UM1967
User manual

Getting started with osxAcousticBF real-time beam forming software expansion for STM32Cube

Introduction

osxAcousticBF software provides an implementation for a real-time adaptive beamforming algorithm: using the PDM or PCM signals acquired from two digital MEMS microphones, it creates a virtual directional microphone pointing to a fixed direction in space. Several configuration of the algorithm are available, allowing the user to find the best tradeoff between audio output quality and resource consumption. Parameters and modalities can be modified at runtime, to grant adaptation to the varying environment conditions.

The osxAcousticBF library is provided in binary format, integrated in a software package providing implementation examples running on the X-NUCLEO-CCA02M1, when connected to a NUCLEO-F401RE.

The example package is designed as an add-on for X-CUBE-MEMSMIC1 package; the library can be easily ported to any STM32F4 microcontroller with FPU unit. The software is based on STM32Cube technology. Information about STM32Cube is available on www.st.com at *http://www.st.com/stm32cube*.

Contents

List of tables

List of figures

1 Licensing information

This software library is licensed under ST OpenSoftwareX Limited License Agreement, (the "OSX-LLA License"). You may obtain a copy of the OSX License at:

OpenSoftwareX license agreement

Some of the library code is based on the CMSIS DSP software library by ARM, a suite of common signal processing functions for use on Cortex-M processor based devices. Licensing terms are available in the attached release_note.html file, in the next lines of this document and on the web at:

http://www.keil.com/pack/doc/CMSIS/DSP/html/index.html.

Some of the library code is based on the open Source patent-free SPEEX software, by Jean-Marc Valin/Xiph.Org Foundation, released under "revised BSD" license. Licensing terms are available in this document, in the release note for this software, in the header file of this software and on the web at: http://www.xiph.org/licenses/bsd/speex/

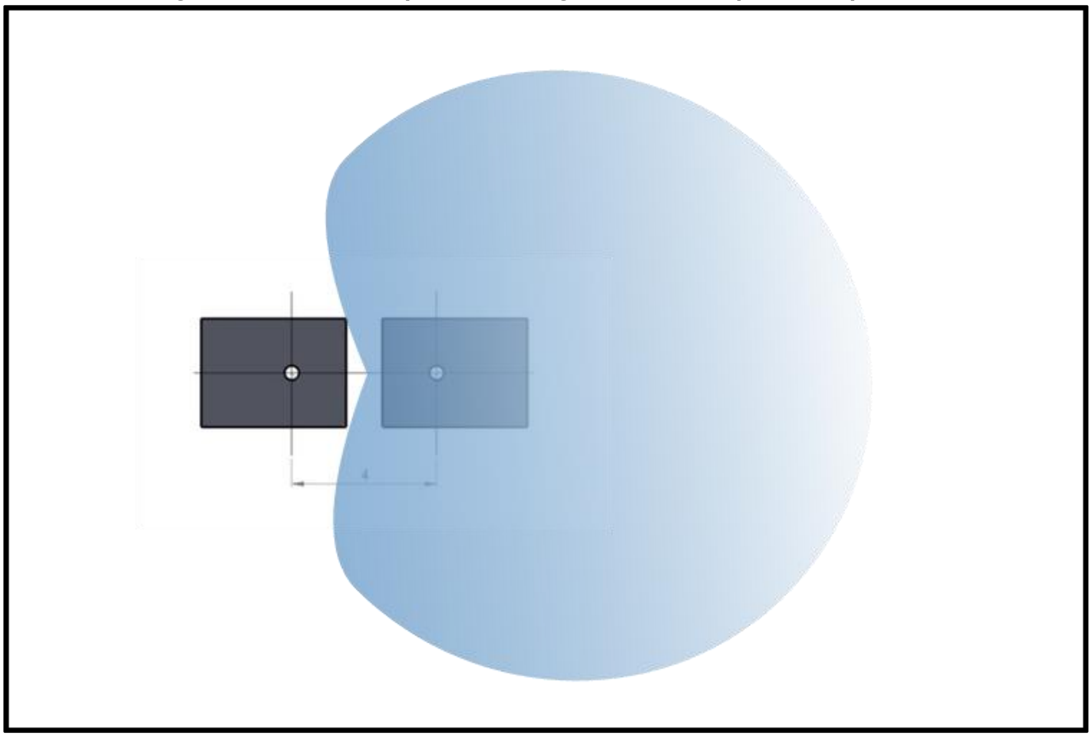
2 osxAcousticBF software library

This library uses the signals from two omnidirectional MEMS digital microphones like ST MP34DT01-M to create a virtual directional microphone. Several algorithm configurations are available for the user to find the best tradeoff between audio output quality and resource consumption. Parameters and modalities can be modified at runtime, to grant adaptation to the varying ambient conditions.

The audio acquisition system must have:

- two microphones placed at a known distance
- audio data acquired in one of the following formats:
  - standard PDM format at a frequency of 1.024 MHz
  - PCM format at 16 kHz (in this case, in the current version, microphone distance must be equal to 21 mm)
- STM32 microcontroller with Floating Point Unit (FPU) to acquire the microphones and run the algorithm.

Figure 1: MEMS microphones arrangement and output beam pattern



Microphone distance less than 40 mm gives best results in terms of audio quality and frequency response.

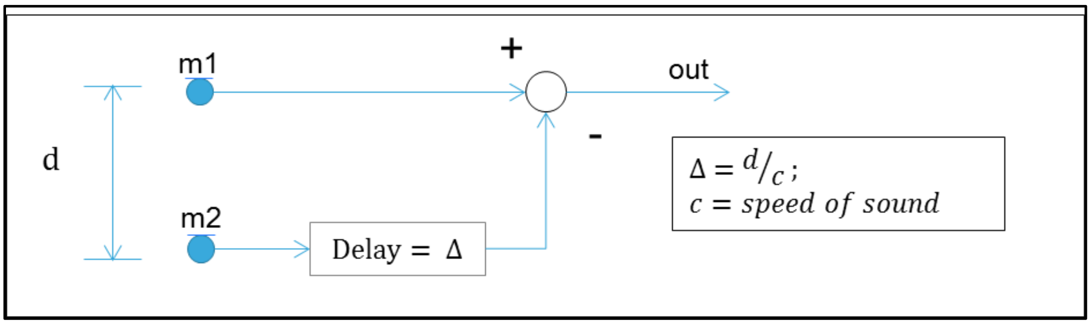
Figure 1: "MEMS microphones arrangement and output beam pattern" shows a typical microphone configuration where the distance between the acoustic holes of two microphones is 4 mm. The blue area represents a cardioid polar pattern that is symmetrical with respect to a line joining the two microphones (typically referred to as an “endfire” configuration). Inverting the order of the microphone signals input into the library results in a beam oriented in the opposite direction. Different directions can thus be obtained with microphone arrays by feeding the library with different signal couples.

The library takes two PDM or PCM input streams from the respective MEMS microphones and is able to output up to two PCM channels (16 kHz, 16 bits per sample) representing the algorithm output (first channel) and the unprocessed output from one of the two omnidirectional microphones (second channel). The latter may be used as a reference for test and evaluation purposes.

2.1 Description

The library supports a first-order Differential Microphone Array (DMA) based on two omnidirectional digital MEMS microphones, such as MP34DT01. A cardioid beam-pattern is implemented by delaying the sound signal captured by one of the microphones by an amount equal to the acoustic delay between the microphones along the “endfire” direction, as shown in *Figure 2: "Cardioid Differential Microphone Array"*.

