Getting started with MotionID motion intensity detection library in X-CUBE-MEMS1 expansion for STM32Cube

Introduction

The MotionID is a middleware library part of X-CUBE-MEMS1 software and runs on STM32. It provides real-time information about the user motion intensity based on data from a device.

It is able to distinguish motion intensity in a range from 0 to 10 (according to the library indexes) corresponding to the following activities: on desk, hand on bed/couch/cushion, light movements, biking, typing/writing, high intensity typing/slow walking, washing hands/walking, fast walking/jogging, running/brushing teeth, sprinting. The library is intended for wrist-based devices.

This library is intended to work with ST MEMS only.

The algorithm is provided in static library format and is designed to be used on STM32 microcontrollers based on the ARM® Cortex®-M0+, ARM® Cortex®-M3, ARM® Cortex®-M33, ARM® Cortex®-M4 or ARM® Cortex®-M7 architecture.

It is built on top of STM32Cube software technology that eases portability across different STM32 microcontrollers.

The software comes with sample implementation running on X-NUCLEO-IKS01A3 or X-NUCLEO-IKS4A1 expansion board on a NUCLEO-F401RE, NUCLEO-L073RZ, NUCLEO-L152RE or NUCLEO-U575ZI-Q development board.

1 Acronyms and abbreviations

Table 1. List of acronyms

Acronym	Description
API	Application programming interface
BSP	Board support package
GUI	Graphical user interface
HAL	Hardware abstraction layer
IDE	Integrated development environment

2 MotionID middleware library in X-CUBE-MEMS1 software expansion for STM32Cube

2.1 MotionID overview

The MotionID library expands the functionality of the X-CUBE-MEMS1 software.

The library acquires data from the accelerometer and provides information about the user motion intensity based on data from a device.

The library is designed for ST MEMS only. Functionality and performance when using other MEMS sensors are not analyzed and can be significantly different from what described in the document.

Sample implementation is available for X-NUCLEO-IKS01A3 or X-NUCLEO-IKS4A1 expansion board mounted on a NUCLEO-F401RE, NUCLEO-L073RZ, NUCLEO-L152RE or NUCLEO-U575ZI-Q development board.

2.2 MotionID library

Technical information fully describing the functions and parameters of the MotionID APIs can be found in the MotionID_Package.chm compiled HTML file located in the Documentation folder.

2.2.1 MotionID library description

The MotionID intensity detection library manages the data acquired from the accelerometer; it features:

- possibility to distinguish motion intensity in a range from 0 to 10 (according to the library indexes) corresponding to the following activities: on desk, hand on bed/couch/cushion, light movements, biking, typing/writing, high intensity typing/slow walking, washing hands/walking, fast walking/jogging, running/ brushing teeth, sprinting.
- intended for wrist-based devices
- recognition based only on accelerometer data
- required accelerometer data sampling frequency of 16 Hz
- resources requirements:
  - Cortex-M0+: 1.2 kB of code and 0.6 kB of data memory
  - Cortex-M3: 1.2 kB of code and 0.6 kB of data memory
  - Cortex-M33: 1.2 kB of code and 0.6 kB of data memory
  - Cortex-M4: 1.1 kB of code and 0.6 kB of data memory
  - Cortex-M7: 1.2 kB of code and 0.6 kB of data memory
- available for ARM Cortex-M0+, ARM Cortex-M3, ARM Cortex-M33, ARM Cortex-M4 or ARM Cortex-M7 architectures

2.2.2 MotionID APIs

The MotionID library APIs are:

- `uint8_t MotionID_GetLibVersion(char *version)`
  - retrieves the library version
  - `*version` is a pointer to an array of 35 characters
  - returns the number of characters in the version string
- `void MotionID_Initialize(MID_mcu_type_t mcu_type)`
  - performs MotionID library initialization and setup of the internal mechanism
  - the CRC module in STM32 microcontroller (in RCC peripheral clock enable register) has to be enabled before using the library
  - `mcu_type` – type of MCU used

