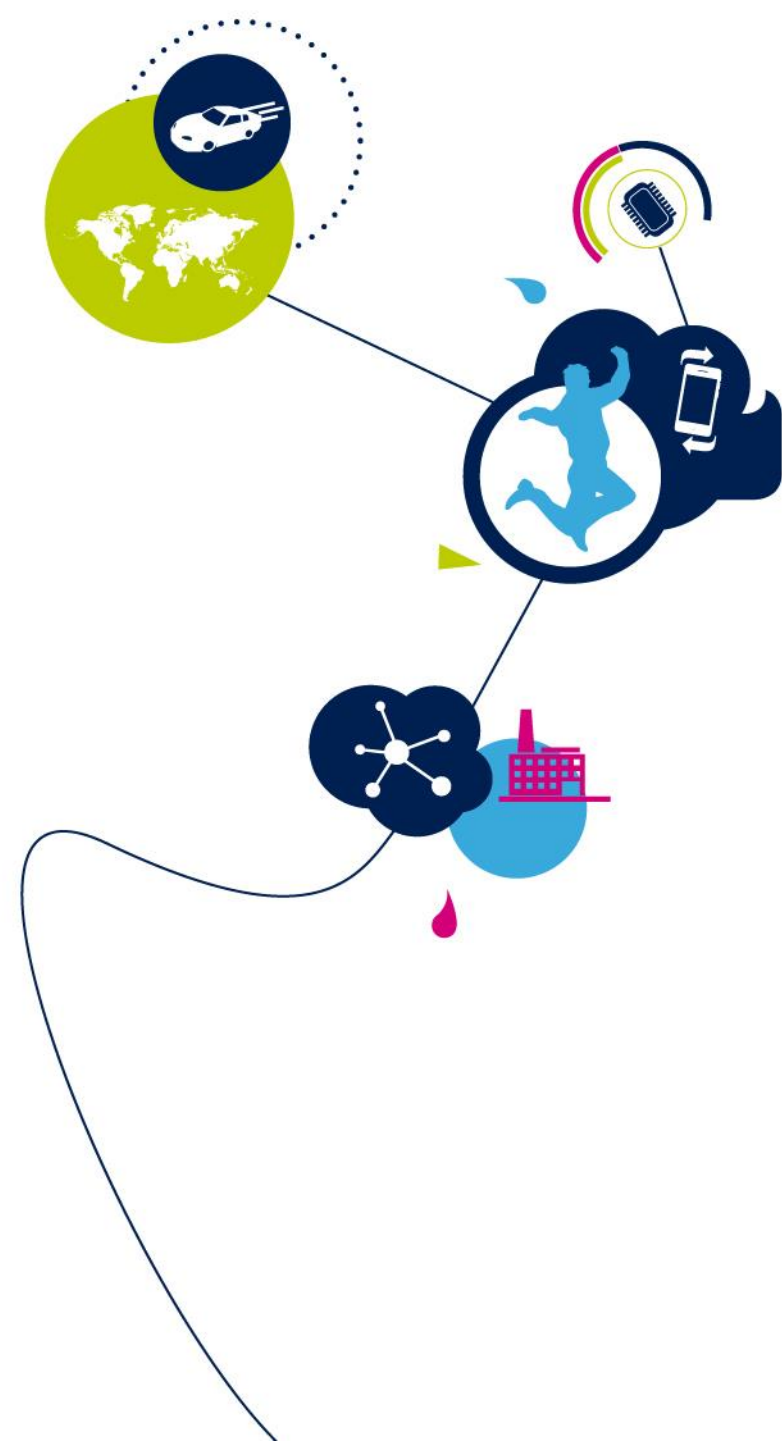


# MC SDK 5.x 介绍

微控制器产品部门

2019年9月



# MC SDK 5.x控制的电机种类

永磁同步电机  
PMSM

直流无刷电机  
BLDC



六步方波控制

FOC矢量控制

MC SDK 5.41

# BLDC/PMSM电机热门应用

- 扫地机器人
- 吸尘器



- 体育器材
- 个人护理



- 风筒
- 水泵
- 抽油烟机
- 电动工具



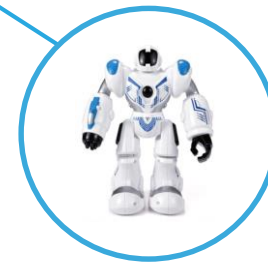
- 空气净化器
- 排气扇
- 落地扇



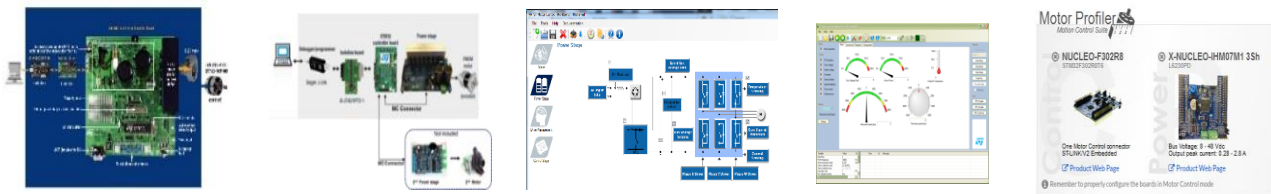
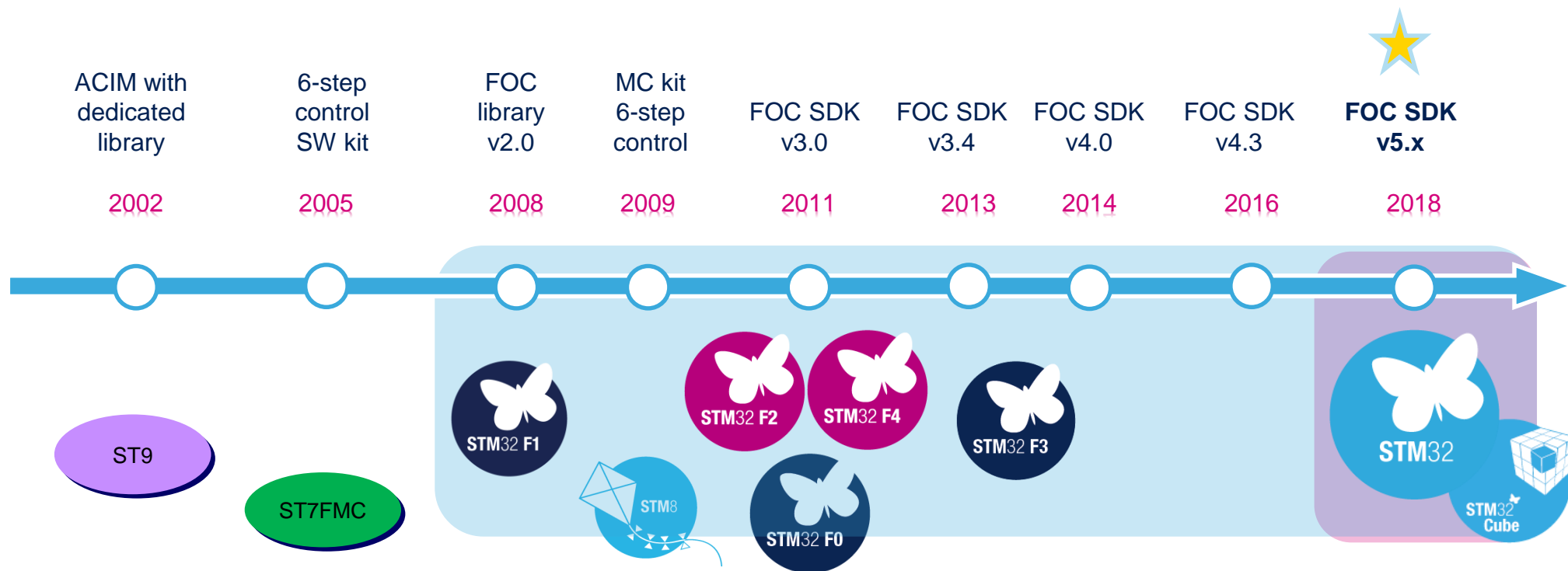
- 冰箱压缩机
- 变频空调
- 变频洗衣机



- 机器人
- 伺服
- 变频器



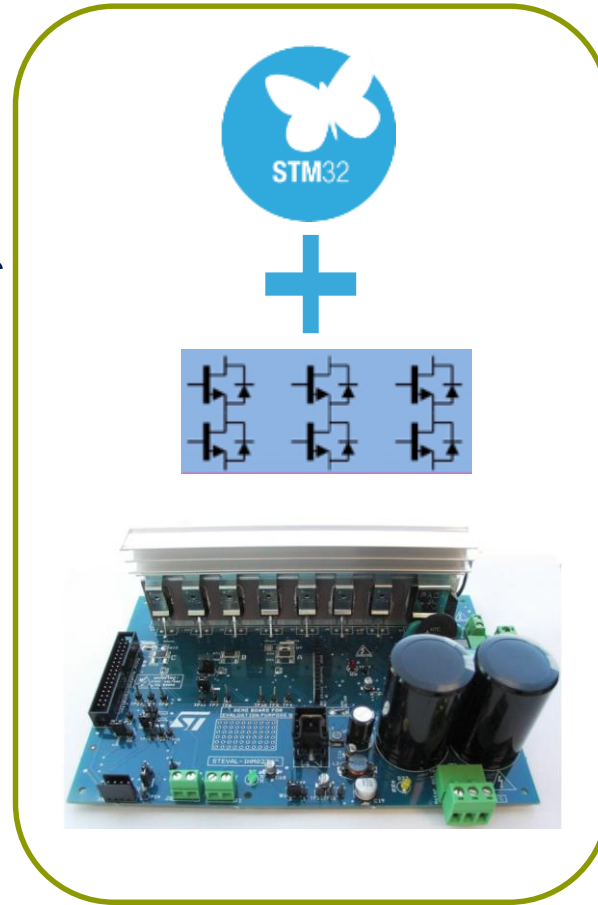
# ST MC SDK 发展路线图



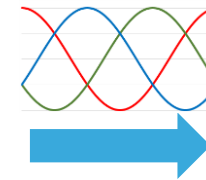


软件层

电机控制程序



硬件层



应用层

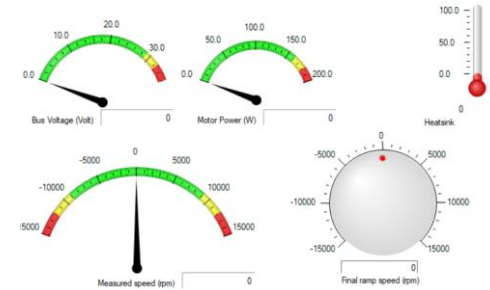
# STM32 MC SDK5.x特点

Motor Profiler  
Motion Control Suite



电机参数自动识别

电机实时交互调试



- STM32F0
- STM32F1
- STM32F3
- STM32F4
- STM32F7
- STM32L4
- STM32G0
- STM32G4

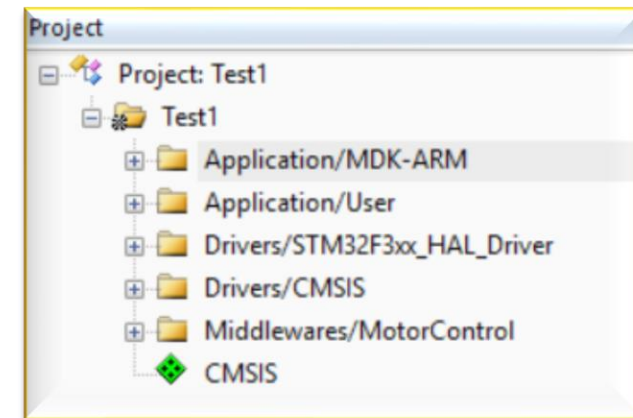
STM32 MC SDK5.4



结合CubeMx  
自动生成电机控制代码

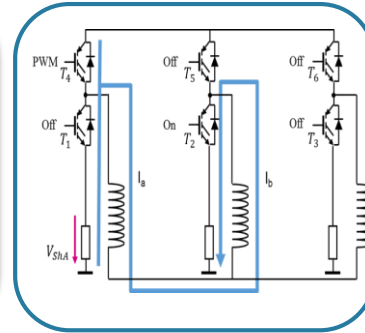
支持有传感/无传感

完整的FOC控制代码



## 电机静止

- Rs 测量
- Ls 测量
- 电流校准运行

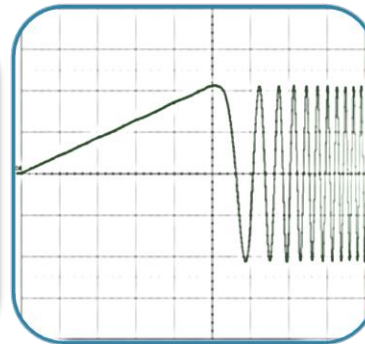


10 sec



## 开环运行

- Ke 参数测试
- 无传感器启动设定
- 切换闭环

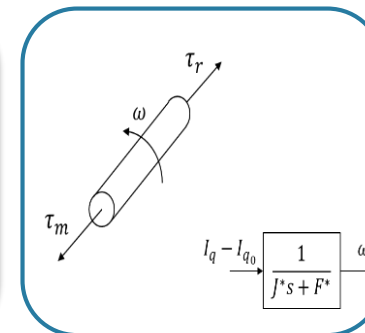


5 sec



## 闭环运行

- 动态系数测试
- 惯性, 阻尼系数测试
- 速度校准



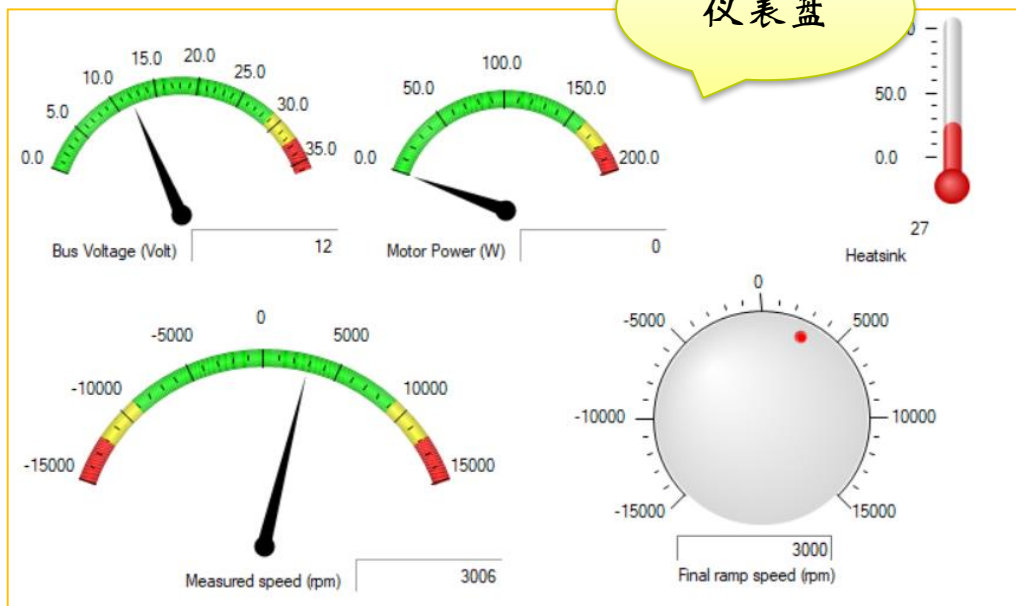
45 sec



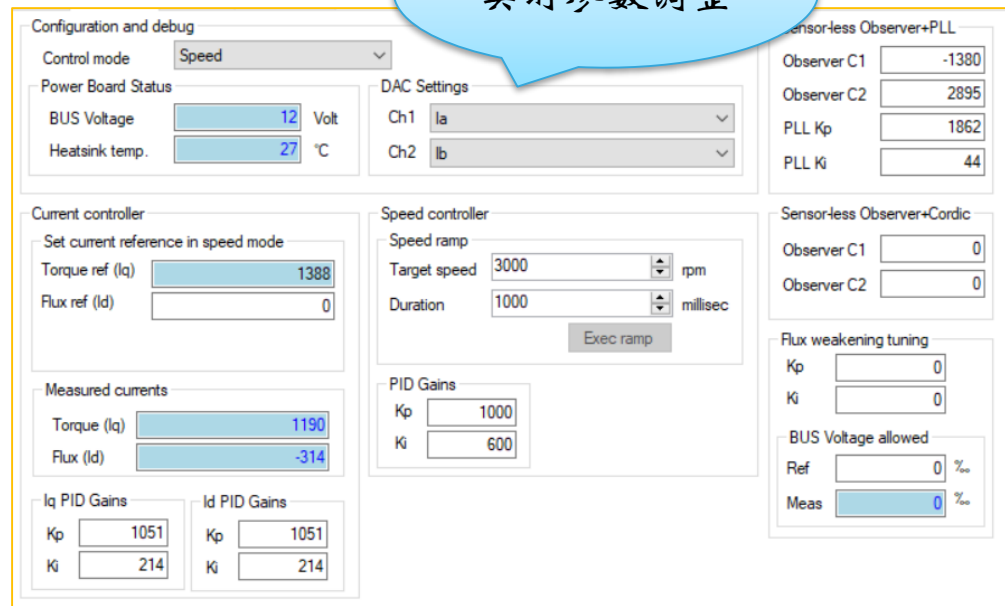
一分钟电机矢量控制运行!

# 电机实时交互调试

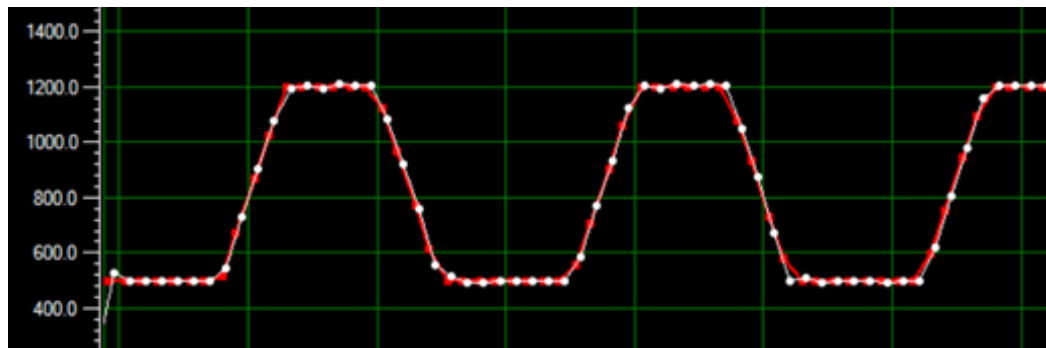
仪表盘



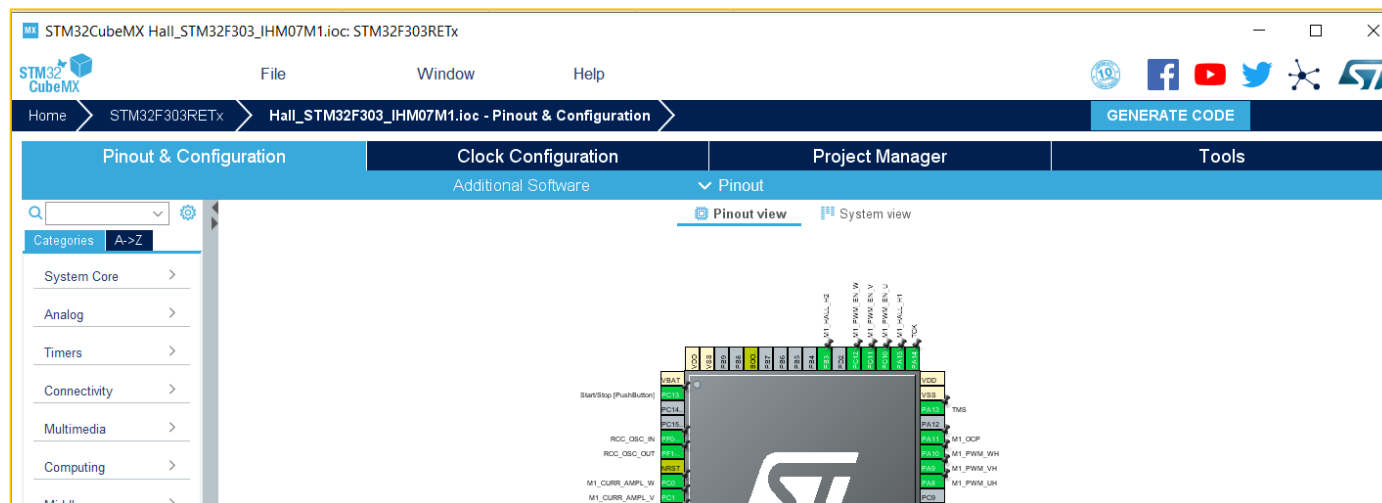
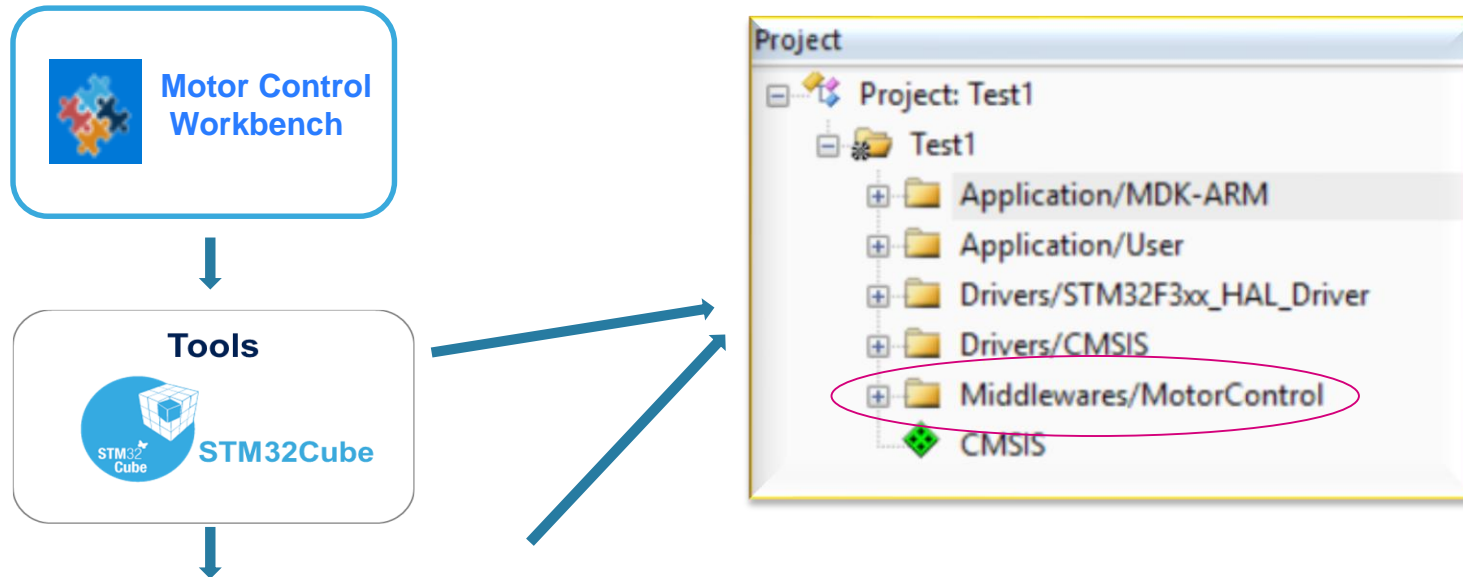
实时参数调整



