FTM8ForgeIDE

Troubleshooting Guide

Fortior Technology (Shenzhen) Co., Ltd.

Contents

1 Statement Sequence ······ **3**

2 Structure Members ······ **4**

3 Array Members in Structure ······ **5**

4 Enumeration Members ······ **6**

5 Project Porting ······ **7**

5.1 Timing Problem 1 ······ 7

5.2 Timing Problem 2 ······ 9

6 Stack Overflow ······ **12**

7 Flash Programming ······ **13**

8 Code Size Reduction ······ **14**

8.1 Redundant Types and Variable Declarations ······ 14

8.2 Enumeration Type ······ 14

8.3 Function Declaration ······ 15

8.4 Statement Selection ······ 16

8.5 Unions ······ 16

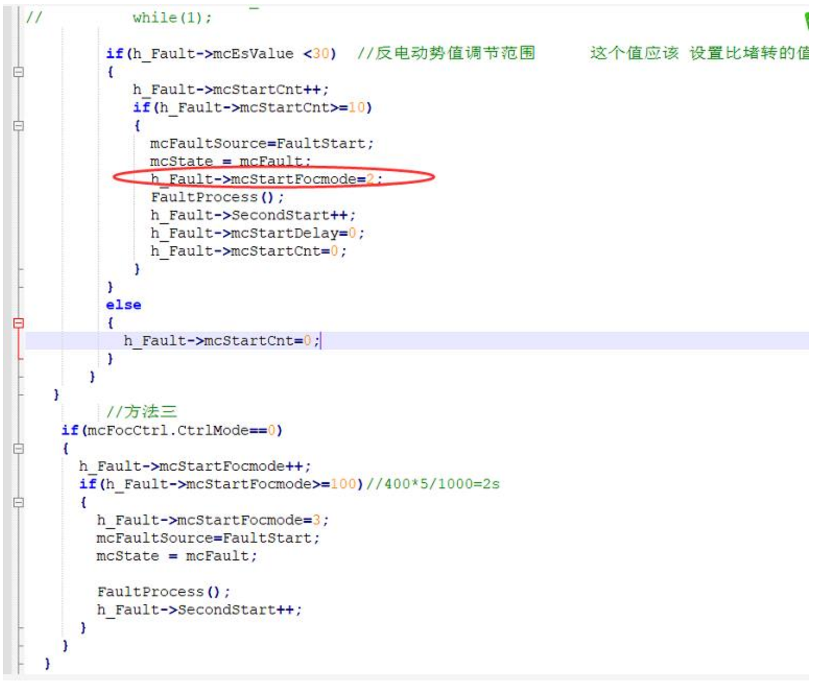
8.6 Logical Operators ······ 16

8.7 Variable Storage Area ······ 17

9 Revision History ······ **18**

1 Statement Sequence

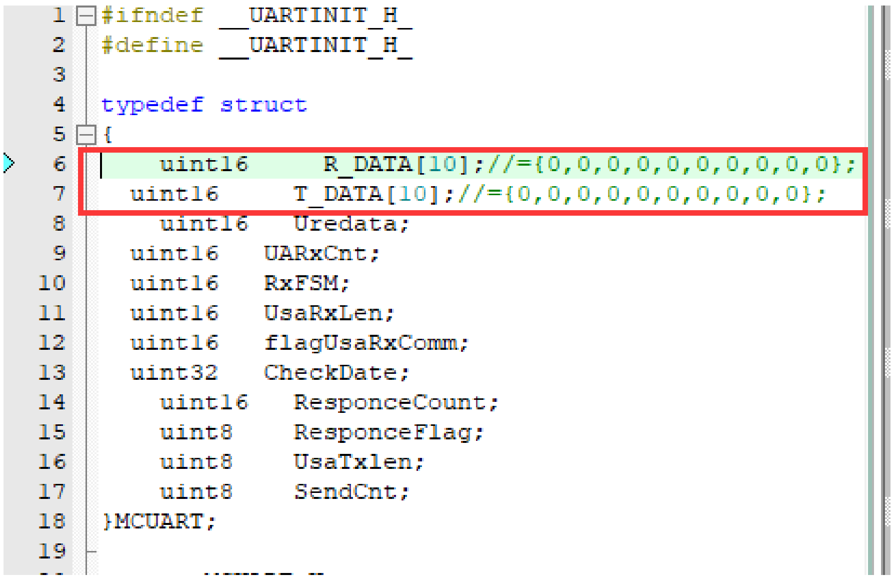
Adjust sequence of the statements. FTCC brings the same elements together.



