Sample Rate Conversion Library for PIC32 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.

Printed on recycled paper.

ISBN: 978-1-62077-014-6

QUALITY MANAGEMENT SYSTEM CERTIFIED BY DNV

== ISO/TS 16949 ==

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.

SAMPLE RATE CONVERSION LIBRARY FOR PIC32 USER'S GUIDE

Table of Contents

NOTES:

SAMPLE RATE CONVERSION LIBRARY FOR PIC32 USER'S GUIDE

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 online help. Select the Help menu, and then Topics to open a list of available online help files.

INTRODUCTION

This chapter contains general information that will be useful to know before you use the Sample Rate Conversion Library for PIC32. Items discussed in this Preface 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 document describes how to use the Sample Rate Conversion Library for PIC32 with the PIC32 families of devices. The document layout is as follows:

- **Chapter 1. "Introduction"** – This chapter provides an overview of the SRC library.
- **Chapter 2. "Installation"** – This chapter describes the installation procedure for the SRC library.
- **Chapter 3. "Application Programming Interface (API)"** – This chapter outlines how the API functions provided in the SRC 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

<table>
<tr><th>Description</th><th>Represents</th><th>Examples</th></tr>
<tr><td rowspan="2">Italic characters</td><td>Referenced books</td><td>MPLAB® IDE User’s Guide</td></tr>
<tr><td>Emphasized text</td><td>...is the only compiler...</td></tr>
<tr><td rowspan="3">Initial caps</td><td>A window</td><td>the Output window</td></tr>
<tr><td>A dialog</td><td>the Settings dialog</td></tr>
<tr><td>A menu selection</td><td>select Enable Programmer</td></tr>
<tr><td>Quotes</td><td>A field name in a window or dialog</td><td>“Save project before build”</td></tr>
<tr><td>Underlined, italic text with right angle bracket</td><td>A menu path</td><td><u>File>Save</u></td></tr>
<tr><td rowspan="2">Bold characters</td><td>A dialog button</td><td>Click OK</td></tr>
<tr><td>A tab</td><td>Click the Power tab</td></tr>
<tr><td>Text in angle brackets < ></td><td>A key on the keyboard</td><td>Press <Enter>, <F1></td></tr>
<tr><td rowspan="7"><code>Plain Courier New</code></td><td>Sample source code</td><td><code>#define START</code></td></tr>
<tr><td>Filenames</td><td><code>autoexec.bat</code></td></tr>
<tr><td>File paths</td><td><code>c:\mcc18\h</code></td></tr>
<tr><td>Keywords</td><td><code>_asm, _endasm, static</code></td></tr>
<tr><td>Command-line options</td><td><code>-Opa+, -Opa-</code></td></tr>
<tr><td>Bit values</td><td><code>0, 1</code></td></tr>
<tr><td>Constants</td><td><code>0xFF, ‘A’</code></td></tr>
<tr><td><code>Italic Courier New</code></td><td>A variable argument</td><td><code>file</code><code>.o</code>, where <code>file</code> can be any valid filename</td></tr>
<tr><td>Square brackets [ ]</td><td>Optional arguments</td><td><code>mcc18 [options] file [options]</code></td></tr>
<tr><td>Curly brackets and pipe character: { | }</td><td>Choice of mutually exclusive arguments; an OR selection</td><td><code>errorlevel {0|1}</code></td></tr>
<tr><td rowspan="2">Ellipses...</td><td>Replaces repeated text</td><td><code>var_name [, var_name...]</code></td></tr>
<tr><td>Represents code supplied by user</td><td><code>void main (void)
{ ...
}</code></td></tr>
<tr><td>Notes</td><td>A Note presents information that we want to re-emphasize, either to help you avoid a common pitfall or to make you aware of operating differences between some device family members. A Note can be in a box, or when used in a table or figure, it is located at the bottom of the table or figure.</td><td>Note: This is a standard note box.

CAUTION
This is a caution note.

Note 1: This is a note used in a table.</td></tr>
</table>

WARRANTY REGISTRATION

Please complete the enclosed Warranty Registration Card and mail it promptly. Sending in the Warranty Registration Card entitles you 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 Sample Rate Conversion Library for PIC32. The following are available and recommended as supplemental reference resources.

PIC32 Family Reference Manual Sections

Family Reference Manual sections are available, which explain the operation of the PIC32 device family architecture and peripheral modules. The specifics of each device family are discussed in the individual family's device data sheet.

