InvenSense

Software User Guide

For

ICM-20x48 eMD

1 CONTENTS

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/

1 OVERVIEW

The purpose of this document is to give an overview of the ICM-20x48 (20648 and 20948) eMD Developer Kit that will allow users to create a custom application based on motion sensors. This document may also serve as a quick start guide for the ICM-20x48 package and its elements, including how to setup and use the provided sample applications.

1.1 INTRODUCTION

The ICM-20x48 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-20x48 eMD solution is an embedded sensors combo (6 axis for ICM20648 respectively 9 axis for ICM20948) on chip, easy to integrate for users developing in wearable and IoT space. The Developer's Kit includes a full sensor software solution.

1.2 ICM-20X48 EMD BASICS

1.2.1 Overview

The ICM-20x48 chip's MEMS are accessible through SPI or I2C.

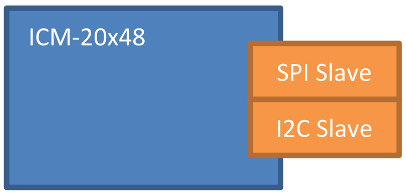


Figure 1 – ICM-20x48 eMD Architecture

1.2.2 ICM-20x48 eMD Hardware Layout

The Nucleo-F411RE board communicates with the ICM-20x48 through SPI, providing a fast communication pathway. Besides the SPI connection, two interrupt lines are connected to the host. These lines are used to signal sensor events, and allow the application to sleep in order to save power.

2 HARDWARE PLATFORM

The eMD platform for ICM-20x48 consists in the following components:
- ST NUCLEO F411-RE board including a STM32 F411 MCU
- InvenSense Nucleo Carrier Board rev C
- InvenSense sensor Daughter Boards (aka DB) for ICM-20648 6-axis sensors or ICM-20948 9-axis sensors

Since ICM-20648 works at 3.3V while ICM-20948 works at 1.8V, the hardware setup for the two chips diverges. For this reason, we will present separately the hardware setups of the two ICs.

2.1 PREREQUISITE

In order to run the sample applications provided as part of the eMD Developer Kit packages, the following 3rd party hardware is required:

- ST NUCLEO-F411RE board
- 2 Mini USB male to USB 2.0 male cables

2.2 ST NUCLEO-F411RE SETUP

The **ST Nucleo-F411RE** 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

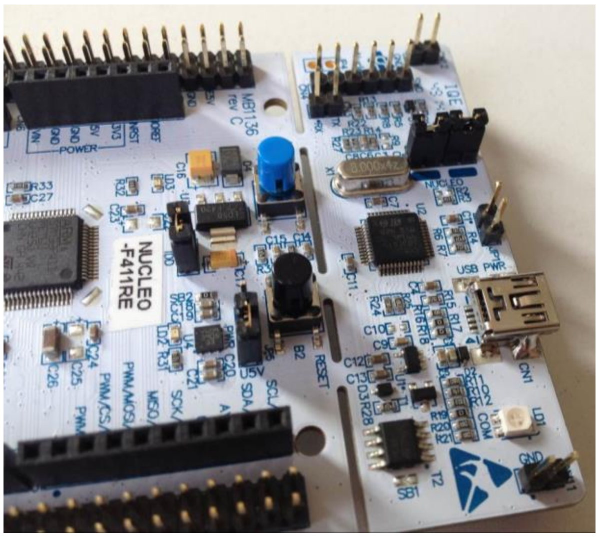
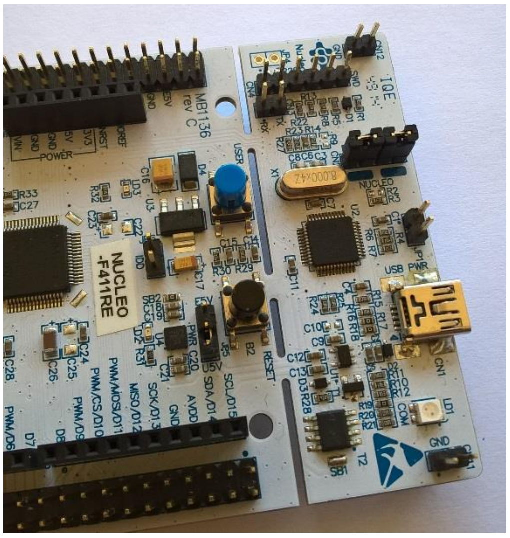
Nucleo's jumpers configuration for **ICM-20648**	Nucleo's jumpers configuration for **ICM-20948**
JP1 Open JP5 U5V JP6 Closed CN2 Closed	JP1 Open JP5 U5V JP6 Open CN2 Closed
	

Table 1 Nucleo's jumpers configuration

2.3 INVENSENSE CARRIER BOARD SETUP

The Invensense Carrier board aims at facilitating the connection of Invensense ICM-20x48 sensor daughter board to the Nucleo-F411RE by plugging it into ST Morpho connectors (CN7 and CN10 on Nucleo board).

Invensense ICM20x48 sensor daughter board must be connected on the CN8 and CN9 of carrier board.

Since on the daughter boards there is not much hardware setup to be done, we will treat the carrier board and daughter as a whole throughout this subchapter.

