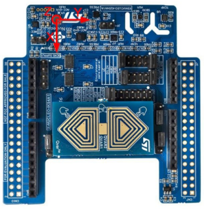
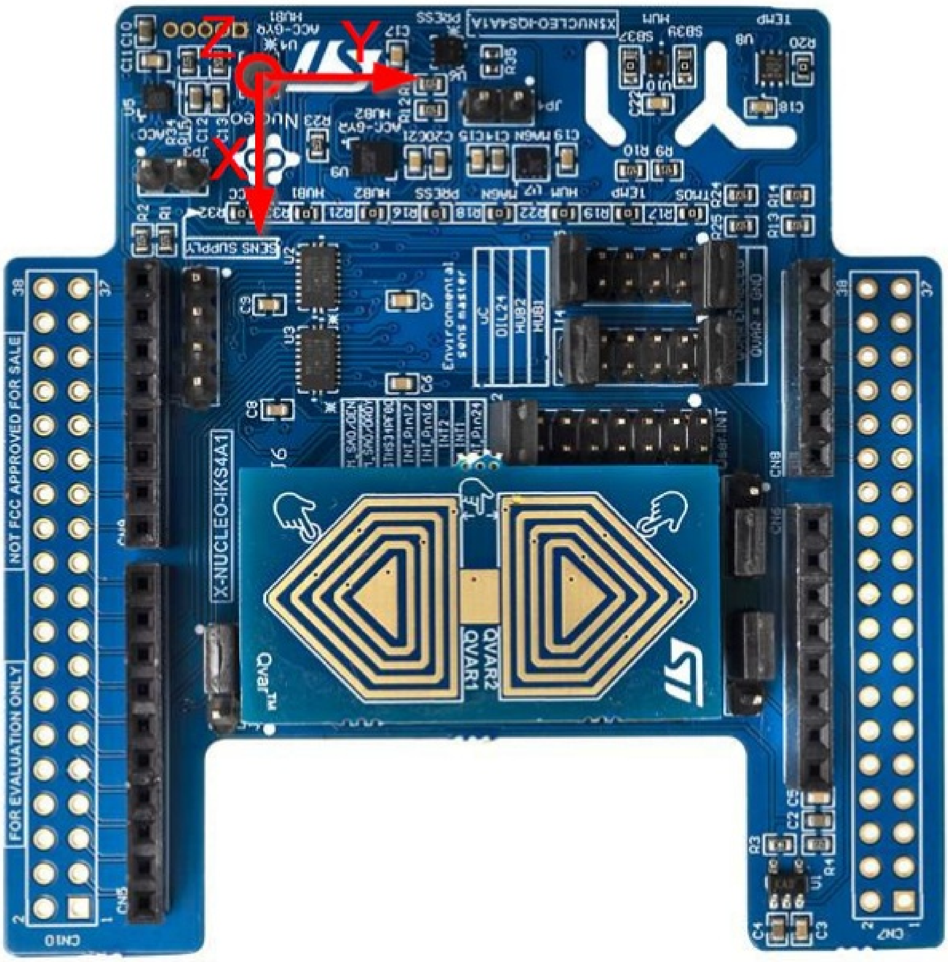


STMicroelectronics UM2193 MotionAR Activity Recognition Library User Manual

Home » STMicroelectronics » STMicroelectronics UM2193 MotionAR Activity Recognition Library User Manual

STMicroelectronics UM2193 MotionAR Activity Recognition Library



Contents

Introduction

The Motion AR is a middleware library part of X-CUBE-MEMS1 software and runs on STM32. It provides real-time information on the type of activity performed by the user. It is able to distinguish the following activities: stationary, walking, fast walking, jogging, biking, driving.

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 that eases 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.

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

Motion AR middleware library in X-CUBE-MEMS1 software expansion

Motion AR overview

The Motion AR library expands the functionality of the X-CUBE-MEMS1 software.
The library acquires data from the accelerometer and provides information on the type of activity performed by the user.

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 on X-NUCLEO-IKS01A3 or X-NUCLEO-IKS4A1 expansion boards, mounted on a NUCLEO-F401RE, NUCLEO-L152RE or NUCLEO-U575ZI-Q development board.

