Application Note

高精度 24bit 差分同步 ADC ADS131M08 与 MSPM0 的应用设计

Leon Yan *East China AA2 MCU FAE*

摘要

ADS131M08 是一款 8 通道、同步采样、24 位、Δ-Σ 模数转换器 (ADC)，具有宽动态范围、低功耗和电能测量特定功能，因此非常适合电能计量、功率计量和断路器应用。ADC 输入可以直接连接到电阻分压器网络或电源变压器来测量电压，或连接到电流互感器或 Rogowski 线圈来测量电流。单独 ADC 通道可以根据传感器输入进行独立配置。低噪声、可编程增益放大器 (PGA) 提供了从 1 到 128 的增益，以放大低电平信号。此外，该器件集成了通道间相位校准、偏移和增益校准寄存器，有助于消除信号链误差。

该器件集成了低漂移、1.2V 基准，成本可控，减小了 PCB 面积。数据输入、数据输出和寄存器映射中可选的循环冗余校验 (CRC) 可以确保通信完整性。

其应用覆盖了商业电表和住宅电表、断路器、保护继电器、电能质量监测仪、电池测试设备、电池管理系统等多个场景。

本应用文档针对 ADS131M08 与 MSPM0 系列微控制器（MCU）的工程化应用方案，系统的剖析了硬件接口、MSPM0 SPI 配置、ADS131M08 特有的寄存器与初始化流程、多通道数据读取等要点。本文档旨在为开发高精度、多通道同步测量系统的嵌入式工程师提供基础的技术指导。

内容

插图清单

1 ADS131M08 介绍

ADS131M08 为 TI 高精度 ADC 系列中 24 位分辨率 8 通道同步采样产品，其 24 位高精度分辨率以及内置可编程增益放大器（PGA）的核心特性，专为需要多通道精确相位对齐和动态范围的应用而生。

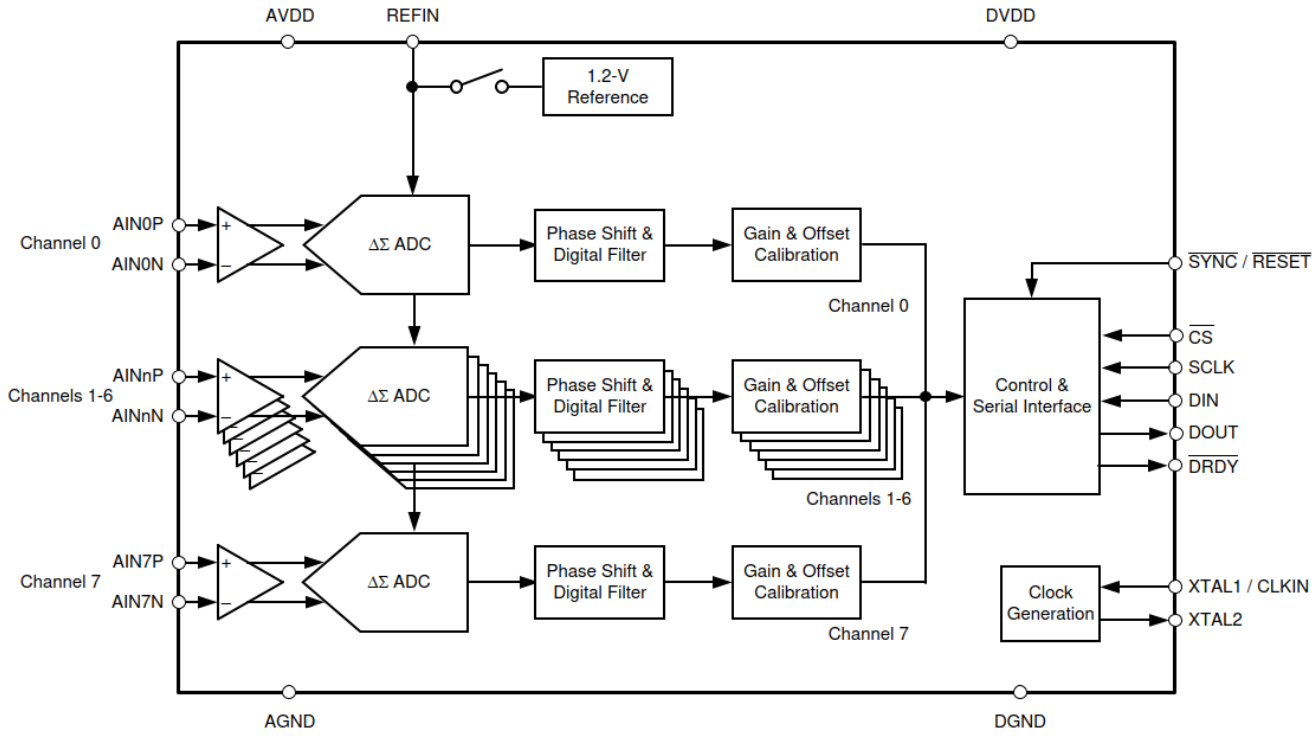

图 1-1. ADS131M08 框图

1.1 ADS131M08 主要应用

ADS131M08 具有宽动态范围、低功耗和电能测量特定功能，非常适合多种工业场合应用。

三相四线/三线电能计量：同时采样 A/B/C 三相电压与电流（共 6 个通道），并结合零序/中线电流检测（第 7、8 通道），是构建满足 IEC 标准 0.5S/0.2S 级三相电表的理想方案。同步采样特性确保了无相位误差的功率计算。

