



Audio and waveform generation using the DAC
in STM32F0xx microcontroller families

Introduction

This application note gives examples for generating audio output signals using the Digital to Analog Converter (DAC) peripheral embedded in the STM32F0xx microcontroller family.

A digital to analog converter, DAC, is a device that has the opposite function to an analog to digital converter, it converts a digital word to a corresponding analog voltage.

The STM32 DAC module is a 12-bit word converter, with one output channel for supporting mono audio.

The DAC can be used in many audio applications such as: security alarms, Bluetooth headsets, talking toys, answering machines, man-machine interfaces, and low-cost music players.

STM32 DAC can also be used for many other analog purposes, such as analog waveform generation and control engineering.

The application note is organized in two main sections:

- [Section 1](#) describes the main features of the STM32 DAC module.
- [Section 2](#) presents two examples.
 - In the first example, DAC is used to generate a sine waveform.
 - In the second example, the DAC is used to generate audio from .WAV files.

Contents

- 1 DAC main features 5**
 - 1.1 Data format 5
 - 1.2 Dedicated timers 5
 - 1.3 DMA capabilities 6
 - 1.4 DMA underrun error 7
 - 1.5 Buffered output 7

- 2 Application examples 9**
 - 2.1 Using the DAC to generate a sine waveform 9
 - 2.1.1 Description 9
 - 2.1.2 Digital Sine waveform pattern preparation 9
 - 2.1.3 Fixing the sine wave frequency 11
 - 2.2 Using the DAC to implement an audio wave player 12
 - 2.2.1 Description 12
 - 2.2.2 Audio wave file specifications 13
 - 2.2.3 .WAV file format 13
 - 2.3 Audio wave player implementation 13

- 3 Conclusion 16**

- 4 Revision history 17**

List of tables

Table 1. Digital and analog sample values of the Sine wave 10
Table 2. Document revision history 17

List of figures

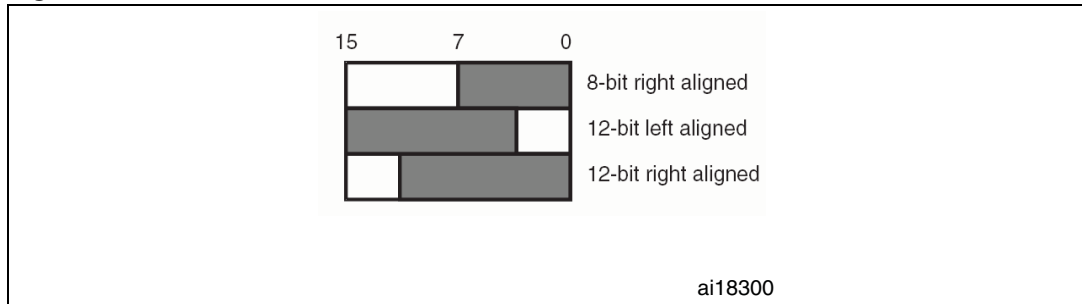
Figure 1.	DAC data format	5
Figure 2.	STM32F0xx DAC trigger channel	5
Figure 3.	DAC interaction without DMA	6
Figure 4.	DAC interaction with DMA	7
Figure 5.	Non buffered channel voltage (with and without load)	8
Figure 6.	Buffered channel voltage (with and without load)	8
Figure 7.	Sine wave model samples	9
Figure 8.	Sine wave generated with ns = 10	11
Figure 9.	Sine wave generated with ns = 255	11
Figure 10.	Flow of data from MicroSD Flash memory to external speakers	12
Figure 11.	Wave Player flowchart	14
Figure 12.	CPU and DMA activities during wave playing process	15

1 DAC main features

1.1 Data format

The DAC accepts data in 3 integer formats: 8-bit, 12-bit right aligned and 12-bit left aligned. A 12-bit value can range from 0x000 to 0xFFF, with 0x000 being the lowest and 0xFFF being the highest value.