Figure 2: Cardioid Differential Microphone Array



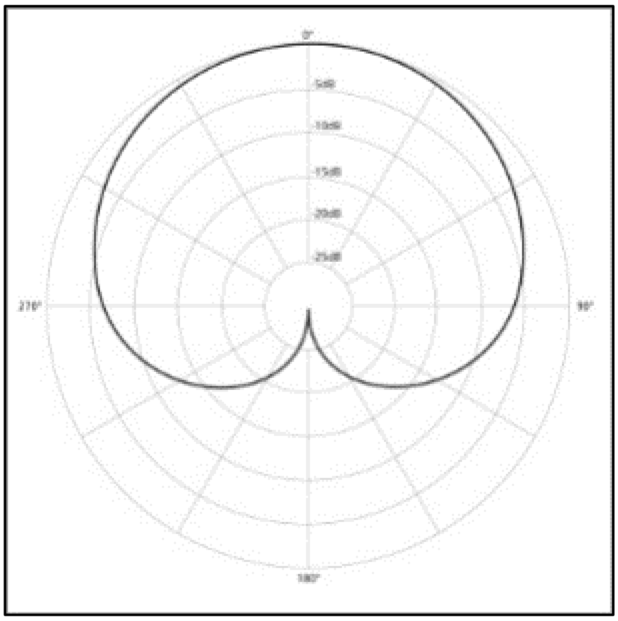
In an extended field context where all the acoustic sources are much further from the microphones than the distance ‘d’ between microphones, the system can be described by the following equation:

Equation 1

$$out(t) = m_1(t) - m_2(t - \Delta);$$

In a typical DMA, the distance between microphones is much shorter than the shortest acoustic wavelength of interest, and the resulting beam pattern is reasonably independent of frequency. *Figure 3: "Ideal cardioid beam pattern"* shows an ideal cardioid beam pattern.

Figure 3: Ideal cardioid beam pattern



2.2 Processing options

The beamforming function implements a modular composition of functional blocks. Its overall complexity is scaled into four levels of algorithm intensity; each of them implementing a function subset. The library can therefore match different user requirements, by allowing levels of trade-off between resource consumption and output quality (both in terms of directivity and SNR). The different processing modes are described below.

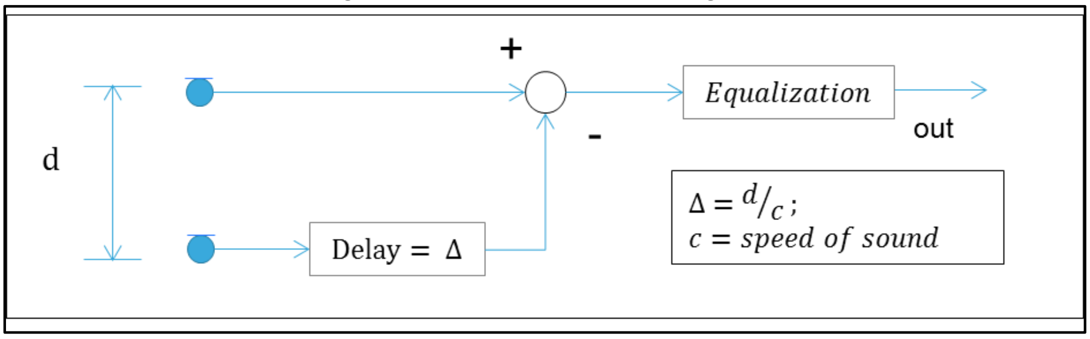
2.2.1 Cardioid basic

When the library is initialized using this option, a cardioid beam former is implemented based on a first-order Differential Microphone Array (DMA).

The Differential Microphone Array configuration alters the frequency response of individual microphones, introducing high-pass behavior rising in frequency at 6 dB/octave. For this reason, a filtering stage is added in order to flatten the system response in the audio frequency range.

Figure 4: "Cardioid basic block diagram" shows a block diagram of the algorithm when used in this setup.

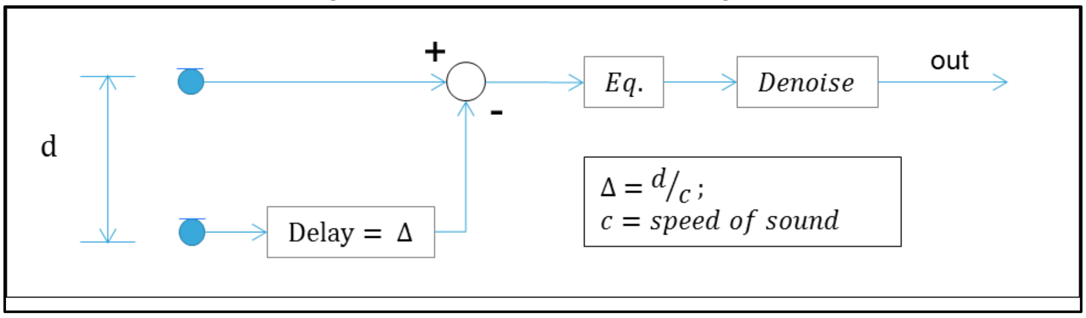
Figure 4: Cardioid basic block diagram



2.2.2 Cardioid denoise

The basic cardioid setup above produces audible noise, particularly at low frequencies, which could be undesirable for certain applications. For this reason, the Cardioid denoise option activates a de-noising filter applied to the output of the cardioid beamforming, as illustrated in *Figure 5: "Cardioid denoise block diagram"*.

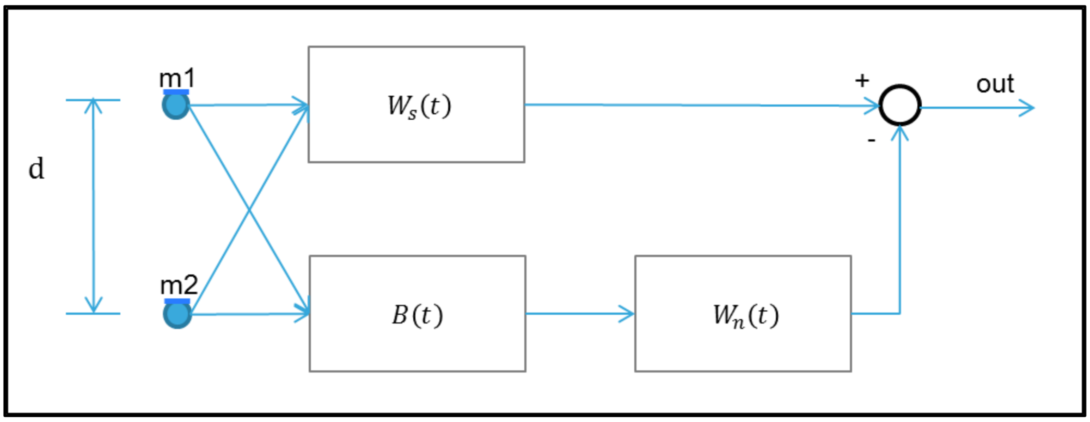
Figure 5: Cardioid denoise block diagram



2.2.3 Strong

The “Strong” option activates a Generalized Sidelobe Canceller (GSC) algorithm adapted to the Differential Microphone Array architecture. With this approach, the output of a cardioid beamformer aimed at the desired source is merged with another signal, defined by a ‘blocking matrix’, which is adaptively filtered so as to minimize the power of unwanted audio components when the two signal paths are recombined *[1]*. A general schema of GSC is shown in *Figure 6: "Generalized Sidelobe Canceller"*.

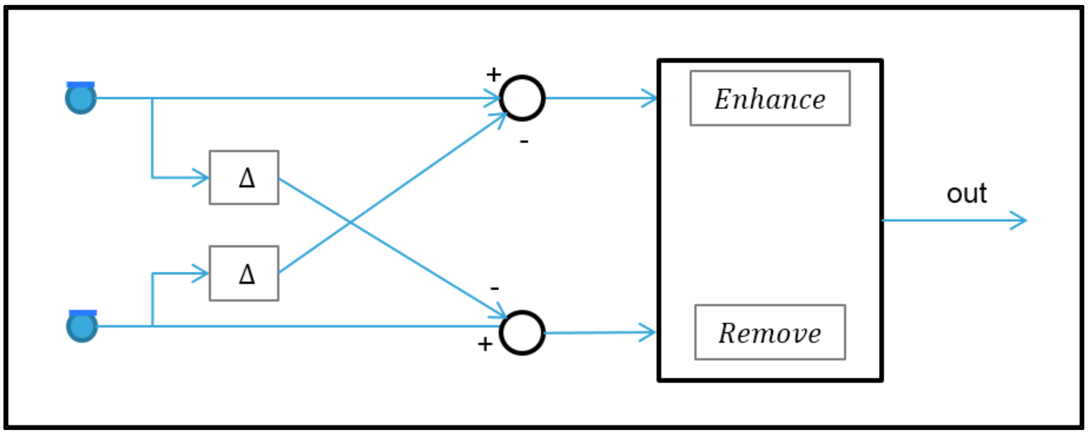
Figure 6: Generalized Sidelobe Canceller



In *Figure 6: "Generalized Sidelobe Canceller"*, the $W_s(t)$ operation implements the cardioid processing scheme shown in *Figure 7: ""Optimization: Strong" block diagram"*, and operator implements a symmetrical schema where the microphones are exchanged so as to aim the beam in the opposite direction, with a ‘zero’ of the beam pattern towards the actual source. This configuration is known as back to back cardioid *[2]*.

