利用 DFSDM 开发 PDM 麦克风应用介绍

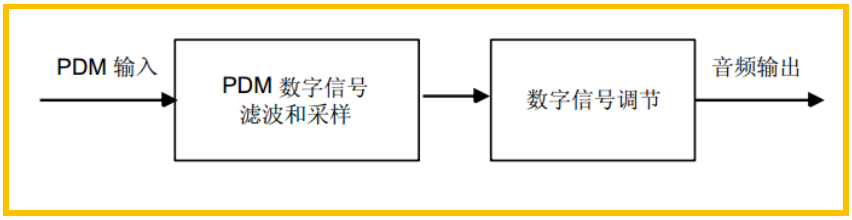
前言

STM32 提供了丰富的音频应用外设，并得益于灵活高效的内部架构，可以支持广泛的音频应用。本文中，在简单介绍音频采集的背景知识后，从应用需求出发，确定麦克风的选用。然后，描述了 STM32 内部 DFSDM （Digital Filter for Sigma-Delta Modulator）在 PDM 麦克风采集中应用。最后逐步介绍如何利用 STM32CubeMX 进行 DFSDM 设计开发，实现 PDM 麦克风声音采集。

一 背景知识

声音通过声学传感器获取模拟信号，经过模数转换器，转换成二进制码 0 和 1，这些 0 和 1 便构成了音频数字信号。PDM 麦克风能够实现上述的模拟信号获取，并输出 PDM 信号。PDM（Pulse Density Modulation）脉冲密度调制，利用脉冲密度表示模拟信号强度。