速度示波器

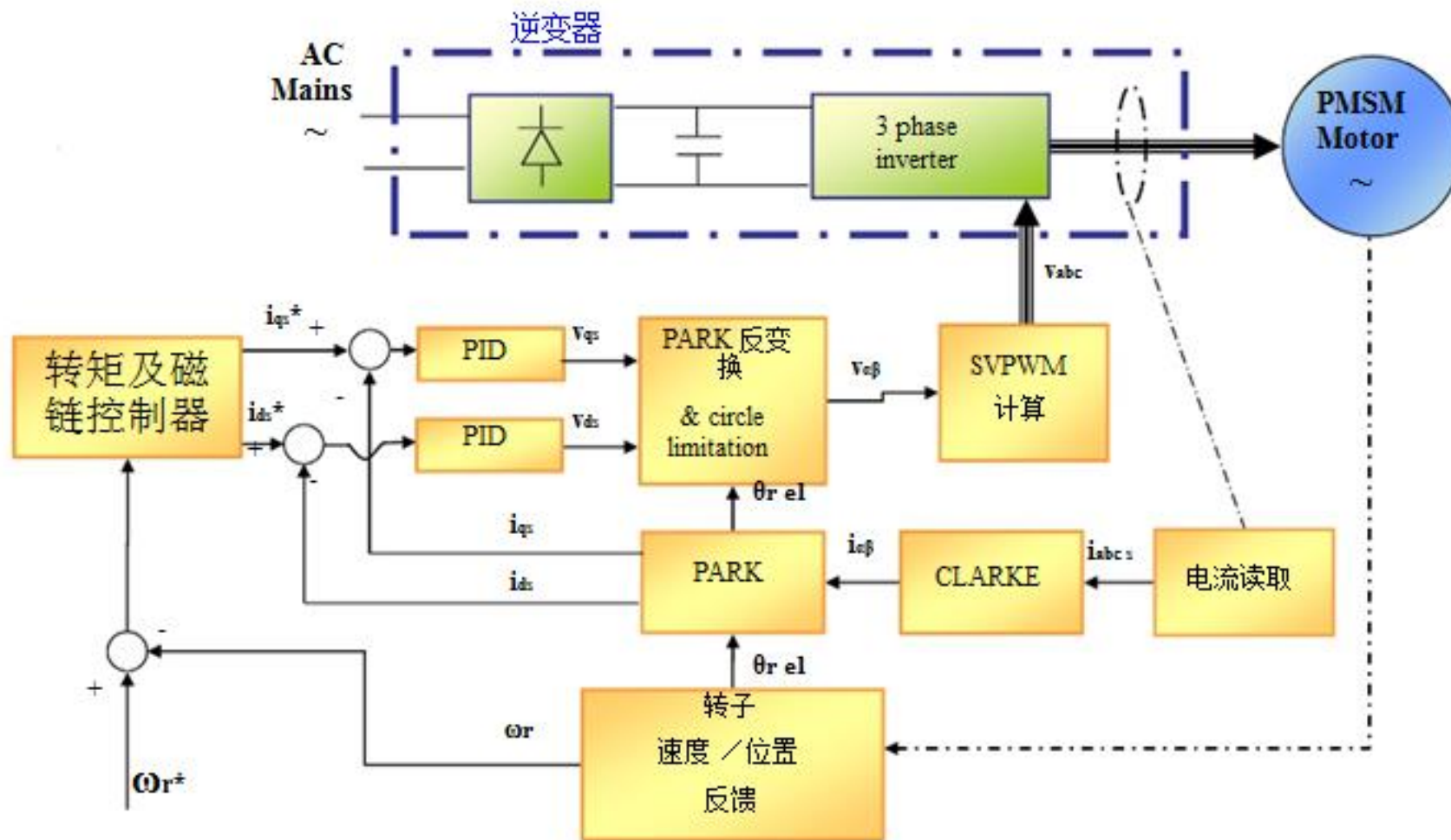


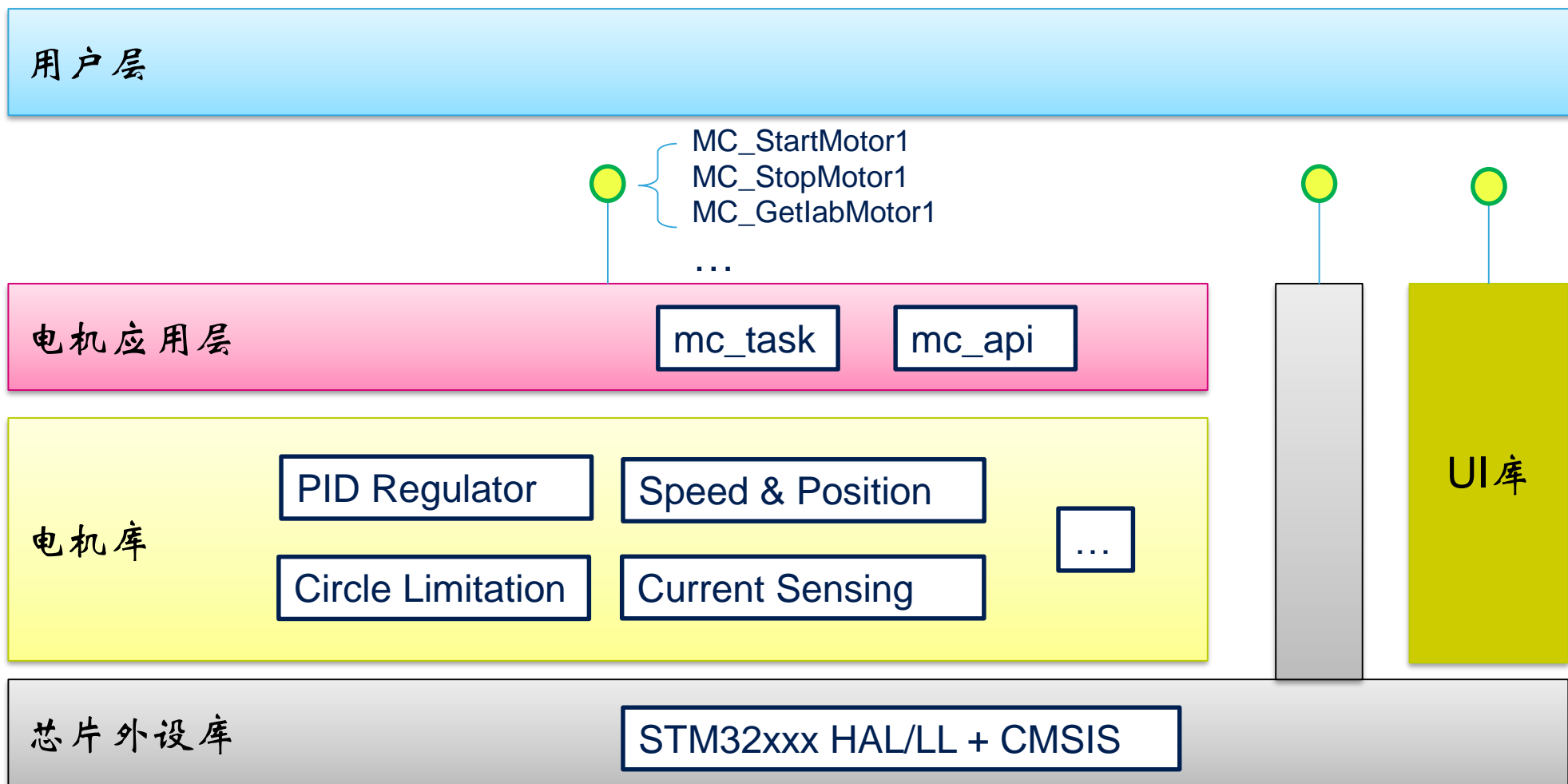
# MC SDK5.x 电机控制代码自动生成



# ST MC SDK 5.x- 性能测试结果

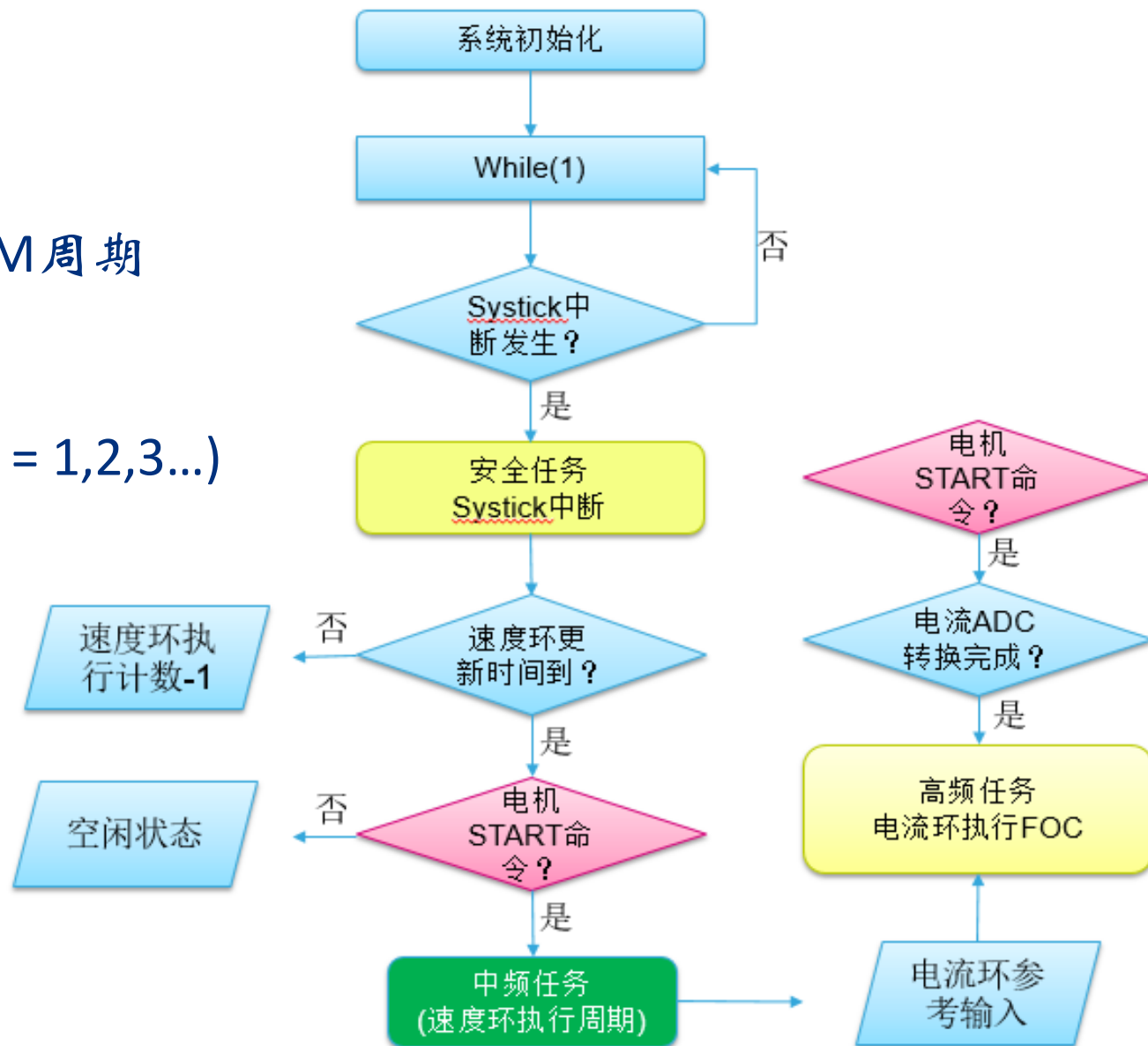
MCSDK5.0							
MCU	Config	Workload (%)	Total Code size (kB)	ro data (B)	RAM (B)	FOC Lib (kB)	HAL (kB)
F072RB	1 Shunt	44.3	18.8	609	3126	12.8	5.1
F072RB	3 Shunt	39.4	19.5	653	2910	12.9	4.5
F303RE	1 Shunt	20.4	22.3	4427	2940	14.4	7.8
F303RE	3 Shunt	18.1	23.6	4179	2884	16.1	7.5
F446RE	1 Shunt	10.2	19.7	625	3122	14.4	5.3
F446RE	3 Shunt	8.2	17.8	603	2840	13.1	4.7
F303VE	DUAL/3S	38.2	20.8	4449	4724	13.1	7.7
F415ZG	DUAL/3S	18.3	19.3	761	4484	14.7	4.6





# MC SDK电机库结构

- 高频环路 (FOC) :PWM周期
- 安全环路: 500us
- 中频环路:  $500\mu s * n$  ( $n = 1, 2, 3 \dots$ )



# ST MC SDK5.x 固件功能

**速度采集**

- 霍尔传感器
- 正交增量编码器
- 无速度/位置传感器

**电流采集**

- 1 shunt
- 3 shunt
- ICS

**直流母线电压检测**

**功率器件温度检测**

**传感器**

**SVPWM**

**执行器**

**FOC**

**MTPA**

**指令轨迹生成**

**弱磁控制**

**电流前馈控制**

**电机参数测量**

**核心功能**

**Speed PI**

**Iq PI**

**Id PI**

**PI 调节器**

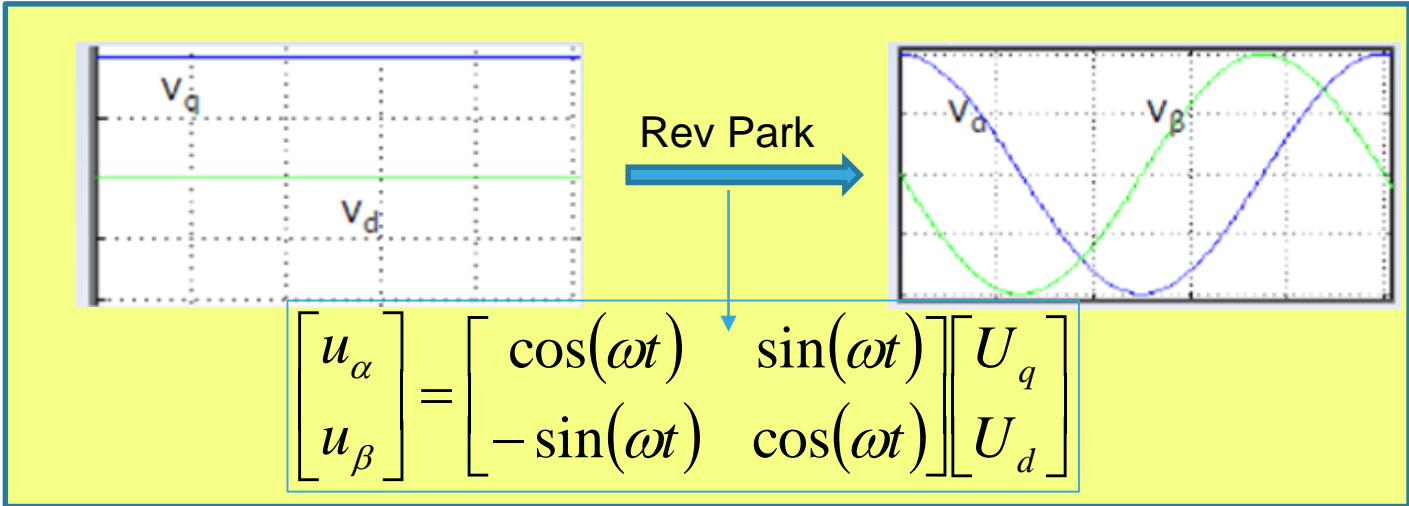
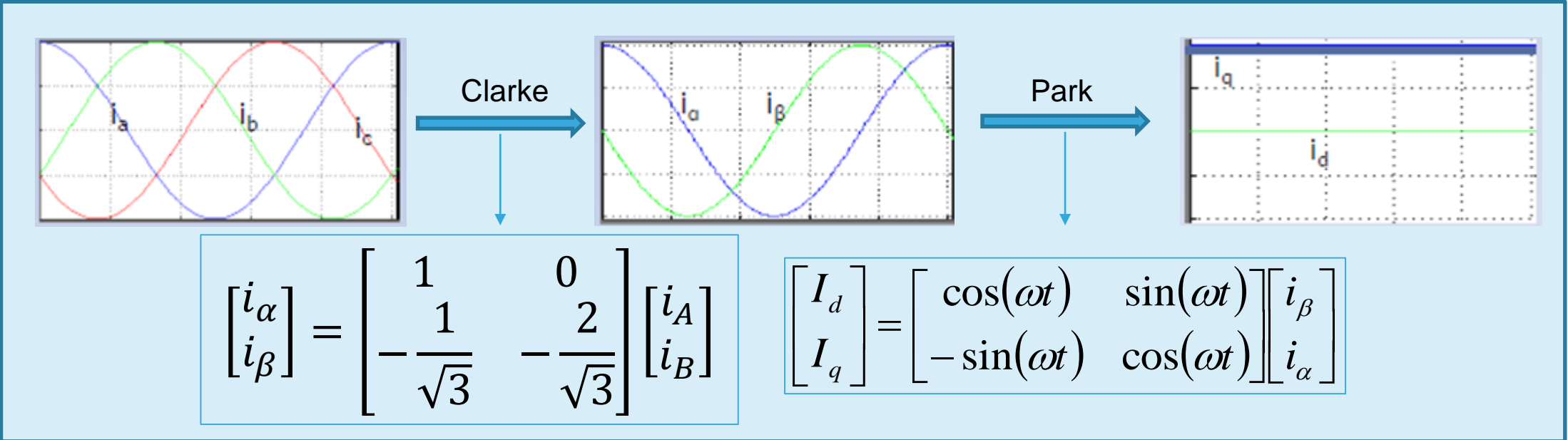
**STM32F0**

**STM32F3**

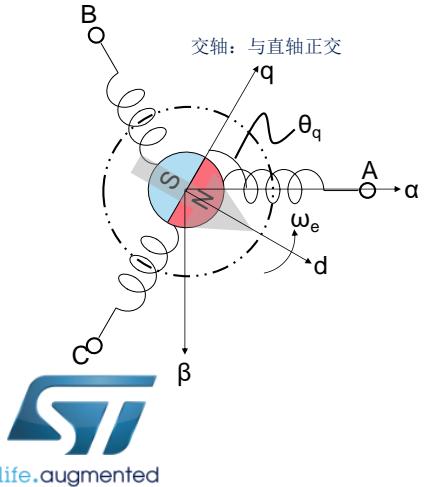
...