多传感器同步采集：在结构健康监测、工业控制中，常需要同时采集多个应变片、热电偶或压力传感器的信号。ADS131M08 的同步采样能力可精准捕捉同一物理瞬间所有传感器的状态，对于分析变量间的瞬时相关性至关重要。

电机控制与功率分析：可用于监测三相电机的三相相电压与相电流，实现精确的功率、效率和功率因数计算。8 个通道可扩展至同时监测直流母线电压等其他关键参数。

声学阵列与振动分析：多个麦克风或加速度传感器通过 ADS131M08 进行同步采集，结合后续信号处理算法，可用于声源定位、模态分析等。

相较于 M02/4/6 通道型号，ADS131M08 在通道集成度与数据统一性上提供了显著的系统级优势，避免了使用多片 ADC 带来的时钟同步与数据拼接的复杂度。

2 MSPM0 微控制器及其 SPI 外设简介

全新一代的 MSPM0 微控制器基于 Cortex M0+内核，主频覆盖 24MHz 到 80MHz，提供了丰富的配置和多种增强功能，在内部的模拟接口上，最高达到 4MHz 采样率的 12bit ADC，以及 12bit 的 1MHz DAC，为数据采集和模拟输出提供了高性能的保障。多个内部集成的 OPA 运放节省了外部器件，降低了 PCB 空间占有率。内部的多路 DMA 通道，为多个外设、存储的数据传输提供了高速通路，降低 MCU 内核的占有率。新引入的 Event 事件系统带来了更加灵活的事件 Post 和 Subscribe 机制，对于各个模块、外设的相互关联带来了更多的可选择性和灵活性。

2.1 MSPM0 SPI 接口

- 支持 MCLK/2 比特率，最高可达 32Mbps（在控制器和外设模式下）
- 可配置为控制器或外设
- 控制器和外设的可配置芯片选择
- 可编程时钟预分频器和比特率
- 可编程数据帧大小从 4 位到 16 位（控制器模式）和 7 位到 16 位（外设模式）
- 支持 PACKEN 功能，允许将两个 16 位 FIFO 条目打包为一个 32 位值以提高 CPU 性能
- 发送和接收 FIFO（四个条目，每个条目 16 位），支持 DMA 数据传输
- 支持 TI 模式、Motorola 模式和 National Microwire 格式

2.2 MSPM0 SPI 工作模式

MSPM0 SPI 接口支持的最高速率可达 32Mbps，支持 TI 模式、Motorola 模式和 National Microwire 格式，在与 ADS131M08 通信的系统中，需要工作在 Mode1（CPOL=0，CPHA=1），空闲时钟为低电平，数据在时钟下降沿（第二个边沿）变化，在时钟上升沿（第一个边沿）被采样。

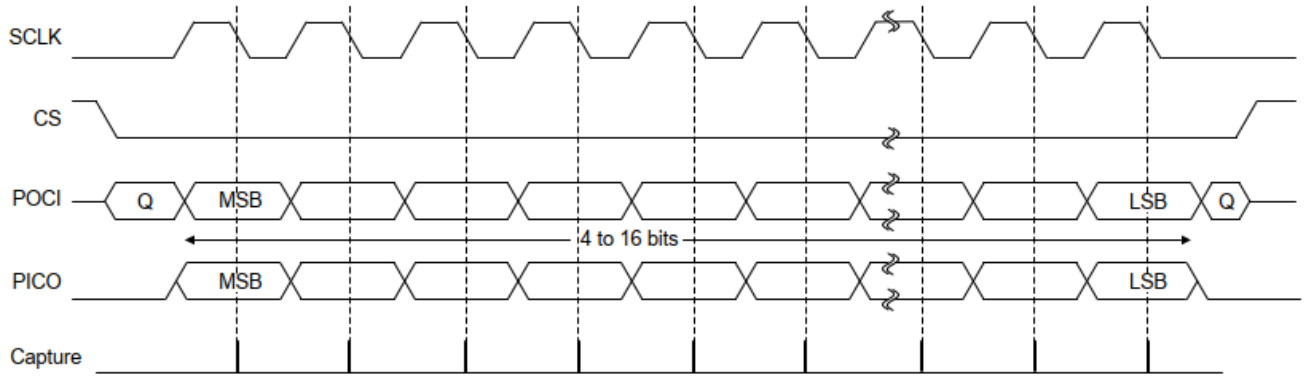

图 2-1. MSPM0 SPI 工作模式 SPO = 0，SPH = 1

3 ADS131M08 管脚功能介绍

在深入寄存器配置与硬件设计前，充分理解 ADS131M08 的核心管脚功能至关重要。作为一款高精度、多通道同步采样 ADC，其管脚布局与设计直接影响系统的性能、功耗与抗干扰能力。以下对各关键管脚组进行功能分解与设计要点说明。

