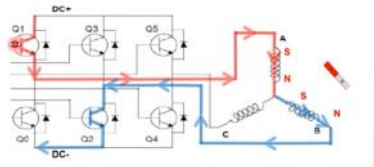
FOC电机

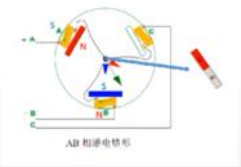
2024年10月30日 13:09

3 无刷电机的硬件控制原理

Designed by ////// DENG FOC



经典的6个MOS管控制无刷电机的电路



星形无刷电机通电时的磁场

FOC算法流程图

Designed by ////// DENG FOC

3 克拉克变换-推导

Designed by ////// DENG FOC

显然，针对 $\alpha-\beta$ 坐标系中的 α 轴

$$I_\alpha = i_a - \sin 30^\circ i_b - \cos 60^\circ i_c$$

$$I_\alpha = i_a - \frac{1}{2} i_b - \frac{1}{2} i_c$$

针对 $\alpha-\beta$ 坐标系中的 β 轴

$$I_\beta = \cos 30^\circ i_b - \cos 30^\circ i_c$$

$$I_\beta = \frac{\sqrt{3}}{2} i_b - \frac{\sqrt{3}}{2} i_c$$

写成矩阵

$$\begin{bmatrix} I_\alpha \\ I_\beta \end{bmatrix} = \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix}$$

克拉克变换 基本形式！！

$$\begin{bmatrix} I_\alpha \\ I_\beta \end{bmatrix} = \frac{2}{3} \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix}$$

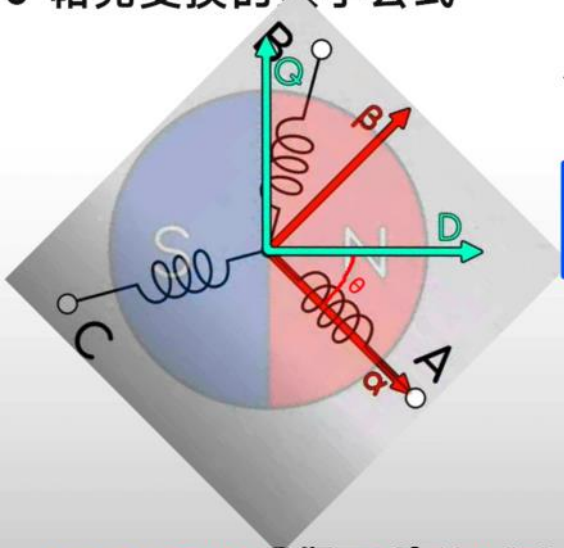
克拉克变换 等幅值形式

$$\begin{cases} i_a = I_\alpha \\ i_b = \dfrac{\sqrt{3} I_\beta - I_\alpha}{2} \\ i_c = \dfrac{-I_\alpha - \sqrt{3} I_\beta}{2} \end{cases}$$

克拉克变换 等幅值形式 逆变换

3 帕克变换的数学公式

Designed by ////// DENG FOC



对于 映射到D轴上的α，β分量：

$$i_d = i_\alpha \cos\theta + i_\beta \sin\theta$$

对于 映射到Q轴上的α，β分量：

$$i_q = -i_\alpha \sin\theta + i_\beta \cos\theta$$

写成矩阵形式

$$\begin{bmatrix} i_d \\ i_q \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix}$$

在定子上叠加A 我们可以把i_alpha 在 Q轴 上的分量

3 整个FOC算法数学过程总结

Designed by ////// DENG FOC

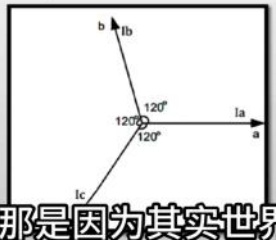
1 电流形式（基于三相电流矢量）

$$\begin{bmatrix} I_\alpha \\ I_\beta \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}^{-1} \begin{bmatrix} I_d \\ I_q \end{bmatrix}$$

$$I_\alpha = I_d\cos\theta - I_q\sin\theta$$

$$I_\beta = I_q\cos\theta + I_d\sin\theta$$

$$\begin{cases} i_a = I_\alpha \\ i_b = \dfrac{\sqrt{3}I_\beta - I_\alpha}{2} \\ i_c = \dfrac{-I_\alpha - \sqrt{3}I_\beta}{2} \end{cases}$$



