



介绍

本应用手册适用于需要 ACM32G103 系列芯片 ADC 模块。它描述了与 ADC 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。

本应用说明应与相关的用户手册、数据表一同阅读。

1. ADC 模块初始化

1.1. 主要特性

- 12 位分辨率，可配置 10 位、8 位或 6 位分辨率
- 转换速率最高可达 3MSPS
- 20 个可用通道，包括外部信号源和内部信号源
 - 内建 BGR 连接到 ADC1
 - 温度传感器连接到 ADC1
 - VBAT 连接到 ADC1
 - The OPA1/2 内部输出连接到 ADC1
 - The OPA2/3 内部输出连接到 ADC2
- 支持单端信号转换和差分信号转换
- 规则转换结束、注入转换结束和发生模拟看门狗事件时产生中断
- 支持单次、连续转换模式
- 支持间断转换模式
- 支持双重模式（两个 ADC 设备）
- 双 ADC 支持交替触发模式
- 最多支持 16 个规则通道组和 4 个注入通道组
- 采样时间可以按通道分别编程
- 规则转换和注入转换均有外部触发选项
- 规则通道转换期间有 DMA 请求产生
- AHB 总线便于系统集成，同时实现高速的读写操作
- 支持过采样：16 位数据寄存器，过采样率支持 2~256 倍
- 数据寄存器可配置数据对齐方式

1.2. 初始化配置

我们首先需要定义需要使用的 ADC 通道数量，以及具体的 ADC 通道号。确定需要使用到的工作模式（独立模式、双 ADC 模式、单端、差分或者是否支持 DMA 等），设置 ADC CLK，根据需求进行初始化配置。下面将描述具体的配置方法和过程。

首先定义一个 ADC 的总结构体变量，例如：

```
ADC_HandleTypeDef ADC_Handle;
```

定义一个 ADC 的通道设置结构体变量，例如：

```
ADC_ChannelConfTypeDef ADC_ChannelConf;
```

以及定义一个 ADC 转换结果 buffer，例如：

```
#define BUFFER_LENGTH    (21)
```

```
uint32_t gadcBuffer [BUFFER_LENGTH];
```

1.2.1. 基本配置

- ADC 时钟分频：

`ADC_Handle.Init.ClockPrescaler = ADC_CLOCK_DIV8;` //这里设置的是 ADC 模块的工作时钟分频，时钟源来自 HCLK，经过 DIV 分频得到，ADC CLK 最大不得超过 48MHZ。

- 数据分辨率：

`ADC_Handle.Init.Resolution = ADC_RESOLUTION_12B;` //可配置 12 位、10 位、8 位或 6 位。

- 数据对齐：

`ADC_Handle.Init.DataAlign = ADC_DATAALIGN_RIGHT;` //左对齐或右对齐。

- 是否支持连续模式：

`ADC_Handle.Init.ConvMode = DISABLE;`

- 是否支持间断模式：

`ADC_Handle.Init.DiscontinuousConvMode = DISABLE;`

- 间断模式通道计数：

`ADC_Handle.Init.NbrOfDiscConversion = 1;` //间断模式通道计数，最大为 8。

- 触发模式：

`ADC_Handle.Init.ExternalTrigConv = ADC_SOFTWARE_START;` // ExternalTrigConv 为 32bit 无符号数，其中 0~19 分别定义每个外部触发源，32 定义为软件触发，具体定义详情在例程的 `hal_adc.h` 中。

- 外部触发极性选择：

`ADC_Handle.Init.ExternalTrigConvEdge = ADC_EXTERNALTRIGCONVEDGE_NONE;` //外部触发边沿选择，为 32bit 无符号数，其中 0 表示禁止触发，1 表示上升沿触发，2 表示下降沿触发，3 表示上升沿和下降沿触发，具体定义详情在例程的 `hal_adc.h` 中

