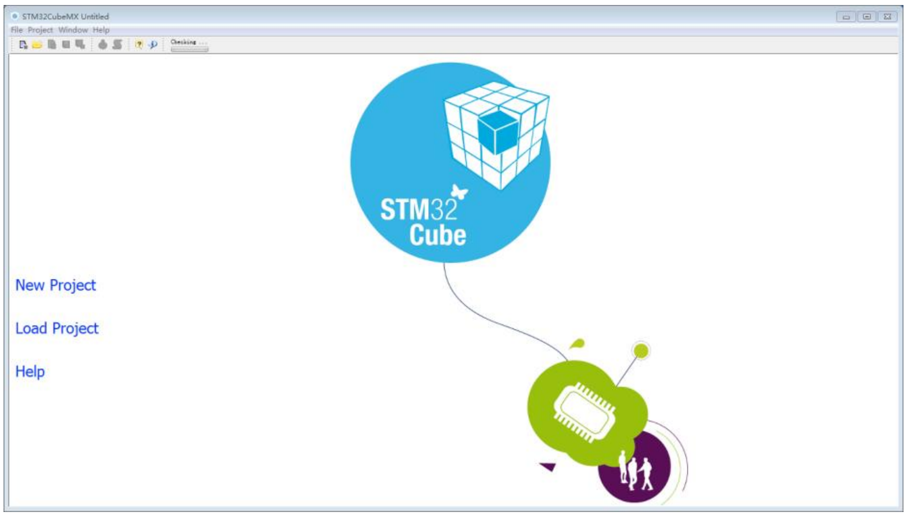
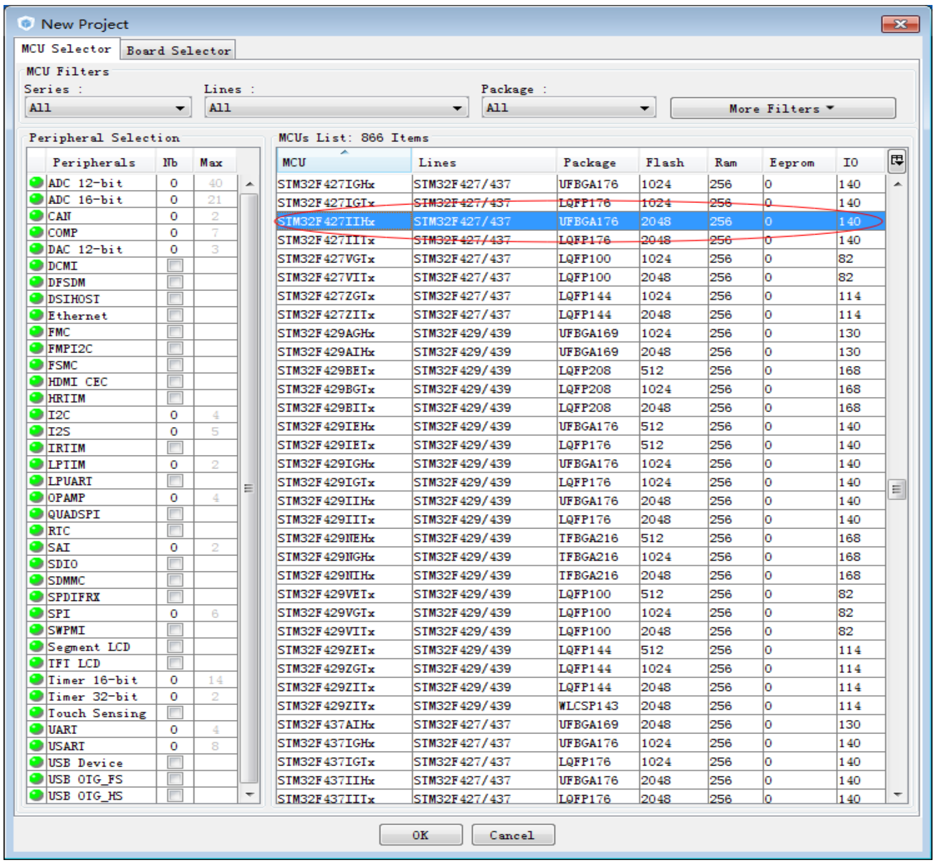
Cube 生成 freertos 工程

1.新建 project



2.芯片选型：

找到与单片机对应的芯片，双击打开



MCU Filters

Series : All

Lines : All

Package : All

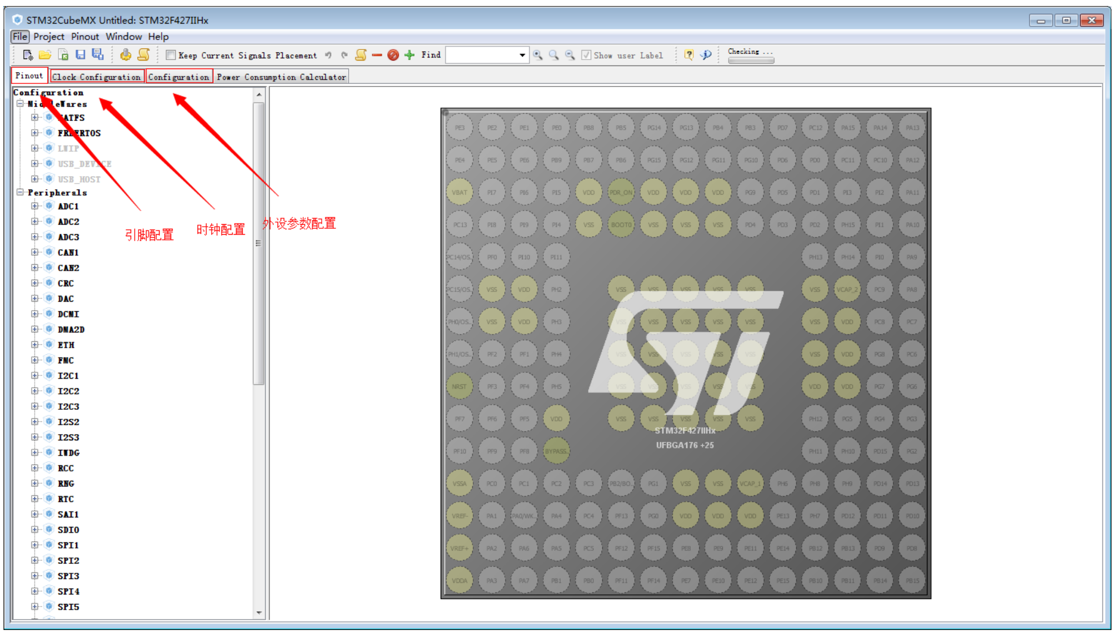
More Filters

Peripheral Selection

	Peripherals	Nb	Max
🟢	ADC 12-bit	0	40
🟢	ADC 16-bit	0	21
🟢	CAN	0	2
🟢	COMP	0	7
🟢	DAC 12-bit	0	3
🟢	DCMI	☐	
🟢	DFSDM	☐	
🟢	DSIHOST	☐	
🟢	Ethernet	☐	
🟢	FMC	☐	
🟢	FMPI2C	☐	
🟢	FSMC	☐	
🟢	HDMI CEC	☐	
🟢	HRTIM	☐	
🟢	I2C	0	4
🟢	I2S	0	5
🟢	IRTIM	☐	
🟢	LPTIM	0	2
🟢	LPUART	☐	
🟢	OPAMP	0	4
🟢	QUADSPI	☐	
🟢	RTC	☐	
🟢	SAI	0	2
🟢	SDIO	☐	
🟢	SDMMC	☐	
🟢	SPDIFRX	☐	
🟢	SPI	0	6
🟢	SWPMI	☐	
🟢	Segment LCD	☐	
🟢	TFT LCD	☐	
🟢	Timer 16-bit	0	14
🟢	Timer 32-bit	0	2
🟢	Touch Sensing	☐	
🟢	UART	0	4
🟢	USART	0	8
🟢	USB Device	☐	
🟢	USB OTG_FS	☐	
🟢	USB OTG_HS	☐	