MPLAB® C Compiler for PIC32 User's Guide (DS51686)

This document details the use of Microchip's MPLAB C Compiler for PIC32 microcontrollers to develop 32-bit applications.

MPLAB® IDE User's Guide (DS51519)

Refer this document for more information pertaining to the installation and implementation of the MPLAB IDE software, as well as the MPLAB Editor and MPLAB SIM Simulator software that are included with it.

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 the Microchip in-circuit emulator, MPLAB REAL ICE™
- **In-Circuit Debuggers** – The latest information on the Microchip in-circuit debugger, 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 device programmer and the PICkit™ 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://www.microchip.com/support

DOCUMENT REVISION HISTORY

Revision A (February 2013)

This is the initial released version of the document.

Chapter 1. Introduction

The Sample Rate Conversion Library for PIC32, provides the ability to convert the sampling rate of real-time raw stereo audio data to select common audio sample rates. The library is designed for streaming audio applications and can be used with analog front ends with limited sample rate capability. The library supports both 16-bit and 24-bit input data resolution.

The 16-bit and 24-bit library supports the following modes:

- Up conversion from a sample rate of 44.1 kHz to 48 kHz
- Up conversion from a sample rate of 32 kHz to 48 kHz

In addition, the 24-bit library supports the following mode as well:

- Down conversion from a sample rate of 48 kHz to 44.1 kHz

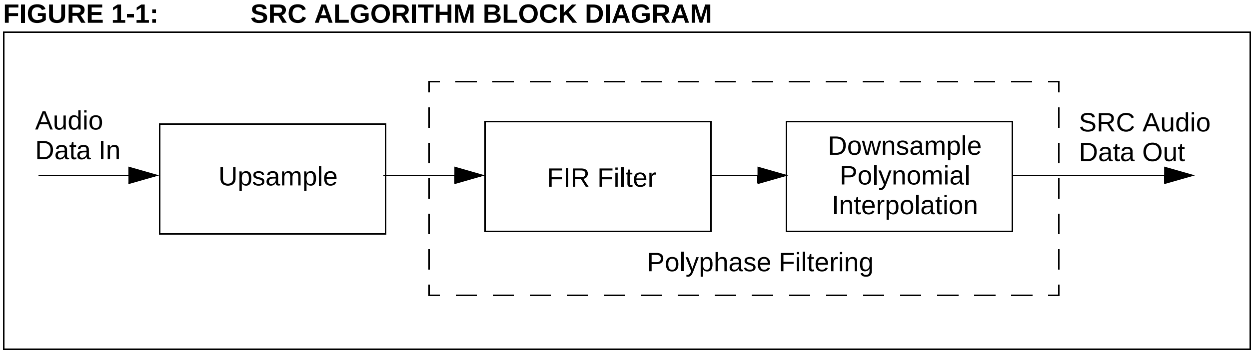
1.1 ALGORITHM OVERVIEW

The SRC algorithm inter-converts the sample rate of a real-time 16-bit or 24-bit stereo audio data stream. A 1 ms frame containing 16-bit or 24-bit stereo audio samples are processed by the SRC application programming interface (API) on each function call. The size of the input audio data in a 1 ms interval will be 48 stereo samples for 48 kHz input, 32 stereo samples for 32 kHz input, and 44 stereo samples for nine consecutive frames, followed by a frame of 45 stereo samples for 44.1 kHz input. The output consists of a similar number of samples based on the output sample rate.

Figure 1-1 shows the basic blocks of the SRC algorithm. The incoming audio data passes through an upsampler or an interpolation stage. The signal then passes through an anti-aliasing low-pass filter followed by a downsampler or decimation stage.

Different designs are used due to the difference in the conversion factors of the supported SRC modes. The 16-bit SRC library is designed for PIC32 devices with a maximum operating frequencies of 40 MHz or 80 MHz. The 24-bit SRC library is designed for PIC32 devices with a maximum operating frequency of 80 MHz. A trade-off between the MIPS and the performance is achieved by these different designs due to the limited MIPS availability.

