InvenSense

Tools User Guide

For

eMD Projects (Nucleo-based)

TABLE OF CONTENTS

1 USEFUL LINKS

InvenSense website:

http://www.InvenSense.com/

IAR website:

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

2 OVERVIEW

The purpose of this document is to give an overview of the set of tools available to use sample applications provided within the eMD Developer Kit that will allow users to create an application based on motion sensors.

2.1 INTRODUCTION

The 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 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.

2.2 EMD BASICS

The eMD Developer Kit provides a software guide to setup eMD platform hardware and flash it with appropriate software. This should be read in the first place.

3 IAR IDE QUICK START

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

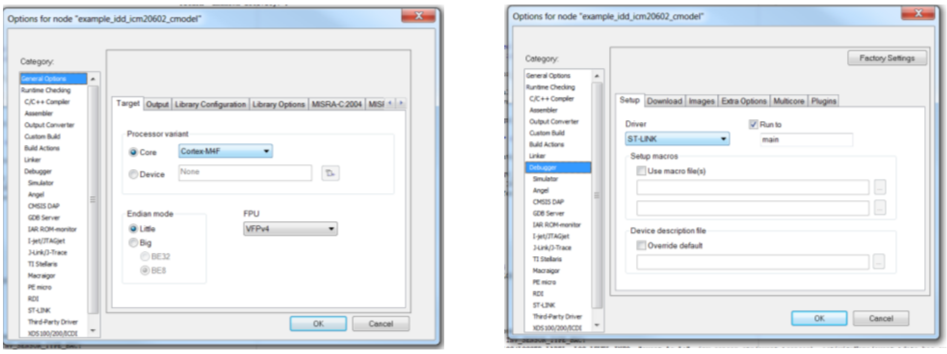
Please refer to ST or IAR documentation for additional information.

3.1 INSTALLING IAR WORKBENCH IDE

- Download IAR workbench IDE from https://www.iar.com/iar-embedded-workbench/. You will have to get a valid license to use it. The required 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; ○ Big (○ BE32, ◉ BE8). 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

3.2 COMPILING AND DOWNLOAD

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

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

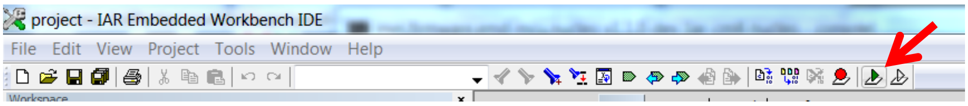


Figure 11 – IAR download and debug

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

4 SENSOR-CLI QUICKSTART

4.1 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 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 brige running on Nucleo or Arduino board)

It also embeds IDD Wrapper communication support; in this case, the target is the embedded wrapper. The real InvenSense device ICM is actually set in the firmware.

4.2 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.

Two main arguments must be passed to *sensor-cli* :

- The targeted device: `--target`
- The adapter to use: `--adapter`

The *target* allows you to specify which chip is connected. In the case of our Nucleo-based applications, the target will be one of the following:

- *20602 or 20603 or 20690 or 20789:* `emdwrapicm20xxx`
- *20649:* `emdwrapicm20649`

The adapter allows you to specify how the board is connected to your computer. In the case of our Nucleo-based applications, the adapter must be set to `dummy`.

For our Nucleo-based applications, you must specify the COM port used (where your USB cable is connected). You can specify it using the `port=\\.\COMxx` argument.

The usual command to run would then be:

```
sensor-cli.exe --adapter=dummy --target=emdwrapicm20xxx,port=\\.\COMxx
```

A useful option is `--level` to enable diagnostic messages (eg: `--level=debug`)

Example: `sensor-cli --target=emdwrapper,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.

Once it is started, the `help` command will provide a list of all supported commands 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*

4.2.1 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.

4.2.2 Display quaternion 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`.

4.2.3 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*

4.3 EXAMPLES

We will describe 2 examples in this section:

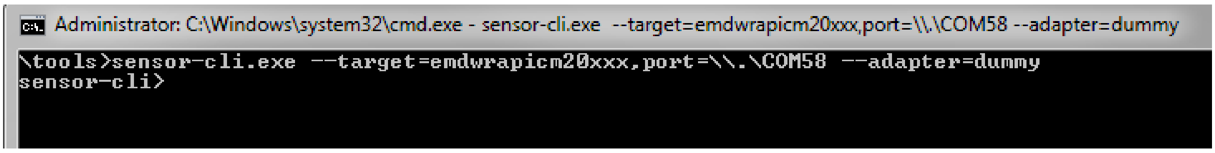
- how to display a cube using Rotation Vector?

- how to log data from accelerometer?

Both example will use an ICM-20xxx device connected to a Nucleo board through one of our provided firmware.