Figure 1. DAC data format



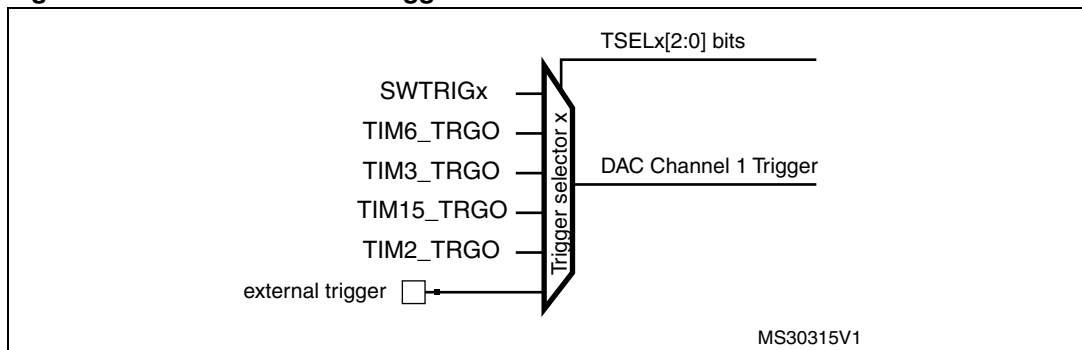
1.2 Dedicated timers

In addition to the software and External triggers, the DAC conversion can be triggered by different timers.

TIM6 is a basic timer and is basically designed for DAC triggering.

Each time a DAC interface detects a rising edge on the selected Timer Trigger Output (TIMx_TRGO), the last data stored in the DAC_DHRx register is transferred to the DAC_DORx register.

Figure 2. STM32F0xx DAC trigger channel

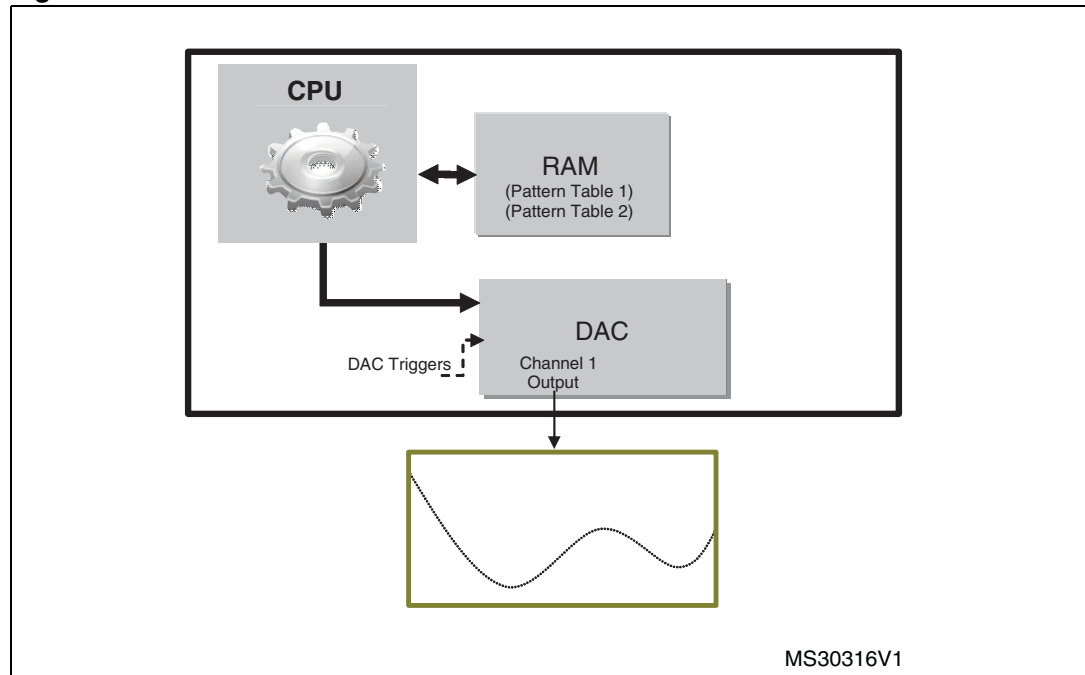


1.3 DMA capabilities

The STM32 microcontrollers have a DMA module with multiple channels. The DAC channel is connected to an independent DMA channel. In the case of STM32F0xx Microcontrollers, the DAC channel is connected to the DMA channel 3.

