dsPIC® DSC Equalizer Library User's Guide

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE**.** Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights.

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
ISO/TS 16949:2009

Trademarks

♻ Printed on recycled paper.

ISBN: 978-1-61341-372-2

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

Table of Contents

NOTES:

Preface

NOTICE TO CUSTOMERS

All documentation becomes dated, and this manual is no exception. Microchip tools and documentation are constantly evolving to meet customer needs, so some actual dialogs and/or tool descriptions may differ from those in this document. Please refer to our web site (www.microchip.com) to obtain the latest documentation available.

Documents are identified with a “DS” number. This number is located on the bottom of each page, in front of the page number. The numbering convention for the DS number is “DSXXXXXA”, where “XXXXX” is the document number and “A” is the revision level of the document.

For the most up-to-date information on development tools, see the MPLAB® IDE on-line help. Select the Help menu, and then Topics to open a list of available on-line help files.

INTRODUCTION

This chapter contains general information that will be useful to know before using the dsPIC® DSC Equalizer Library. Items discussed in this chapter include:

- Document Layout
- Conventions Used in this Guide
- Warranty Registration
- Recommended Reading
- The Microchip Web Site
- Development Systems Customer Change Notification Service
- Customer Support
- Document Revision History

DOCUMENT LAYOUT

This user’s guide describes how to use the dsPIC DSC Equalizer Library. The document is organized as follows:

- **Chapter 1. “Introduction”** – This chapter introduces the dsPIC DSC Equalizer Library and provides a brief overview of equalization and the library features. It also outlines requirements for a host PC.
- **Chapter 2. “Installation”** – This chapter provides instructions for installing the library files and describes the contents of the source files, include files, demo files and archive files.
- **Chapter 3. “Equalizer Demonstration”** – This chapter provides a hands-on demonstration of equalization in a working application.
- **Chapter 4. “Application Programming Interface (API)”** – This chapter outlines how the API functions provided in the dsPIC DSC Equalizer Library can be included in your application software via the Application Programming Interface.

CONVENTIONS USED IN THIS GUIDE

This manual uses the following documentation conventions:

DOCUMENTATION CONVENTIONS

Description	Represents	Examples
Arial font:		
Italic characters	Referenced books	*MPLAB® IDE User's Guide*
	Emphasized text	...is the *only* compiler...
Initial caps	A window	the Output window
	A dialog	the Settings dialog
	A menu selection	select Enable Programmer
Quotes	A field name in a window or dialog	"Save project before build"
Underlined, italic text with right angle bracket	A menu path	*File>Save*
Bold characters	A dialog button	Click **OK**
	A tab	Click the **Power** tab
N'Rnnnn	A number in verilog format, where N is the total number of digits, R is the radix and n is a digit.	4'b0010, 2'hF1
Text in angle brackets < >	A key on the keyboard	Press <Enter>, <F1>
Courier New font:		
Plain Courier New	Sample source code	`#define START`
	Filenames	`autoexec.bat`
	File paths	`c:\mcc18\h`
	Keywords	`_asm, _endasm, static`
	Command-line options	`-Opa+, -Opa-`
	Bit values	`0, 1`
	Constants	`0xFF, 'A'`
Italic Courier New	A variable argument	*`file`*`.o`, where *`file`* can be any valid filename
Square brackets [ ]	Optional arguments	`mcc18 [options]` *`file`* `[options]`
Curly brackets and pipe character: { | }	Choice of mutually exclusive arguments; an OR selection	`errorlevel {0|1}`
Ellipses...	Replaces repeated text	`var_name [, var_name...]`

	Represents code supplied by user	`void main (void)`
		`{ ...`
		`}`

WARRANTY REGISTRATION

Please complete the enclosed Warranty Registration Card and mail it promptly. Sending in the Warranty Registration Card entitles users to receive new product updates. Interim software releases are available at the Microchip web site.

RECOMMENDED READING

This user's guide describes how to use the dsPIC DSC Equalizer Library. Other useful documents include:

dsPIC® DSC Acoustic Echo Cancellation Library User's Guide (DS70134)

This manual provides information you can use to incorporate acoustic echo cancellation capability into your embedded solution using the dsPIC® DSC Acoustic Echo Cancellation Library.

dsPIC® DSC Line Echo Cancellation Library User's Guide (DS70170)

This manual provides information you can use to incorporate line echo cancellation capability into your embedded solution using the dsPIC® DSC Line Echo Cancellation Library.

dsPIC® DSC Noise Suppression Library User's Guide (DS70133)

This manual provides information you can use to incorporate noise suppression capability into your embedded solution using the dsPIC DSC® Noise Suppression Library.

dsPIC30F Family Reference Manual (DS70046)

Refer to this document for detailed information on dsPIC30F device operation. This reference manual explains the operation of the dsPIC30F DSC family architecture and peripheral modules but does not cover the specifics of each device. Refer to the appropriate device data sheet for device-specific information.

dsPIC33F/PIC24H Family Reference Manual Sections

Refer to these documents for detailed information on dsPIC33F/PIC24H device operation. These reference manual sections explain the operation of the dsPIC33F/PIC24H DSC and MCU family architecture and peripheral modules, but do not cover the specifics of each device. Refer to the appropriate device data sheet for device-specific information.

dsPIC33E/PIC24E Family Reference Manual Sections

Refer to this documents for detailed information on dsPIC33E/PIC24E device operation. These reference manual sections explain the operation of the dsPIC33E/PIC24E DSC and MCU family architecture and peripheral modules, but do not cover the specifics of each device. Refer to the specific device data sheet for device-specific information.

16-bit MCU and DSC Programmer's Reference Manual (DS70157)

This manual is a software developer's reference for the dsPIC30F and dsPIC33F 16-bit MCU families of devices. It describes the instruction set in detail and also provides general information to assist in developing software for the dsPIC30F and dsPIC33F DSC families.

MPLAB® Assembler, Linker and Utilities for PIC24 MCUs and dsPIC® DSCs User's Guide (DS51317)

MPLAB Assembler for PIC24 MCUs and dsPIC® DSCs (formerly MPLAB ASM30) produces relocatable machine code from symbolic assembly language for the dsPIC DSC and PIC24 MCU device families. The assembler is a Windows console application that provides a platform for developing assembly language code. The assembler is a port of the GNU assembler from the Free Software Foundation (www.fsf.org).

MPLAB® C Compiler for PIC24 MCUs and dsPIC® DSCs User's Guide (DS51284)

This document describes the features of the optimizing C compiler, including how it works with the assembler and linker. The assembler and linker are discussed in detail, in the *"MPLAB® Assembler, Linker and Utilities for PIC24 MCUs and dsPIC® DSCs User's Guide"* (DS51317).

MPLAB® IDE Simulator, Editor User's Guide (DS51025)

Refer to this document for more information pertaining to the installation and implementation of the MPLAB Integrated Development Environment (IDE) Software.

To obtain any of these documents, visit the Microchip web site at www.microchip.com.

Microsoft® Windows® Manuals

This user's guide assumes that you are familiar with the Microsoft Windows operating system. Many excellent references exist for this software program and should be referenced for general operation of Windows.

THE MICROCHIP WEB SITE

Microchip provides online support via our web site at www.microchip.com. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user’s guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

DEVELOPMENT SYSTEMS CUSTOMER CHANGE NOTIFICATION SERVICE

Microchip’s customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at www.microchip.com, click on Customer Change Notification and follow the registration instructions.

