Getting started with MotionSD standing vs sitting desk detection library in X-CUBE-MEMS1 expansion for STM32Cube

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, ARM® Cortex®-M33, ARM® Cortex®-M4 or ARM® Cortex®-M7 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-IKS01A3 or X-NUCLEO-IKS4A1 expansion board on NUCLEO-F401RE, 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 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-IKS01A3 or X-NUCLEO-IKS4A1 expansion boards, mounted on a NUCLEO-F401RE, NUCLEO-L152RE or NUCLEO-U575ZI-Q 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: 2.2 kB of code and 1.1 kB of data memory
  - Cortex-M33: 2.3 kB of code and 1.1 kB of data memory
  - Cortex-M4: 2.3. kB of code and 1.1 kB of data memory
  - Cortex-M7: 2.3 kB of code and 1.1 kB of data memory
- available for ARM® Cortex®-M3, ARM® Cortex®-M33, ARM Cortex-M4 or ARM Cortex-M7 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-IKS4A1 accelerometer sensor has an SEU orientation (x -South, y-East, z-Up), so the string is “seu”.

Figure 1. Example of sensor orientations

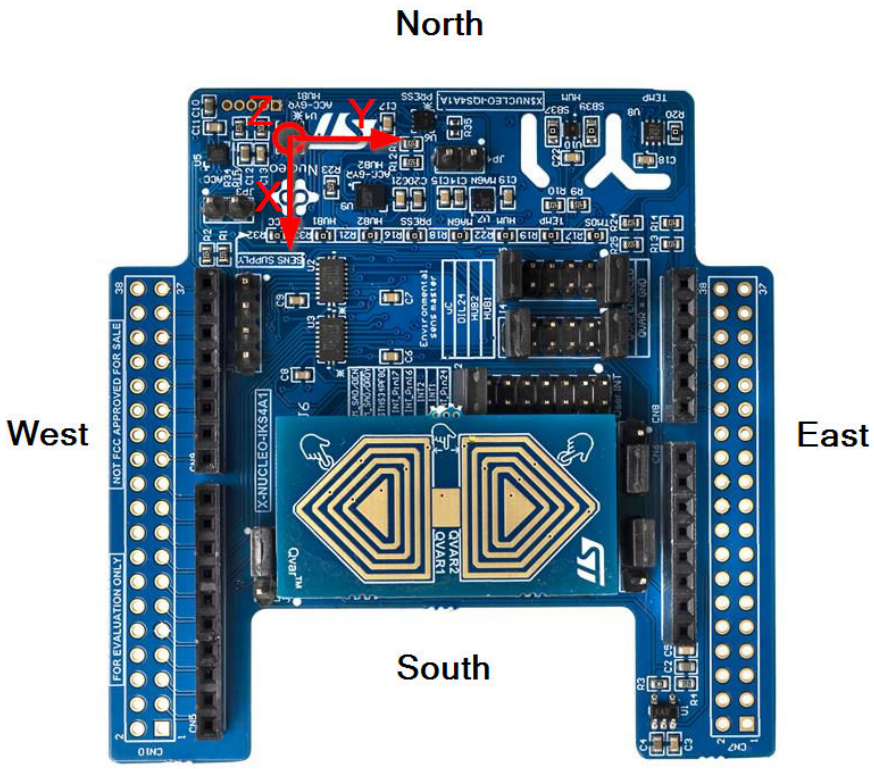
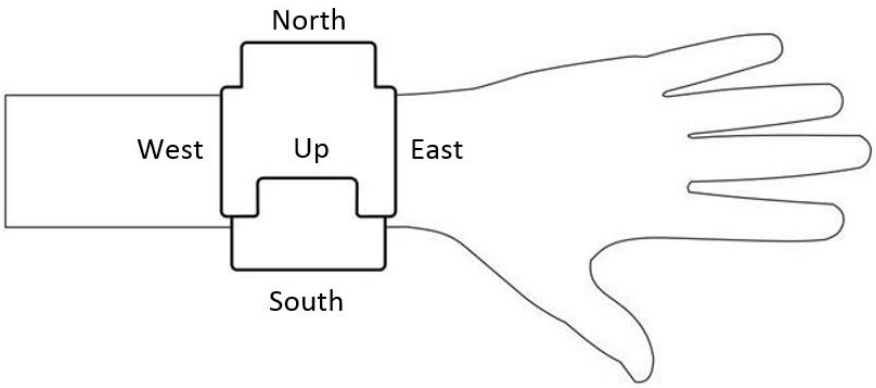
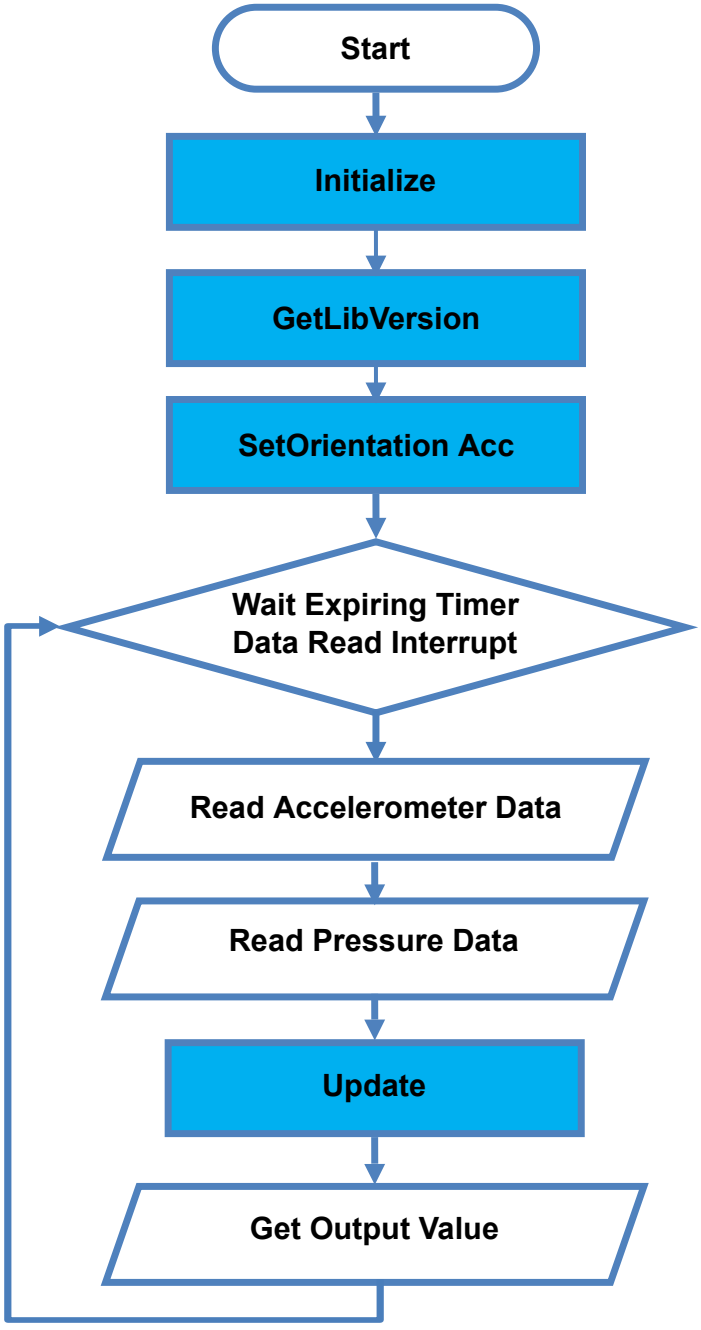


Figure 2. Orientation system for wrist-worn devices



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 and pressure sensors and gets the position code.