从 PDM 位流中获取数据，还需要经过如下图环节才能获得模拟信号幅度对应的数字量。



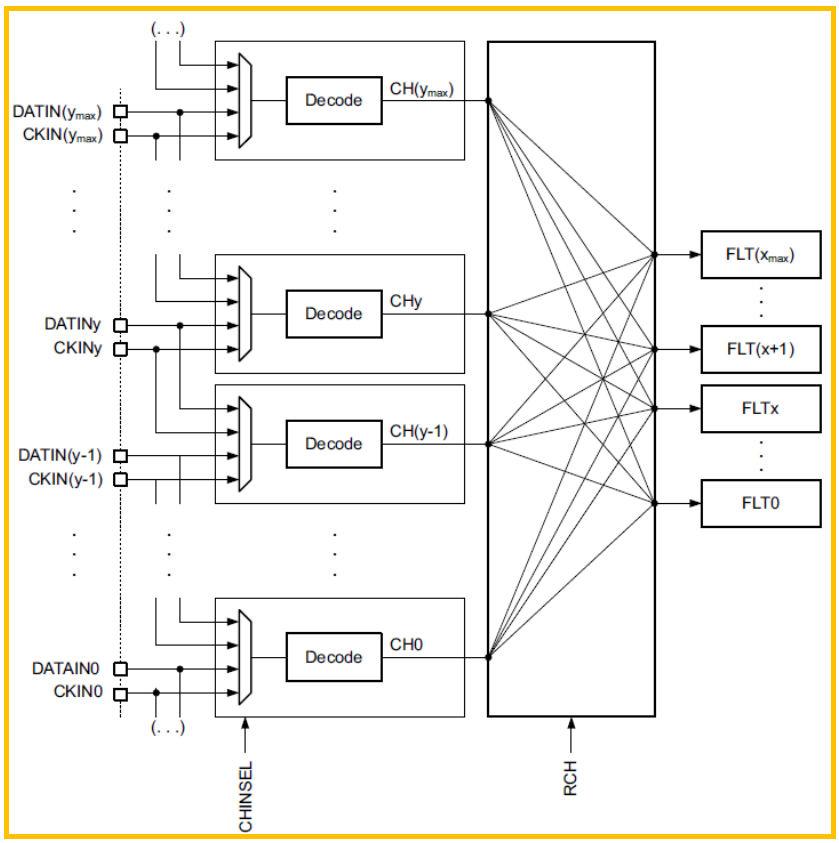
二 应用需求及 DFSDM 支持分析

在音频应用开发前，需要根据应用需求，对麦克风个数、支持编码类型、采样率及分辨率等进行确定。下面围绕 DFSDM 在这些需求方面的支持情况进行分析。

2.1 麦克风数量

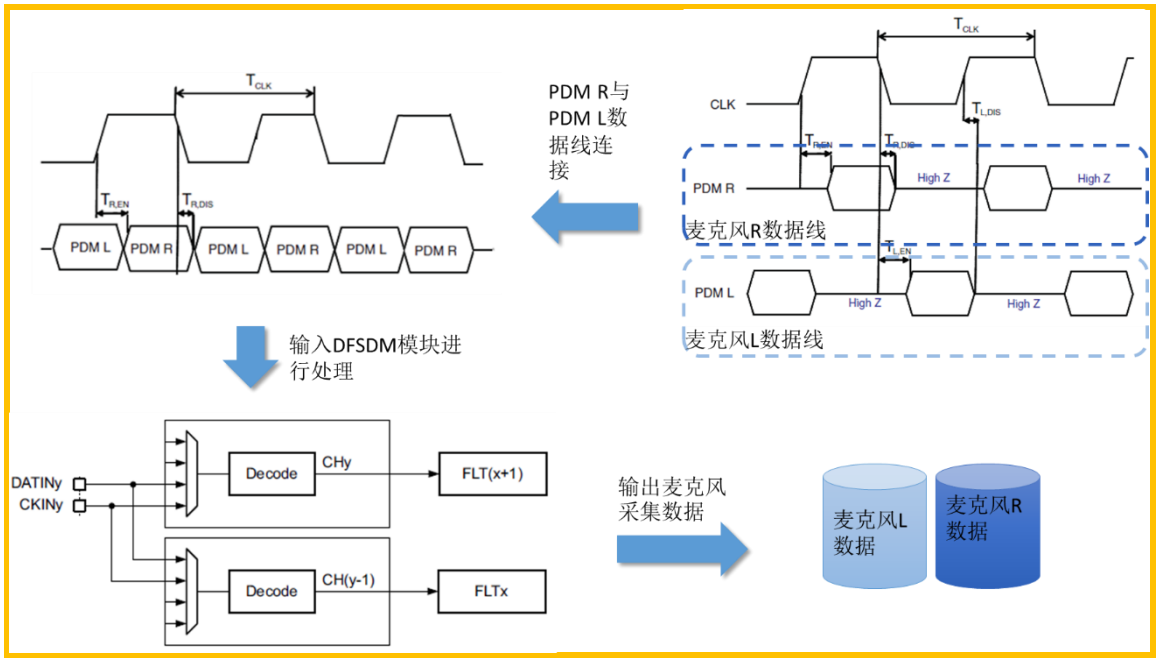
同时运行的最大麦克风数量，对于 DFSDM，由 DFSDM 中滤波器数量决定。简单理解，就是一个滤波器对应一个麦克风。注意这种简单等同并不适用于非同时采样的应用场景。

麦克风的数量不直接对应通道数量，如下图。



可映射任一 DFSDM 通道单元至滤波器单元。对于通道 CH(y-1)，数据线可来源于 DATIN(y-1)引脚，也可来自于 DATIN(y)，在通道单元中可以选择获取数据的时刻（上升沿或者下降沿）。这样带来的益处是，可利用内部两个通道单元对同一个数据线上数据进行分离并处理获得采样数据。而这个应用，直接满足了双通道数据在同一条数据线上的数据采集场景。

上述描述的应用场景中，数据处理流向如下图所示：



注 1：图中 CLK 线硬件设计上不一定需要连接，DFSDM 可在内部关联，实现利用输出时钟作为时钟输入，具体可通过参考手册了解。

2.2 编码类型

DFSDM 支持 PDM、曼彻斯特编码，支持具有类似编码的麦克风器件，具体可通过参考手册了解。

2.3 采样率

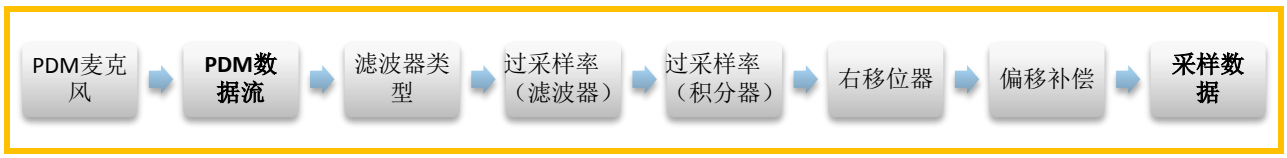
DFSDM 通过时钟源、滤波模式、快速模式选择、过采样配置，实现不同采样率的支持。能够支持常见的 8k,16k,22k,44k,48k 的采样率，也能够支持特殊应用场合所需的更高采样率，例如 192k, 384k 等。

更多关于 DFSDM 采样率介绍，以及具体计算公式可通过参考手册了解。

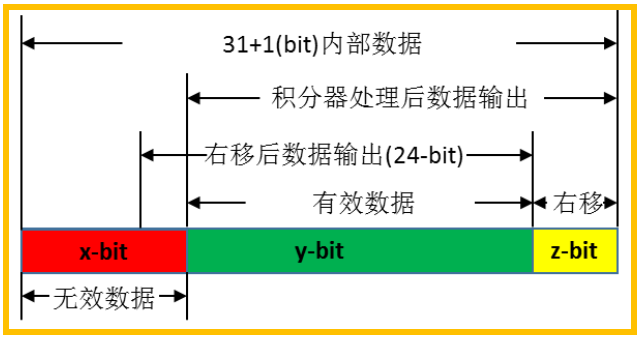
2.4 分辨率

DFSDM 具有 24 位数据寄存器，可通过配置实现不同分辨率的支持，有效数据最高支持到 24 位。同时，新的 HAL 库支持全硬件获取 16 位采样数据，不增加 CPU 负载。