FIGURE 1-1: SRC ALGORITHM BLOCK DIAGRAM



1.1.1 32 kHz to 48 kHz Sample Rate Conversion

The input is upsampled by a factor of 3 followed by a Finite Impulse Response (FIR) filter to smooth the signal. A gain factor is applied to the smoothed signal to compensate for the loss caused by inserting the zeros. The resulting intermediate signal is then downsampled by a factor of 2 to obtain an output audio signal at a sampling rate of 48 kHz. Since downsampling creates redundancy in the filtering of the sample that is decimated, the filtering of this sample can be skipped, resulting in significant instruction cycle execution savings. This is a simplified form of the polyphase filtering technique, which improves the speed of the SRC. In this mode, every 1 ms of audio frame is expected to have 32 stereo samples at input since the input sample rate is 32 kHz. There are 48 stereo samples at output since the output sample rate is 48 kHz.

1.1.2 44.1 kHz to 48 kHz Sample Rate Conversion

The input is upsampled by a factor of 2 in the 16-bit library and by factor of 4 in the 24-bit library, by inserting zeros after every input sample followed by a FIR filter, which is applied to smooth the signal. A gain factor is applied to the smoothed signal to compensate for the loss caused by inserting the zeros. Polynomial interpolation is used to reduce every sequence of 147 samples at 88.2 kHz in the 16-bit library and 294 samples at 176.4 kHz in the 24-bit library to 80 samples at 48 kHz. Polyphase filtering is also employed to reduce redundancy. In this mode, the first nine audio frames are expected to contain 44 stereo samples and the tenth frame is expected to contain 45 stereo samples at the input since the input sample rate is 44.1 kHz. There are 48 stereo samples at output since the output sample rate is 48 kHz.

1.1.3 48 kHz to 44.1 kHz Sample Rate Conversion

The input is upsampled by a factor of 4 in the 24-bit library, by inserting zeros after every input sample followed by a FIR filter, which is applied to smooth the signal followed by applying a gain factor. Polynomial interpolation is used to reduce every sequence of 640 samples at 88.2 kHz to 147 samples at 44.1 kHz. This ensures the sampling rate of the output audio data to be 44.1 kHz. Polyphase filtering is also employed to reduce redundancy. In this mode, there are 48 stereo samples at input since the input sample rate is 48 kHz. At the output, the first nine audio frames are expected to contain 44 stereo samples and the tenth frame is expected to contain 45 stereo samples since the output sample rate is 44.1 kHz.

1.1.4 Filter Design of Input FIR Filter

For 32 kHz input mode, a low-pass filter is utilized with a very steep roll-off to limit aliasing effects. The cut-off frequency lies below one-sixth of the intermediate sample rate. This is below the Nyquist frequency, so that aliasing effects should not be a problem when downsampling. The cut-off frequency used is approximately 15 kHz for both the 16-bit and 24-bit libraries, which may be desirable when optimizing the filter design to adjust the cut-off frequency so that the first stop-band null covers the input Nyquist frequency.

For 44.1 kHz input mode, the cut-off frequency of the low-pass filter lies below one-fourth of the intermediate sample rate in the 16-bit library. The cut-off frequency used is 19.7 kHz. In the 24-bit library, the cut-off frequency of the low-pass filter lies below one-eighth of the intermediate sample rate at 16 kHz.

For 48 kHz input mode, the cut-off frequency of the low-pass filter lies below one-eighth of the intermediate sample rate at 16 kHz in the 24-bit library.

A more aggressive optimization of the filter is needed, which is provided by an equiripple design technique in all the filter designs. The respective passband ripple for the above filters are captured in the tables 2-2, 2-3, 2-4. The stop attenuation for filters in the 44.1 kHz and the 48 kHz input modes is less than -70 dB, and less than -100 dB for 32 kHz input mode. The overall processing load is dominated by the FIR filter; however, there is a trade-off between filter design and the quality of the output in both the 16-bit and 24-bit libraries. The trade-off is a bit more in the 24-bit library because of the increased resolution and the resulting intermediate instructions on the 32-bit PI32 devices for handling that data resolution.

NOTES:

Chapter 2. Installation

This chapter describes the installation procedure for the Sample Rate Conversion Library for PIC32 and includes resource usage.

Topics covered include:

- Installation Procedure
- Resource Usage
- Using the Audacity Audio Editing Tool

2.1 INSTALLATION PROCEDURE

The SRC library is available as a download from the Sample Rate Conversion Web page at: www.microchip.com/SRC. After downloading and extracting the files, run the installer to install the library in the desired path. The 24-bit SRC installer installs the 24-bit SRC library and the 16-bit SRC installer installs the 16-bit SRC library.