The Development Systems product group categories are:

- **Compilers** – The latest information on Microchip C compilers and other language tools. These include the MPLAB® C compiler; MPASM™ and MPLAB 16-bit assemblers; MPLINK™ and MPLAB 16-bit object linkers; and MPLIB™ and MPLAB 16-bit object librarians.
- **Emulators** – The latest information on Microchip in-circuit emulators. This includes the MPLAB ICE 2000, MPLAB ICE 4000 and MPLAB REAL ICE™.
- **In-Circuit Debuggers** – The latest information on the Microchip in-circuit debuggers, MPLAB ICD 2 and MPLAB ICD 3.
- **MPLAB IDE** – The latest information on Microchip MPLAB IDE, the Windows® Integrated Development Environment for development systems tools. This list is focused on the MPLAB IDE, MPLAB SIM simulator, MPLAB IDE Project Manager and general editing and debugging features.
- **Programmers** – The latest information on Microchip programmers. These include the MPLAB PM3 and PRO MATE® II device programmers and the PICSTART® Plus and PICkit™ 1, 2 and 3 development programmers.

CUSTOMER SUPPORT

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: http://support.microchip.com

DOCUMENT REVISION HISTORY

Revision A (November 2008)

This is the initial released version of this document.

Revision B (July 2011)

This revision includes the following updates:

- Updated the first paragraph in **Chapter 1. "Introduction"**, which now includes references to the dsPIC33E family of devices
- Revised **Chapter 2. "Installation"**
- Updated Table 2-1 and Table 2-3 in **2.2 "Equalizer Library Files"**
- Updated **Chapter 3. "Equalizer Demonstration"**, which now distinguishes between dsPIC33F and dsPIC33E devices (see **3.1 "Equalizer Demonstration for the dsPIC33F Device Family"** and **3.2 "Equalizer Demonstration for the dsPIC33E Device Family"**)
- Revised **4.1 "Adding the Equalizer Library to an Application"** in **Chapter 4. "Application Programming Interface (API)"**
- Updated all tables in **4.4 "Resource Requirements"**
- Updates to formatting and minor text changes have been incorporated throughout the document

Chapter 1. Introduction

This chapter introduces the dsPIC DSC Equalizer Library. This library, which supports the dsPIC33F and dsPIC33E families of devices, provides functionality to adjust the spectral characteristics of a voice band signal. This manual provides information you can use to incorporate the Equalizer (EQ) library in your embedded solution. Topics covered include:

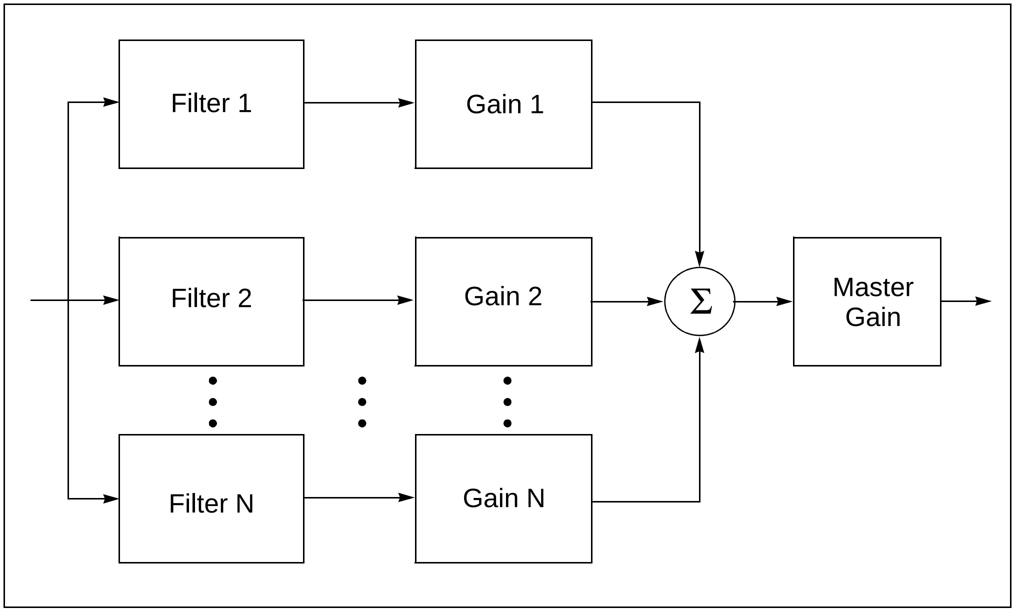
- Equalizer Overview
- Features
- Host System Requirements

1.1 EQUALIZER OVERVIEW

An Equalizer is a system that allows the spectral characteristics of a signal to be changed. Figure 1-1 shows the conceptual block diagram of how this is achieved.

In speech and audio applications, signal processing functions and related operations may suppress or boost certain frequencies of a signal. This manifests as a change in the tonal properties of the output signal as compared to the input. Additionally, input and output devices (microphones, speakers, etc.) may emphasize or de-emphasize certain frequencies in a signal due to their mechanical characteristics and limitations. An Equalizer enables compensation for these changes by providing the user the ability to modify the spectral characteristics of the signal.

FIGURE 1-1: EQUALIZER CONCEPTUAL BLOCK DIAGRAM



The equalizer splits the input signal into different frequency bands (component signals) using a set of band-pass filters. The center frequency for these filters is fixed. The filter output signal gain is specified by the application. The component signals are then summed up, amplified by a user-specified gain (master gain) and then provided as the output signal.

The EQ library uses a fixed 8 kHz sampling rate and is especially suitable for applications such as:

- Hands-free cell phone kits
- Speaker phones
- Intercoms
- Teleconferencing systems

The EQ library is written almost entirely in assembly and is highly optimized to make extensive use of the dsPIC DSC device instruction set and advanced addressing modes. The EQ library provides an `EQ_init()` function for initializing the various data structures required by the algorithm and an `EQ_apply()` function to equalize the signal. You can easily call both functions through a well documented Application Programmer's Interface (API).

1.2 FEATURES

Key features of the EQ library include:

- Simple user interface – only one library file and one header file
- All functions can be called from a C application program
- Compatible with the Microchip C30 Compiler, Assembler and Linker
- Highly optimized assembly code that uses DSP instructions and advanced addressing modes
- Eight EQ bands centered at 31, 62, 125, 250, 500, 1000, 2000 and 4000 Hz
- Quality factor of 1.4 for each band-pass filter
- Individual band gain control from 0 to -18 dB
- Master gain control from 0 to 12 dB
- Can be integrated with the dsPIC DSC Noise Suppression, Acoustic Echo Cancellation (AEC), and Line Echo Cancellation (LEC) libraries
- Demo application source code is provided with the library
- Can process multiple audio streams
- Does not require scratch memory
- Run time control of key algorithm parameters is provided

1.3 HOST SYSTEM REQUIREMENTS

The EQ Library requires a PC-compatible system with these attributes:

- Intel® Pentium® class or higher processor, or equivalent
- HTML browser
- 16 MB RAM (minimum)
- 40 MB available hard drive space (minimum)
- Microsoft® Windows® 98, Windows 2000, Windows NT, Windows XP, Windows Vista or Windows 7

Chapter 2. Installation

This chapter describes the various files in the Equalizer (EQ) Library and includes instructions for installing the EQ Library on your laptop or PC for use with dsPIC DSC device programming tools. Topics covered include:

- Installation Procedure
- Equalizer Library Files

2.1 INSTALLATION PROCEDURE

Use the following procedure to install the library:

1. Double click `EQ setup.exe`. The license agreement appears in a new window.
2. Review the license agreement and click **I Agree** to continue. The Installation Destination dialog appears.
3. Specify the location (i.e., directory) where the library is to be installed, and then click **Install**.
4. Click **Close** to close the dialog. This completes the EQ library installation.