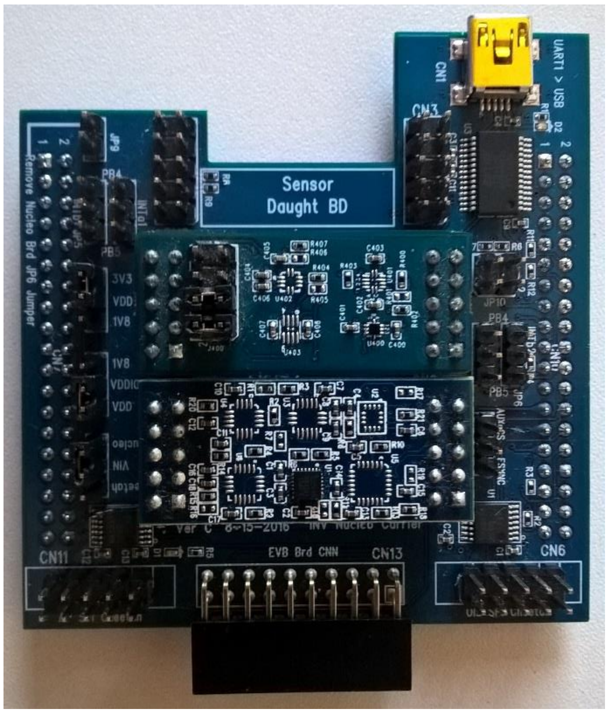
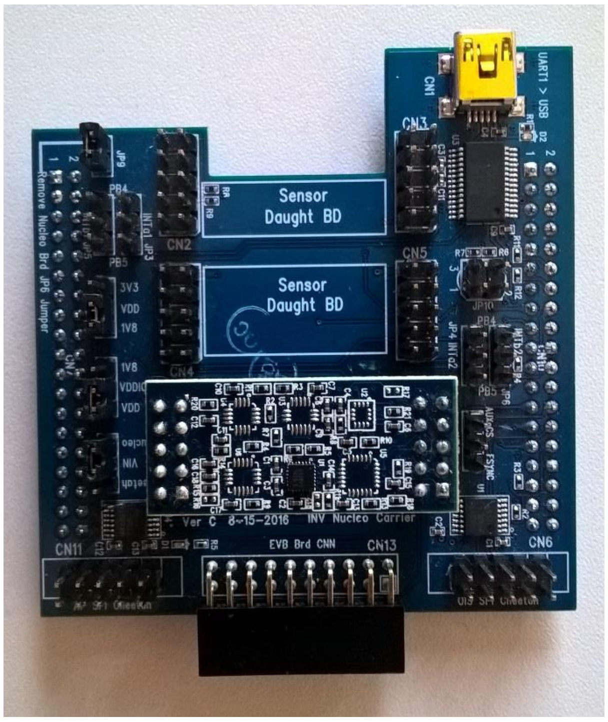
Carrier board jumpers configuration for **ICM-20648**	Carrier board jumpers configuration for **ICM-20948**
JP1 VDD-VDDIO JP2 VDD-3V3 JP8 VIN-NUCLEO JP9 Open	JP1 VDD-VDDIO JP2 VDD-1V8 JP8 VIN-NUCLEO JP9 Closed
 Sensor Daught BD UART1 > USB CN1 CN3 EVB Brd CNN CN11 CN13 CN6	 Sensor Daught BD Sensor Daught BD UART1 > USB CN1 CN3 CN5 EVB Brd CNN CN11 CN13 CN6

Table 2 Carrier board jumpers configuration

2.4 COMPLETE HARDWARE SETUP

In order to complete the hardware setup, please follow the steps below:

1) Plug the Invensense carrier board into the ST Morpho connector of the Nucleo
2) Plug the Invensense ICM20x48 sensor daughter board into the INV Sensor Daughter Board slot of the Invensense carrier board
3) For ICM-20648, plug the AKM sensor daughter board onto CN4 and CN5 of the Invensense carrier board
4) On the Nucleo board, connect CN1 to the PC’s USB using the mini USB to USB 2.0 cable
5) On the Invensense Carrier board, connect CN1 to the PC’s USB using the mini USB to USB 2.0 cable
6) The result should be a setup identical to the one in the table below