Use the following procedure to add the library to the application:

1. In the application MPLAB workspace, right-click **Library Files** in the Project Window, and select **Add files**.
2. Browse to the location of the archive file, which is located in the `libs` folder within the installation directory and choose one of the following files:
   - `libSRC_LITE_44_1KHz_to_48KHz_v1_0.a` (24-bit library)
   - `libSRC_LITE_32KHz_to_48KHz_v1_0.a` (24-bit library)
   - `libSRC_LITE_48KHz_to_44_1KHz_v1_0.a` (24-bit library)
   - `libSRC_LITE_PIC32_v1_0.a` (16-bit library)
   - `libSRC_FULL_PIC32_v1_0.a` (16-bit library)
3. Select the desired file and click **Open**. The SRC library is now added to the application.

2.2 RESOURCE USAGE

2.2.1 Performance and Library Mode Resource Consumption for 16-bit SRC Implementation

Table 2-1 and Table 2-2 provide performance and library mode resource consumption information for PIC32 devices.

TABLE 2-1: PERFORMANCE AND RESOURCE CONSUMPTION FOR PIC32 LIBRARY MODES

SRC Library Version	SRC Mode	MIPS	Code Size (bytes)	Data Size (bytes)	SNR (dB)[1]
Lite Version	32k to 48 kHz	27.4	5684	1284	82
	44.1k to 48 kHz	30.1			82
Full Version	32k to 48 kHz	33.5	5760	1364	84
	44.1k to 48 kHz	36.7			82

Note 1: Tested with a 1 kHz full-scale sinusoidal signal.

2.2.2 Performance and Library Mode Resource Consumption for 24-bit SRC Implementation

Table 2-2, Table 2-3, and Table 2-4 provide the MIPS, Program Memory, Data memory used along with the Total Harmonic Distortion with Noise (THDN) measured with select tones across the audio frequency range for the various modes in the 24-bit version of the SRC library.

> **Note:** A trade-off between the MIPS and the performance is achieved in the 24-bit version of the SRC library due to the limited MIPS availability.

TABLE 2-2: RESOURCE USAGE FOR 48 kHz TO 44.1 kHz 24-BIT SAMPLE RATE CONVERSION

Resource	Tone Frequency (Hz)	THDN (dBFS)
Passband Ripple < ±1.5 dB MIPS = 57 Program Memory = 5916 bytes Data Memory = 6976 bytes	100	-70
	1K	-73
	5K	-70
	10K	-71
	12K	-70
	15K	-71

TABLE 2-3: RESOURCE USAGE FOR 44.1 kHz TO 48 kHz 24-BIT SAMPLE RATE CONVERSION

Resource	Tone Frequency (Hz)	THDN (dBFS)
Passband Ripple < ±2 dB MIPS = 61 Program Memory = 5984 bytes Data Memory = 6612 bytes	100	-75
	1K	-70
	5K	-70
	10K	-76
	12K	-69
	15K	-69

TABLE 2-4: RESOURCE USAGE FOR 32 kHz TO 48 kHz 24-BIT SAMPLE RATE CONVERSION

Resource	Tone Frequency (Hz)	THDN (dBFS)
Passband Ripple < ±0.2 dB MIPS = 53 Program Memory = 2608 bytes Data Memory = 3328 bytes	100	-104
	1K	-104
	5K	-110
	10K	-115
	12K	-105
	15K	-105

Note 1: The tones used for testing are 24-bit stereo audio tones.

2: The compiler used to generate the resource usage data is XC32 v1.20.

3: The libraries are compiled with a 32-bit instruction set and with compiler speed optimization (-O3) enabled.

2.3 USING THE AUDACITY AUDIO EDITING TOOL

The SRC library operates in real-time on streaming audio with raw stereo audio samples. However, if the user wants to analyze the library other than in real-time, the Audacity audio editing tool can be used to create and analyze `.RAW` audio files. The Audacity audio editing tool is a good tool to edit, create, and analyze high-quality audio streams.

To use the Audacity audio editing tool, do the following:

1. Download and install Audacity audio editing tool from: http://audacity.sourceforge.net/
2. Create the `.RAW` audio file.