Motion AR library

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

Motion AR library description

- The Motion AR activity recognition library manages data acquired from accelerometer; it features:
- possibility to distinguish the following activities: stationary, walking, fast walking, jogging, biking, driving
- recognition based on accelerometer data only
- required accelerometer data sampling frequency: 16 Hz
- resources requirements:
  - Cortex-M3: 8.5 kB of code and 1.4 kB of data memory
  - Cortex-M33: 7.8 kB of code and 1.4 kB of data memory
  - Cortex-M4: 7.9 kB of code and 1.4 kB of data memory
  - Cortex-M7: 8.1 kB of code and 1.4 kB of data memory
- available for ARM Cortex-M3, Cortex-M33, Cortex-M4 and Cortex-M7 architectures

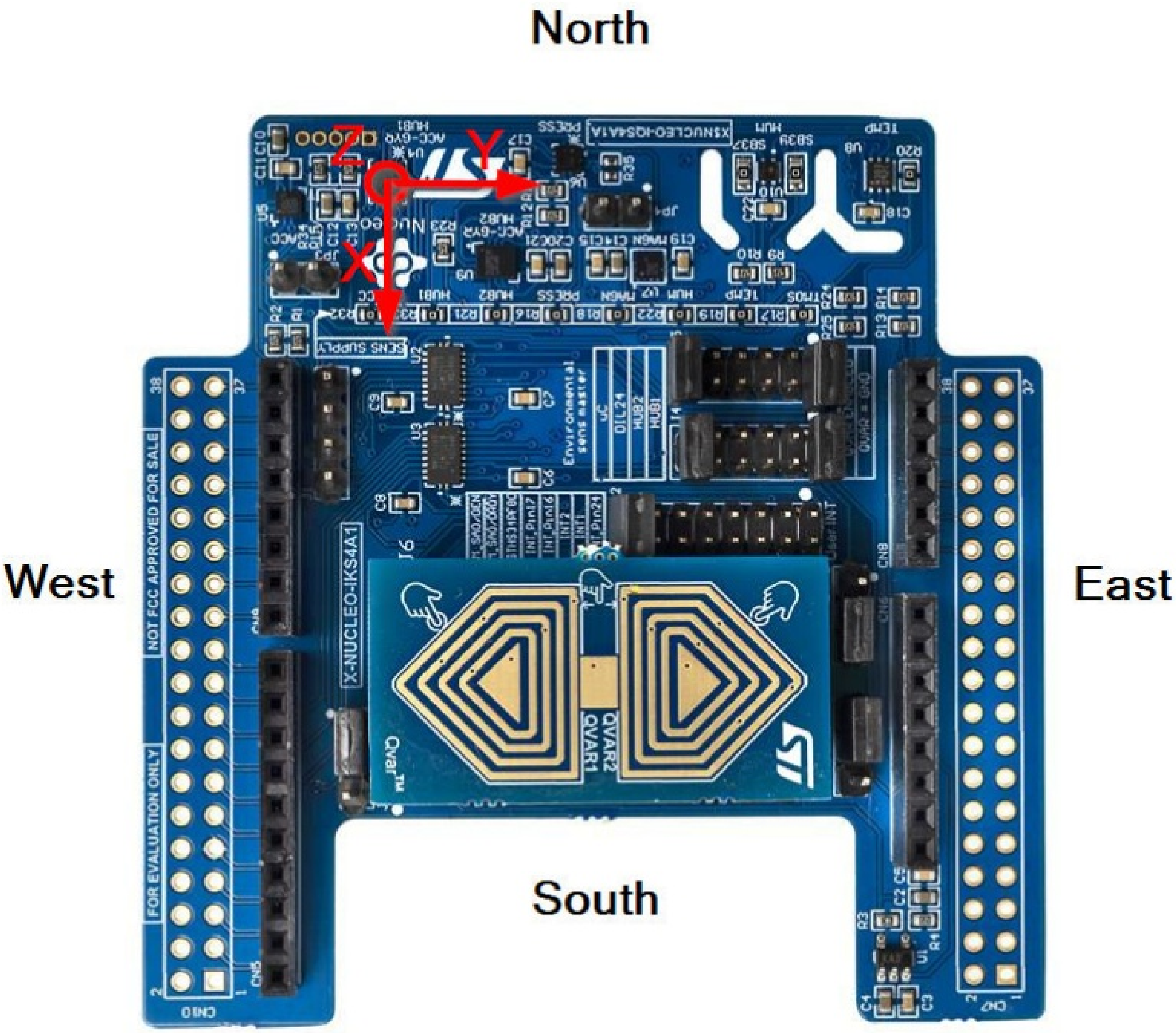
MotionAR APIs

The MotionAR APIs are:

