

AN11819

LPC5411X USB Audio Application

Rev. 1.0 — 20 May 2016

Application note

Document information

Info	Content
Keywords	LPC5411X, Full Speed USB, Audio, I2S
Abstract	This application note introduces audio playback and record through the standard USB audio class on the LPC5411x MCU. A software application is provided as part of the LPCOpen package to show how the application works.



Revision history

Rev	Date	Description
1.0	20160520	Initial revision.

Contact information

For more information, please visit: <http://www.nxp.com>

For sales office addresses, please send an email to: salesaddresses@nxp.com

1. Introduction

The NXP LPC5411x MCU family, with support for two I2S interfaces and full-speed USB is ideal for USB audio applications. This application highlights the LPC5411x MCU as a low power 'plug and play' USB audio device using the LPC54114 audio and voice recognition kit (OM13090). Once setup, the device can be plugged into a host system that supports the USB audio class and audio can be played back or recorded through the Kit's headphone and line-in connectors respectively. Supported host systems include Windows, Linux, and Mac systems. No special software or drivers need to be installed on the host systems for this application example.

1.1 Highlights of the USB audio application

The USB audio application highlights the following LPC5411x MCU features:

- Use of the 48 MHz internal FRO oscillator as the base clock for the USB and system.
- Cortex M4 MCU and internal RAM (IRAM).
- USB audio class support on the USB full-speed device using the in-ROM USB driver.
- I²C interface for CODEC control implemented on Flexcomm Interface 4.
- I²S transmit and receive interfaces for CODEC audio data implemented on Flexcomm Interface 6 and Flexcomm Interface 7.
- System PLL used as audio clock for I²S interface and CODEC.
- Very low power usage during USB suspend.
- Debug output support on a UART on Flexcomm Interface 0.
- Audio clock synchronization to host clock.

2. Software Design

This section provides an operational overview of the software and supported interfaces for the USB audio application.

2.1 Overview of how the USB Audio application works

When the application is started on the LPC5411x LPCXpresso board and a USB cable is connected between the board and an audio host system, the USB enumerates as an audio class device. The audio class device has several audio endpoints (Isochronous IN and OUT endpoints) and the control endpoint. The IN endpoint is used for streaming of audio data from the board (record), while the OUT endpoint is used for streaming of audio data to the board (playback). The control endpoint is used for audio control: volume, mute, and unmute.

USB Isochronous transfers guarantee bandwidth but do not guarantee data. Audio data is transferred between the host system and board at a specific periodic rate. Some USB transfers may contain data, while others may be empty. The host system will supply data to the board on the OUT endpoint at 48 stereo KSamples/sec (if the host has been requested to play audio). For record, the host system requests data from the board. The board will supply audio data to the host system at 48 stereo KSamples/sec.

2.1.1 Audio sample clocks and buffering

Although both the host system and the board handle stereo data at 48KSamples/sec, they do not use the same audio sample clock. If the host audio clock is running faster than the board's audio clock, the host will eventually send more data than the board can accept in its audio buffers (overflow). If the board's audio clock is running faster than the host system's audio clock, the board's audio buffers will eventually underflow. To these underflow/overflow problems, the application implements a clock adjustment and buffering scheme.

Audio data is buffered on the board in audio buffers to and from the host system. For playback, a few milliseconds of audio data is pre-buffered from the host system prior to starting playback. During playback, the level of the audio buffer is monitored to verify that the buffer is not growing or shrinking beyond the pre-buffered size, which may lead to audio buffer overflow or underflow conditions. If the software detects the buffer is growing or shrinking, it will make very minor adjustments to the audio clock rate on the board to speed up or reduce the audio clock rate. This audio clock rate change increases or reduces audio data rate on the I²S interface and audio CODEC. These clock rate changes are so small that there is no noticeable audio difference in the played back audio output.

During simultaneous playback and record, the board's audio clock rate is adjusted based on the playback data. On record only, the board's audio clock rate is adjusted based on the record data's audio buffer size. If the software determines that the host system is consuming recorded audio data faster or slower than the I²S/CODEC is providing it, it will speed up or slow down the audio clock as necessary.

Audio clock adjustments are made on some of the USB Start of Frame (SOF) events, which occurs about 1000 times/sec. The board's audio data buffer levels are monitored and the audio clock is adjusted as part of the software's SOF event handler.