**MCU**

# 矢量变换公式以及示意图



$$p_3 = \frac{3}{2} p_2$$



# ST MC SDK5.x实现矢量变换的固件源程序

➤ 具体文件夹如下：

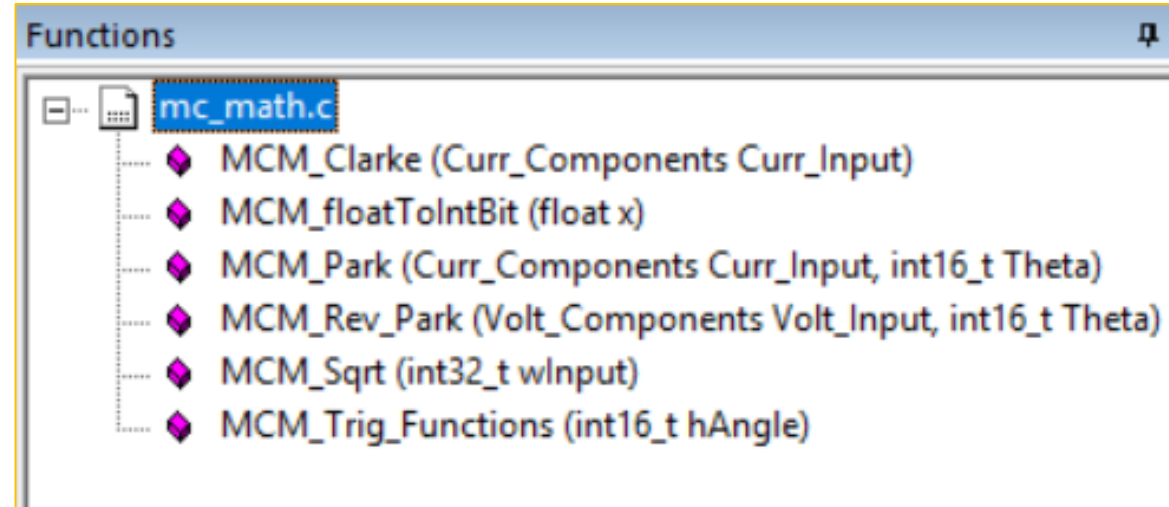
- ✓ xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\Any\Src

➤ 文件名如下：

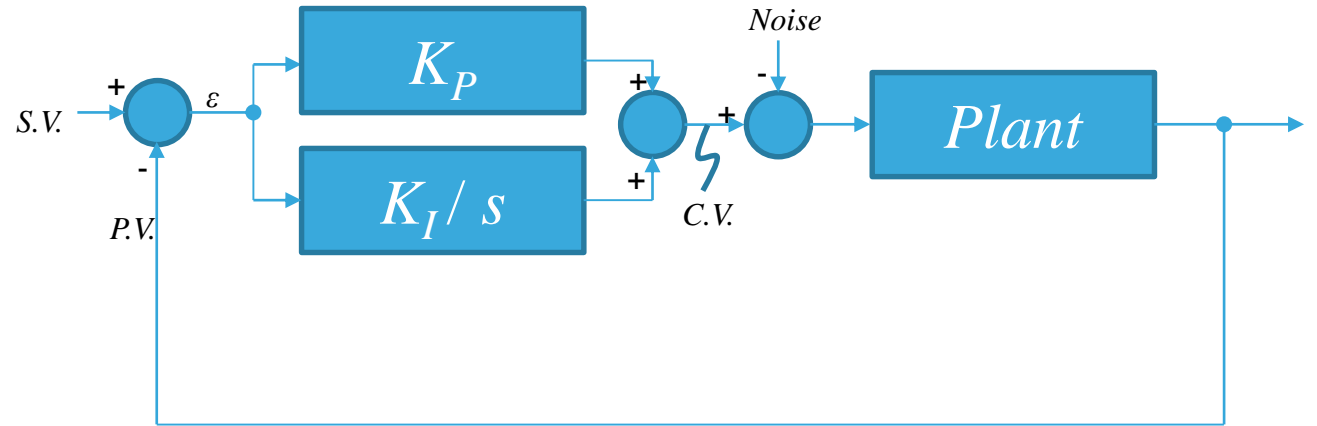
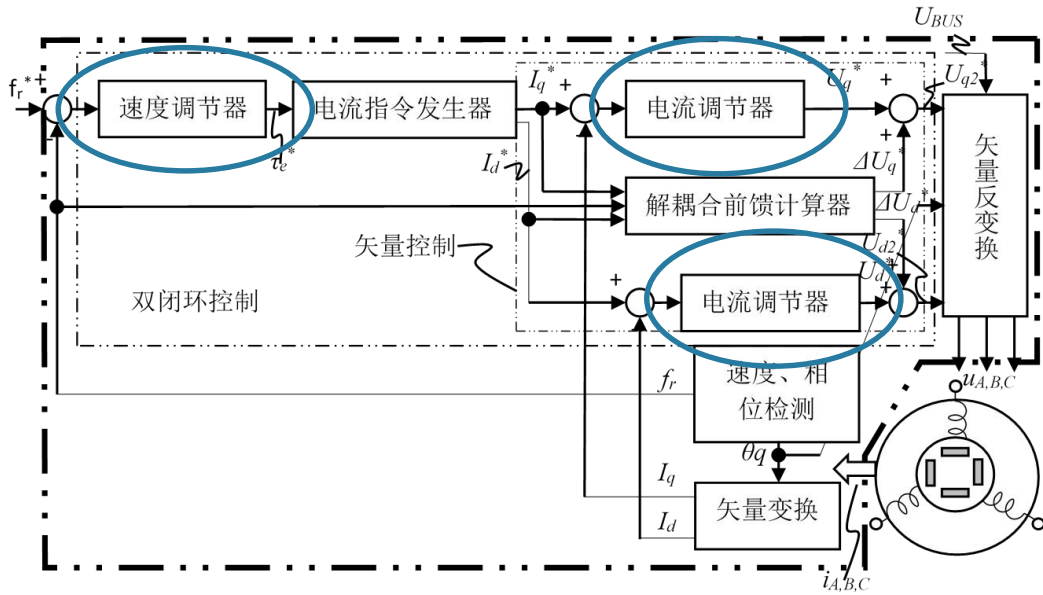
- ✓ mc\_math.c

➤ 函数名：

- ✓ MCM\_Clarke
- ✓ MCM\_Park
- ✓ MCM\_Rev\_Park



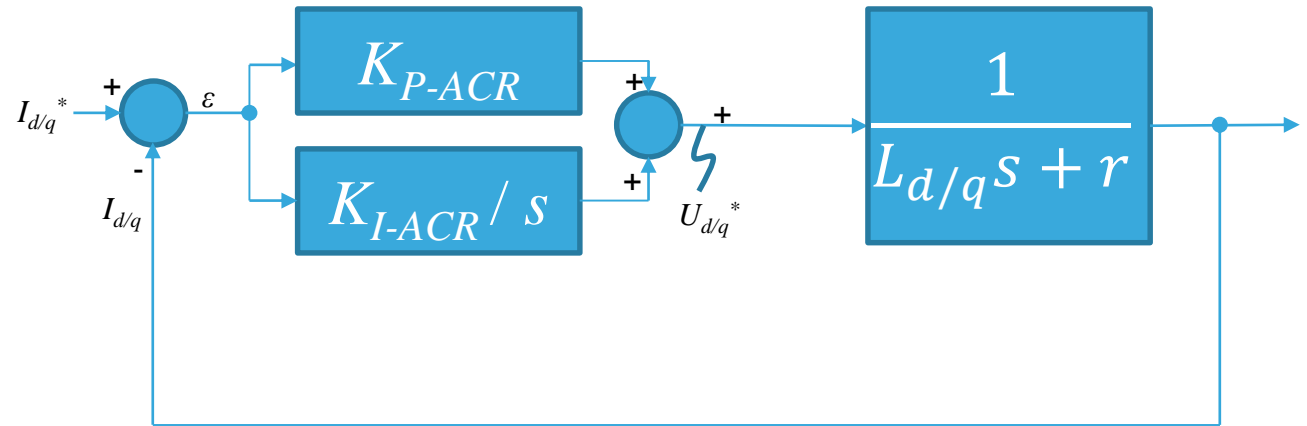
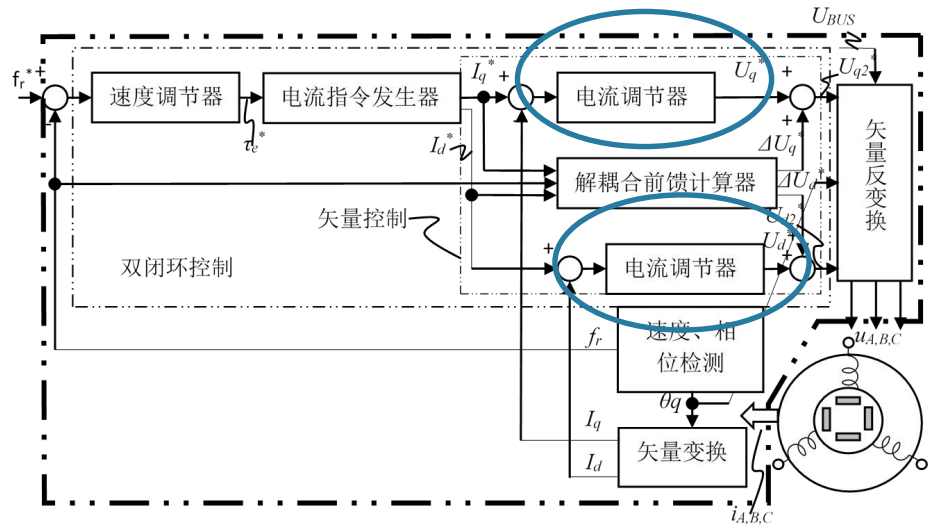
# 矢量控制 —— 控制器



控制器	设定值 S.V.	实际值 P.V.	控制值 C.V.	对象传递函数
速度控制器	$f_{r-ref}(f_r^*)$	$f_{r-fb}(f_r)$	$\tau_{ref}(\tau^*)$ or $I_{q-ref}(I_q^*)$	$\frac{1}{Js + F}$
电流控制器 (忽略 dq 之间的耦合)	$I_{d/q-ref}(I_{d/q}^*)$	$I_{d/q-fb}(I_{d/q})$	$U_{d/q-ref}(U_{d/q}^*)$	$\frac{1}{L_{d/q}s + r}$

# 电流调节器

## --拉普拉斯域设定



$$K_{P-ACR-d} = L_d \omega_{B-ACR}$$

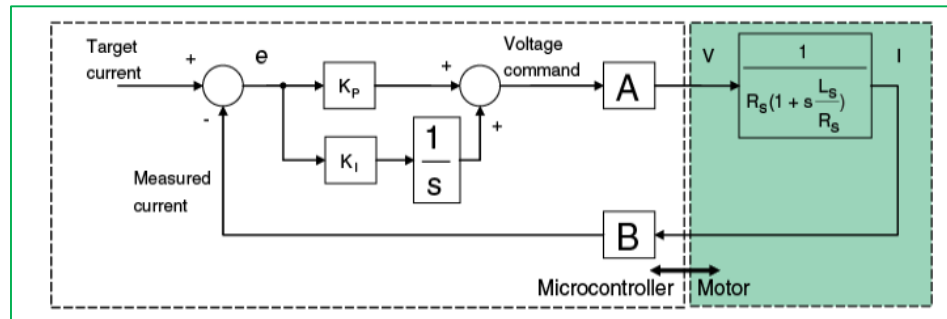
$$K_{P-ACR-q} = L_q \omega_{B-ACR}$$

$$K_{I-ACR} = r \omega_{B-ACR}$$

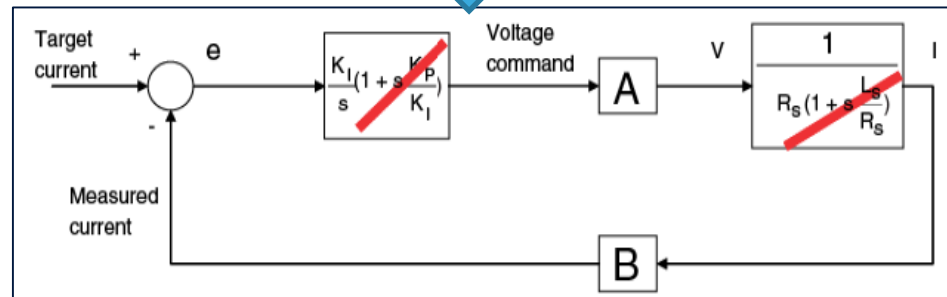
□ 电流调节器的开环增益为

$$G_{ACR-Openloop}(s) = \left( K_{P-ACR-d/q} + \frac{K_{I-ACR}}{s} \right) \frac{1}{L_{d/q}s + r} = \frac{\omega_{B-ACR}}{s}$$

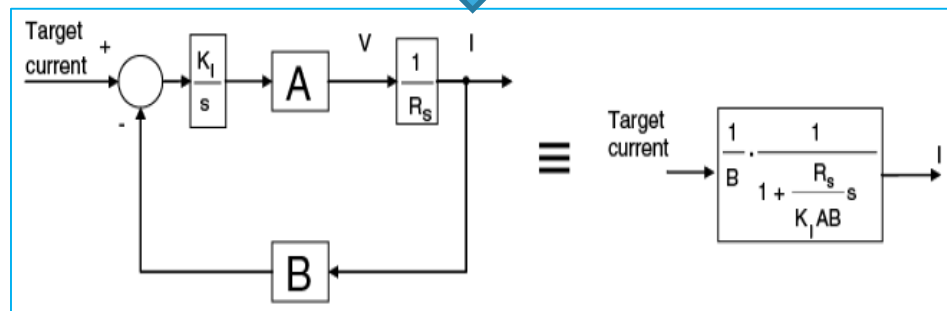
## --ST MC SDK5.x WB 设定



令  $K_P/K_I = L_S/R_S$



变为一阶系统



$$K_P = L_S \frac{\omega_C}{AB}$$

$$K_I = \frac{R_S \cdot \omega_C \cdot T}{AB}$$

$$AB = \frac{V_{BusDC} \cdot R_{shunt} \cdot A_{op}}{3.3}$$

$$A = \frac{V_{BusDC}}{2^{16}}$$

$$B = \frac{R_{shunt} A_{op} 2^{16}}{3.3}$$

## 拉普拉斯域PI系数设定方法

$$K_{P-ASR} = J\omega_{B-ASR}$$

$$K_{I-ASR} = F\omega_{B-ASR}$$

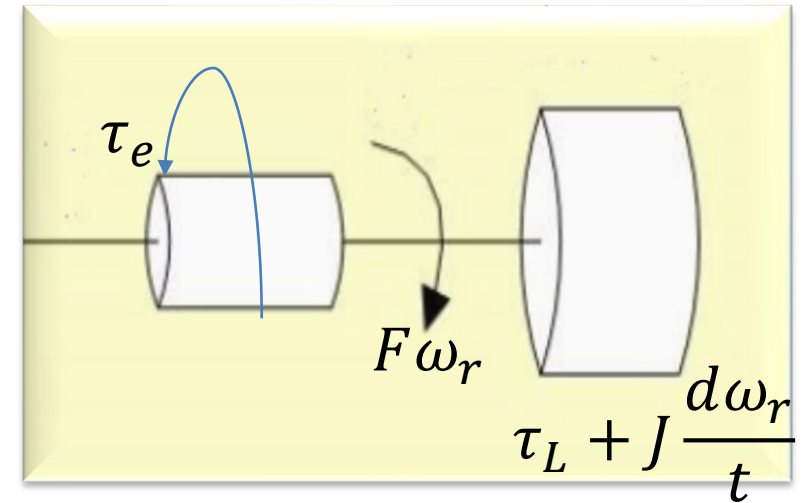
$J$ : 转动惯量 [kg/m<sup>2</sup>];

$F$ : 阻力系数 [Nm/[rad/s]];

$\omega_{B-ASR}$ : 速度调节器通带宽度 [rad/s];

$K_{P-ASR}$ : 速度调节器比例系数;

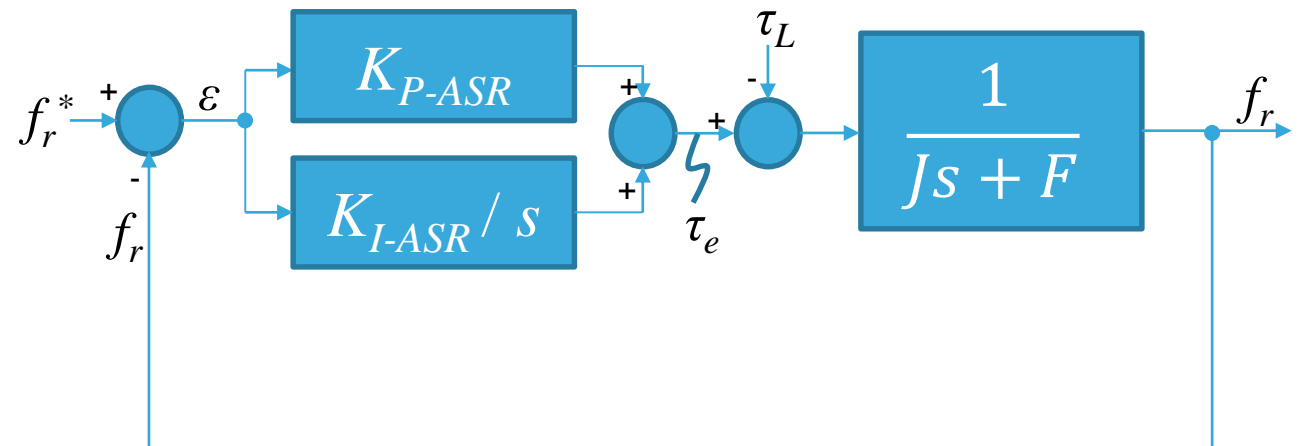
$K_{I-ASR}$ : 速度调节器积分系数。



## WB 中PI系数的设定

$$K_{P-ASR-WB} = \frac{J_{WB}}{1000000} \frac{k(2\pi f_{B-ASR})p}{10}$$

$$K_{I-ASR-WB} = \frac{K_{P-ASR-WB} T_{s-ASR-WB}}{\tau_{mech} \cdot 1000}$$



$$k_{\tau} = 1.5 \sqrt{\frac{2}{3} \frac{60}{1000 \cdot 2\pi}}, k = \frac{65536 \cdot R_{shunt} \cdot G_{OPAMP}}{3.3k_{\tau}}, \tau_{mech} = \frac{J_{WB}}{F_{WB}}, f_{B-ASR} = \frac{0.5}{\tau_{mech}} \times 30$$

# PID在ST MC SDK5.x 固件中的实现

21

## ➤ 具体文件夹如下:

- ✓ xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\Any\Src

## ➤ 文件名如下:

- ✓ pid\_regulator.c

## ➤ 函数名:

- ✓ PI\_Controller(库中只使用了PI)

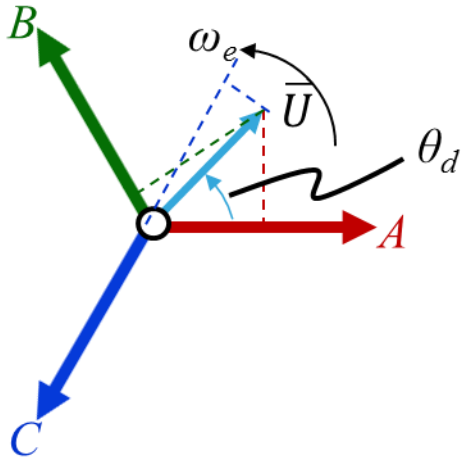
```
pid_regulator.c
328 #endif
329 /**
330  * @brief This function compute the output of a PI regulator sum of its
331  *        proportional and integral terms
332  * @param pHandle: handler of the current instance of the PID component
333  * @param wProcessVarError: current process variable error, intended as the reference
334  *        value minus the present process variable value
335  * @retval computed PI output
336  */
337 int16_t PI_Controller( PID_Handle_t * pHandle, int32_t wProcessVarError )
338 {
339     int32_t wProportional_Term, wIntegral_Term, wOutput_32, wIntegral_sum_temp;
340     int32_t wDischarge = 0;
341     int16_t hUpperOutputLimit = pHandle->hUpperOutputLimit;
342     int16_t hLowerOutputLimit = pHandle->hLowerOutputLimit;
343
344     /* Proportional term computation*/
345     wProportional_Term = pHandle->hKpGain * wProcessVarError;
346
347     /* Integral term computation */
348     if ( pHandle->hKiGain == 0 )
349     {
350         pHandle->wIntegralTerm = 0;
351     }
352     else
353     {
```

# PID参数在ST MC SDK5.x固件中的位置

- 具体文件夹如下：
  - ✓ xxx\Inc
- 文件名如下：
  - ✓ drive\_parameters.h
- 参数见右图所示

```
drive_parameters.h
217
218 /* Gains values for torque and flux control loops */
219 #define PID_TORQUE_KP_DEFAULT      1051
220 #define PID_TORQUE_KI_DEFAULT      214
221 #define PID_TORQUE_KD_DEFAULT      100
222 #define PID_FLUX_KP_DEFAULT        1051
223 #define PID_FLUX_KI_DEFAULT        214
224 #define PID_FLUX_KD_DEFAULT        100
225                                     电流环PID参数
226 /* Torque/Flux control loop gains dividers*/
227 #define TF_KP_DIV                   16384
228 #define TF_KI_DIV                   16384
229 #define TF_KD_DIV                   8192
230 #define TFDIFFERENTIAL_TERM_ENABLING DISABLE
231
232 /* Speed control loop */
233 #define SPEED_LOOP_FREQUENCY_HZ     1000 /*!<Execution
234
235 #define PID_SPEED_KP_DEFAULT         1000
236 #define PID_SPEED_KI_DEFAULT         600
237 #define PID_SPEED_KD_DEFAULT         0
238 /* Speed PID parameter dividers */ 速度环PID参数
239 #define SP_KP_DIV                    16
240 #define SP_KI_DIV                    256
241 #define SP_KD_DIV                    16
242 /* USER CODE BEGIN PID_SPEED_INTEGRAL_INIT_DIV */
243 #define PID_SPEED_INTEGRAL_INIT_DIV 1 /* */
244 /* USER CODE END PID_SPEED_INTEGRAL_INIT_DIV */
```

