



应用笔记

ACM32H5 系列芯片
MDAC 应用笔记

版本: V1.0

日期: 2025-01-09

上海航芯电子科技股份有限公司

1. 概述

本应用手册适用于 ACM32H5 系列芯片 MDAC 模块。它描述了与 MDAC 模块相关的设置和功能使用方法，以便在应用程序中进行优化设计。

本应用手册需要结合用户手册中对应的 MDAC 模块部分，DAC 模块部分、SDK 中相关的 demo 及 HAL 库驱动一起阅读。

2. MDAC 概述

多通道数字/模拟转换器 (MDAC, multi-channel DAC) 支持 12 个电压 DAC 和 4 个电流 DAC。电压 DAC (以下简称 VDAC) 可以将 12 位的数字数据转换为外部引脚上的电压输出。电流 DAC (以下简称 IDAC) 可以将 10 位数字数据转换为外部引脚的电流信号。其中通道 0~11 为电压型 DAC, 通道 12~15 为电流型 DAC。

2.1. 主要特性

支持 16 个通道 DAC, 其中 12 个通道为电压型, 4 个通道为电流型

每个转换器对应 1 个输出通道,

VDAC 支持 12 位分辨率

IDAC 支持 10 位分辨率

VDAC 支持校准功能

IDAC 出厂 Trim

VDAC 输入参考电压 VREF+

3. 应用使用说明

本章节内容将介绍 MDAC 的使用方法。

MDAC 电压工作范围 1.8~3.6V

VDAC VREF+电压范围 1.8~3.6V

VDAC buffer on 时电压输出范围 $0.2V \sim V_{REF+} + 0.2V$ 。

VDAC buffer off 时电压输出范围 $0V \sim V_{REF+}$ 。

IDAC 灌入电流范围 0~32mA。

以上数据详见数据手册

VDAC 电压输出计算公式: $V_{dac_out} = code/4095 * V_{REF+}$

IDAC 输出 sink 电流计算公式: $I_{dac} = code/1023 * I_{ref}$ (其中 I_{ref} 通过 I_{turn} 进行修调)

3.1. 参考源 V_{REF} 的设置

因为 VDAC 的电压输出是以为 V_{REF} 参考的。从 $0V \sim V_{REF}$ 的范围对应了码字 0~0xFFF。

V_{REF} 应该选择精度高纹波小的电压源，以提高 VDAC 输出的精度， V_{REF} 不能大于 V_{DDA}

3.2. 初始化

MDAC 的初始化主要是打开 MDAC 模块的时钟以及配置相应的 GPIO 管脚

MDAC 的初始化使用函数:

```
HAL_StatusTypeDef HAL_MDAC_Init(MDAC_HandleTypeDef *hmdac)
和 void HAL_MDAC_MspInit(MDAC_HandleTypeDef *hmdac)
```

需要在 acm32h5xx_hal_msp.c 中改写用户自定义的 HAL_MDAC_MspInit 函数。管脚需要配置成 GPIO_MODE_ANALOG_SWITCH_OFF, “模拟输入且 siwtch off 模式”, 详见 hal_gpio.h

3.3. 通道配置

在 MDAC 进行输出之前，需要对相应的通道进行配置，包括：

- VDAC 输出 buffer 是否打开
- VDAC 是否输出到 PAD
- VDAC 输出是否开启采样保持模式
- VDAC 的校准方式
- IDAC 的档位调节
- VDAC 和 IDAC 输出使能开启

VDAC 的通道配置使用函数:

```
HAL_StatusTypeDef HAL_MDAC_ConfigVoltageChannel(MDAC_HandleTypeDef* hmdac, uint32_t Channel,
VDAC_ChannelConfigTypeDef* Config)
```

IDAC 的通道配置使用函数:

```
HAL_StatusTypeDef HAL_MDAC_ConfigCurrentChannel(MDAC_HandleTypeDef* hmdac, uint32_t Channel,
IDAC_ChannelConfigTypeDef* Config)
```

例：

```
typedef struct
{
    FunctionalState SelfCalibrateEn;

    FunctionalState SampleEn;          /*!< Specifies voltage DAC sample and hold mode.
                                         This parameter can be ENABLE or DISABLE */
    uint32_t BufferMode;               /*!< Specifies if voltage DAC output buffer is enabled or not.
                                         This parameter can be MDAC_VDAC_BUFFER_ENABLE or
                                         MDAC_VDAC_BUFFER_DISABLE */
}VDAC_ChannelConfigTypeDef;
```

```
void MDAC_VoltageDAC_ChannelConfig(uint8_t channel)
{
    VDAC_ChannelConfigTypeDef config;

    config.BufferMode = MDAC_VDAC_BUFFER_DISABLE;    //buffer 输出关闭
    config.SelfCalibrateEn = DISABLE;                //DISABLE 表示使用 Efuse 中校准值进行校准
    config.SampleEn = ENABLE;                        //开启采样保持模式
    HAL_MDAC_ConfigVoltageChannel(&MDAC_Handle, channel, &config);
}
```

