XHSC 小华半导体
XIAOHUA SEMICONDUCTOR

32 位微控制器

TIM3 与 ADC 模块在电机 FOC 控制中单电阻采样联动操作说明

应用笔记

Rev1.0　2023 年 12 月

适用对象

产品系列	产品型号	产品系列	产品型号
F 系列	HC32F030 HC32F176 HC32F170 HC32F072 HC32F190 HC32F196 HC32F420	L 系列	HC32L130 HC32L136 HC32L176 HC32L170 HC32L072 HC32L073 HC32L190 HC32L196

声　明

★ 小华半导体有限公司（以下简称：“XHSC”）保留随时更改、更正、增强、修改小华半导体产品和/或本文档的权利，恕不另行通知。用户可在下单前获取最新相关信息。XHSC 产品依据购销基本合同中载明的销售条款和条件进行销售。

★ 客户应针对您的应用选择合适的 XHSC 产品，并设计、验证和测试您的应用，以确保您的应用满足相应标准以及任何安全、安保或其它要求。客户应对此独自承担全部责任。

★ XHSC 在此确认未以明示或暗示方式授予任何知识产权许可。

★ XHSC 产品的转售，若其条款与此处规定不同，XHSC 对此类产品的任何保修承诺无效。

★ 任何带有“®”或“™”标识的图形或字样是 XHSC 的商标。所有其他在 XHSC 产品上显示的产品或服务名称均为其各自所有者的财产。

★ 本通知中的信息取代并替换先前版本中的信息。

目　录

1 概述

本篇应用笔记主要介绍关于小华半导体通用 MCU 控制定时器模块与 ADC 模块在电机 FOC 控制中采用单电阻采样时的联动操作说明。

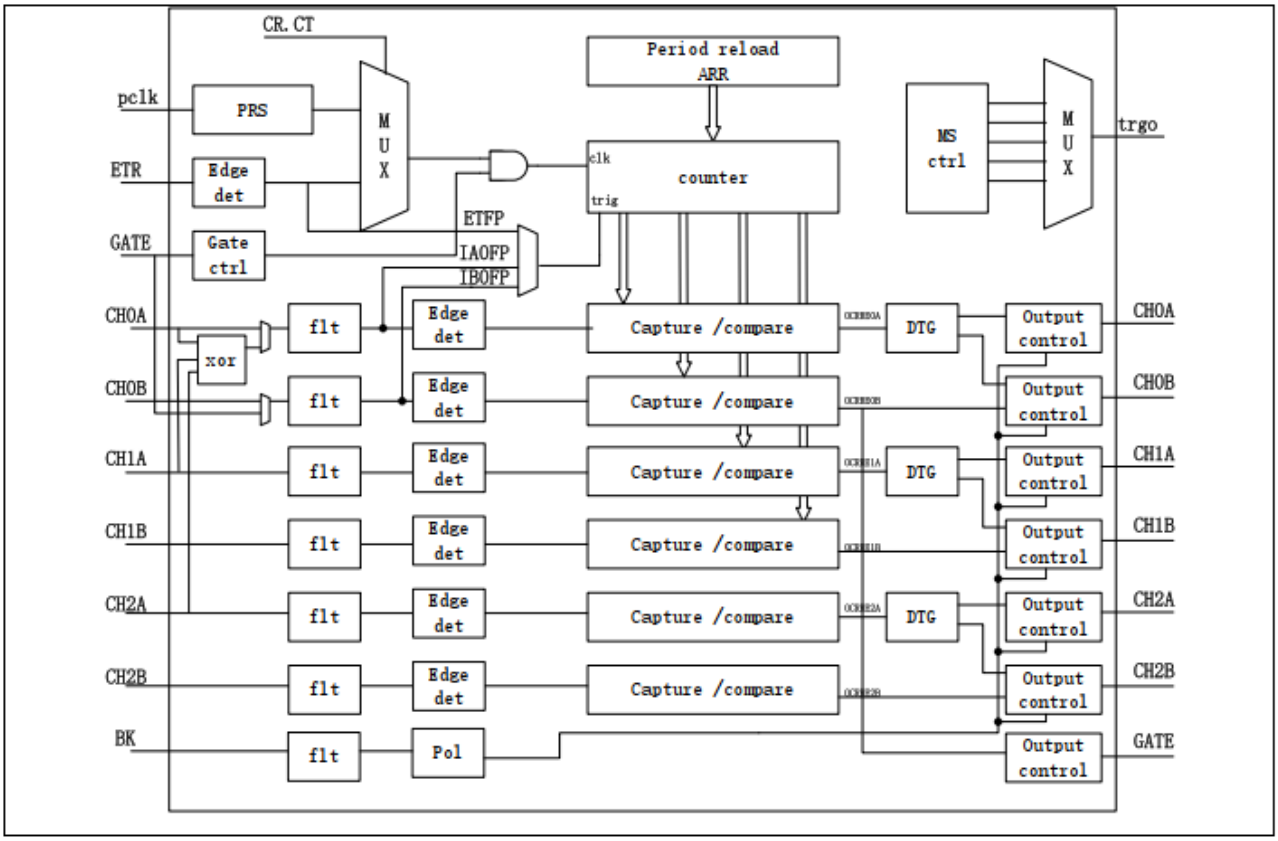
2 单电阻采样的实现方式

电机变频控制中，单电阻采样会遇到一些挑战，空间矢量脉宽调制器（SVPWM）在空间矢量的扇区边界和低调制区域的时候，会存在占空比两长一短和两短一长以及三个几乎一样长的时刻。这种情况下，有效矢量持续的时间少于电流采样时间，如果不进行移相，则会出现采样错误。当采取移相的方式实现时，FOC 变频控制 PWM 占空比的寄存器值发生改变，但是如果采用此寄存器进行 ADC 采样，此刻硬件开关的噪声会给采样带来较大的误差，影响控制精度及噪声。所以在单电阻采样时，需要考虑采样延时。此时通用 MCU 芯片需要有可以存放相位移动后的占空比值，并且能够产生触发条件触发 ADC 同步采样。

小华通用 MCU Cortex M0+系列系统有两类定时器可以实现 FOC 控制及同步延时触发 ADC 采样功能。一种采用专用比较器比较触发 ADC，详细操作参考另一篇 AN《TIMER4 与 ADC 模块在电机 FOC 控制中单电阻采样联动操作说明》，另一种采用同步定时器的方式比较触发 ADC。第二种方式的具体实现在下文中进行详细描述。

2.1 定时器 TIM3 同步比较触发 ADC 实现单电阻采样模式

小华半导体通用控制中的通用控制定时器(TIM3)是由 1 个计数单元和 6 个比较单元组成的定时器，支持 6 个独立 PWM 输出或 3 对互补 PWM 输出，可用于三相电机控制的定时器模块。该定时器支持三角波和锯齿波两种波形模式，可生成各种 PWM 波形；支持缓存功能；支持 TIM0/1/2/3 同步，支持 AOS 触发功能，支持 BRAKE（刹车）控制，Cortex M0 系列产品中搭载各 1 个单元的 Timer0/1/2/3。下图为 TIM3 的基本框图。从图中可以看出 timer 具有硬件插入死区的 3 路互补 PWM，用来输出 FOC 控制中的 PWM 输出。另外还有 VC 比较输出可以控制刹车功能，外部 BK 端口输入可控制刹车，系统 fail 可以控制刹车功能。通过 CR.BG 可以实现软件刹车功能，控制输出端口到设定的状态。此外，TRGO 输出信号可以连接到其他定时器的 ITR 信号进行 TIMER 互联，从而达到 timer 同步触发来实现 ADC 采样点需要延时触发的功能。



2.1.1 模式 2/3 互补 PWM 输出

在死区定时器模式（TIMx_DTR.DTEN=01）下，通用比较基准寄存器（OCCR*I）的值发生比较匹配产生的内部输出信号（in_op*I）和 PWM 死区控制寄存器（PDAR/ PDBR）的设定值通过时序偏移，以硬件方式实现互补 PWM 输出。在该模式下，TIM4_<t>_O*H 端口输出的极性与 in_op*I 相同，TIM4_<t>_O*L 端口输出的极性与 in_op*I 相反。