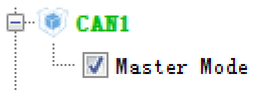
MCUs List: 866 Items

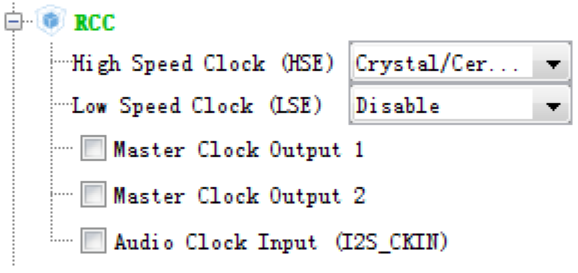
MCU	Lines	Package	Flash	Ram	Eeprom	IO
STM32F427IGHx	STM32F427/437	UFBGA176	1024	256	0	140
STM32F427IGTx	STM32F427/437	LQFP176	1024	256	0	140
STM32F427IIHx	STM32F427/437	UFBGA176	2048	256	0	140
STM32F427IITx	STM32F427/437	LQFP176	2048	256	0	140
STM32F427VGTx	STM32F427/437	LQFP100	1024	256	0	82
STM32F427VITx	STM32F427/437	LQFP100	2048	256	0	82
STM32F427ZGTx	STM32F427/437	LQFP144	1024	256	0	114
STM32F427ZITx	STM32F427/437	LQFP144	2048	256	0	114
STM32F429AGHx	STM32F429/439	UFBGA169	1024	256	0	130
STM32F429AIHx	STM32F429/439	UFBGA169	2048	256	0	130
STM32F429BETx	STM32F429/439	LQFP208	512	256	0	168
STM32F429BGTx	STM32F429/439	LQFP208	1024	256	0	168
STM32F429BITx	STM32F429/439	LQFP208	2048	256	0	168
STM32F429IEHx	STM32F429/439	UFBGA176	512	256	0	140
STM32F429IETx	STM32F429/439	LQFP176	512	256	0	140
STM32F429IGHx	STM32F429/439	UFBGA176	1024	256	0	140
STM32F429IGTx	STM32F429/439	LQFP176	1024	256	0	140
STM32F429IIHx	STM32F429/439	UFBGA176	2048	256	0	140
STM32F429IITx	STM32F429/439	LQFP176	2048	256	0	140
STM32F429NEHx	STM32F429/439	TFBGA216	512	256	0	168
STM32F429NGHx	STM32F429/439	TFBGA216	1024	256	0	168
STM32F429NIHx	STM32F429/439	TFBGA216	2048	256	0	168
STM32F429VETx	STM32F429/439	LQFP100	512	256	0	82
STM32F429VGTx	STM32F429/439	LQFP100	1024	256	0	82
STM32F429VITx	STM32F429/439	LQFP100	2048	256	0	82
STM32F429ZETx	STM32F429/439	LQFP144	512	256	0	114
STM32F429ZGTx	STM32F429/439	LQFP144	1024	256	0	114
STM32F429ZITx	STM32F429/439	LQFP144	2048	256	0	114
STM32F429ZIYx	STM32F429/439	WLCSP143	2048	256	0	114
STM32F437AIHx	STM32F427/437	UFBGA169	2048	256	0	130
STM32F437IGHx	STM32F427/437	UFBGA176	1024	256	0	140
STM32F437IGTx	STM32F427/437	LQFP176	1024	256	0	140
STM32F437IIHx	STM32F427/437	UFBGA176	2048	256	0	140
STM32F437IITx	STM32F427/437	LQFP176	2048	256	0	140

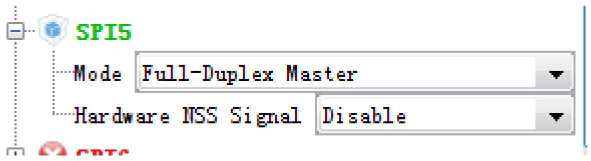


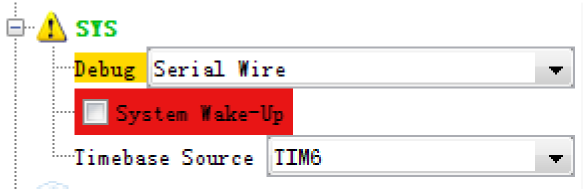
3.一些常见模块的配置

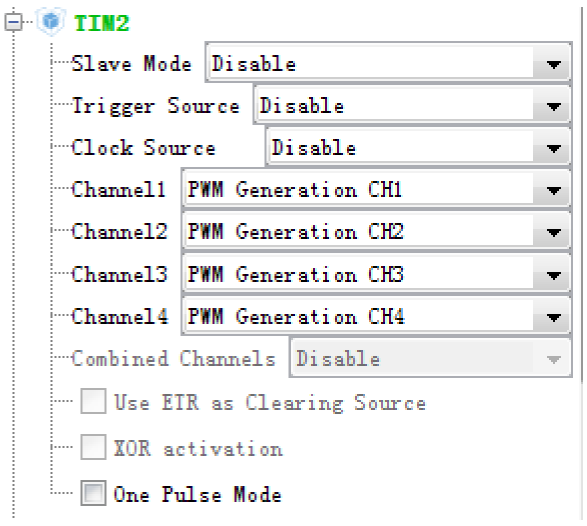
（1）引脚配置


CAN1
- Master Mode ☑

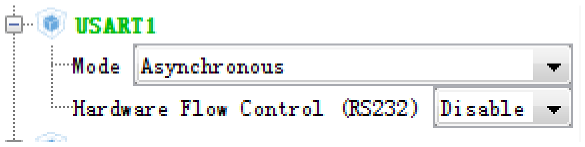

RCC
- High Speed Clock (HSE): Crystal/Cer...
- Low Speed Clock (LSE): Disable
- Master Clock Output 1 ☐
- Master Clock Output 2 ☐
- Audio Clock Input (I2S_CKIN) ☐