The installation process creates the folder named `EQ v2.0`, which contains the files described in **Section 2.2 "Equalizer Library Files"**.

2.2 EQUALIZER LIBRARY FILES

The dsPIC DSC Equalizer Library CD creates a directory labeled `EQ v2.0`. This directory contains these five folders:

- `demo`
- `doc`
- `h`
- `lib`
- `wavefiles`

2.2.1 demo Folder

This folder contains files that are required by the EQ Library Quick Start Demonstration. Table 2-1 describes the files in this folders.

TABLE 2-1: DEMONSTRATION FILES

File Name	Description
`dsPIC33F EQ demo.hex`	Demonstration hexadecimal file for dsPIC33F.
`dsPIC33E EQ demo.hex`	Demonstration hexadecimal file for dsPIC33E.
`dsPIC33F EQ demo.mcp`	Demonstration MPLAB® IDE Project file for dsPIC33F.
`dsPIC33E EQ demo.mcp`	Demonstration MPLAB IDE Project file for dsPIC33E.
`cleanup.bat`	A batch file script for cleaning the intermediate build files.
`h\dsPICDEM1_1Plus.h`	C header file for the dsPICDEM™ 1.1 Plus Development Board routines.
`h\MEB.h`	C header file for the Multimedia Expansion Board (MEB) routines.
`h\lcd.h`	C header file defining the interface to the LCD driver.
`h\eq_api.h`	C header file defining the interface to the EQ Library.
`h\SI3000Drv.h`	C header file defining the interface to the Si3000 Codec Driver.
`h\WM8731CodecDrv.h`	C header file defining the interface to the WM8731 Codec driver.
`libs\eqlibv2_33F.a`	The EQ Library archive file for dsPIC33F.
`libs\eqlibv2_33E.a`	The EQ Library archive file for dsPIC33E.
`src\dsPICDEM1_1Plus.c`	C source files containing routines for the dsPICDEM1.1 Plus Development Board.
`src\MEB.c`	C source file containing routines for the MEB.
`src\lcd_strings.c`	C source file for the LCD display driver.
`src\main.c`	C source file containing the main speech processing routine.
`src\SI3000Drv.c`	C source file containing the code for the Si3000 Codec.
`src\WM8731CodecDrv.c`	C source file containing the code for the WM8731 Codec.
`src\lcd.s`	Assembly routines for communicating with the LCD controller.

2.2.2 doc Folder

This folder contains the electronic user's guide for the dsPIC DSC Equalizer Library. To view this document, double click the file name. The user's guide can also be downloaded from the Microchip web site (www.microchip.com).

2.2.3 h Folder

This folder contains an include file for the EQ Library as listed in Table 2-2.

TABLE 2-2: INCLUDE FILE

File Name	Description
eq_api.h	Include file that contains the interface to the EQ Library. This file must be included in the application to use the library.

2.2.4 lib Folder

This folder contains a library archive file for the EQ Library as listed in Table 2-3. The archive names are suffixed with the names of the target device families, 33F or 33E.

TABLE 2-3: LIBRARY FILE

File Name	Description
eqlibv2_33F.a	This is the EQ Library archive file for dsPIC33F. This file must be included in the application in order to use the library.
eqlibv2_33E.a	This is the EQ Library archive file for dsPIC33E. This file must be included in the application in order to use the library.

2.2.5 wavefiles Folder

This folder contains sample WAVE files that can be used with the EQ demonstration example or for prototyping. The available WAVE files are listed in Table 2-4.

TABLE 2-4: WAVE FILES

File Name	Description
highbass1.wav	Sample WAVE file with emphasized bass frequencies.
highmids.wav	Sample WAVE file with emphasized mid-range frequencies.
hightreble1.wav	Sample WAVE file with emphasized treble frequencies.

NOTES:

Chapter 3. Equalizer Demonstration

This chapter provides a hands-on demonstration of the Equalizer (EQ) library in a working application.

3.1 EQUALIZER DEMONSTRATION FOR THE dsPIC33F DEVICE FAMILY

The following topics are covered in this section:

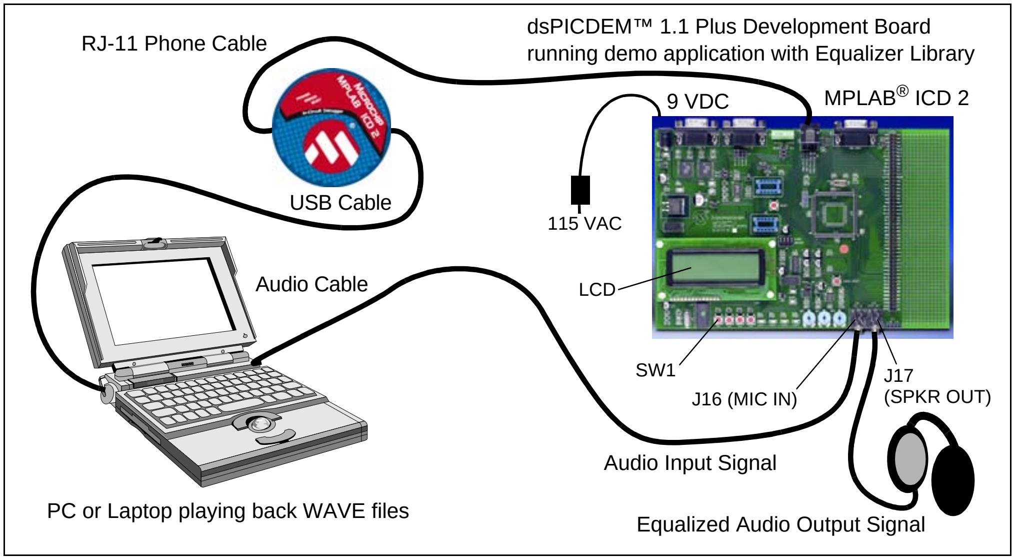
- Demonstration Summary
- Demonstration Setup
- Demonstration Procedure
- Demonstration Code Description

3.1.1 Demonstration Summary

A demonstration application program included with the EQ library demonstrates the functionality of the library when used with a dsPIC33F Digital Signal Controller (DSC). In the demonstration setup (illustrated in Figure 3-1), a dsPICDEM™ 1.1 Plus Development Board is configured as a system that receives a speech signal through its microphone input port, equalizes the sampled signal and plays out an equalized signal through the speaker output port. The on-board Si3000 codec is used as the microphone and speaker interface.

A PC is used to drive speech signals through an audio cable from the PC's Speaker Out port to J16 (MIC IN) on the dsPICDEM 1.1 Plus Development Board. A headset or speaker is connected to J17 (SPKR OUT) on the dsPICDEM 1.1 Plus Development Board.