# 空间电压矢量PWM(SVPWM)(1/3)

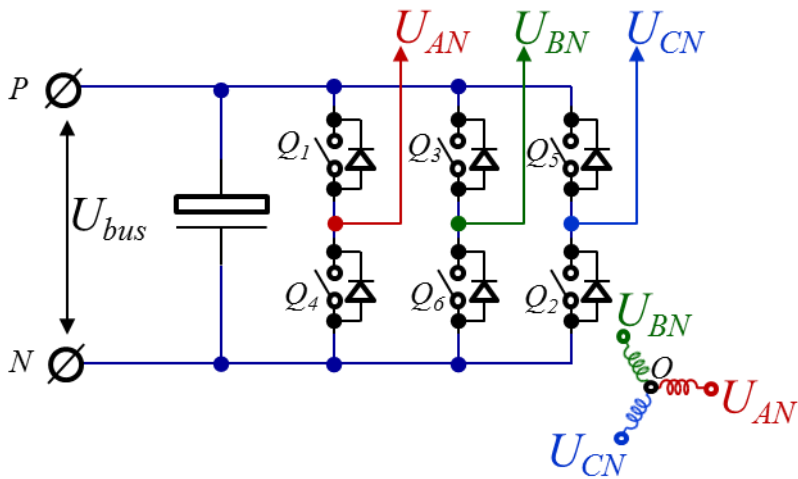


$$U_{AO} = U_m \cos(\theta_d) = U_m \cos(\omega_e t)$$

$$U_{BO} = U_m \cos(\theta_d - 120^\circ) = U_m \cos(\omega_e t - 120^\circ)$$

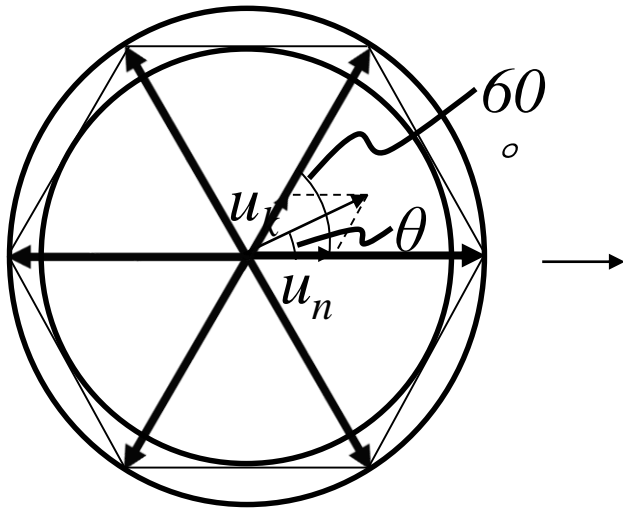
$$U_{CO} = U_m \cos(\theta_d + 120^\circ) = U_m \cos(\omega_e t + 120^\circ)$$

$$|\vec{U}| = U_m$$

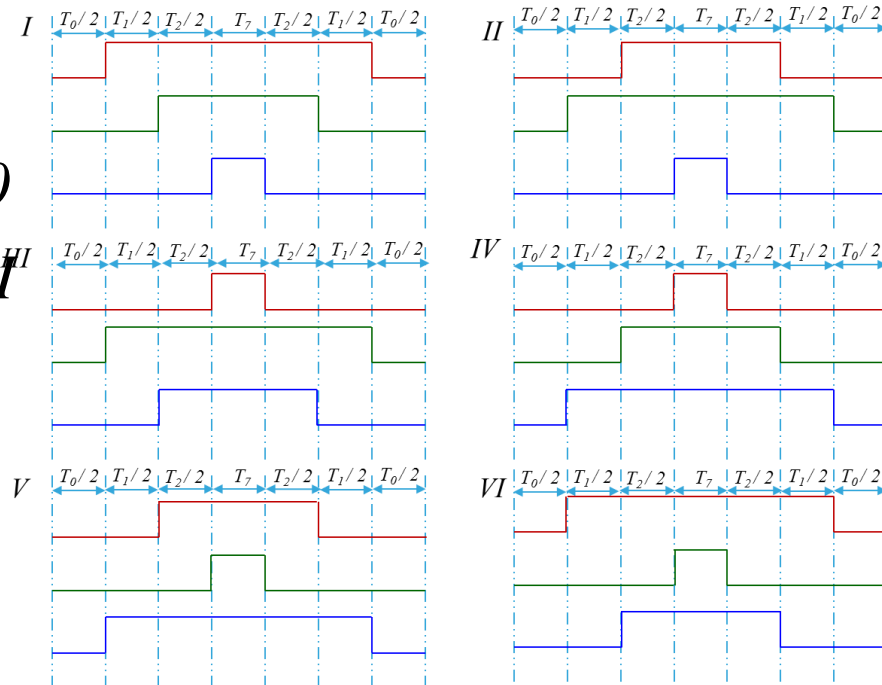
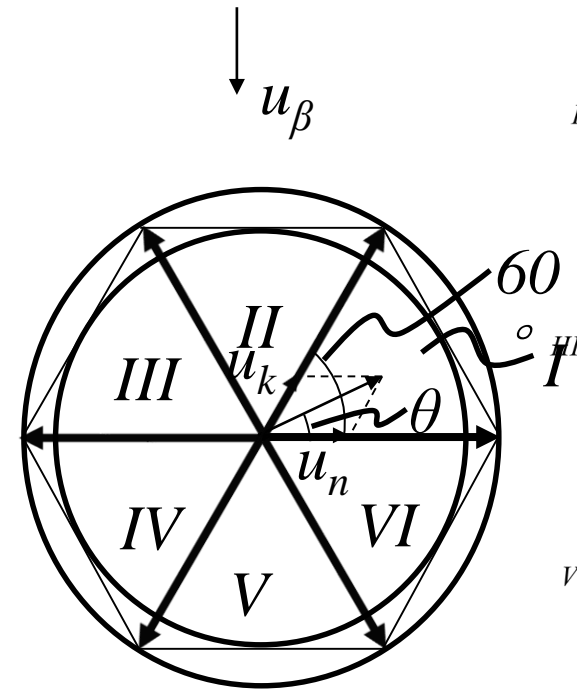


vector	$Q_1(Q_4)$	$Q_3(Q_6)$	$Q_5(Q_2)$	$U_{AN} [u]$	$U_{BN}[u]$	$U_{CN}[u]$	$U_{AO} [u]$	$U_{BO}[u]$	$U_{CO}[u]$
$\vec{U}_0$	OFF(ON)	OFF(ON)	OFF(ON)	0	0	0	0	0	0
$\vec{U}_1$	ON(OFF)	OFF(ON)	OFF(ON)	$U_{bus}$	0	0	$2U_{bus}/3$	$-U_{bus}/3$	$-U_{bus}/3$
$\vec{U}_2$	ON(OFF)	ON(OFF)	OFF(ON)	$U_{bus}$	$U_{bus}$	0	$U_{bus}/3$	$U_{bus}/3$	$-2U_{bus}/3$
$\vec{U}_3$	OFF(ON)	ON(OFF)	OFF(ON)	0	$U_{bus}$	0	$-U_{bus}/3$	$2U_{bus}/3$	$-U_{bus}/3$
$\vec{U}_4$	OFF(ON)	ON(OFF)	ON(OFF)	0	$U_{bus}$	$U_{bus}$	$-2U_{bus}/3$	$U_{bus}/3$	$U_{bus}/3$
$\vec{U}_5$	OFF(ON)	OFF(ON)	ON(OFF)	0	0	$U_{bus}$	$-U_{bus}/3$	$-U_{bus}/3$	$2U_{bus}/3$
$\vec{U}_6$	ON(OFF)	OFF(ON)	ON(OFF)	$U_{bus}$	0	$U_{bus}$	$U_{bus}/3$	$-2U_{bus}/3$	$-U_{bus}/3$
$\vec{U}_7$	ON(OFF)	ON(OFF)	ON(OFF)	$U_{bus}$	$U_{bus}$	$U_{bus}$	0	0	0

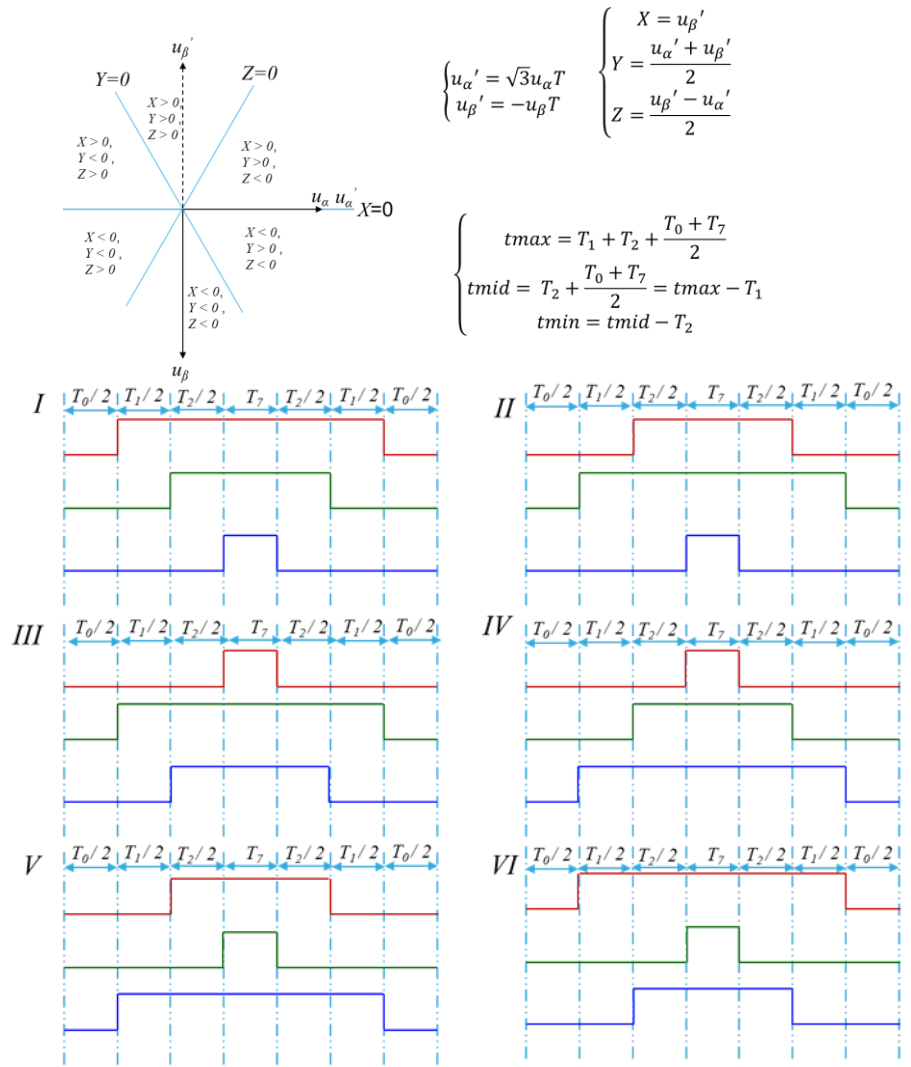
# 空间电压矢量PWM(SVPWM)(2/3)



Sector	I	II	III	IV	V	VI
$n$	1	3	3	5	5	1
$k$	2	2	4	4	6	6



# 空间电压矢量PWM(SVPWM)(3/3)



sector	T1	T2	T0+T7
I	-Z	X	T + Z - X
II	Z	Y	T - Y - Z
III	X	-Y	T - X + Y
IV	-X	Z	T + X - Z
V	-Y	-Z	T + Y + Z
VI	Y	-X	T + X - Y

sector	tA	tB	tC
I	T/2 + (X - Z)/2	tA + Z	tB - X
II	T/2 + (Y - Z)/2	tA + Z	tA - Y
III	T/2 + (Y - X)/2	tC + X	tA - Y
IV	T/2 + (X - Z)/2	tA + Z	tB - X
V	T/2 + (Y - Z)/2	tA + Z	tA - Y
VI	T/2 + (Y - X)/2	tC + X	tA - Y

# SVPWM在ST MC SDK5.x固件中的实现

## ➤ 具体文件夹如下:

- ✓ xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\Any\Src

## ➤ 文件名如下:

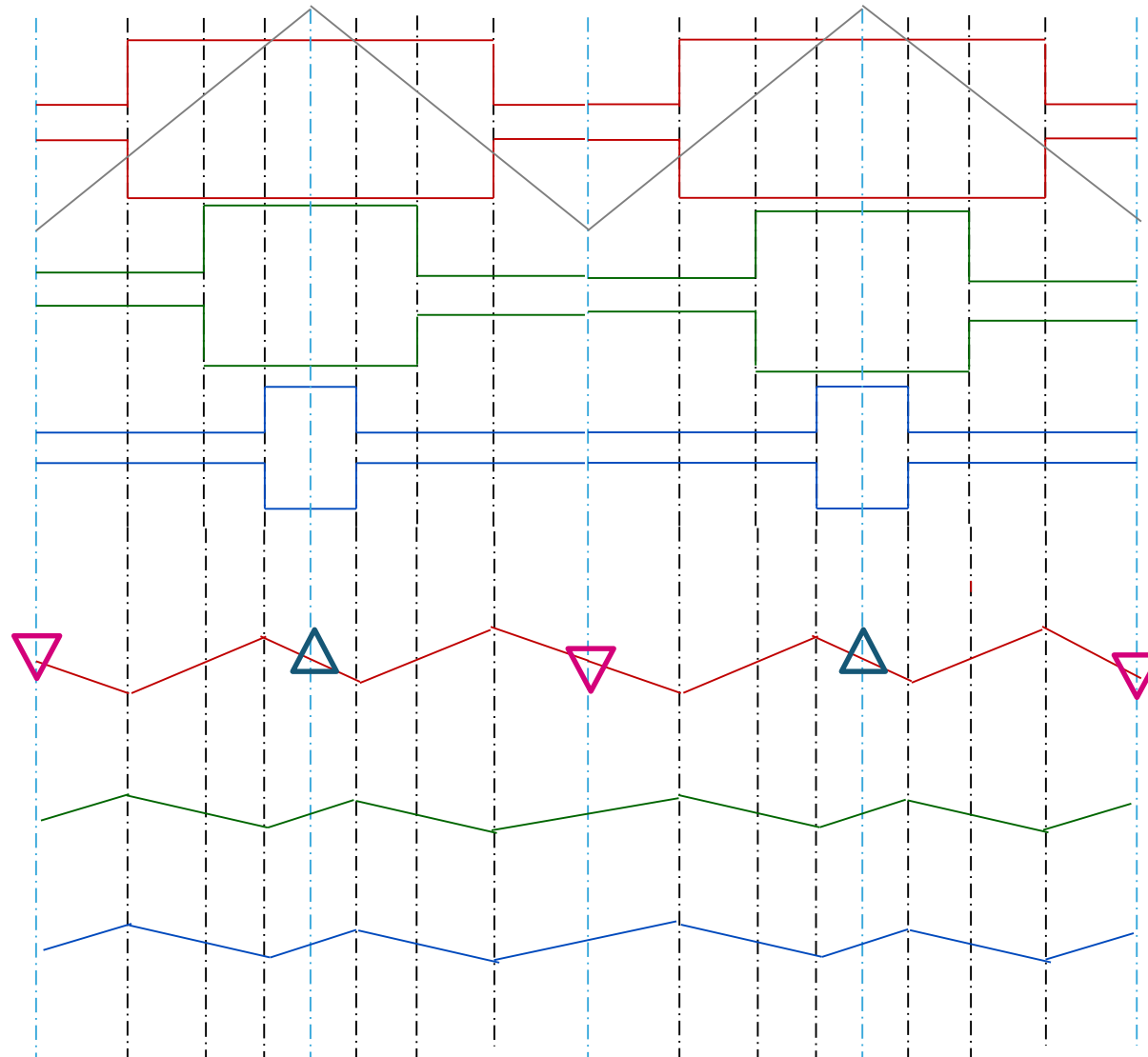
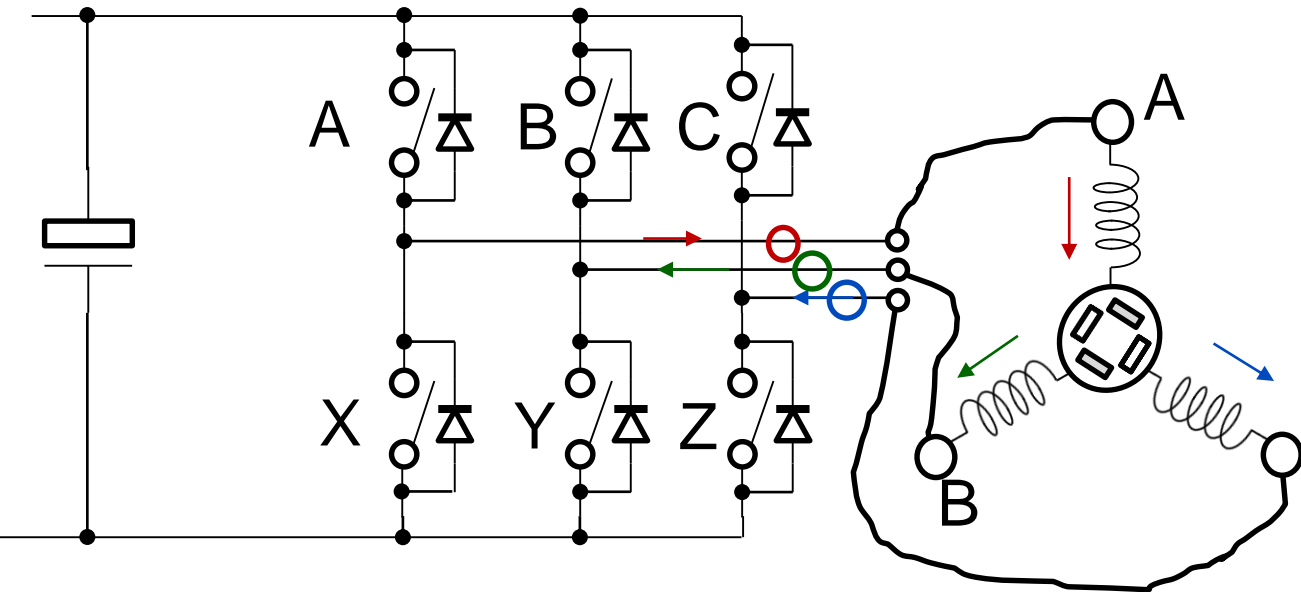
- ✓ pwm\_curr\_fdbk.c

## ➤ 函数名:

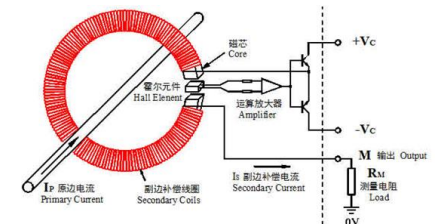
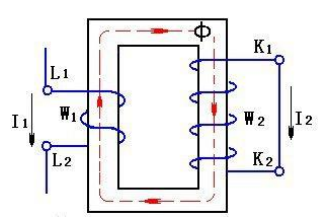
- ✓ PWMC\_SetPhaseVoltage

```
pwm_curr_fdbk.c
149 *
150 * @retval Returns #MC_NO_ERROR if no error occurred or #MC_FOC_DURATION if the duty cycles were
151 *         set too late for being taken into account in the next PWM cycle.
152 */
153 uint16_t PWMC_SetPhaseVoltage( PWMC_Handle_t * pHandle, Volt_Components Valfa_beta )
154 {
155     int32_t wX, wY, wZ, wUAlpha, wUBeta, wTimePhA, wTimePhB, wTimePhC;
156     PWMC_SetSampPointSectX_Cb_t pSetADCSamplingPoint;
157
158     wUAlpha = Valfa_beta.qV_Component1 * ( int32_t )pHandle->hT_Sqrt3;
159     wUBeta = -( Valfa_beta.qV_Component2 * ( int32_t )( pHandle->hPWMperiod ) ) * 2;
160
161     wX = wUBeta;
162     wY = ( wUBeta + wUAlpha ) / 2;
163     wZ = ( wUBeta - wUAlpha ) / 2;
164
165     /* Sector calculation from wX, wY, wZ */
166     if ( wY < 0 )
167     {
168         if ( wZ < 0 )
169         {
170             pHandle->hSector = SECTOR_5;
171             wTimePhA = ( int32_t )( pHandle->hPWMperiod ) / 4 + ( ( wY - wZ ) / ( int32_t )262144 );
172             wTimePhB = wTimePhA + wZ / 131072;
173             wTimePhC = wTimePhA - wY / 131072;
174             pSetADCSamplingPoint = pHandle->pFctSetADCSampPointSect5;
175         }
176     }
}
```