SPI5
- Mode: Full-Duplex Master
- Hardware NSS Signal: Disable

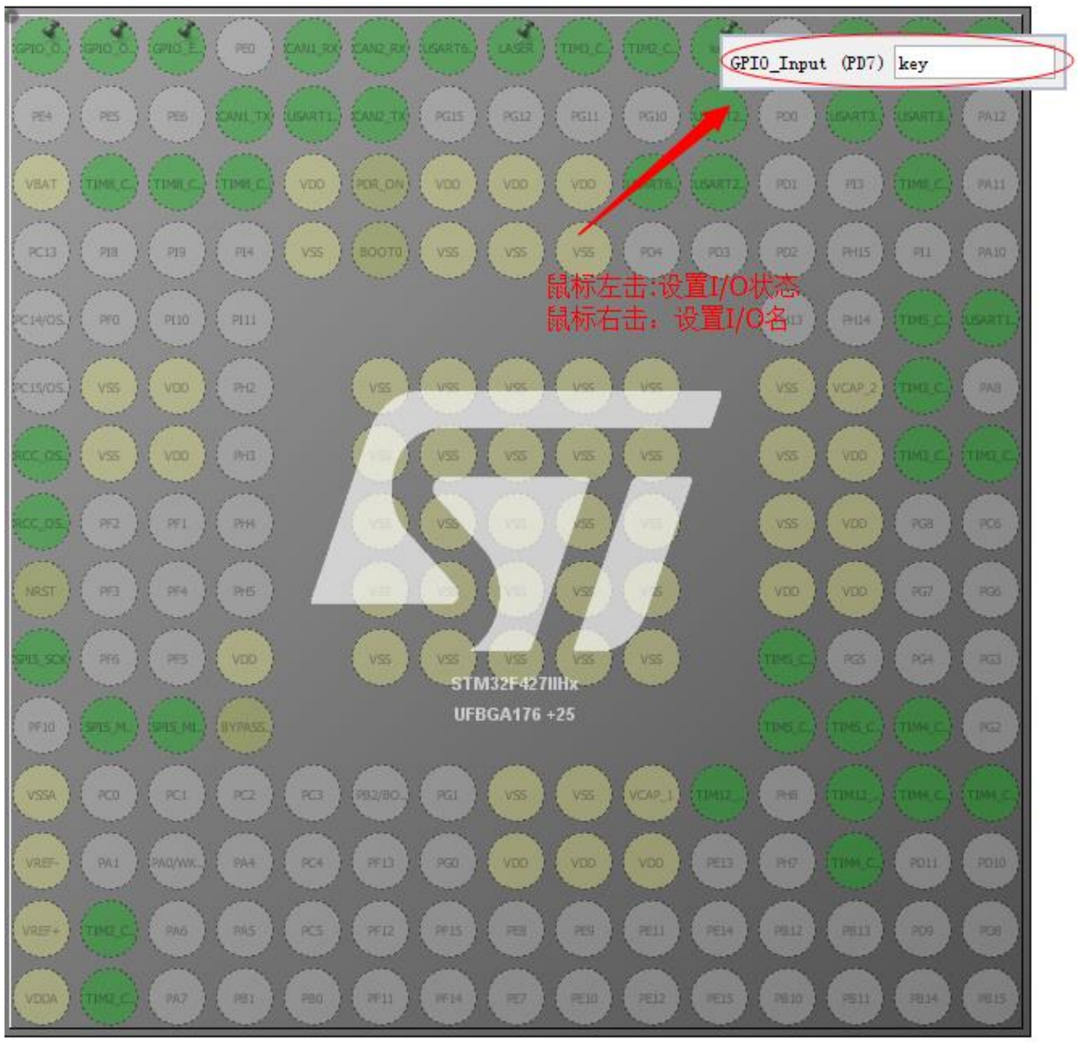

SYS
- Debug: Serial Wire
- System Wake-Up ☐
- Timebase Source: TIM6



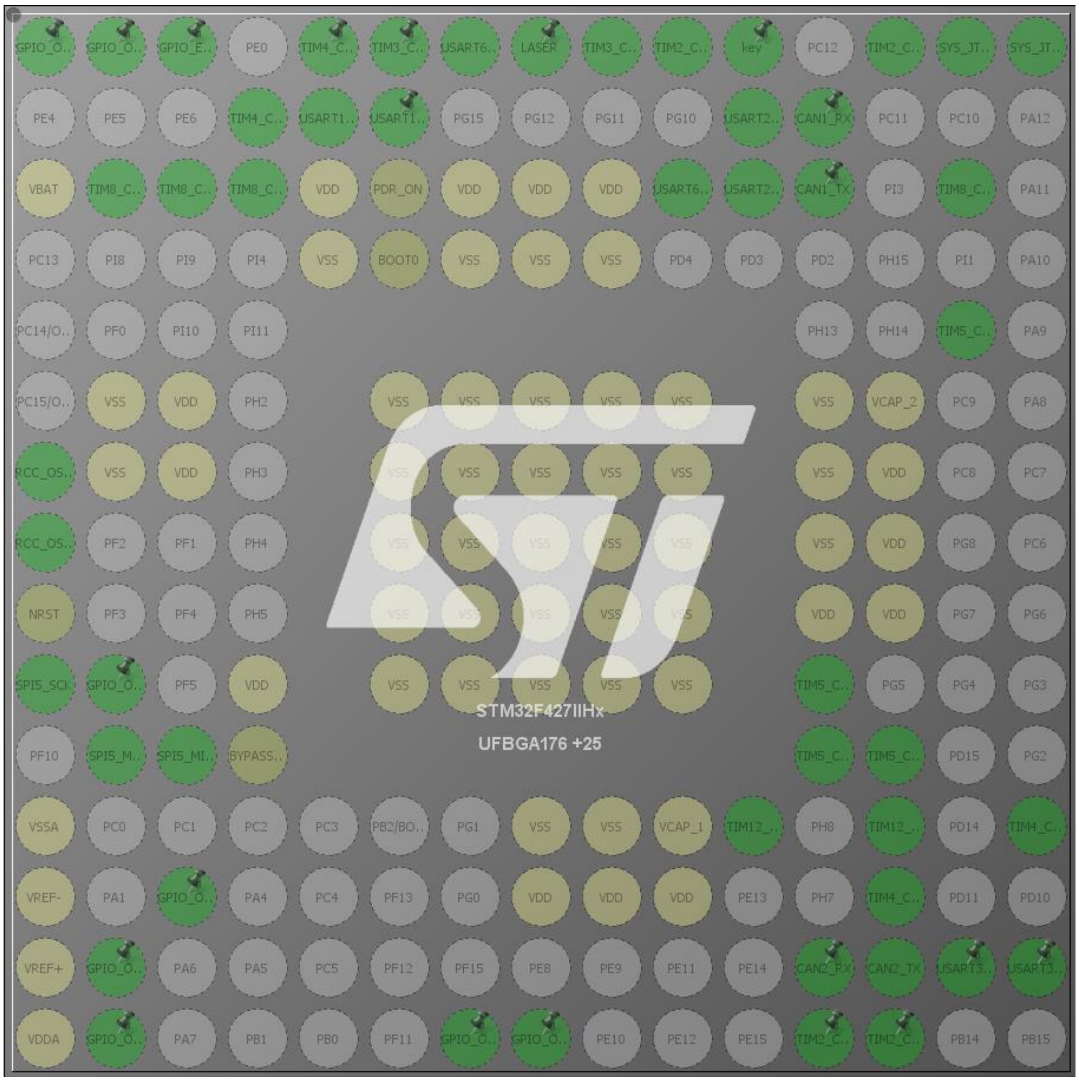
TIM2
- Slave Mode: Disable
- Trigger Source: Disable
- Clock Source: Disable
- Channel1: PWM Generation CH1
- Channel2: PWM Generation CH2
- Channel3: PWM Generation CH3
- Channel4: PWM Generation CH4
- Combined Channels: Disable
- [ ] Use ETR as Clearing Source
- [ ] XOR activation
- [ ] One Pulse Mode



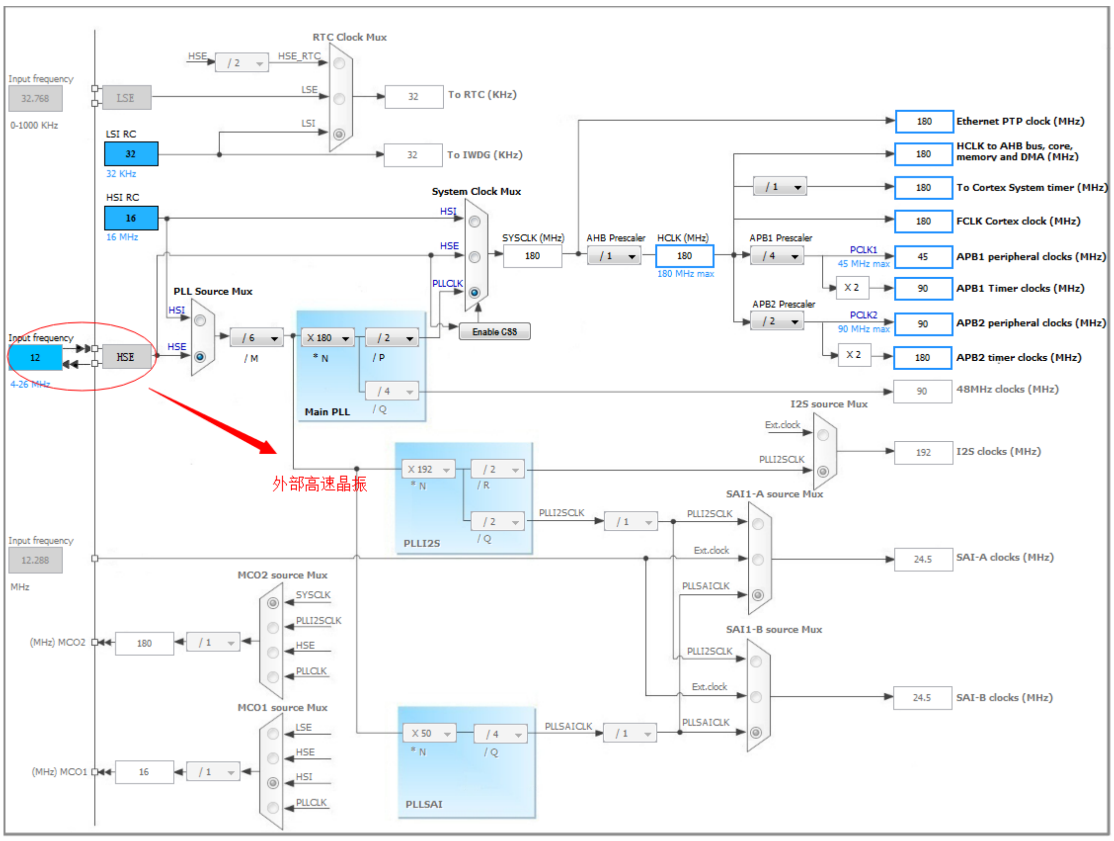
USART1
- Mode: Asynchronous
- Hardware Flow Control (RS232): Disable



