = q0c[0]*qin[3] + q0c[3]*qin[0] + q0c[1]*qin[2] - q0c[2]*qin[1];
          // Normalize
          float tmp = sqrtf(qout[0]*qout[0] + qout[1]*qout[1] + qout[2]*qout[2] + qout[3]*qout[3]);
          if (tmp > 0 ) {
                    qout[0] /= tmp;
                    qout[1] /= tmp;
                    qout[2] /= tmp;
                    qout[3] /= tmp;
          }
}
```

5 DOCUMENT INFORMATION

5.1 REVISION HISTORY

REVISION	DATE	DESCRIPTION	AUTHOR
1.0	March 24, 2016	Initial version.	Paul Besson Axel Zbitak
1.1	March 29, 2016	Add current limitations to relevant paragraphs. Add IddWrapper sections Add Nucleo carrier board setup	Paul Besson Aurélie Fontaine
1.2	March 30, 2016	Review limitations.	Aurélie Fontaine
1.3	April 5, 2016	Add Usage Examples Add available fsr Update paragraph numbers	Axel Zbitak
1.4	April 8, 2016	Update HW connection and Carrier board setup OIS feature support description SensorStudio Player appendix	Aurélie Fontaine
1.5	April 8, 2016	Updated SensorStudio Player appendix	Thomas Muguet
1.6	April 28, 2016	Updated SensorStudio Player appendix for r4	Thomas Muguet
1.7	April 29, 2016	Rotate figure 8 to match board reference frame as described Add I2C Primary compass support ; hardware platform and software specifications Add architecture diagram	Paul Besson Aurelie Fontaine
1.8	May 5,2016	Align documentation according the firmware default application changes Correct limitations Add links to needed drivers	Aurelie Fontaine
1.9	July 1st,2016	Add Hardware environment description for Nucleo Carrier board rev B Update some new option descriptions for the main IDD ICM-20690/ICM-20602 example Add dedicated EIS and OIS application description	Aurelie Fontaine
1.10	July, 22,2016	Update HW configurations tables and some options definitions Add MIPS measurements	Aurelie Fontaine
1.10	August 26, 2016	Add achievable frequencies	Loic Michallon
1.11	September 1, 2016	Add LITE example application. Add ICM-20603 support. Add AK09915 support. Update FW package description including the tools package. Remove USB –SPI bridge application from the package.	Aurelie Fontaine
1.12	September 20, 2016	Add time measured for MIPS measurements. Add required accel freq in frequency table. Remove warning about Lite example for 50 Hz.	Loic Michallon
1.13	September 28, 2016	Complete frequency table	Aurélie Fontaine
1.14	October 24, 2016	Update LITE example description	Aurélie Fontaine
1.15	November 4, 2016	Add note to set the reference position to the GRV cube and avoid drift at start-up due to design limitation Add code size table Rework MIPS measurement paragraph Update FW packages description	Aurelie Fontaine
2.0	November 25, 2016	Split current guide in several guides : - Tools guide - Software metrics guide - Application Note for internal development - This guide with less information	Julie Gonin
2.1	December 01, 2016	Replace MSB picture by nucleo figure for reference frame Remove SMARTMOTION and IDD_WRAPPER define and mix FULL and IddWrapper applications description Keep only one configuration possible for JP10 on carrier board	Julie Gonin
2.2	December 12, 2016	Bring back the SMART MOTION and IDD WRAPPER define option definition. Remove RUN SELFTEST AT STARTUP define option. Add Set mounting matrix command support and predictive quaternion as supported sensor in FULL/LITE examples. Add Linear Acc, Gravity and Geomag RV sensors in LITE example. Add a paragraph to explain the self-test and algorithm bias storage mechanism	Aurélie Fontaine

Table 1. Revision History