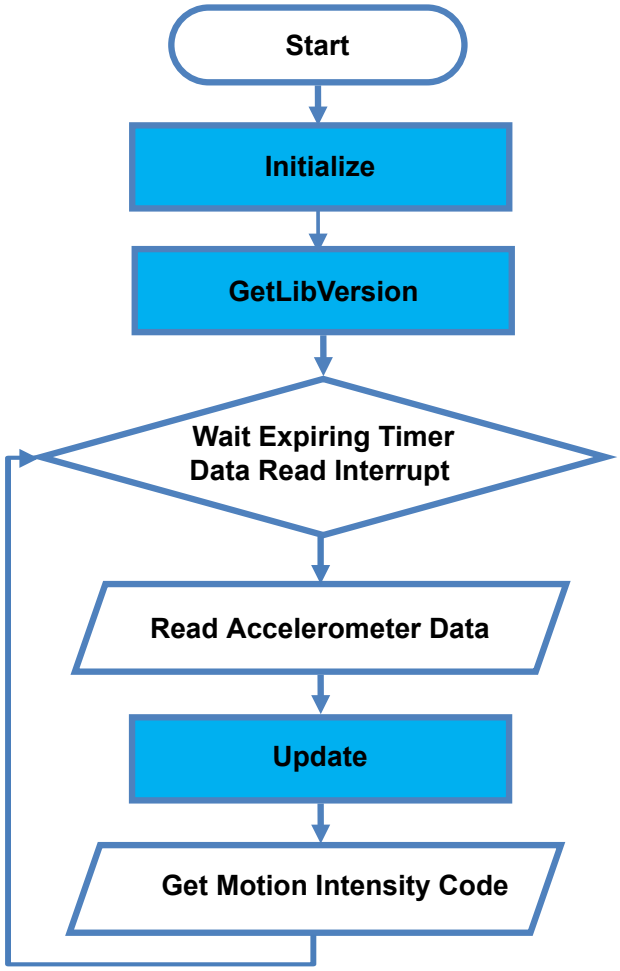
Note: This function must be called before using the intensity detection library.

- `void MotionID_ResetLib(void)`
  - resets the library

- `void MotionID_Update(MID_input_t *data_in, MID_output_t *data_out)`
  - executes intensity detection algorithm
  - `*data_in` parameter is a pointer to a structure with input data
  - the parameters for the structure type `MID_input_t` are:
    - `AccX` is the accelerometer sensor value in X axis in g
    - `AccY` is the accelerometer sensor value in Y axis in g
    - `AccZ` is the accelerometer sensor value in Z axis in g
  - `*data_out` parameter is a pointer to an enum with the following with:
    - `MID_ON_DESK = 0`
    - `MID_BED_COUCH_PILLOW = 1`
    - `MID_LIGHT_MOVEMENTS = 2`
    - `MID_BIKING = 3`
    - `MID_TYPING_WRITING = 4`
    - `MID_HI_TYPING__SLOW_WALKING = 5`
    - `MID_WASHING_HANDS_WALKING = 6`
    - `MID_FWALKING = 7`
    - `MID_FWALKING_JOGGING = 8`
    - `MID_FJOGGING_BRUSHING = 9`
    - `MID_SPRINTING = 10`

2.2.3 API flow chart

Figure 1. MotionID API logic sequence



2.2.4 Demo code

The following demonstration code reads data from the accelerometer sensor and gets the motion intensity code.

```
[…]
#define VERSION_STR_LENG 35
[…]

/*** Initialization ***/
char lib_version[VERSION_STR_LENG];

/* Intensity Detection API initialization function */
MotionID_Initialize(MID_MCU_STM32);

/* Optional: Get version */
MotionID_GetLibVersion(lib_version);

[…]

/*** Using Intensity Detection algorithm ***/
Timer_OR_DataRate_Interrupt_Handler()
```

```
{
MID_input_t data_in;
MID_output_t data_out;

/* Get acceleration X/Y/Z in g */
MEMS_Read_AccValue(&data_in.AccX, &data_in.AccY, &data_in.AccZ);

/* Intensity Detection algorithm update */
MotionID_Update(&data_in, &data_out);
}
```

2.2.5 Algorithm performance

The intensity detection algorithm only uses data from the accelerometer and runs at a low frequency (16 Hz) to reduce power consumption.

Algorithm latency is 1 second.