- uint8_t MotionAR_GetLibVersion(char *version)
  - retrieves the version of the library
  - *version is a pointer to an array of 35 characters
  - returns the number of characters in the version string
- void MotionAR_Initialize(void)
  - performs MotionAR 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

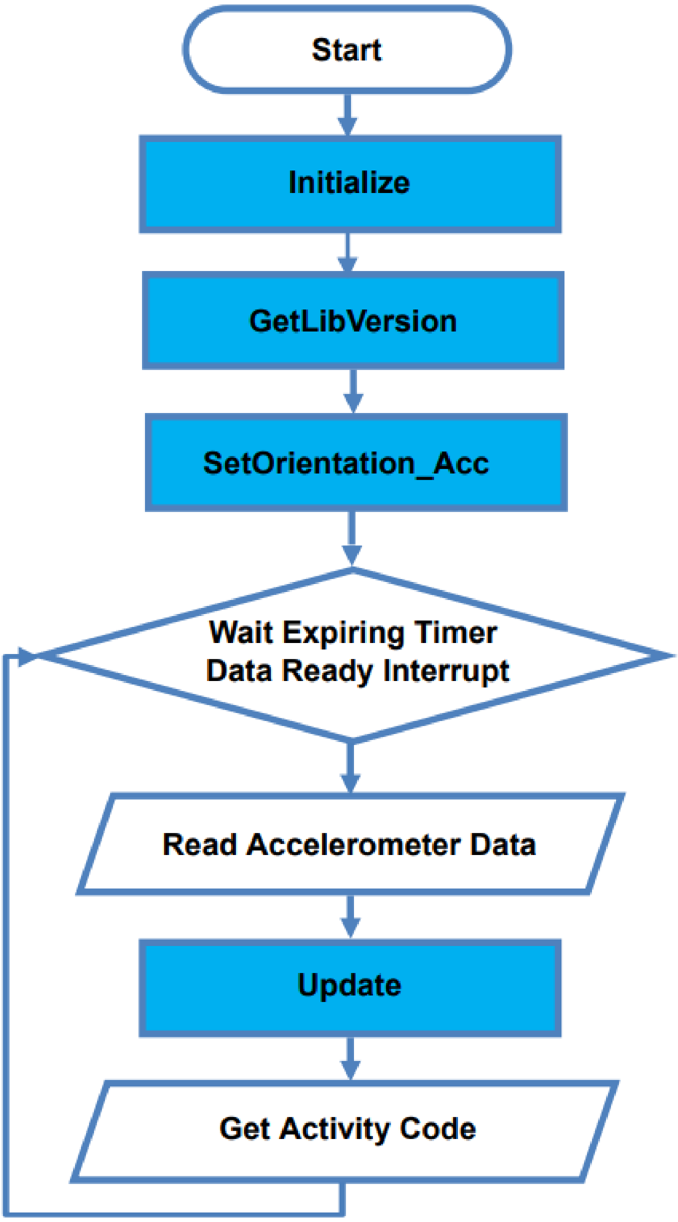
    Note: This function must be called before using the accelerometer calibration library.
- void MotionAR_Reset(void)
  - resets activity recognition algorithms
- void MotionAR_Update(MAR_input_t *data_in, MAR_output_t *data_out, int64_t timestamp)
  - executes activity recognition algorithm
  - *data_in parameter is a pointer to a structure with input data
  - the parameters for the structure type MAR_input_t are:
    - acc_x is accelerometer sensor value in X axis in g
    - acc_y is accelerometer sensor value in Y axis in g
    - acc_z is accelerometer sensor value in Z axis in g
  - *data_out parameter is a pointer to enum with the following items:
    - MAR_NOACTIVITY = 0
    - MAR_STATIONARY = 1
    - MAR_WALKING = 2
    - MAR_FASTWALKING = 3
    - MAR_JOGGING = 4
    - MAR_BIKING = 5
    - MAR_DRIVING = 6
  - timestamp is a relative time for actual sample in ms
- void MotionAR_ Set Orientation_ Acc(const char *acc_ orientation)
  - sets the accelerometer data orientation
  - configuration is usually performed immediately after the Motion AR_ 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 (x-South, y-East, z-Up), so the string is: “seu”.