FIGURE 3-1: dsPIC33F EQUALIZER DEMONSTRATION SETUP


RJ-11 Phone Cable
dsPICDEM™ 1.1 Plus Development Board
running demo application with Equalizer Library
9 VDC
MPLAB® ICD 2
USB Cable
115 VAC
Audio Cable
LCD
SW1
J16 (MIC IN)
J17
(SPKR OUT)
Audio Input Signal
PC or Laptop playing back WAVE files
Equalized Audio Output Signal

You can use the `.wav` files provided with the demo (in the `wavefiles` folder of the installation directory) as unequalized speech signals, or you can provide your own signals. The unequalized input signal is captured by the on-board Si3000 voice band codec and the Data Converter Interface (DCI) module of the dsPIC DSC device.

The dsPIC DSC device then plays out the processed (equalized) signal through the device's DCI module and the on-board Si3000 codec. When started, the program initializes with the Equalizer turned OFF, indicated by LED1 turned OFF and OFF being written to the LCD screen. With the Equalizer off, the signal heard in the headset is the same as the input signal.

The Equalizer is enabled by pressing SW1. LED1 is now turned ON and ON is written to the LCD. Depending on the Equalizer settings, the speech signal heard on the microphone will sound different. Use SW2 to change the EQ band of interest. Use SW3 and SW4 to change the EQ levels.

Switch on the Repeat function in the Media Player on your PC to allow the `.wav` file to run continuously, and then observe when the signal is not equalized and when it is equalized. Repeat the process with several `.wav` files.

> **Note:** Some media players insert a break before each repeat of the `.wav` file. If you want to avoid this, a sound editor program, such as Audacity, can provide for continuous looping. Audacity, which is a free, cross-platform sound editor, is available from http://audacity.sourceforge.net/.

3.1.2 Demonstration Setup

The demo application is intended to run with a dsPIC33F device on a dsPICDEM 1.1 Plus development board (not included with the software license).

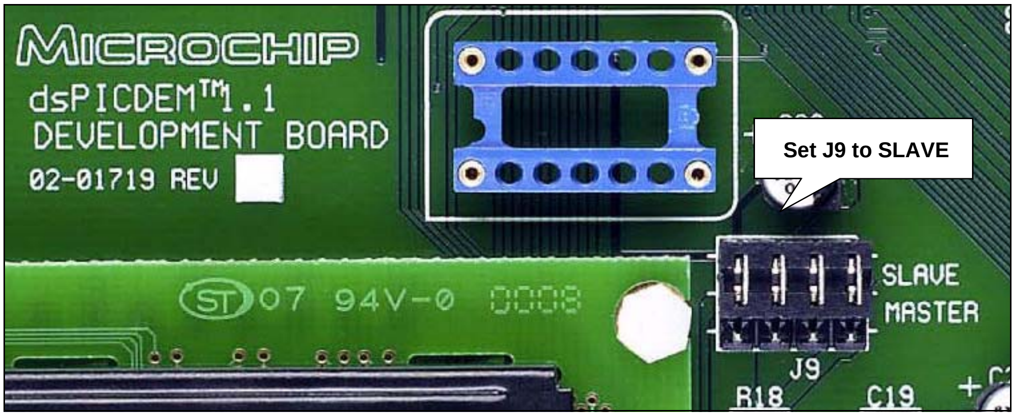
Use the procedures outlined in the following sections to set up the demonstration.

3.1.2.1 CONFIGURE THE dsPICDEM 1.1 PLUS DEVELOPMENT BOARD

Before applying power, configure the board as follows:

1. Set jumper J9 (adjacent to the oscillator socket) to the **SLAVE** position (see Figure 3-2). This setting allows the on-board Si3000 codec chip to function as a serial clock Slave.
2. Connect the audio cable between the speaker out port on the PC and the MIC IN jack (J16) on the dsPICDEM 1.1 Plus Development Board.
3. Connect a headset or speaker to the SPKR OUT jack (J17).
4. Connect the MPLAB ICD 2 between the PC (USB cable) and the dsPICDEM 1.1 Plus Development Board (RJ-11 phone cable).
5. Connect the 9V power supply to power-up the dsPICDEM 1.1 Plus Development Board.

FIGURE 3-2: DEMONSTRATION BOARD SETUP



> **Note:** MPLAB REAL ICE can be used in place of MPLAB ICD 3.

3.1.2.2 SET UP THE DEMONSTRATION

After the board is configured correctly, the setup should resemble Figure 3-1.

3.1.2.3 PROGRAM THE dsPIC DSC DEVICE

Use this process to load the equalizer demo onto the dsPIC DSC device on the dsPICDEM 1.1 Plus development board.

1. On your PC, launch MPLAB IDE and open the `dsPIC33F EQ demo.mcp` project located in the `demo` folder. For more information on using MPLAB IDE, refer to the *“MPLAB® IDE User’s Guide”* (DS51025).
2. Import the project hexadecimal file: *File>Import>dsPIC33F EQ demo.hex*
3. Select *Programmer>Connect* to link the MPLAB ICD 3 to the dsPIC DSC device target. The Output window shows that the MPLAB ICD 3 is ready.
4. Select *Programmer>Program*. The Output window displays the download process and indicates that the programming has succeeded.
5. When the program is loaded, disconnect the MPLAB ICD 3 from the board (remove the phone cable from the MPLAB ICD 3 connector). When you have done this, you will see the Equalizer information in the LCD display.

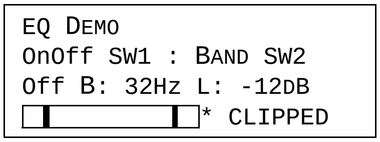
3.1.3 Demonstration Procedure

With the demo application programmed into the device, the demonstration is ready to run. You can use the provided `.wav` files, which are located in the `wavefiles` folder of the installation directory as unequalized speech signals, or you can provide your own signals.

The unequalized input signal is sampled through the on-board Si3000 voice band codec and the Data Converter Interface (DCI) module of the dsPIC DSC device. The dsPIC DSC device then plays out the processed (equalized) signal through the device's DCI module and the on-board Si3000 codec.

The demo application relays the state of operation via the LEDs and the LCD. While the application is loading and initializing the on-chip and off-chip peripherals, a boot screen appears, which then switches automatically to the run-time screen as shown in Figure 3-3.

FIGURE 3-3: DEMONSTRATION RUN-TIME LCD SCREEN



```
EQ DEMO
OnOff SW1 : BAND SW2
Off B: 32Hz L: -12DB
               * CLIPPED
```

The run-time screen displays the following:

1. The name of the algorithm.
2. SW1 is used to turn the Equalizer ON and OFF. SW2 changes the band to be modified. SW3 and SW4 can be used to change the gain for the selected band.
3. The current state of the algorithm, selected band (B) and the current gain level (L) for the selected band.
4. A VU meter showing the input level.

The individual band gain level can be changed in decrements of -1 dB from 0 through -18 dB. The master gain can be changed from 0 to 12 dB in increments of 1 dB.

When started, the program initializes with the Equalizer turned OFF, indicated by LED1 turned OFF and OFF displayed on the LCD. With the Equalizer off, the signal heard in the headset is the same as the input signal.

The Equalizer is enabled by pressing SW1. LED1 is now turned ON and ON is displayed on the LCD and the speech signal heard on the headset is equalized

Turn on the Repeat function in your PC's media player to allow the `.wav` file to run continuously. Then observe when the signal is equalized and when it is not. To experiment with the effect of different frequencies, play several of the `.wav` files provided in the `wavefiles` folder.

3.1.4 Demonstration Code Description

The demonstration code runs on a dsPIC33F device, using the Primary Oscillator as the clock source with the PLL set for 40 MIPS operation.