2.1.2 I²S and CODEC

The I²S transmit interrupt is used to provide audio playback data to the I²S transmit block. The interrupt will trigger when the I²S transmit FIFO is half empty. The interrupt will move data from the audio playback buffer to the I²S FIFO. The I²S block will output data to the CODEC for playback based on the audio clock rate.

For audio record, the I²S block reads data from the CODEC based on the audio clock rate. The I²S receive interrupt will trigger when the I²S receive FIFO is half full. The interrupt will move data from the I²S FIFO to the audio record data buffer.

2.1.3 Background tasks

There are several background tasks that may occur on specific events from the host system: audio volume and power control. Background tasks are lower priority than the USB and audio playback or record tasks.

The host system controls the volume of playback on the board on the USB's control endpoint. Volume changes and mute/unmute events from the host system are handled in the CODEC's background task.

The background power task places the board into low power standby when the CPU is idle, or places the board in a very low power state if the host system suspends USB.

2.2 System initialization and setup

The USB audio application starts with early system initialization, which sets up pin states as needed for the interfaces and system clocking as needed for USB, the M4 core/IRAM/buses, and peripherals.

After early system initialization, the rest of the system and interface peripherals are setup. These include the UART, I²C control interface to the CODEC, two I²S data interfaces to the CODEC, USB, and the audio clock to the CODEC. At this point, the audio CODEC is also configured via the I²C bus.

2.2.1 Pin setup

All used pins are setup to their mux state as needed by the system. This includes Flexcomm I²C Interface, Flexcomm I²S Interface, Flexcomm UART Interface, USB, and the audio clock output. All other unused pins are setup in a high impedance input state to reduce power usage in the system. [Table 1](#) shows the pins that are used by this application.

Table 1. USB Audio application pin mapping
LPC54114 MCU pin mapping for the USB Audio application

Interface	Pin name	Pin function	Peripheral
UART	P0.0_U0RX	UART receive	Flexcomm Interface 0/UART
	P0.1_U0TX	UART transmit	Flexcomm Interface 0/UART
I ² C	P0.25 (FC4_I2C_SCL)	I ² C clock pin to CODEC	Flexcomm Interface 4/I ² C
	P0.26 (FC4_I2C_SDA)	I ² C data pin to CODEC	Flexcomm Interface 4/I ² C
I ² S transmit	P0.5 (FC6_I2S_TX)	I ² S transmit data (to CODEC)	Flexcomm Interface 6/I ² S

Interface	Pin name	Pin function	Peripheral
	P0.6 (FC6_I2S_WS)	I ² S transmit frame (to CODEC)	Flexcomm Interface 6/I ² S
	P0.7 (FC6_I2S_SCK)	I ² S transmit clock (to CODEC)	Flexcomm Interface 6/I ² S
I ² S receive	P1.12 (FC7_I2S_SCK)	I ² S receive clock (to CODEC)	Flexcomm Interface 7/I ² S
	P1.7 (FC7_I2S_RX)	I ² S receive data (from CODEC)	Flexcomm Interface 7/I ² S
	P1.8 (FC7_I2S_WS)	I ² S receive frame (to CODEC)	Flexcomm Interface 7/I ² S
Audio clock	P1.2_MCLK	USB D+ Audio MCLK (to CODEC)	System PLL
USB	USB_DP	USB D+	USB
	USB_DN	USB D-	USB
	P1.6 (USB_VBUS)	USB power (VBUS)	USB

2.2.2 Peripheral functions

Table 2 lists the function of each peripheral or interface used by the board or USB Audio application.

Table 2. Peripherals and interfaces mapped to application function

Peripheral or interface	USB Audio application function
UART	Used for status messages with the DEBUGOUT() macro. Outputs text strings to a console connected to the debugger.
Free Running Oscillator (FRO)	The 48 MHz output from the FRO is used to clock the USB, the Cortex M4 and IRAM, and various system buses.
System PLL	The system PLL is used as the audio clock source. It's routed to the CODEC via the MCLK signal and is used as the clock source for the I ² S blocks.
I ² C	The I ² C interface is connected to the audio CODEC. It is used for CODEC configuration and setup, volume control, and CODEC status.
I ² S	The I ² S interface handles the serial data to and from the audio CODEC. Two I ² S blocks are used – one for playback and one for record.
USB (target)	The USB peripheral on the LPC5411x MCU is used for the USB interface between the LPC5411x and the host system. This is designated J4 on the LPCXpresso54114 board.
USB (debugger)	The USB connector labelled J7 on the LPCXpresso54114 board is used as the debugger interface between the LPC5411x MCU and the host system running the tool chain.
Audio output	The 'Audio HP/Line-Out' plug on the MIC/Audio/OLED shield board is used as the analog audio output from the CODEC. This can be connected to a pair of headphones or the line-in of another system.
Audio input	The 'Audio Line-In' plug on the MIC/Audio/OLED shield board is used as the analog audio input from a Line-in source. It is connected to the audio CODEC.