- 是否支持 DMA：

`ADC_Handle.Init.DMAMode = ADC_DMAMODE_DISABLE;`

- 溢出模式：
ADC_Handle.Init.OverMode = ADC_OVERMODE_DISABLE; //溢出时是否保留上次采样数据
- 是否支持过采样：
ADC_Handle.Init.OverSampMode = ADC_OVERSAMPMODE_DISABLE; //过采样使能
- 过采样率：
ADC_Handle.Init.Oversampling.Ratio = ADC_OVERSAMPLING_RATIO_2; //过采样率，为 32bit 无符号数，其中 0~7 分别定义过采样倍数 2~256 倍，具体定义详情在例程的 hal_adc.h 中。
- 过采样移位系数：
ADC_Handle.Init.Oversampling.RightBitShift = ADC_RIGHTBITSHIFT_2; //过采样移位系数，为 32bit 无符号数，其中 0~8 分别定义右移位系数，具体定义详情在例程的 hal_adc.h 中。
- 规则组过采样触发模式：
ADC_Handle.Init.Oversampling.TriggeredMode = 0; //一次触发进行 1 次或 N 次 ADC 转换
- 是否支持模拟看门狗：
ADC_Handle.Init.AnalogWDGEn = ADC_ANALOGWDGEN_DISABLE;
- 使能 ADC 通道：
ADC_Handle.Init.ChannelEn = ADC_CHANNEL_0_EN; //参数 ChannelEn 为 32bit 无符号数，其中 bit0~20 分别定义每个 ADC 通道使能，具体定义详情在例程的 hal_adc.h 中。ADC 对应的具体 GPIO 配置在底层函数 HAL_ADC_MspInit 中完成。
- 获取 ADC 寄存器基地址：
ADC_Handle.Instance = ADC;
- 获取 ADC 转换数据 buffer 指针：
ADC_Handle.AdcResults = & gadcBuffer [0];
- 其他参数设置按照例程默认设置
- 初始化 ADC 模块：
HAL_ADC_Init(&ADC_Handle);

1.2.2. 添加 ADC 转换通道

- 设置总通道数
ADC_Handle.ChannelNum = 1;
设置的通道数，必须与添加的通道总数一致，必须在初始化过程中已经设置该通道对应的 GPIO 为模拟通道且已经使能，否则可能导致采样数据丢失或者出错。
- 添加 ADC 通道

ADC_ChannelConf.Channel = ADC_CHANNEL_0; //参数 Channel 为 32bit 无符号数, 其中 0~19 分别定义每个 ADC 通道, 具体定义详情在例程的 hal_adc.h 中。

- 转换顺序

ADC_ChannelConf.Sq = ADC_SEQUENCE_SQ1; //参数 Sq 为 32bit 无符号数, 其中 1~16 分别定义在规则序列中的转换顺序, 具体定义详情在例程的 hal_adc.h 中。

- 采样时间

ADC_ChannelConf.Smp = ADC_SMP_CLOCK_320; //参数 Smp 为 32bit 无符号数, 其中 0~15 分别定义通道采样时间, 具体定义详情在例程的 hal_adc.h 中。

- 单端或差分模式

ADC_ChannelConf.SingleDiff = ADC_SINGLE_ENDED; //单端或差分模式

- 偏移通道选择

ADC_ChannelConf.OffsetNumber = ADC_OFFSET_NONE;

- 偏移量

ADC_ChannelConf.Offset = 0; //偏移量, 0~4095

- 偏移计算方式

ADC_ChannelConf.OffsetSign = ADC_OFFSET_SIGN_NEGATIVE; //转换结果加上或减去偏移量

- 偏移结果格式

ADC_ChannelConf.OffsetSaturation = ADC_OFFSET_SATURATION_DISABLE; //计算结果为有符号数或无符号数

HAL_ADC_ConfigChannel(&ADC_Handle,&ADC_ChannelConf);

每调用一次 HAL_ADC_ConfigChannel 函数, 添加一个 ADC 通道, 添加多个 ADC 通道需多次设置。

其中参数 Channel 是 ADC 通道号。参数 Sq 为规则序列的序列号, 规则序列号必须从 1 开始连续设置, 每个序列设置的 ADC 通道号可以是任意 ADC 通道号。

2. ADC 数据采集

2.1. 轮询方式

运行 HAL_ADC_Polling 获取 ADC 通道数据。轮询方式受限于 MCU 运行速度，无法支持太高采样频率，实际采样频率和 MCU 速度和应用有关，在应用中测试。

2.2. 中断方式

中断方式可以方便的处理注入采样，看门狗事件等。

初始化完通道，配置好中断回调函数。通过执行 HAL_ADC_Start_IT 函数启动规则通道采样，通过执行 HAL_ADC_InjectedStart_IT 函数启动注入通道采样（如果开启注入通道）