The file, `main.c`, contains the main function for the demo application. This main function allocates all the variables and arrays in data memory that are needed for DCI data buffering, as well as the blocks of data memory that need to be allocated for the EQ Library functions.

The main function calls the `EQ_init()` function from the EQ Library, which initializes the EQ algorithm to its default state.

The main function also calls the `SI3000_open()` function to initialize the DCI module, the Si3000 codec, and the DCI interrupt. The DCI module acts as a Master and drives the serial clock and frame synchronization lines. The Si3000 codec acts as a Slave. The DCI module is set for the multi-channel Frame Sync Operating mode, with 16-bit data words and 16 data words or time slots per frame, of which only one transmit slot and one receive slot are used in this demonstration.

Subsequently, this function initializes the Si3000 codec. The codec is reset, by connecting the RF6 pin of the dsPIC DSC device to the Reset pin of the Si3000, holding RF6 low for 100 cycles and then bringing it high. The codec is configured for a sample rate of 8 kHz. The MIC Gain is set to 10 dB and the Receive Gain is set to 0 dB. Both speakers are set to Active and the Transmit Gain is set to 0 dB. The Analog Attenuation parameter is set to 0 dB. After initializing all of the Si3000 control registers, a delay is introduced for calibration of the Si3000 to occur. Finally, the DCI interrupt is enabled.

The main processing loop reads the Si3000 driver for a frame of speech data. This data is copied into the `sigIn` array and passed as an input to the `EQ_apply()` function. The function operates in place, that is, the output is stored in the `sigIn` array itself. The array is then written to the Si3000 driver for playback onto the headphones.

The display on the LCD is made possible by initialization of the SPI module in the `InitSPI` function, and LCD driver functions and LCD string definitions present in the `lcd.s` and `lcd_strings.c` files, respectively.

To toggle the Equalizer ON or OFF, external interrupts for SW1 are enabled. In the main loop, the value of the variable, `equalize`, is read and passed to `EQ_apply()` as the enable flag. If `equalize` is '`0`', the Equalizer is still called, but the input/output buffer is not changed.

3.2 EQUALIZER DEMONSTRATION FOR THE dsPIC33E DEVICE FAMILY

The following topics are covered in this section:

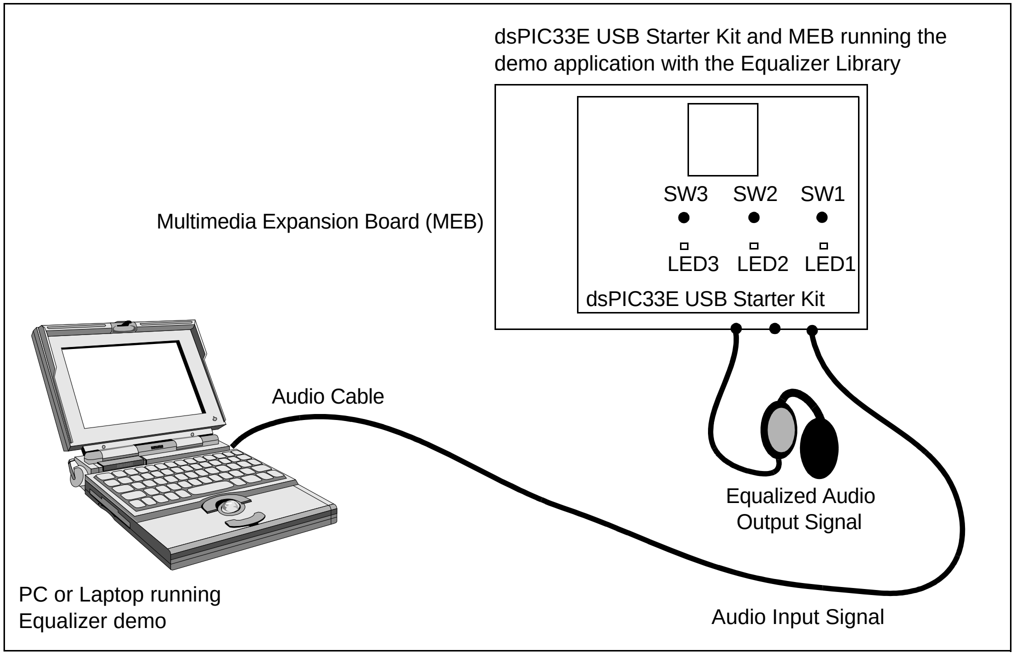
- Demonstration Summary
- Demonstration Setup
- Demonstration Procedure
- Demonstration Code Description

3.2.1 Demonstration Summary

A demonstration application program included with the EQ library demonstrates the functionality of the library when used with a dsPIC33E Digital Signal Controller (DSC). In the demonstration setup (illustrated in Figure 3-4), a Multimedia Expansion Board (MEB) in conjunction with a dsPIC33E USB Starter Kit is configured as a system that receives a speech signal through its microphone input port, equalizes the sampled signal and plays out an equalized signal through the speaker output port. The on-board WM8731 codec is used as the microphone and speaker interface.

A PC is used to drive speech signals through an audio cable from the PC’s Speaker Out port to the microphone input of the MEB. A headset or speaker is connected to the speaker output of the MEB.

FIGURE 3-4: dsPIC33E EQUALIZER DEMONSTRATION SETUP


dsPIC33E USB Starter Kit and MEB running the demo application with the Equalizer Library
Multimedia Expansion Board (MEB)
SW3
SW2
SW1
LED3
LED2
LED1
dsPIC33E USB Starter Kit
Audio Cable
Equalized Audio Output Signal
PC or Laptop running Equalizer demo
Audio Input Signal

You can use the `.wav` files provided with the demo (in the `wavefiles` folder of the installation directory) as unequalized speech signals, or you can provide your own signals. The unequalized input signal is captured by the on-board WM8731 audio codec and the Data Converter Interface (DCI) module of the dsPIC DSC device.

The dsPIC DSC device then outputs the processed (equalized) signal through the device's DCI module and the on-board WM8731 codec. When started, the program initializes with the Equalizer turned OFF. With the Equalizer off, the signal heard in the headset is the same as the input signal.

The Equalizer is enabled by pressing the switch, SW1, on the dsPIC33E USB Starter Kit. Depending on the Equalizer settings, the speech signal heard on the microphone will sound different. Use S1 on the LCD side of the MEB to change the EQ band of interest. Use SW2 and SW3 on the dsPIC33E USB Starter Kit to change the EQ levels.

Switch on the Repeat function in the Media Player on your PC to allow the .wav file to run continuously, and then observe when the signal is not equalized and when it is equalized. Repeat the process with several .wav files.

> **Note:** Some media players insert a break before each repeat of the .wav file. If you want to avoid this, a sound editor program, such as Audacity, can provide for continuous looping. Audacity, which is a free, cross-platform sound editor, is available from http://audacity.sourceforge.net/.

3.2.2 Demonstration Setup

The demo application is intended to run on an MEB and a dsPIC33E USB Starter Kit (not included with the software license).

Use the procedures outlined in the following sections to set up the demonstration.

3.2.2.1 CONFIGURE THE MEB AND dsPIC33E USB STARTER KIT

Before applying power, you need to configure the board:

1. Insert a dsPIC33E USB Starter Kit into the starter kit connector on the MEB.
2. Connect the audio cable between the Speaker Out port on the PC and the microphone jack (J7) on the MEB.
3. Connect a headset or speaker to the headphone jack (J8) of the MEB.
4. Connect the dsPIC33E USB Starter Kit to a PC using the USB A-to-Mini B cable provided with the Starter Kit.