3. Environment

The LPC5411x USB audio application has been implemented with the commercially available LPCXpresso LPC54114 audio and voice recognition Kit board (OM13090), which comprises an LPCXpresso 54114 and a MIC/Audio/OLED shield board which plugs into the LPCXpresso board.

3.1 Hardware

This section describes the required hardware and how to set it up for the application.

3.1.1 Board set

- LPC54114 audio and voice recognition Kit board (OM13090). For more details visit the following link:

<http://www.nxp.com/products/software-and-tools/hardware-development-tools/lpcxpresso-boards/lpc54114-audio-and-voice-recognition-kit:OM13090>

3.1.2 Debugger

- Integrated CMSIS-DAP debugger on LPCXpresso LPC54114 board.

3.1.3 Miscellaneous

- 2 USB micro cables (for connection between host machine and target and for connection between host machine and integrated debugger).
- CMSIS-DAP firmware is pre-programmed on the LPCXpresso54114 during manufacture, however it is advisable to update to the latest available version. CMSIS-DAP is installed using LPCScript tool. Go to <https://www.lpcware.com/LPCScript> for the latest information on how to use LPCScript to update your board's CMSIS-DAP firmware.

3.1.4 Board setup

Both the LPCXpresso54114 board and the shield board use default jumper settings with the exception of jumper JP3 on the MIC/Audio/OLED shield board. The jumper settings are provided below for reference and should be verified and changed to match the list below if needed. The shield board should be inserted with the OLED display situated towards the middle of the LPCXpresso54114 board and the audio connectors at the outside edge of the LPCXpresso54114 board.

LPC54114 LPCXpresso board jumper settings:

- JP1: OPEN
- JP2: Pins 1-2 (LOC)
- JP5: OPEN
- JP6: OPEN
- JP9: Pins 2-3
- JP10: CLOSED

MIC/Audio/OLED shield board jumper settings:

- JP1: OPEN
- JP2: OPEN
- JP3: Pins 1-2
- P1-1-2: CLOSED
- P1-3-4: CLOSED
- P1-4-5: CLOSED
- P2: OPEN
- P3: OPEN
- P4-1-2: CLOSED
- P4-3-4: CLOSED
- P4-5-6: CLOSED
- P5-1-2: OPEN
- P5-3-4: OPEN
- P5-5-6: CLOSED

3.1.5 USB connections

The USB connector labelled Link (J7) on the LPCXpresso54114 is used for the integrated CMSIS-DAP debugger interface between the target (LPC5411x) MCU and the host machine that is running the development tools. Connect a micro USB cable between these 2 connectors.

CMSIS-DAP should already be installed using LPCScrypt. See Section 3.1.3 for more information.

The USB connector labelled Target (J5) is used for the USB interface between the target MCU (LPC5411x) and the audio host machine. The audio host machine can be a Linux, Windows, or MAC system.

3.1.6 Audio interface connections

A pair of headphones or powered speakers are required to hear playback from the Kit. Plug this into the 'Audio HP\Line out' plug on the MIC/Audio/OLED shield board.

To record audio to the Kit, you will need to drive an audio signal at 'line in' level into the 'Audio Line-in' plug on the MIC/Audio/OLED shield board. Using a standard microphone will not work as signal levels from such devices are typically too low. A 'line out' signal, as driven from a laptop, or the headphone output of a smart phone will work.

3.2 Software

This section describes development tools and driver software required for the application.

3.2.1 Supported development tool chains

The USB audio example is provided with projects that support the following development tool chains:

- LPCXpresso Integrated Development Environment (IDE) (version 8.1 or later).
- Keil uVision.
- IAR Embedded Workbench for ARM.