   Note: Refer to the tutorial on using the Track drop-down menu: http://manual.audacityteam.org/man/Track_Drop-Down_Menu

   a) Import the audio file into Audacity.
   b) Ensure that the audio imported is stereo audio. If not, create a copy of the audio signal for the missing channel by selecting *Track > Make Stereo Track*.
   c) Select *Track > Set Sample Format > 24-bit PCM*.
   d) Select *File > Export*.
   e) Select *Save as Type > Other uncompressed files*.
   f) Click **Options** and select *Header > RAW (header-less)*.
   g) Select *Encoding > Signed 32-bit PCM*.
   h) Specify a filename and click **Save**.
3. Importing the output `.RAW` audio file.
   a) Select *File> Import > Raw Data*.
   b) Select the file by browsing the folders and click **Open**.
   c) Select *Encoding > Signed 32-bit PCM*.
   d) Select *Byte Order > Little-endian*.
   e) Select *Channels > 2 Channels (Stereo)*.
   f) Enter the sample rate in the Sample Rate text box.
   g) Click **Import**.
   h) Play or analyze the imported audio file.

Chapter 3. Application Programming Interface (API)

This chapter describes the Application Programming Interface (API) to the Sample Rate Conversion Library for PIC32.

The API functions of the SRC Library are easy to use. Depending on the library version, the prototype declaration of the API functions, the state buffer used, and buffer sizes to be used for the SRC modes are located in the header file. The archive file is the library file to be used in the project workspace. Example 3-1 and Example 3-2 list the available header and archive files for the 16-bit and 24-bit versions of the SRC library, respectively.

EXAMPLE 3-1: 16-BIT HEADER AND ARCHIVE FILES

```
|
|----- h
|
|            src_api.h        (header file for SRC APIs)
|
|
|----- libs
|
|            libSRC_x_y_v1_0.a (SRC library archive)
```

The 24-bit SRC libraries have been split according to the different conversion modes to provide the user an opportunity to minimize the program memory requirement on the SRC by selectively including the desired conversion mode.

EXAMPLE 3-2: 24-BIT HEADER AND ARCHIVE FILES

```
|
|--------h
|
|      src_api_LITE_44_1KHz_to_48KHz.h
|      src_api_LITE_32KHz_to_48KHz.h
|      src_api_LITE_48KHz_to_44_1KHz.h
|
|--------libs
|
|      libSRC_LITE_44_1KHz_to_48KHz_v1_0.a
|      libSRC_LITE_32KHz_to_48KHz_v1_0.a
|      libSRC_LITE_48KHz_to_44_1KHz_v1_0.a
```

3.1 SAMPLE RATE CONVERSION LIBRARY API FUNCTIONS

3.1.1 16-bit Library API Functions

This section lists and describes the two API functions that are available in the 16-bit version of the SRC Library.

SRC_init()

Description

The API function, SRC_init(), configures the conversion mode to be used and initializes the local state buffer required by the SRC library. The SRC state contains buffer pointers, split buffers needed for the filters, and buffers for polynomial interpolation.

Prototype

```
void SRC_init(int* ptrSRC_state, int cd_flag);
```

Arguments

ptrSRC_state	a pointer to the state memory for this instance of the SRC library
cd_flag	SRC_32KHZ_TO_48KHZ – 32 kHz to 48 kHz conversion mode SRC_44_1KHZ_TO_48KHZ – 44.1 kHz to 48 kHz conversion mode

Example

```
#define CD_FLAG SRC_32KHZ_TO_48KHZ
.
.
.
int srcStateMem[SRC_STATE_MEM_SIZE_INT];
.
.
.
SRC_init(srcStateMem, CD_FLAG);
```

SRC_apply()

Description

The API function, `SRC_apply()`, is the function call that performs sample rate conversion on the input audio signal. The API takes pointer to the input buffer, pointer to the output buffer along with a pointer to the SRC state. It also takes the size of the input audio buffer as a parameter. The size of the input audio data buffer is 64 for input sample rate of 32 kHz and 88 or 90 for the input sample rate of 44.1 kHz.

Prototype

```
int SRC_apply(int* ptrSRC_state, short* Sin, short* Sout,
   int readCount);
```

Arguments

`ptrSRC_state`	a pointer to the state memory for this instance of SRC
`Sin`	a pointer to the input buffer of size with incoming audio signal `SRC_IN_PROC_SIZE`
`Sout`	a pointer to the output buffer of size `SRC_OUT_PROC_SIZE`
`readCount`	the actual number of samples to be read from the input buffer

Example

```
#define CD_FLAG SRC_32KHZ_TO_48KHZ
.
.
.
short    Sin[SRC_IN_PROC_SIZE];
short    Sout[SRC_OUT_PROC_SIZE];
int      srcStateMem[SRC_STATE_MEM_SIZE_INT];
int      read_count;
int      write_count;
.
.
.
read_count = 64;
.
.
.
SRC_init(srcStateMem, CD_FLAG);
.
.
.
write_count = SRC_apply(srcStateMem, Sin, Sout, read_count);
```


3.1.2 24-bit Library API Functions

The 24-bit SRC libraries have been split according to the different conversion modes to provide the user an opportunity to minimize the program memory requirement on the SRC by selectively including the desired conversion mode.