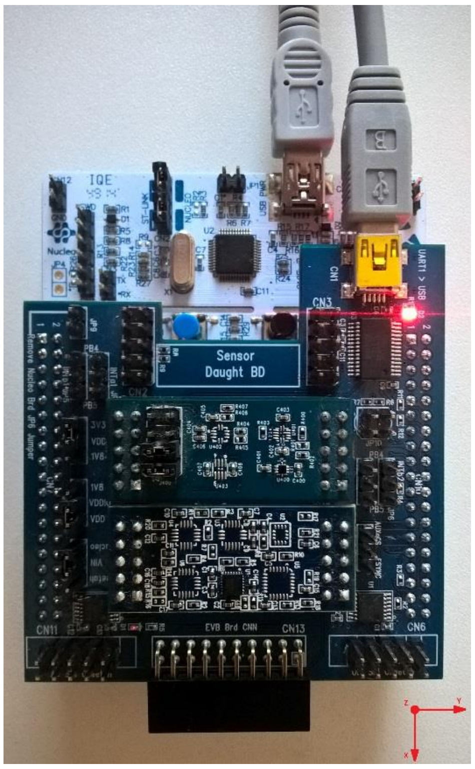
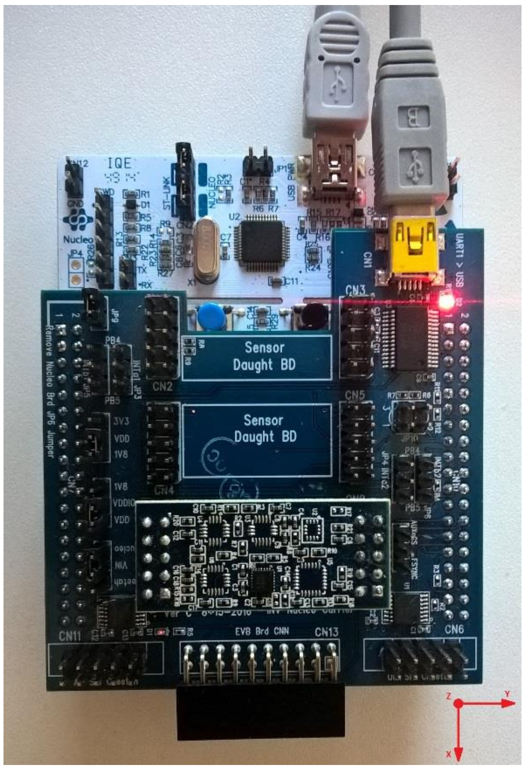
Complete hardware setup for **ICM-20648**	Complete hardware setup for **ICM-20948**
	

Table 3 Complete hardware setup

3 SOFTWARE ENVIRONMENT

3.1 PREREQUISITE

In order to build and use the sample applications provided as part for the eMD Developer Kit packages, the following 3rd party software is required:

1. IAR Workbench IDE: https://www.iar.com/iar-embedded-workbench/ (see appendix A for installation steps and usage)
    a. To build, flash and debug provided FW application
2. An RS232 terminal emulator (such as Putty: http://www.putty.org/)
    a. To retrieve traces from the FW application
3. Optional, ST-LINK utility: http://www.st.com/web/en/catalog/tools/PF258168
    a. To load FW binaries and access to the USB VCOM port of the NUCLEO

3.2 EMD DEVELOPER KIT INVENSENSE PACKAGES

One package is for ICM-20x48:

- *invn.firmware.emd-mcu.nucleo.icm*20x48 *-iar-cm4 -fpu-x.y.z.tar.gz* where x is to be replaced with 6 or 9 according to the used package.

3.3 FW PACKAGE DESCRIPTION

The InvenSense ICM-20x48 eMD package includes all the necessary files to create a custom application using ICM-20x48 device.

This package is structured as follows:

- ***release/bin*** **:** contains FW binaries of the sample applications available in the package
- ***sources***:
    - ***examples***: contains sample code for NUCLEO-F411RE that demonstrates how to control and retrieve data from ICM-20x48 device using InvenSense Device Driver library (libIDD).
    - ***Invn***: InvenSense libraries sources
    - ***board-hal***: contains the low level drivers for NUCLEO used in sample applications
    - ***stm32f4x***: contains the ST libraries for STM32F4xx
- ***doc*****:** contains libIDD doxygen documentation in html and this user guide
- ***tools:*** contains a command line application called *sensor-cli* that controls and displays data from the Invensense device on a windows computer

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 SAMPLE APPLICATIONS

4.1 OVERVIEW

The following application is available in the package (in *…/sources/examples/*):

- ***example-icm20x48*** (where x is to be replaced with 6 or 9 according to the used package) – this application allows *sensor-cli* to control the ICM-20x48 device by means of a dedicated protocol. For details, please refer to chapter *Running the example application*.

4.2 BUILDING THE EXAMPLE APPLICATION

A ready-to-use IAR project is available for each provided application. In order to launch the IAR IDE, look for the file with *.eww* extension in the *sources/examples/example-icm20x48* (where x is to be replaced with 6 or 9 according to the used package) folder from the firmware package.

For opening an IAR project, building the FW and loading it to the NUCLEO board, please refer to appendix A.

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

4.3 RUNNING THE EXAMPLE APPLICATION

In this configuration, the ICM drivers run on the STM32 MCU, and sensor data is sent over USB through Dynamic Protocol. The main UART (carrier board CN1) is reserved for this protocol communication. Additional FW traces are still available on the secondary UART (Nucleo CN1).

To retrieve data from the ICM device, please configure your hardware according to the *Hardware Platform* chapter and then proceed to the steps below:

1) Launch *ST-LINK Utility*
2) Flash the Nucleo with the ***example-icm20x48*** (where x is to be replaced with 6 or 9 according to the used package) FW which you can find in the firmware package as described in chapter *FW Package description*
3) Open a windows console and navigate to the **tools** folder which you can find in the firmware package
4) Launch ***sensor-cli*** using the command below, and refer to *Appendix C – sensor-cli quickstart* for further details on how to use it:

```
sensor-cli --target=emdwrapicm20x48,port=\\.\COMxx --adapter=dummy
```

Notes:

- *COMxx* is the COM port of the mini USB to USB 2.0 cable connected to carrier board, enumerated as a VCOM port

Limitations:

- ICM-20x48 DMP3 image is hard-coded into the STM32's firmware image.

