

---

# Getting started with MotionAT active time library in X-CUBE-MEMS1 expansion for STM32Cube

## Introduction

The MotionAT middleware library is part of the [X-CUBE-MEMS1](#) software and runs on STM32. It combines results from activity recognition for wrist, motion intensity detection and pedometer for wrist algorithms. It also provides real-time information about the number of active seconds (that is how long the user was active with the wearable device as a smart watch).

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<sup>®</sup> Cortex<sup>®</sup>-M3, ARM<sup>®</sup> Cortex<sup>®</sup>-M33, ARM<sup>®</sup> Cortex<sup>®</sup>-M4 or ARM<sup>®</sup> Cortex<sup>®</sup>-M7 architecture.

It is built on top of [STM32Cube](#) software technology to ease portability across different STM32 microcontrollers.

The software comes with sample implementation running on an [X-NUCLEO-IKS01A3](#) or [X-NUCLEO-IKS4A1](#) expansion board on a [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 MotionAT middleware library in X-CUBE-MEMS1 software expansion for STM32Cube

### 2.1 MotionAT overview

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

The library acquires data from the accelerometer and provides information about the number of active seconds (how long the user was active) with the wearable 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.

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 MotionAT library

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

#### 2.2.1 MotionAT library description

The MotionAT active time library manages the data acquired from the accelerometer; it features:

- Possibility of detecting the number of active seconds
- Recognition based on accelerometer data only
- Required accelerometer data sampling frequency of 50 Hz
- Measurement based on the accelerometer data only
- 16.5 Kbyte of code memory and 5.6 Kbyte of data memory usage
- Resources requirements:
  - Cortex®-M3: 21.8 KB of code and 5.5 KB of data memory
  - Cortex®-M33: 20.9 KB of code and 5.5 KB of data memory
  - Cortex®-M4: 23.0 KB of code and 5.5 KB of data memory
  - Cortex®-M7: 22.9 KB of code and 5.5 KB of data memory
- Available for ARM® Cortex®-M3, ARM® Cortex®-M33, ARM® Cortex®-M4, and ARM® Cortex®-M7 architectures

#### 2.2.2 MotionAT APIs

The MotionAT library APIs are:

- `uint8_t MotionAT_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 MotionAT_Initialize(void)`
  - performs MotionAT library initialization and setup of the internal mechanism including the dynamic memory allocation
  - the CRC module in STM32 microcontroller (in RCC peripheral clock enable register) has to be enabled before using the library

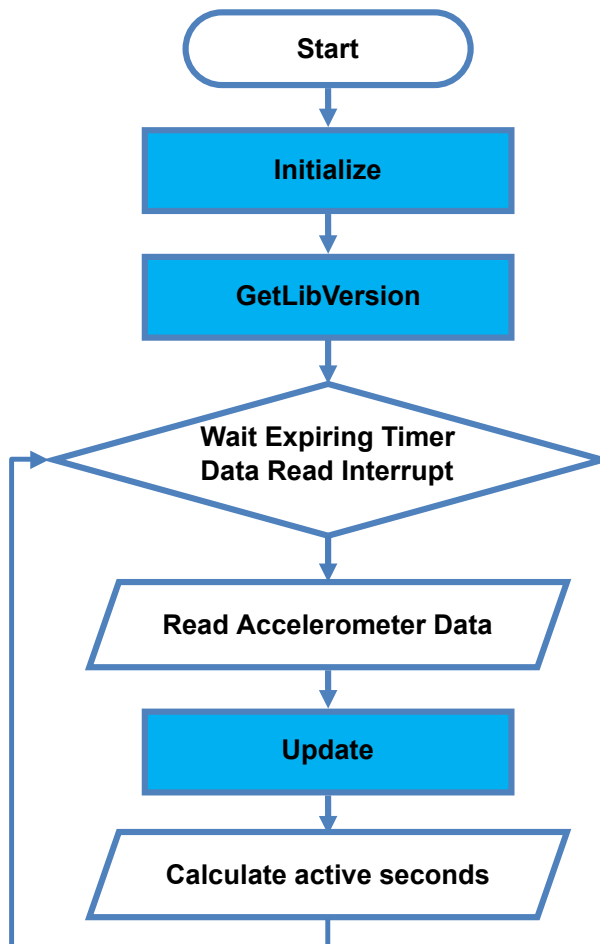
*Note:* This function must be called before using the active time library.