SRC_init_x()

Description

The API function, `SRC_init_x()`, configures the conversion mode to be used and initializes the local state buffer required by the 24-bit version of the SRC library. The SRC state contains buffer pointers, split buffers needed for the filters, and buffers for polynomial interpolation.

Prototype

```
void SRC_init_44_1KHz_to_48KHz(int* ptrSRC_state);
void SRC_init_32KHz_to_48KHz(int* ptrSRC_state);
void SRC_init_48KHz_to_44_1KHz(int* ptrSRC_state);
```

Arguments

`ptrSRC_state`	a pointer to the state memory for this instance of SRC

Example

```
int srcStateMem[SRC_STATE_MEM_SIZE_INT];
.
.
.
SRC_init_44_1KHz_to_48KHz(srcStateMem);
```

SRC_apply_x()

Description

The API function, SRC_apply_x(), is the function call in the 24-bit version of the library that performs sample rate conversion on the input audio signal. The API takes pointer to the input buffer with unpacked stereo audio data, pointer to the output buffer which will contain unpacked stereo audio along with a pointer to the SRC state. It also takes the size of the input audio buffer as a parameter. The size of the input audio data buffer is 32 for input sample rate of 32 kHz, 48 for input sample rate of 48 kHz and 44 or 45 for the input sample rate of 44.1 kHz. Depending on the mode selected, the size of the output data will be 32 for output sample rate of 32 kHz, 48 for output sample rate of 48 kHz and 44 or 45 for the output sample rate of 44.1 kHz.

> **Note:** The readCount has been updated to be the number of stereo samples as opposed to the 16-bit version of the library where it is the total number of samples including the left and right channels.

Prototype

```
int   SRC_apply_44_1KHz_to_48KHz(int* ptrSRC_state,
                                 AudioStereo32b* Sin,
                                 AudioStereo32b* Sout,
                                 int readCount);
int   SRC_apply_32KHz_to_48KHz(int* ptrSRC_state,
                                 AudioStereo32b* Sin,
                                 AudioStereo32b* Sout,
                                 int readCount);
int   SRC_apply_48KHz_to_44_1KHz(int* ptrSRC_state,
                                 AudioStereo32b* Sin,
                                 AudioStereo32b* Sout,
                                 int readCount);
```

Arguments

ptrSRC_state	pointer to the state memory for this instance of SRC
Sin	pointer to the AudioStereo32b input stereo audio data buffer (AudioStereo32b data type is a structure with two 32-bit members one for left channel audio data and one for right channel audio data)
Sout	pointer to the AudioStereo32b output stereo audio data buffer
readCount	the actual number of samples to be read from the input buffer

Example

```
AudioStereo32b Sin[SRC_IN_PROC_SIZE];
AudioStereo32b Sout[SRC_OUT_PROC_SIZE];
int       srcStateMem[SRC_STATE_MEM_SIZE_INT];
int       read_count;
int       write_count;
.
.
.
read_count = 32;
.
.
.
SRC_init_44_1KHz_to_48KHz(srcStateMem);
.
.
.
write_count = SRC_apply _44_1KHz_to_48KHz(srcStateMem, Sin, Sout,
    read_count);
```

NOTES:

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: 86-571-2819-3187
Fax: 86-571-2819-3189

China - Hong Kong SAR
Tel: 852-2943-5100
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: 86-755-8864-2200
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 - Osaka
Tel: 81-6-6152-7160
Fax: 81-6-6152-9310

Japan - Tokyo
Tel: 81-3-6880- 3770
Fax: 81-3-6880-3771

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: 886-3-5778-366
Fax: 886-3-5770-955

Taiwan - Kaohsiung
Tel: 886-7-213-7828
Fax: 886-7-330-9305

Taiwan - Taipei
Tel: 886-2-2508-8600
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

11/29/12