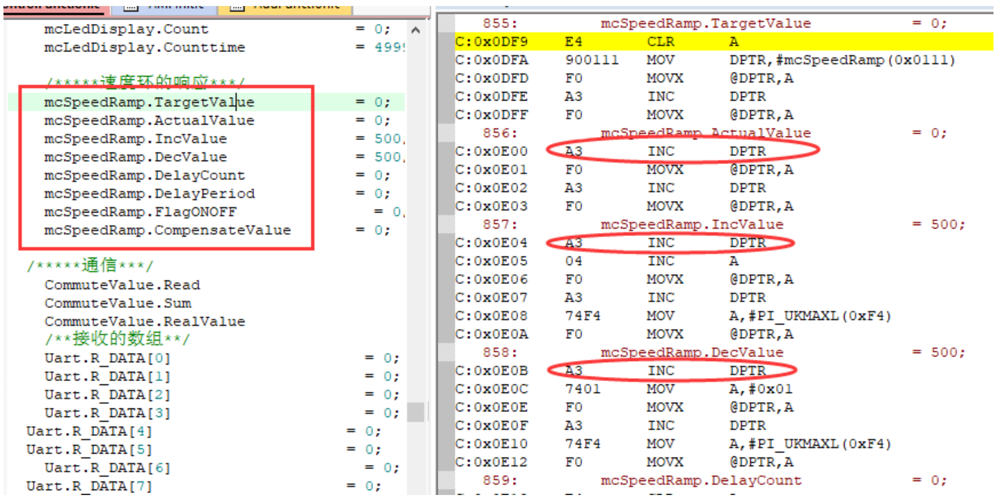
```
//          while(1);

            if(h_Fault->mcEsValue <30)  //反电动势值调节范围      这个值应该 设置比堵转的
            {
               h_Fault->mcStartCnt++;
               if(h_Fault->mcStartCnt>=10)
               {
                 mcFaultSource=FaultStart;
                 mcState = mcFault;
                 h_Fault->mcStartFocmode=2;
                 FaultProcess();
                 h_Fault->SecondStart++;
                 h_Fault->mcStartDelay=0;
                 h_Fault->mcStartCnt=0;
               }
            }
            else
            {
              h_Fault->mcStartCnt=0;
            }
          }
      }
            //方法三
       if(mcFocCtrl.CtrlMode==0)
       {
         h_Fault->mcStartFocmode++;
         if(h_Fault->mcStartFocmode>=100)//400*5/1000=2s
         {
           h_Fault->mcStartFocmode=3;
           mcFaultSource=FaultStart;
           mcState = mcFault;

           FaultProcess();
           h_Fault->SecondStart++;
         }
       }
    }
```

2 Structure Members

As shown below, structure members are initialized or assigned with variables in the defined order so as to reduce code space.



```
#ifndef __UARTINIT_H_
#define __UARTINIT_H_

typedef struct
{
    uint16      R_DATA[10];//={0,0,0,0,0,0,0,0,0,0};
  uint16      T_DATA[10];//={0,0,0,0,0,0,0,0,0,0};
    uint16    Uredata;
  uint16    UARxCnt;
  uint16    RxFSM;
  uint16    UsaRxLen;
  uint16    flagUsaRxComm;
  uint32    CheckDate;
    uint16    ResponceCount;
    uint8     ResponceFlag;
    uint8     UsaTxlen;
    uint8     SendCnt;
}MCUART;

