Introduction

The MotionSD middleware library is part of the X-CUBE-MEMS1 software and runs on STM32. It provides real-time information about the user working mode: sitting at the desk or standing desk position. The library is intended for wrist-worn 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®-M3 or ARM® Cortex®-M4 architecture.

It is built on top of STM32Cube software technology to ease portability across different STM32 microcontrollers.

The software comes with a sample implementation running on an X-NUCLEO-IKS01A2 or X-NUCLEO-IKS01A3 expansion board on NUCLEO-F401RE, NUCLEO-L476RG or NUCLEO-L152RE development board.
1 Acronyms and abbreviations

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application programming interface</td>
</tr>
<tr>
<td>BSP</td>
<td>Board support package</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical user interface</td>
</tr>
<tr>
<td>HAL</td>
<td>Hardware abstraction layer</td>
</tr>
<tr>
<td>IDE</td>
<td>Integrated development environment</td>
</tr>
</tbody>
</table>
2 MotionSD middleware library in X-CUBE-MEMS1 software expansion for STM32Cube

2.1 MotionSD overview

The MotionSD library expands the functionality of the X-CUBE-MEMS1 software. The library acquires data from the accelerometer and pressure sensor and provides information about the user position. It is able to distinguish the user working mode: sitting at the desk or standing desk position. 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.

A sample implementation is available for X-NUCLEO-IKS01A2 and X-NUCLEO-IKS01A3 expansion boards, mounted on a NUCLEO-F401RE, NUCLEO-L476RG or NUCLEO-L152RE development board.

2.2 MotionSD library

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

2.2.1 MotionSD library description

The MotionSD standing vs sitting desk detection library manages the data acquired from the accelerometer and pressure sensor; it features:

- detection of standing or sitting desk position
- recognition based on the accelerometer and pressure sensor data only
- required accelerometer and pressure sensor data sampling frequency of 25 Hz
- resources requirements:
  - Cortex-M3: 2.1 kB of code and 1.1 kB of data memory
  - Cortex-M4: 2.1 kB of code and 1.1 kB of data memory
- available for ARM® Cortex®-M3 and ARM Cortex-M4 architectures

2.2.2 MotionSD APIs

The MotionSD library APIs are:

- `uint8_t MotionSD_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 MotionSD_Initialize(void)`
  - performs MotionSD 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
  - Note: This function must be called before using the library.

- `void MotionSD_Update(MSD_input_t *data_in, MSD_output_t *data_out)`
  - executes standing vs sitting desk detection algorithm
  - `*data_in` parameter is a pointer to a structure with input data
  - the parameters for the structure type `MSD_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
    - `Press` is the atmospheric pressure in hPa
  - `*data_out` parameter is a pointer to an enum with the following items:
- MSD_UNKNOWN_DESK = 0
- MSD_SITTING_DESK = 1
- MSD_STANDING_DESK = 2

- void MotionSD_Reset(void)
  - resets standing vs sitting desk detection algorithm

- void MotionSD_SetOrientation_Acc(const char *acc_orientation)
  - this function is used to set the accelerometer data orientation
  - configuration is usually performed immediately after the MotionSD_Initialize function call
  - *acc_orientation parameter is a pointer to a string of three characters indicating the direction of each of the positive orientations of the reference frame used for accelerometer data output, in the sequence x, y, z. Valid values are: n (north) or s (south), w (west) or e (east), u (up) or d (down).
  - As shown in the figure below, the X-NUCLEO-IKS01A2 accelerometer sensor has an NWU orientation (x -North, y - West, z - Up), so the string is "nwu".

**Figure 1. Example of sensor orientations**
Figure 2. Orientation system for wrist-worn devices
2.2.3 API flow chart

Figure 3. MotionSD API logic sequence

2.2.4 Demo code
The following demonstration code reads data from the accelerometer sensor and gets the position code.

```
#define VERSION_STR_LEN 35
```
2.2.5 Algorithm performance

The standing vs sitting desk detection algorithm uses data from the accelerometer and pressure sensor and runs at a low frequency (25 Hz) to reduce power consumption.