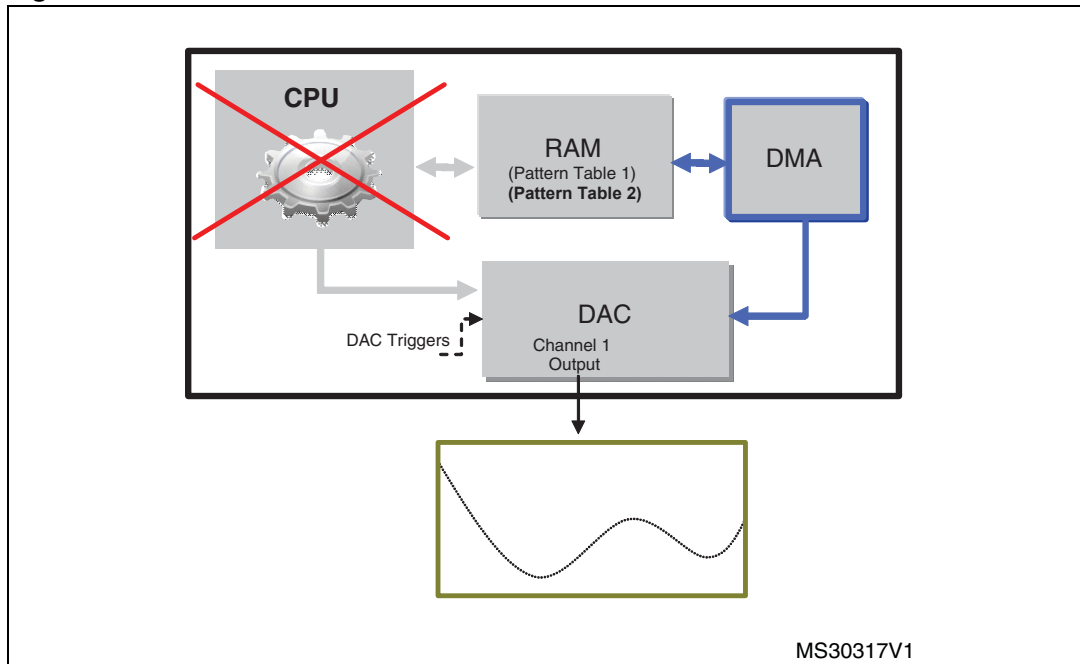
When DMA is not utilized, the CPU is used to provide DAC with the pattern waveform. Generally the waveform is saved in a memory (RAM), and the CPU is in charge of transferring the data from RAM to the DAC.

Figure 3. DAC interaction without DMA



When using the DMA, the overall performance of the system is increased by freeing up the core. This is because data is moved from memory to DAC by DMA, without needing any actions by the CPU. This keeps CPU resources free for other operations.

Figure 4. DAC interaction with DMA



1.4 DMA underrun error

When the DMA is used to provide DAC with the pattern waveform, there are some cases where the DMA transfer is faster than the DAC conversion. In this case, DAC detects that a part of the pattern waveform has been ignored and was not converted. It then sets the "DMA underrun Error" flag.

The underrun error can be handled using a shared IRQ channel with the triggering Timer or by a dedicated interrupt when DAC is not triggered by TIM6.

1.5 Buffered output

To drive external loads without using an external operational amplifier, the DAC channel has an embedded output buffer which can be enabled and disabled depending on the user application.

When the DAC output is not buffered, and there is a load in the user application circuit, the voltage output will be lower than the desired voltage. Enabling the buffer, the voltage output and the voltage desired are similar.

Figure 5. Non buffered channel voltage (with and without load)