```

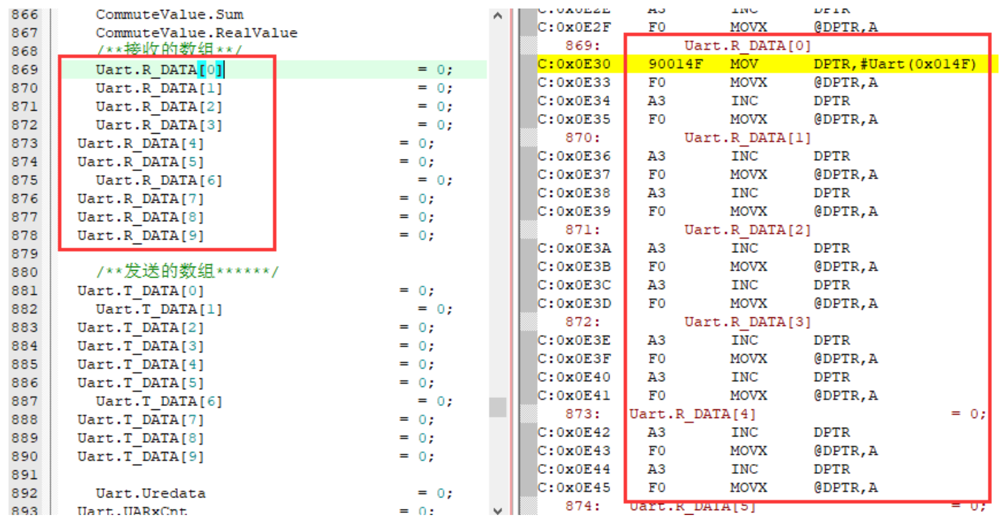


```
    mcLedDisplay.Count                        = 0;
    mcLedDisplay.Counttime                    = 499

    /*****速度环的响应***/
    mcSpeedRamp.TargetValue                   = 0;
    mcSpeedRamp.ActualValue                   = 0;
    mcSpeedRamp.IncValue                      = 500,
    mcSpeedRamp.DecValue                      = 500,
    mcSpeedRamp.DelayCount                    = 0;
    mcSpeedRamp.DelayPeriod                   = 0;
    mcSpeedRamp.FlagONOFF                       = 0,
    mcSpeedRamp.CompensateValue               = 0;
  /*****通信***/
    CommuteValue.Read
    CommuteValue.Sum
    CommuteValue.RealValue
    /**接收的数组**/
    Uart.R_DATA[0]                              = 0;
    Uart.R_DATA[1]                              = 0;
    Uart.R_DATA[2]                              = 0;
    Uart.R_DATA[3]                              = 0;
  Uart.R_DATA[4]                              = 0;
  Uart.R_DATA[5]                              = 0;
    Uart.R_DATA[6]                              = 0;
  Uart.R_DATA[7]                              = 0;
```

```
   855:          mcSpeedRamp.TargetValue              = 0;
C:0x0DF9    E4          CLR       A
C:0x0DFA    900111      MOV       DPTR,#mcSpeedRamp(0x0111)
C:0x0DFD    F0          MOVX      @DPTR,A
C:0x0DFE    A3          INC       DPTR
C:0x0DFF    F0          MOVX      @DPTR,A
   856:          mcSpeedRamp.ActualValue              = 0;
C:0x0E00    A3          INC       DPTR
C:0x0E01    F0          MOVX      @DPTR,A
C:0x0E02    A3          INC       DPTR
C:0x0E03    F0          MOVX      @DPTR,A
   857:          mcSpeedRamp.IncValue                 = 500;
C:0x0E04    A3          INC       DPTR
C:0x0E05    04          INC       A
C:0x0E06    F0          MOVX      @DPTR,A
C:0x0E07    A3          INC       DPTR
C:0x0E08    74F4        MOV       A,#PI_UKMAXL(0xF4)
C:0x0E0A    F0          MOVX      @DPTR,A
   858:          mcSpeedRamp.DecValue                 = 500;
C:0x0E0B    A3          INC       DPTR
C:0x0E0C    7401        MOV       A,#0x01
C:0x0E0E    F0          MOVX      @DPTR,A
C:0x0E0F    A3          INC       DPTR
C:0x0E10    74F4        MOV       A,#PI_UKMAXL(0xF4)
C:0x0E12    F0          MOVX      @DPTR,A
   859:          mcSpeedRamp.DelayCount               = 0;
```