Table 2. Cortex-M4 and Cortex-M3: elapsed time (µs) algorithm

Cortex-M4 STM32F401RE at 84 MHz			Cortex-M3 STM32L152RE at 32 MHz		
Min	Avg	Max	Min	Avg	Max
1	2	32	3	26	756

Table 3. Cortex-M0, Cortex-M33 and Cortex-M7: elapsed time (µs) algorithm

Cortex-M0+ STM32L073RZ at 32 MHz			Cortex-M33 STM32U575ZI-Q at 160 MHz			Cortex-M7 STM32F767ZI at 96 MHz		
Min	Avg	Max	Min	Avg	Max	Min	Avg	Max
<1	64	2000	<1	<1	16	2	3	49

2.3 Sample application

The MotionID middleware can be easily manipulated to build user applications; a sample application is provided in the Application folder.

It is designed to run on a NUCLEO-F401RE, NUCLEO-L073RZ, NUCLEO-L152RE or NUCLEO-U575ZI-Q development board connected to an X-NUCLEO-IKS01A3 or X-NUCLEO-IKS4A1 expansion board.

The application recognizes the user motion intensity in real-time. Data can be displayed through a GUI.

USB cable connection is required to monitor real-time data. The board is powered by the PC via USB connection. This allows the user to display detected user motion intensity, accelerometer data, time stamp and eventually other sensor data, in real-time, using the **MEMS-Studio** GUI application.

2.4 MEMS Studio

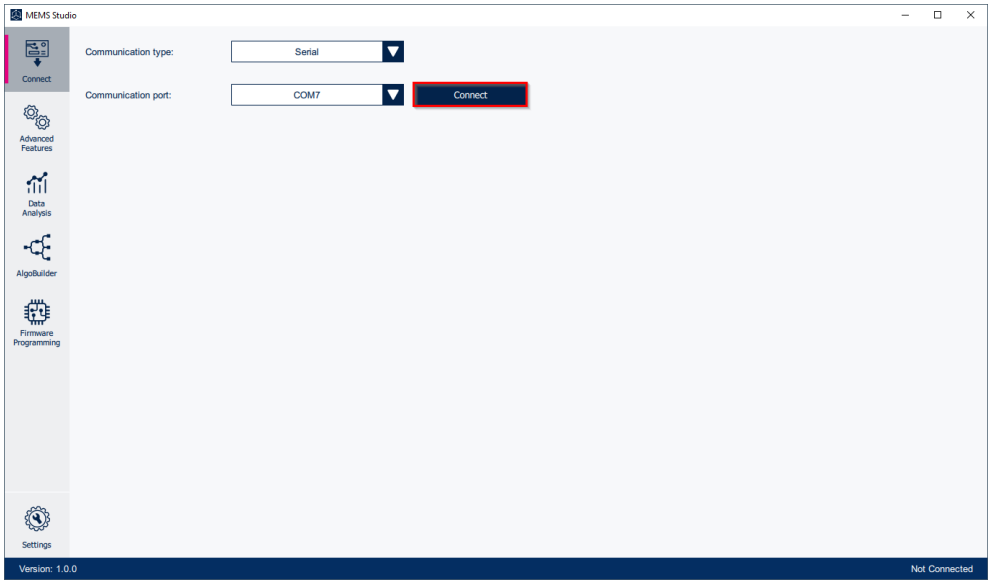
The sample application uses the MEMS-Studio GUI application, which can be downloaded from www.st.com.

Step 1. Ensure that the necessary drivers are installed and the STM32 Nucleo board with appropriate expansion board is connected to the PC.

Step 2. Launch the MEMS-Studio application to open the main application window.

If an STM32 Nucleo board with supported firmware is connected to the PC, it is automatically detected the appropriate COM port. Press **Connect** button to open this port.