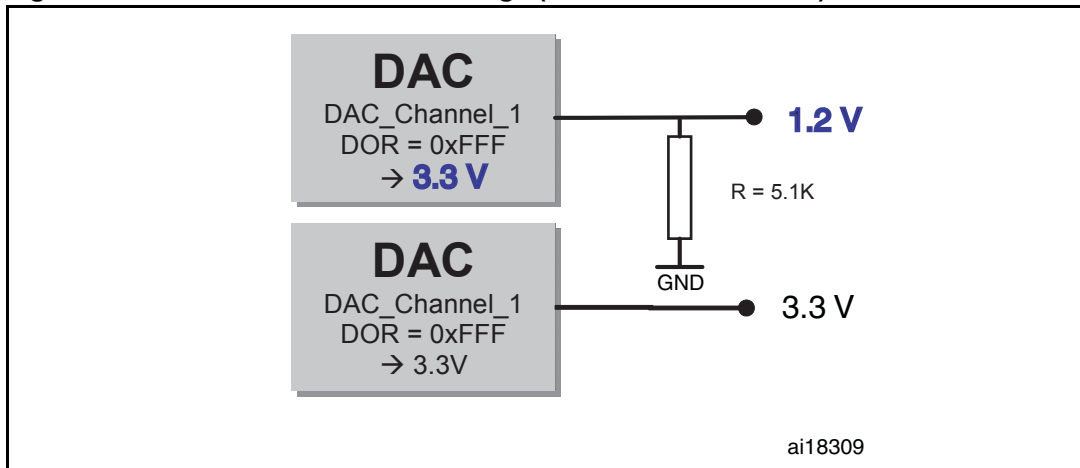
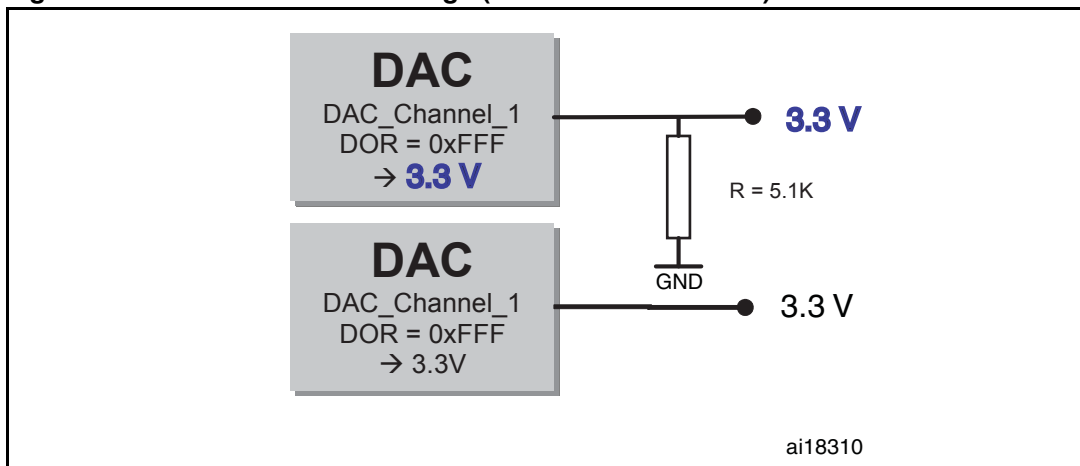


Figure 6. Buffered channel voltage (with and without load)



2 Application examples

2.1 Using the DAC to generate a sine waveform

2.1.1 Description

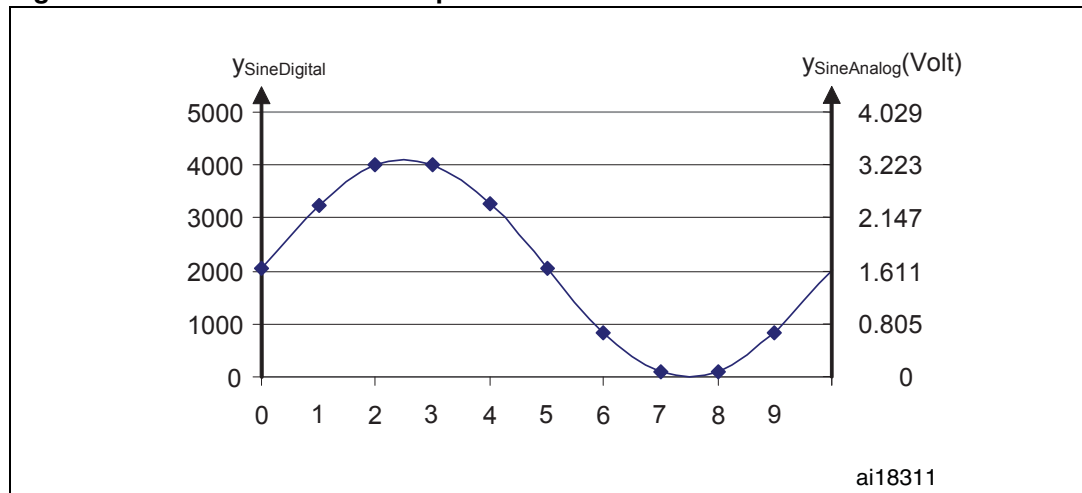
This example describes step by step how to generate a sine waveform.