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

/*** Initialization ***/
char lib_version[VERSION_STR_LENG];
char acc_orientation[] ="seu";

/* Standing vs Sitting Desk Detection initialization function */
MotionSD_Initialize();

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

/* Set accelerometer orientation */
MotionSD_SetOrientation_Acc(acc_orientation);

[…]

/*** Using Standing vs Sitting Desk Detection algorithm ***/
Timer_OR_DataRate_Interrupt_Handler()
{
  MSD_input_t data_in;
  MSD_output_t data_out;

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

  /* Get atmospheric pressure in hPa */
  MEMS_Read_PressValue(&data_in.Press);

  /* Standing vs Sitting Desk Detection update */
  MotionSD_Update(&data_in, &data_out);
}
```

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. 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
32	45	50	165	270	322

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

Cortex-M33 STM32U575ZI-Q at 160 MHz			Cortex-M7 STM32F767ZI at 96 MHz		
Min	Avg	Max	Min	Avg	Max
14	20	22	63	82	86

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-L152RE or NUCLEO-U575ZI-Q development board connected to an X-NUCLEO-IKS01A3 or X-NUCLEO-IKS4A1 expansion board.

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.

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

2.4 MEMS-Studio application

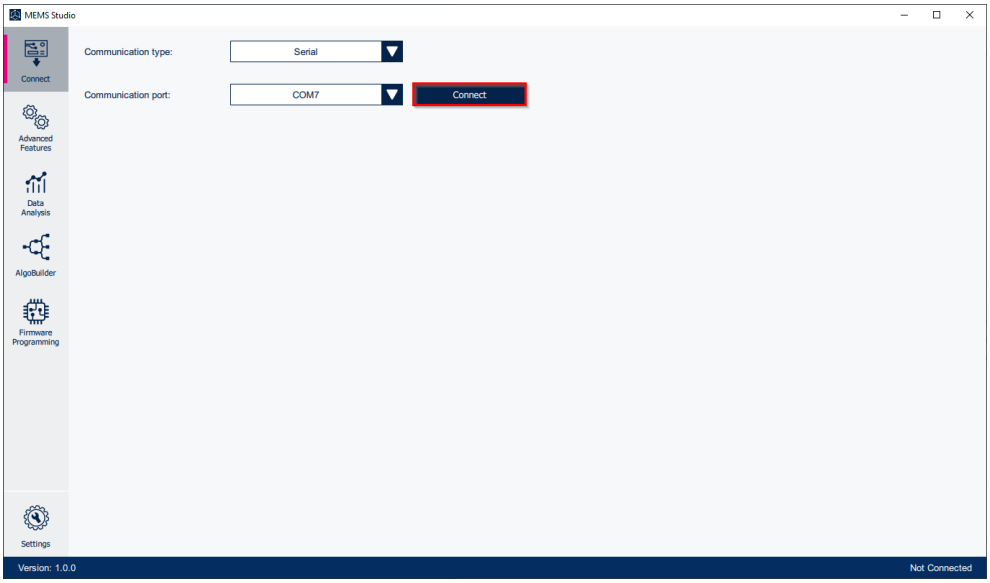
The sample application uses the MEMS-Studio GUI application, which can be downloaded from www.st.com.