The algorithm latency is 10 – 30 seconds.

Note: When testing the algorithm with the STM32 Nucleo board, ensure the board is oriented perpendicular to the forearm, to simulate wristband position.

Table 2. Elapsed time (µs) algorithm

<table>
<thead>
<tr>
<th></th>
<th>Cortex-M4 STM32F401RE at 84 MHz</th>
<th>Cortex-M3 STM32L152RE at 32 MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SW4STM32 2.6.0 (GCC 7.2.1)</td>
<td>SW4STM32 2.6.0 (GCC 7.2.1)</td>
</tr>
<tr>
<td></td>
<td>IAR EWARM 7.80.4</td>
<td>IAR EWARM 7.80.4</td>
</tr>
<tr>
<td></td>
<td>Keil μVision 5.24</td>
<td>Keil μVision 5.24</td>
</tr>
<tr>
<td></td>
<td>Min Ave Max</td>
<td>Min Ave Max</td>
</tr>
<tr>
<td></td>
<td>50 Ave Max 62</td>
<td>247 Ave 443 109</td>
</tr>
<tr>
<td></td>
<td>64 Ave Max 63</td>
<td>176 Ave 320 447</td>
</tr>
</tbody>
</table>

2.3 Sample application

The MotionSD 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-L476RG or NUCLEO-L152RE development board connected to an X-NUCLEO-IKS01A2 or X-NUCLEO-IKS01A3 expansion board.

The application recognizes a user position in real-time. The data can be displayed through a GUI or stored in the board for offline analysis.

Stand-alone mode

In stand-alone mode, the sample application allows the user to detect his position changes and store the information in the MCU flash memory.
The STM32 Nucleo board may be supplied by a portable battery pack (to make the user experience more comfortable, portable and free of any PC connections).

### Table 3. Power supply scheme

<table>
<thead>
<tr>
<th>Power source</th>
<th>JP1 settings</th>
<th>Working mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>USB PC cable</td>
<td>JP1 open</td>
<td>PC GUI driven mode</td>
</tr>
<tr>
<td>Battery pack</td>
<td>JP1 closed</td>
<td>Stand-alone mode</td>
</tr>
</tbody>
</table>

**Figure 4. STM32 Nucleo: LEDs, button, jumper**

The above figure shows the user button B1 and the three LEDs on the NUCLEO-F401RE board. Once the board is powered, LED LD3 (PWR) turns ON and the tricolor LED LD1 (COM) begins blinking slowly due to the missing USB enumeration (refer to UM1724 on www.st.com for further details).

**Note:** After powering the board, LED LD2 blinks once indicating the application is ready.

When the user button B1 is pressed, the system starts acquiring data from the accelerometer sensor and detects the carry position. During this acquisition mode, fast LED LD2 blinking indicates that the algorithm is running; the detected user pose is stored in the MCU internal flash memory. Data are automatically saved every 5 minutes to avoid excessive data loss in case of an unforeseen power fault.

Pressing button B1 a second time stops the algorithm and data storage and LED LD2 switches off. Pressing the button again starts the algorithm and data storage once again.

The flash sector dedicated to data storage is 128 KB, allowing memorization of more than 16,000 data sets.

To retrieve these data, the board must be connected to a PC running Unicleo-GUI. When stored data is retrieved via the GUI, the MCU flash sector dedicated to this purpose is cleared.

If LED LD2 is ON after powering the board, it represents a warning message indicating the flash memory is full.

**Note:** Optionally, the MCU memory can be erased by holding the user push button down for at least 5 seconds. LED LD2 switches OFF and then blinks 3 times to indicate that the data stored in the MCU has been erased. This
option is available only after power ON or reset of the board while LED LD2 is ON indicating the flash memory is full.

When the application runs in stand-alone mode and the flash memory is full, the application switches to PC GUI drive mode and LED LD2 switches OFF.

The flash memory must be erased by downloading data via the Unicleo-GUI or the user push button (see the above note).

**PC GUI drive mode**

In this mode, a USB cable connection is required to monitor real-time data. The board is powered by the PC via USB connection. This working mode allows the user to display real-time counter values, accelerometer and pressure data, time stamp and any other sensor data, using the Unicleo-GUI.