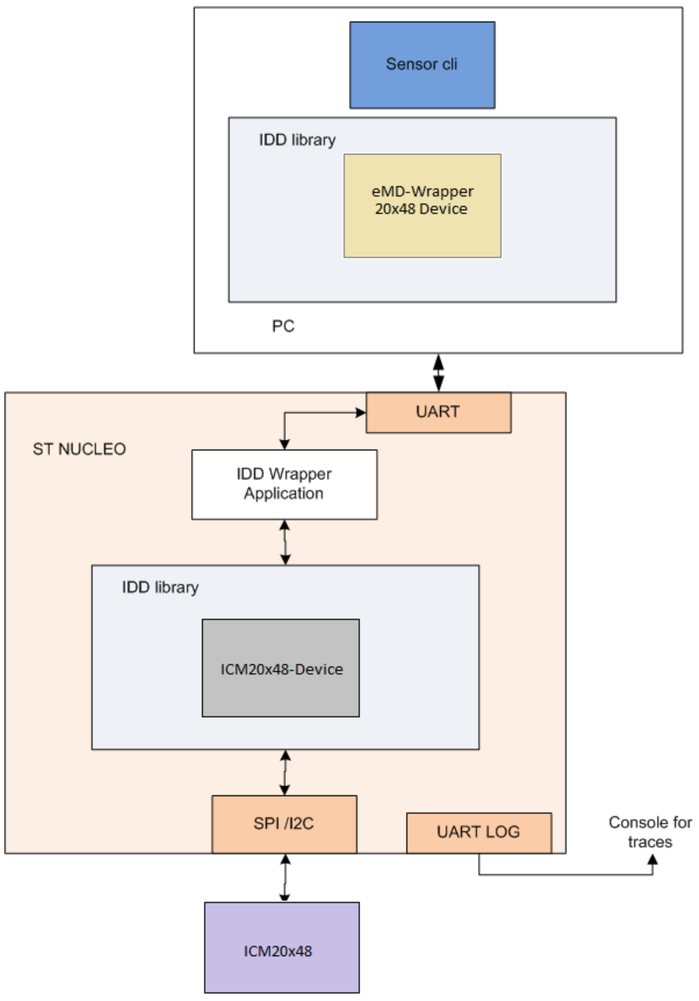
- Available sensors are :
  - Raw Accelerometer / Gyroscope
  - Calibrated Accelerometer
  - Calibrated Gyroscope
  - Calibrated Magnetometer
  - Uncalibrated Gyroscope
  - Uncalibrated Magnetometer
  - Game Rotation Vector
  - Rotation Vector
  - Geomagnetic Rotation Vector
  - BAC
  - Step Detector
  - Step Counter

  - SMD
  - PickUp
  - Tilt
  - Gravity
  - Linear acceleration
  - Orientation
  - B2S

- Available commands are :
  b. Ping a sensor
  c. Start a sensor
  d. Stop a sensor
  e. Change sensor output data rate
  f. Set sensor configuration (FSR)
  g. Set mounting matrix
  h. Set offset
  i. Set timeout

- STM32 SPI master speed connected to ICM-20x48 is configured to be 1.5MHz.

Architecture diagram:



4.4 CODE SIZE

Code size measurement in release for idd-wrapper is the following one. ICM-20x48 driver with features described previously requires less than 25kB.

in FLASH :

Blocks	(kB)
TOTAL	**73.37**
Application	22.90
Low level drivers	10.39
Lib IDD	40.01
Device 20x48	2.30
Driver 20x48	25.58

in RAM :

Blocks	(kB)
TOTAL	**54.49**
Application	6.38
Low level driver	44.41
Lib IDD	1.70

4.5 SENSOR CONFIGURATION

4.5.1 Frequencies

The table below sums up the achievable frequency for each sensor.

Sensor	Reporting Frequencies (Hz)		Reporting mode	Required Accel frequency (Hz)	Required Gyro frequency (Hz)
	Min	Max			
Accelerometer	1	225	*Continuous*	=	x
Raw Accelerometer	1	225	*Continuous*	=	x
Gyroscope	1	225	*Continuous*	x	=
Raw Gyroscope	1	225	*Continuous*	x	=
Uncalibrated Gyroscope	1	225	*Continuous*	x	=
Magnetometer	*1*	70	*Continuous*	x	x
Uncalibrated Magnetometer	*1*	70	*Continuous*	x	x
Game Rotation Vector	*50*	225	*Continuous*	x	x
Rotation Vector	50	225	*Continuous*	x	x
Geomag Rotation Vector	1	*225*	*Continuous*	x	x
Gravity	50	*225*	*Continuous*	x	x
Linear Acceleration	50	*225*	*Continuous*	=	x
Orientation	50	*225*	*Continuous*	x	x
SMD			*One shot*		
Step Counter			*On change*		
Step Detector			*On change*		
Tilt			*On change*		
Pickup			*On change*		
BAC			*On change*		
B2S			*On change*		

‘=’ means that the MEMS frequency will be the same as the corresponding sensor.
‘x’ means that it doesn’t use the corresponding MEMS.

Accelerometer is raw accelerometer value scaled to output value in g. Associated accuracy will always be 0 since not mapped to calibrated accelerometer data.