DFSDM 分辨率由过采样率，滤波器类型和右移位器决定，更多内容可通过参考手册了解。



在不同处理环节数据分辨率情况如下图所示。



积分器最大数据输出范围如下表。例如当 FOSR 为 64，IOSR 为 1，采用 $Sinc^3$ 滤波，在不考虑后续处理环节时，输出范围为±262144，分辨率能够达到 19 位。并可通过右移位器灵活获得需要的有效数据位数。

Filter 过采样率(FOSR)	积分器过采样率(IOSR)	$Sinc^1$	$Sinc^2$	FastSinc	$Sinc^3$	$Sinc^4$	$Sinc^5$
x	y	± x ▪ y	± x^2 ▪ y	± 2x^2 ▪ y	± x^3 ▪ y	± x^4 ▪ y	± x^5 ▪ y

三 前期准备

出于将重心放在 DFSDM 应用介绍，简化其他环节考虑。本文中实现例在 ST 提供的 NUCLEO-L476RG 和 X-NUCLEO-CCA02M1 板展开。

考虑到利用 DFSDM 实现 PDM 麦克风采集，首先根据 UM1900 对麦克风板 X-NUCLEO-CCA02M1 进行处理，使其支持基于 DFSDM 采集的两路 PDM 麦克风。需要准备软硬件资源如下表。

STM32 板	ST 麦克风板	Cable	STM32CubeMX	Cube 软件包	IDE
NUCLEO-L476RG * 1 **链接：** http://www.st.com/content/st_com/en/products/evaluation-tools/product-evaluation-tools/mcu-eval-tools/stm32-mcu-eval-tools/stm32-mcu-nucleo/nucleo-l476rg.html	X-NUCLEO-CCA02M1 *1 **链接：** http://www.st.com/content/st_com/en/products/ecosystems/stm32-open-development-environment/stm32-nucleo-expansion-boards/stm32-ode-sense-hw/x-nucleo-cca02m1.html	USB 线缆 * 1 (Type A ←→ Mini B)	**链接：** http://www.st.com/content/st_com/en/products/development-tools/software-development-tools/stm32-software-development-tools/stm32-configurators-and-code-generators/stm32cubemx.html	STM32CubeL4 **说明：**在安装 STM32CubeMX 后，在其“菜单栏\Help\Install New Libraries”中安装 STM32CubeL4.本文实现例中采用的是 STM32Cube_FW_L4_V1.8.0	IAR **链接：** https://www.iar.com/ **说明：**本文中以 IAR 为例。除 IAR 外，CubeMX 支持 MDK、TrueStudio、SW4STM32

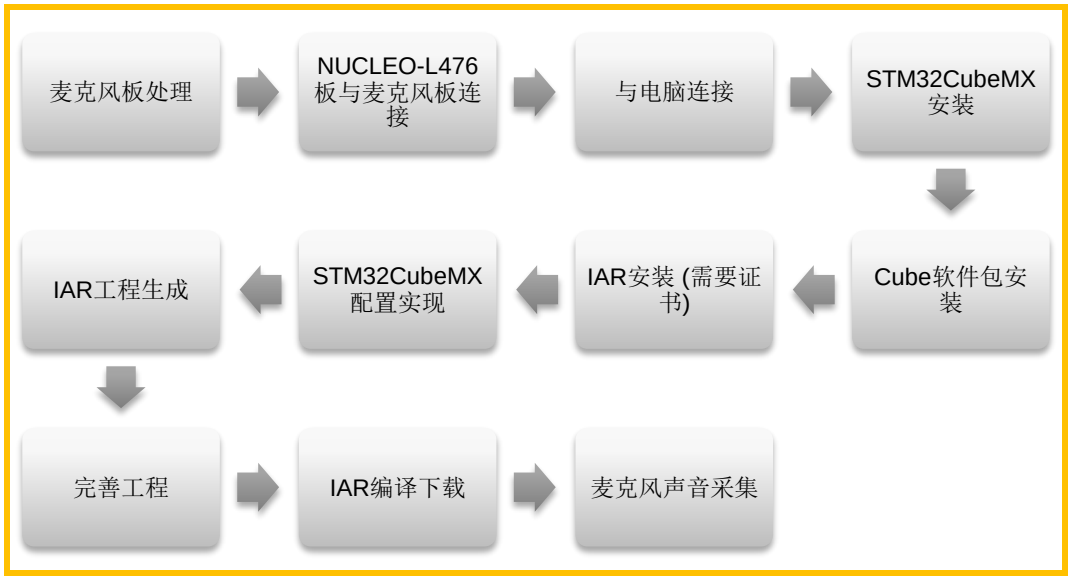
四 实现过程

4.1 应用实现

利用 X-NUCLEO-CCA02M1 板上已有的两路 PDM 麦克风，可实现最多两路麦克风数据采集。

在本例中，先将应用需求定为两路麦克风采集，采样率为 8KHz，分辨率为 16-bit。后续介绍如何利用 STM32CubeMX 生成遵循应用需求的工程，以及在获得工程后，如何启动采样，实现麦克风采集的应用。