In this working mode, data are not stored in the MCU flash memory.

### 2.4 Unicleo-GUI application

The sample application uses the Windows Unicleo-GUI utility, 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 Unicleo-GUI application to open the main application window. If an STM32 Nucleo board with supported firmware is connected to the PC, it is automatically detected and the appropriate COM port is opened.

**Figure 5. Unicleo main window**

![Unicleo main window](image)

**Step 3.** Start and stop data streaming by using the appropriate buttons on the vertical tool bar. The data coming from the connected sensor can be viewed in the User Messages tab.
Step 4. Click on the Standing vs Sitting Desk icon in the vertical tool bar to open the dedicated application window.

If the board has been working in standalone mode and the user wants to retrieve stored data, press Download Off-line Data button to upload the stored activities data to the application. This operation automatically deletes acquired data from microcontroller.

Note: Download Off-line Data button is not available while data streaming is active.

Press the Save Off-line Data to File button to save the uploaded data in a .tsv file.

Step 5. Click on the Datalog icon in the vertical tool bar to open the datalog configuration window:
you can select which sensor and activity data to save in files. You can start or stop saving by clicking on the corresponding button.

**Figure 8. Datalog window**
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 board
3. UM2128: Getting started with Unicelo-GUI for motion MEMS and environmental sensor software expansion for STM32Cube
## Revision history

**Table 4. Document revision history**

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>22-Sep-2017</td>
<td>1</td>
<td>Initial release.</td>
</tr>
<tr>
<td>25-Jan-2018</td>
<td>2</td>
<td>Added references to NUCLEO-L152RE development board.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added Figure 2. Orientation system for wrist-worn devices and Table 2. Elapsed time (µs) algorithm.</td>
</tr>
<tr>
<td>21-Mar-2018</td>
<td>3</td>
<td>Updated Introduction and Section 2.1 MotionSD overview.</td>
</tr>
<tr>
<td>18-Feb-2019</td>
<td>4</td>
<td>Updated Table 2. Elapsed time (µs) algorithm.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added X-NUCLEO-IKS01A3 expansion board compatibility information.</td>
</tr>
</tbody>
</table>
## Contents

1. Acronyms and abbreviations .................................................................2

2. MotionSD middleware library in X-CUBE-MEMS1 software expansion for STM32Cube .................................................................3

   2.1 MotionSD overview ......................................................................3

   2.2 MotionSD library .........................................................................3

       2.2.1 MotionSD library description ...............................................3

       2.2.2 MotionSD APIs .................................................................3

       2.2.3 API flow chart ....................................................................5

       2.2.4 Demo code ........................................................................6

       2.2.5 Algorithm performance ....................................................7

   2.3 Sample application .......................................................................7

   2.4 Unicleo-GUI application ...........................................................9

3. References ......................................................................................12

Revision history ..................................................................................13
List of tables

Table 1. List of acronyms ................................................................. 2
Table 2. Elapsed time (µs) algorithm ........................................... 7
Table 3. Power supply scheme ....................................................... 8
Table 4. Document revision history ............................................... 13
## List of figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td>Example of sensor orientations</td>
<td>4</td>
</tr>
<tr>
<td>Figure 2</td>
<td>Orientation system for wrist-worn devices</td>
<td>5</td>
</tr>
<tr>
<td>Figure 3</td>
<td>MotionSD API logic sequence</td>
<td>6</td>
</tr>
<tr>
<td>Figure 4</td>
<td>STM32 Nucleo: LEDs, button, jumper</td>
<td>8</td>
</tr>
<tr>
<td>Figure 5</td>
<td>Unicleo main window</td>
<td>9</td>
</tr>
<tr>
<td>Figure 6</td>
<td>User Messages tab</td>
<td>10</td>
</tr>
<tr>
<td>Figure 7</td>
<td>Standing vs. Sitting Desk Detection window</td>
<td>10</td>
</tr>
<tr>
<td>Figure 8</td>
<td>Datalog window</td>
<td>11</td>
</tr>
</tbody>
</table>