2.2 ADC 模块单电阻采样应用

12 位 ADC 是一种采用逐次逼近方式的模拟数字转换器，模拟输入通道可以任意组合成一个序列，一个序列可以进行单次扫描转换或连续扫描转换。支持对任意指定通道进行连续多次转换并对转换结果进行平均。ADC 模块还搭载模拟看门狗功能，对任意指定通道的转换结果进行监视，检测其是否超出用户设定的范围。

2.2.1 ADC 通道与触发源选择

扫描转换模式分为顺序扫描转换和插队扫描转换两种模式。两种模式各自单独工作时，均可连续对多个通道进行多次转换；当两种模式同时工作时，则优先对插队扫描配置的通道进行转换。

顺序扫描转换模式最多可进行 16 次连续转换，转换的总次数由 ADC_SQR2.CNT 进行配置；可配置所有 30 个通道进行转换，待转换通道由 ADC_SQRx.CHxMux 进行配置。该模式既可通过设置 ADC_SqrStart.Start 位启动也可通过设置 ADC_ExtTrigger0 的外部触发启动。

插队扫描转换模式最多可进行 4 次连续转换，转换的总次数由 ADC_JQR.CNT 进行配置；可配置所有 30 个通道进行转换，待转换通道由 ADC_JQR.CHxMux 进行配置。该模式既可通过设置 ADC_JqrStart.Start 位启动也可通过设置 ADC_ExtTrigger1 的外部触发启动。启动转换后，ADC 模块依次转换 CHxMux~CH0Mux 中配置的通道直到总转换次数完成。

而在实际应用中，电机控制实时性要求较高，电流采样要求实时反馈，所以建议选择插队采样做为单电阻采样的配置。

3 M0+系列单电阻采样应用配置说明

3.1 操作概述

小华半导体通用控制定时器模块在电机 FOC 单电阻采样控制中的操作流程为：

1) 产生 3 组互补带死区，带缓存的 PWM 信号写入比较配置寄存器，用于驱动三相电机。

2) 使能同步 TIME0/1/2 互联功能中的 CCRx 比较触发功能，选择立即生效，在特定时刻产生 ADC 触发信号，触发 ADC 对电流信号的采样。

3) 使能 ADC 转换完成触发 DMA 功能，配置 DMA 源地址，目的地址进行 ADC 输出搬运或者触发中断使能在中断中读取数据。

4) 启动 ADC，启动 TIM3 及同步 TIME0/1/2 计数，开始输出及采样（本文采用 TIM0 做为同步计数触发定时器）。

3.2 操作流程

3.2.1 互补带死区的 PWM 信号及同步单点比较 PWM 信号生成

1) 配置外设时钟，计数器计数值，计数模式，IO 状态
在 FOC 控制中，大部分应用场景中的载波选择三角波模式，生成具有目标周期频率的三角载波，产生计数上溢中断或计数下溢中断（以计数上溢中断为例）。

2) 配置比较输出电平，事件更新模式以及缓存使能
设定比较输出所对应的端口输出选择，比较输出信号来源，使能事件更新模式以及比较输出模式。

3) PWM 输出流程
设定 PWM 的输出模式（互补带死区模式）、PWM 输出极性是否翻转、GPIO 初始输出电平，死区的宽度等 PWM 输出信息。

4) 根据 1/2 步配置同步 TIM0，使能 TIM0 为单点触发比较模式，无 PWM 输出。

5) 配置 TIM3 为同步启动中的主输出模式，TIM0 配置为从输出模式，同步触发信息选择 TIM3。

3.2.2 ADC 触发信号生成

TIM3 模块运行时，根据单电阻采样时间点的特性，不能在 CCR 的比较点立即产生比较触发采样（采样有误差），而 TIM3 无专用比较器及触发事件，所以采用同步 TIM0，并采用 TIM0 的比较值寄存器装载 TIM3 比较占空比延时以后的比较值进行触发单电阻采样，在设定的计数方向上，需要在【0-周期值】的任意数值上匹配 ADC 采样两次，并且发出的两次占空比值的间距不能小于电流采样的时间。所以需要采用两路比较寄存器做为触发 ADC 事件的触发源。配置外设 ADC、TIM 时钟，使能 ADC 插队采样模式。