In the implementation of the "Strong" type, an adaptive filter fed with back to back cardioid signals removes the noise components from the target source and outputs a signal characterized by a very directive pattern. The last processing step is the application of a de-noising filter, like in the Cardioid denoise option, which removes residual noise. *Figure 7: ""Optimization: Strong" block diagram"* highlights the process.

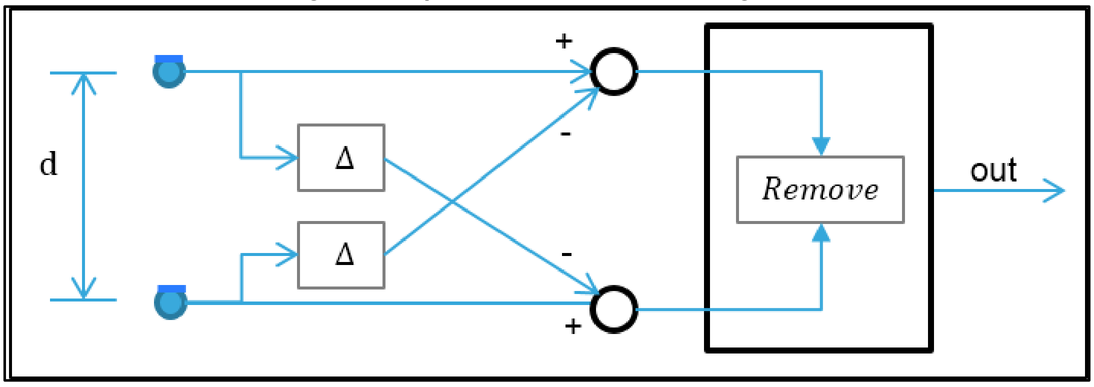
Figure 7: "Optimization: Strong" block diagram



2.2.4 ASR ready

The ASR ready option is a subset of the strong optimization, in which the final de-noising filter is not activated. This produces an output signal characterized by the same directivity pattern as “Strong”, but residual noise is not removed. This leads savings in terms of resources without affecting the ASR performance. *Figure 8: ""Optimization: ASR" block diagram"* shows the process for this optimization level. For ASR performance tests, please refer to *Section 2.4.2: "ASR test"*

Figure 8: "Optimization: ASR" block diagram



2.3 Microphone matching

In order to achieve the best results, the microphone sensitivities must be properly matched. The beamforming library provides a means for setting a calibration parameter corresponding to a gain difference between the two microphones.

2.4 Tests

Polar pattern and ASR tests were performed in an anechoic chamber to document the beamforming polar pattern and overall output quality.

2.4.1 Polar pattern

To evaluate the shape of the beam pattern created by the algorithm, a testing environment was set up in an anechoic chamber using a high quality loudspeaker and a rotating support with a 4 mm differential microphone array mounted on top. The microphone array was rotated manually in steps of 10 degrees, and Gaussian white noise was played by the loudspeaker. For comparison purposes, the system was set up so to generate three algorithm outputs at the same time:

1. the omnidirectional signal of one of the two microphones composing the subsystem
2. one of the two cardioids created (no optimization)
3. the overall system (ASR-ready optimization)

The "cardioid denoise" and "strong" enhancements were not used in this test because their output polar pattern is exactly the same as in the"basic cardioid" and "ASR-ready" optimizations respectively, because the final denoising steps don't affect the polar pattern. The power of the acquired audio signal was computed on a host for each direction of arrival and for each array output.

Figure 9: "Beam patterns" shows the respective polar patterns for omnidirectional microphone (blue), Basic cardioid (red), and ASR-ready (green).

Figure 9: Beam patterns

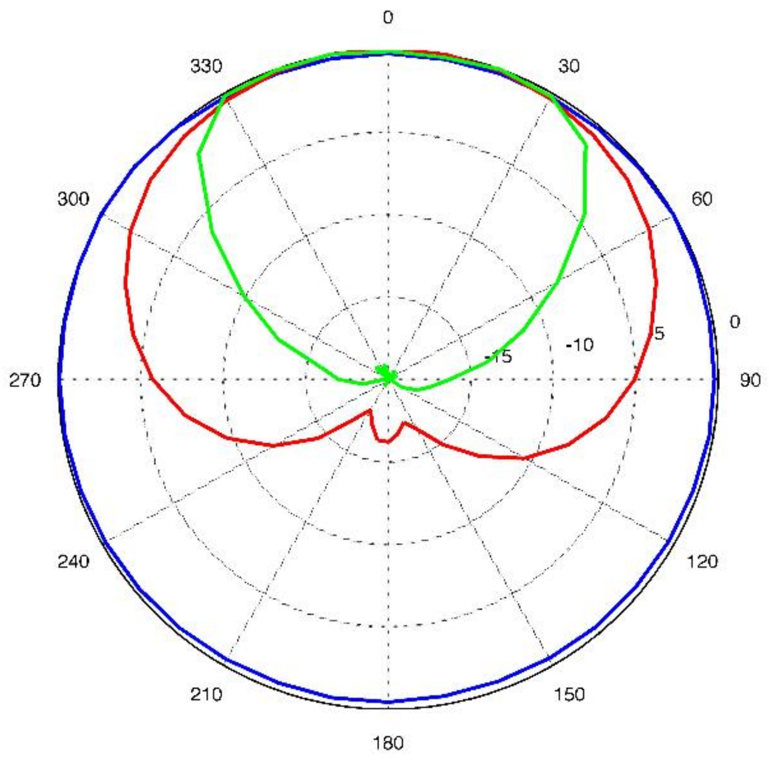
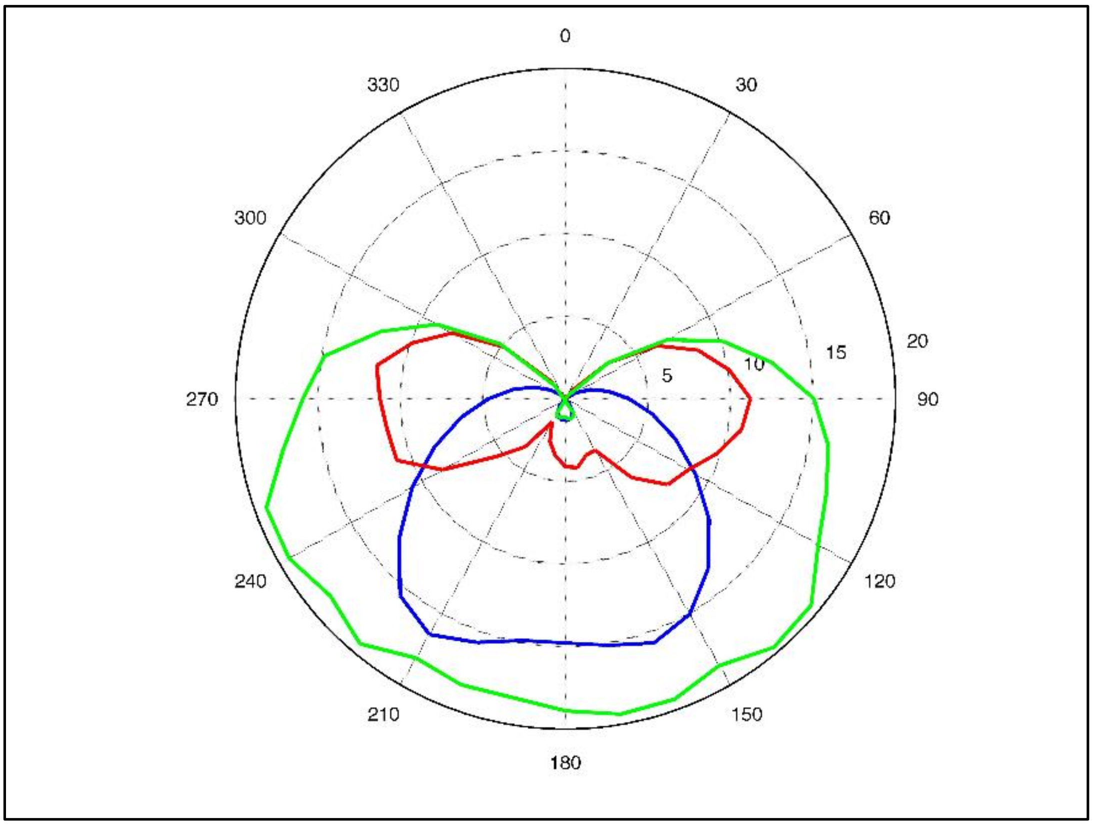


Figure 10: "Improvement in directionality" shows a comparative analysis of the improvement in directionality gained with each type of processing:

- the green line represents the directionality improvement of the overall system (ASR-ready optimization) over the single omnidirectional microphone
- the red line shows the improvement of the ASR-ready optimization over the standard DMA cardioid beamforming (no optimization).
- the blue one represents the comparison between standard cardioid beamforming (no optimization) and single omnidirectional microphone.