A sine waveform is also called a sine tone with a single frequency, it is known as a pure tone or sinus tone. The sine tones are traditionally used as stimuli in determining the various responses of the auditory system.

2.1.2 Digital Sine waveform pattern preparation

To prepare the digital pattern of the waveform, we have to do some mathematics. Our objective is to have 10 digital pattern data (samples) of a sine wave form which varies from 0 to $2 \cdot \pi$.

Figure 7. Sine wave model samples



The sampling step is $(2 \cdot \pi) / n_s$ (number of samples).

The result value of $\sin(x)$ is between -1 and 1, we have to recalibrate it to have a positive sinewave with samples varying between 0 and 0xFFFF (which corresponds to the range from 0 V to 3.3 V).

$$Y_{\text{SineDigital}}(x) = \left(\sin\left(x \cdot \frac{2\pi}{n_s}\right) + 1 \right) \left(\frac{0xFFFF + 1}{2} \right)$$

Digital inputs are converted to output voltages on a linear conversion between 0 and VREF+. The analog output voltages on the DAC channel pin are determined by the following equation:

$$DAC_{Output} = V_{REF} \frac{DOR}{DAC_MaxDigitalValue}$$

Note: For right-aligned 12-bit resolution: DAC_MaxDigitalValue = 0xFFF
 For right-aligned 8-bit resolution: DAC_MaxDigitalValue = 0xFF

So the analog sine waveform ySineAnalog can be determined by the following equation:

$$y_{SineAnalog}(x) = 3.3V_{olt} \frac{y_{SineDigital}(x)}{0xFFF}$$

Table 1. Digital and analog sample values of the Sine wave

Sample (x)	Digital Sample Value ySineDigital (x)	Analog Sample Value (Volt) ySineAnalog(x)
0	2048	1.650
1	3251	2.620
2	3995	3.219
3	3996	3.220
4	3253	2.622
5	2051	1.653
6	847	0.682
7	101	0.081
8	98	0.079
9	839	0.676

The table is saved in the memory and transferred by the DMA, the transfer is triggered by the same timer that triggers the DAC.

2.1.3 Fixing the sine wave frequency

To fix the frequency of the sine wave signal, you have to set the frequency of the Timer Trigger output.

The frequency of the produced sine wave is:

$$f_{\text{Sinewave}} = \frac{f_{\text{TimerTRGO}}}{n_s}$$

So, if TIMx_TRGO is 1 MHz, the frequency of the DAC sine wave is 10 kHz.

Note: To have a high quality sine wave curve, it is recommended to use a high number of samples n_s .

Figure 8. Sine wave generated with $n_s = 10$

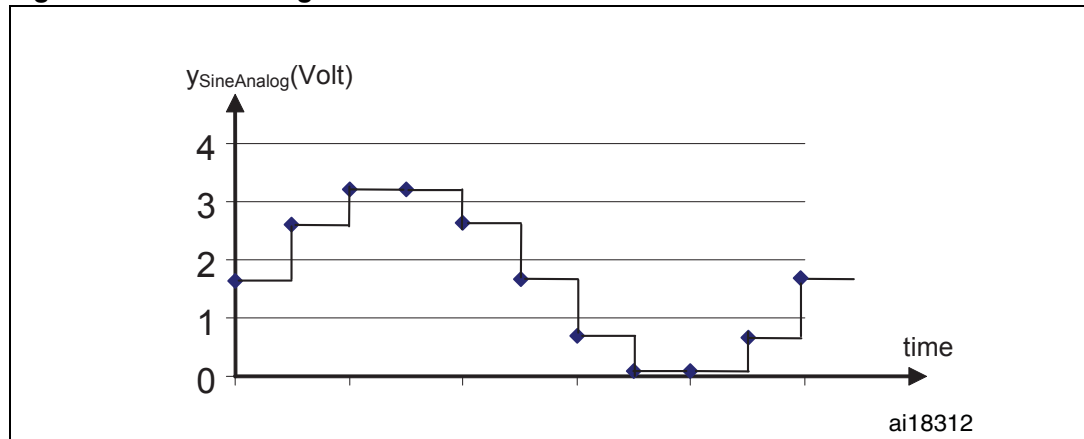
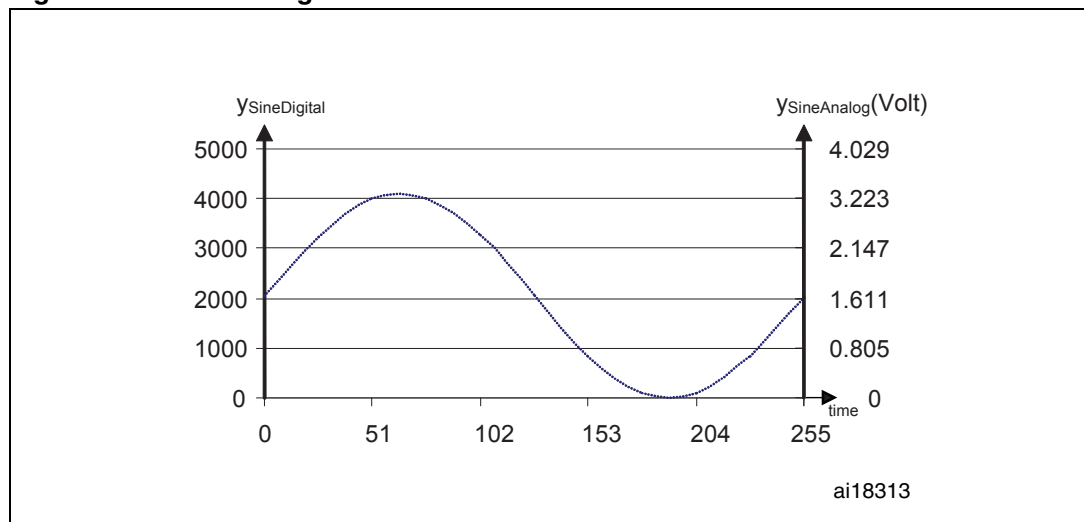


