

Introduction

This document gives an overview of the whole audio processing modules available and how they can be connected together to fulfill some audio Use Cases. It is intended to the programmer who integrates the audio modules into a main program.

Examples of supported audio processing chains are also provided.

Table 1. Applicable products

| Type | Reference products |
|------------------|--------------------|
| Microcontrollers | STM32-AUDIO100A |

Contents

- 1 Description 5**
 - 1.1 List of products supported by the audio processing modules 5

- 2 Audio Modules Integration Overview 6**
 - 2.1 Supported Audio Modules 6
 - 2.2 Audio Modules APIs overview 6
 - 2.3 Audio Modules Integration 7

- 3 Audio Chains Examples 8**
 - 3.1 Home Audio Demo 8
 - 3.2 Car Audio Demo 9

- 4 Revision history 10**

List of tables

| | | |
|----------|---------------------------------------|----|
| Table 1. | Applicable products | 1 |
| Table 2. | Available audio processing list | 6 |
| Table 3. | Document revision history | 10 |

List of figures

| | | |
|-----------|-----------------------------------|---|
| Figure 1. | Home audio processing chain | 8 |
| Figure 2. | Car audio processing chain | 9 |

1 Description

1.1 List of products supported by the audio processing modules

All listed audio libraries are designed to run on a Cortex M4 core with FPU usage, so it can be integrated and run on any STM32F40xx, STM32F41xx, STM32F42xx, or STM32F43xx platforms.

2 Audio Modules Integration Overview

2.1 Supported Audio Modules

In table below are listed all the audio processing modules available for integration. All these modules are libraries generated using EWARM tool chain, with internal 32-bits processing and supporting either 16 or 32 bits I/O buffers.

Table 2. Available audio processing list

| Modules | User Manual | Features |
|---------------------------------------|-------------|--|
| SRC236 | UM1641 | Sampling Rate Converter supporting ratios 2, 3, 6, 1/2, 1/3, 1/6, 3/2 and 2/3 |
| SRC441 | UM1640 | Sampling Rate Converter for specific 44.1->48 kHz conversion (10 ms framing) |
| OmniSurround Stereo Widener | UM1633 | Audio virtualization for a 1.0/2.0 input stream to a widened 2.0 output stream |
| OmniSurround Multichannel Virtualizer | UM1655 | Audio virtualization for 5.1/7.1 input stream to a virtualized 2.0 output stream |
| GrEq | UM1798 | 10-bands graphical equalizer for 48 kHz input signal |
| BIQ | UM1625 | Generic Biquad filtering library |
| BAM | UM1778 | Bass Manager (including compressor + limiter) |
| SVC | UM1642 | Smart Volume Control (including compressor) |
| GAM | Under Dev | Gain manager |
| Panning | Under Dev | Handles panning, balancing and sweet spot for Car Audio UC (4.0 output) |
| Beeper | Under Dev | Chimes generation for Car Audio UC |

Please refer to corresponding User Manuals for more information on each audio module.

2.2 Audio Modules APIs overview

All audio modules are optimized and packaged in generic audio APIs:

- `xxx_reset()`: Resets modules and initializes static memory.
- `xxx_setParam()`: Sets module static parameters.
- `xxx_getParam()`: Gets module static parameters values.
- `xxx_setConfig()`: Sets module dynamic parameters.
- `xxx_getConfig()`: Gets module dynamic parameters values.
- `xxx_process()`: Process routine to be called at each frame.

“xxx” refers to any module prefix (BAM for Bass Manager for instance).

2.3 Audio Modules Integration

First of all, all the static and dynamic memory used by each module must be allocated by the integration framework. Dynamic and static structures are hidden to the integration framework, but their sizes are exported as constant in xxx_glo.h file, so memory allocation can be done as written below:

```
/* xxx memory structure memory allocation */  
void *static_mem_ptr = malloc(XXX_STATIC_MEM_SIZE);  
void *dynamic_mem_ptr = malloc(XXX_DYNAMIC_MEM_SIZE);
```

Then, it is needed to allocate memory for input and output audio buffers.

All audio modules are HW protected to lock their usage on STM32 platforms so before calling any xxx_reset() routine, it is needed to enable platform CRC-32 and reset it, as shown with below lines:

```
RCC->AHB1ENR |= RCC_AHB1ENR_CRCEN;  
CRC->CR |= CRC_CR_RESET;  
:  
error = xxx_reset(XXX_STATIC_MEM_PTR, XXX_DYNAMIC_MEM_PTR);
```

In a Cube environment, it could be:

```
__CRC_CLK_ENABLE();  
__HAL_CRC_DR_RESET(hcrc);           // with CRC_HandleTypeDef *hcrc  
:  
error = xxx_reset(XXX_STATIC_MEM_PTR, XXX_DYNAMIC_MEM_PTR);
```

xxx_reset() routines as well as all other APIs can now be called by the integration framework.