3.1 ADS131M08 管脚功能

ADS131M08 采用独立的模拟与数字电源轨设计，以实现最佳的信噪比（SNR）并抑制数字噪声对模拟通道的干扰。

AVDD / AGND (模拟电源/地)：为内部模拟前端、PGA、调制器和基准电压缓冲器供电。典型电压为 3.3V。设计中要考虑采用 LDO 等低噪声电源供电，并靠近芯片使用高品质的退耦电容，AVSS 应与模拟地平面良好连接。

DVDD / DGND (数字电源/地)：为数字内核、SPI 接口和时钟电路供电。典型电压为 1.8V 至 3.6V，可与 MSPM0 的 I/O 电压匹配。设计中同样需要退耦，可以与系统数字电源共用。DGND 连接到数字地平面，可以设计单点与模拟地（AGND）相连，避免地环路噪声，这里需要注意的是，如果单点接地的布局不合适，反而会带来额外噪声。

模拟输入管脚：AINxP / AINxN ADS131M08 支持最多 8 路全差分输入通道（AIN0P/N 至 AIN7P/N）。全差分输入结构能有效抑制共模噪声（如 50/60Hz 工频干扰），提供比单端输入更高的动态范围和抗扰度。

注意输入范围由 PGA 增益（1 至 128）和参考电压决定，需确保传感器输出信号在此范围内，避免削波。

输入阻抗方面，ADS131M08 输入阻抗为 330K 欧，可以在每对差分输入端串联小电阻（如 10-100Ω）并并联 TVS/ESD 保护器件，以限制瞬态电流并提高 ESD 性能。布局上差分对走线尽量做到等长、等距、紧密耦合，并远离高速数字信号线（如 SCLK）。

时钟管脚：CLKIN ADS131M08 需要一个外部主时钟来驱动其调制器和数字滤波器。典型时钟频率为 2.048 MHz、4.096 MHz 或 8.192 MHz（对应不同的输出数据速率 ODR）。时钟占空比需接近 50%，且抖动（Jitter）应尽可能低，因为时钟抖动会直接转换为 ADC 的噪声。也可以与 MSPM0 CLKOUT 的配合，为 ADS131M08 提供稳定、同步的时钟源，确保 ADC 与 MCU 的时钟同步，在多器件级联时可以做到采样同步。

SPI 接口管脚：SCLK, MOSI, MISO, CS SPI 是配置 ADS131M08 寄存器和读取转换数据的数字接口，SPI 接口电压由 DVDD 决定，需确保与 MSPM0 主控的 I/O 电平兼容。采用标准 4 线 SPI 连接，支持 SPI 模式 1（CPOL=0, CPHA=1）。需注意，CS 信号在整个 SPI 通信期间必须保持低电平。

基准电压管脚：REFIN ADC 的精度直接依赖于参考电压的稳定性，ADS131M08 内部集成了 1.2V 基准，用户也可以通过 REFIN 引脚从外部输入参考源，同时配置 EXTREF_EN 寄存器位来使能外部输入路径。

DRDY（Data Ready）： DRDY 是 ADS131M08 中一个至关重要的状态指示信号管脚，在数据传输过程中起到核心的同步和控制作用。当一个新的完整数据帧（包含所有使能通道的转换结果）准备就绪，可以读取时，DRDYn 引脚会从高电平变为低电平。这样可以通知到 MCU 端，MCU 通过中断或轮询的模式查询到后，进行 ADC 结果读取。

SYNC/RESET 管脚： SYNC/RESET 是一个多功能管脚，当保持低电平超过 tw(RSL)（2048 CLKIN 周期：250uS）时间长度的情况下，会触发 ADS131M08 复位。在正常工作过程中，当 SYNC/RESET 产生一个下降沿脉冲，其脉冲宽度小于 tw(SYL)（2047 CLKIN 周期：249uS）但是大于一个 CLKIN 周期（122nS）的情况下，会触发同步采样。

3.2 ADS131M08 SPI 接口

实现与 ADS131M08 的稳定高效通信，需要对 SPI 协议细节进行分析理解。本章从几个要点进行深入分析，避免工程应用中的错误使用。

SPI 模式：ADS131M08 支持 SPI 模式 1（CPOL=0， CPHA=1），空闲时钟为低电平，数据在时钟下降沿（第二个边沿）变化，在时钟上升沿（第一个边沿）被采样。这是与 MSPM0 SPI 外设最匹配的模式。