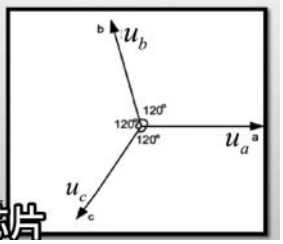
2 电压形式（基于三相电压矢量）

$$\begin{bmatrix} U_\alpha \\ U_\beta \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}^{-1} \begin{bmatrix} U_d \\ U_q \end{bmatrix}$$

$$U_\alpha = U_d\cos\theta - U_q\sin\theta$$

$$U_\beta = U_q\cos\theta + U_d\sin\theta$$

$$\begin{cases} u_a = U_\alpha \\ u_b = \dfrac{\sqrt{3}U_\beta - U_\alpha}{2} \\ u_c = \dfrac{-U_\alpha - \sqrt{3}U_\beta}{2} \end{cases}$$



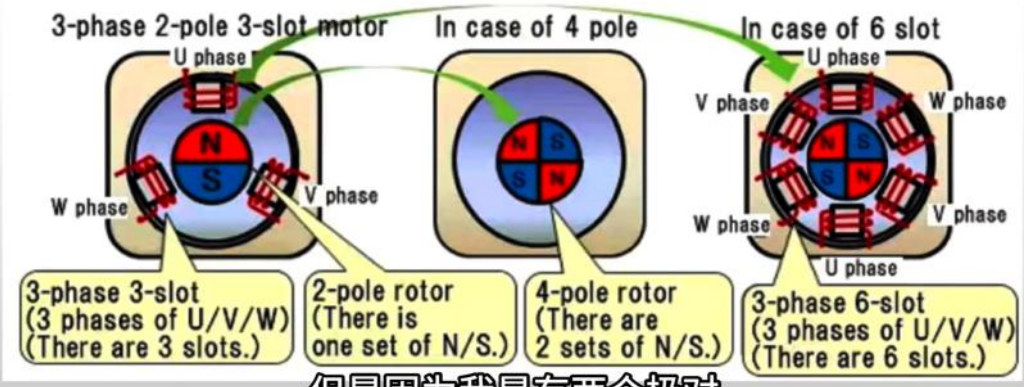
那是因为其实世界上几乎所有的驱动芯片

5 电角度和机械角度的关系

Designed by ////// DENG FOC

电角度=机械角度x极对数

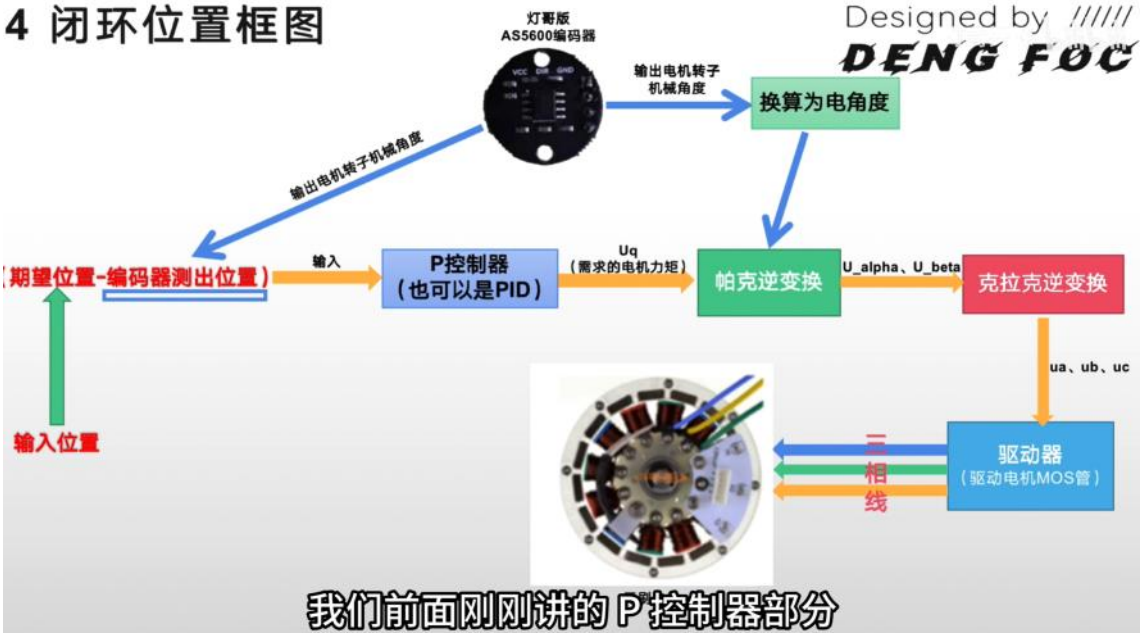
电机的极数就是电动机的磁极数，磁极分N极和S极，一般磁极数是成对出现，如2极电机，4极电机，一般把1个N极和1个S极称为一对磁极，也就是极对数为1