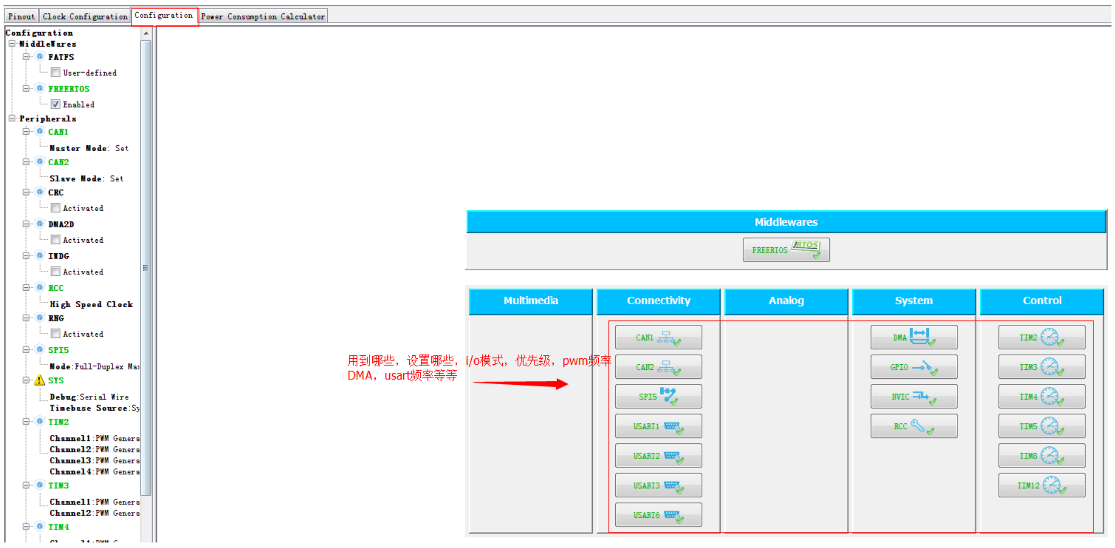
最后的所有配置：



（2）时钟配置

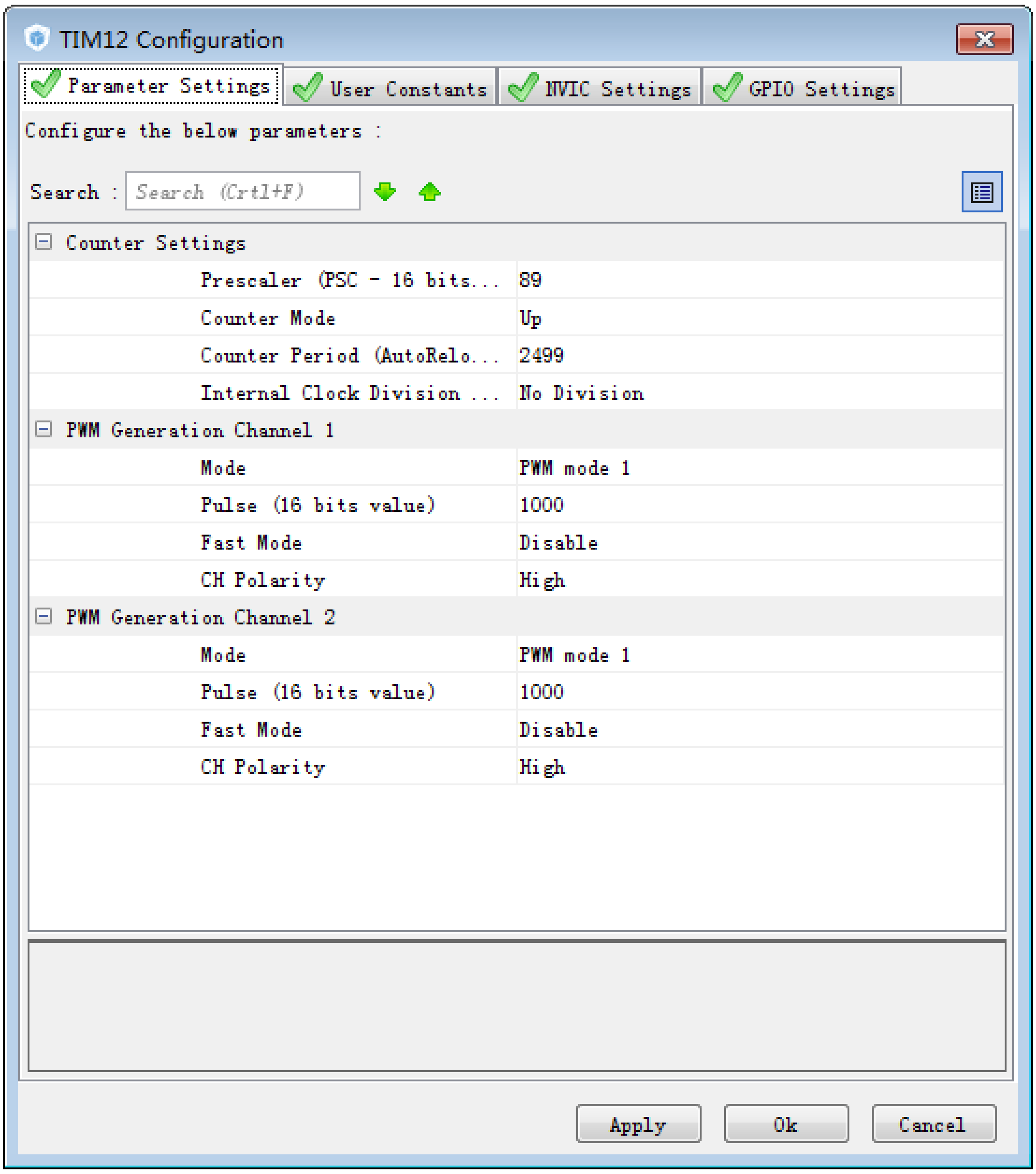


（3）外设参数配置


Pinout | Clock Configuration | Configuration | Power Consumption Calculator

```
Configuration
⊟ MiddleWares
  ⊟ FATFS
     ☐ User-defined
  ⊟ FREERTOS
     ☑ Enabled
⊟ Peripherals
  ⊟ CAN1
     Master Mode: Set
  ⊟ CAN2
     Slave Mode: Set
  ⊟ CRC
     ☐ Activated
  ⊟ DMA2D
     ☐ Activated
  ⊟ IWDG
     ☐ Activated
  ⊟ RCC
     High Speed Clock
  ⊟ RNG
     ☐ Activated
  ⊟ SPI5
     Mode:Full-Duplex Ma
  ⊟ SYS
     Debug:Serial Wire
     Timebase Source:Sy
  ⊟ TIM2
     Channel1:PWM Genera
     Channel2:PWM Genera
     Channel3:PWM Genera
     Channel4:PWM Genera
  ⊟ TIM3
     Channel1:PWM Genera
     Channel2:PWM Genera
  ⊟ TIM4
```

用到哪些，设置哪些，i/o模式，优先级，pwm频率
DMA，usart频率等等

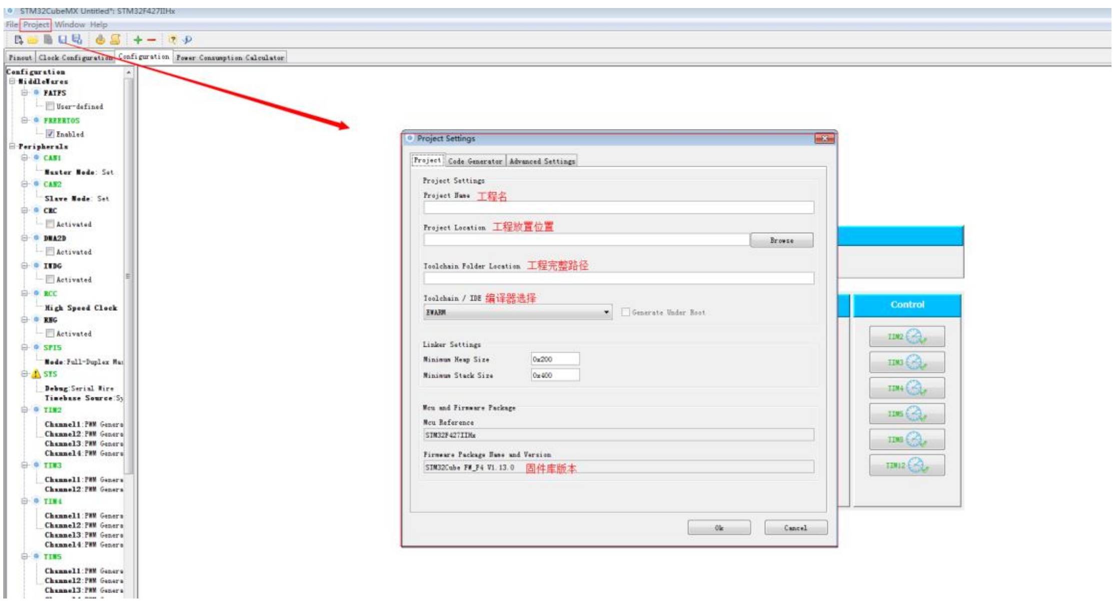


Configure the below parameters :