3 Array Members in Structure

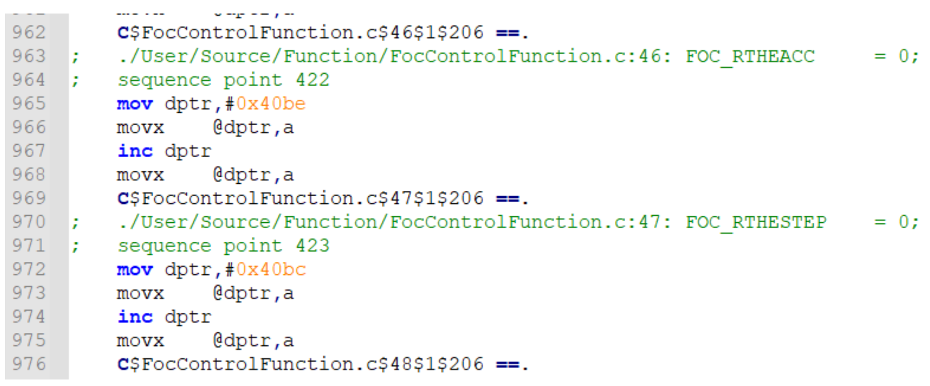
As shown below, array members in the structure are initialized or assigned with variables in the defined order so as to reduce code space.



```
    CommuteValue.Sum
    CommuteValue.RealValue
    /**接收的数组**/
    Uart.R_DATA[0]                      = 0;
    Uart.R_DATA[1]                      = 0;
    Uart.R_DATA[2]                      = 0;
    Uart.R_DATA[3]                      = 0;
  Uart.R_DATA[4]                      = 0;
  Uart.R_DATA[5]                      = 0;
    Uart.R_DATA[6]                      = 0;
  Uart.R_DATA[7]                      = 0;
  Uart.R_DATA[8]                      = 0;
  Uart.R_DATA[9]                      = 0;

    /**发送的数组******/
  Uart.T_DATA[0]                      = 0;
    Uart.T_DATA[1]                      = 0;
  Uart.T_DATA[2]                      = 0;
  Uart.T_DATA[3]                      = 0;
  Uart.T_DATA[4]                      = 0;
  Uart.T_DATA[5]                      = 0;
    Uart.T_DATA[6]                      = 0;
  Uart.T_DATA[7]                      = 0;
  Uart.T_DATA[8]                      = 0;
  Uart.T_DATA[9]                      = 0;

    Uart.Uredata                        = 0;
  Uart.UARxCnt                        = 0;
```

```
C:0x0E2E   A3       INC        DPTR
C:0x0E2F   F0       MOVX       @DPTR,A
   869:          Uart.R_DATA[0]
C:0x0E30   90014F   MOV        DPTR,#Uart(0x014F)
C:0x0E33   F0       MOVX       @DPTR,A
C:0x0E34   A3       INC        DPTR
C:0x0E35   F0       MOVX       @DPTR,A
   870:          Uart.R_DATA[1]
C:0x0E36   A3       INC        DPTR
C:0x0E37   F0       MOVX       @DPTR,A
C:0x0E38   A3       INC        DPTR
C:0x0E39   F0       MOVX       @DPTR,A
   871:          Uart.R_DATA[2]
C:0x0E3A   A3       INC        DPTR
C:0x0E3B   F0       MOVX       @DPTR,A
C:0x0E3C   A3       INC        DPTR
C:0x0E3D   F0       MOVX       @DPTR,A
   872:          Uart.R_DATA[3]
C:0x0E3E   A3       INC        DPTR
C:0x0E3F   F0       MOVX       @DPTR,A
C:0x0E40   A3       INC        DPTR
C:0x0E41   F0       MOVX       @DPTR,A
   873:   Uart.R_DATA[4]                            = 0;
C:0x0E42   A3       INC        DPTR
C:0x0E43   F0       MOVX       @DPTR,A
C:0x0E44   A3       INC        DPTR
C:0x0E45   F0       MOVX       @DPTR,A
   874:   Uart.R_DATA[5]                            = 0;
```

If we don’t follow the order, the array members in our IDE are shown as below. But if we follow the order, we can save 2byte code space.