1) 配置 ADC 通道，采样时间；

2) 配置选择 ADC 的触发源为 TIM0；

3) 使能 DMA 或者中断接收采样值。

3.2.3 启动 TIM3 计数

启动 TIM3 计数，同步启动 TIM0 开始进行计数。

```
en_result_t Tim3_M23_Run(void)
{
    en_result_t enResult = Ok;
    M0P_TIM3_MODE23->M23CR_f.CTEN = TRUE;
    return enResult;
}
```

3.2.4 注意事项

1) 缓存功能建议开启，确保 FOC 控制中时序，防止产生未预期的问题。

2) ADC 顺序扫描与插队扫描不能同时进行，不要有冲突。

3) 可以根据采样电流重构方式的不同，更改上升沿及下降沿触发方式。

4) 可以根据驱动的极性对 PWM 输出的极性进行匹配，并且同步更改触发点。

5) 需要根据采样时钟及外部负载设定采样时间。

6) 关于 ADC 采样精度提升可以参考 AN《AN_如何提高 ADC 采样精度_Rev1.0》。

7) TIM3 同步 TIM0 的两路 PWM 比较触发采样（样例）模式下，需要用中断方式读取数据，并且需要软件计数来判断采样顺序。

8) 也可以采用 TIM3 同步 TIM0/1/2 中的两个 timer，用不同的 TIM 比较触发 ADC，此时可以通过 DMA 或者中断的方式读取数据，无需软件判断采样顺序。