4.2 开发流程

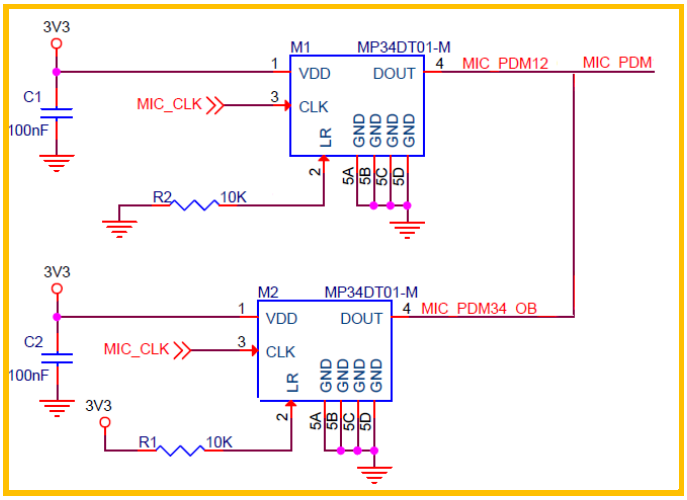


4.3 STM32CubeMX 配置实现

STM32CubeMX 操作流程如下图所示。

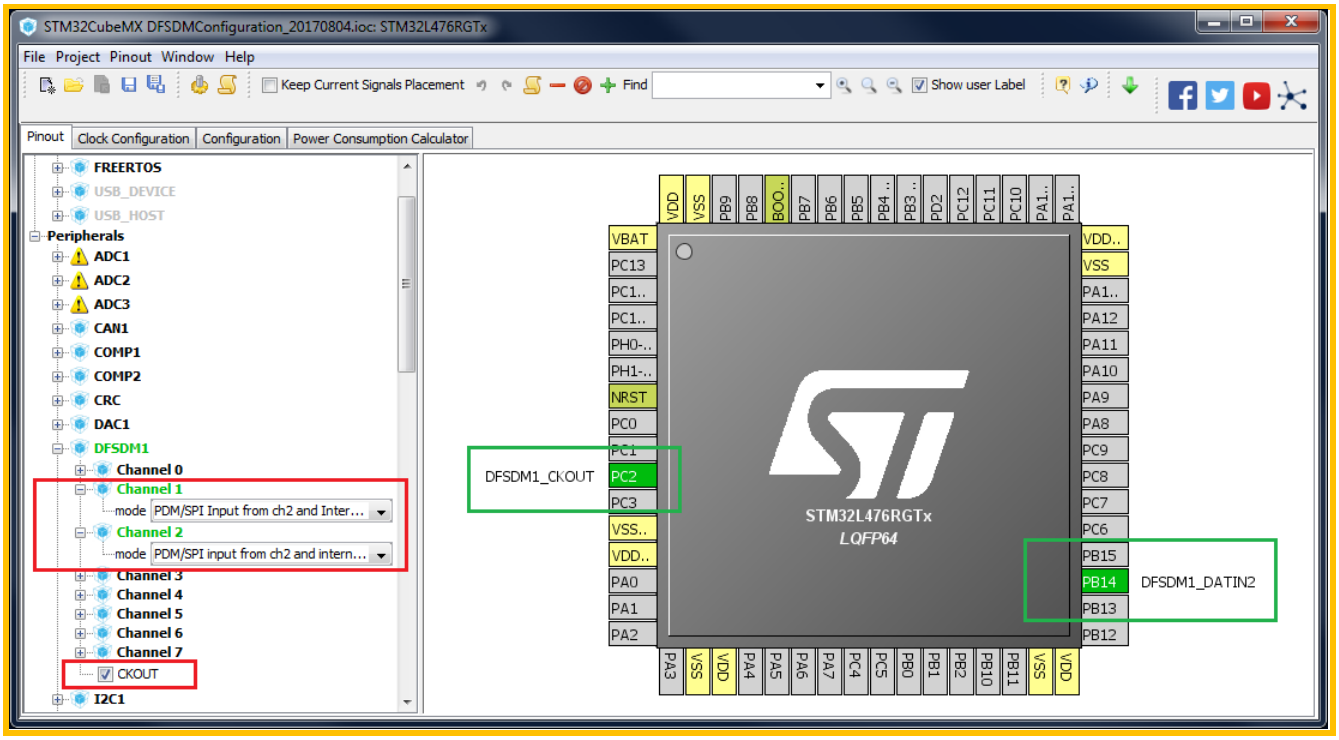
a. DFSDM 通道选择

根据 X-NUCLEO- CCA02M1 板原理图，可知在将其处理成支持 DFSDM 采集的两路麦克风时，麦克风总线引脚与 STM32L476RG 连接情况如下。