时钟速率限制：最高 SCLK 频率为 25 MHz，超速通信将导致数据错位，简易在调试阶段先用低速率进行验证，然后再更改位高频率。

数据格式：所有 SPI 通信均为高位在前（MSB first）。命令字与数据帧的 LSB 对齐需在软件层面处理。

片选（CS）：CS 在整个通信期间必须保持低电平，包括命令、响应和数据帧的完整传输。过早拉高 CS 将中断通信并可能导致芯片状态机错误。这里需要留意一般不能使用 SPI 硬件片选信号，无法满足 CS 在整个通信期间必须保持低电平，需要使用 GPIO 口来操作 CS 信号。

4 ADS131M08 命令与数据帧结构详解

ADS131M08 包含 8 组操作命令，主要分为两部分：基本控制命令和寄存器访问命令。

COMMAND	DESCRIPTION	COMMAND WORD	RESPONSE
NULL	No operation	0000 0000 0000 0000	STATUS register
RESET	Reset the device	0000 0000 0001 0001	1111 1111 0010 1000
STANDBY	Place the device into standby mode	0000 0000 0010 0010	0000 0000 0010 0010
WAKEUP	Wake the device from standby mode to conversion mode	0000 0000 0011 0011	0000 0000 0011 0011
LOCK	Lock the interface such that only the NULL, UNLOCK, and RREG commands are valid	0000 0101 0101 0101	0000 0101 0101 0101
UNLOCK	Unlock the interface after the interface is locked	0000 0110 0101 0101	0000 0110 0101 0101
RREG	Read *nnn nnnn* plus 1 registers beginning at address *a aaaa a*	101*a aaaa annn nnnn*	*dddd dddd dddd dddd* or 111*a aaa annn nnnn* (1)
WREG	Write *nnn nnnn* plus 1 registers beginning at address *a aaaa a*	011*a aaaa annn nnnn*	010*a aaaa ammm mmmm* (2)

(1) When *nnn nnnn* is 0, the response is the requested register data *dddd dddd dddd dddd*. When *nnn nnnn* is greater than 0, the response begins with 111*a aaaa annn nnnn*, followed by the register data.

(2) In this case *mmm mmmm* represents the number of registers that are actually written minus one. This value may be less than *nnn nnnn* in some cases.

图 4-1. ADS131M08 Command 命令格式

4.1 基本控制命令：

基本控制命令为表格中的前 6 个，这些都是固定长度的二进制命令字。

- NULL (00000000000000)：空操作，通常用于维持通信或作为占位符
- RESET (00000000000001)：软件复位，恢复芯片到初始状态
- STANDBY (000000000100010)：进入低功耗待机模式
- WAKEUP (0000000000110011)：从待机模式唤醒到正常工作模式
- LOCK (00000101010101)：锁定接口，防止意外写入
- UNLOCK (00000110010101)：解锁接口，恢复正常操作

4.2 寄存器访问命令：

寄存器访问命令为表格中的最后 2 个，也是最重要的两个。

- RREG 命令格式：读命令标识 (101)

  地址字段 (a aaaa a)：7 位寄存器起始地址

  数量字段 (nnn nnnn)：7 位数量，实际读取寄存器数 = nnn nnnn + 1
- WREG 命令格式：写命令标识 (011)

  地址字段 (a aaaa a)：7 位寄存器起始地址

  数量字段 (nnn nnnn)：7 位数量，实际写入寄存器数 = nnn nnnn + 1

4.3 通信协议关键特性：

1. 批量寄存器访问：RREG 和 WREG 都支持批量操作，可以一次性读写多个连续地址的寄存器，这大大提高了数据传输效率，特别是在需要配置多个参数时。
2. 接口安全锁定机制：LOCK 命令锁定接口后，只能执行 NULL、UNLOCK 和 RREG 命令，这有效防止了意外写入导致的配置错误，UNLOCK 命令用于恢复完全访问权限。
3. 写入操作反馈：WREG 命令的响应中包含实际写入数量的确认信息，这为验证写入操作的成功性和完整性提供了机制，增强了通信的可靠性。
4. 工作模式管理：STANDBY/WAKEUP 命令对工作模式切换，支持低功耗待机模式，适用于电池供电应用。