3.2.2.2 SET UP THE DEMONSTRATION

After the board is configured correctly, the setup should resemble Figure 3-4.

3.2.2.3 PROGRAM THE dsPIC DSC DEVICE

Use this process to load the equalizer demo onto the dsPIC DSC device on the dsPIC33E USB Starter Kit.

1. On your PC, launch MPLAB IDE and open the dsPIC33E EQ demo.mcp project located in the demo folder. For more information on using MPLAB IDE, refer to the *"MPLAB® IDE User's Guide"* (DS51025).
2. Import the project hexadecimal file: *File>Import>dsPIC33E EQ demo.hex*
3. Choose **Starter Kit on Board** as the programmer, and then select *Programmer>Connect* to link to the dsPIC DSC device target. The Output window confirms that the target device is ready.
4. Select *Programmer>Program*. The Output window displays the download process and indicates that the programming has succeeded.

> **Note:** After programming, unplug and reconnect the USB cable to the dsPIC33E USB Starter Kit to ensure that the WM8731 audio codec can be reconfigured.

3.2.3 Demonstration Procedure

With the demo application programmed into the device, the demonstration is ready to run. You can use the provided .wav files, which are located in the wavefiles folder of the installation directory as unequalized speech signals, or you can provide your own signals.

The unequalized input signal is sampled through the on-board WM8731 audio codec and the Data Converter Interface (DCI) module of the dsPIC DSC device. The dsPIC DSC device then plays out the processed (equalized) signal through the device's DCI module and the on-board WM8731 codec.

When started, the program initializes with the Equalizer turned OFF, as indicated by the LED5 on the MEB being turned OFF. With the Equalizer off, the signal heard in the headset is the same as the input signal.

The Equalizer is enabled by pressing SW1 on the dsPIC33E USB Starter Kit. LED5 is now turned ON and the speech signal heard on the headset is equalized.

Use switch S1 on the LCD side of the MEB to change the audio spectral band that is to be modified by the Equalizer. The binary value of the current selection is represented by LED1 through LED4 on the MEB. Each button press increases the selected band by one, and band 8 can be used to change the master gain of the algorithm.

Pressing switch SW2 on the dsPIC33E USB Starter Kit increases the gain of the selected band, while pressing switch SW3 reduces the gain.

The individual band gain level can be changed in decrements of -1 dB from 0 through -18 dB. The master gain can be changed from 0 to 12 dB in increments of 1 dB.

Turn on the Repeat function in your PC's media player to allow the `.wav` file to run continuously. Then observe when the signal is equalized and when it is not. To experiment with the effect of different frequencies, play several of the `.wav` files provided in the `wavefiles` folder.

3.2.4 Demonstration Code Description

The demonstration code runs on a dsPIC33E device, using the Primary Oscillator as the clock source with the PLL set for 40 MIPS operation.

The file, `main.c`, contains the main function for the demo application. This main function allocates all the variables and arrays in data memory that are needed for DCI data buffering, as well as the blocks of data memory that need to be allocated for the EQ Library functions.

The main function calls the `EQ_init()` function from the EQ Library, which initializes the EQ algorithm to its default state.

The main function also calls the `WM8731Init()` function to initialize the DCI module, the WM8731 codec, and the DCI interrupt. The WM8731 codec acts as a Master and drives the serial clock and frame synchronization lines. The DCI module is set for the multi-channel Frame Sync Operating mode, with 16-bit data words and 2 data words or time slots per frame (i.e., two transmit slots and two receive slots).

Subsequently, the `WM8731Start()` function is used to enable the DCI and I^{2}C modules. The codec is configured for a sample rate of 8 kHz.

The main processing loop reads the WM8731 driver for a frame of speech data. This data is copied into the `sigIn` array and passed as an input to the `EQ_apply()` function. The function operates in place, that is, the output is stored in the `sigIn` array itself. The array is then written to the WM8731 driver for playback onto the headphones.

To toggle the Equalizer ON or OFF, a variable named `equalize` is used. In the main loop, the value of the variable, `equalize`, is read and passed to `EQ_apply()` as the enable flag. If `equalize` is '`0`', the Equalizer is still called, but the input/output buffer is not changed.

NOTES:

Chapter 4. Application Programming Interface (API)

This chapter describes in detail the Application Programming Interface (API) to the EQ library. Topics covered include:

- Adding the Equalizer Library to an Application
- Equalizer Band Control
- Library Usage
- Resource Requirements
- Equalizer Library API Functions
- Application Tips

4.1 ADDING THE EQUALIZER LIBRARY TO AN APPLICATION

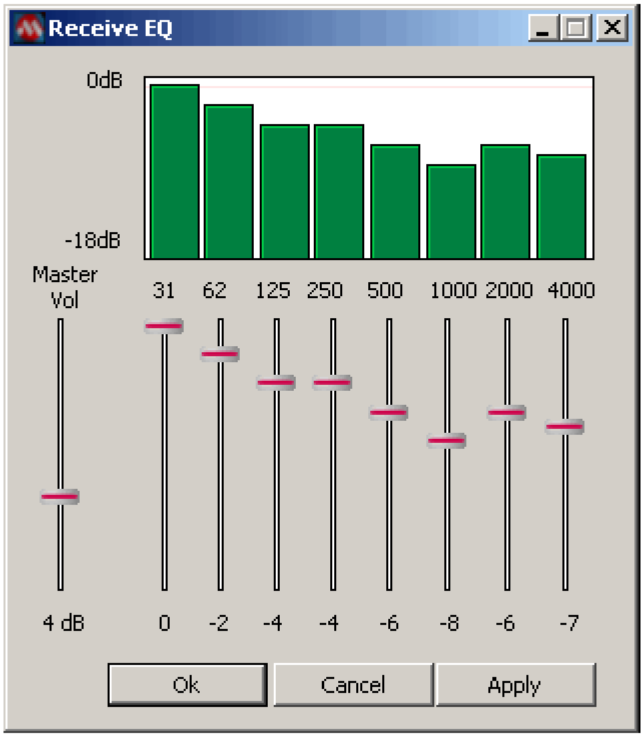
To use the EQ library in an application, the library archive must be added to the application project workspace and the file, `eq_api.h`, must be included in application code. Use following procedure to add the library to the application.

1. In the application MPLAB IDE Workspace, right click **Library Files** in the Project Window and select **Add files.**
2. Browse to the location of either the `eqlibv2_33F.a` or the `eqlibv2_33E.a` file (available in the `lib` folder in the installation directory).
3. Select the file and click **Open**.
4. The library is now added to the application.

4.2 EQUALIZER BAND CONTROL

Figure 4-1 shows a typical front-end software interface of an equalizer. The different frequency bands control the characteristics of the speech signal. A "muddy" or overly damped speech signal can be brightened by reducing the amplitude of the 31 Hz, 62 Hz and 125 Hz bands. A weak speech signal can be strengthened by increasing the amplitude of the 250 Hz, 500 Hz and 1000 Hz bands. If the speech signal is overly bright, hisses, or contains other high-frequency artifacts, the signal can be equalized by strengthening the 31 Hz, 62 Hz and 125 Hz bands. Alternately, the 2000 Hz and 4000 Hz bands could be attenuated.