Figure 10: Improvement in directionality

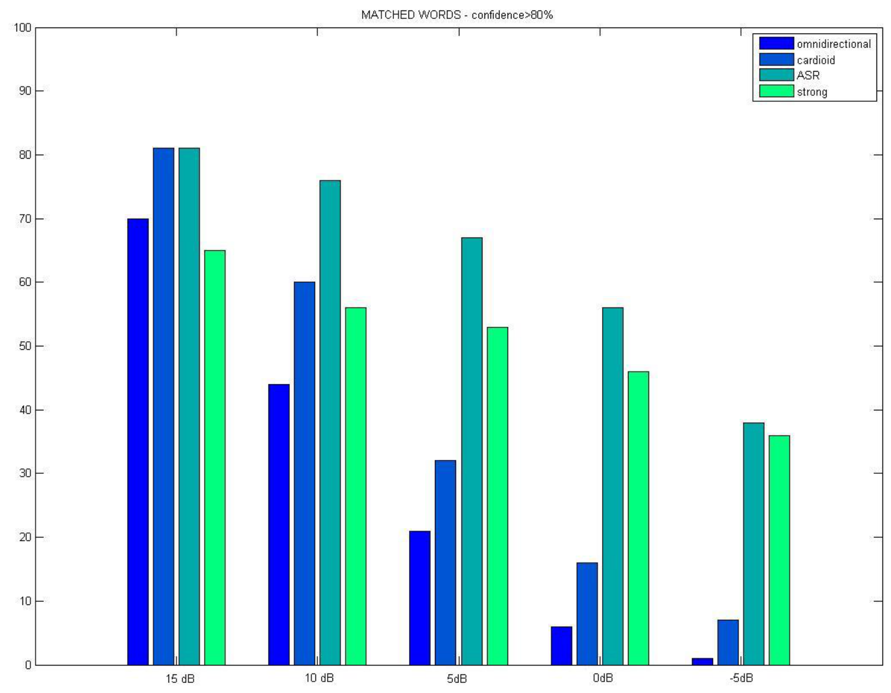


2.4.2 ASR test

For this test, a series of isolated words were reproduced by a high quality loudspeaker placed in the beam direction (0 degrees in the polar pattern), while white noise was reproduced at 90 degrees. Several measurements were taken with changes to the ratio between the noise and voice and using different data sets built using words chosen from the same phonetic groups in order to minimize the difference between word intelligibility. The recorded signals were sent to the online Google ASR service for evaluation.

Figure 11: "ASR test results" shows how the system improves ASR performance even in low Signal to Noise Ratio (SNR) scenarios.

Figure 11: ASR test results



3 osxAcousticBF software expansion for X-CUBE-MEMSMIC1

3.1 Overview

The osxAcousticBF software package expands the functionality of STM32Cube. It is an add-on for the X-CUBE-MEMSMIC1 pack and exploits its features for digital MEMS microphone acquisition, decimation and streaming. For further information regarding X-CUBE-MEMSMIC1, including the microphone acquisition process and software downloads, please refer to www.st.com.

For details on how to merge the two packages into a single package, refer to *Firmware and software*.

The key features of this add-on package are:

- osxAcousticBF library middleware
- sample implementation available for X-NUCLEO-CCA02M1 board plugged onto a NUCLEO-F401RE, based on the X-CUBE-MEMSMIC1 package.
- detailed chm documentation of the library API

3.2 Architecture

The package extends X-CUBE-MEMSMIC1 by providing:

- an additional middleware component for the osxAcousticBF library
- a sample implementation that exploits the X-CUBE-MEMSMIC1 capabilities for microphone acquisition, decimation and streaming.

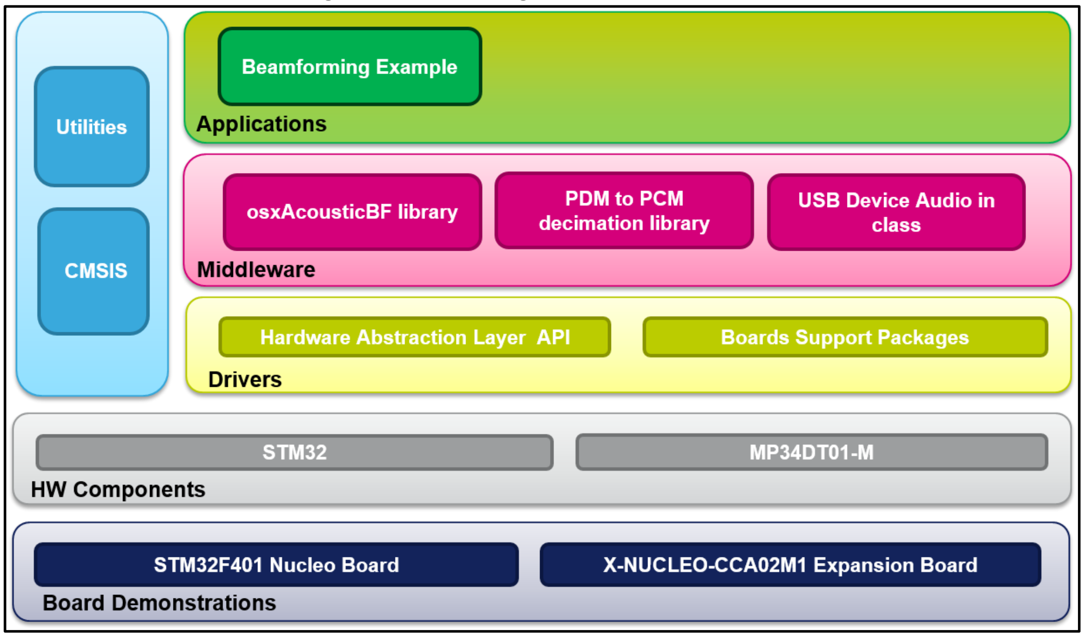
The two packages must therefore be merged in order to obtain a fully functional application based on the osxAcousticBF library, available to the user as a reference for software design and library integration.

The resulting application architecture includes the following layers needed to access the microphone data:

- **STM32Cube HAL layer**: consists of a set of simple, generic, multi-instance APIs (application programming interfaces) which interact with the upper layer applications, libraries and stacks. These generic and extension APIs are based on a common framework which allows any layers they built on, such as the middleware layer, to implement their functions without requiring specific hardware information for a given microcontroller unit (MCU). This structure improves library code reusability and guarantees easy portability across other devices.
- **Board Support Package (BSP) layer**: provides software support for the STM32 Nucleo board peripherals, excluding the MCU. These specific APIs provide a programming interface for certain board specific peripherals like LEDs, user buttons, etc and can also be used to fetch individual board version information. It also provides support for initializing, configuring and reading data.

Specifically, for the X-NUCLEO-CCA02M1, the BSP provides the interface for the MP34DT01-M digital MEMS microphones.