Figure 1. Sensor orientation example



API flow char

Figure 2. Motion AR API logic sequence



Demo code

The following demonstration code reads data from accelerometer sensor and gets the activity code

```
[…]
#define VERSION_STR_LENG 35
[…]
/*** Initialization ***/
char lib_version[VERSION_STR_LENG];
char acc_orientation[] = "seu";
/* Activity recognition API initialization function */
MotionAR_Initialize();
/* Optional: Get version */
MotionAR_GetLibVersion(lib_version);
/* Set accelerometer orientation */
MotionAR_SetOrientation_Acc(acc_orientation);
[…]
/*** Using activity recognition algorithm ***/
Timer_ OR_ Data Rate_ Interrupt_ Handler()
{

MAR_input_t data_ in;
MAR_ output_ t activity;
/* Get acceleration X/Y/Z in g */
MEMS_Read_AccValue(&data_in.acc_x, &data_in.acc_y, &data_in.acc_z);
/* Get current time in ms */
TIMER_Get_TimeValue(&timestamp_ms);
/* Activity recognition algorithm update */
MotionAR_Update(data_in, data_out, timestamp_ms);
}
```

Algorithm performance

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

Table 2. Algorithm performance

Activity	Detection probability (typical)(1)	Best performance	Susceptible	Carry positions
Stationary	92.27%		Holding in hand and heavy texting	All: trouser pocket, shirt pocket, back pocket, near the head, etc.
Walking	99.44%	Step rate ≥ 1.4 step/s	Step rate ≤ 1.2 step/s	all
Fast walking	95.94%	Step rate ≥ 2.0 step/s		All
Jogging	98.49%	Step rate ≥ 2.2 step/s	Duration < 1 minute; speed < 8 Km/h	Trouser pocket, arm swing, in- hand
Biking	91.93%	Outdoor speed ≥11 Km/h	Passenger seat, glove compartment	Backpack, shirt pocket, trouser pocket
Driving	78.65%	Speed ≥ 48 Km/h	Passenger seat, glove compartment	Cup holder, dash board, shirt pocket, trouser pocket

1. Typical specifications are not guaranteed

Table 3. 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
2	6	153	8	130	4883

Table 4. 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
< 1	2	74	5	9	145

Sample application

The MotionAR 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 performed activities in real-time. Data can be displayed through a GUI. The algorithm recognizes stationary, walking, fast walking, jogging, bike riding and driving activities. 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 the activity detected, accelerometer data, time stamp and eventually other sensor data, in real-time, using the MEMS-Studio GUI application.