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, the appropriate COM port is authomatically detected. Press [**Connect**] button to open this port.

Figure 4. MEMS-studio connect



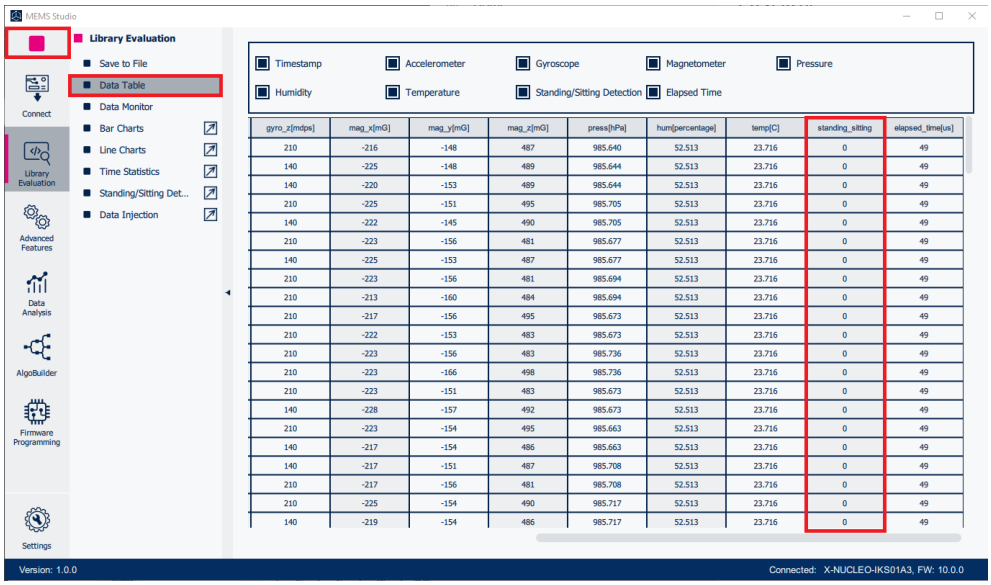
Step 3. When connected to STM32 Nucleo board with supported firmware [**Library Evaluation**] tab is opened. To start and stop data streaming toggle the appropriate start / stop button on the outer vertical tool bar.

Figure 5. Start

Figure 6. Stop

The data coming from the connected sensor can be viewed selecting the [**Data Table**] tab on the inner vertical tool bar.

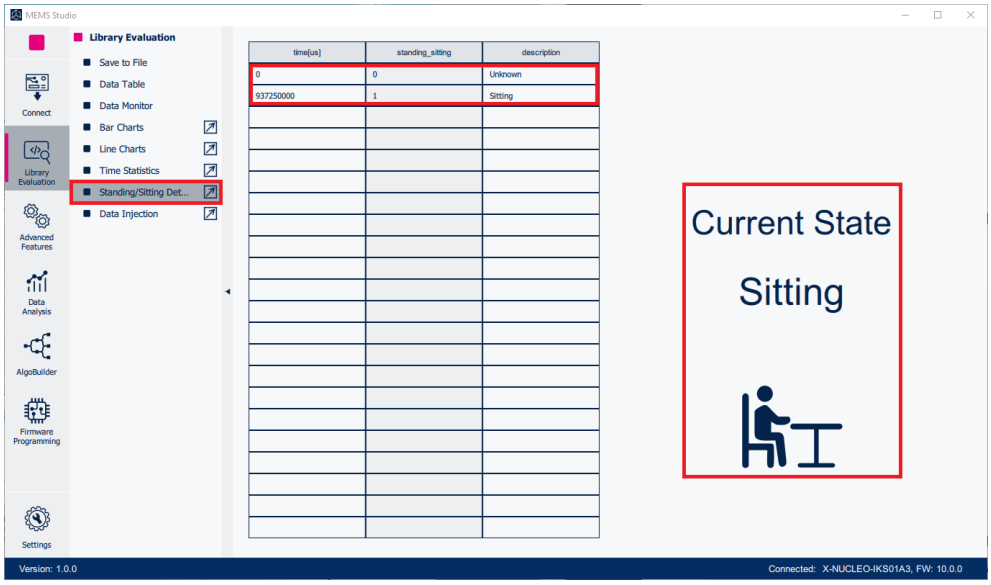
Figure 7. MEMS-Studio - Library evaluation - Data table



gyro_z[mdps]	mag_x[mG]	mag_y[mG]	mag_z[mG]	press[hPa]	hum[percentage]	temp[C]	standing_sitting	elapsed_time[us]
210	-216	-148	487	985.640	52.513	23.716	0	49
140	-225	-148	489	985.644	52.513	23.716	0	49
140	-220	-153	489	985.644	52.513	23.716	0	49
210	-225	-151	495	985.705	52.513	23.716	0	49
140	-222	-145	490	985.705	52.513	23.716	0	49
210	-223	-156	481	985.677	52.513	23.716	0	49
140	-225	-153	487	985.677	52.513	23.716	0	49
210	-223	-156	481	985.694	52.513	23.716	0	49
210	-213	-160	484	985.694	52.513	23.716	0	49
210	-217	-156	495	985.673	52.513	23.716	0	49
210	-222	-153	483	985.673	52.513	23.716	0	49
210	-223	-156	483	985.736	52.513	23.716	0	49
210	-223	-166	498	985.736	52.513	23.716	0	49
210	-223	-151	483	985.673	52.513	23.716	0	49
140	-228	-157	492	985.673	52.513	23.716	0	49
210	-223	-154	495	985.663	52.513	23.716	0	49
140	-217	-154	486	985.663	52.513	23.716	0	49
140	-217	-151	487	985.708	52.513	23.716	0	49