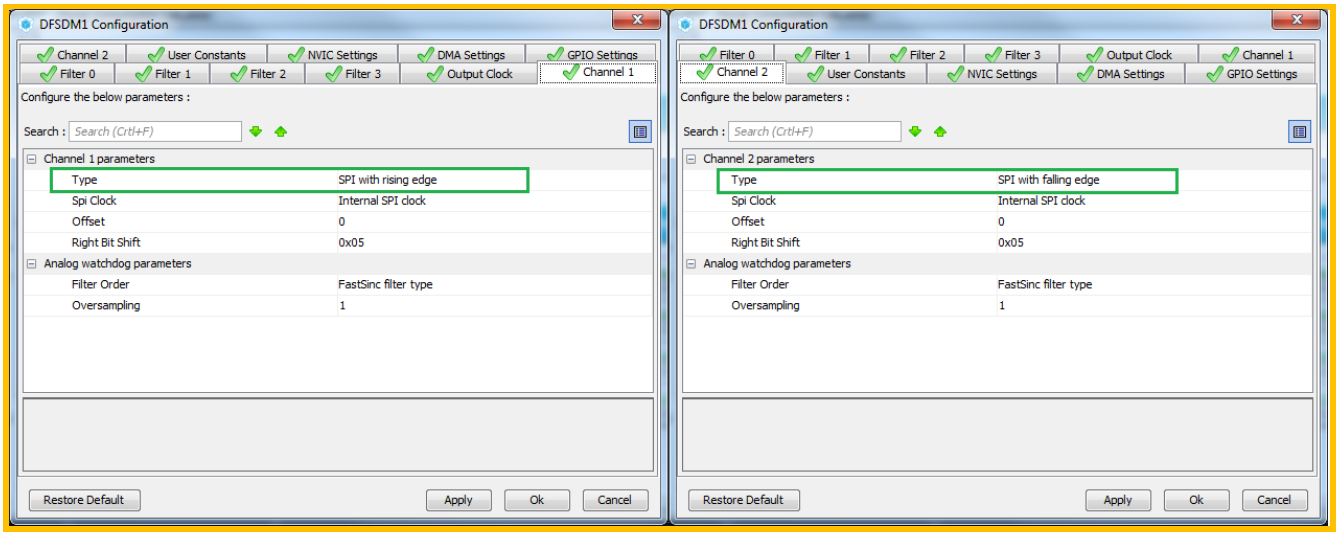
X-NUCLEO-CCA02M1	NUCLEO-L476RG	DFSDM
MIC_CLK	PC2	Clock out
MIC_PDM	PB14	Channel1 & Channel2

在 STM32CubeMX 中，选择 Channel1 中“ PDM/SPI Input from ch2 and internal clock ”和 Channel2 中“ PDM/SPI Input from ch2 and internal clock ”，并选择“CKOUT”，如下图所示，PC2、PB14 自动对应与 DFSDM 的 Clock out 和 Data In 功能脚。



b. **配置通道**

切换至“Configuration” 标签页，点击“Control\DFSDM”打开 DFSDM 配置界面。由于选择了通道 1 和通道 2，这里可以对这两个通道进行配置。配置情况如下图。