Linear Acceleration relies on GRV and Accelerometer. When enabled the output frequency of all three sensors will be the fastest of the three.

Gravity relies on GRV. When enabled, the output of both sensors will be the fastest of the two.

Orientation relies on Rotation Vector. When enabled, the output of both sensors will be the fastest of the two.

Raw Gyroscope, Gyroscope and Uncalibrated Gyroscope, if enabled together, will output data at the same frequency, being the quickest one amongst f_{rgyr} , f_{gyr} and f_{ugyr}

Game Rotation Vector will output at its own frequency, no matter what other sensor frequencies are.

4.5.2 FSR

Accelerometer and Gyroscope MEMS FSR can be reconfigured with the following set of values :

- When applied to raw accelerometer or accelerometer sensor :
  - 2g
  - 4g
  - 8g
  - 16g
- When applied to raw gyroscope or gyroscope or uncalibrated gyroscope sensor :
  - 250dps
  - 500dps
  - 1000dps
  - 2000dps

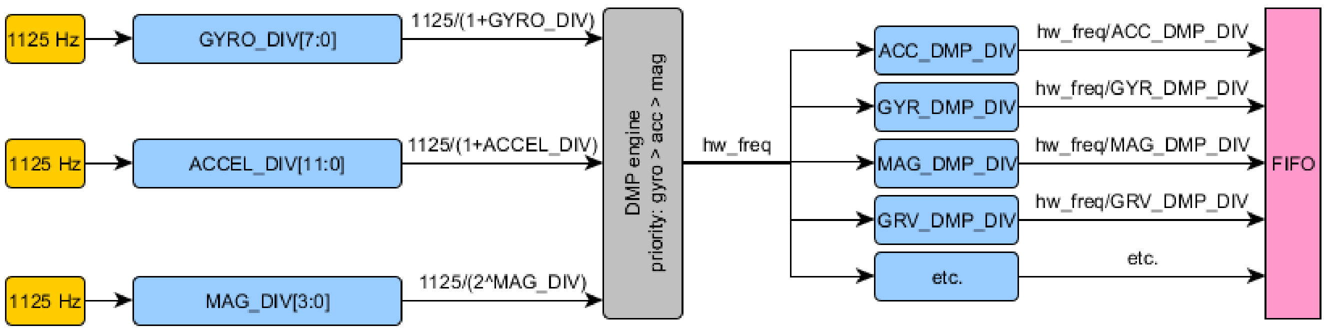
5 APPENDIX A - SYSTEM KNOWN ISSUES

- When any accelerometer-only sensor is already enabled (Accelerometer or Raw accelerometer) if a gyroscope-based sensor is enabled (Calibrated Gyroscope, Uncalibrated Gyroscope or Game Rotation Vector), system can stop reporting data for up to 50ms.

- 20x48 SPI slave interface speed should not be set higher than 2.5MHz to ensure sensor data consistency

- When Accelerometer and Gyroscope are enabled both, with different sample rate. When the accelerometer is stopped, 1 or 2 gyroscope sample are triggered with wrong rate.

- The 20x48 contains two clock division stages, as we can see in the figure below. Because the DMP engine outputs the *hw_freq* as either 1125/(1+GYRO_DIV) or 1125/(1+ACCEL_DIV) or 1125/(2^MAG_DIV) based on priorities, there are certain limitations in terms of output data rates.
  For example, if one requests 100 Hz ODR from the accelerometer and 50 Hz ODR from the gyroscope, the actual output data rate of the accelerometer will be 112 Hz while the actual output data rate of the gyroscope will be 56 Hz. This happens because *hw_freq* = 1125/(1+GYRO_DIV)=1125/(1+9)=112.5Hz (the gyroscope has a higher priority than the accelerometer), ACC_DMP_DIV will be set to 1 and GYR_DMP_DIV will be set to 2. As such, the actual output data rate will always be greater than or equal to the requested output data rate.



6 APPENDIX B - IAR IDE QUICK START

This section provides high-level information to install and use IAR IDE with NUCLEO.

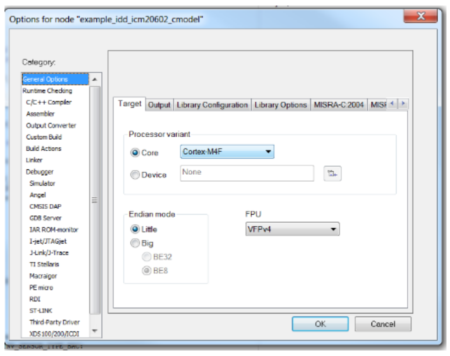
Please refer to ST or IAR documentation for additional information.

6.1 INSTALLING IAR WORKBENCH IDE

- Download IAR workbench IDE from https://www.iar.com/iar-embedded-workbench/. You will have to get an available license to use it. The requested IAR toolchain is 7.0.

- Once the software is installed, you should be able to open IAR workspace from example folder. Open the desired *.eww file and IAR workbench will open a new window.

- On the left side you should see the workspace and all sources files. You shouldn't have to reconfigure the tools but you can check by left click on project options in " general Options" -> "Target" that the core is properly set to Cortex-M4F. And in debugger Category, check that ST-LINK is selected.