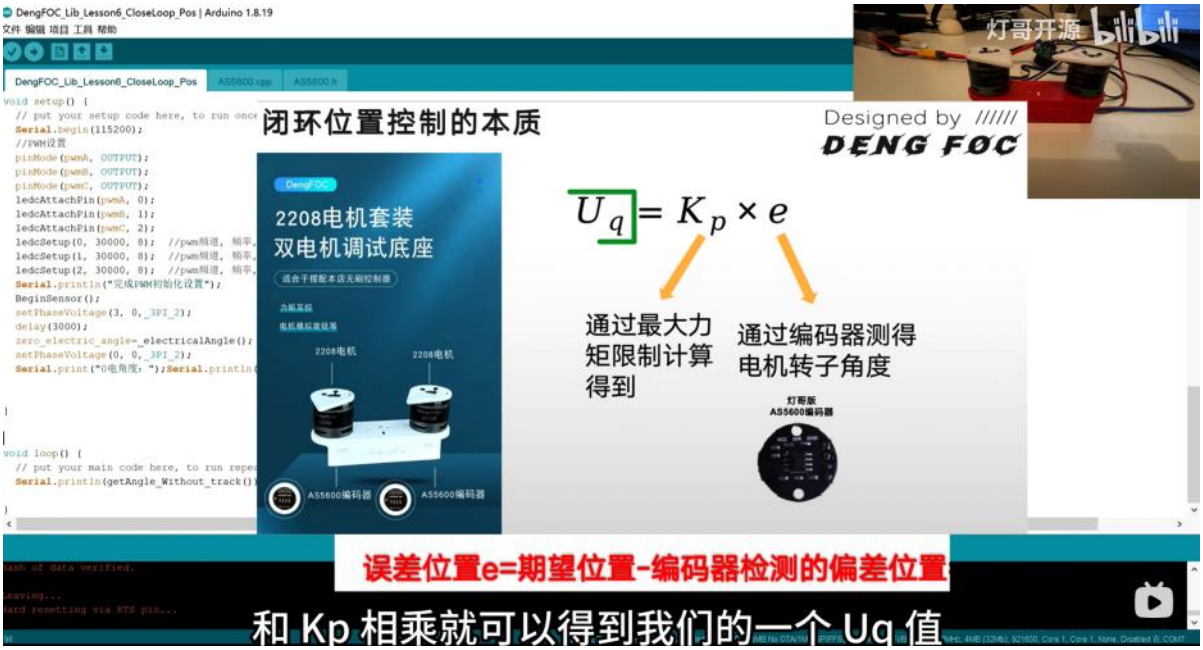
但是因为我是有两个极对

4 闭环位置框图

Designed by ////// DENG FOC

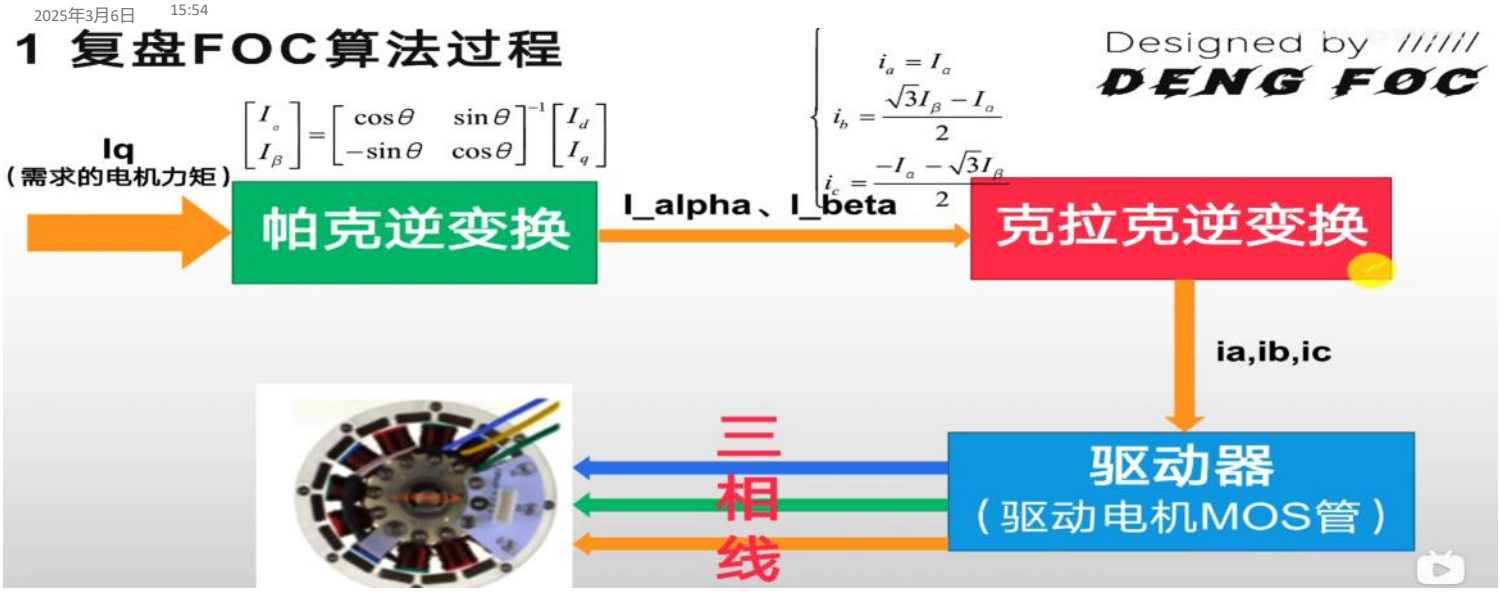


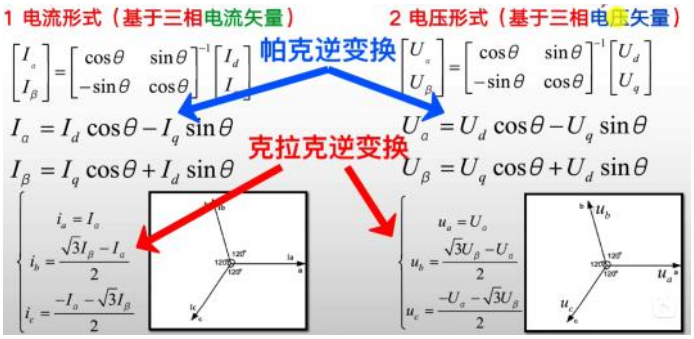
我们前面刚刚讲的P控制器部分


DengFOC_Lib_Lesson6_CloseLoop_Pos | Arduino 1.8.19
文件 编辑 项目 工具 帮助
闭环位置控制的本质
Designed by //////
DENG FOC
灯哥开源
2208电机套装
双电机调试底座
适合于搭配本店无刷控制器
2208电机
2208电机
AS5600编码器
AS5600编码器
$U_q = K_p \times e$
通过最大力矩限制计算得到
通过编码器测得电机转子角度
灯哥版
AS5600编码器
误差位置e=期望位置-编码器检测的偏差位置
和 Kp 相乘就可以得到我们的一个 Uq 值