Type：配置数据读取时刻。SPI with rising edge 在时钟的上升沿读取数据；SPI with falling edge 在时钟下降沿读取数据。

Spi clock：总线时钟源。Internal SPI clock 利用内部时钟，对应为 CKOUT 时钟。

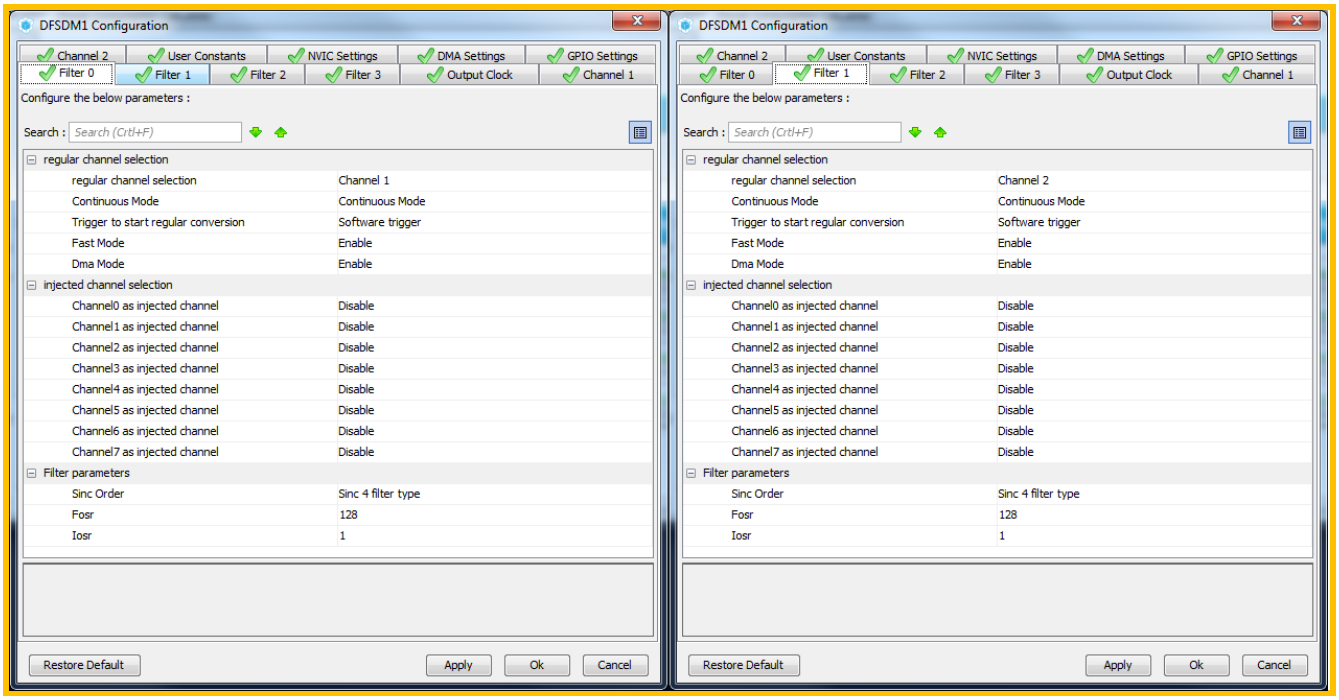
Offset：数据偏移量补偿。

Right Bit Shift : 右移位。右移位的确定，涉及到获取有效数据的位数，如“分辨率”小节中图所示。需要结合滤波器和积分器配置及分辨率需求进行确定。本文中，经过滤波器和积分器处理后输出数据分辨率为 29-bit，所以将右移位设置为 5，从而在 24 位数据寄存器中获得有效的 24-bit 数据。

Analog watchdog parameters : DFSDM 中模拟看门狗参数设置。应用例中没涉及到此功能，参数保持默认。

c. 配置滤波器

切换至“Filter0”/“Filter1”标签页，配置情况如下图。



Regular channel selection: 数据来源。选择与外部麦克风连接的通道。

Continuous mode: 转换模式。选择连续转换模式。

Trigger to start regular conversion: 启动转换的触发源。

Fast mode: 快速模式。在连续转换数据源来自于同一个通道时可启用，能够提高转换速度。本文应用例满足条件，使能 Fast mode。

DMA mode : DMA 模式。如果无法使能，需参考后面“配置 DMA”小节，先完成对 DMA 参数的配置。

Inject channel selection: 注入通道选择。应用例中没涉及到此功能，参数保持默认。

Sinc Order: 滤波类型。

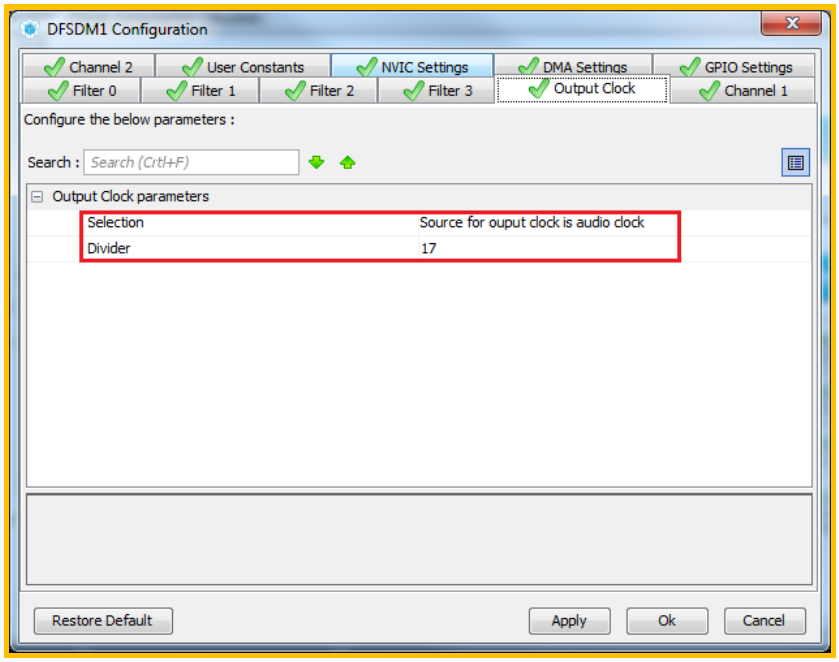
Fosr: 滤波器过采样率。

Iosr: 积分器过采样率。

其中，Sinc order, Fosr 和 Iosr 共同决定了积分器处理后的内部数据分辨率。根据“分辨率”小节，在如上配置时，积分器最大输出分辨率为 29 位。