4 样例代码

以下是介绍本 AN 基于 TIM3 及 ADC 模块在电机 FOC 控制中的相关外设配置。实例以 HC32F170 芯片为例。

4.1 M0+系列中 TIM3 输出死区 PWM 及同步 TIM0 触发单电阻采样样例配置

```
void App_Timer3Cfg(uint16_t u16Period, uint16_t u16DeadTimeCnt,uint8_t u8PeakIsrMaskCnt)
{
    uint16_t                        u16CntValue;
    uint16_t                        u16CHxACompare;
    stc_tim3_mode23_cfg_t           stcTim3BaseCfg;
    stc_tim3_m23_compare_cfg_t      stcTim3PortCmpCfg;
    stc_tim3_m23_adc_trig_cfg_t     stcTim3TrigAdc;
    stc_tim3_m23_dt_cfg_t           stcTim3DeadTimeCfg;
    stc_tim3_m23_master_slave_cfg_t stcTim3MasterSlavecfg;

    /*结构体初始化清零*/

    DDL_ZERO_STRUCT(stcTim3BaseCfg);
    DDL_ZERO_STRUCT(stcTim3PortCmpCfg);
    DDL_ZERO_STRUCT(stcTim3TrigAdc);
    DDL_ZERO_STRUCT(stcTim3DeadTimeCfg);

    /*Timer3 外设时钟使能*/

    Sysctrl_SetPeripheralGate(SysctrlPeripheralTim3, TRUE);

    /*TIM3  的模式 23 功能初始化  */

    stcTim3BaseCfg.enWorkMode       = Tim3WorkMode3;
    stcTim3BaseCfg.enCT             = Tim3Timer;
    stcTim3BaseCfg.enPRS            = Tim3PCLKDiv1;
    stcTim3BaseCfg.enPWMTypeSel     = Tim3ComplementaryPWM;
    stcTim3BaseCfg.enPWM2sSel       = Tim3DoublePointCmp;
    stcTim3BaseCfg.bOneShot         = FALSE;
    stcTim3BaseCfg.bURSSel          = FALSE;
    Tim3_Mode23_Init(&stcTim3BaseCfg);
    Tim3_M23_ARRSet(u16Period, TRUE);
    u16CHxACompare = 0;
    Tim3_M23_CCR_Set(Tim3CCR0A, u16CHxACompare);
    Tim3_M23_CCR_Set(Tim3CCR1A, u16CHxACompare);
    Tim3_M23_CCR_Set(Tim3CCR2A, u16CHxACompare);

    /*比较输出端口配置*/

    stcTim3PortCmpCfg.enCHxACmpCtrl     = Tim3PWMMode2;
    stcTim3PortCmpCfg.enCHxAPolarity    = Tim3PortPositive;
    stcTim3PortCmpCfg.bCHxACmpBufEn     = TRUE;
```

```
    stcTim3PortCmpCfg.enCHxACmpIntSel = Tim3CmpIntNone;
    stcTim3PortCmpCfg.enCHxBCmpCtrl   = Tim3PWMMode2;
    stcTim3PortCmpCfg.enCHxBPolarity  = Tim3PortPositive;
    stcTim3PortCmpCfg.bCHxBCmpBufEn = TRUE;
    stcTim3PortCmpCfg.enCHxBCmpIntSel = Tim3CmpIntNone;
    Tim3_M23_PortOutput_Cfg(Tim3CH0, &stcTim3PortCmpCfg);
    Tim3_M23_PortOutput_Cfg(Tim3CH1, &stcTim3PortCmpCfg);
    Tim3_M23_PortOutput_Cfg(Tim3CH2, &stcTim3PortCmpCfg);

    /*TIM 同步主从模式配置*/

    stcTim3MasterSlavecfg.enMasterSlaveSel = Tim3MasterMode;
    stcTim3MasterSlavecfg.enMasterSrc = Tim3MasterCTEN;
    Tim3_M23_MasterSlave_Set(&stcTim3MasterSlavecfg);

    /*TIM 死区及事件更新周期配置*/

    stcTim3DeadTimeCfg.bEnDeadTime        = TRUE;
    stcTim3DeadTimeCfg.u8DeadTimeValue   = u16DeadTimeCnt;
    Tim3_M23_DT_Cfg(&stcTim3DeadTimeCfg);
    Tim3_M23_SetValidPeriod(u8PeakIsrMaskCnt);