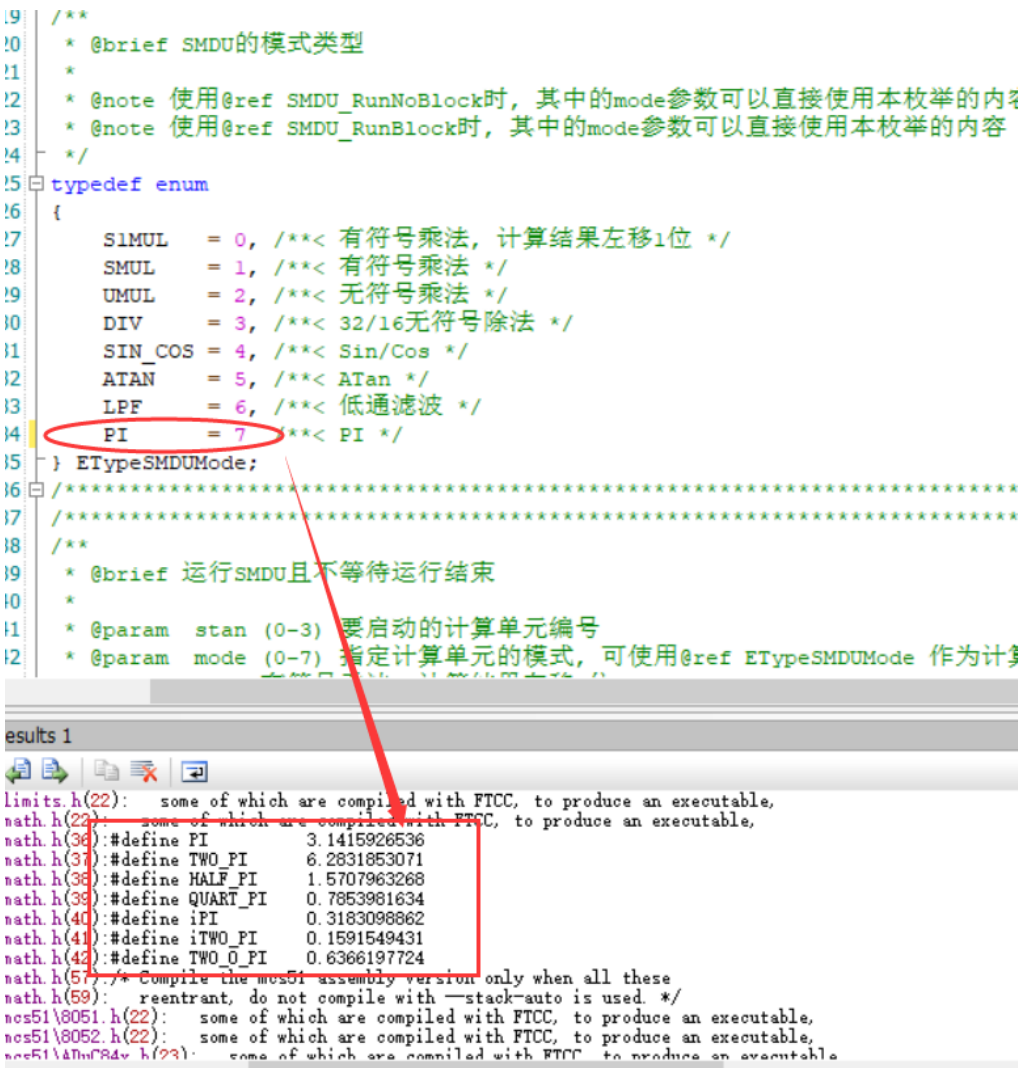


```
    C$FocControlFunction.c$46$1$206 ==.
;   ./User/Source/Function/FocControlFunction.c:46: FOC_RTHEACC     = 0;
;   sequence point 422
    mov dptr,#0x40be
    movx    @dptr,a
    inc dptr
    movx    @dptr,a
    C$FocControlFunction.c$47$1$206 ==.
;   ./User/Source/Function/FocControlFunction.c:47: FOC_RTHESTEP    = 0;
;   sequence point 423
    mov dptr,#0x40bc
    movx    @dptr,a
    inc dptr
    movx    @dptr,a
    C$FocControlFunction.c$48$1$206 ==.
```

4 Enumeration Members

Enumeration type MUST NOT have a member with the same name as the macro name defined in the system library function. As shown below, ETypeSMDUMode have a member called PI, and the header file in Math Library also contains a macro called PI (or Pi, ratio of circumference to diameter). In this case, if a PI is defined in the IDE, invocation error may occur.