Figure 12: Resulting software architecture



3.3 Folder structure

The following folders are included in the package:

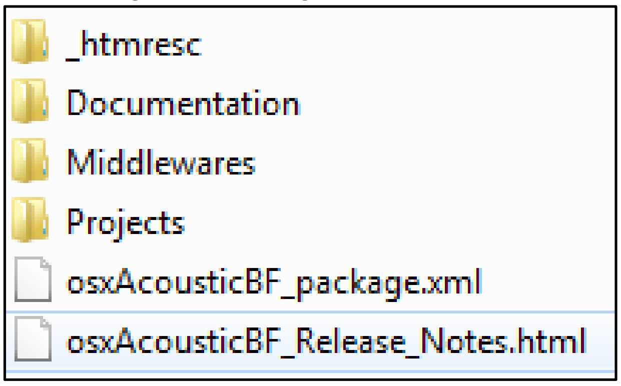
Documentation: contains a compiled HTML file generated from the source code, detailing the software components and APIs.

Drivers: contains the HAL drivers, the board specific drivers for each supported board or hardware platform, including those for the onboard components and the CMSIS layer which is a vendor-independent hardware abstraction layer for the Cortex-M processor series.

Middlewares: contains the osxAcousticBF library binary code, documentation and license information.

Projects: contains a sample application used to demonstrate the library, provided for the NUCLEO-F401RE platform with three development environments: (IAR) Embedded Workbench for ARM, RealView Microcontroller Development Kit (MDK-ARM) and System Workbench for STM32 (SW4STM32).

Figure 13: Package folder structure



_htmresc
Documentation
Middlewares
Projects
osxAcousticBF_package.xml
osxAcousticBF_Release_Notes.html

3.4 APIs

Detailed technical information fully describing the functions and parameters of the osxAcousticBF APIs can be found in the osxAcousticBF_Package.chm compiled HTML file located in the Documentation folder of the software package. In the same document, you can find general descriptions regarding all the osxAcoustic library APIs and the concepts behind their design.

osxAcousticBF is provided as a node-locked library which allows derivative firmware images to run on a specific STM32 Nucleo device only. Licensing activation codes must be requested from ST and included in the project (and become part of the build process) prior to attempting its usage. The resulting firmware binary image will therefore be node-locked.

For complete information about the open.AUDIO license agreement, please refer to the license file located in the Middlewares/ST/ STM32_OSX_AcousticBF_Library folder.

4 System setup guide

4.1 Hardware setup

The library sample application needs the following hardware:

- NUCLEO-F401RE board (visit for further details).
- X-NUCLEO-CCA02M1 expansion board: for this specific application, the board solder jumpers must be configured in order to acquire the two onboard microphones, which is the default factory setting. The correct setup is described in *[3]*. Refer to the same user manual for information regarding board connections and supplying power.

4.2 Firmware and software

The main steps required to setup the environment are:

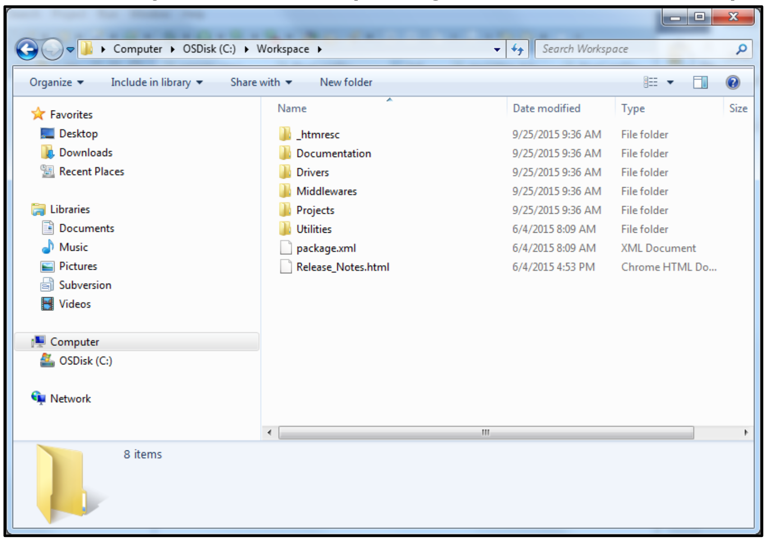
1. Additional package download and unpack
2. Library package installation
3. Node locking procedure
4. Compiling and running

4.2.1 Additional package download and unpack

The application is designed as an add-on project for the X-CUBE-MEMSMIC1 v1.1, which you can find at *www.st.com*.

After downloading, unpack the zip file in a directory of your choice. This will become the workspace used to test and work with the library. Avoid using paths names longer than 240 characters because it can cause problems in some tool chains during project build. In our example below, the workspace is placed in the “C:\Workspace” folder.

Figure 14: Directory content after unpacking the X-CUBE-MEMSMIC1 package



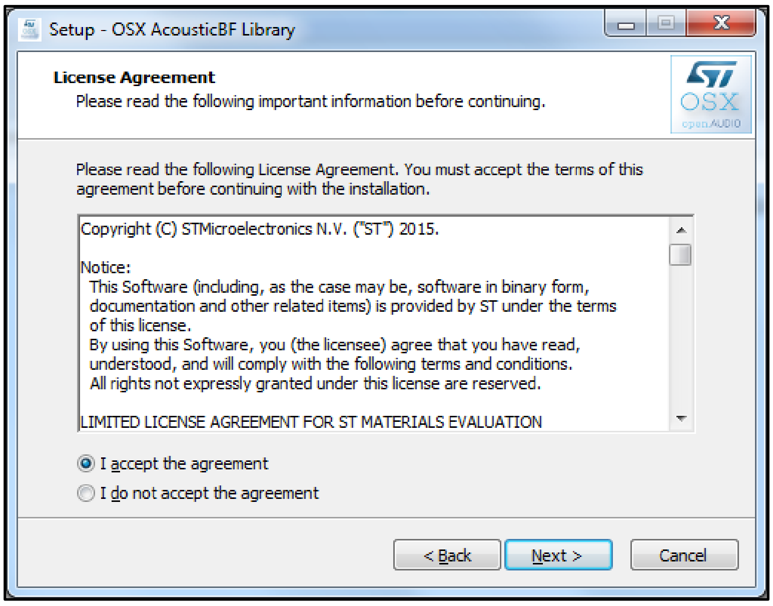
Name	Date modified	Type	Size
_htmresc	9/25/2015 9:36 AM	File folder	
Documentation	9/25/2015 9:36 AM	File folder	
Drivers	9/25/2015 9:36 AM	File folder	
Middlewares	9/25/2015 9:36 AM	File folder	
Projects	9/25/2015 9:36 AM	File folder	
Utilities	6/4/2015 8:09 AM	File folder	
package.xml	6/4/2015 8:09 AM	XML Document	
Release_Notes.html	6/4/2015 4:53 PM	Chrome HTML Do...	

4.2.2 Library package installation

Double click the “osxAcousticBF_Setup” library installer available at *www.st.com* in order to start the installation procedure.

The first step is to accept the License Agreement:

Figure 15: License Agreement



License Agreement

Please read the following important information before continuing.

Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.

Copyright (C) STMicroelectronics N.V. ("ST") 2015.

Notice:
This Software (including, as the case may be, software in binary form, documentation and other related items) is provided by ST under the terms of this license.
By using this Software, you (the licensee) agree that you have read, understood, and will comply with the following terms and conditions.
All rights not expressly granted under this license are reserved.

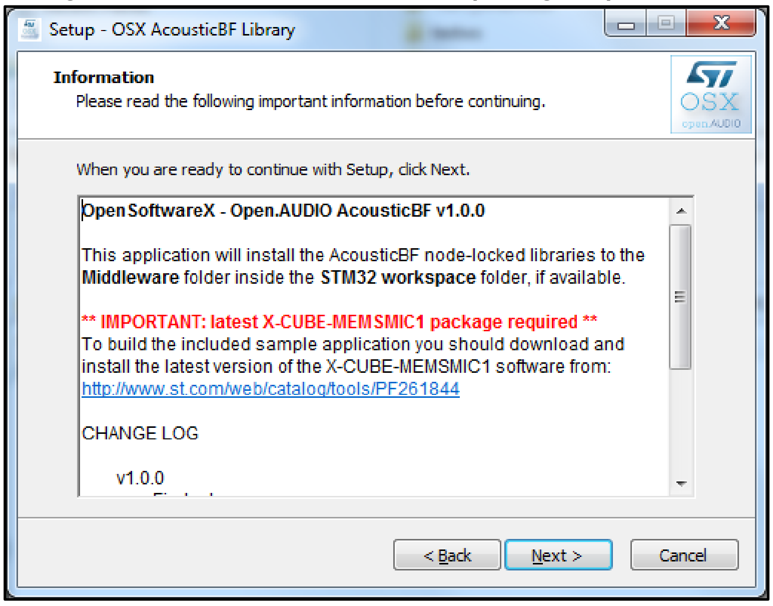
LIMITED LICENSE AGREEMENT FOR ST MATERIALS EVALUATION

[✓ I accept the agreement]

I do not accept the agreement

Then the procedure details the additional packages required:

Figure 16: Information and additional package requirements



Information

Please read the following important information before continuing.