# 电流采样 — ICS



	ACCT	DCCT
频率范围	>0 Hz ~ tens kHz	DC ~ 100kHz
退磁	需要	不需要
成本	低	高



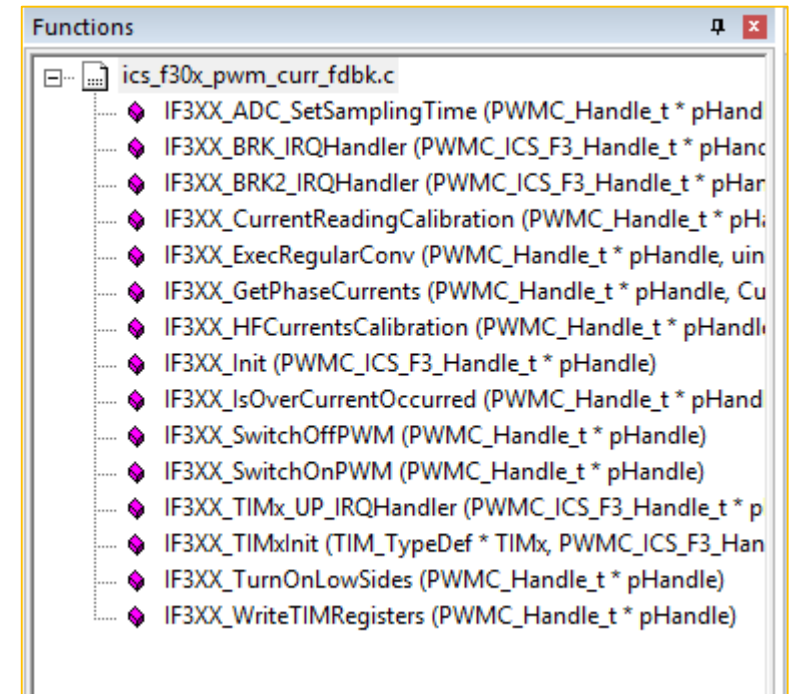
# ST MC SDK5.x ICS 采样固件

## ➤ 针对STM32系列芯片都有各自的文件：

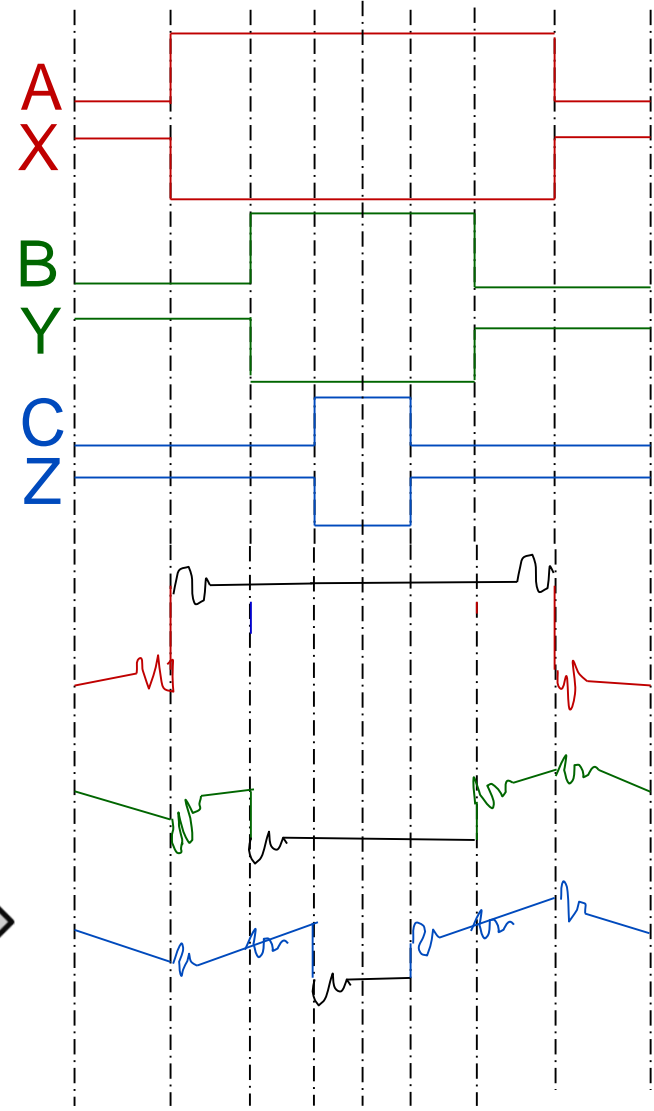
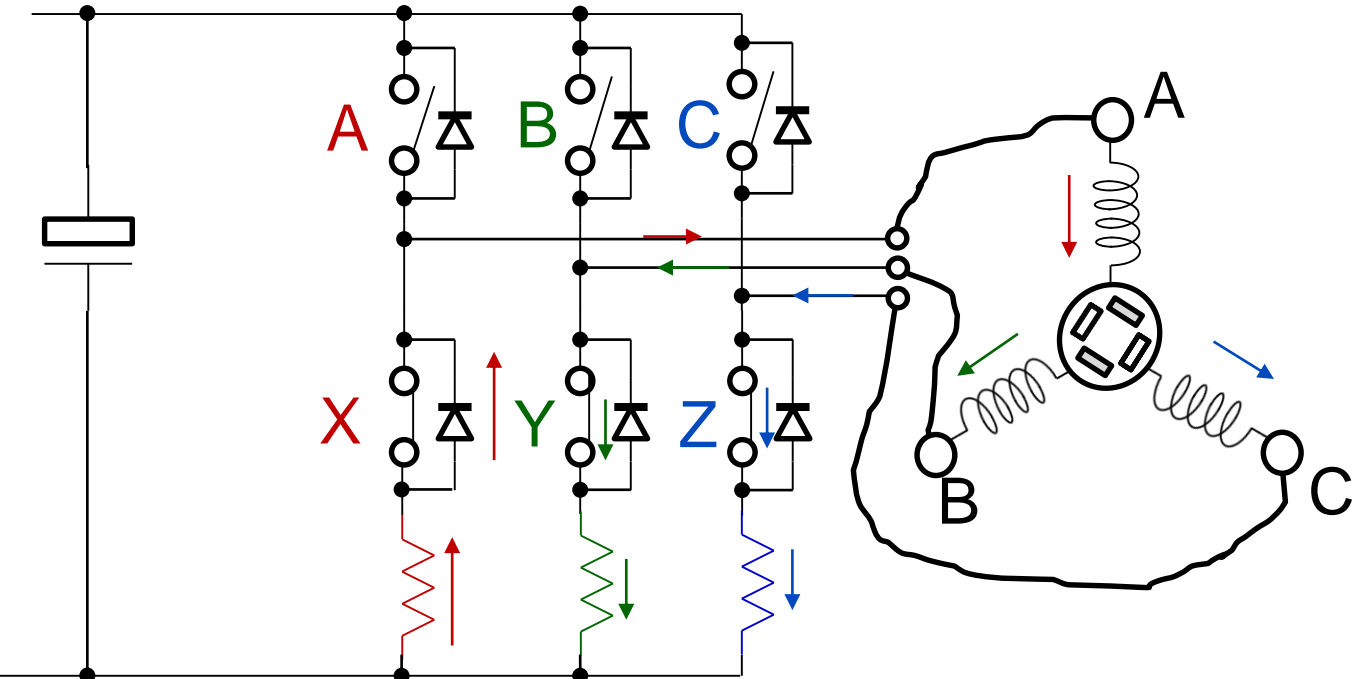
- ✓ xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\F1xx\Src
- ✓ xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\F3xx\Src
- ✓ xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\F4xx\Src
- ✓ xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\F7xx\Src
- ✓ xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\L4xx\Src

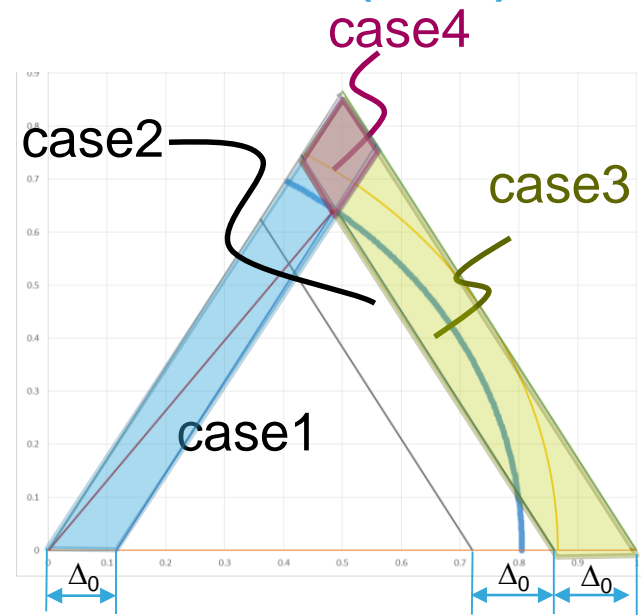
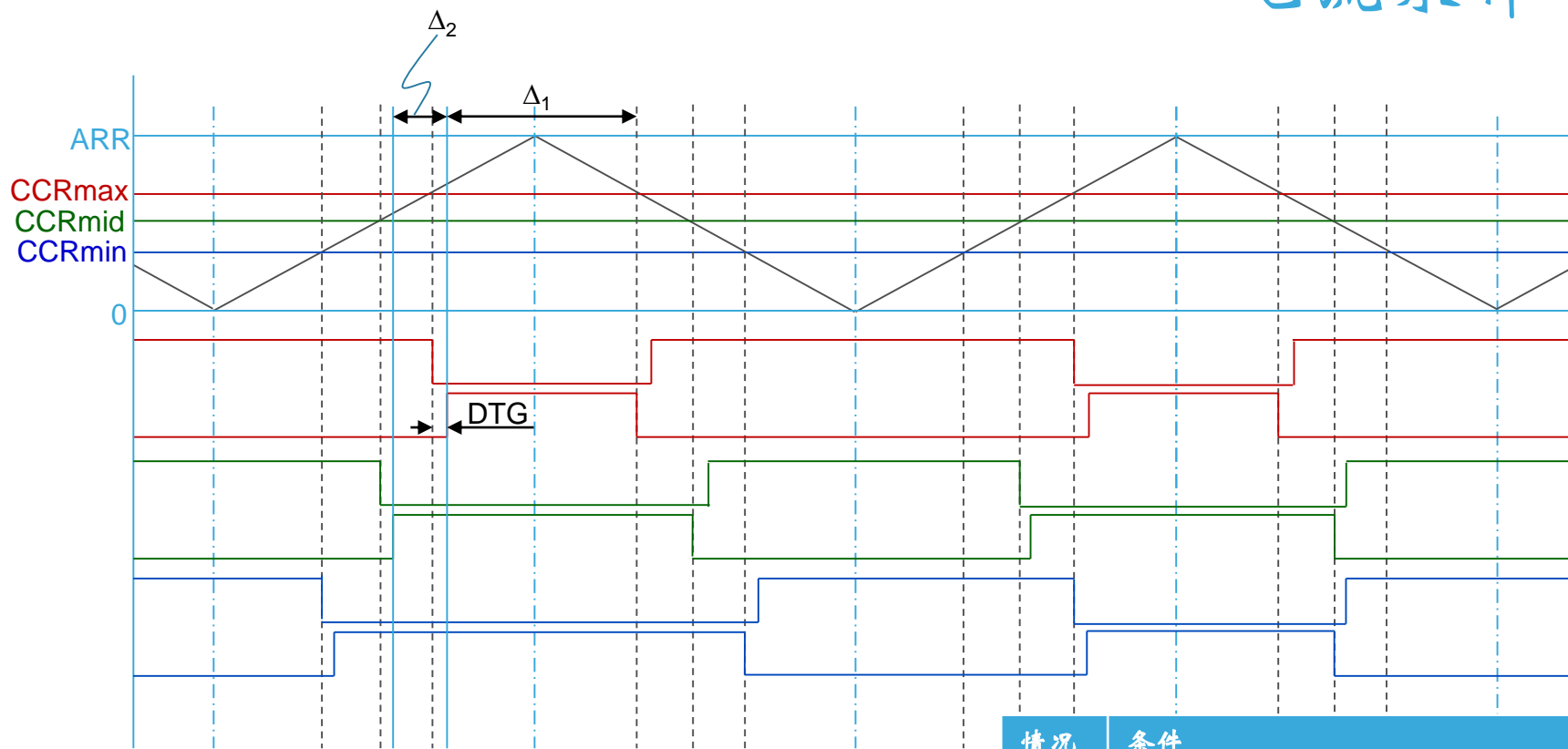
## ➤ 文件名称

- ✓ ics\_yxx\_pwm\_curr\_fdbk.c
- ✓ y=f1, f3, f4, f7, l4



```
Functions
ics_f30x_pwm_curr_fdbk.c
IF3XX_ADC_SetSamplingTime (PWMC_Handle_t * pHand
IF3XX_BRK_IRQHandler (PWMC_ICS_F3_Handle_t * pHand
IF3XX_BRK2_IRQHandler (PWMC_ICS_F3_Handle_t * pHar
IF3XX_CurrentReadingCalibration (PWMC_Handle_t * pHi
IF3XX_ExecRegularConv (PWMC_Handle_t * pHandle, uin
IF3XX_GetPhaseCurrents (PWMC_Handle_t * pHandle, Cu
IF3XX_HFCurrentsCalibration (PWMC_Handle_t * pHandl
IF3XX_Init (PWMC_ICS_F3_Handle_t * pHandle)
IF3XX_IsOverCurrentOccurred (PWMC_Handle_t * pHand
IF3XX_SwitchOffPWM (PWMC_Handle_t * pHandle)
IF3XX_SwitchOnPWM (PWMC_Handle_t * pHandle)
IF3XX_TIMx_UP_IRQHandler (PWMC_ICS_F3_Handle_t * p
IF3XX_TIMxInit (TIM_TypeDef * TIMx, PWMC_ICS_F3_Han
IF3XX_TurnOnLowSides (PWMC_Handle_t * pHandle)
IF3XX_WriteTIMRegisters (PWMC_Handle_t * pHandle)
```





$\Delta_1 = 2 * (ARR - CCRmax - DTG)$   
 $\Delta_2 = CCRmax - CCRmid$   
 $\Delta_0 = CNT\_Ton + CNT\_Tring + CNT\_TADCSH(COV),$   
 $Tring > TADCsta$   
 $\Delta_0 = CNT\_Ton + CNT\_TADCsta + CNT\_TADCSH(COV), TADCsta \geq Tring$

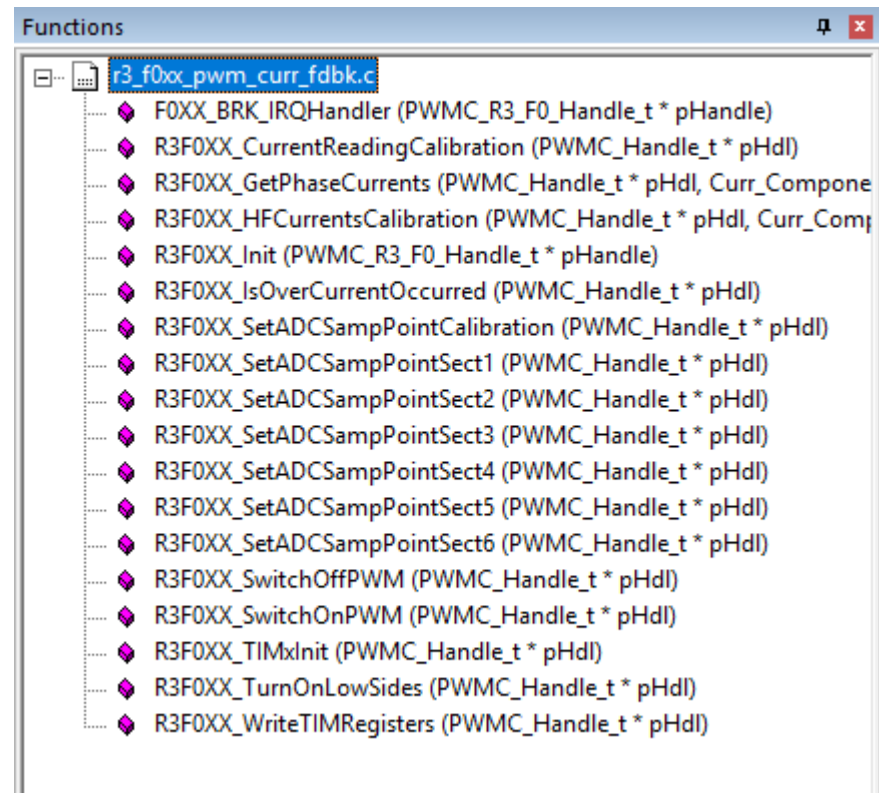
情况	条件	采样点
1	$\Delta_1 > \max(2 * (CNT\_Ton + CNT\_Trise + CNT\_Ring + tdead/2), CNT\_TADCsta + CNT\_TADCSH(COV) - tdead/2)$	Middle of PWM
2	$\Delta_1 > \Delta_0$	CCRmax + tdead + ton + tring + ε
3	$\Delta_2 > \Delta_0 > \Delta_1$	CCRmid + tdead + ton + tring + ε
4	$\Delta_1 < \Delta_0$ and $\Delta_2 < \Delta_0$	Not available

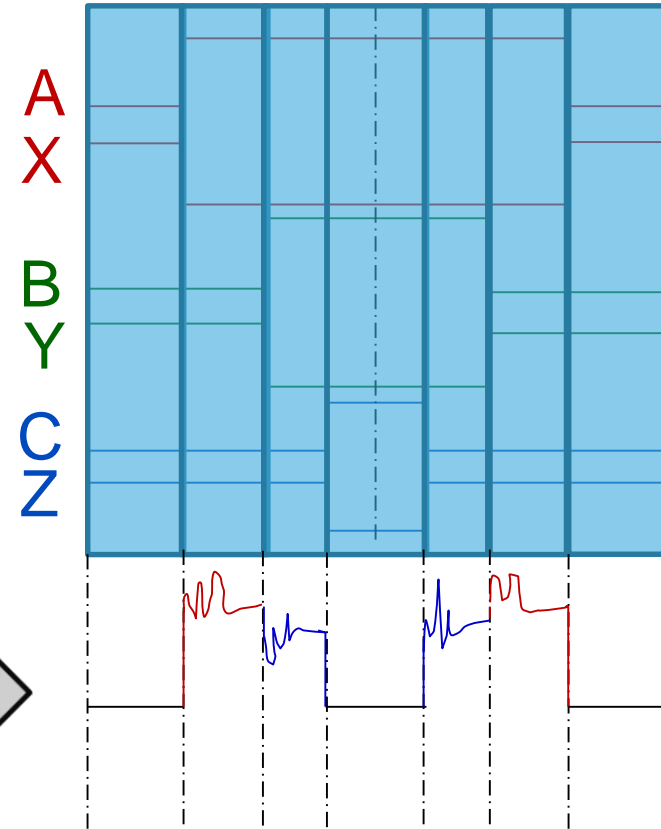
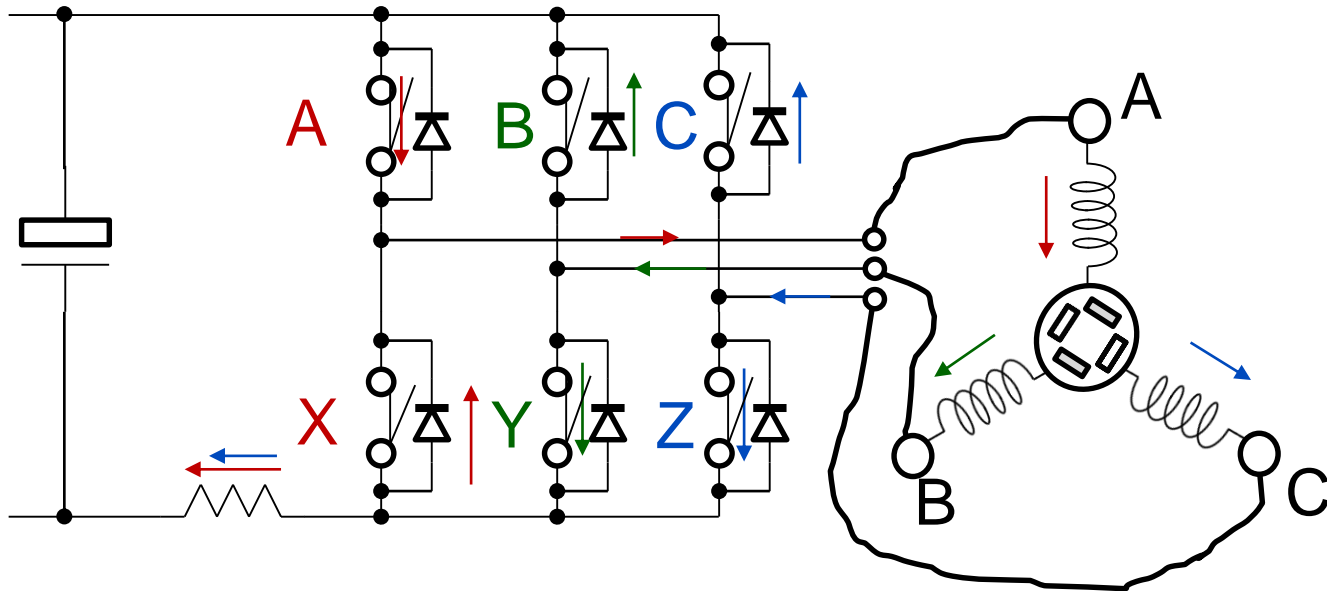
## □ 针对STM32系列芯片都有各自的文件：

1. xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\F0xx\Src
2. xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\F1xx\Src
3. xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\F3xx\Src
4. xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\F4xx\Src
5. xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\F7xx\Src
6. xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\L4xx\Src

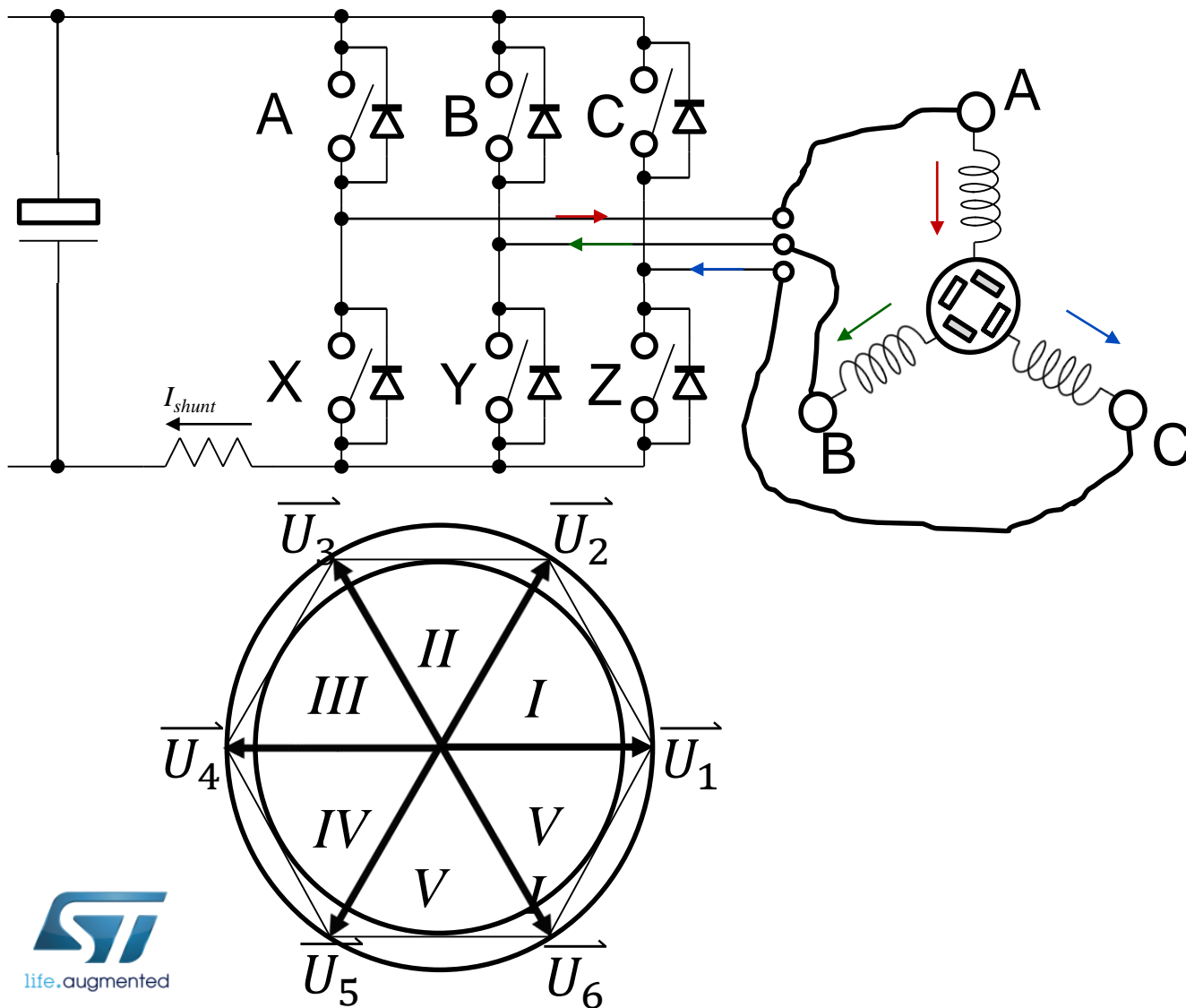
## □ 文件名称

- r3\_z\_yxx\_pwm\_curr\_fdbk.c
  - ✓ y=f0, f1, f3, f4, f7, l4
  - ✓ z=1,4(for dual motor),[]s





- ❑ 在一个PWM周期内采集两相电流数据
- ❑ 可根据  $I_a + I_b + I_c = 0$  构造出三相电流



□ 空间电压矢量与单电阻检测到的信号对应的相电流的关系

Vector	A(X)	B(Y)	C(Z)	$I_{shunt}$
$\vec{U}_0$	OFF(ON)	OFF(ON)	OFF(ON)	0
$\vec{U}_1$	ON(OFF)	OFF(ON)	OFF(ON)	$i_A$
$\vec{U}_2$	ON(OFF)	ON(OFF)	OFF(ON)	$-i_C$
$\vec{U}_3$	OFF(ON)	ON(OFF)	OFF(ON)	$i_B$
$\vec{U}_4$	OFF(ON)	ON(OFF)	ON(OFF)	$-i_A$
$\vec{U}_5$	OFF(ON)	OFF(ON)	ON(OFF)	$i_C$
$\vec{U}_6$	ON(OFF)	OFF(ON)	ON(OFF)	$-i_B$
$\vec{U}_7$	ON(OFF)	ON(OFF)	ON(OFF)	0

# ST MC SDK5.x 单电阻采样固件

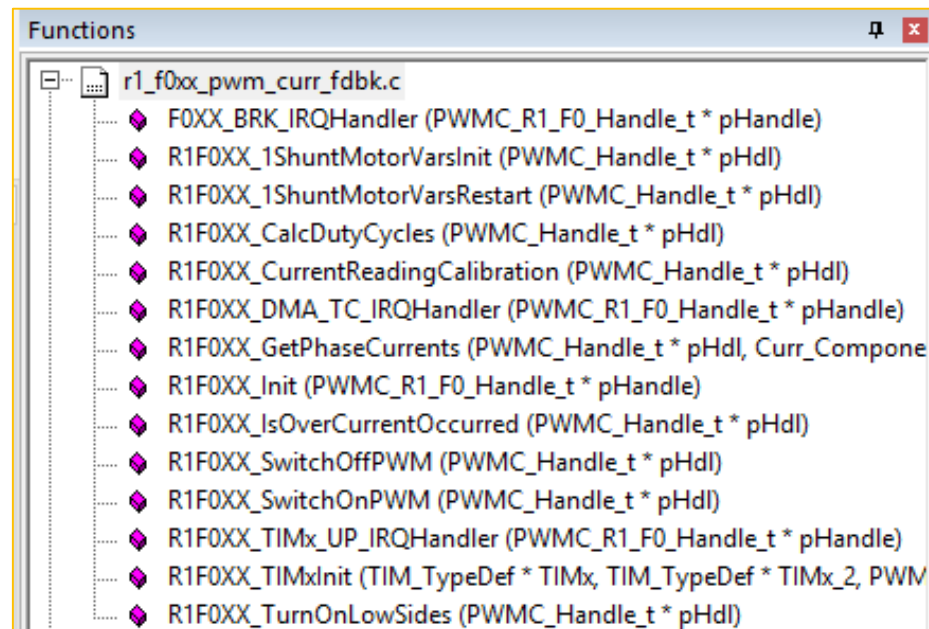
34

## ➤ 针对STM32系列芯片都有各自的文件：

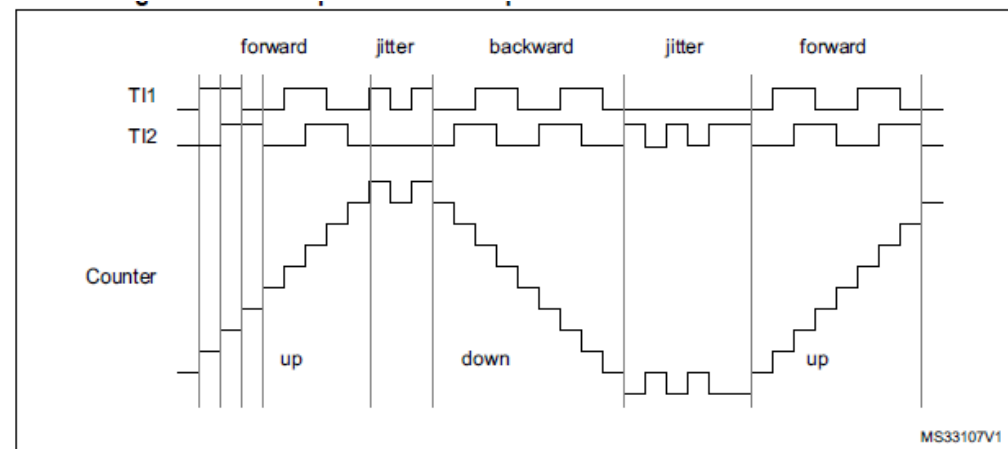
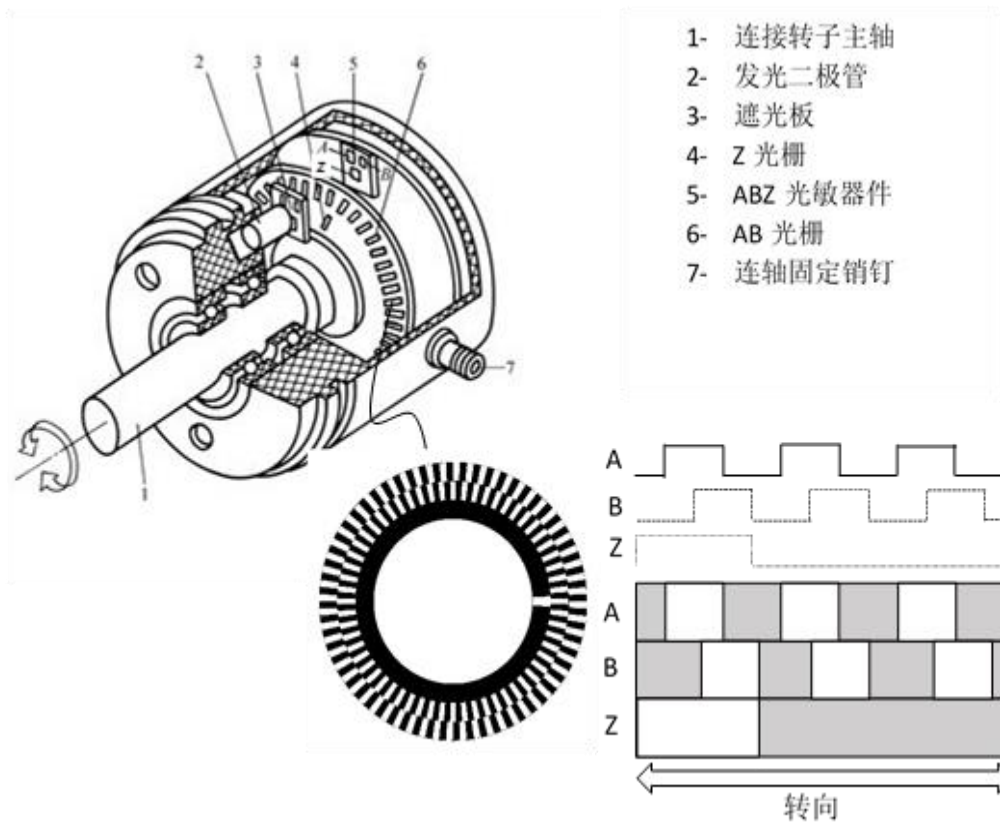
- ✓ xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\F0xx\Src
- ✓ xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\F1xx\Src
- ✓ xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\F3xx\Src
- ✓ xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\F4xx\Src
- ✓ xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\F7xx\Src
- ✓ xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\L4xx\Src

## ➤ 文件名称

- ✓ r1\_yxx\_pwm\_curr\_fdbk.c
- ✓ y=f0, f1, f3, f4, f7, l4



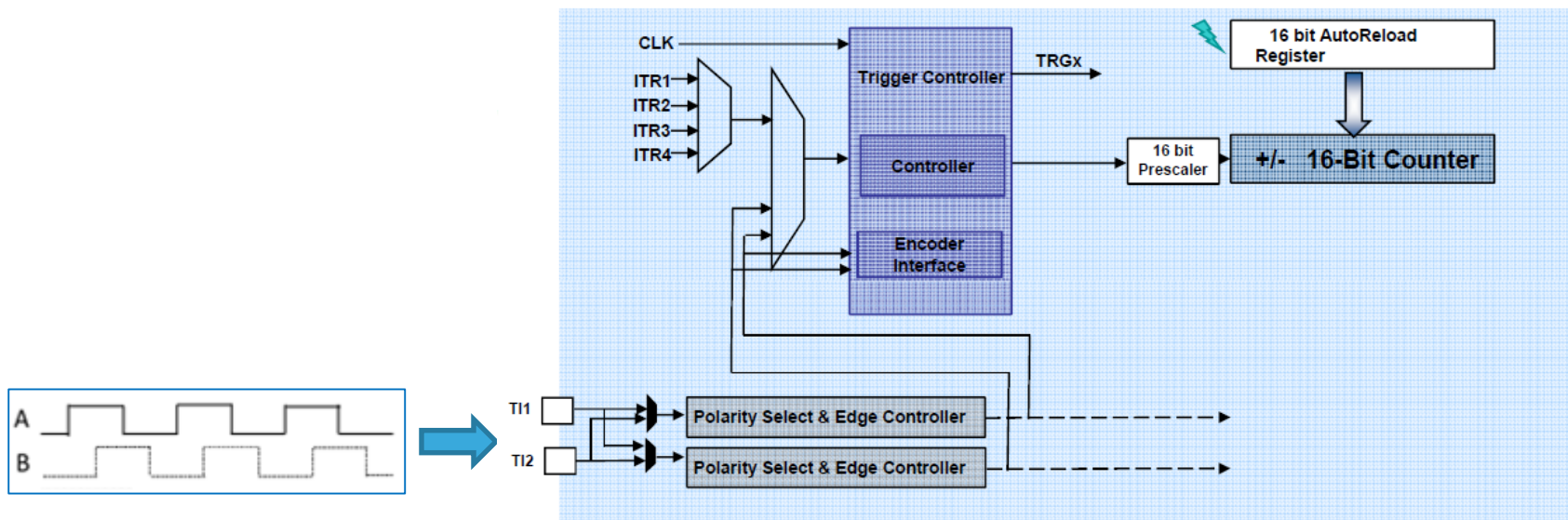
# 位置速度检测 — Encoder



- 使用增量编码器时，在第一次电机启动，任意保护停止或者MCU复位后都要进行预定位操作。
- Z信号（一圈一个）可以使用外部中断或者外部Timer捕捉模式，代表编码器的0度位置，可以用于校准角度位置，可以使用DMA模式对编码器模块赋值；

# STM32 硬件 Encoder 接口

- 在全系列STM32中都有硬件增量编码器Encoder接口
- 每个正交沿都可为加/减计数



## CubeMx 中的配置

The screenshot shows the configuration for TIM2 in STM32CubeMX. The settings are as follows:

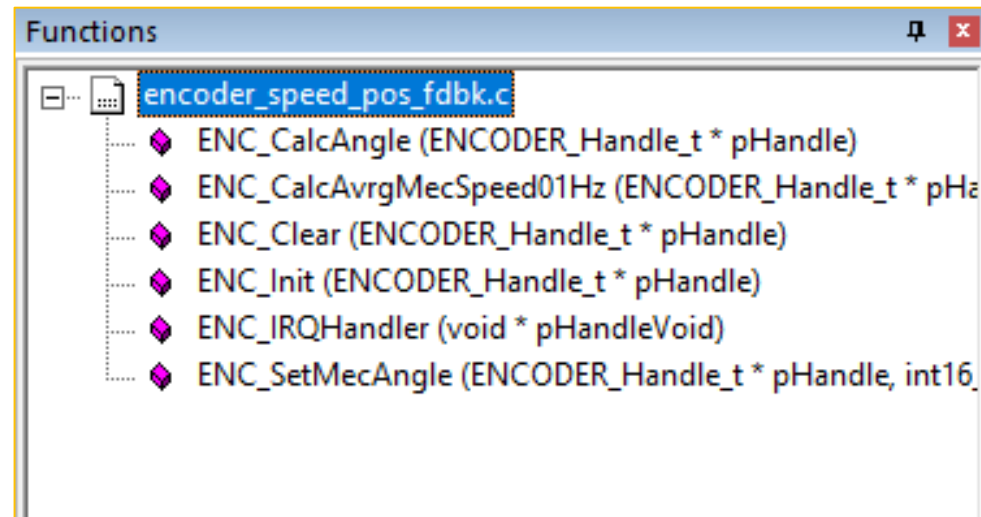
- Slave Mode: Disable
- Trigger Source: Disable
- Clock Source: Disable
- Channel1: Disable
- Channel2: Disable
- Channel3: Disable
- Channel4: Disable
- Combined Channels: Encoder Mode
- ETR IO as Clearing Source
- XOR activation
- One Pulse Mode

➤ 具体文件夹如下:

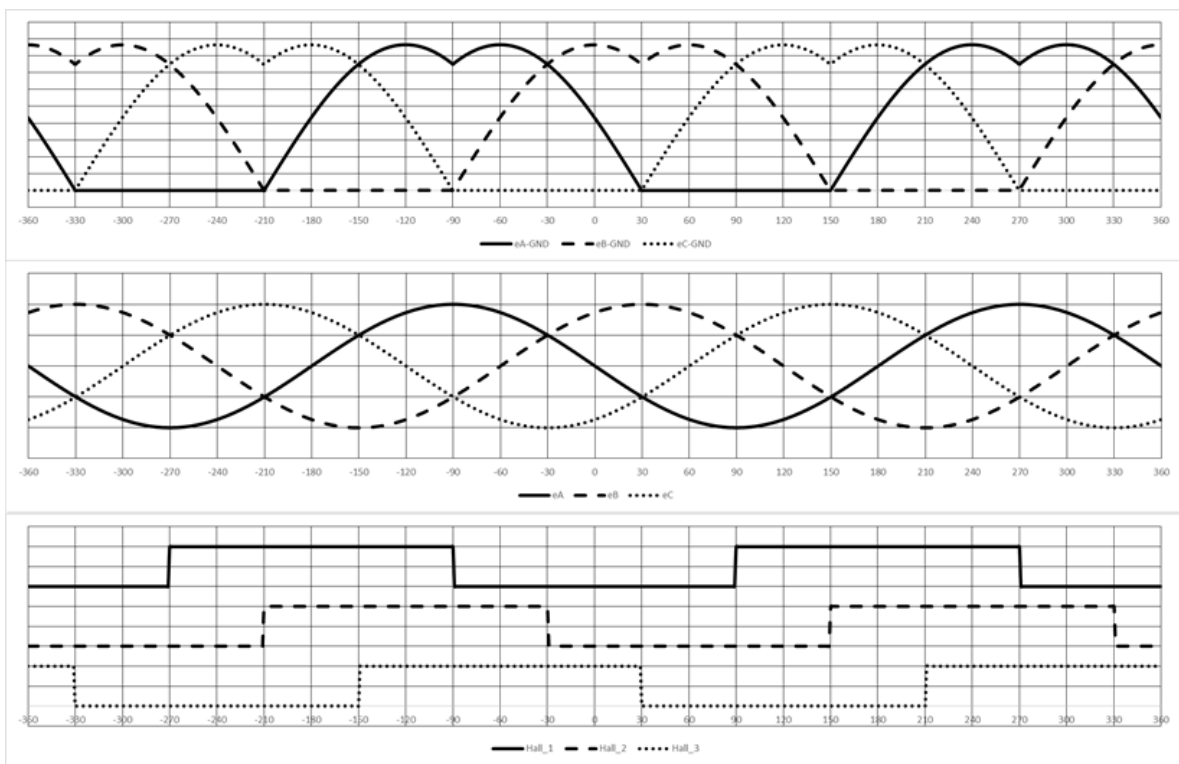
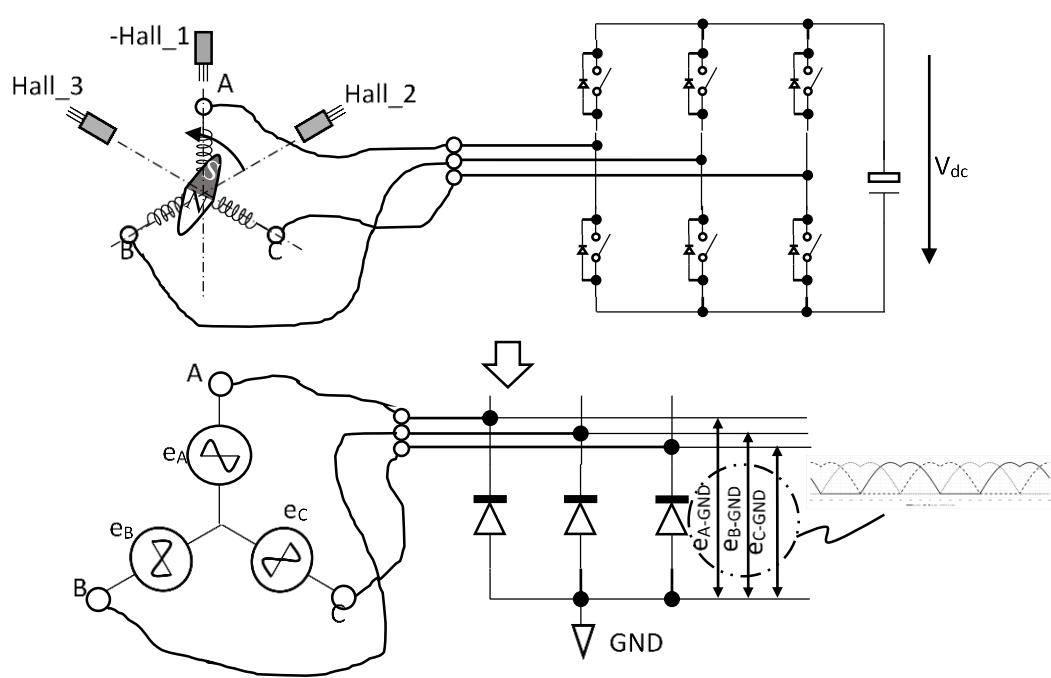
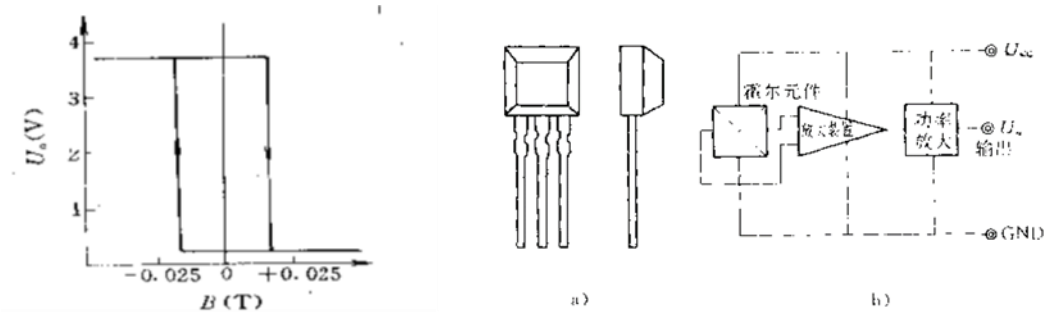
- ✓ xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\Any\Src