123.DengFoc开环

2025年3月6日 15:54





电角度=机械角度x极对数

```
/* USER CODE BEGIN Header */
/**
  ******************************************************************************
  * @file           : main.c
  * @brief          : Main program body
  ******************************************************************************
  * @attention
  *
  * Copyright (c) 2025 STMicroelectronics.
  * All rights reserved.
  *
  * This software is licensed under terms that can be found in the LICENSE file
  * in the root directory of this software component.
  * If no LICENSE file comes with this software, it is provided AS-IS.
  *
  ******************************************************************************
  */
/* USER CODE END Header */
/* Includes ------------------------------------------------------------------*/
#include "main.h"
#include "i2c.h"
#include "tim.h"
#include "usart.h"
#include "gpio.h"
/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */
#include "OLED.h"
#include <stdio.h>
#include "Heryuan.h"
```

```c
#include <math.h>
#define PWM_A_PIN GPIO_PIN_1
#define PWM_A_PORT GPIOA
#define PWM_B_PIN GPIO_PIN_2
#define PWM_B_PORT GPIOA
#define PWM_C_PIN GPIO_PIN_3
#define PWM_C_PORT GPIOA
#define _constrain(amt, low, high) ((amt) < (low) ? (low) : ((amt) > (high) ? (high) :
(amt)))
#define PI 3.14159265358979323846f
float voltage_power_supply = 12.6f;
float shaft_angle = 0, open_loop_timestamp = 0;
float zero_electric_angle = 0, Ualpha, Ubeta = 0, Ua = 0, Ub = 0, Uc = 0, dc_a = 0, dc_b =
0, dc_c = 0;
/* USER CODE END Includes */
/* Private typedef -----------------------------------------------------------*/
/* USER CODE BEGIN PTD */
/* USER CODE END PTD */
/* Private define ------------------------------------------------------------*/
/* USER CODE BEGIN PD */
/* USER CODE END PD */
/* Private macro -------------------------------------------------------------*/
/* USER CODE BEGIN PM */
/* USER CODE END PM */
/* Private variables ---------------------------------------------------------*/
/* USER CODE BEGIN PV */
/* USER CODE END PV */
/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
/* USER CODE BEGIN PFP */
/* USER CODE END PFP */
/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */
int fputc(int ch,FILE *f)
{
  HAL_UART_Transmit(&huart1,(uint8_t *)&ch,1,0xffff);
  return ch;
}

void Sguan_SerialSendByte(uint8_t Byte)
{
  HAL_UART_Transmit(&huart1,&Byte,1,HAL_MAX_DELAY);
}
void Sguan_SerialSendArray(uint8_t *dert,uint16_t Length)
{
  for (uint16_t i = 0; i < Length; i++)
  {
    Sguan_SerialSendByte(dert[i]);
  }
}
void Sguan_SreialSendString(char *str)
{
  for (uint16_t i = 0; str[i] != '\0'; i++)
  {
    Sguan_SerialSendByte(str[i]);
  }
}

float _electricalAngle(float shaft_angle, int pole_pairs)
{
  return (shaft_angle * pole_pairs);
}

float _normalizeAngle(float angle)
{
  float a = fmod(angle, 2 * PI);
```

```
  return a >= 0 ? a : (a + 2 * PI);
}

void setPwm(float Ua, float Ub, float Uc)
{
  dc_a = _constrain(Ua / voltage_power_supply, 0.0f, 1.0f);
  dc_b = _constrain(Ub / voltage_power_supply, 0.0f, 1.0f);
  dc_c = _constrain(Uc / voltage_power_supply, 0.0f, 1.0f);
  __HAL_TIM_SET_COMPARE(&htim2, TIM_CHANNEL_2, (uint32_t)(dc_a * 255));
  __HAL_TIM_SET_COMPARE(&htim2, TIM_CHANNEL_3, (uint32_t)(dc_b * 255));
  __HAL_TIM_SET_COMPARE(&htim2, TIM_CHANNEL_4, (uint32_t)(dc_c * 255));
}

void setPhaseVoltage(float Uq, float Ud, float angle_el) {
  angle_el = _normalizeAngle(angle_el + zero_electric_angle);
  Ualpha = -Uq * sin(angle_el);
  Ubeta = Uq * cos(angle_el);
  Ua = Ualpha + voltage_power_supply / 2;
  Ub = (sqrt(3) * Ubeta - Ualpha) / 2 + voltage_power_supply / 2;
  Uc = (-Ualpha - sqrt(3) * Ubeta) / 2 + voltage_power_supply / 2;
  setPwm(Ua, Ub, Uc);