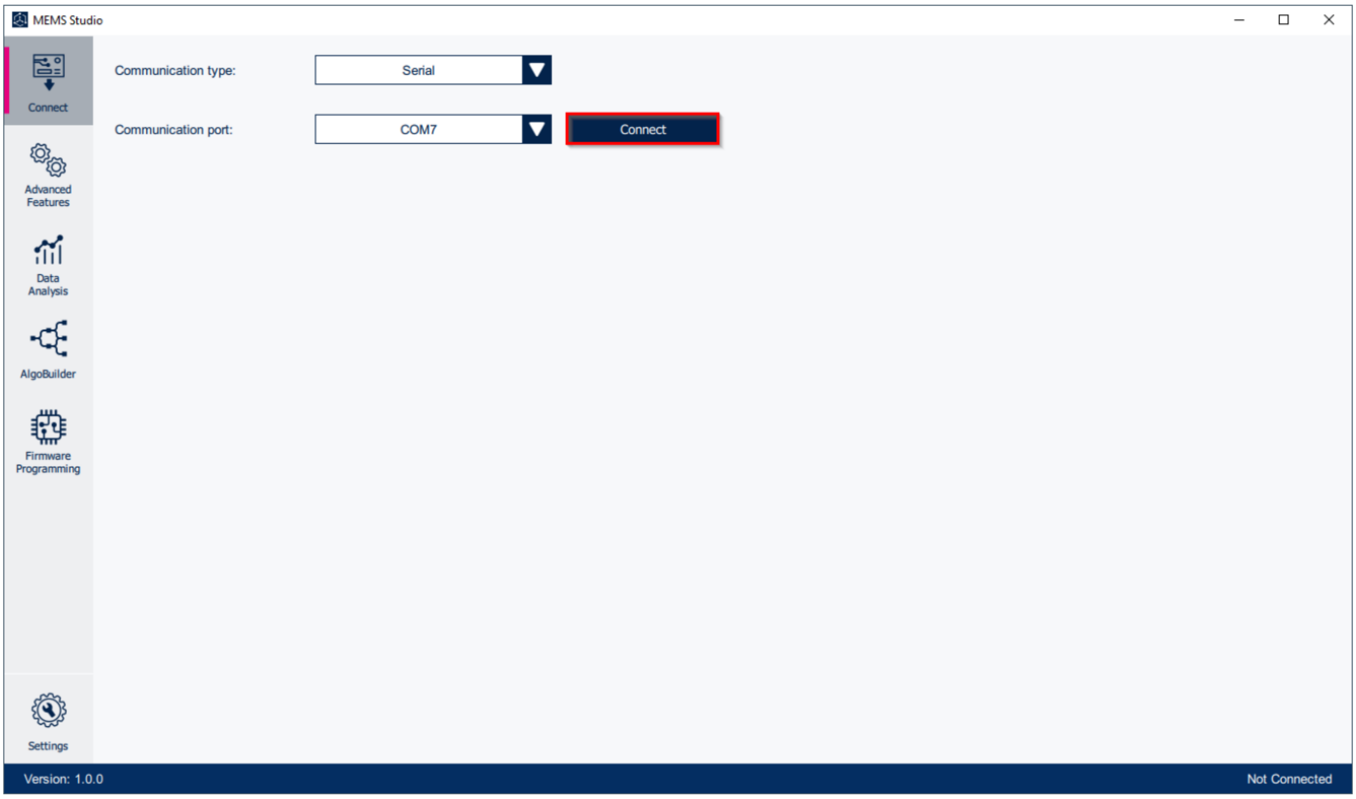
MEMS-Studio application

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, it is automatically detected the appropriate COM port. Press Connect button to open this port.