When you are ready to continue with Setup, click Next.

OpenSoftwareX - Open.AUDIO AcousticBF v1.0.0

This application will install the AcousticBF node-locked libraries to the **Middleware** folder inside the **STM32 workspace** folder, if available.

**** IMPORTANT: latest X-CUBE-MEMSMIC1 package required ****
To build the included sample application you should download and install the latest version of the X-CUBE-MEMSMIC1 software from:
http://www.st.com/web/catalog/tools/PF261844

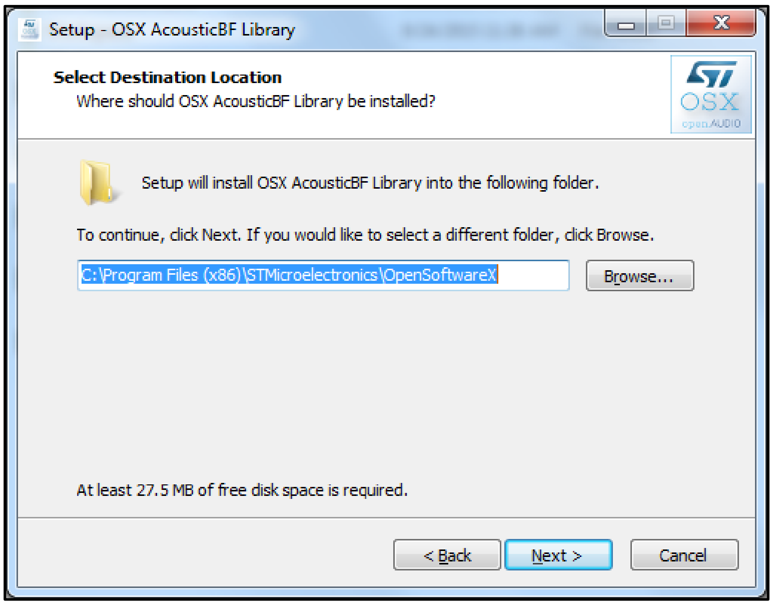
CHANGE LOG

v1.0.0

Now you can choose the directory where you want to install the library software. Note that this is not the path for the firmware, but where the PC software related information (uninstall and other information) shall be stored.

This folder does not necessarily have to be your workspace folder.

Figure 17: Installation directory for the PC software



Select Destination Location

Where should OSX AcousticBF Library be installed?

Setup will install OSX AcousticBF Library into the following folder.

To continue, click Next. If you would like to select a different folder, click Browse.

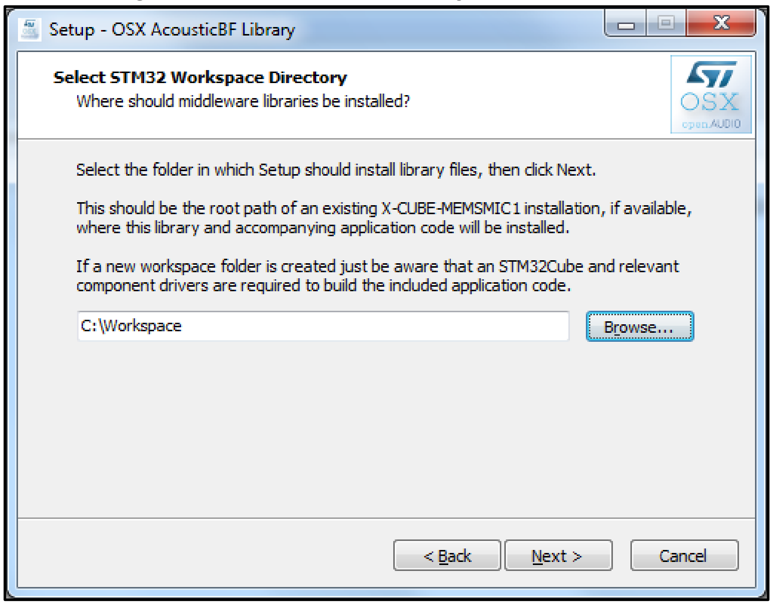
C:\Program Files (x86)\STMicroelectronics\OpenSoftwareX

At least 27.5 MB of free disk space is required.

Then you choose the directory where the firmware will be unpacked.

This folder must be your desired workspace directory (“C:\Workspace” in our example).

Figure 18: Installation directory for the firmware



Select STM32 Workspace Directory

Where should middleware libraries be installed?

Select the folder in which Setup should install library files, then click Next.

This should be the root path of an existing X-CUBE-MEMSMIC1 installation, if available, where this library and accompanying application code will be installed.

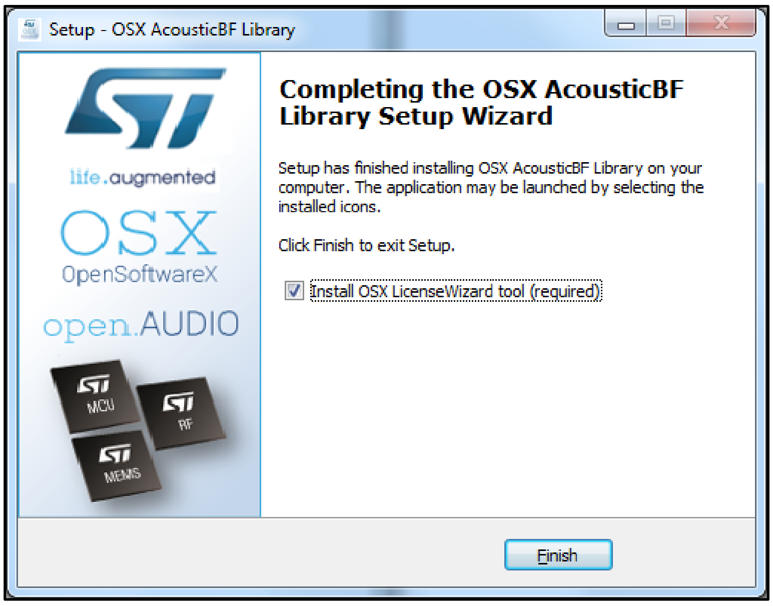
If a new workspace folder is created just be aware that an STM32Cube and relevant component drivers are required to build the included application code.

C:\Workspace

Finally you can complete the installation procedure.

The LicenseWizard software is required for the next steps.

Figure 19: OSX LicenseWizard tool installation



Completing the OSX AcousticBF Library Setup Wizard

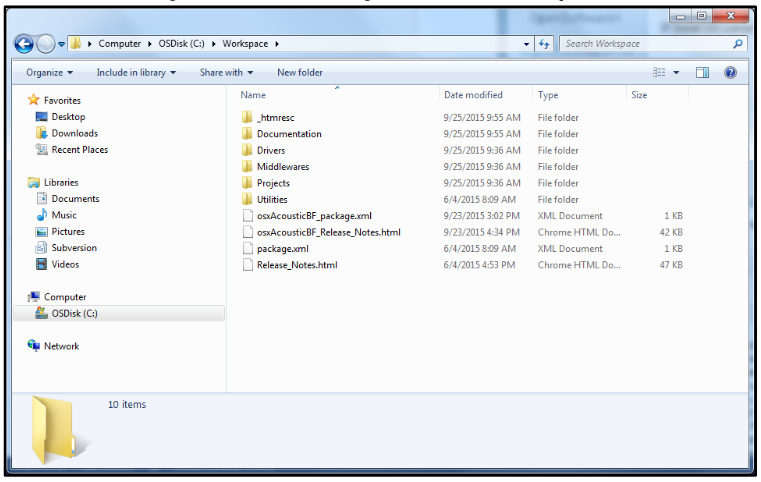
Setup has finished installing OSX AcousticBF Library on your computer. The application may be launched by selecting the installed icons.