5 ADS131M08 寄存器详解

ADS131M08 的寄存器配置是实现精确多通道测量的核心，以下是对其关键寄存器的详细说明，涵盖状态监控、配置、中断及数据读取等多个方面。

ADDRESS	REGISTER	RESET VALUE	BIT 15 / BIT 7	BIT 14 / BIT 6	BIT 13 / BIT 5	BIT 12 / BIT 4	BIT 11 / BIT 3	BIT 10 / BIT 2	BIT 9 / BIT 1	BIT 8 / BIT 0
DEVICE SETTINGS AND INDICATORS (Read-Only Registers)										
00h	ID	28xxh	RESERVED				CHANCNT[3:0]			
			RESERVED							
01h	STATUS	0500h	LOCK	F_RESYNC	REG_MAP	CRC_ERR	CRC_TYPE	RESET	WLENGTH[1:0]	
			DRDY7	DRDY6	DRDY5	DRDY4	DRDY3	DRDY2	DRDY1	DRDY0
GLOBAL SETTINGS ACROSS CHANNELS										
02h	MODE	0510h	RESERVED		REGCRC_EN	RX_CRC_EN	CRC_TYPE	RESET	WLENGTH[1:0]	
			RESERVED			TIMEOUT	DRDY_SEL[1:0]		DRDY_HiZ	DRDY_FMT
03h	CLOCK	FF0Eh	CH7_EN	CH6_EN	CH5_EN	CH4_EN	CH3_EN	CH2_EN	CH1_EN	CH0_EN
			RESERVED			OSR[2:0]			PWR[1:0]	
04h	GAIN1	0000h	RESERVED	PGAGAIN3[2:0]			RESERVED	PGAGAIN2[2:0]		
			RESERVED	PGAGAIN1[2:0]			RESERVED	PGAGAIN0[2:0]		
05h	GAIN2	0000h	RESERVED	PGAGAIN7[2:0]			RESERVED	PGAGAIN6[2:0]		
			RESERVED	PGAGAIN5[2:0]			RESERVED	PGAGAIN4[2:0]		
06h	CFG	0600h	RESERVED			GC_DLY[3:0]				GC_EN
			CD_ALLCH	CD_NUM[2:0]			CD_LEN[2:0]			CD_EN
07h	THRSHLD_MSB	0000h	CD_TH_MSB[15:8]							
			CD_TH_MSB[7:0]							
08h	THRSHLD_LSB	0000h	RESERVED				DCBLOCK[3:0]			

图 5-1. ADS131M08 寄存器-1

ID 和 STATUS 寄存器：在调试 SPI 通信过程中，可以通过读取这两个寄存器来简介判断 SPI 通信是否正常，同时可以看到当前芯片的状态。

MODE 寄存器（地址 0x02）：配置全局工作模式。这里包含了软件 RESET 位、通信帧长度设置、CRC 设置、DRDY 设置几个关键点。

CLOCK 寄存器（地址 0x03）：配置过采样率（OSR）和调制器功耗模式。

GAIN1 与 GAIN2 寄存器（地址 0x04, 0x06）：这是 ADS131M08 的关键配置。GAIN1 寄存器独立控制通道 0 至通道 3 的 PGA 增益。GAIN2 寄存器独立控制通道 4 至通道 7 的 PGA 增益。

CFG 寄存器（地址 0x05）：全局配置寄存器，配置全局斩波参数和电流检测功能参数。

THRSHLD_MSB 与 THRSHLD_LSB 寄存器：用来配置电流检测模式参数。

CHANNEL-SPECIFIC SETTINGS						
09h	CH0_CFG	0000h	PHASE0[9:2]			
			PHASE0[1:0]	RESERVED	DCBLK0_DIS0	MUX0[1:0]
0Ah	CH0_OCAL_MSB	0000h	OCAL0_MSB[15:8]			
			OCAL0_MSB[7:0]			
0Bh	CH0_OCAL_LSB	0000h	OCAL0_LSB[7:0]			
			RESERVED			
0Ch	CH0_GCAL_MSB	8000h	GCAL0_MSB[15:8]			
			GCAL0_MSB[7:0]			
0Dh	CH0_GCAL_LSB	0000h	GCAL0_LSB[7:0]			
			RESERVED			

图 5-2. ADS131M08 寄存器-2

CHx_CFG 寄存器：配置各通道独立参数，包含各个通道的 ADC 输入配置、DC 隔直和相位补偿。

CH0_OCAx_xxx，CH0_GCALx_xxx 寄存器：包含各个通道的 Offset 校准值、Gain 校准值。

6 ADS131M08 典型配置流程与要点

6.1 ADS131M08 初始化流程

- 上电复位后，为了确保可靠性，可以通过 SYNC/RESET 管脚进行二次复位，然后读取 ID 和 STATUS 寄存器以确认 SPI 通信和 ADS131M08 状态正常。
- 配置 MODE 寄存器，设定数据字长、CRC 和 DRDY 模式。
- 配置 CLOCK 寄存器，选择 OSR 和调制器功率模式以平衡精度与速度。
- 通过 GAIN1/GAIN2 寄存器为每个通道设定合适的 PGA 增益。
- 配置 CFG 寄存器，配置全局斩波参数和电流检测功能参数。
- 通过 CHx_CFG 寄存器逐个使能并配置所需通道，建议关闭未用通道。
- 加载校准过的每个通道的 Offset 校准值和 Gain 校准值。
- 关键配置完成后，可向 LOCK 寄存器写入锁定序列，防止运行时配置被意外更改。