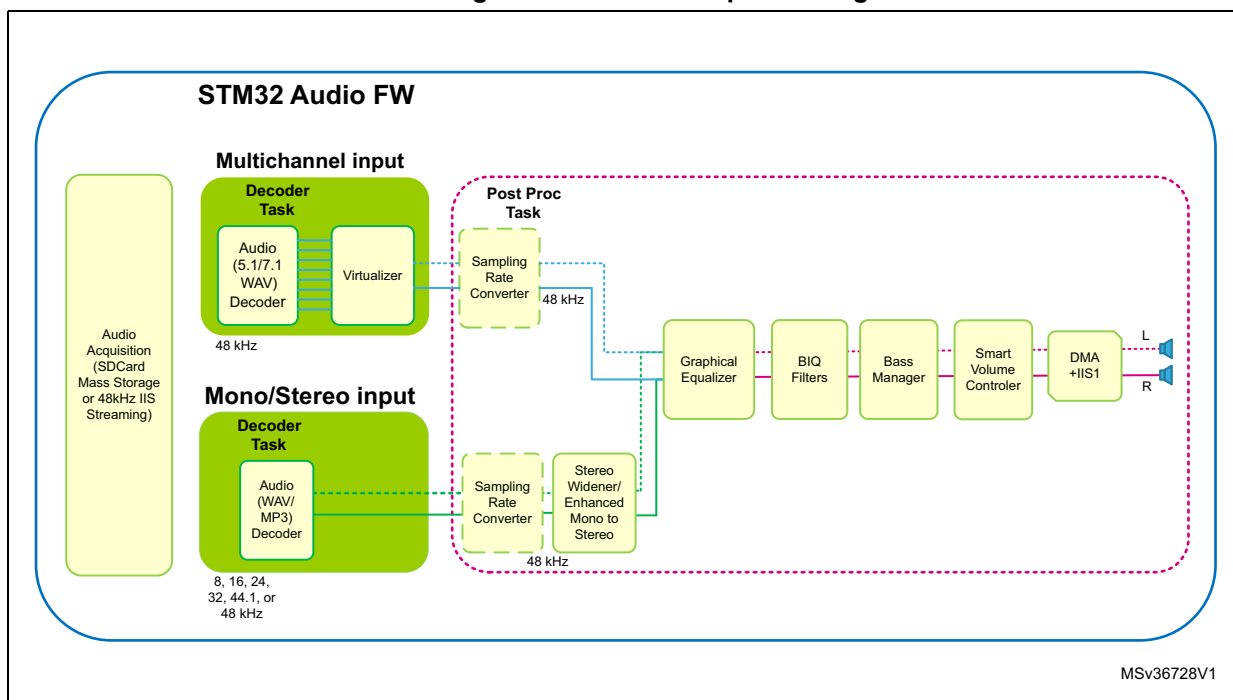
3 Audio Chains Examples

Please find below two audio chains as example of integration of audio processing modules, both of them being controlled with our Audio Tuning Tool. Thanks to contact your local support to help handling such demos.

3.1 Home Audio Demo

This Demo runs on a STM3240G-Eval board, with SDCard Mass Storage input.

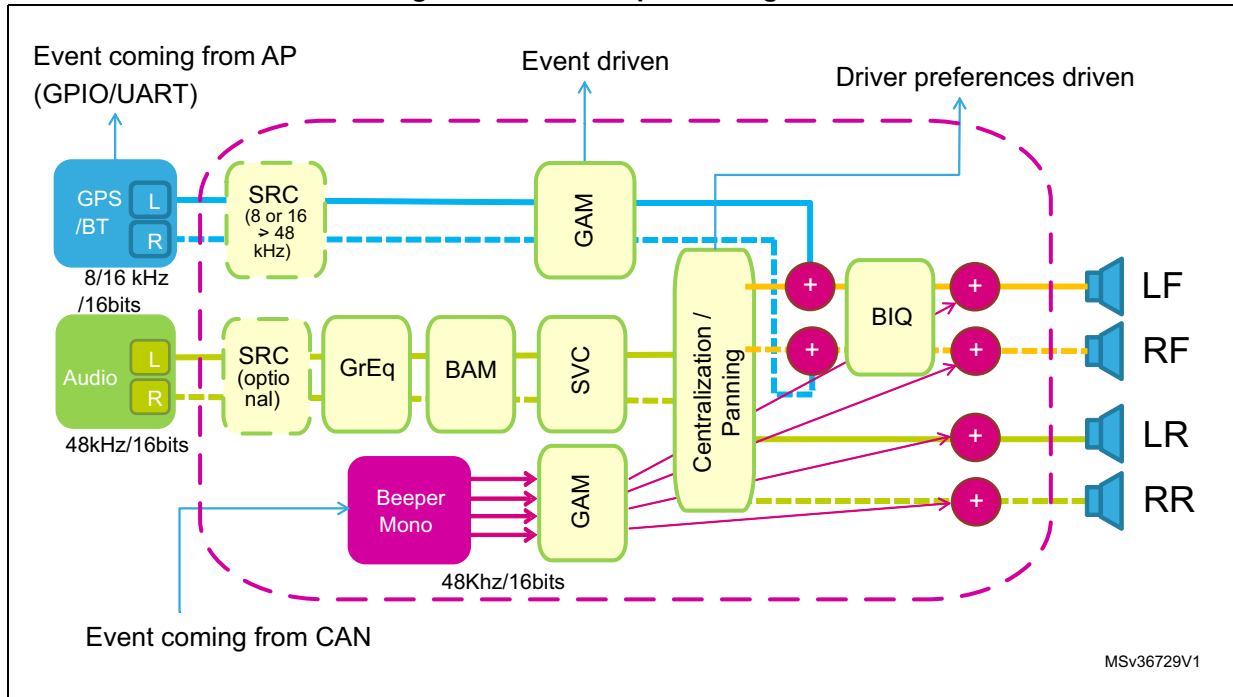
Figure 1. Home audio processing chain



3.2 Car Audio Demo

This Demo runs on a modified STM324x9I-Eval board, with some extension boards to handle external audio amplifiers.

Figure 2. Car audio processing chain



MSv36729V1

4 Revision history

Table 3. Document revision history

| Date | Revision | Changes |
|-------------|----------|-----------------|
| 07-Jan-2015 | 1 | Initial release |

IMPORTANT NOTICE – PLEASE READ CAREFULLY

STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST's terms and conditions of sale in place at the time of order acknowledgement.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of Purchasers' products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

ST and the ST logo are trademarks of ST. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2015 STMicroelectronics – All rights reserved