The version numbers for the each supported tool chain are included as part of the LPCOpen package releases.

3.2.2 LPCOpen

The USB audio solution is based on the LPCOpen package for LPC5411x MCUs. The LPCOpen package provides chip and board level drivers and system start-up support that is used with the USB audio application. To download the latest LPCOpen package for the LPC54114 audio and voice recognition Kit visit the link:

<https://www.lpcware.com/content/nxpfiler/lpcopen-software-development-platform-lpc5411x-packages>

The specific LPCOpen drivers and support code used with this application are:

- Chip layer USB ROM driver API (lpc_chip_5411x)
 - Redirects USB calls to use ROM code.
- Power library (lpc_chip_5411x)
 - Low power library functions.
- Flexcomm Interface driver (lpc_chip_5411x)
 - Used for initial Flexcomm Interface setup for I²C, I²S, and UART.
- I²C master driver (lpc_chip_5411x)
 - Used for CODEC control interface.
- I²S driver (lpc_chip_5411x)
 - Used for I²S setup, control, and audio data transfer.
- UART driver (lpc_chip_5411x)
 - Used for debug output support.
- Board layer SystemInit() code (lpc_board_lpcxpresso_54114)
 - Board pin muxing setup, initial system clock setup, UART (debug channel), LEDs.

The USB audio application provides the following capabilities:

- Startup code
 - Initial system startup code entry point, interrupt vectors, C runtime initialization.
- USB descriptors, audio class initialization, and data handling functions.

- Audio CODEC driver
 - Support for the Wolfson WM8904.
- I²C control functions for the audio CODEC.
- I²S setup, control, and data handling.
- Application startup and initialization (main.c)
- Background tasks
 - Audio CODEC volume level and mute control.
 - Power state control.

4. Software setup, build, and deployment

The USB audio application is released as part of the LPCOpen package for the LPC54114 audio and voice recognition Kit. LPCOpen packages are available for LPCXpresso and Keil/IAR tool chains. See *Section 3.2.2 LPCOpen* for information on where to download the latest LPCOpen package for the LPCXpresso LPC54114 board.

For LPCXpresso IDE, the project files are released as an LPCXpresso project archive file. This file can be directly imported into an LPCXpresso IDE workspace.

For Keil uVision and IAR Embedded Workbench, the project files are released as a zipped project structure file. This file should be unpacked somewhere on the host machine so the projects can be used with Keil uVision or IAR Embedded Workbench.

4.1 LPCXpresso IDE

This section describes how to import, build and deploy the application in LPCXpresso IDE.

4.1.1 Project import into the IDE

The USB audio project should be imported into a new LPCXpresso workspace. When starting LPCXpresso, it will request a workspace name. Select a name that does not already exist for the workspace. When LPCXpresso appears, the Project Explorer panel should be empty (no projects).

To import the USB audio project into the workspace, select 'Import project(s)' from the Quickstart panel. This will start the project import dialog.

Click the 'Browse' button next to the 'Archive' text box and locate the project archive file that was included with this application note. The project archive file will have a 'ZIP' extension. Press the 'Next' button to continue the dialog.

Select all the projects by checking the box next to them in the project list and then select the 'Finish' button. This will start the project import process into the workspace and close the dialog.

After import, you should have multiple projects that are viewable in the Project Explorer panel. The USB Audio project is called 'usbdrum_audio'. Note that other projects not related to USB audio may also be in the project list.

4.1.2 Building the project

The project, by default, will be at the 'Debug' build configuration after import into LPCXpresso IDE. This configuration is suited for debugging. Prior to building the project, it is recommended to set all the project build configurations to the fully optimized 'Release' configuration. This can be done by selecting all the projects in the LPCXpresso IDE Project Explorer panel and then selecting 'Build Configurations->Set Active->Release' from the 'Project' menu.

To build the project, first select only the 'usbdrum_audio' project in the Project Explorer panel. In the Quickstart panel, press the 'Build usbdrum_audio' entry to start the build. The build process will first build the chip and board projects and then build the USB audio application.

4.1.3 Deploying the project

To program the LPCXpressoLPC54114 board with the USB Audio application image, select 'Debug' from the Quickstart panel. The image will be programmed into the board's flash memory and the debugger will start the code halting it at the main() entry point. Select 'Terminate' from the 'Run' menu to disconnect the debugger and allow the program to run normally.