210	-217	-156	481	985.708	52.513	23.716	0	49
210	-225	-154	490	985.717	52.513	23.716	0	49
140	-219	-154	486	985.717	52.513	23.716	0	49

Step 4. Select the [**Standing/Sitting Detection**] tab on the inner vertical tool bar to open the dedicated application view. There you can see visual representation of the current state and log with previous changes.

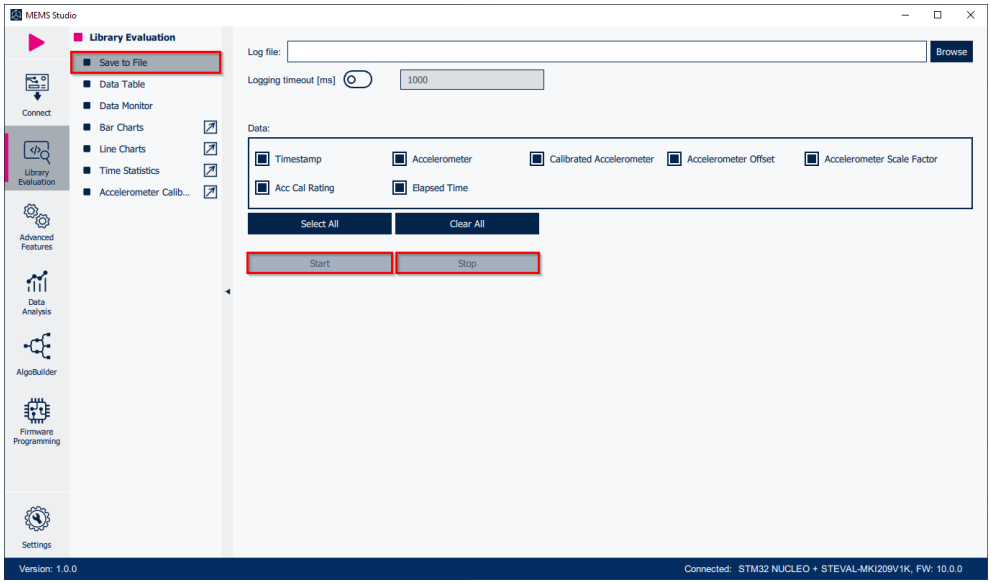
Figure 8. MEMS-Studio - Library Evaluation - Standing/Sitting detection



Step 5. Select the [**Save to File**] tab on the inner vertical tool bar to open the data logging configuration window.

You can select which sensor and activity data to save to log files. You can start or stop logging by clicking on the corresponding [**Start/Stop**] button.

Figure 9. 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
22-Sep-2017	1	Initial release.
25-Jan-2018	2	Added references to NUCLEO-L152RE development board. Added Figure 2. Orientation system for wrist-worn devices and Table 2. Elapsed time (µs) algorithm.
21-Mar-2018	3	Updated Introduction and Section 2.1 MotionSD overview.
18-Feb-2019	4	Updated Table 2. Elapsed time (µs) algorithm. Added X-NUCLEO-IKS01A3 expansion board compatibility information.
24-May-2024	5	Updated Introduction, Section 2.1: MotionSD overview, Section 2.2.1: MotionSD library description, Section 2.2.2: MotionSD APIs, Section 2.2.4: Demo code, Section 2.2.5: Algorithm performance, Section 2.3: Sample application and Section 3: References. Replaced Unicleo-GUI application with Section 2.4: MEMS-Studio application.

Contents

List of tables

List of figures