➤ 文件名称

- ✓ encoder\_speed\_pos\_fdbk.c

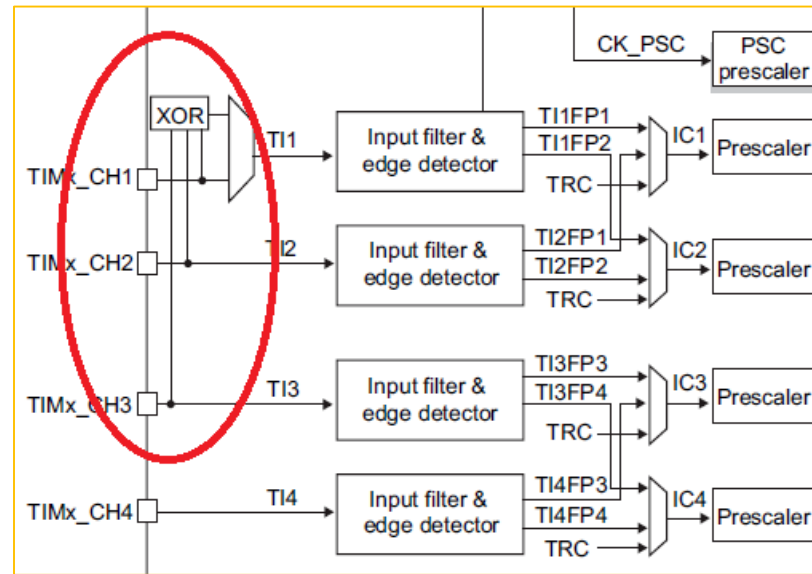
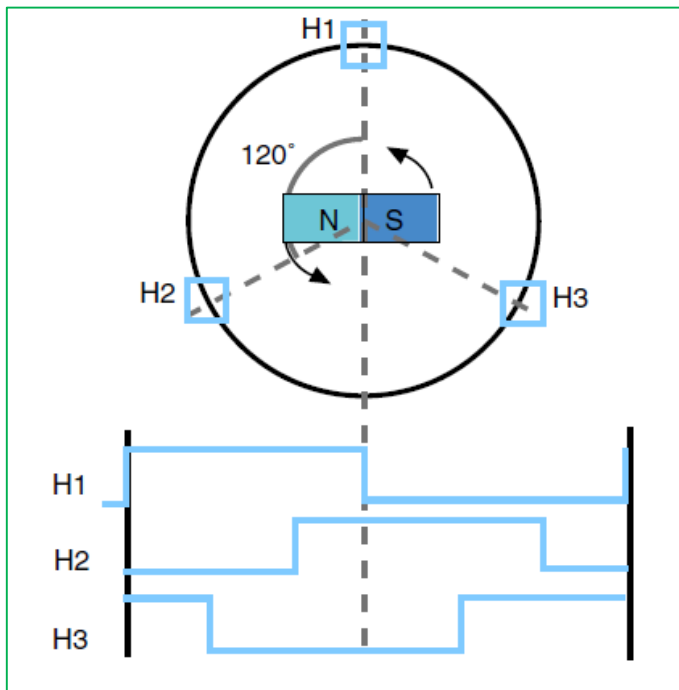


# 位置速度检测 — Hall传感器

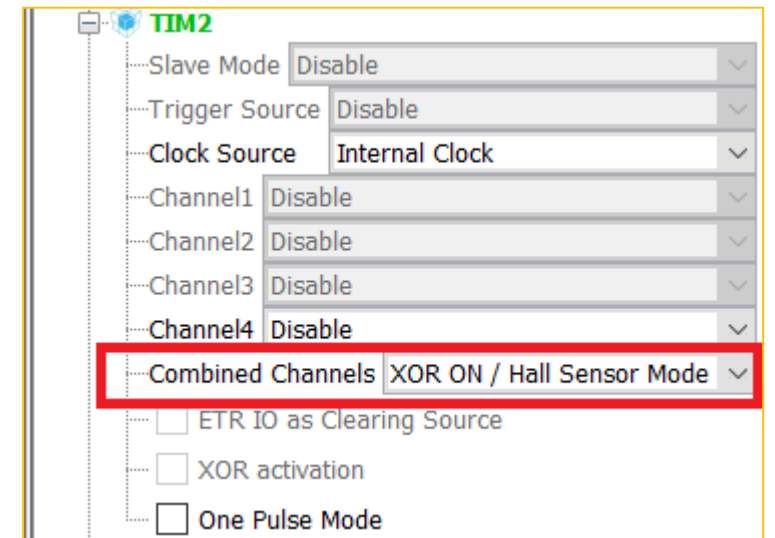


对于60度的Hall信号，可以任意调换三个信号中的任意一个即可得到和120度的处理相似，我们可以很方便使用软件处理。

- 在全系列STM32中都有硬件Hall接口(XOR输入)
- 可以每个Hall跳变沿都产生中断



CubeMx 中的配置



# ST MC SDK5.x Hall传感器固件

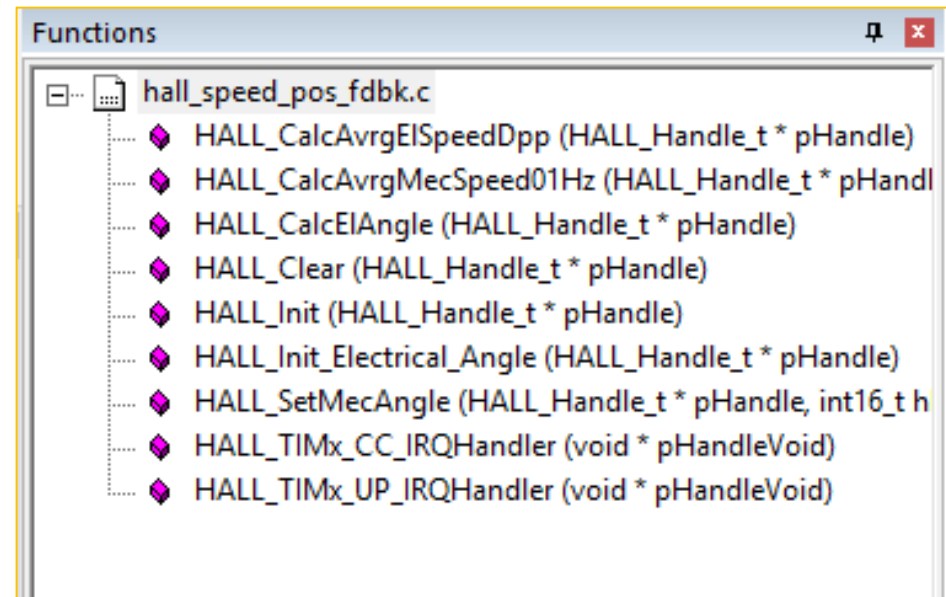
40

## ➤ 具体文件夹如下:

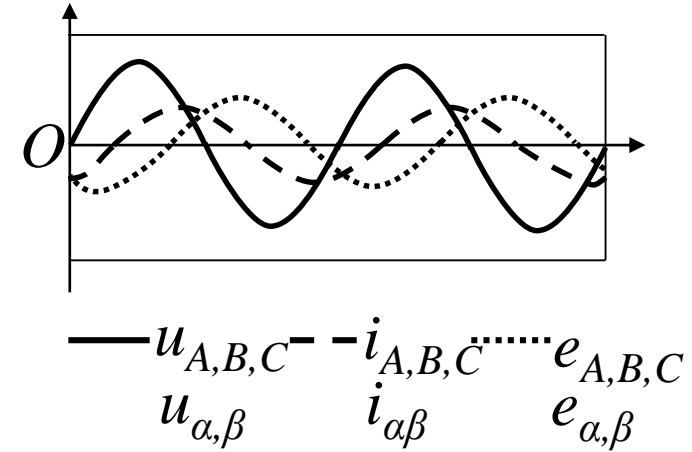
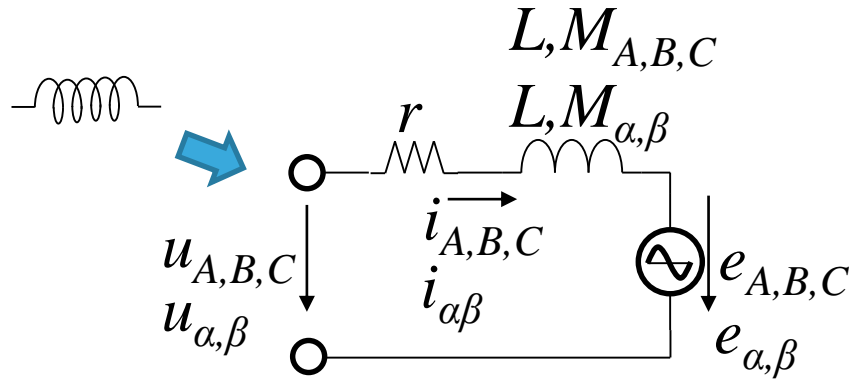
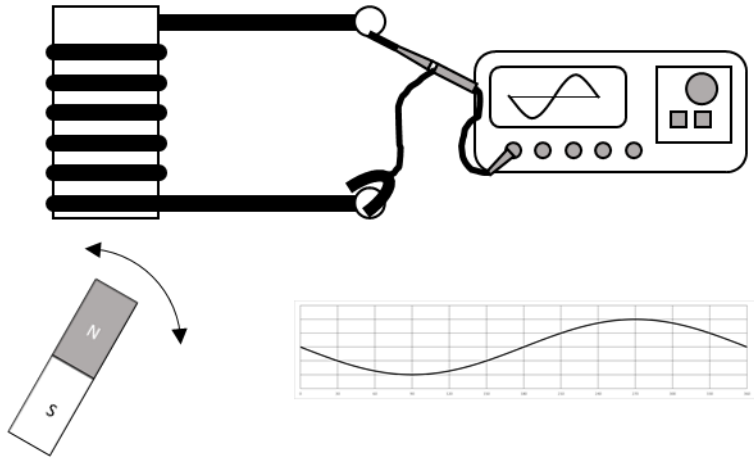
- ✓ xxx\MCSDK\_v5.2.0\MotorControl\MCSDK\MCLib\Any\Src

## ➤ 文件名称

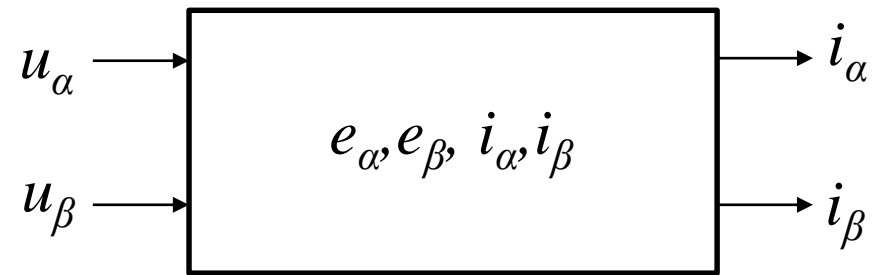
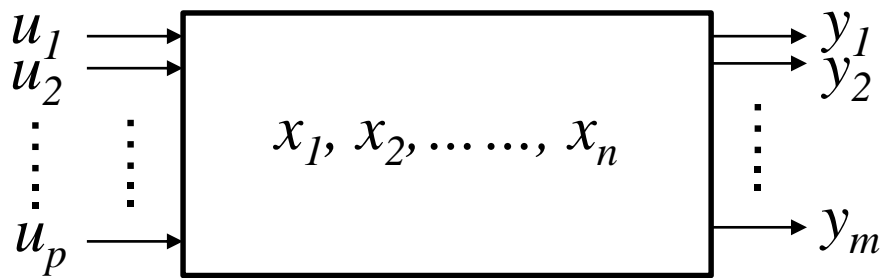
- ✓ hall\_speed\_pos\_fdbk.c



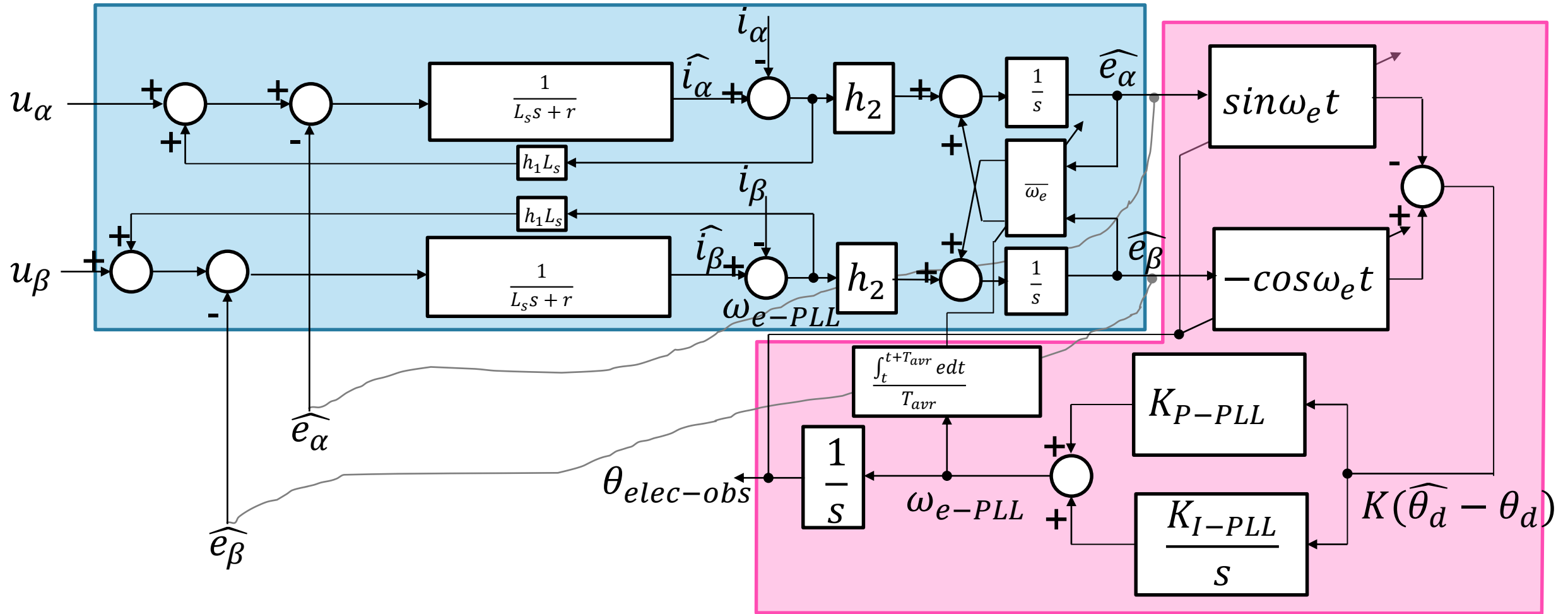
# 位置速度检测 — 观测器(1/2)



$$u \rightarrow i (\tau_e) \rightarrow \omega_r, \omega_e = p \cdot \omega_r \rightarrow e$$



# 位置速度检测 — 观测器(2/2)



OBSERVER

PLL

# ST MC SDK5.x 观测器STO的固件

➤ 对于X-CUBE-MCSDK：观测器的固件以库的形式提供。




✓ xxx\MCSDK\_v5.2.0\MotorControlLib

➤ 头文件名如下：

✓ sto\_cordic\_speed\_pos\_fdbk.h

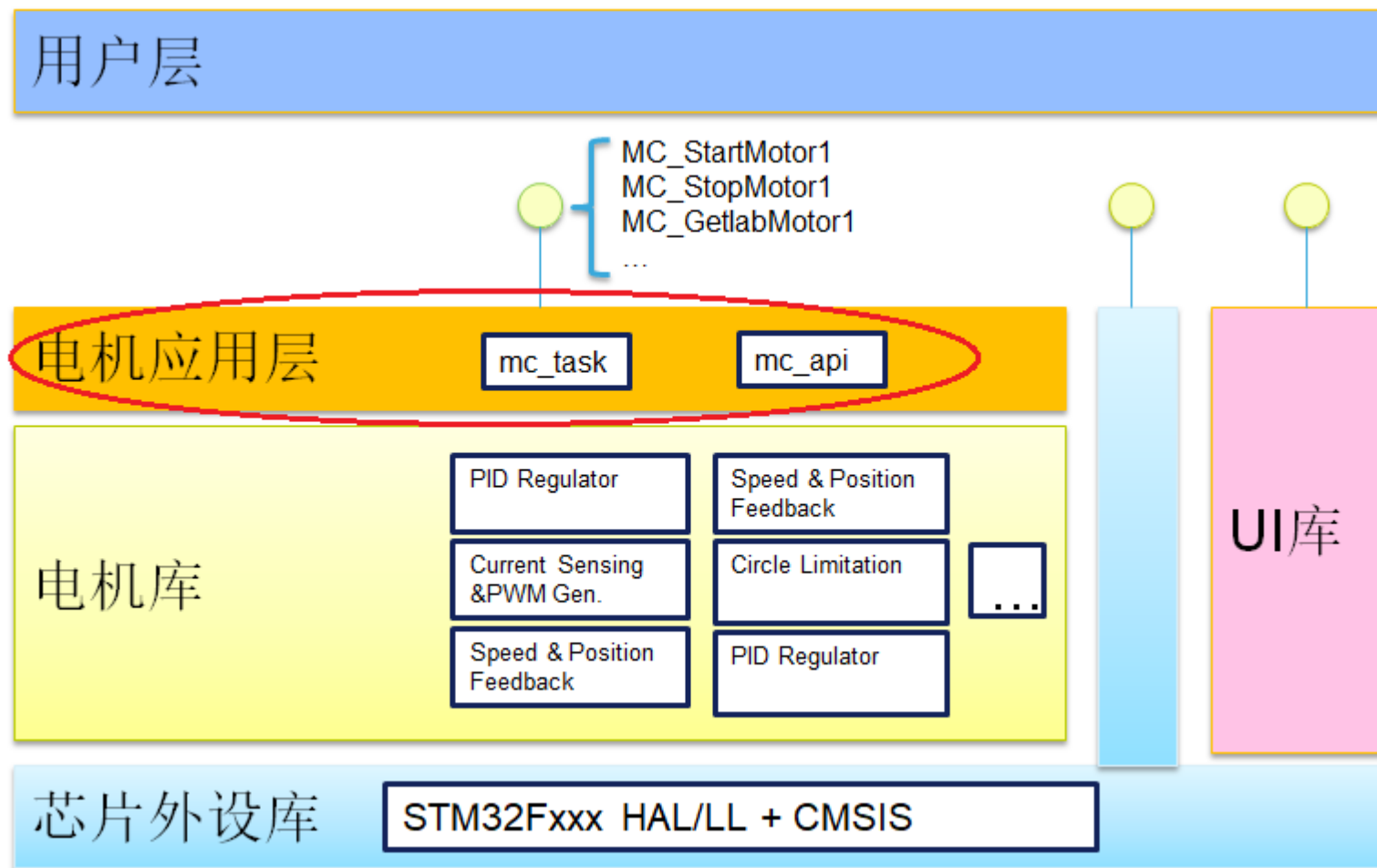
✓ sto\_pll\_speed\_pos\_fdbk.h

✓ sto\_speed\_pos\_fdbk.h

 sto_cordic_speed_pos_fdbk.h	STO+Cordic头文件
 sto_pll_speed_pos_fdbk.h	STO+PLL头文件
 sto_speed_pos_fdbk.h	STO Handle定义