Options for node "example_idd_icm20602_cmodel"

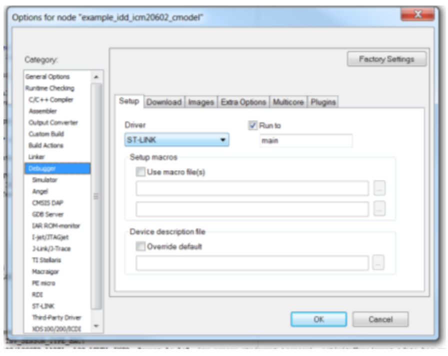
Category: General Options

Target

Processor variant: Core [✓] Cortex-M4F; Device: None

Endian mode: Little [✓]

FPU: VFPv4



Options for node "example_idd_icm20602_cmodel"

Category: Debugger

Setup

Driver: ST-LINK

Run to [✓]: main

Setup macros: Use macro file(s) [ ]

Device description file: Override default [ ]

Figure 9-10 – IAR project options

6.2 COMPILING AND DOWNLOADING

- Connect ST Nucleo board through Micro USB and let the windows install the driver.

- On the top of IAR IDE, select Download and Debug.

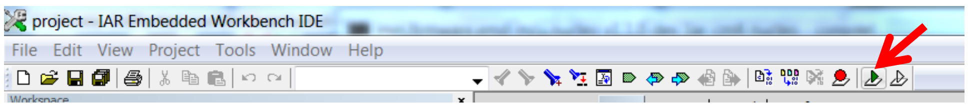


Figure 11 – IAR download and debug

- This will download the application on the ST Nucleo board. You should be able to debug the firmware from the main.c file.

7 APPENDIX C – SENSOR-CLI QUICKSTART

7.1 OVERVIEW

sensor-cli is a command line application used to control InvenSense device from a PC for evaluation and testing purpose.

Included features, non-exhaustive list:

- Start/stop/configure sensors for an InvenSense devices
- Display data on the screen
- Log sensor data to file

It is embeds support for several 'adapters' (in charge of communicating with the device), such as:

- Cheetah (USB/SPI adapter from TotalPhase)
- Aardvark (USB/I2C adapter from TotalPhase)
- Nucleo/Arduino bridge (custom USB/SPI bridge running on Nucleo or Arduino board)

7.2 USAGE

Being a command line application, *sensor-cli* should be launched from a console with few mandatory arguments.

sensor-cli --help will provide brief information about sensor-cli invocation.

The main option to pass to sensor-cli are the target device (--*target*) and adapter to use (--*adapter*).

Another useful option is --*level* to enable diagnostic messages (eg: --level=debug)

Example: *sensor-cli --target=emdwrapicm20x48,port=\\.\COMxx --adapter=dummy --level=debug*

If the device is correctly connected and setup successful, *sensor-cli* will display a prompt and wait for command to be input by the user.

The '*help*' command will provide a list of all supported command with a short description. Adding the command name to the 'help' command will give detailed information about a specific command.

```
sensor-cli> help en

Start a sensor designated by its short name or its id.

If 'period' and 'timeout' are given, this command will also configure the sensor period and sensor batch
timeout.

Synopsys: enable sensor_id [period_ms] [timeout_ms]

          enable sensor_name [period_ms] [timeout_ms]
```

The '*quit*' or '*exit*' command should be used to properly exit *sensor-cli*

7.2.1 Controlling sensors

Sensor can be enabled and disabled using the '*en*' (for enable) and '*dis*' (for disable) commands. Output data rate can be changed by the '*odr*' command.

Those commands expect as a first argument, the sensor id or name to be controlled. Those can be obtained with the '*ids*' command:

```
sensor-cli> ids

  1 [0x01] acc - SENSOR_ACCELEROMETER

 13 [0x0d] atemp - SENSOR_AMBIENT_TEMPERATURE

 29 [0x1d] axis - SENSOR_3AXIS

 28 [0x1c] b2s - SENSOR_B2S

 26 [0x1a] bac - SENSOR_BAC

 20 [0x14] georv - SENSOR_GEOMAG_ROTATION_VECTOR

 24 [0x18] glance - SENSOR_GLANCE_GESTURE

  9 [0x09] gra - SENSOR_GRAVITY
 15 [0x0f] grv - SENSOR_GAME_ROTATION_VECTOR
  4 [0x04] gyr - SENSOR_GYROSCOPE
 21 [0x15] hrm - SENSOR_HEART_RATE
 12 [0x0c] humidity - SENSOR_HUMIDITY
  5 [0x05] light - SENSOR_LIGHT
 10 [0x0a] linacc - SENSOR_LINEAR_ACCELERATION
  2 [0x02] mag - SENSOR_MAGNETOMETER
  0 [0x00] reserved - SENSOR_RESERVED
  3 [0x03] ori - SENSOR_ORIENTATION
 27 [0x1b] pdr - SENSOR_PDR
...
```

In order to start calibrated accelerometer, one would use the command '*en acc*' and in order to start the RAW gyroscope '*en rgyr*'.

In order to find out if your device supports a certain sensor, use the '*ping*' command.

7.2.2 Display quarternion as a cube

The 'cube' command allows the user to display a rotating cube based on the quaternion data.

To display orientation data corresponding to GAME ROTATION VECTOR sensor, the command would be '*cube on grv*' and '*cube off grv*' to close the cube windows.