d. 配置时钟

切换至“Output Clock” 标签页，配置情况如下图。



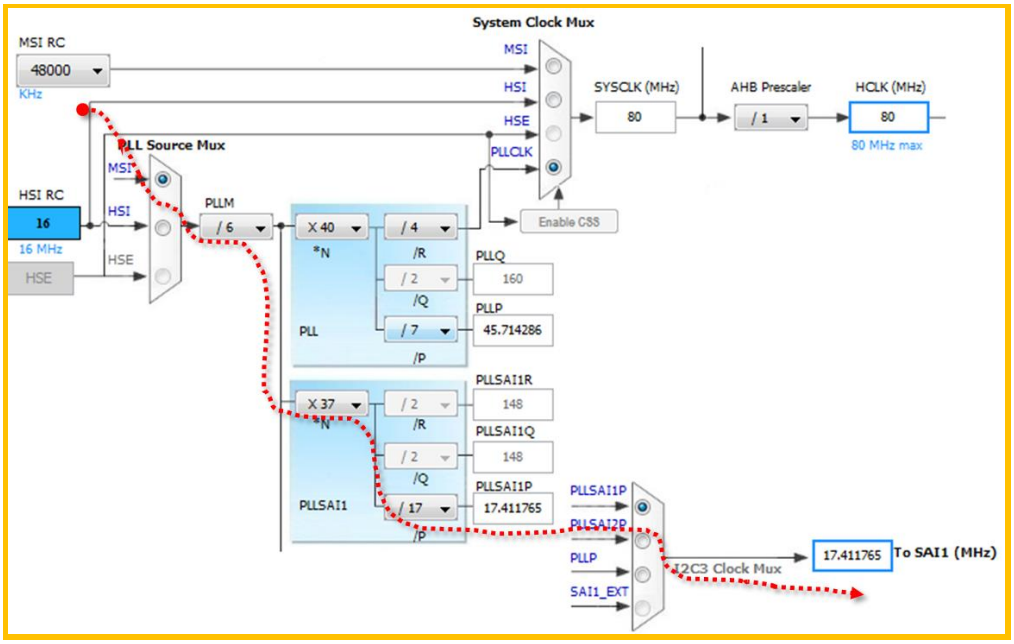
Selection: CKOUT 时钟源选择。

Divider: 时钟预分频因子。CKOUT = Audio clock / divider。

其中，Audio clock 来源于 SAI1 时钟，如下图配置，SAI1 时钟配置为 17.411765MHz。CKOUT 输出时钟频率为 1.0242MHz。