此时 ADS131M08 已经开始正常工作，DRDY 管脚会发出对应 OSR 的低电平脉冲，意味着 ADC 的采样结果在周期性的更新。

6.2 ADS131M08 初始化流程中的要点

- 上电复位后，应遵循以下顺序对 ADS131M08 进行初始化，以确保各通道按预期工作。
- 时钟与接口准备：先确保 MSPM0 的 CLKOUT 或外置晶振已输出稳定时钟至 ADS131M08 的 CLKIN 引脚，并完成 MSPM0 SPI 外设的主模式配置（模式 1，适当速率如 2 MHz）。
- 配置 MODE 寄存器时设置 24 位字长，其它字长会使得通信变的复杂，如果 MCU 型号可以支持 32bit 字长传输的话可以考虑配置为 32 位字长。使能 CRC、配置 DRDY 行为。
- 配置 GAIN1 和 GAIN2 寄存器时需要根据每个通道的信号幅度，分别设定通道 0-3 和通道 4-7 的 PGA 增益。这是匹配不同传感器量程的关键步骤。
- GAIN1/GAIN2 的配置必须与硬件前端设计严格匹配，防止信号超量程或未能充分利用 ADC 动态范围。
- 配置 CLOCK 寄存器时根据目标输出数据率和精度要求，设置 OSR 和调制器功耗模式。采样率越高意味着 ENOB 会更低，同时 ADC 采样转换噪声更高。
- 注意 CFG 寄存器中如果使能斩波功能，最终的 ADC 采样率会比配置的 OSR 速率更低，这是正常现象，需要用示波器实际测量得到最终采样率。

7 MSPM0 SPI 配置和 API 函数实现

时钟配置：将 SPI 外设时钟（例如从系统时钟分频）配置为不超过 ADC 限制的速率。建议初始调试使用较低速率（如 2 MHz）。

SPI 模式：Clock Polarity 为 Low，Phase 为 Data captured on second clock edge，以满足模式 1 的配置。同时 SPI 的 Frame Size 配置为 8Bit。

CS 管脚：由于 ADS131M08 要求在整个通信过程中 CS 要持续保持低电平，例如在读取 8 通道 ADC 转换结果的过程中，会持续 30 个字节的读过程，这个过程中 CS 要全程保持低电平，MCU 的硬件 CS 信号是无法满足当前情况，所以需要软件操作 GPIO 来实现 CS 功能。在 MSPM0 的 SPI 配置中，需要选择三线制的 SPI 接口，这样 CS 就由软件来控制。

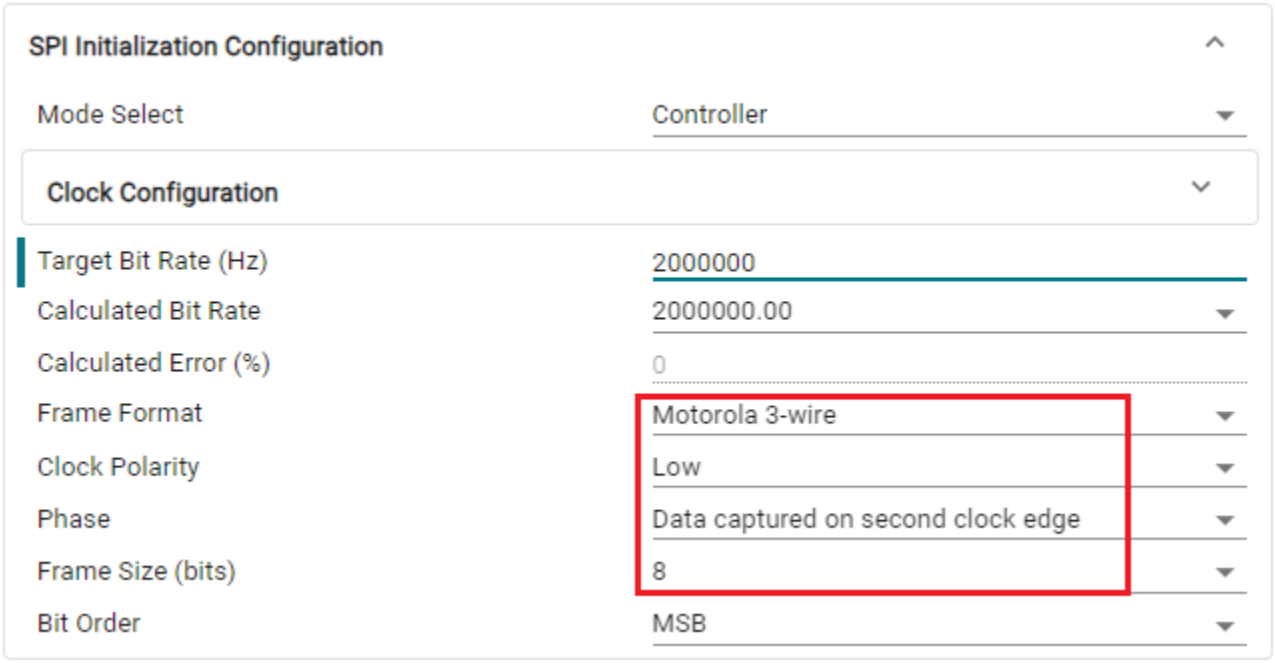

图 7-1. MSPM0 SPI 配置

7.1 MSPM0 SPI 收发函数

这里主要实现最底层的 SPI 数据收发，当前使用最基础的阻塞模式，用户可以后续考虑中断模式。

```
uint8_t spiSendReceiveByte(const uint8_t dataTx)
{
    /* --- INSERT YOUR CODE HERE ---
    * This function should send and receive single bytes over the SPI.
    * NOTE: This function does not control the /CS pin to allow for
    * more programming flexibility.
    */
  while (DL_SPI_isBusy(SPI_0_INST));
  DL_SPI_transmitData8(SPI_0_INST, dataTx);
  while (DL_SPI_isBusy(SPI_0_INST));
  return(DL_SPI_receiveDataBlocking8(SPI_0_INST));
}
```