Important note: the cube display is independent of the sensor control. Notice that grv sensor must be started separately with command *'en grv'*.

7.2.3 Redirecting sensor events

By default, sensor events are displayed to the main windows (the one that invocates sensor-cli), which can make it difficult to enter commands while events are reported.

It is possible to redirect sensor-events to:

- a file with '*disp > file_name*'
- the void with '*disp off*'
- to another window with '*disp >| named_pipe*'

The later command relies on operating system named-pipe. The *pipe-cat* is provided with *sensor-cli* to easily create named pipe under Windows. See *pipe-cat –help*

7.2.4 Examples

We will describe 2 examples in this section :

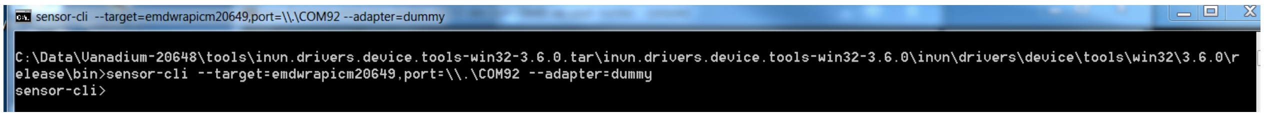
- how to display a cube using Rotation Vector ?

- how to log data from accelerometer?

Both example will use Icm20x48 device connected with SPI, you will have to setup full hardware connection .

run sensor-cli from a console with emdwrapicm20x48 target and nucleo adapter :

```
sensor-cli --target=emdwrapicm20x48,port=\\.\COM92 --adapter=dummy
```



```
sensor-cli  --target=emdwrapicm20649,port=\\.\COM92 --adapter=dummy

C:\Data\Vanadium-20648\tools\invn.drivers.device.tools-win32-3.6.0.tar\invn.drivers.device.tools-win32-3.6.0\invn\drivers\device\tools\win32\3.6.0\r
elease\bin>sensor-cli --target=emdwrapicm20649,port=\\.\COM92 --adapter=dummy
sensor-cli>
```

Your console should look like the one in the above picture.

7.2.4.1 How to display a cube using Rotation Vector

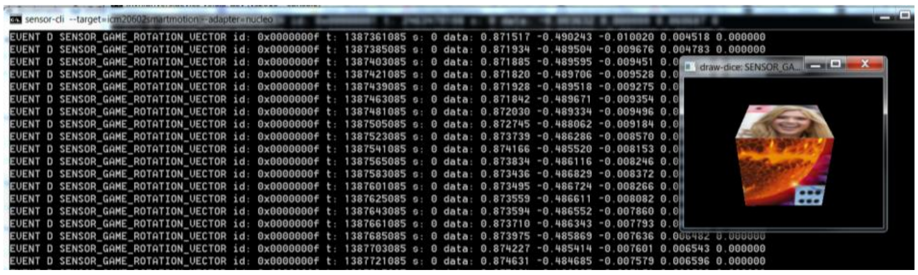
In order to have a good quaternion behavior, it is recommended to calibrate accelerometer and gyroscope before. Now enable the cube display for Game Rotation Vector Sensor (GRV) :

```
sensor-cli> cube on grv
```

A new window displaying a cube will be opened.

Now enable the GRV with a data output period of 20 ms by typing:

```
sensor-cli> en grv 20
```


sensor-cli --target=icm20602smartmotion --adapter=nucleo

```
EVENT D SENSOR_GAME_ROTATION_VECTOR id: 0x0000000f t: 1387361085 s: 0 data: 0.871517 -0.490243 -0.010020 0.004518 0.000000
EVENT D SENSOR_GAME_ROTATION_VECTOR id: 0x0000000f t: 1387385085 s: 0 data: 0.871934 -0.489504 -0.009676 0.004783 0.000000
EVENT D SENSOR_GAME_ROTATION_VECTOR id: 0x0000000f t: 1387403085 s: 0 data: 0.871885 -0.489595 -0.009451 0.0
EVENT D SENSOR_GAME_ROTATION_VECTOR id: 0x0000000f t: 1387421085 s: 0 data: 0.871820 -0.489706 -0.009528 0.0
EVENT D SENSOR_GAME_ROTATION_VECTOR id: 0x0000000f t: 1387439085 s: 0 data: 0.871928 -0.489518 -0.009275 0.0
EVENT D SENSOR_GAME_ROTATION_VECTOR id: 0x0000000f t: 1387463085 s: 0 data: 0.871842 -0.489671 -0.009354 0.0
EVENT D SENSOR_GAME_ROTATION_VECTOR id: 0x0000000f t: 1387481085 s: 0 data: 0.872030 -0.489334 -0.009496 0.0
EVENT D SENSOR_GAME_ROTATION_VECTOR id: 0x0000000f t: 1387505085 s: 0 data: 0.872745 -0.488062 -0.009184 0.0
EVENT D SENSOR_GAME_ROTATION_VECTOR id: 0x0000000f t: 1387523085 s: 0 data: 0.873739 -0.486286 -0.008570 0.0
EVENT D SENSOR_GAME_ROTATION_VECTOR id: 0x0000000f t: 1387541085 s: 0 data: 0.874166 -0.485520 -0.008153 0.0
EVENT D SENSOR_GAME_ROTATION_VECTOR id: 0x0000000f t: 1387565085 s: 0 data: 0.873834 -0.486116 -0.008246 0.0
EVENT D SENSOR_GAME_ROTATION_VECTOR id: 0x0000000f t: 1387583085 s: 0 data: 0.873436 -0.486829 -0.008372 0.0
EVENT D SENSOR_GAME_ROTATION_VECTOR id: 0x0000000f t: 1387601085 s: 0 data: 0.873495 -0.486724 -0.008266 0.0
EVENT D SENSOR_GAME_ROTATION_VECTOR id: 0x0000000f t: 1387625085 s: 0 data: 0.873559 -0.486611 -0.008082 0.0
EVENT D SENSOR_GAME_ROTATION_VECTOR id: 0x0000000f t: 1387643085 s: 0 data: 0.873594 -0.486552 -0.007860 0.0
EVENT D SENSOR_GAME_ROTATION_VECTOR id: 0x0000000f t: 1387661085 s: 0 data: 0.873710 -0.486343 -0.007793 0.0
EVENT D SENSOR_GAME_ROTATION_VECTOR id: 0x0000000f t: 1387685085 s: 0 data: 0.873975 -0.485869 -0.007636 0.006482 0.000000
EVENT D SENSOR_GAME_ROTATION_VECTOR id: 0x0000000f t: 1387703085 s: 0 data: 0.874227 -0.485414 -0.007601 0.006543 0.000000
EVENT D SENSOR_GAME_ROTATION_VECTOR id: 0x0000000f t: 1387721085 s: 0 data: 0.874631 -0.484685 -0.007579 0.006596 0.000000
```