Figure 9. Sine wave generated with $n_s = 255$

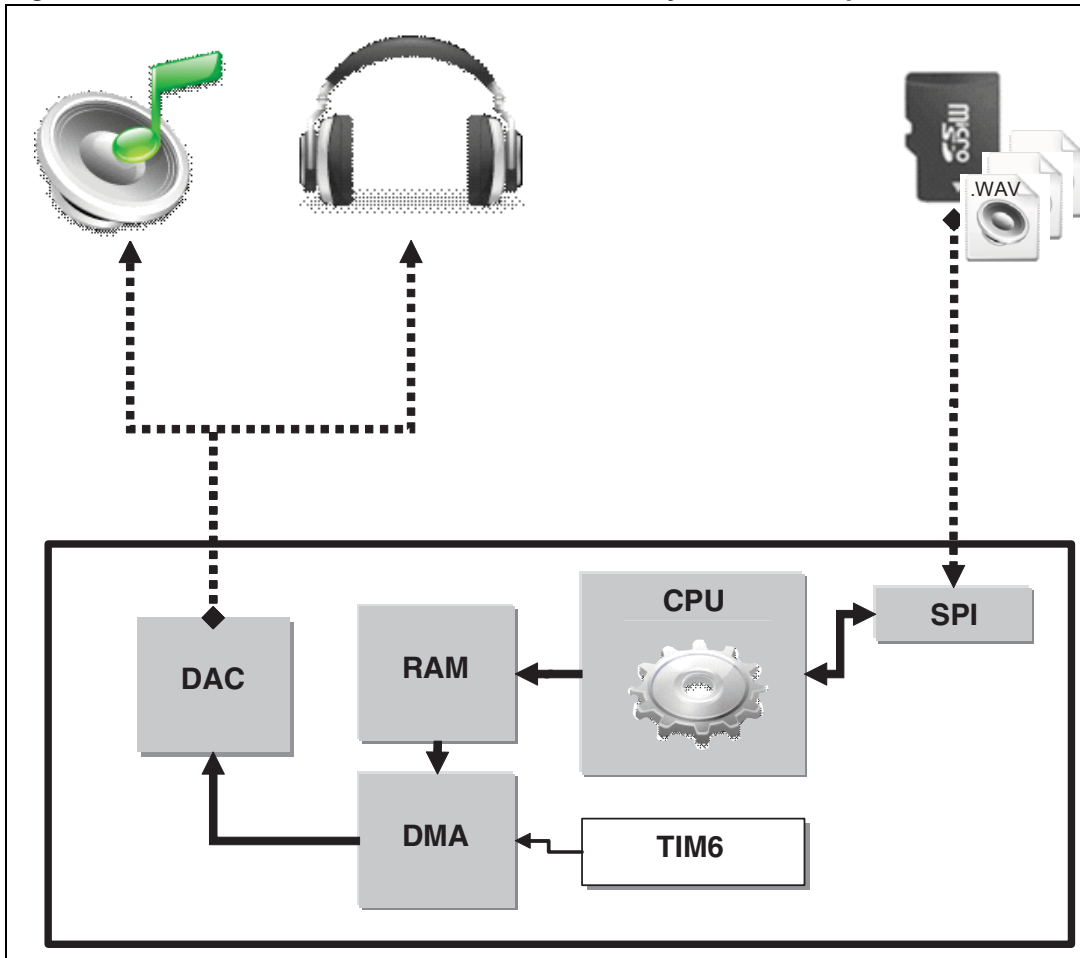


2.2 Using the DAC to implement an audio wave player

2.2.1 Description

The purpose of this application demo is to provide an audio player solution for the STM32 microcontroller for playing .WAV files. The approach is optimized to use a minimum number of external components, and offers the flexibility for end-users to use their own .WAV files. The audio files are provided to the STM32 from a MicroSD memory card.

Figure 10. Flow of data from MicroSD Flash memory to external speakers



The audio wave player demonstration described in this section is a part of the STM320518-EVAL demonstration firmware. You can download this firmware and the associated user manual (UM1520) from the STMicroelectronics website www.st.com.

2.2.2 Audio wave file specifications

This application assumes that the .WAV file to be played has the following format:

- Audio Format: PCM (an uncompressed wave data format in which each value represents the amplitude of the signal at the time of sampling)
- Sample rate: may be 8000, 11025, 22050 or 44100 Hz
- Bits Per Sample: 8-bit (Audio sample data values are in the range [0-255])
- Number of Channels: 1 (Mono)

2.2.3 .WAV file format

The .WAV file format is a subset of the Resource Interchange File Format (RIFF) specification used for the storage of multimedia files. A RIFF file starts with a file header followed by a sequence of data chunks. A .WAV file is often just a RIFF file with a single "WAVE" chunk consisting of two sub-chunks:

1. a **fmt** chunk specifying the data format