4.2 Keil uVision

This section describes how to import, build and deploy the application in Keil uVision.

4.2.1 Open the project files in Keil uVision

To open the project files in Keil uVision, first locate the Keil uVision projects file area in the unpacked LPCOpen package. This is 'lpc5411x\prj_xpresso54114\keil' directory in the unpacked LPCOpen file area.

Double click the 'examples_usbdrom.uvmpw' file to open up the projects in Keil uVision. This will open up a number of projects related to USB ROM for Keil uVision. Note that other projects not related to USB audio may also be in the project list.

4.2.2 Building the project

Prior to building the USB Audio project, build the chip and board libraries of LPCOpen first. These built libraries are needed by the USB audio project.

To build the LPC5411x chip library:

1. Right click on the 'project_lib_chip_5411x' project in the Project viewer.
2. Select 'Set as active project'.
3. Select 'Build' from the 'Project' menu to build the chip library.

To build the LPCXpresso54114 board library:

1. Right click on the 'project_lib_board_lpcxpresso_54114' project in the Project viewer.
2. Select 'Set as active project'.
3. Select 'Build' from the 'Project' menu to build the board library.

To build the USB Audio application:

1. Right click on the 'project_usbdrom_audio' project in the Project viewer.
2. Select 'Set as active project'.
3. Select 'Build' from the 'Project' menu to build the application.

4.2.3 Deploying the project

To update the LPCXpresso54114 board with the USB Audio application image, select 'Download' from the 'Flash' menu with the 'project_usbdrom_audio' project selected as the active project in the Project viewer. The image will be programmed into the board's flash memory. Reset the board to start the program.

4.3 IAR Embedded Workbench for ARM

This section describes how to import, build and deploy the application in IAR EWARM.

4.3.1 Opening the multi-project file in IAR Embedded Workbench

To open the project files in IAR Embedded Workbench, first locate the IAR Embedded Workbench projects file area in the unpacked LPCOpen package. This is 'lpc5411x\prj_xpresso54114\iar' directory in the unpacked LPCOpen file area.

Double click the 'examples_usbdrom.eww' file to open up the projects in IAR Embedded Workbench. This will open up a number of projects related to USB ROM for IAR Embedded Workbench. Note that other projects not related to USB audio may also be in the project list.

4.3.2 Building the project

Prior to building the USB Audio project, the chip and board libraries of LPCOpen need to be built first. These built libraries are needed by the USB audio project.

To build the LPC5411x chip library:

1. Right click on the 'lib_chip_5411x' project in the workspace panel.
2. Select 'Set as active' to set the LPC5411x chip library as the active project.
3. Select 'Make' from the 'Project' menu to build the chip library.

To build the LPCXpresso54114 board library:

1. Right click on the 'lib_board_lpcxpresso_54114' project in the Workspace panel
2. Select 'Set as active' to set the LPC54114 board library as the active project.
3. Select 'Make' from the 'Project' menu to build the board library.

To build the USB Audio application:

1. Right click on the 'usbdrom_audio' project in the Workspace panel
2. Select 'Set as active' to set the USB audio project as the active project.
3. Select 'Build' from the 'Project' menu to build the application.

4.3.3 Deploying the project

To update the LPCXpresso54114 board with the USB Audio application image, select 'Download->Download active application' from the 'Project' menu with the 'usbdrom_audio' project selected as the active project in the Workspace panel. The image will be programmed into the board's flash memory. Reset the board to start the program.

5. Playing and recording audio

After the LPCXpresso54114 board has been updated with the Audio class application, it can be plugged into an audio host machine, such as, a Windows, Linux, or a MAC system to playback or record audio. Audio is always played back or recorded at a 48 KHz sample rate only in stereo with a 16-bit sample size.

The LPC54114 will enumerate over USB as a standard audio class device and become available on the host system as an USB audio playback or record device.

Using an application that can playback audio data, such as a media player, allows audio to be played on the LPC54114 board if it is selected on the audio host system as the playback device. Starting an audio record session will record data from the LPC54114 board if it is selected as the recording device. Both playback and record can occur simultaneously.