FIGURE 4-1: EQUALIZER SOFTWARE INTERFACE



Master Vol	31	62	125	250	500	1000	2000	4000
4 dB	0	-2	-4	-4	-6	-8	-6	-7

4.3 LIBRARY USAGE

To use the library functions, include the file, `eq_api.h`, in the application source code. This file can be copied from the `h` folder (located in the installation directory) to the application project folder.

The Equalizer library has been designed to be usable in a re-entrant environment. This enables the algorithm to process many independent channels of audio, each channel having its own setting and parameters.

The following coding steps need to be performed to enable use of the Equalizer library. Example 4-1 shows the corresponding code.

1. **Allocate the memory for the EQ algorithm state holder:** The EQ state holder has two components, one in X memory and the other in Y memory. The state holder in X memory is an integer (int) type array of size `EQ_XSTATE_MEM_SIZE_INT` starting at an address boundary of two bytes. The state holder in Y memory is an int type array of size `EQ_YSTATE_MEM_SIZE_INT` starting at an address boundary of two bytes. Every audio channel to be processed will require its own state holder.

> **Note:** In some dsPIC33E devices, the Y memory is located in the Extended Data Space (EDS). In such cases, the Y scratch memory must be tagged with the `__eds__` keyword and assigned an EDS attribute.

2. **Create the frequency band gains array:** The gain of the individual frequency bands in the Equalizer is specified by a char type array of size 8. A value at index ‘n’ in this array corresponds to the gain of frequency band ‘n’. For example, `gains1[0]` corresponds to the gain of the 32 Hz band, `gains1[1]` corresponds to the gain of the 125 Hz band, and so on.
3. **Initialize the EQ algorithm for each audio channel:** Use the `EQ_init()` function to initialize the EQ algorithm for each audio channel.
4. **Set the Master Gain for each audio channel:** Use the `EQ_setMasterGain()` function to set the master gain for each audio channel.
5. **Set the frequency band gains:** Use the `EQ_setGain()` function to set the gain for the equalizer frequency bands. Note how the `gains[]` array is passed to this function to specify the gain.
6. **Apply the Equalizer to an Audio Frame:** Use the `EQ_apply()` function to process an audio frame through the equalizer. The `EQ_apply()` function operates in place, that is, the output is stored back in the input array.

EXAMPLE 4-1: CODE TO ENABLE USE OF THE EQUALIZER LIBRARY (dsPIC33F EXAMPLE)

```
/* Channel 1 memory structure */

int eqStateMemX1[EQ_XSTATE_MEM_SIZE_INT] _XBSS(2);      /* Step 1 */
int eqStateMemY1[EQ_YSTATE_MEM_SIZE_INT] _YBSS(2);      /* Step 1 */

/* Channel 2 memory structure */

int eqStateMemX2[EQ_XSTATE_MEM_SIZE_INT] _XBSS(2);      /* Step 1 */
int eqStateMemY2[EQ_YSTATE_MEM_SIZE_INT] _YBSS(2);      /* Step 1 */

#define GAINS {0,0,0,0,0,0,0,0}                         /* Step 2 */
char gains1[EQ_NO_FREQS] = GAINS;               /* Step 2 - Channel 1 */
char gains2[EQ_NO_FREQS] = GAINS;               /* Step 2 - Channel 2 */
int masterGain1 = 0;                            /* Step 2 - Channel 1 */
int masterGain2 = 0;                            /* Step 2 - Channel 2 */

int main(void)
{

    EQ_init(eqStateMemX1,eqStateMemY1);                 /* Step 3 */
    EQ_init(eqStateMemX2,eqStateMemY2);                 /* Step 3 */

    EQ_setMasterGain(eqStateMemX1,masterGain1);         /*Step 4 */
    EQ_setMasterGain(eqStateMemX2,masterGain2);         /*Step 4 */

    EQ_setGain(eqStateMemX1,gains1);                    /* Step 5 */
    EQ_setGain(eqStateMemX2,gains2);                    /* Step 5 */

    EQ_apply(eqStateMemX1,eqStateMemY1,audio1,EQ_TRUE); /* Step 6 */
    EQ_apply(eqStateMemX2,eqStateMemY2,audio2,EQ_TRUE); /* Step 6 */

}
```

4.4 RESOURCE REQUIREMENTS

The EQ Library requires the following resources while running on the dsPIC DSC device.

4.4.1 Program Memory Usage

TABLE 4-1: PROGRAM MEMORY USAGE

Type	Size (bytes)	Section
Code in Program Memory	900	`.libeq`
Tables in Program Memory	165 (dsPIC33F) 0 (dsPIC33E)	`.const`
Total Program Memory	1065 (dsPIC33F) 900 (dsPIC33E)	—

4.4.2 Data Memory Usage

TABLE 4-2: DATA MEMORY USAGE

Function/Type	Size (bytes)	Alignment	Section
`eqStateMemX`	88	2	X data memory
`eqStateMemY`	80	2	Y data memory
`sigIn`	160	2	Y data memory
Tables in Data Memory	0 (dsPIC33F) 110 (dsPIC33E)	2	X data memory
Total Data Memory	328 (dsPIC33F) 438 (dsPIC33E)	—	—

4.4.3 Estimated Dynamic Memory Usage

TABLE 4-3: ESTIMATED DYNAMIC MEMORY USAGE

Section	Size (bytes)
Heap	0
Stack	< 300

4.4.4 Computational Speed

TABLE 4-4: COMPUTATIONAL SPEED

Function	MIPS	Typical Call Frequency
`EQ_init()`	< 0.5	Once
`EQ_apply()`	2.5	10 ms
All other functions	Minimal	As required

4.4.5 Data Format

The data type of `sigIn` can be 10-bit, 12-bit or 16-bit linear PCM data. The Equalizer algorithm automatically adjusts for the data format used.

4.5 EQUALIZER LIBRARY API FUNCTIONS

This section lists and describes the API functions that are available in the dsPIC DSC Equalizer Library. The functions are listed below followed by their individual detailed descriptions.

- EQ_init
- EQ_apply
- EQ_setGain
- EQ_getGain
- EQ_setMasterGain
- EQ_getMasterGain
- EQ_TRUE
- EQ_FALSE
- EQ_FRAME
- EQ_XSTATE_MEM_SIZE_INT
- EQ_YSTATE_MEM_SIZE_INT
- EQ_DEFAULT_MASTER_GAIN
- EQ_NO_FREQS
- EQ_MAX_MASTER_GAIN
- EQ_MAX_BAND_ATTEN
- EQ_31_BAND
- EQ_62_BAND
- EQ_125_BAND
- EQ_250_BAND
- EQ_500_BAND
- EQ_1000_BAND
- EQ_2000_BAND
- EQ_4000_BAND

EQ_init

Description

Initializes the EQ algorithm.

Include

```
eq_api.h
```

Prototype

```
void EQ_init(int* ptrStateX, int* ptrStateY);
```

Arguments

ptrStateX	pointer to the X memory for this instance of EQ
ptrStateY	pointer to the Y memory for this instance of EQ

Return Value

None.

Remarks

None.

Code Example

```
int   eq_state_mem_x [EQ_XSTATE_MEM_SIZE_INT] _XBSS(2);
int   eq_state_mem_y [EQ_YSTATE_MEM_SIZE_INT] _YBSS(2);
.
.
.
EQ_init(eq_state_mem_x, eq_state_mem_y);
```

EQ_apply

Description

Applies equalization to the current frame of data.