2. a **data** chunk containing the actual sample data

The WAVE file format starts with the RIFF header: it indicates the file length.

Next, the fmt chunk describes the sample format, it contains information about: Format of the wave audio : (PCM/...), Number of channels (mono/stereo), sample rate (number of samples per seconds : e.g., 22050), and the sample Data size (e.g. 8bit/16bit). Finally, the data chunk contains the sample data.

2.3 Audio wave player implementation

The Audio wave player application is based on the SPI, DMA, TIM6, and DAC peripherals.

At start up, the application first uses the SPI to interface with the MicroSD card and parses its content, using the FatFs file system, looking for available .wav files in the USER folder. Once a valid .wav file is found, it is read back through the SPI, and the data is transferred using the CPU to a buffer array located in the RAM. The DMA is used to transfer data from RAM to the DAC peripheral. TIM6 is used to trigger the DAC which will convert the Audio digital data to an analog waveform.

Before the audio data can be played, the header of the WAV file is parsed so that the sampling rate of the data and its length can be determined.

The task of reproducing audio is achieved by using sampled data (data contained in the .WAV file) to update the value of the DAC output, this data is coded in 8 bits (with values from 0 to 255),

The DAC Channel 1 is triggered by TIM6 at regular intervals specified by the sample rate of the .WAV file header.

The .WAV files are read from the MicroSD Memory using a DosFS file system.