Search :

Counter Settings	
Prescaler (PSC - 16 bits...	89
Counter Mode	Up
Counter Period (AutoRelo...	2499
Internal Clock Division ...	No Division
PWM Generation Channel 1	
Mode	PWM mode 1
Pulse (16 bits value)	1000
Fast Mode	Disable
CH Polarity	High
PWM Generation Channel 2	
Mode	PWM mode 1
Pulse (16 bits value)	1000
Fast Mode	Disable
CH Polarity	High

4.生成 Keil 工程



Project Settings

Project | Code Generator | Advanced Settings

Project Settings

Project Name 工程名

Project Location 工程放置位置

Toolchain Folder Location 工程完整路径

Toolchain / IDE 编译器选择: EWARM

☐ Generate Under Root

Linker Settings

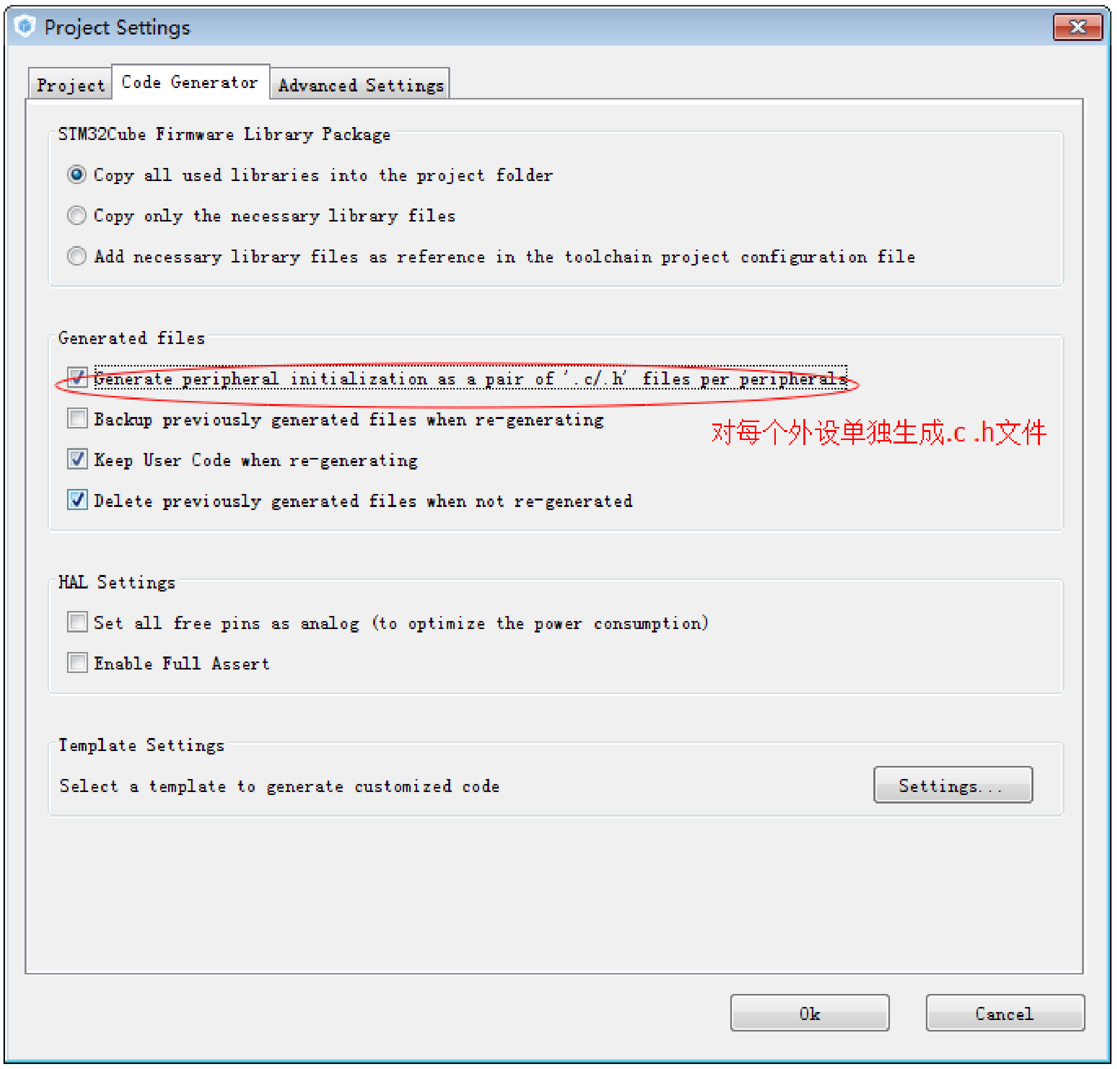
Minimum Heap Size: 0x200

Minimum Stack Size: 0x400

Mcu and Firmware Package

Mcu Reference: STM32F427IIHx

Firmware Package Name and Version: STM32Cube FW_F4 V1.13.0 固件库版本



Project Settings

Project | Code Generator | Advanced Settings

STM32Cube Firmware Library Package

● Copy all used libraries into the project folder

○ Copy only the necessary library files

○ Add necessary library files as reference in the toolchain project configuration file

Generated files

☑ Generate peripheral initialization as a pair of '.c/.h' files per peripherals

☐ Backup previously generated files when re-generating

☑ Keep User Code when re-generating

☑ Delete previously generated files when not re-generated

对每个外设单独生成.c .h文件

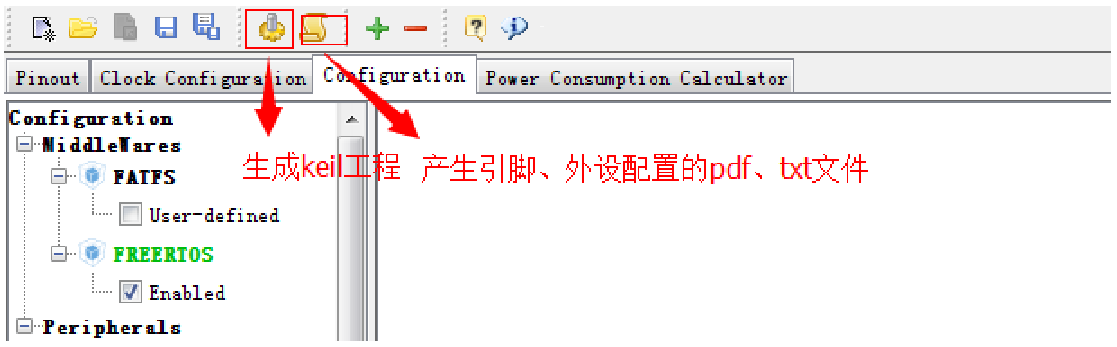
HAL Settings

☐ Set all free pins as analog (to optimize the power consumption)