Figure 3. MEMS-Studio – Connect



Communication type: Serial

Communication port: COM7

Version: 1.0.0

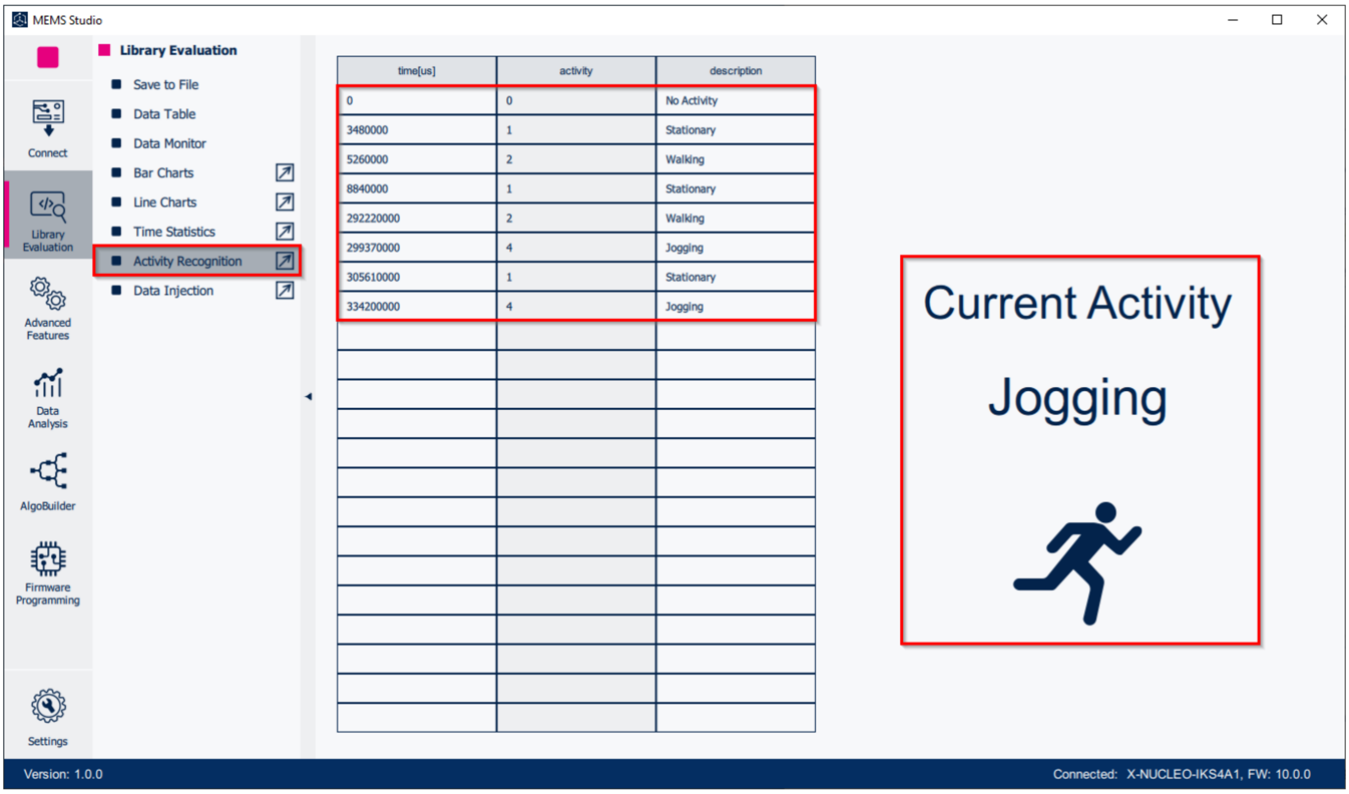
Not Connected

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 4. MEMS-Studio – Library Evaluation – Data Table

Figure 5. MEMS-Studio – Library Evaluation – Activity Recognition



time[us]	activity	description
0	0	No Activity
3480000	1	Stationary
5260000	2	Walking
8840000	1	Stationary
292220000	2	Walking
299370000	4	Jogging
305610000	1	Stationary
334200000	4	Jogging