# API函数在MC SDK5.x中的位置

- 一般的电机操作调用API就足够控制基本的电机运行
- 更像是行为描述操作
- 基于电机库，用于用户调用



# API函数列表 (1/2)

函数名称	函数参量	函数返回	函数功能
MC_StartMotor1	void	bool	启动电机
MC_StopMotor1	void	bool	停止电机
MC_ProgramSpeedRampMotor1	speed,time	void	设定速度以及时间
MC_ProgramTorqueRampMotor1	torque,time	void	设定力矩以及时间
MC_SetCurrentReferenceMotor1	Iqref , Idref	void	设定Iq , Id参考
MC_GetCommandStateMotor1	void	MCI_CommandState_t	返回指令执行状态
MC_StopSpeedRampMotor1	void	bool	停止速度指令执行
MC_HasRampCompletedMotor1	void	bool	指令是否执行完成
MC_GetMecSpeedReferenceMotor1	void	int16	返回机械参考速度
MC_GetMecSpeedAverageMotor1	void	int16	返回平均机械速度
MC_GetLastRampFinalSpeedMotor1	void	int16	返回上次指令速度
MC_GetControlModeMotor1	void	STC_Modality_t	返回控制模式
MC_GetImposedDirectionMotor1	void	int16	返回电机转动方向
MC_GetSpeedSensorReliabilityMotor1	void	bool	返回当前速度传感器是否可信
MC_GetPhaseCurrentAmplitudeMotor1	void	Is	返回电流值

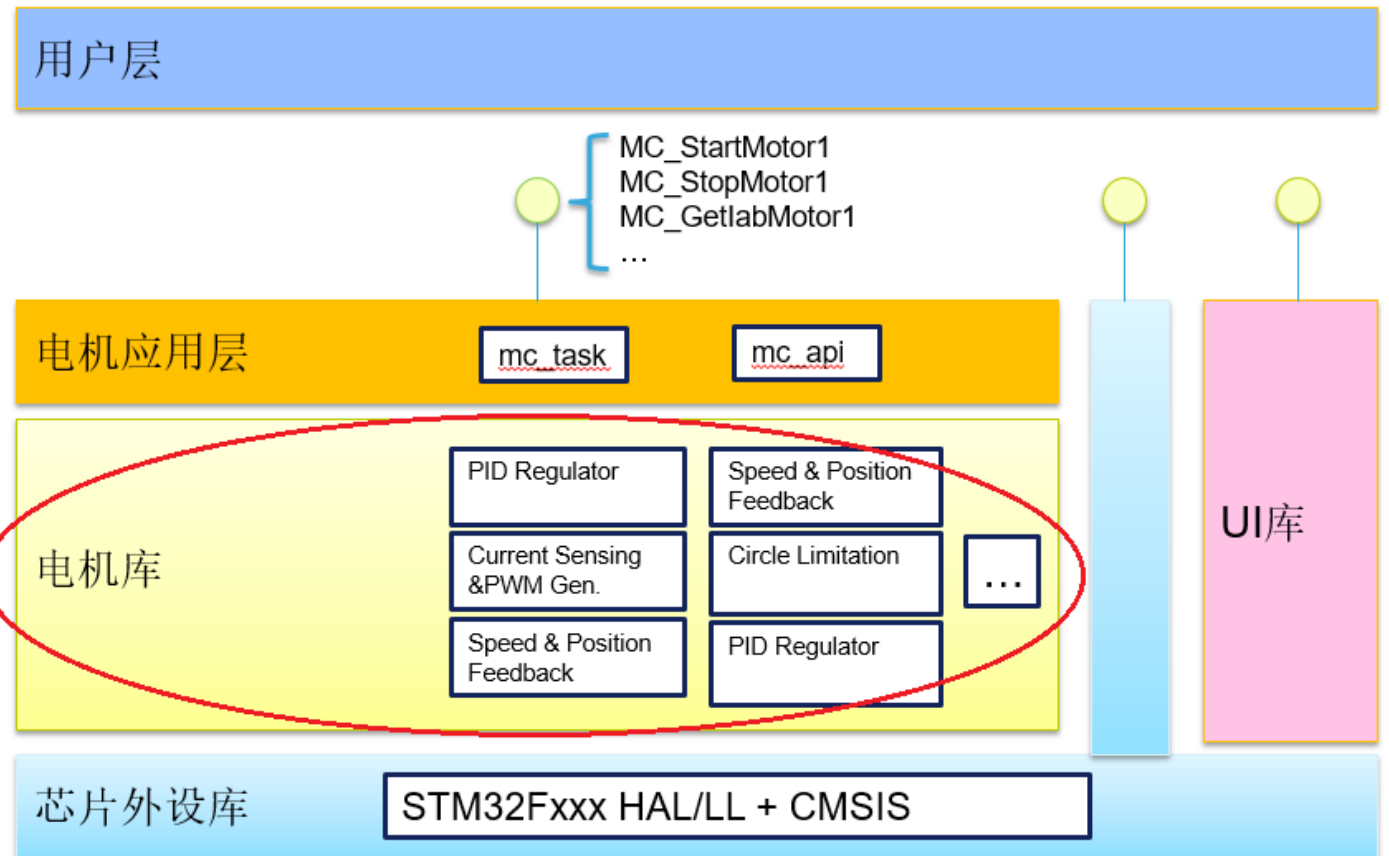
# API函数列表 (2/2)

函数名称	函数参量	函数返回	函数功能
MC_GetPhaseVoltageAmplitudeMotor1	void	Vs	返回电压值
MC_GetIabMotor1	void	Ia · Ib	返回a, b相电流
MC_GetIalphabetaMotor1	void	I $\alpha$ · I $\beta$	返回clark变换后的I $\alpha$ , I $\beta$
MC_GetIqdMotor1	void	Id · Iq	返回park变换后的Id, Iq
MC_GetIqdfrefMotor1	void	Idref · Iqref	返回Id, Iq参考
MC_GetVqdMotor1	void	Vd · Vq	返回变换电压量Vd, Vq
MC_GetValphabetaMotor1	void	V $\alpha$ · V $\beta$	返回变换电压量V $\alpha$ , V $\beta$
MC_GetElAngledppMotor1	void	Angle dpp	返回电角度DPP数据
MC_GetTerefMotor1	void	Iqref	返回电流参考
MC_SetIdrefMotor1	Idref	void	设定电流Id参考
MC_Clear_IqdfrefMotor1	void	void	Iq, Id数据回到默认值
MC_AcknowledgeFaultMotor1	void	bool	清除异常状态
MC_GetOccurredFaultsMotor1	void	Fault	得到发生了的Fault状态
MC_GetCurrentFaultsMotor1	void	Fault	得到当前的Fault状态
MC_GetSTMStateMotor1	void	State	得到电机状态

# 电机库在MC SDK5.x中的位置

给电机控制开发者  
更大发挥空间!!!

- 涉及底层操作
- 可以调用在API中没有涉及的函数
- 修改需要熟悉电机运行框架
- 可以根据实际需求修改对应代码

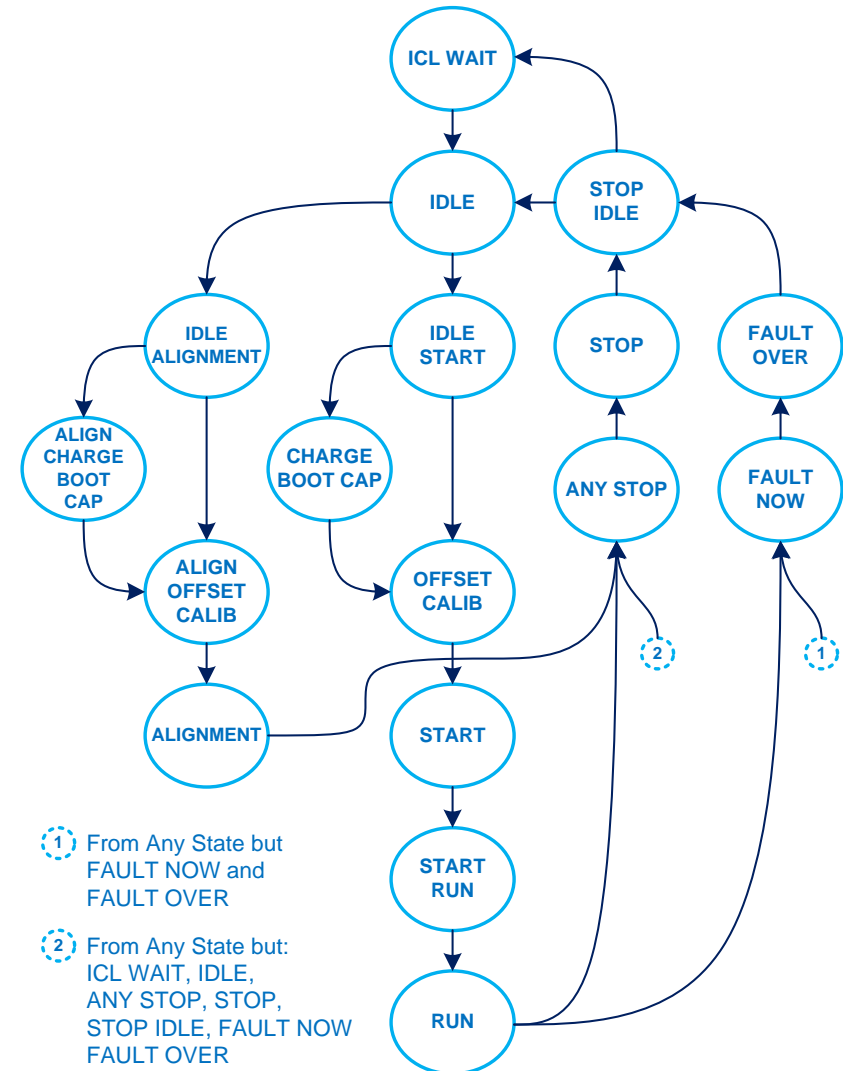


# 电机库底层源文件

源文件	描述
bus_voltage_sensor.c	母线电压
circle_limitation.c	电压极限限制
enc_align_ctrl.c	编码器初始定位控制
encoder_speed_pos_fdbk.c	编码器传感器相关
hall_speed_pos_fdbk.c	Hall传感器相关
mc_math.c	数学计算
motor_power_measurement.c	平均功率计算
ntc_temperature_sensor.c	NTC温度传感
open_loop.c	开环控制
pid_regulator.c	PID环路控制
pqd_motor_power_measurement.c	功率计算
pwm_common.c	TIMER同步使能
pwm_curr_fdbk.c	SVPWM, ADC设定相关接口
r_divider_bus_voltage_sensor.c	实际母线电压采集
virtual_bus_voltage_sensor.c	虚拟母线电压
ramp_ext_mgr.c	无传感开环转闭环控制
speed_pos_fdbk.c	速度传感接口
speed_torq_ctrl.c	速度力矩控制
state_machine.c	电机状态相关
virtual_speed_sensor.c	无传感开环运行相关

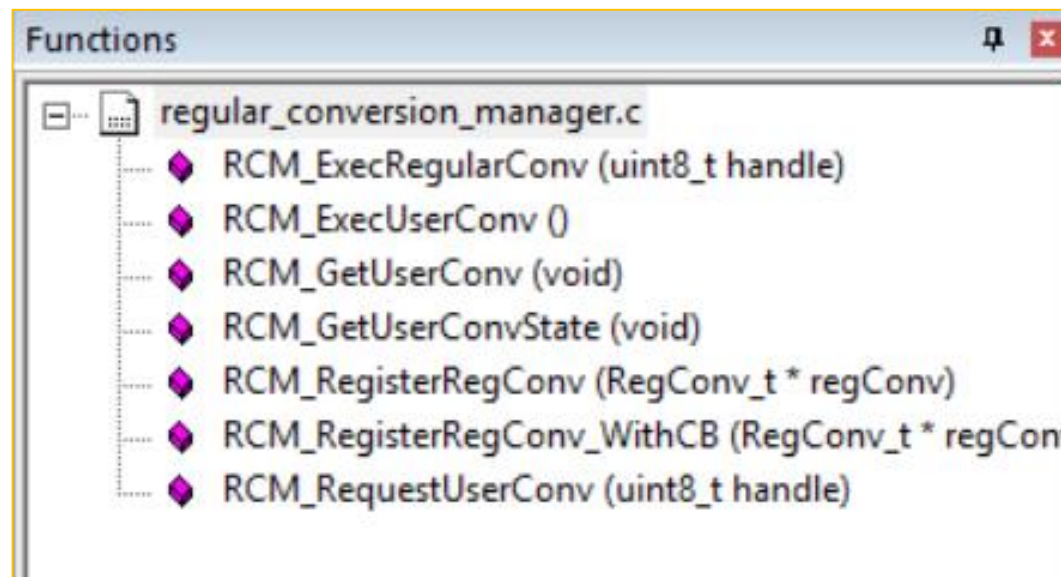
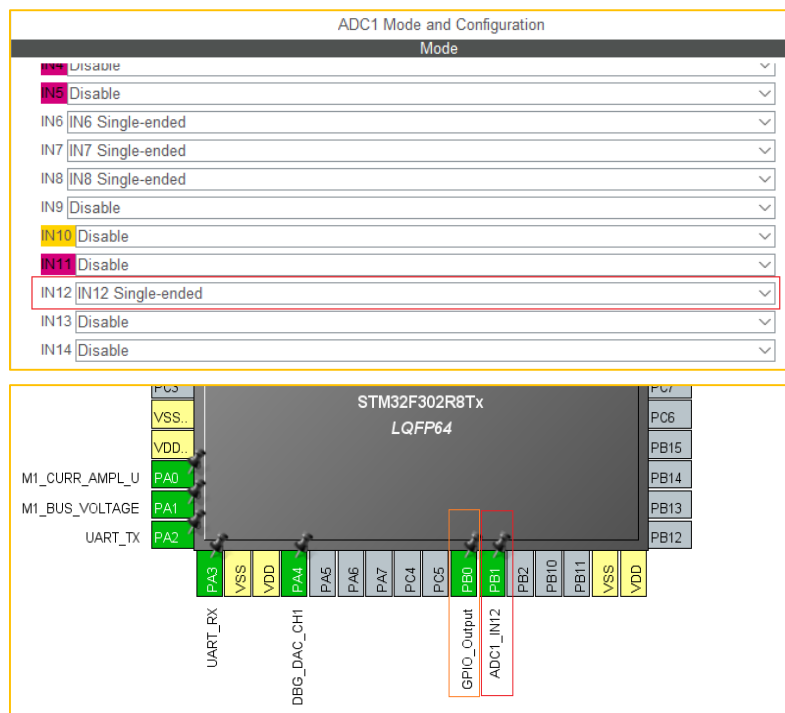
# 电机控制状态机

- 程序中都有相应的状态机来控制电机状态。
- 用户不可以直接操作状态机，而应该通过 SDK 提供的 API 来改变电机状态。
- 如果添加客户自定义状态机，需要符合电机控制库规则。



# 可扩展性增强-增加其他ADC采样

- 方便客户增加自行的ADC采样，同时又不会影响电机控制的ADC采样
- 电机库特意为客户增加了相关可调用函数
- 文件名称：regular\_conversion\_manager.c



# 可扩展性增强-增加客户程序

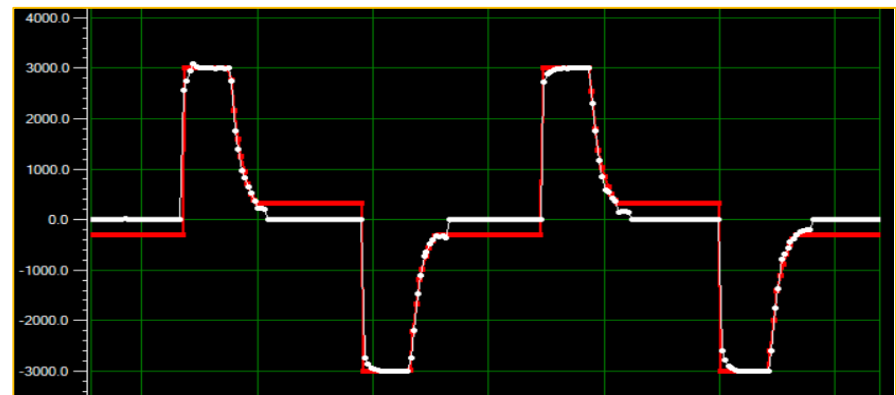
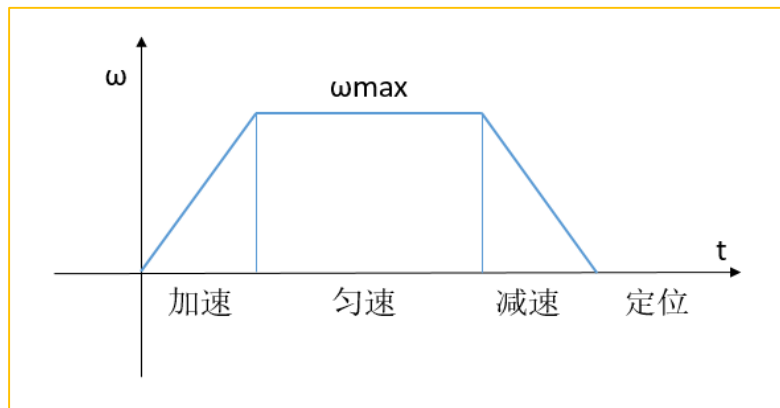
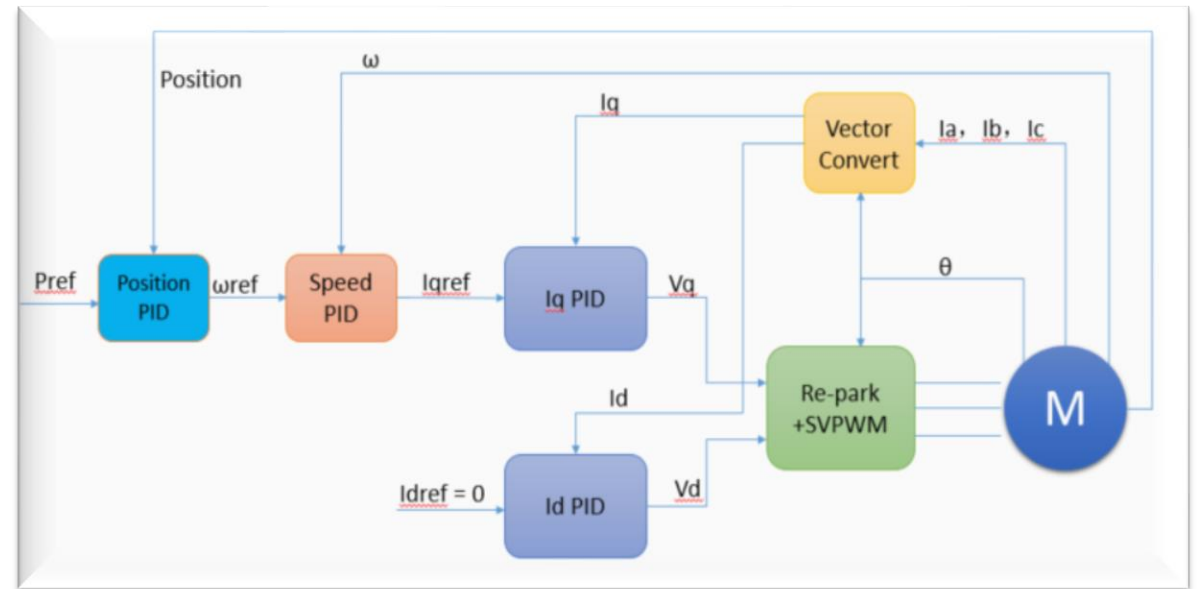
51

➤ 清晰的架构可以很方便添加客户程序

✓ 举例：添加位置环

❖ 仿照速度环路增加位置环路

- 位置环PID结构体
- 位置控制的结构体
- 位置环速度参考输出计算
- 位置环力矩参考输出计算



➤ 访问[www.st.com](http://www.st.com)，可找到链接进行下载



版本	代码	下载
X-CUBE-MCSDK	开源代码，除观测器，MTPA，弱磁，电流前馈外	可直接下载
X-CUBE-MCSDK -FUL	开源代码	需要邮箱注册审批

# STM32的电机相关资源列表

	中文培训材料	线上培训视频
电机	 <p>2019 STM32电机培训</p>	 <p>2019 STM32电机视频</p> <ul style="list-style-type: none"><li>- 1,ST MC SDK5.x概览</li><li>- 2,永磁同步电动机矢量控制基础</li><li>- 3,电动机相电流检测与重构方法及转子位置检测与估计方法</li><li>- 4,ST MC SDK 5.2 WB应用指南及ST MC SDK5.2 固件详解</li><li>- 5,应用ST MC SDK5.2 及ST 硬件评价板调试电机实例</li></ul>

# Releasing your creativity



- 谢谢 -



life.augmented