2.3. 外部触发方式

初始化完通道，配置好外部触发事件。启动通道采样，等待外部触发。

2.4. DMA 方式

初始化完通道，配置好 DMA 通道，运行 HAL_ADC_Start_DMA 获取 ADC 通道数据。

2.5. 数据格式

获取到的数据中包含通道号和采样结果，bit16~20 为通道号，bit0~15 为采样结果。设置为单端模式时，采样结果为 12bit 无符号数。设置为差分模式时，采样结果为 16bit 补码形式有符号数，符号位为最高位 bit15，在未使用过采样功能的情况下，仅低 11 位数据有效。

2.6. 结果计算

采样结果计算，根据采样结果 outdata 和 ADC 参考电压 VREF 计算采样结果电压 Vadc:

单端模式，outdata 和 Vadc 为无符号数： $V_{adc} = outdata * VREF / 4095$;

差分模式，outdata 和 Vadc 为有符号数： $V_{adc} = outdata * VREF / 2047$;

3. 应用例程说明

提供规则通道、注入通道、双 ADC 模式应用例程。且都可以通过修改 main 函数中，函数的输入参数来切换不同 demo。包含轮询、中断、外部触发、DMA 方式，以及注入同步、规则同步模式示例。

3.1. ADC 获取 VREFP

在 VREFN 接地的情况下，VREFP 的值即是用于计算最终 ADC 采样结果电压时所用的参考电压值。由于 VREFP 可能来自外部输入（见 5.2），因此不同的项目 VREFP 接的外部参考电压不一致；有时候对 ADC 的参考电压精度有更高的要求，而板级之间 VDDA 存在着差异。在这些情况下，我们可以采集内部稳定的带隙电压值（接入 BGR 通道），利用出厂时 VREFP 为 3V 采集的带隙电压校准值（存储在 NVR 区，见例程），来计算当前的 VREFP。具体见例程中函数 ADC_GetVrefP()。

注意：

- 1、如果用户有高精度的已知基准电压源或者对 ADC 采样精度要求不是特别高（电机应用通常要求较高），可以省略该步骤，直接使用已知电压值进行计算。

4. 常见问题汇总

4.1. VREFP 来源与用法

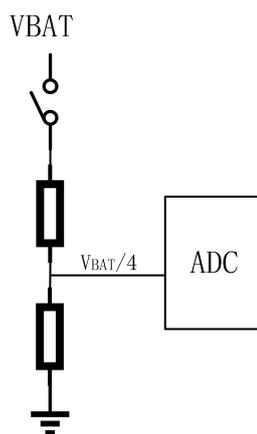
- 对于 ACM32G103 系列，只有 LQFP100/QFN48 封装芯片的 VREFP 引脚才引出；对于其他封装的芯片，VREFP 和 VDDA 在芯片内部进行了短接。
- VREFP 和 VDDA 内部短接的封装，VREFP 的电压等于 VDDA 的电压，ADC 的正向参考电压等于 VDDA 的电压；VREFP 和 VDDA 内部未短接的封装，ADC 的参考电压默认使用 VREFP 引脚所接的外部电压，也可以使用 ADC 模块内嵌 VREF 电压，详见 ADC_TSREF 寄存器。

4.2. 输入电压采用外部电阻分压方式对采样时间的影响

当采样输入是经由外部电阻分压输送到 ADC 引脚时，相当于有电阻串联到 ADC 内部的充电电路，充电时间会延长，相对应的采样时间也会延长，建议此时软件增大采样周期。

4.3. VBAT 分压电路

当 VBAT 为无负载电压时，ADC 输入端为 1/4 分压；如果 VBAT 带负载，如 VBAT 输入点的电压为电阻分压所得，如下图所示，那么外部对地电阻会和 ADC 内部的对地电阻并联，需要综合考虑。



联系我们

公司：上海航芯电子科技股份有限公司
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

版本维护

版本	日期	作者	描述
V1.0	2023-05-11	Aisinochip	初始版

本文档的所有部分，其著作权归上海爱信诺航芯电子科技有限公司（简称航芯公司）所有，未经航芯公司授权许可，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，航芯公司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。