7.2 ADS131M08 发送 Command 函数

这里进行 ADS131M08 Command 帧的发送，其发送的方式与寄存器读写不同，需要单独处理。

```
uint16_t sendSpecialCommand(uint16_t opcode)
{
```

```
    /* tAssert if this function is used to send any of the following opcodes */
    tAssert(OPCODE_RREG != opcode); /* Use "readSingleRegister()" */
    tAssert(OPCODE_WREG != opcode); /* Use "writeSingleRegister()" */
    tAssert(OPCODE_LOCK != opcode); /* Use "lockRegisters()" */
    tAssert(OPCODE_UNLOCK != opcode); /* Use "unlockRegisters()" */
    tAssert(OPCODE_RESET != opcode); /* Use "resetDevice()" */

     // Build TX and RX byte array
#ifdef ENABLE_CRC_IN
     uint8_t dataTx[8] = { 0 }; // 2 words, up to 4 bytes each = 8 bytes maximum
     uint8_t dataRx[8] = { 0 };
#else
     uint8_t dataTx[4] = { 0 }; // 1 word, up to 4 bytes long = 4 bytes maximum
     uint8_t dataRx[4] = { 0 };
#endif
     uint8_t numberOfBytes = buildSPIarray(&opcode, 1, dataTx);

      /* Set the nCS pin LOW */
    setCS(LOW);
     // Send the opcode (and crc word, if enabled)
    int i;
    for (i = 0; i < numberOfBytes; i++)
       {
         dataRx[i] = spiSendReceiveByte(dataTx[i]);
       }
     /* Set the nCS pin HIGH */
    setCS(HIGH);
     // Combine response bytes and return as a 16-bit word
     uint16_t adcResponse = combineBytes(dataRx[0], dataRx[1]);
   return adcResponse;
}
```

7.3 ADS131M08 写寄存器函数

这里进行 ADS131M08 寄存器写的操作，需要按照其标准格式进行帧格式的组合。

```
voidwriteSingleRegister(uint8_t address, uint16_t data)
{
      /* Check that the register address is in range */
    tAssert(address < NUM_REGISTERS);

      // (OPTIONAL) Enforce certain register field values when
      // writing to the MODE register to fix the operation mode
    if (MODE_ADDRESS == address)
      {
       data = enforce_selected_device_modes(data);
      }
     // Build TX and RX byte array
#ifdef ENABLE_CRC_IN
      uint8_t dataTx[12] = { 0 }; // 3 words, up to 4 bytes each = 12 bytes maximum
      uint8_t dataRx[12] = { 0 };
#else
      uint8_t dataTx[8] = { 0 }; // 2 words, up to 4 bytes long = 8 bytes maximum
      uint8_t dataRx[8] = { 0 };
#endif
      uint16_t opcodes[2];
      opcodes[0] = OPCODE_WREG | (((uint16_t) address) << 7);
      opcodes[1] = data;
      uint8_t numberOfBytes = buildSPIarray(&opcodes[0], 2, dataTx);
      // Send command
   spiSendReceiveArrays(dataTx, dataRx, numberOfBytes);
      // Update internal array
    registerMap[address] = data;

      // (RECOMMENDED) Read back register to confirm register write was successful
    readSingleRegister(address);
     // NOTE: Enabling the CRC words in the SPI command will NOT prevent an invalid W
}
```

7.4 ADS131M08 读寄存器函数

这里进行 ADS131M08 寄存器读的操作，需要按照其标准格式进行帧格式的组合。

```
uint16_t readSingleRegister(uint8_t address)
{
      /* Check that the register address is in range */
    tAssert(address < NUM_REGISTERS);

      // Build TX and RX byte array
#ifdef ENABLE_CRC_IN
      uint8_t dataTx[8] = { 0 }; // 2 words, up to 4 bytes each = 8 bytes maximum
      uint8_t dataRx[8] = { 0 };
#else
      uint8_t dataTx[4] = { 0 }; // 1 word, up to 4 bytes long = 4 bytes maximum
      uint8_t dataRx[4] = { 0 };
#endif
      uint16_t opcode = OPCODE_RREG | (((uint16_t) address) << 7);
      uint8_t numberOfBytes = buildSPIarray(&opcode, 1, dataTx);

      // [FRAME 1] Send RREG command
    spiSendReceiveArrays(dataTx, dataRx, numberOfBytes);

      // [FRAME 2] Send NULL command to retrieve the register data
    registerMap[address] = sendSpecialCommand(OPCODE_NULL);

    returnregisterMap[address];
}
```