You will first need to run *sensor-cli* from a console with appropriate target and adapter :

```
sensor-cli.exe --adapter=dummy --target=emdwrapicm20xxx,port=\\.\COMxx
```



Administrator: C:\Windows\system32\cmd.exe - sensor-cli.exe --target=emdwrapicm20xxx,port=\\.\COM58 --adapter=dummy

```
\tools>sensor-cli.exe --target=emdwrapicm20xxx,port=\\.\COM58 --adapter=dummy
sensor-cli>
```

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

4.3.1 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.

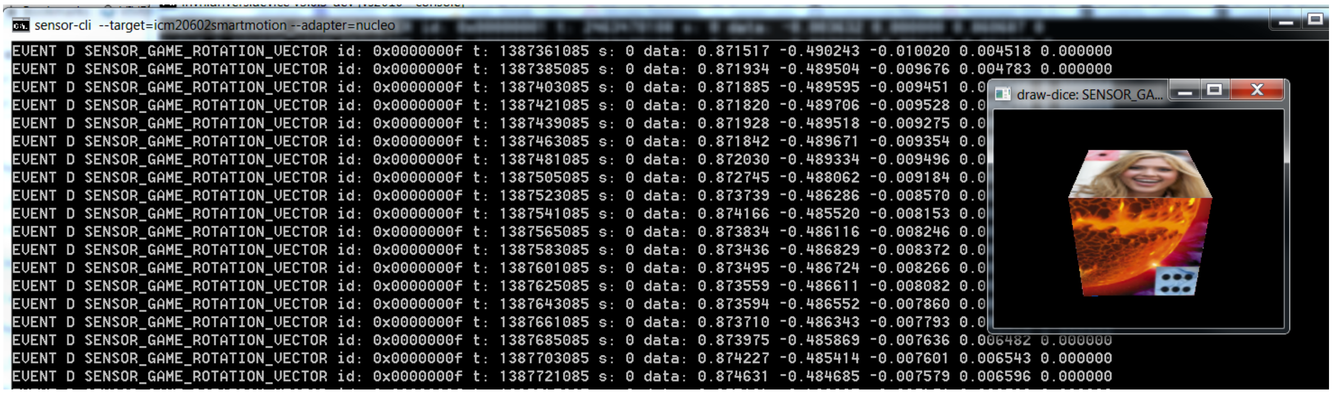
Now enable the cube display for Game Rotation Vector sensor (GRV):

```
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 msec 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:

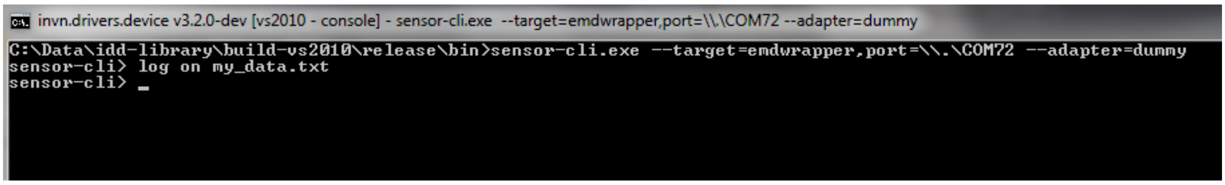
```
sensor-cli> dis grv
```

Data output will be stopped.

4.3.2 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
```



invn.drivers.device v3.2.0-dev [vs2010 - console] - sensor-cli.exe --target=emdwrapper,port=\\.\COM72 --adapter=dummy

```
C:\Data\idd-library\build-vs2010\release\bin>sensor-cli.exe --target=emdwrapper,port=\\.\COM72 --adapter=dummy
sensor-cli> log on my_data.txt
sensor-cli> _
```

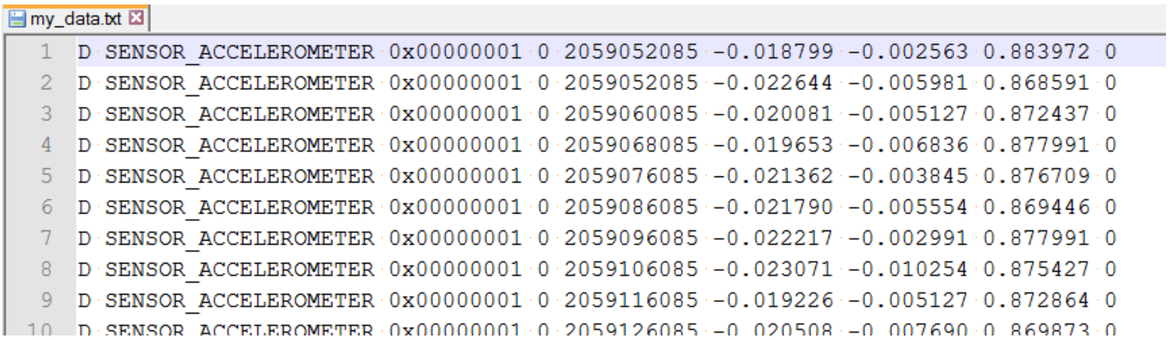
Then enable sensors, eg: accelerometer with 10 msec data rate interval.

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

After you performing your tests you can disable the sensor.

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

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
```