- `void MotionAT_Deinitialize(void)`
  - performs MotionAT library deinitialization including the dynamic memory deallocation

- `void MotionAT_Update(MAT_input_t *data_in, MAT_output_t *data_out)`
  - executes active time algorithm
  - `*data_in` parameter is a pointer to a structure with input data
  - the parameters for the structure type `MAT_input_t` are:
    - `acc_x` is the accelerometer sensor value in X axis in g
    - `acc_y` is the accelerometer sensor value in Y axis in g
    - `acc_z` is the accelerometer sensor value in Z axis in g
  - `*data_out` parameter is a pointer to a structure with output data
  - the parameter for the structure type `MAT_output_t` is:
    - `active` equal to 1 if the user was active during the last algorithm run, otherwise it is equal to 0

### 2.2.3 API flow chart

Figure 1. MotionAT API logic sequence



### 2.2.4 Demo code

```
[...]
#define VERSION_STR LENG      35
[...]
```

```
/** Initialization **/
char lib_version[VERSION_STR LENG];

/* Active Time API initialization function */
MotionAT_Initialize();

/* OPTIONAL */
/* Get library version */
MotionAT_GetLibVersion(lib_version);

[...]
```

```
/** Using Active Time algorithm **/
Timer_OR_DataRate_Interrupt_Handler()
{
    MAT_input_t MAT_data_in;
    MAT_output_t MAT_data_out;

    /* Get acceleration X/Y/Z in g */
    MEMS_Read_AccValue(&MAT_data_in.acc_x, &MAT_data_in.acc_y, &MAT_data_in.acc_z);

    /* Run Active Time algorithm */
    MotionAT_Update(&MAT_data_in, &MAT_data_out);

    /* Count active seconds */
    /* 1/50Hz = 0.02s per 1 algorithm run */
    active_seconds += MAT_data_out.active * 0.02f;
}

[...]
```

### 2.2.5 Algorithm performance

The active time algorithm only uses data from the accelerometer and runs at a low frequency (50 Hz) to reduce power consumption.

It detects and provides real-time information about the number of active seconds (how long the user was active with his wearable device).

**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
38	55	619	302	520	6912

**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
58	67	361	289	323	1622

### 2.3 Sample application

The MotionAT 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 application recognizes the active seconds in real-time. The data can be displayed through a GUI.

The USB cable connection is required to monitor real-time data. The board is powered by the PC via USB connection.

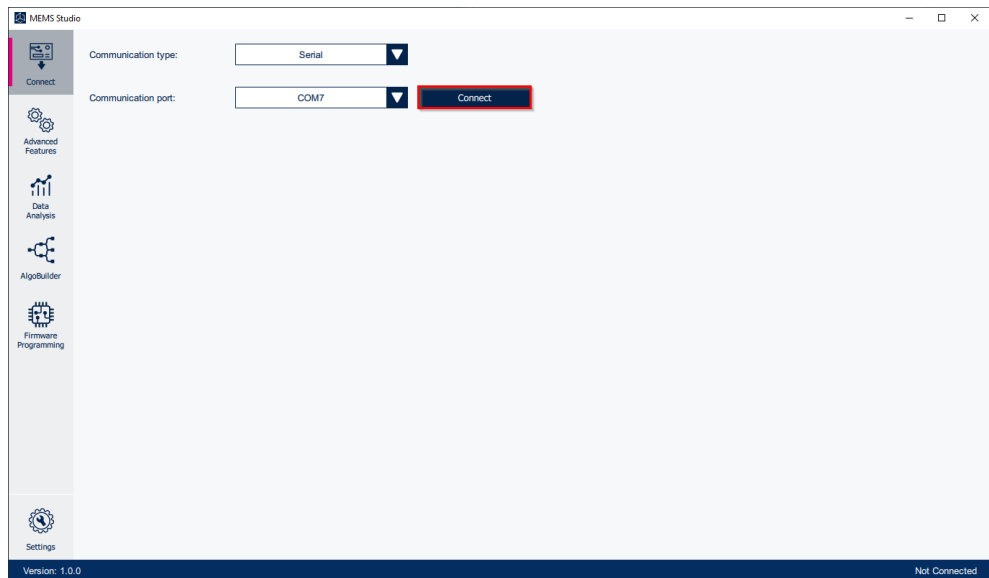
This allows the user to display whether the user was active during last single algorithm run, the amount of active seconds, accelerometer data, time stamp and eventually other sensor data, in real-time, using the MEMS-Studio GUI application.