}

float velocityOpenloop(float target_velocity)
{
  uint32_t now_us = HAL_GetTick() * 1000;
  float Ts = (now_us - open_loop_timestamp) * 1e-6f;
  if (Ts <= 0 || Ts > 0.5f) Ts = 1e-3f;
  shaft_angle = _normalizeAngle(shaft_angle + target_velocity * Ts);
  float Uq = voltage_power_supply / 3;
  setPhaseVoltage(Uq, 0, _electricalAngle(shaft_angle, 7));
  open_loop_timestamp = now_us;
  return Uq;
}
/* USER CODE END 0 */
/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */
  /* USER CODE END 1 */
  /* MCU Configuration--------------------------------------------------------*/
  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();
  /* USER CODE BEGIN Init */
  /* USER CODE END Init */
  /* Configure the system clock */
  SystemClock_Config();
  /* USER CODE BEGIN SysInit */
  /* USER CODE END SysInit */
  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_I2C2_Init();
  MX_USART1_UART_Init();
  MX_TIM2_Init();
  /* USER CODE BEGIN 2 */
  HAL_GPIO_WritePin(GPIOB,GPIO_PIN_6,GPIO_PIN_RESET);
  HAL_GPIO_WritePin(GPIOB,GPIO_PIN_7,GPIO_PIN_SET);
  OLED_Init();
  /* USER CODE END 2 */
  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
```

```
    /* USER CODE END WHILE */
    /* USER CODE BEGIN 3 */
    uint16_t Angle = 0;
    uint8_t IIC_Buffer[2] = {0};
    HAL_I2C_Mem_Read(&hi2c2,0x6c,0x0c,I2C_MEMADD_SIZE_8BIT,IIC_Buffer,2,50);

    Angle = IIC_Buffer[0] << 8;
    Angle = Angle | IIC_Buffer[1];
    Angle = Angle*0.08789;
    velocityOpenloop(-1);

    OLED_ShowString(0,0,"I Love Just!",OLED_8X16);
    OLED_ShowNum(0,20,Angle,3,OLED_8X16);
    OLED_Update();
  }
  /* USER CODE END 3 */
}
/**
  * @brief System Clock Configuration
  * @retval None
  */
void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
  /** Initializes the RCC Oscillators according to the specified parameters
  * in the RCC_OscInitTypeDef structure.
  */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSE;
  RCC_OscInitStruct.HSEState = RCC_HSE_ON;
  RCC_OscInitStruct.HSEPredivValue = RCC_HSE_PREDIV_DIV1;
  RCC_OscInitStruct.HSIState = RCC_HSI_ON;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
  RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSE;
  RCC_OscInitStruct.PLL.PLLMUL = RCC_PLL_MUL9;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }
  /** Initializes the CPU, AHB and APB buses clocks
  */
  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                              |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV2;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;
  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_2) != HAL_OK)
  {
    Error_Handler();
  }
}
/* USER CODE BEGIN 4 */
/* USER CODE END 4 */
/**
  * @brief  This function is executed in case of error occurrence.
  * @retval None
  */
void Error_Handler(void)
{
  /* USER CODE BEGIN Error_Handler_Debug */
  /* User can add his own implementation to report the HAL error return state */
  __disable_irq();
  while (1)
  {
  }
```

```
  /* USER CODE END Error_Handler_Debug */
}
#ifdef  USE_FULL_ASSERT
/**
  * @brief  Reports the name of the source file and the source line number
  *         where the assert_param error has occurred.
  * @param  file: pointer to the source file name
  * @param  line: assert_param error line source number
  * @retval None
  */
void assert_failed(uint8_t *file, uint32_t line)
{
  /* USER CODE BEGIN 6 */
  /* User can add his own implementation to report the file name and line number,
     ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line) */
  /* USER CODE END 6 */
}
#endif /* USE_FULL_ASSERT */
```

simpleFocMini

2025年3月6日 17:12

UART串口

霍尔磁编码器

电脑

简单的FOCMiniV1.0 版本 简单的FOCMiniV1.1 版

MCU核心板

simpleFoc驱动板

三相无刷电机

PD诱骗电路板to12V

OLED显示屏