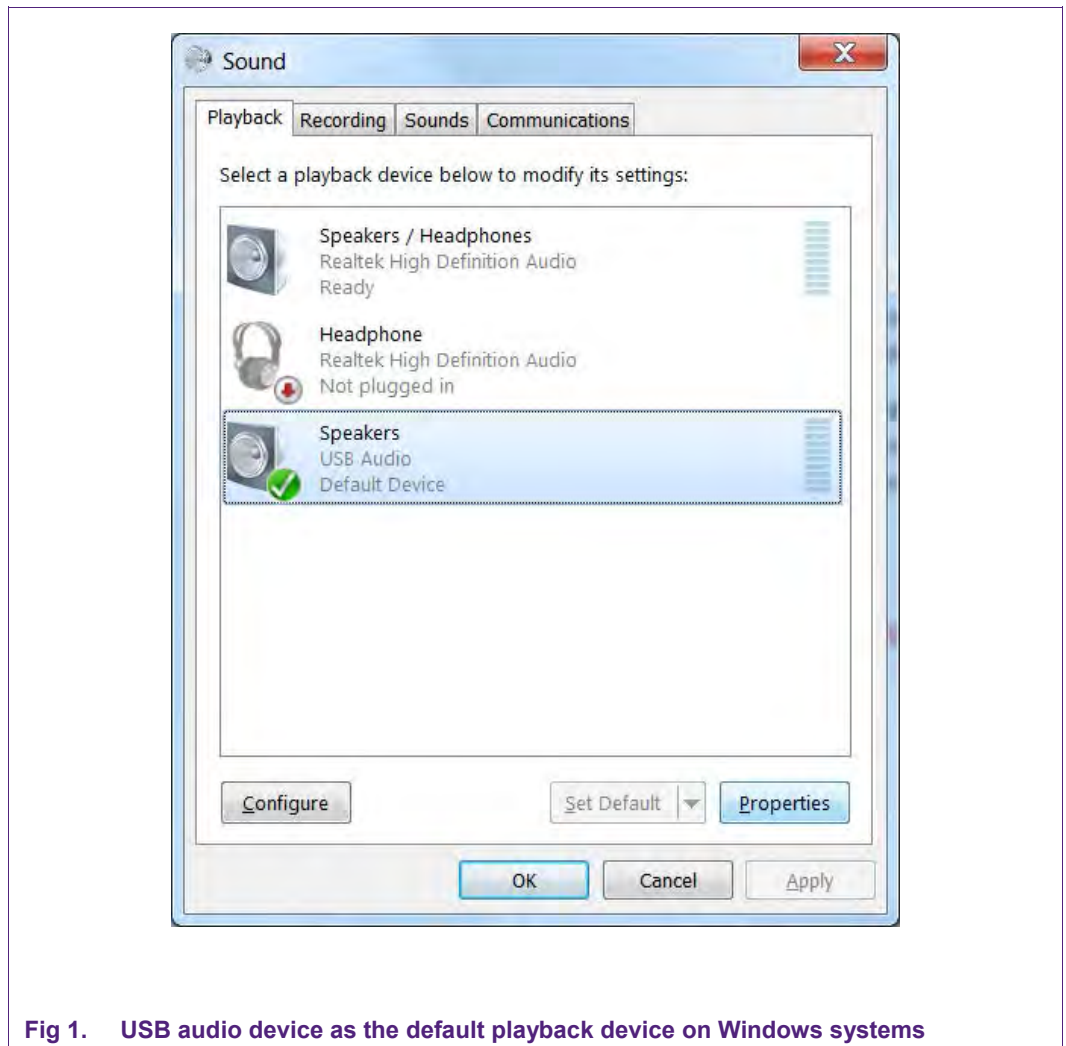


Fig 1. USB audio device as the default playback device on Windows systems

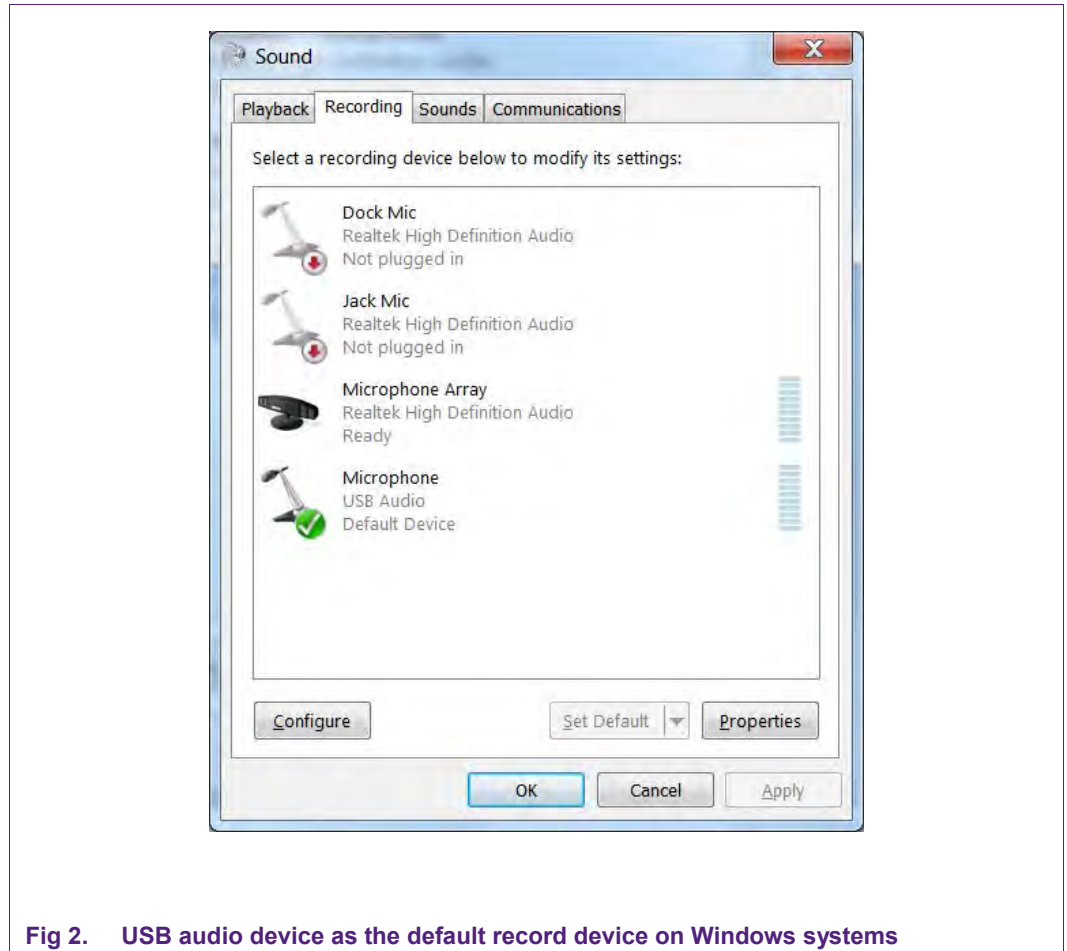


Fig 2. USB audio device as the default record device on Windows systems

5.1 Volume level and audio mute

The volume of the audio playback on the LPC54114 board can also be controlled with the standard volume control playback functions of the host system. Mute and unmute will also work. For audio record, the record level is controlled, similar to playback volume, through the standard volume control record functions of the host system.