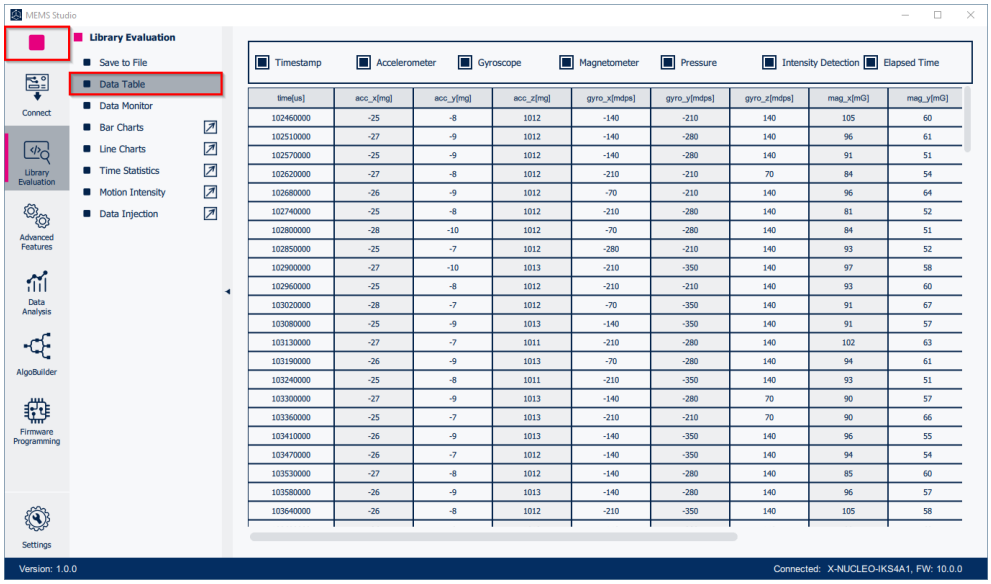
Figure 2. MEMS-Studio - Connect



Step 3. When connected to STM32 Nucleo board with supported firmware *Library Evaluation* tab is opened.

To start and stop data streaming toggle the appropriate start / stop button on the outer vertical tool bar.
The data coming from the connected sensor can be viewed selecting the *Data Table* tab on the inner vertical tool bar.

Figure 3. MEMS-Studio - Library Evaluation - Data Table



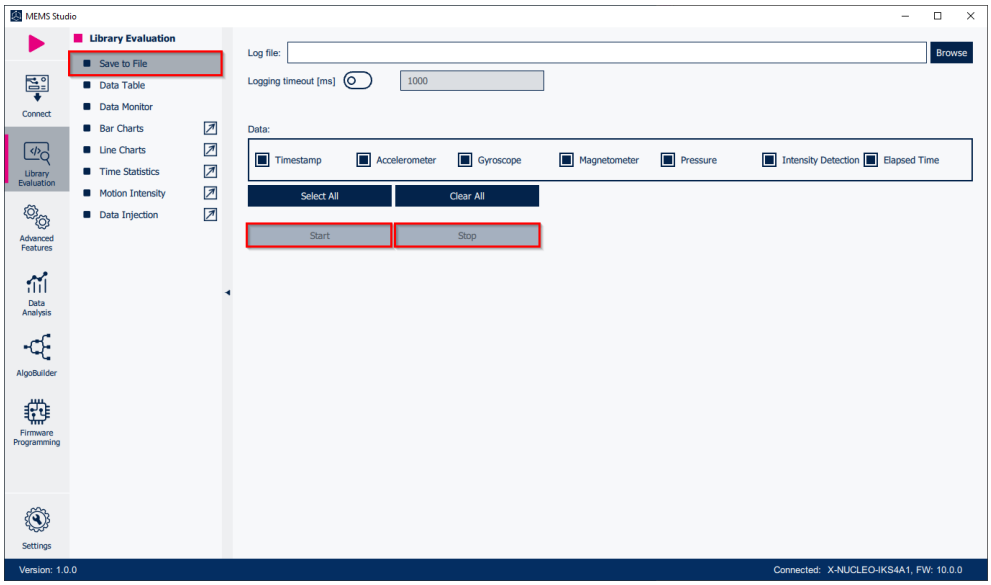
time[us]	acc_x[mg]	acc_y[mg]	acc_z[mg]	gyro_x[mdps]	gyro_y[mdps]	gyro_z[mdps]	mag_x[mG]	mag_y[mG]
102460000	-25	-8	1012	-140	-210	140	105	60
102510000	-27	-9	1012	-140	-280	140	96	61
102570000	-25	-9	1012	-140	-280	140	91	51
102620000	-27	-8	1012	-210	-210	70	84	54
102680000	-26	-9	1012	-70	-210	140	96	64
102740000	-25	-8	1012	-210	-280	140	81	52
102800000	-28	-10	1012	-70	-280	140	84	51
102850000	-25	-7	1012	-280	-210	140	93	52
102900000	-27	-10	1013	-210	-350	140	97	58
102960000	-25	-8	1012	-210	-210	140	93	60
103020000	-28	-7	1012	-70	-350	140	91	67
103080000	-25	-9	1013	-140	-350	140	91	57
103130000	-27	-7	1011	-210	-280	140	102	63
103190000	-26	-9	1013	-70	-280	140	94	61
103240000	-25	-8	1011	-210	-350	140	93	51
103300000	-27	-9	1013	-140	-280	70	90	57
103360000	-25	-7	1013	-210	-210	70	90	66
103410000	-26	-9	1013	-140	-350	140	96	55
103470000	-26	-7	1012	-140	-350	140	94	54
103530000	-27	-8	1012	-140	-280	140	85	60
103580000	-26	-9	1013	-140	-280	140	96	57
103640000	-26	-8	1012	-210	-350	140	105	58