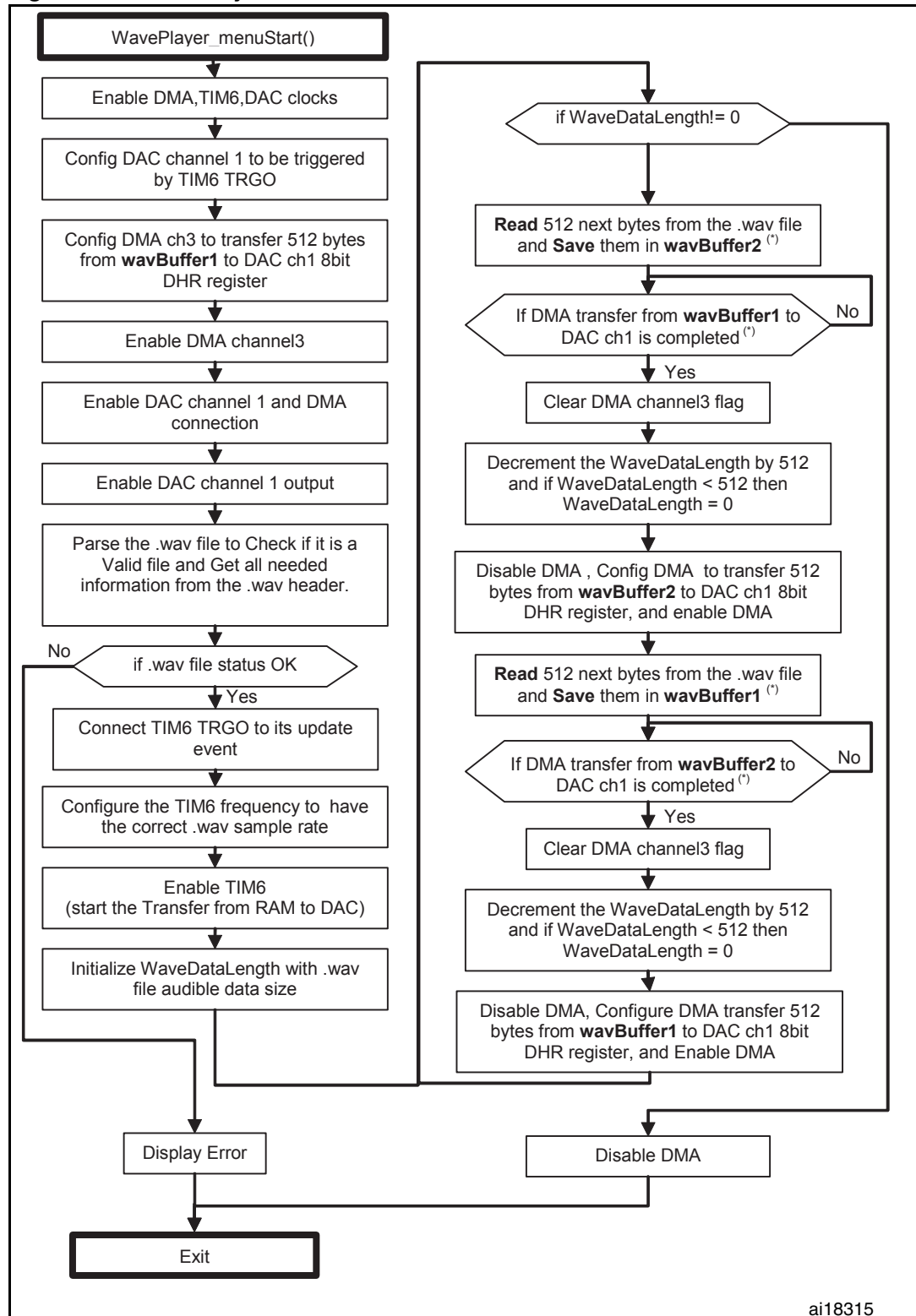
In the Demo code, code files handling the waveplayer demo are:

waveplayer.c

waveplayer.h

The wave player demo is called using the `WavePlayerMenu_Start()` function which has the following flowchart.

Figure 11. Wave Player flowchart



ai18315

(*) when DMA is transferring data from one RAM buffer, CPU is transferring data from the MicroSD Flash memory to the other RAM buffer.



3 Conclusion

The first part of this application note and both examples given in [Section 2](#) of this document have been provided to help you get familiar with the DAC's main features. The first example (in [Section 2.1](#)) shows how to generate an analog waveform, using the example of a sine waveform.

The second example (in [Section 2.2](#)) offers a straightforward and flexible solution for using the STM32, to play .WAV files, stored in an SPI MicroSD Flash memory.

You can use these examples as starting points for developing your own solution using the STM32 DAC.

4 Revision history

Table 2. Document revision history

Date	Revision	Changes
02-May-2012	1	Initial release

Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST's terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST'S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY TWO AUTHORIZED ST REPRESENTATIVES, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER'S OWN RISK.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

© 2012 STMicroelectronics - All rights reserved

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Philippines - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

www.st.com