☐ Enable Full Assert

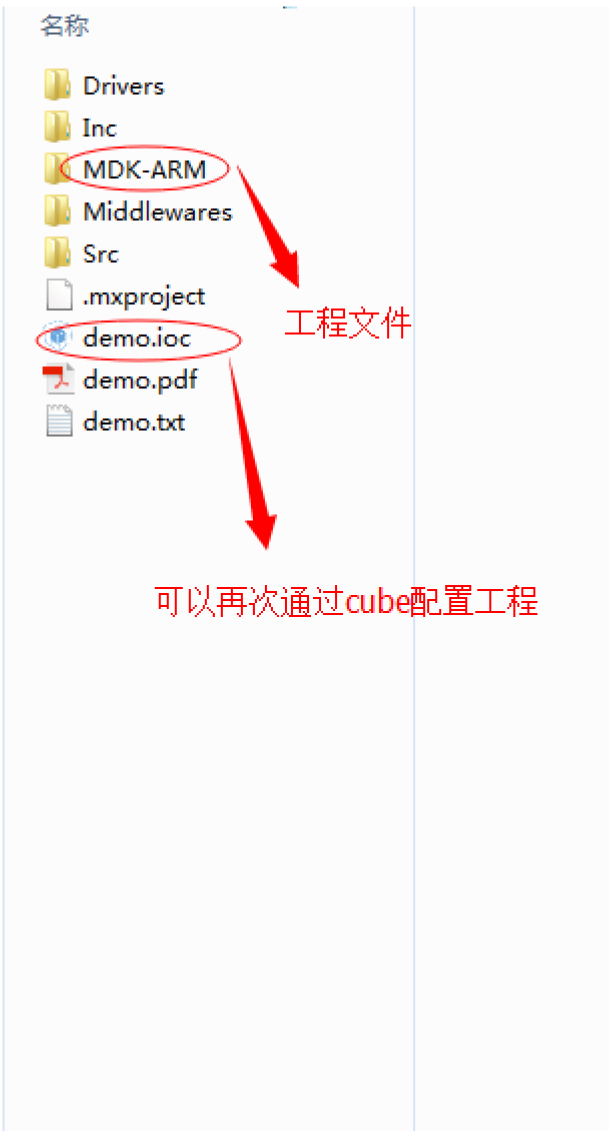
Template Settings

Select a template to generate customized code

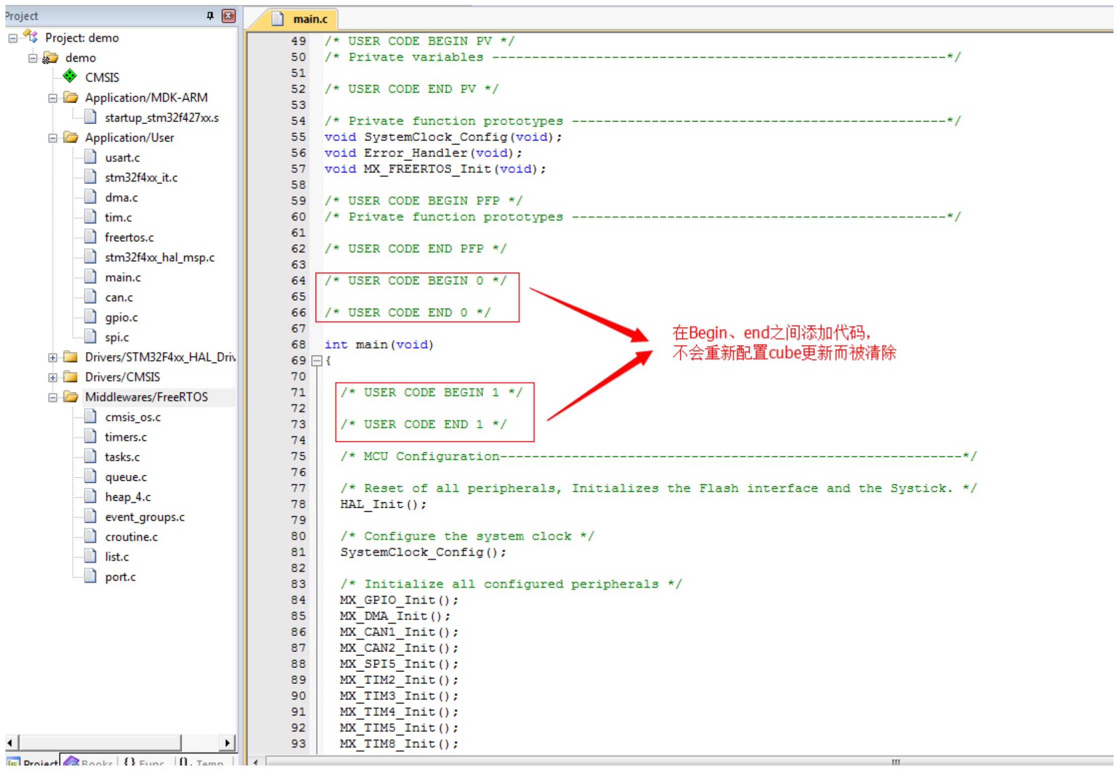


5. 工程文件：

（1）生成的工程：



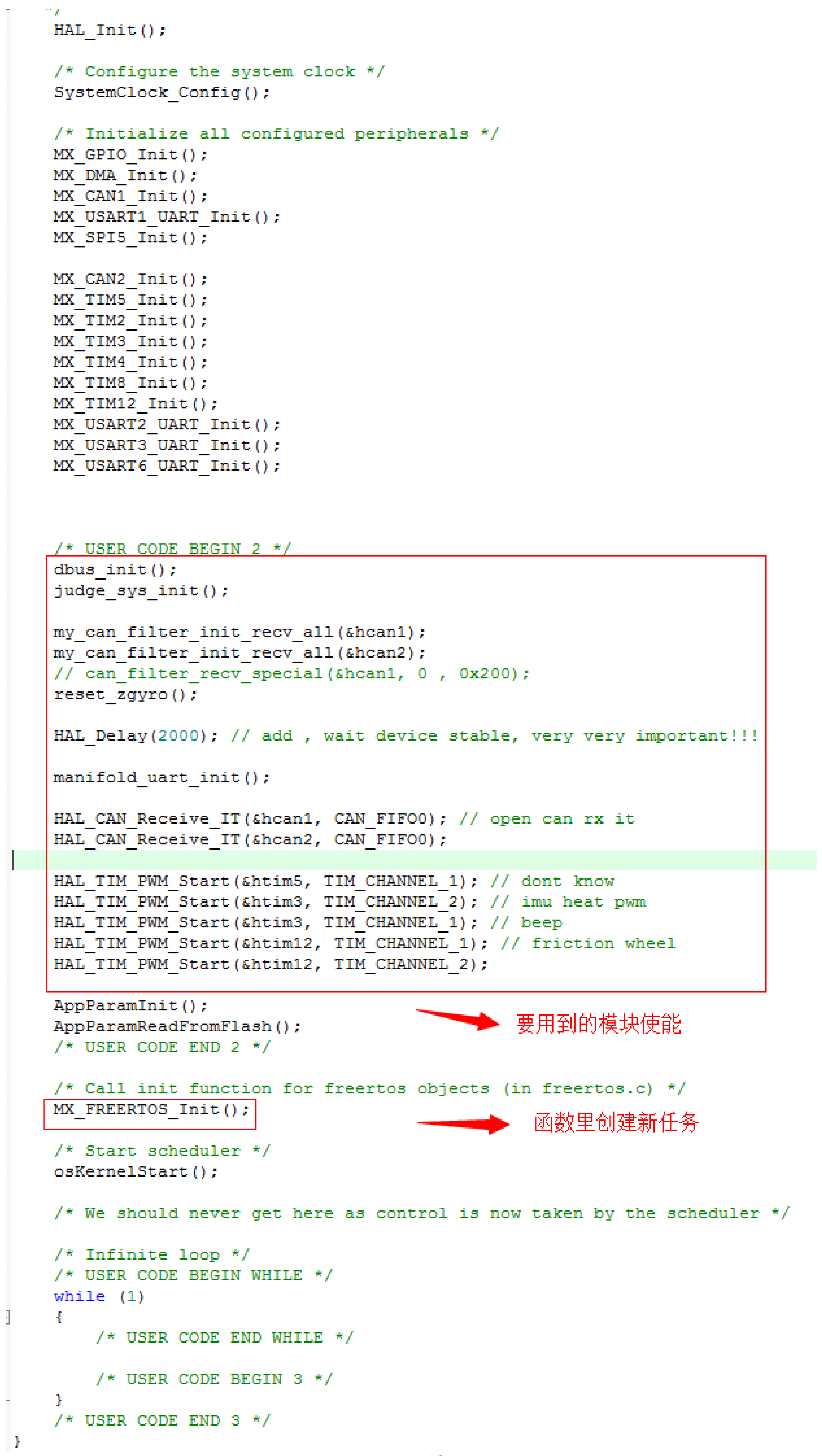
（2）代码编辑区



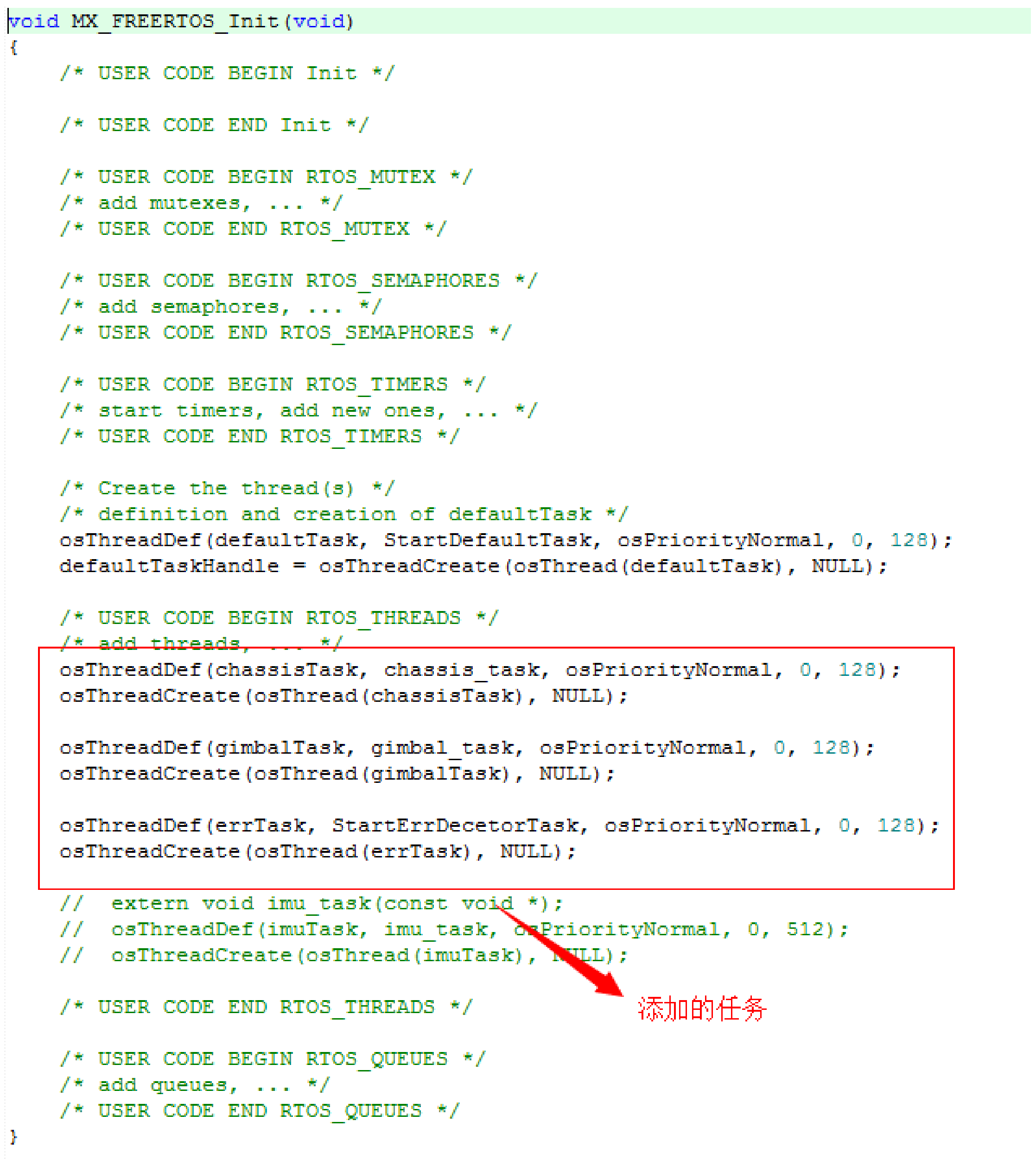
```c
/* USER CODE BEGIN PV */
/* Private variables ---------------------------------------------------------*/

/* USER CODE END PV */

/* Private function prototypes -----------------------------------------------*/
void SystemClock_Config(void);
void Error_Handler(void);
void MX_FREERTOS_Init(void);

/* USER CODE BEGIN PFP */
/* Private function prototypes -----------------------------------------------*/

/* USER CODE END PFP */

/* USER CODE BEGIN 0 */

/* USER CODE END 0 */

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();

  /* Configure the system clock */
  SystemClock_Config();

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_DMA_Init();
  MX_CAN1_Init();
  MX_CAN2_Init();
  MX_SPI5_Init();
  MX_TIM2_Init();
  MX_TIM3_Init();
  MX_TIM4_Init();
  MX_TIM5_Init();
  MX_TIM8_Init();
```

在Begin、end之间添加代码，
不会重新配置cube更新而被清除

（3）快速入门

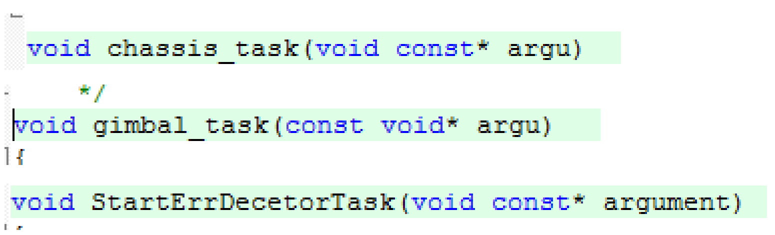


```
  */
  HAL_Init();

  /* Configure the system clock */
  SystemClock_Config();

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_DMA_Init();
  MX_CAN1_Init();