7.5 ADS131M08 读 ADC 数据函数

当 MCU 端检测到 DRDY 的下降沿的时候，意味着当前配置的通道已经完成了采样和转换，即可进行 ADS131M08 结果读取的操作，这里可以直接连续读取最多 8 个通道的值。

```
boolreadData(adc_channel_data *DataStruct)
{
     int i;
        uint8_t crcTx[4] = { 0 };
        uint8_t dataRx[4] = { 0 };
        uint8_t bytesPerWord = getWordByteLength();

#ifdef ENABLE_CRC_IN
       // Build CRC word (only if "RX_CRC_EN" register bit is enabled)
       uint16_t crcWordIn = calculateCRC(&DataTx[0], bytesPerWord * 2, 0xFFFF);
       crcTx[0] = upperByte(crcWordIn);
       crcTx[1] = lowerByte(crcWordIn);
#endif

       /* Set the nCS pin LOW */
     setCS(LOW);
#if 1
       uint8_t dataRxTemp[32];
       uint8_t dataTxTemp[32] = {0};

    setCS(LOW);
    spiSendReceiveByteMulti(dataTxTemp, dataRxTemp, 30); //3 + 8*3 + 3 = 30
    setCS(HIGH);
       DataStruct->response = combineBytes(dataRxTemp[0], dataRxTemp[1]);
       DataStruct->channel0 = signExtend(&dataRxTemp[3]);
       DataStruct->channel1 = signExtend(&dataRxTemp[6]);
       DataStruct->channel2 = signExtend(&dataRxTemp[9]);
       DataStruct->channel3 = signExtend(&dataRxTemp[12]);
       DataStruct->channel4 = signExtend(&dataRxTemp[15]);
       DataStruct->channel5 = signExtend(&dataRxTemp[18]);
       DataStruct->channel6 = signExtend(&dataRxTemp[21]);
       DataStruct->channel7 = signExtend(&dataRxTemp[24]);

    return 1;
#endif
}
```

8 总结

本文针对 ADS131M08 与 MSPM0 微控制器的组合，为需要高精度、多通道同步数据采集的系统提供了一个高性价比的方案，其核心优势体现在：

集成度与同步性：单芯片提供 8 通道真同步采样，简化了系统架构，从根本上消除了多片 ADC 方案的时钟同步与数据对齐难题。

主从协同：可以选择使用 MSPM0 的 CLKOUT 为 ADS131M08 提供时钟源，在某些应用中以满足 MCU 与 ADC 的同步需求。

灵活的精度配置：独立的 GAIN1/GAIN2 寄存器允许对前 4 通道和后 4 通道进行差异化的增益设置，更好的适配三相计量等混合信号幅度的应用场景。

通过本文档来引导用户透彻理解 ADS131M08 的寄存器设置、SPI 接口通信、硬件管脚设计要点，为用户在产品中应用 MSPM0 和 ADS131M08 的组合方案打下基础。

9 参考文档

1. ADS131M08 8 通道、同步采样、24 位 ΔΣ ADC 数据表 (Rev. B)
2. MSPM0G350x 混合信号微控制器，具备 CAN-FD 接口 数据表 (Rev. C)
3. How Arm Cortex-M0+ MCUs optimize general-purpose processing, sensing and control
4. MSPM0 MCU 开发指南 (Rev. G)
5. SysConfig 如何助推嵌入式系统开发

重要通知和免责声明

TI“按原样”提供技术和可靠性数据（包括数据表）、设计资源（包括参考设计）、应用或其他设计建议、网络工具、安全信息和其他资源，不保证没有瑕疵且不做出任何明示或暗示的担保，包括但不限于对适销性、与某特定用途的适用性或不侵犯任何第三方知识产权的暗示担保。

这些资源可供使用 TI 产品进行设计的熟练开发人员使用。您将自行承担以下全部责任：(1) 针对您的应用选择合适的 TI 产品，(2) 设计、验证并测试您的应用，(3) 确保您的应用满足相应标准以及任何其他安全、安保法规或其他要求。

这些资源如有变更，恕不另行通知。TI 授权您仅可将这些资源用于研发本资源所述的 TI 产品的相关应用。严禁以其他方式对这些资源进行复制或展示。您无权使用任何其他 TI 知识产权或任何第三方知识产权。对于因您对这些资源的使用而对 TI 及其代表造成的任何索赔、损害、成本、损失和债务，您将全额赔偿，TI 对此概不负责。

TI 提供的产品受 TI 销售条款)、TI 通用质量指南 或 ti.com 上其他适用条款或 TI 产品随附的其他适用条款的约束。TI 提供这些资源并不会扩展或以其他方式更改 TI 针对 TI 产品发布的适用的担保或担保免责声明。 除非德州仪器 (TI) 明确将某产品指定为定制产品或客户特定产品，否则其产品均为按确定价格收入目录的标准通用器件。

TI 反对并拒绝您可能提出的任何其他或不同的条款。

最后更新日期：2025 年 10 月