Include

eq_api.h

Prototype

```
void EQ_apply(int* ptrStateX, int* ptrStateY, int* sig_in, int
enable);
```

Arguments

ptrStateX	a pointer to the X memory for this instance of EQ
ptrStateY	a pointer to the Y memory for this instance of EQ
sig_in	a pointer to the input/output buffer of size EQ_FRAME
enable	a flag to indicate if EQ is required for this buffer (EQ_TRUE/EQ_FALSE)

Return Value

None.

Remarks

The EQ algorithm is process-in-place meaning that the output is passed back to the input buffer. Setting Enable to EQ_FALSE returns an unprocessed buffer of data, but the EQ algorithm still runs in the background.

Code Example

```
int eq_state_mem_x  [EQ_XSTATE_MEM_SIZE_INT]   _XBSS(2);
int eq_state_mem_y  [EQ_YSTATE_MEM_SIZE_INT]   _YBSS(2);
int sig_in          [EQ_FRAME]   _XBSS(2);
.
.
.
EQ_init(eq_state_mem_x, eq_state_mem_y);
.
.
.
EQ_apply(eq_state_mem_x, eq_state_mem_y, sig_in, EQ_TRUE);
```

EQ_setGain

Description

Sets the equalizer gains

Include

eq_api.h

Prototype

```
void EQ_setGain(int* ptrStateX, char* gains)
```

Arguments

ptrStateX	a pointer to the X memory for this instance of EQ
gains	a vector containing integer values between 0 and the value of EQ_MAX_BAND_ATTEN representing the desired EQ in dB below maximum

Return Value:

None.

Remarks

Each gain is represented by one byte, hence the char* type. Each entry corresponds to a band, as follows:

- gains[EQ_31_BAND] is the gain for the 31 Hz band
- gains[EQ_62_BAND] is the gain for the 62 Hz band
- gains[EQ_125_BAND] is the gain for the 125 Hz band
- gains[EQ_250_BAND] is the gain for the 250 Hz band
- gains[EQ_500_BAND] is the gain for the 500 Hz band
- gains[EQ_1000_BAND] is the gain for the 1000 Hz band
- gains[EQ_2000_BAND] is the gain for the 2000 Hz band
- gains[EQ_4000_BAND] is the gain for the 4000 Hz band

Code Example

```
EQ_setGain(ptrStateX, gains);
```

Sets the desired EQ levels to the values held in the vector gains for the instance of the algorithm ptrStateX.

EQ_getGain

Description

Gets the current EQ gains as an 8 byte array.

Include

`eq_api.h`

Prototype

```
void EQ_getGain(int* ptrStateX, char* gains)
```

Arguments

`ptrStateX`	a pointer to the X memory for this instance of EQ
`gains`	an array in which the gain value will be stored

Return Value

None.

Remarks

Each gain is represented by one byte, hence the `char*` type.

Code Example

```
EQ_getGain(ptrStateX, gains);
```

For the instance of the algorithm `ptrStateX`, it gets the current EQ levels from the equalizer and stores them to the array `gains`.

EQ_setMasterGain

Description

Sets the overall gain factor for the equalizer. This is to compensate for loss of overall level caused by multiple EQ settings.

Include

```
eq_api.h
```

Prototype

```
void EQ_setMasterGain(int* ptrStateX, int input_gain);
```

Arguments

`ptrStateX`	a pointer to the X memory for this instance of EQ
`input_gain`	a non-negative integer value from 0 to 12 representing the desired gain in dB

Return Value:

None.

Remarks

None.

Code Example

```
EQ_setMasterGain(ptrStateX, input_gain);
```

EQ_getMasterGain

Description

Returns the current master gain setting in dB.

Include

```
eq_api.h
```

Prototype

```
int EQ_getMasterGain(int* ptrStateX)
```

Arguments

`ptrStateX`	a pointer to the X memory for this instance of EQ

Return Value

Master gain setting.

Remarks

None.

Code Example

```
int master_gain;
master_gain = EQ_getMasterGain(ptrStateX)
```

EQ_TRUE

Description

Used to indicate true to the EQ algorithm.

Value

1

EQ_FALSE

Description

Used to indicate false to the EQ algorithm.

Value

0

EQ_FRAME

Description

The size of the input buffer processed.

Value

80

EQ_XSTATE_MEM_SIZE_INT

Description

Size in integers of the memory location required for the X-State memory.

Value

(EQ_YSTATE_MEM_SIZE_INT + 4)

EQ_YSTATE_MEM_SIZE_INT

Description

Size in integers of the memory location required for the Y-State memory.

Value

(EQ_NO_FREQS * 5)

EQ_DEFAULT_MASTER_GAIN

Description

Default master gain setting in dB.

Value

0

EQ_NO_FREQS

Description

Number of frequency bands used by the equalization algorithm.

Value

8

EQ_MAX_MASTER_GAIN

Description

Maximum master gain setting in dB.

Value

12

EQ_MAX_BAND_ATTEN

Description

Maximum EQ band attenuation.

Value

18

EQ_31_BAND

Description

Index value for the 31 Hz band in the gain array.

Value

0

EQ_62_BAND

Description

Index value for the 62 Hz band in the gain array.

Value

1

EQ_125_BAND

Description

Index value for the 125 Hz band in the gain array.

Value

2

EQ_250_BAND

Description

Index value for the 250 Hz band in the gain array.

Value

3

EQ_500_BAND

Description

Index value for the 500 Hz band in the gain array.

Value

4

EQ_1000_BAND

Description

Index value for the 1000 Hz band in the gain array.

Value

5

EQ_2000_BAND

Description

Index value for the 2000 Hz band in the gain array.

Value

6

EQ_4000_BAND

Description

Index value for the 4000 Hz band in the gain array.

Value

7

4.6 APPLICATION TIPS

The Equalizer algorithm performance can be optimized by proper selection of parameters. In general, experimentation with this library is encouraged. Your feedback and comments are welcome, as they help to guide the direction of future development.

The following are some tips for affecting the performance of the algorithm:

1. The optimum input signal levels for testing audio and communication systems are generally considered to lie between -10 dBm0 and -30 dBm0. If digital input speech levels have peaks that are up to three-fourths of full range, then good use is being made of the available precision; levels higher than this carry a risk of amplitude clipping.
2. Every audio data frame should preferably be processed by the Equalizer. This allows the Equalizer to track the audio signal. In cases where equalization is not desired, the `EQ_apply()` function should be called with enable parameters set to `EQ_FALSE`.

NOTES:

Index

Worldwide Sales and Service

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://www.microchip.com/support
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Cleveland
Independence, OH
Tel: 216-447-0464
Fax: 216-447-0643

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

Santa Clara
Santa Clara, CA
Tel: 408-961-6444
Fax: 408-961-6445

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8569-7000
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

China - Chongqing
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

China - Hangzhou
Tel: 86-571-2819-3180
Fax: 86-571-2819-3189

China - Hong Kong SAR
Tel: 852-2401-1200
Fax: 852-2401-3431

China - Nanjing
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

China - Qingdao
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

China - Shanghai
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xian
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

China - Xiamen
Tel: 86-592-2388138
Fax: 86-592-2388130

China - Zhuhai
Tel: 86-756-3210040
Fax: 86-756-3210049

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

Japan - Yokohama
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Hsin Chu
Tel: 886-3-6578-300
Fax: 886-3-6578-370

Taiwan - Kaohsiung
Tel: 886-7-213-7830
Fax: 886-7-330-9305

Taiwan - Taipei
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

UK - Wokingham
Tel: 44-118-921-5869
Fax: 44-118-921-5820

05/02/11