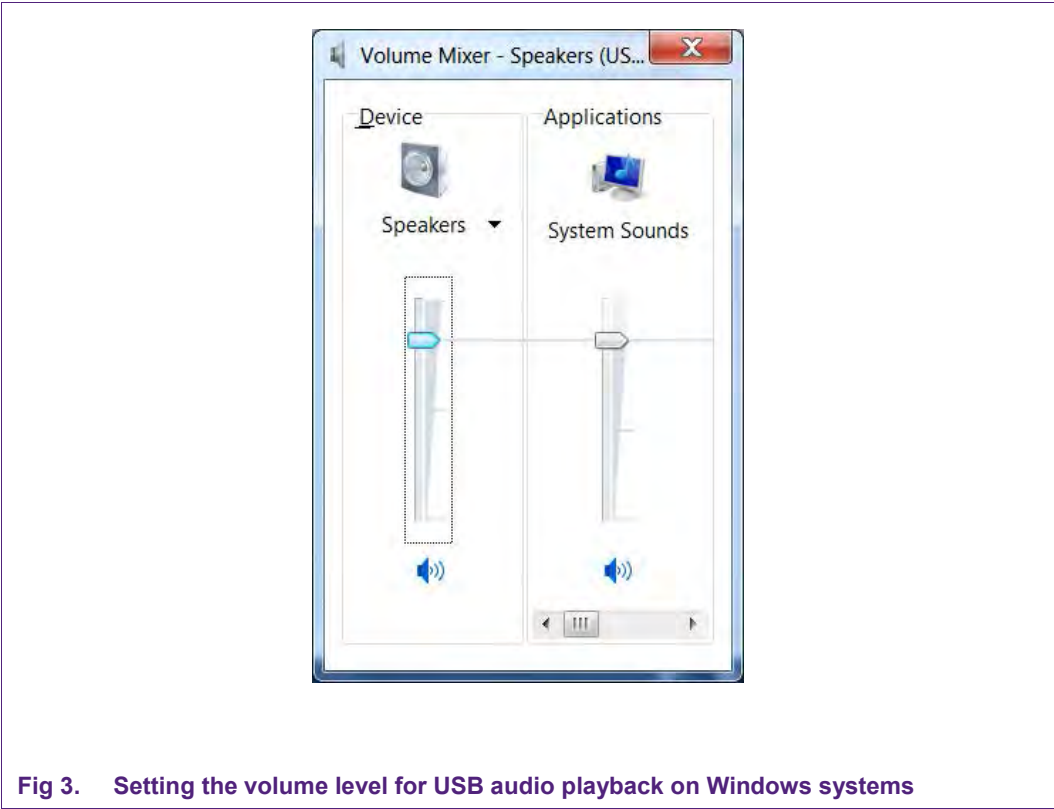


Fig 3. Setting the volume level for USB audio playback on Windows systems

6. Legal information

6.1 Definitions

Draft — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

6.2 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no

representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

6.3 Licenses

Purchase of NXP <xxx> components

<License statement text>

6.4 Patents

Notice is herewith given that the subject device uses one or more of the following patents and that each of these patents may have corresponding patents in other jurisdictions.

<Patent ID> — owned by <Company name>

6.5 Trademarks

Notice: All referenced brands, product names, service names and trademarks are property of their respective owners.

<Name> — is a trademark of NXP B.V.

7. List of figures

- Fig 1. USB Audio device as the default playback device on Windows systems 14
- Fig 2. USB Audio device as the default record device on Windows systems 15
- Fig 3. Setting the volume level for USB Audio playback on Windows systems 16

8. List of tables

Table 1. USB Audio application pin mapping5
Table 2. Peripherals and interfaces mapped to application function6

9. Contents

Document information	1	4.3.1	Opening the multi-project file in IAR Embedded Workbench.....	13
1. Introduction	3	4.3.2	Building the project.....	13
1.1 Highlights of the USB audio application	3	4.3.3	Deploying the project.....	13
2. Software Design	4	5. Playing and recording audio	14	
2.1 Overview of how the USB Audio application works.....	4	5.1	Volume level and audio mute	15
2.1.1 Audio sample clocks and buffering.....	4	6. Legal information	17	
2.1.2 I ² S and CODEC	5	6.1	Definitions.....	17
2.1.3 Background tasks.....	5	6.2	Disclaimers.....	17
2.2 System initialization and setup	5	6.3	Licenses	17
2.2.1 Pin setup	5	6.4	Patents	17
2.2.2 Peripheral functions	6	6.5	Trademarks	17
3. Environment	7	7. List of figures	18	
3.1 Hardware.....	7	8. List of tables	19	
3.1.1 Board set.....	7	9. Contents	20	
3.1.2 Debugger	7			
3.1.3 Miscellaneous	7			
3.1.4 Board setup.....	7			
3.1.5 USB connections.....	8			
3.1.6 Audio interface connections	8			
3.2 Software	8			
3.2.1 Supported development tool chains	9			
3.2.2 LPCOpen	9			
4. Software setup, build, and deployment	11			
4.1 LPCXpresso IDE	11			
4.1.1 Project import into the IDE	11			
4.1.2 Building the project.....	11			
4.1.3 Deploying the project	12			
4.2 Keil uVision	12			
4.2.1 Open the project files in Keil uVision	12			
4.2.2 Building the project.....	12			
4.2.3 Deploying the project	12			
4.3 IAR Embedded Workbench for ARM.....	12			

Please be aware that important notices concerning this document and the product(s) described herein, have been included in the section 'Legal information'.

© NXP B.V. 2016.

All rights reserved.

For more information, please visit: <http://www.nxp.com>
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 20 May 2016

Document identifier: AN11819

© NXP B.V. 2016. All rights reserved.