Data will be displayed and the cube will move according to the board motion.

Then you can stop the sensor:

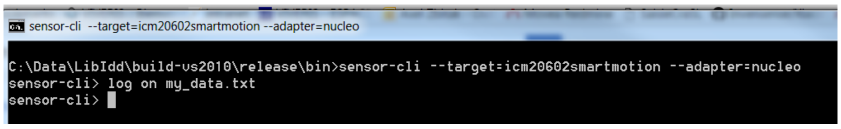
```
sensor-cli> dis grv
```

Data output will be stopped.

7.2.4.2 How to log data

Firstly, enable log and set a destination file. If you don't set a path, the file will be created in the current directory.

```
sensor-cli> log on my_data.txt
```


sensor-cli --target=icm20602smartmotion --adapter=nucleo

```
C:\Data\LibIdd\build-vs2010\release\bin>sensor-cli --target=icm20602smartmotion --adapter=nucleo
sensor-cli> log on my_data.txt
sensor-cli>
```

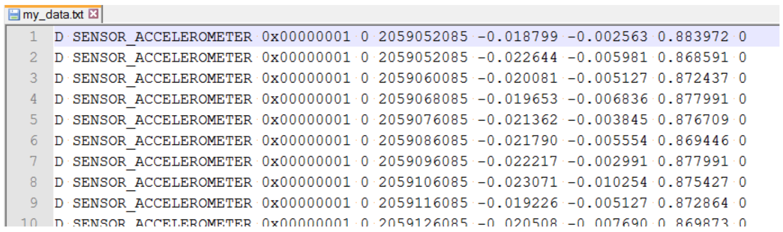
Then enable sensors, eg: accelerometer with 10 ms data rate interval.

```
sensor-cli> en acc 10
```

After you performing your tests you can disable the sensor.

```
sensor-cli> dis acc 10
```

Finally, you can check data from your log file:


my_data.txt

```
D SENSOR_ACCELEROMETER 0x00000001 0 2059052085 -0.018799 -0.002563 0.883972 0
D SENSOR_ACCELEROMETER 0x00000001 0 2059052085 -0.022644 -0.005981 0.868591 0
D SENSOR_ACCELEROMETER 0x00000001 0 2059060085 -0.020081 -0.005127 0.872437 0
D SENSOR_ACCELEROMETER 0x00000001 0 2059068085 -0.019653 -0.006836 0.877991 0
D SENSOR_ACCELEROMETER 0x00000001 0 2059076085 -0.021362 -0.003845 0.876709 0
D SENSOR_ACCELEROMETER 0x00000001 0 2059086085 -0.021790 -0.005554 0.869446 0
D SENSOR_ACCELEROMETER 0x00000001 0 2059096085 -0.022217 -0.002991 0.877991 0
D SENSOR_ACCELEROMETER 0x00000001 0 2059106085 -0.023071 -0.010254 0.875427 0
D SENSOR_ACCELEROMETER 0x00000001 0 2059116085 -0.019226 -0.005127 0.872864 0
D SENSOR_ACCELEROMETER 0x00000001 0 2059126085 -0.020508 -0.007690 0.869873 0
```

8 DOCUMENT INFORMATION

8.1 REVISION HISTORY

REVISION	DATE	DESCRIPTION	AUTHOR
1.0	January 10, 2017	Initial version.	Axel Zbitak
1.1	January 20, 2017	Update code-size, supported sensors and supported frequencies	Loic Michallon
1.2	February 17, 2017	Update *hardware setup* chapter. Update *system known issues* chapter. Update frequencies table. Update code-size.	Bogdan Birsan

Table 4. Revision History