  MX_USART1_UART_Init();
  MX_SPI5_Init();

  MX_CAN2_Init();
  MX_TIM5_Init();
  MX_TIM2_Init();
  MX_TIM3_Init();
  MX_TIM4_Init();
  MX_TIM8_Init();
  MX_TIM12_Init();
  MX_USART2_UART_Init();
  MX_USART3_UART_Init();
  MX_USART6_UART_Init();



  /* USER CODE BEGIN 2 */
  dbus_init();
  judge_sys_init();

  my_can_filter_init_recv_all(&hcan1);
  my_can_filter_init_recv_all(&hcan2);
  // can_filter_recv_special(&hcan1, 0 , 0x200);
  reset_zgyro();

  HAL_Delay(2000); // add , wait device stable, very very important!!!

  manifold_uart_init();

  HAL_CAN_Receive_IT(&hcan1, CAN_FIFO0); // open can rx it
  HAL_CAN_Receive_IT(&hcan2, CAN_FIFO0);

  HAL_TIM_PWM_Start(&htim5, TIM_CHANNEL_1); // dont know
  HAL_TIM_PWM_Start(&htim3, TIM_CHANNEL_2); // imu heat pwm
  HAL_TIM_PWM_Start(&htim3, TIM_CHANNEL_1); // beep
  HAL_TIM_PWM_Start(&htim12, TIM_CHANNEL_1); // friction wheel
  HAL_TIM_PWM_Start(&htim12, TIM_CHANNEL_2);

  AppParamInit();
  AppParamReadFromFlash();
  /* USER CODE END 2 */

  /* Call init function for freertos objects (in freertos.c) */
  MX_FREERTOS_Init();

  /* Start scheduler */
  osKernelStart();

  /* We should never get here as control is now taken by the scheduler */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
      /* USER CODE END WHILE */

      /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}
```