3.3.1. VDAC 的校准

VDAC 的校准有两种方式，一种是采用自校准方式，另一种是通过出厂配置的 Efuse 值进行校准。

自校准函数：

```
HAL_StatusTypeDef HAL_MDAC_VoltageSelfcalibrate(MDAC_HandleTypeDef *hmdac, uint32_t Channel)
```

自校准完成后，会将合适的 trim 值写入到对应 VDAC 通道的 VDACx_CR 的 OTRIM 字段

采用出厂 Efuse 值校准是读取 Efuse 对应的 trim 字段，将该值写入到对应 VDAC 通道的 VDACx_CR 的 OTRIM 字段

3.3.2. IDAC 的校准

```
void MDAC_CurrentDAC_ChannelConfig(uint8_t channel)
{
    IDAC_ChannelConfigTypeDef config;
    config.ITurn = 6; //ITURN_FACTORY;
    HAL_MDAC_ConfigCurrentChannel(&MDAC_Handle, channel, &config);
}
```

IDAC 的校准通过配置 config 变量的 ITurn 字段，ITurn 的修调范围为 0~31。

配置 ITurn 的值小于 32 时，将会使用用户自定义的修调值进行配置对应 IDAC 通道的 IDACx_CR 的 ITURN 字段。

如果设置 ITurn 为 ITURN_FACTORY，将会读取 Efuse 相应的 trim 字段来配置对应 IDAC 通道的 IDACx_CR 的 ITURN 字段。

3.4. 通道输出

需要注意的是，MDAC 的控制输出（即 DACx_DOR）的 16 个通道在硬件上是统一编号的，前 12 个为 VDAC，后 4 个为 IDAC。但是软件上是把 MDAC 的输出通道根据 VDAC 和 IDAC 区分开来的。VDAC 的输出通道编号是从 0 到 11 进行编号，IDAC 的输出通道也是从 0 开始到 3 进行编号的（对应的硬件通道则是 12~15）。

3.4.1. VDAC 的电压输出

如果 VDAC 输出的通道在配置时关闭的 buffer，那么将无法直接驱动负载，需要外接缓冲器 buffer（例如 TI 的 TLVx333）才能驱动负载。

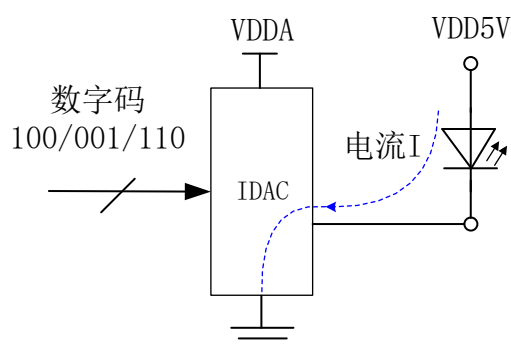
VDAC 的输出使用函数：

```
HAL_MDAC_SetVoltageValue(&MDAC_Handle, VDAC_CHANNEL_0, 0xFFF); //VDAC 通道 0 输出最大码字 0xFFF
```

$$VDAC \text{ output voltage} = \left(\frac{VREFP}{4095} \right) * DOR$$

3.4.2. IDAC 的电流灌入

这里讲的 IDAC 的输出其实是从电源端灌入多少毫安的电流到 IDAC 管脚。



IDAC 的典型应用电路

IDAC 的输出使用函数：

```
HAL_MDAC_SetCurrentValue(&MDAC_Handle, IDAC_CHANNEL_0, 0x3FF); //IDAC 通道 0 灌入最大码字电流
```

$$IDAC \text{ sink current} = \left(\frac{I_{ref}}{1023} \right) * DOR$$

其中 I_{ref} 是通过 IDAC_CR.ITURE 进行修调后的满量程电流。IO 接 3.3V 和 5V 电源，对 sink 电流的影响很小，大概在 0.1% 左右。

4. 版本历史

版本	日期	作者	描述
V0.1	2024-11-27	陆建钢	First release
V0.2	2024-12-11	陆建钢	审阅后更新
V1.0	2024-01-09	陆建钢	修改第三章描述不准确的地方

版权声明

本文档的所有部分，其著作权归上海航芯电子科技有限公司（简称航芯科技）所有，未经航芯科技授权许可，任何个人及组织不得复制、转载、仿制本文档的全部或部分组件。本文档没有任何形式的担保、立场表达或其他暗示，若有任何因本文档或其中提及的产品所有资讯所引起的直接或间接损失，航芯科技及所属员工恕不为其担保任何责任。除此以外，本文档所提到的产品规格及资讯仅供参考，内容亦会随时更新，恕不另行通知。

联系我们

公司：上海航芯电子科技有限公司
地址：上海市闵行区合川路 2570 号科技绿洲三期 2 号楼 702 室
邮编：200241
电话：+86-21-6125 9080
传真：+86-21-6125 9080-830
Email： service@aisinochip.com
Website： www.aisinochip.com