Step 4. Select the *Motion Intensity* tab on the inner vertical tool bar to open the dedicated application status view.

Figure 4. MEMS-Studio - Library Evaluation - Motion Intensity

Step 5. Select the *Save to File* tab on the inner vertical tool bar to open the data logging configuration window. Select which sensor and activity data to save to log file. You can start or stop saving by clicking on the corresponding Start / Stop button.

Figure 5. MEMS-Studio - Library Evaluation - Save to File



3 References

All of the following resources are freely available on www.st.com.

1. UM1859: Getting started with the X-CUBE-MEMS1 motion MEMS and environmental sensor software expansion for STM32Cube
2. UM1724: STM32 Nucleo-64 boards (MB1136)
3. UM3233: Getting started with MEMS-Studio

Revision history

Table 4. **Document revision history**

Date	Version	Changes
10-May-2017	1	Initial release.
26-Jan-2018	2	Added references to NUCLEO-L152RE development board and Table 2. Elapsed time (µs) algorithm.
20-Mar-2018	3	Updated Introduction and Section 2.1 MotionID overview.
21-Nov-2018	4	Updated Figure 2. STM32 Nucleo: LEDs, button, jumper. Added Table 3. Cortex-M0+: elapsed time (µs) algorithm and references to ARM® Cortex®-M0+ and NUCLEO-L073RZ development board.
20-Feb-2019	5	Updated Table 2. Cortex-M4 and Cortex-M3: elapsed time (µs) algorithm, Table 3. Cortex-M0+: elapsed time (µs) algorithm, Figure 3. Unicleo main window, Figure 4. User Messages tab, Figure 5. Motion intensity detection window and Figure 6. Datalog window.
25-Mar-2020	6	Updated Introduction, Section 2.2.1 MotionID library description and Section 2.2.5 Algorithm performance. Added ARM Cortex-M7 architecture compatibility information.
28-Mar-2024	7	Updated Section Introduction, Section 2.1: MotionID overview, Section 2.2: MotionID library, Section 2.3: Sample application and Section 2.4: MEMS Studio.

Contents

1 **Acronyms and abbreviations** 2

2 **MotionID middleware library in X-CUBE-MEMS1 software expansion for STM32Cube** 3

2.1 MotionID overview 3

2.2 MotionID library 3

2.2.1 MotionID library description 3

2.2.2 MotionID APIs 3

2.2.3 API flow chart 5

2.2.4 Demo code 5

2.2.5 Algorithm performance 6

2.3 Sample application 6

2.4 MEMS Studio 6

3 **References** 9

Revision history 10

List of tables

Table 1. List of acronyms . . . 2
Table 2. Cortex-M4 and Cortex-M3: elapsed time (µs) algorithm . . . 6
Table 3. Cortex-M0, Cortex-M33 and Cortex-M7: elapsed time (µs) algorithm . . . 6
Table 4. Document revision history . . . 10

List of figures

Figure 1. MotionID API logic sequence 5
Figure 2. MEMS-Studio - Connect 7
Figure 3. MEMS-Studio - Library Evaluation - Data Table 7
Figure 4. MEMS-Studio - Library Evaluation - Motion Intensity 8
Figure 5. MEMS-Studio - Library Evaluation - Save to File 8