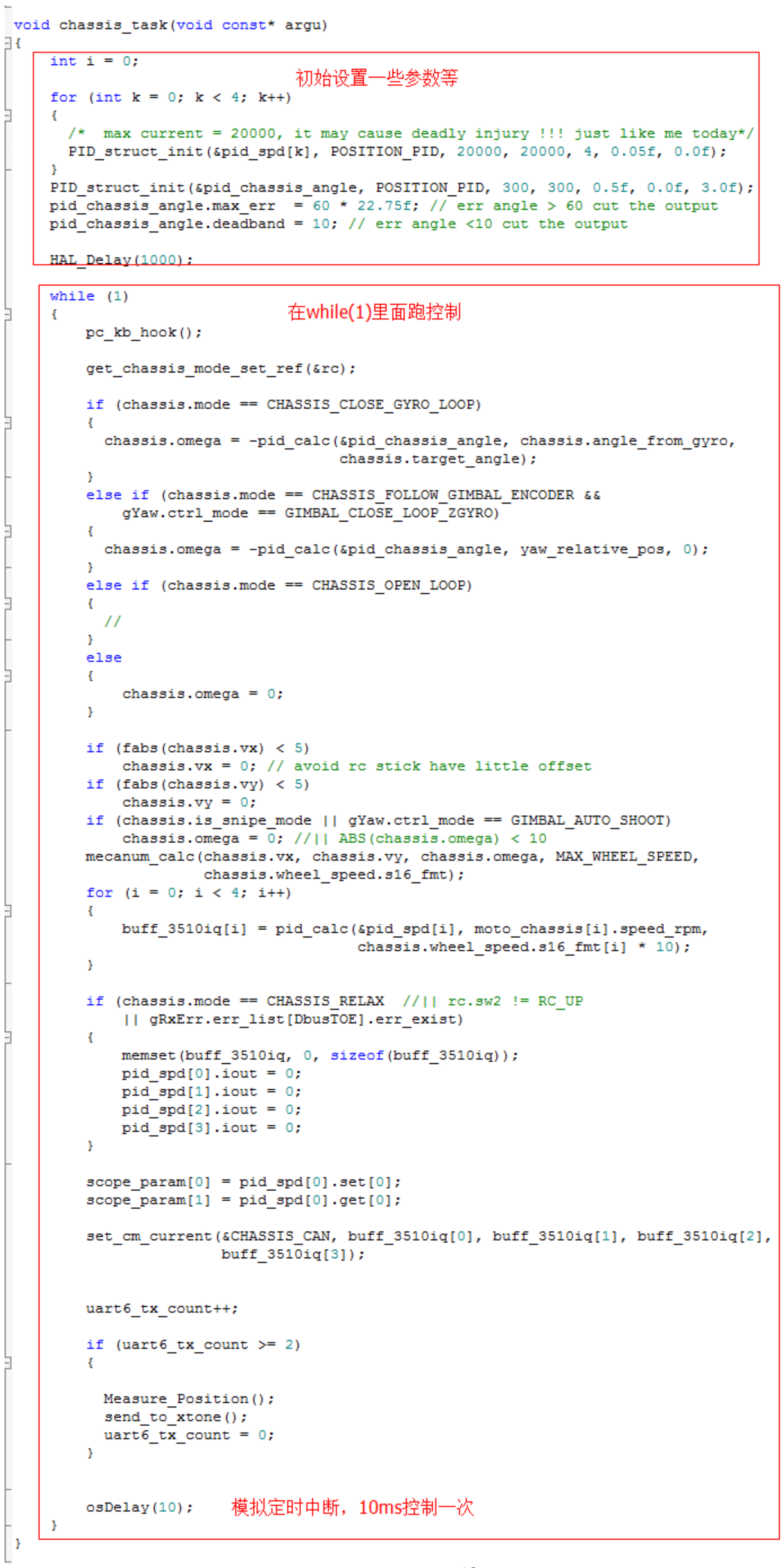
```
void MX_FREERTOS_Init(void)
{
    /* USER CODE BEGIN Init */

    /* USER CODE END Init */

    /* USER CODE BEGIN RTOS_MUTEX */
    /* add mutexes, ... */
    /* USER CODE END RTOS_MUTEX */

    /* USER CODE BEGIN RTOS_SEMAPHORES */
    /* add semaphores, ... */
    /* USER CODE END RTOS_SEMAPHORES */

    /* USER CODE BEGIN RTOS_TIMERS */
    /* start timers, add new ones, ... */
    /* USER CODE END RTOS_TIMERS */

    /* Create the thread(s) */
    /* definition and creation of defaultTask */
    osThreadDef(defaultTask, StartDefaultTask, osPriorityNormal, 0, 128);
    defaultTaskHandle = osThreadCreate(osThread(defaultTask), NULL);

    /* USER CODE BEGIN RTOS_THREADS */
    /* add threads, ... */
    osThreadDef(chassisTask, chassis_task, osPriorityNormal, 0, 128);
    osThreadCreate(osThread(chassisTask), NULL);

    osThreadDef(gimbalTask, gimbal_task, osPriorityNormal, 0, 128);
    osThreadCreate(osThread(gimbalTask), NULL);

    osThreadDef(errTask, StartErrDecetorTask, osPriorityNormal, 0, 128);
    osThreadCreate(osThread(errTask), NULL);

    //  extern void imu_task(const void *);
    //  osThreadDef(imuTask, imu_task, osPriorityNormal, 0, 512);
    //  osThreadCreate(osThread(imuTask), NULL);

    /* USER CODE END RTOS_THREADS */

    /* USER CODE BEGIN RTOS_QUEUES */
    /* add queues, ... */
    /* USER CODE END RTOS_QUEUES */
}
```

创建的三个任务函数



```
void chassis_task(void const* argu)
    */
void gimbal_task(const void* argu)
{
void StartErrDecetorTask(void const* argument)
```

任务函数结构：



```
void chassis_task(void const* argu)
{
    int i = 0;
                            初始设置一些参数等
    for (int k = 0; k < 4; k++)
    {
      /*  max current = 20000, it may cause deadly injury !!! just like me today*/
      PID_struct_init(&pid_spd[k], POSITION_PID, 20000, 20000, 4, 0.05f, 0.0f);
    }
    PID_struct_init(&pid_chassis_angle, POSITION_PID, 300, 300, 0.5f, 0.0f, 3.0f);
    pid_chassis_angle.max_err  = 60 * 22.75f; // err angle > 60 cut the output
    pid_chassis_angle.deadband = 10; // err angle <10 cut the output

    HAL_Delay(1000);

    while (1)
    {                           在while(1)里面跑控制
        pc_kb_hook();

        get_chassis_mode_set_ref(&rc);

        if (chassis.mode == CHASSIS_CLOSE_GYRO_LOOP)
        {
          chassis.omega = -pid_calc(&pid_chassis_angle, chassis.angle_from_gyro,
                                     chassis.target_angle);
        }
        else if (chassis.mode == CHASSIS_FOLLOW_GIMBAL_ENCODER &&
            gYaw.ctrl_mode == GIMBAL_CLOSE_LOOP_ZGYRO)
        {
          chassis.omega = -pid_calc(&pid_chassis_angle, yaw_relative_pos, 0);
        }
        else if (chassis.mode == CHASSIS_OPEN_LOOP)
        {
          //
        }
        else
        {
            chassis.omega = 0;
        }

        if (fabs(chassis.vx) < 5)
            chassis.vx = 0; // avoid rc stick have little offset
        if (fabs(chassis.vy) < 5)
            chassis.vy = 0;
        if (chassis.is_snipe_mode || gYaw.ctrl_mode == GIMBAL_AUTO_SHOOT)
            chassis.omega = 0; //|| ABS(chassis.omega) < 10
        mecanum_calc(chassis.vx, chassis.vy, chassis.omega, MAX_WHEEL_SPEED,
                     chassis.wheel_speed.s16_fmt);
        for (i = 0; i < 4; i++)
        {
            buff_3510iq[i] = pid_calc(&pid_spd[i], moto_chassis[i].speed_rpm,
                                       chassis.wheel_speed.s16_fmt[i] * 10);
        }

        if (chassis.mode == CHASSIS_RELAX  //|| rc.sw2 != RC_UP
            || gRxErr.err_list[DbusTOE].err_exist)
        {
            memset(buff_3510iq, 0, sizeof(buff_3510iq));
            pid_spd[0].iout = 0;
            pid_spd[1].iout = 0;
            pid_spd[2].iout = 0;
            pid_spd[3].iout = 0;
        }

        scope_param[0] = pid_spd[0].set[0];
        scope_param[1] = pid_spd[0].get[0];

        set_cm_current(&CHASSIS_CAN, buff_3510iq[0], buff_3510iq[1], buff_3510iq[2],
                       buff_3510iq[3]);


        uart6_tx_count++;

        if (uart6_tx_count >= 2)
        {

          Measure_Position();
          send_to_xtone();
          uart6_tx_count = 0;
        }


        osDelay(10);    模拟定时中断，10ms控制一次
    }
}
```