Click Finish to exit Setup.

[✓] Install OSX LicenseWizard tool (required)

At this stage, your Workspace folder should resemble the image below.

Figure 20: Resulting firmware workspace



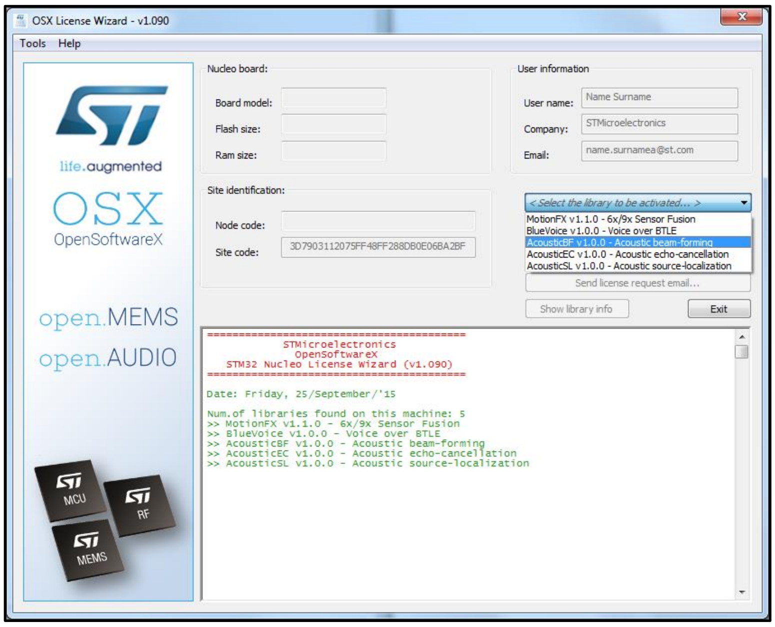
Name	Date modified	Type	Size
_htmresc	9/25/2015 9:55 AM	File folder	
Documentation	9/25/2015 9:55 AM	File folder	
Drivers	9/25/2015 9:36 AM	File folder	
Middlewares	9/25/2015 9:36 AM	File folder	
Projects	9/25/2015 9:36 AM	File folder	
Utilities	6/4/2015 8:09 AM	File folder	
osxAcousticBF_package.xml	9/23/2015 3:02 PM	XML Document	1 KB
osxAcousticBF_Release_Notes.html	9/23/2015 4:34 PM	Chrome HTML Do...	42 KB
package.xml	6/4/2015 8:09 AM	XML Document	1 KB
Release_Notes.html	6/4/2015 4:53 PM	Chrome HTML Do...	47 KB

You will find the installed library in the Middleware folder, while the sample application projects are in the Project folder.

4.2.3 Node locking procedure

Connect a NUCLEO-F401RE board to the PC, run the OSX LicenseWizard you installed in the previous steps and choose the library you want to activate.

Figure 21: Library selection


OSX License Wizard - v1.090
Tools
Help
life.augmented
OSX
OpenSoftwareX
open.MEMS
open.AUDIO
MCU
RF
MEMS
< Select the library to be activated... >
Send license request email...
Show library info
Exit

Nucleo board:

Board model:

Flash size:

Ram size:

User information

User name: Name Surname

Company: STMicroelectronics

Email: name.surnamea@st.com

Site identification:

Node code:

Site code: 3D7903112075FF48FF288DB0E06BA2BF

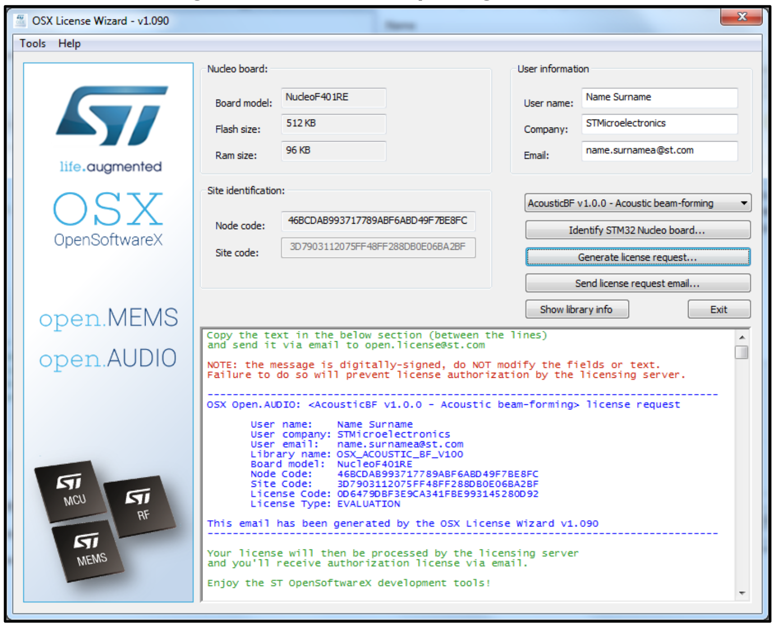
MotionFX v1.1.0 - 6x/9x Sensor Fusion
BlueVoice v1.0.0 - Voice over BTLE
AcousticBF v1.0.0 - Acoustic beam-forming
AcousticEC v1.0.0 - Acoustic echo-cancellation
AcousticSL v1.0.0 - Acoustic source-localization

```
===========================================
            STMicroelectronics
              OpenSoftwareX
   STM32 Nucleo License Wizard (v1.090)
===========================================

Date: Friday, 25/September/'15

Num.of libraries found on this machine: 5
>> MotionFX v1.1.0 - 6x/9x Sensor Fusion
>> BlueVoice v1.0.0 - Voice over BTLE
>> AcousticBF v1.0.0 - Acoustic beam-forming
>> AcousticEC v1.0.0 - Acoustic echo-cancellation
>> AcousticSL v1.0.0 - Acoustic source-localization
```

Now use the appropriate buttons on the wizard to:

- identify the STM32 Nucleo board
- generate the license request
- send the request e-mail

Figure 22: License request generation


OSX License Wizard - v1.090
Tools
Help
life.augmented
OSX
OpenSoftwareX
open.MEMS
open.AUDIO
MCU
RF
MEMS
Identify STM32 Nucleo board...
Generate license request...
Send license request email...
Show library info
Exit

Nucleo board:

Board model: NucleoF401RE

Flash size: 512 KB

Ram size: 96 KB

User information

User name: Name Surname

Company: STMicroelectronics

Email: name.surnamea@st.com

Site identification:

Node code: 46BCDAB993717789ABF6ABD49F7BE8FC

Site code: 3D7903112075FF48FF288DB0E06BA2BF

AcousticBF v1.0.0 - Acoustic beam-forming

```
Copy the text in the below section (between the lines)
and send it via email to open.license@st.com

NOTE: the message is digitally-signed, do NOT modify the fields or text.
Failure to do so will prevent license authorization by the licensing server.

---------------------------------------------------------------------------
OSX Open.AUDIO: <AcousticBF v1.0.0 - Acoustic beam-forming> license request

       User name:    Name Surname
       User company: STMicroelectronics
       User email:   name.surnamea@st.com
       Library name: OSX_ACOUSTIC_BF_V100
       Board model:  NucleoF401RE
       Node Code:    46BCDAB993717789ABF6ABD49F7BE8FC
       Site Code:    3D7903112075FF48FF288DB0E06BA2BF
       License Code: 0D6479DBF3E9CA341FBE993145280D92
       License Type: EVALUATION

This email has been generated by the OSX License Wizard v1.090
---------------------------------------------------------------------------

Your license will then be processed by the licensing server
and you'll receive authorization license via email.

Enjoy the ST OpenSoftwareX development tools!
```

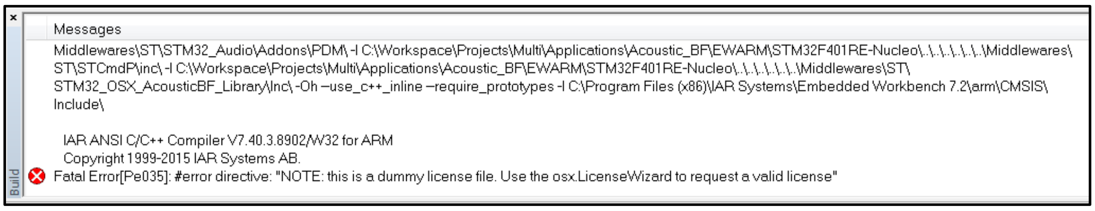
The license mail should arrive after a few moments.