## 2.4 MEMS-Studio application

The sample application uses the **MEMS-Studio** GUI application, which can be downloaded from [www.st.com](http://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 automatically detected. 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.

Figure 3. Start



Figure 4. Stop



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

Figure 5. MEMS-Studio - Library evaluation - Data table

The screenshot shows the MEMS Studio interface. On the left, the 'Library Evaluation' tab is active, and the 'Data Table' sub-tab is selected. The main window displays a table of sensor data. The table has the following columns: time[s], acc\_x[mg], acc\_y[mg], cal\_acc\_x[mg], cal\_acc\_y[mg], acc\_offset\_x[mg], acc\_offset\_y[mg], acc\_scale\_x, and acc\_scale\_y. The data shows alternating rows of 1.000 and 2.000 for acc\_x, and -299.000 and -298.000 for acc\_y, with other values remaining constant at 1.000 or 0.000.

time[s]	acc_x[mg]	acc_y[mg]	cal_acc_x[mg]	cal_acc_y[mg]	acc_offset_x[mg]	acc_offset_y[mg]	acc_scale_x	acc_scale_y
15610000	1.000	-299.000	1.000	-299.000	0.000	0.000	1.000	1.000
15620000	2.000	-300.000	2.000	-300.000	0.000	0.000	1.000	1.000
15630000	2.000	-299.000	2.000	-299.000	0.000	0.000	1.000	1.000
15640000	2.000	-299.000	2.000	-299.000	0.000	0.000	1.000	1.000
15650000	2.000	-298.000	2.000	-298.000	0.000	0.000	1.000	1.000
15660000	2.000	-299.000	2.000	-299.000	0.000	0.000	1.000	1.000
15670000	1.000	-300.000	1.000	-300.000	0.000	0.000	1.000	1.000
15680000	1.000	-299.000	1.000	-299.000	0.000	0.000	1.000	1.000
15690000	1.000	-299.000	1.000	-299.000	0.000	0.000	1.000	1.000
15700000	2.000	-299.000	2.000	-299.000	0.000	0.000	1.000	1.000
15710000	2.000	-299.000	2.000	-299.000	0.000	0.000	1.000	1.000
15720000	2.000	-299.000	2.000	-299.000	0.000	0.000	1.000	1.000
15730000	2.000	-299.000	2.000	-299.000	0.000	0.000	1.000	1.000
15740000	1.000	-300.000	1.000	-300.000	0.000	0.000	1.000	1.000
15740000	1.000	-300.000	1.000	-300.000	0.000	0.000	1.000	1.000
15750000	1.000	-300.000	1.000	-300.000	0.000	0.000	1.000	1.000
15760000	1.000	-300.000	1.000	-300.000	0.000	0.000	1.000	1.000
15770000	1.000	-299.000	1.000	-299.000	0.000	0.000	1.000	1.000
15780000	2.000	-298.000	2.000	-298.000	0.000	0.000	1.000	1.000
15790000	2.000	-299.000	2.000	-299.000	0.000	0.000	1.000	1.000

**Step 4.** Select the **[Active Time]** tab on the inner vertical tool bar to open the dedicated application status view.