    u16CntValue = 0;
    Tim3_M23_Cnt16Set(u16CntValue);
    Tim3_ClearAllIntFlag();
    Tim3_Mode23_EnableIrq(Tim3UevIrq);
    EnableNvic(TIM3_IRQn, IrqLevel1, TRUE);
}

void App_Timer0Cfg(uint16_t u16Period,   uint8_t u8PeakIsrMaskCnt)
{
    uint16_t                                u16CntValue;
    stc_bt_mode23_cfg_t                     stcBtBaseCfg;
    stc_bt_m23_compare_cfg_t                stcBtPortCmpCfg;
    stc_bt_m23_adc_trig_cfg_t               stcBtTrigAdc;
    stc_bt_m23_dt_cfg_t                     stcBtDeadTimeCfg;
    stc_bt_m23_master_slave_cfg_t           stcBtMasterSlaveCfg;
    DDL_ZERO_STRUCT(stcBtBaseCfg);
    DDL_ZERO_STRUCT(stcBtPortCmpCfg);
    DDL_ZERO_STRUCT(stcBtTrigAdc);
    DDL_ZERO_STRUCT(stcBtDeadTimeCfg);
    DDL_ZERO_STRUCT(stcBtMasterSlaveCfg);
     Sysctrl_SetPeripheralGate(SysctrlPeripheralBaseTim, TRUE);

     /*TIM0  的模式 23 功能初始化  */

    stcBtBaseCfg.enWorkMode      = BtWorkMode3;
    stcBtBaseCfg.enCT            = BtTimer;
    stcBtBaseCfg.enPRS           = BtPCLKDiv1;
```

```
    //stcBtBaseCfg.enCntDir      = BtCntUp;
    stcBtBaseCfg.enPWMTypeSel   = BtComplementaryPWM;
    stcBtBaseCfg.enPWM2sSel     = BtSinglePointCmp;          //单点比较功能
    stcBtBaseCfg.bOneShot       = FALSE;
    stcBtBaseCfg.bURSSel        = FALSE;
    Bt_Mode23_Init(TIM0, &stcBtBaseCfg);
    Bt_M23_ARRSet(TIM0, u16Period, TRUE);
    Bt_M23_CCR_Set(TIM0, BtCCR0A, 200);
    Bt_M23_CCR_Set(TIM0, BtCCR0B, 1000);
    stcBtPortCmpCfg.enCH0ACmpCtrl   = BtPWMMode2;
    stcBtPortCmpCfg.enCH0APolarity  = BtPortPositive;
    stcBtPortCmpCfg.bCh0ACmpBufEn   = TRUE;
    stcBtPortCmpCfg.enCh0ACmpIntSel = BtCmpIntFall;
    stcBtPortCmpCfg.enCH0BCmpCtrl   = BtPWMMode2;
    stcBtPortCmpCfg.enCH0BPolarity  = BtPortPositive;
    stcBtPortCmpCfg.bCH0BCmpBufEn   = TRUE;
    stcBtPortCmpCfg.enCH0BCmpIntSel = BtCmpIntFall;
     Bt_M23_PortOutput_Cfg(TIM0, &stcBtPortCmpCfg);
     /*TIM0 同步从模式及触发 ADC 配置*/
    stcBtTrigAdc.bEnCH0ACmpTrigADC = FALSE;
    stcBtTrigAdc.bEnCH0BCmpTrigADC = TRUE;
    stcBtTrigAdc.bEnUevTrigADC = FALSE;
    stcBtTrigAdc.bEnTrigADC     = TRUE;
    Bt_M23_TrigADC_Cfg(TIM0, &stcBtTrigAdc);
    stcBtMasterSlaveCfg.enMasterSlaveSel =BtSlaveMode;
    stcBtMasterSlaveCfg.enSlaveModeSel = BtSlaveTrigMode;
    stcBtMasterSlaveCfg.enTsSel =BtTs4TIM3TRGO;
    Bt_M23_MasterSlave_Set(TIM0, &stcBtMasterSlaveCfg);
    Bt_M23_SetValidPeriod(TIM0,u8PeakIsrMaskCnt);
    u16CntValue = 0;
    Bt_M23_Cnt16Set(TIM0, u16CntValue);
    Bt_ClearAllIntFlag(TIM0);
}
```