Current Activity

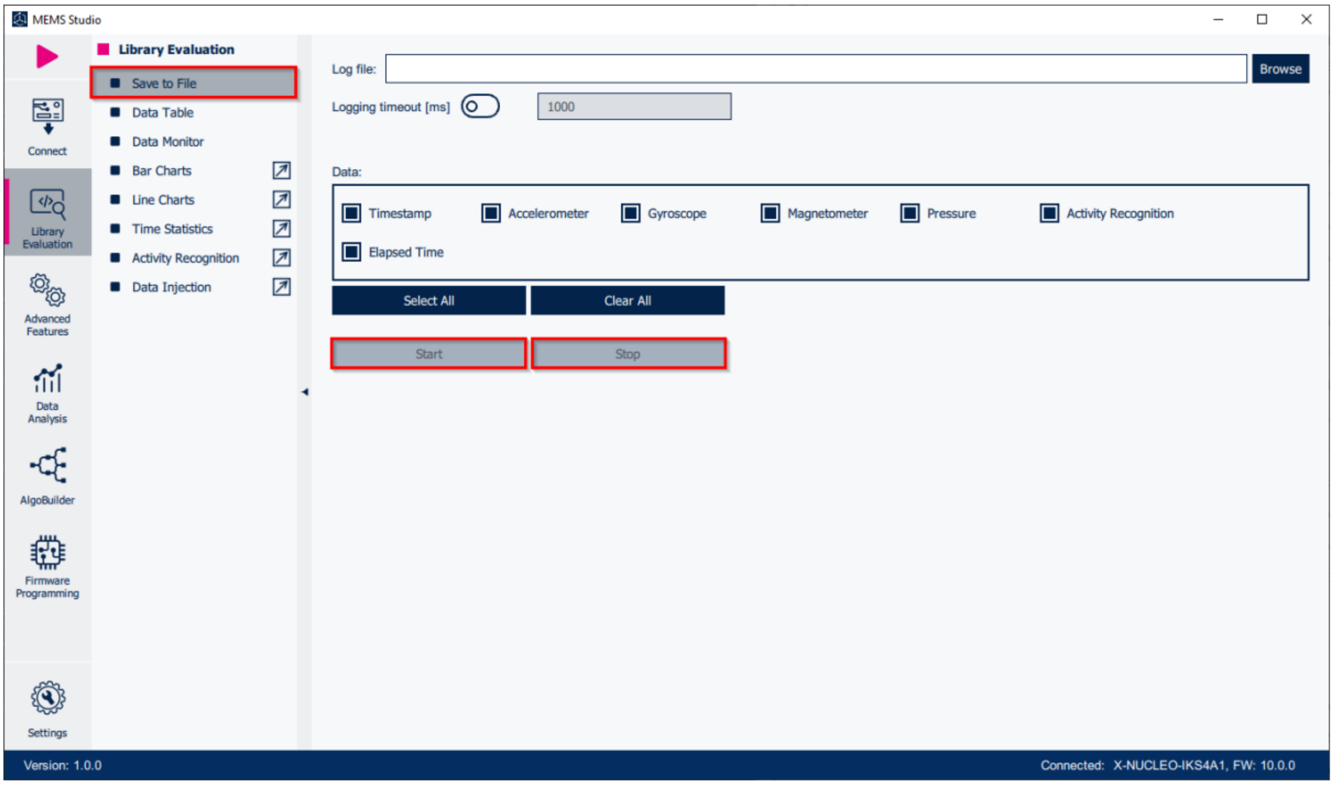
Jogging

Version: 1.0.0

Connected: X-NUCLEO-IKS4A1, FW: 10.0.0

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 6. MEMS-Studio – Library Evaluation – Save to File



Log file:

Logging timeout [ms] ○ 1000

Data:

☒ Timestamp ☒ Accelerometer ☒ Gyroscope ☒ Magnetometer ☒ Pressure ☒ Activity Recognition

☒ Elapsed Time

Version: 1.0.0

Connected: X-NUCLEO-IKS4A1, FW: 10.0.0

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 5. Document revision history

Date	Version	Changes
10-Apr-2017	1	Initial release.
26-Jan-2018	2	Updated Section 3 Sample application. Added references to NUCLEO-L152RE development board and Table 3. Elapsed time (µs) algorithm.
19-Mar-2018	3	Updated Introduction, Section 2.1 Motion AR overview and Section 2.2.5 Algorithm performance.
14-Feb-2019	4	Updated Figure 1. Sensor orientation example, Table 3. Elapsed time (µs) algorithm and Figure 3. STM32 Nucleo: LEDs, button, jumper. Added X-NUCLEO-IKS01A3 expansion board compatibility information.
20-Mar-2019	5	*Updated Section 2.2.2 Motion AR APIs, Figure 3. MEMS-Studio – Connect, Figure 4. MEMS-Studio – Library Evaluation – Data Table, Figure 5. MEMS-Studio – Library Evaluation – Activity Recognition and Figure 6. MEMS-Studio – Library Evaluation – Save to File.*
04-Apr-2024	6	Update **Section Introduction**, **Section 2.1: MotionAR overview**, **Section 2.2.1: MotionAR library description**, MotionAR APIs, **Section 2.2.4: Demo code**, **Section 2.2.5: Algorithm performance**, **Section 3: Sample application** and **Section 4: MEMS-Studio application**.

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**. 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.

Documents / Resources

 UM2193 User manual Getting started with MotionAR activity recognition library in X-CUBE-MEMS1 expansion for STM32Cube Introduction The MotionAR is a middleware library part of X-CUBE-MEMS1 software and runs on STM32. It provides real-time information on the type of activity performed by the user. It is able to distinguish the following activities: stationary, walking, fast walking, jogging, biking, driving. 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 that eases 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. UM2193 - Rev 6 - April 2024 For further information contact your local STMicroelectronics sales office. www.st.com	STMicroelectronics UM2193 MotionAR Activity Recognition Library [pdf] User Manual UM2193 MotionAR Activity Recognition Library, UM2193, MotionAR Activity Recognition Library , Activity Recognition Library, Recognition Library, Library

References

- User Manual