**Figure 6. MEMS-Studio - Library Evaluation - Active Time**

The screenshot displays the MEMS-Studio interface with the 'Library Evaluation' section active. The 'Active Time' tab is selected in the inner vertical tool bar. The main area shows a table of activity data and a 'Current State' indicator.

time[us]	active_state	active_time[s]
0	Inactive	0
11700000	Active	0
12600000	Active	1
13500000	Active	2
14400000	Active	3
15310000	Active	4
16210000	Active	5
17110000	Active	6
18010000	Inactive	6
178560000	Active	7
179460000	Inactive	7
180370000	Active	8
181250000	Active	9
182150000	Active	10
183050000	Active	11
183950000	Active	12
184850000	Active	13
185760000	Active	14

To the right of the table, a 'Current State' indicator shows 'Active' with a walking stick icon.

**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 saving by clicking on the corresponding **[Start/Stop]** button.

**Figure 7. MEMS-Studio - Library Evaluation - Save to File**

The screenshot displays the MEMS-Studio interface with the 'Library Evaluation' section active. The 'Save to File' tab is selected in the inner vertical tool bar. The main area shows the data logging configuration window.

The configuration window includes a 'Log file' field with a 'Browse' button, a 'Logging timeout (ms)' field set to 1000, and a 'Data' section with checkboxes for:

- Timestamp
- Accelerometer
- Calibrated Accelerometer
- Accelerometer Offset
- Accelerometer Scale Factor
- Acc: Cal Rating
- Elapsed Time

Below the checkboxes are 'Select All' and 'Clear All' buttons. At the bottom, there are 'Start' and 'Stop' buttons.



### 3 References

---

All of the following resources are freely available on [www.st.com](http://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
31-Jan-2018	1	Initial release.
21-Mar-2018	2	Updated Introduction and Section 2.1 MotionAT overview.
14-Feb-2019	3	Updated Table 2. Elapsed time ( $\mu$ s) algorithm and Figure 2. STM32 Nucleo: LEDs, button, jumper. Added X-NUCLEO-IKS01A3 expansion board compatibility information.
24-May-2024	4	Updated Introduction, Section 2.1: MotionAT overview, Section 2.2.1: MotionAT library description, Section 2.2.2: MotionAT APIs, Section 2.2.4: Demo code, Section 2.2.5: Algorithm performance, Section 2.3: Sample application. Replaced Unicleo-GUI application with Section 2.4: MEMS-Studio application.

## Contents

<b>1</b>	<b>Acronyms and abbreviations</b>	<b>2</b>
<b>2</b>	<b>MotionAT middleware library in X-CUBE-MEMS1 software expansion for STM32Cube</b>	<b>3</b>
<b>2.1</b>	MotionAT overview	3
<b>2.2</b>	MotionAT library	3
<b>2.2.1</b>	MotionAT library description	3
<b>2.2.2</b>	MotionAT APIs	3
<b>2.2.3</b>	API flow chart	4
<b>2.2.4</b>	Demo code	5
<b>2.2.5</b>	Algorithm performance	5
<b>2.3</b>	Sample application	5
<b>2.4</b>	MEMS-Studio application	6
<b>3</b>	<b>References</b>	<b>9</b>
	<b>Revision history</b>	<b>10</b>

## List of tables

<b>Table 1.</b>	List of acronyms . . . . .	2
<b>Table 2.</b>	Cortex-M4 and Cortex-M3: elapsed time ( $\mu$ s) algorithm. . . . .	5
<b>Table 3.</b>	Cortex-M33 and Cortex-M7: elapsed time ( $\mu$ s) algorithm. . . . .	5
<b>Table 4.</b>	Document revision history . . . . .	10

## List of figures

Figure 1.	MotionAT API logic sequence . . . . .	4
Figure 2.	MEMS-studio connect . . . . .	6
Figure 3.	Start . . . . .	7
Figure 4.	Stop . . . . .	7
Figure 5.	MEMS-Studio - Library evaluation - Data table . . . . .	7
Figure 6.	MEMS-Studio - Library Evaluation - Active Time . . . . .	8
Figure 7.	MEMS-Studio - Library Evaluation - Save to File . . . . .	8

**IMPORTANT NOTICE – READ CAREFULLY**

STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST’s terms and conditions of sale in place at the time of order acknowledgment.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of purchasers’ products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

ST and the ST logo are trademarks of ST. For additional information about ST trademarks, refer to [www.st.com/trademarks](http://www.st.com/trademarks). All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2024 STMicroelectronics – All rights reserved