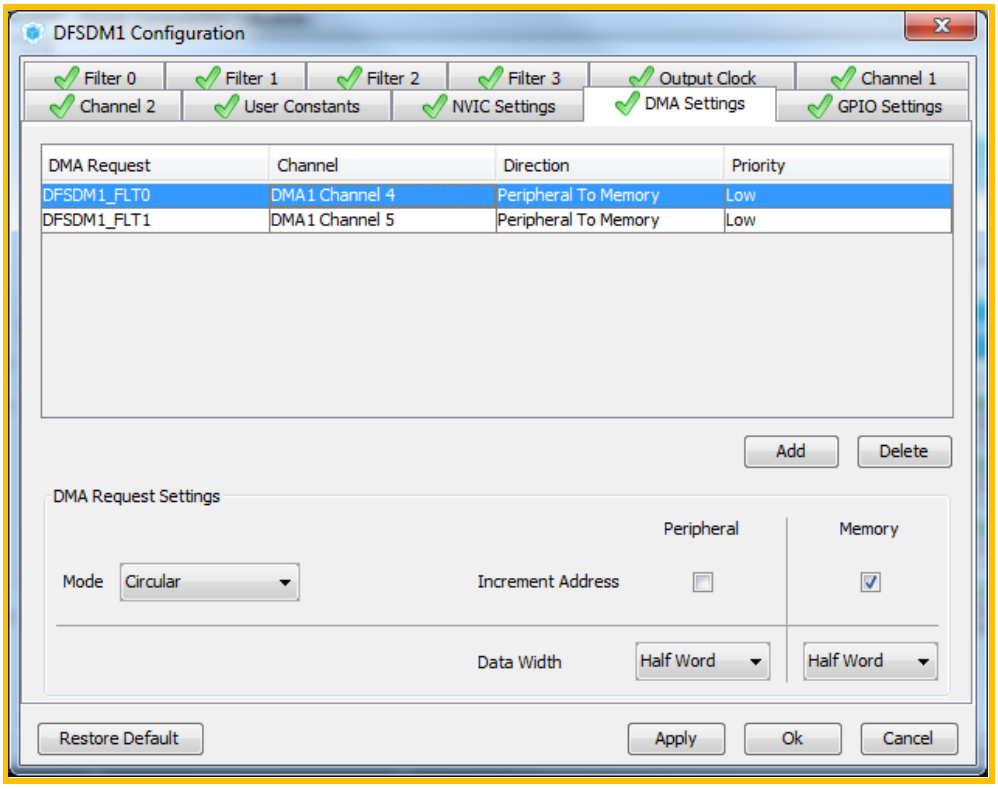
在 Fast mode 情况下，采样率 Fs 如下：

$$FS = \frac{F\_CKOUT}{Fosr * Iosr} = \frac{1.0242}{128 * 1} MHz = 8KHz$$



e. **配置 DMA**

切换至“DMA Settings” 标签页，配置情况如下图。



DMA 配置中，选择 Circular 模式，可实现循环向数据 buffer 中填充采样数据。

Data Width 设置为 Half Word，以便实现只获取数据寄存器的高 16 位数据，实现 16-bit 分辨率数据采集。

f. **生成工程**

在 STM32CubeMX\Help 打开帮助文档 UM1718，参考文档，生成 IAR 工程。至此获得与硬件对应，支持两路麦克风，采样率为 8KHz，分辨率为 16-bit 的初始化软件工程。

g. **启动采样**

完成上述步骤后，即可调用 HAL 库中提供的函数，启动麦克风数据采集。建议利用 DMA 方式实现麦克风采集，占用 CPU 开销最小。下述为启动采样的实现例程。在例程中通过调用 HAL_DFSDM_FilterRegularMsbStart_DMA 启动采样，这个函数实现了利用 DMA 接收 DFSDM 的数据寄存器的高 16 位数据，并传输到分配的 Buffer 空间中。

注：HAL_DFSDM_FilterRegularMsbStart_DMA()的实现需要在 DMA 配置中将 Data Width 设置为 Half Word。

由于 DMA 采用 Circular 模式，采样数据向 buffer 空间的搬运连续进行，而半传输完成和传输完成中断回调函数被执行时，意味新的数据被填充至 Buffer 空间，用户可以在其中增加处理指令，完成利用 DFSDM 的麦克风数据采集应用的开发。

```
#define SAMPLE_FREQ             8000
#define BYTE_PER_SAMPLE         2
#define MICROPHEN_NUMBER        2
#define FRAME_NUMBER            2
//16bit sample resolution
#define BUF_LENGTH              (SAMPLE_FREQ/1000*MICROPHEN_NUMBER*FRAME_NUMBER)
…
/* Buffer 分配 */
int16_t Buf_Mic0[BUF_LENGTH];
```

```
int16_t Buf_Mic1[BUF_LENGTH];
…
main()
{
…
/*启动采样，在初始化完成后调用*/
HAL_DFSDM_FilterRegularMsbStart_DMA(&hdfsdm1_filter0,Buf_Mic0,BUF_LENGTH);
HAL_DFSDM_FilterRegularMsbStart_DMA(&hdfsdm1_filter1,Buf_Mic1,BUF_LENGTH);
…
While (1){}
…
}
…
/* I2S DMA 回调函数 */
void HAL_DFSDM_FilterRegConvHalfCpltCallback(DFSDM_Filter_HandleTypeDef *hdfsdm_filter)
{
    //半传输完成，可在此添加对采样数据的处理
}
void HAL_DFSDM_FilterRegConvCpltCallback(DFSDM_Filter_HandleTypeDef *hdfsdm_filter)
{
    //传输完成，可在此添加对采样数据的处理
}
```

除此之外，还可以调用 HAL_DFSDM_FilterRegularStart_DMA 启动采样，而这种实现获取的整个 32 位的数据寄存器内容，获取数据需要利用软件指令将低 8 位，非采样数据移除。同时这种实现，需要在 DMA 配置中，将 Data Width 配置为 Word。

五 小结

本文虽然尽可能详细的介绍在 PDM 麦克风采集中，DFSDM 作用及实现步骤。但并没有涉及到所有 DFSDM 支持功能的介绍，而是侧重于 DFSDM 在 PDM 麦克风采集中涉及的功能介绍，更多 DFSDM 功能介绍可以通过参考手册了解。另外不同系列 STM32 的 DFSDM 支持情况会略有差异，需以对应型号的参考手册为准。

相关文档

UM1900	Getting started with the digital MEMS microphone expansion board based on MP34DT01-M for STM32 Nucleo
UM0351	STM32L4x5 and STM32L4x6 advanced ARM®-based 32-bit MCUs
UM1718	STM32CubeMX for STM32 configuration and initialization C code generation

重要通知 – 请仔细阅读

意法半导体公司及其子公司（“ST”）保留随时对ST 产品和/ 或本文档进行变更、更正、增强、修改和改进的权利，恕不另行通知。买方在订货之前应获取关于ST 产品的最新信息。ST 产品的销售依照订单确认时的相关ST 销售条款。

买方自行负责对ST 产品的选择和使用， ST 概不承担与应用协助或买方产品设计相关的任何责任。

ST 不对任何知识产权进行任何明示或默示的授权或许可。

转售的ST 产品如有不同于此处提供的信息的规定，将导致ST 针对该产品授予的任何保证失效。

ST 和ST 徽标是ST 的商标。所有其他产品或服务名称均为其各自所有者的财产。

本文档中的信息取代本文档所有早期版本中提供的信息。