4.2.4 Compiling the example projects

Open the project with your favorite tool chain. In this example, the projects for three IDEs are located in “C:\Workspace\Projects\Multi\Applications\Acoustic_BF”.

Try to compile the project: you should receive an error in the osx_license.h file.

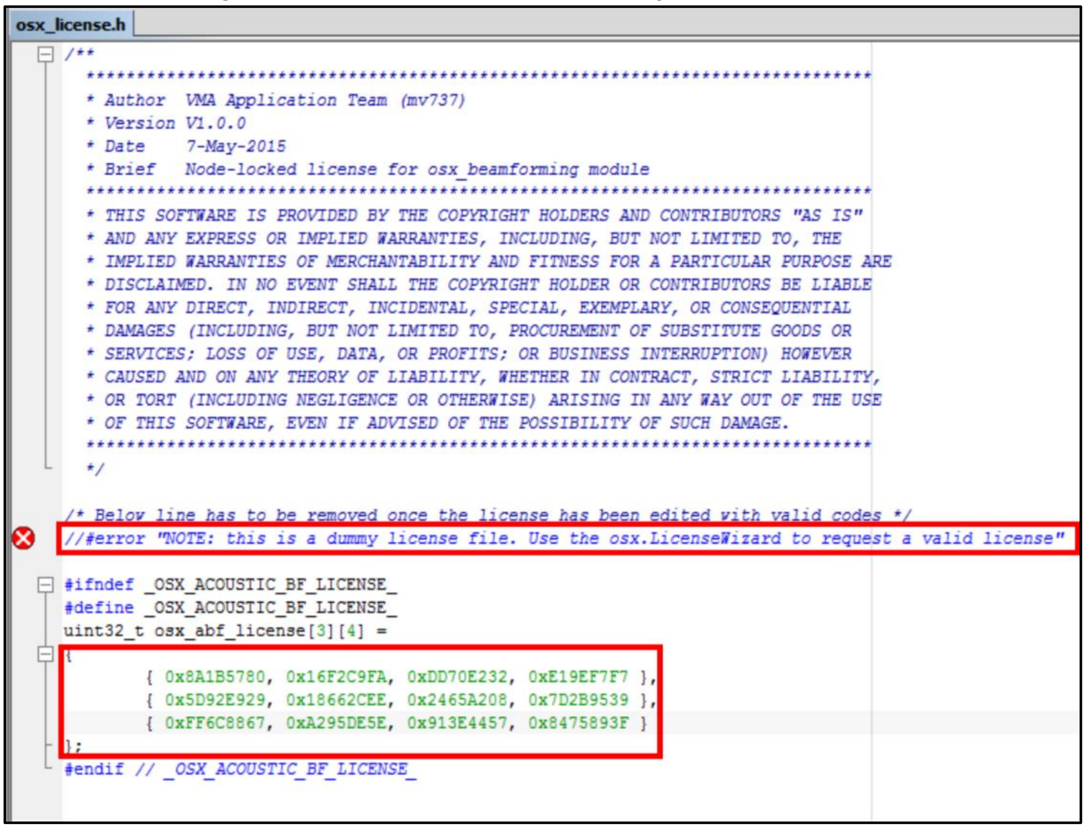
Figure 23: Error while building the original osx_license.h file



```
Messages
Middlewares\ST\STM32_Audio\Addons\PDM\ -I C:\Workspace\Projects\Multi\Applications\Acoustic_BF\EWARM\STM32F401RE-Nucleo\..\..\..\..\..\..\Middlewares\
ST\STCmdP\inc\ -I C:\Workspace\Projects\Multi\Applications\Acoustic_BF\EWARM\STM32F401RE-Nucleo\..\..\..\..\..\..\Middlewares\ST\
STM32_OSX_AcousticBF_Library\Inc\ -Oh –use_c++_inline –require_prototypes -I C:\Program Files (x86)\IAR Systems\Embedded Workbench 7.2\arm\CMSIS\
Include\

  IAR ANSI C/C++ Compiler V7.40.3.8902/W32 for ARM
  Copyright 1999-2015 IAR Systems AB.
Fatal Error[Pe035]: #error directive: "NOTE: this is a dummy license file. Use the osx.LicenseWizard to request a valid license"
```

Open the osx-license.h file and substitute the dummy license array with the one you received by mail from the Open.Audio server. Also, comment the define directive depicting the error.

Figure 24: osx_license.h file after the required modifications



```
osx_license.h
/**
  ******************************************************************************
  * Author   VMA Application Team (mv737)
  * Version  V1.0.0
  * Date     7-May-2015
  * Brief    Node-locked license for osx_beamforming module
  ******************************************************************************
  * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
  * AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
  * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
  * DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE
  * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
  * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
  * SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
  * CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
  * OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
  * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
  ******************************************************************************
  */

/* Below line has to be removed once the license has been edited with valid codes */
//#error "NOTE: this is a dummy license file. Use the osx.LicenseWizard to request a valid license"

#ifndef _OSX_ACOUSTIC_BF_LICENSE_
#define _OSX_ACOUSTIC_BF_LICENSE_
uint32_t osx_abf_license[3][4] =
{
        { 0x8A1B5780, 0x16F2C9FA, 0xDD70E232, 0xE19EF7F7 },
        { 0x5D92E929, 0x18662CEE, 0x2465A208, 0x7D2B9539 },
        { 0xFF6C8867, 0xA295DE5E, 0x913E4457, 0x8475893F }
};
#endif // _OSX_ACOUSTIC_BF_LICENSE_
```

You should be now able to compile and flash the board for which you requested the license in order to use the library example.

5 Sample application description

A sample application using the osxAcousticBF library with NUCLEO-F401RE and X-NUCLEO-CCA02M1 boards is provided in the Projects directory. Ready to build projects are available for multiple IDEs.

The application is designed to acquire the two microphones soldered on the X-NUCLEO-CCA02M1 board, perform beamforming and stream two audio channels via USB to a host PC.

The two audio streams contain:

- the algorithm output (first channel)
- an omnidirectional microphone as a reference (second channel)

5.1 Firmware steps

Follow these steps to obtain the desired behavior:

- Initialize and start the USB audio input driver and middleware. This allows a host PC to recognize the device as a standard multichannel USB microphone.
- Initialize microphone acquisition using the relevant BSP function.
- Initialize the osxAcousticBF library.
- Start the audio acquisition that will trigger library execution.
- Processed data and the omnidirectional microphone reference data are passed to USB driver every millisecond.

All the operations related to the osxAcousticBF library are performed in dedicated functions in the audio_application.c file.

Further details about the library API can be found in the chm help file in the Documentation folder.

5.2 Example program execution

In order to record audio from the device and evaluate beamforming, you need to install third party software, like the Audacity® freeware program, to save or play the streamed signal.

Detailed information on how to setup the host system for audio recording can be found in the X-CUBE-MEMSMIC1 user manual, available at www.st.com.

Further information about Audacity® can be obtained at: *http://audacityteam.org/?lang=en*

6 Library profiling

Profiling was performed in order to evaluate the library resource consumption in terms of MIPS, RAM and Flash. Detailed information can be found in the osxAcousticBF_Package.chm compiled HTML file located in the Documentation folder.

7 References

1. McCowan, I. (2001). Microphone arrays : A tutorial.
2. Mingsian R. Bai, J.-G. I. (2010). Acoustic Array Systems: Theory, Implementation, and Application. Wiley.
3. User manual, UM1900 - Getting started with the digital MEMS microphones expansion board based on MP34DT01-M for STM32 Nucleo. STMicroelectronics

8 Revision history

Table 1: Document revision history

Date	Version	Changes
01-Dec-2015	1	Initial release.
11-Mar-2016	2	Updated *Section 2.4.1: "Polar pattern"*