4.2 M0+系列 ADC 单电阻采样

```
void App_AdcCfg(void)
{
    stc_adc_cfg_t                   stcAdcCfg;
    stc_adc_jqr_cfg_t               stcAdcJqrCfg;
    stc_adc_sqr_cfg_t               stcAdcSqrCfg;
    DDL_ZERO_STRUCT(stcAdcCfg);
    DDL_ZERO_STRUCT(stcAdcJqrCfg);
    Sysctrl_SetPeripheralGate(SysctrlPeripheralAdcBgr, TRUE);
    Gpio_SetAnalogMode(GpioPortA, GpioPin0);            //PA00 配置为模拟通道

    /*ADC 模块初始化配置*/
    Adc_Enable();
    M0P_BGR->CR_f.BGR_EN = 0x1u;//BGR 必须使能
    M0P_BGR->CR_f.TS_EN   = 0x0u;
    delay100us(1);
    stcAdcCfg.enAdcMode             = AdcScanMode;              //连续采样模式
    stcAdcCfg.enAdcClkDiv           = AdcMskClkDiv2;            //Adc 工作时钟  PCLK/2
    stcAdcCfg.enAdcSampCycleSel = AdcMskSampCycle8Clk;   //采样时钟  8 个周期
    stcAdcCfg.enAdcRefVolSel      = AdcMskRefVolSelAVDD;   //内部 AVDD
    stcAdcCfg.enAdcOpBuf            = AdcMskBufDisable;       //内部电压跟随器关闭
    stcAdcCfg.enInRef               = AdcMskInRefDisable;     //内部参考电压 Disable
    Adc_Init(&stcAdcCfg);
    /*配置插队扫描转换  */
    Adc_CfgJqrChannel(AdcJQRCH0MUX, AdcExInputCH0);
    stcAdcJqrCfg.bJqrDmaTrig = FALSE;     //禁止 JQR 转换触发 DMA
    stcAdcJqrCfg.u8JqrCnt = 1;              //转换起始通道(3-1 已在库函数内计算)
    Adc_JqrModeCfg(&stcAdcJqrCfg);        //配置插队扫描转换模式
    Adc_JqrExtTrigCfg(AdcMskTrigTimer0, TRUE);              //Timer0 触发插队扫描转换
    Adc_EnableIrq();                                        //使能 Adc 中断
    EnableNvic(ADC_DAC_IRQn, IrqLevel0, TRUE);              //Adc 开中断
}
```

版本修订记录

版本号	修订日期	修订内容
Rev1.0	2023/12/21	初版发布。