Solution: Define different names with those in the library. For example, you can use the name PII or P_I to replace PI.



```
/**
 * @brief SMDU的模式类型
 *
 * @note 使用@ref SMDU_RunNoBlock时，其中的mode参数可以直接使用本枚举的内容
 * @note 使用@ref SMDU_RunBlock时，其中的mode参数可以直接使用本枚举的内容
 */
typedef enum
{
    S1MUL    = 0, /**< 有符号乘法，计算结果左移1位 */
    SMUL     = 1, /**< 有符号乘法 */
    UMUL     = 2, /**< 无符号乘法 */
    DIV      = 3, /**< 32/16无符号除法 */
    SIN_COS  = 4, /**< Sin/Cos */
    ATAN     = 5, /**< ATan */
    LPF      = 6, /**< 低通滤波 */
    PI       = 7, /**< PI */
} ETypeSMDUMode;
/*************************************************************************
/*************************************************************************
/**
 * @brief 运行SMDU且不等待运行结束
 *
 * @param stan (0-3) 要启动的计算单元编号
 * @param mode (0-7) 指定计算单元的模式，可使用@ref ETypeSMDUMode 作为计算
```

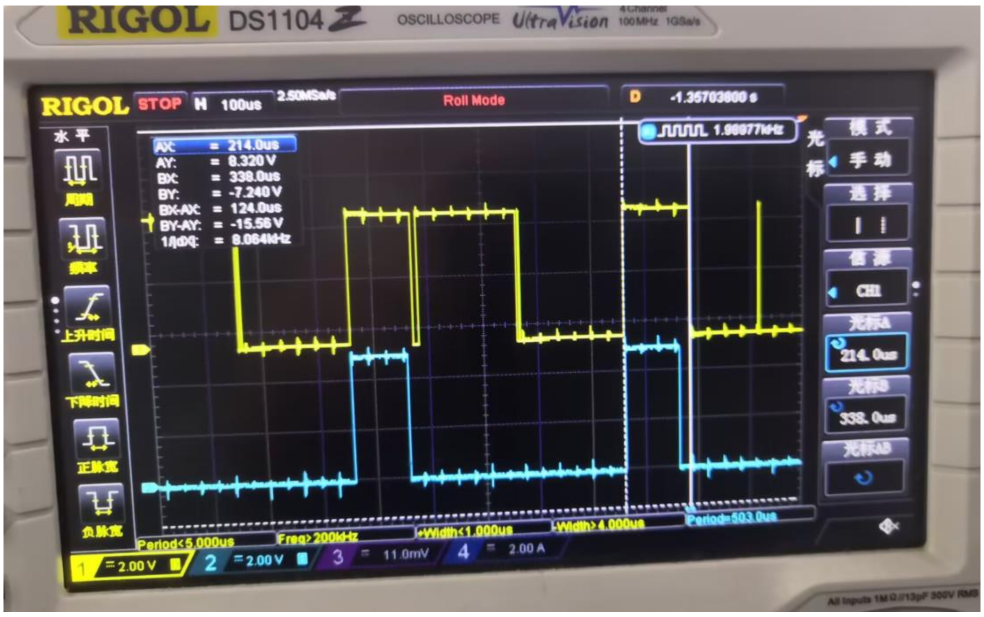
esults 1

```
limits.h(22):   some of which are compiled with FTCC, to produce an executable,
nath.h(22):   some of which are compiled with FTCC, to produce an executable,
nath.h(36):#define PI          3.1415926536
nath.h(37):#define TWO_PI      6.2831853071
nath.h(38):#define HALF_PI     1.5707963268
nath.h(39):#define QUART_PI    0.7853981634
nath.h(40):#define iPI         0.3183098862
nath.h(41):#define iTWO_PI     0.1591549431
nath.h(42):#define TWO_O_PI    0.6366197724
nath.h(57):/* Compile the mcs51 assembly version only when all these
nath.h(59):   reentrant, do not compile with --stack-auto is used. */
ncs51\8051.h(22):   some of which are compiled with FTCC, to produce an executable,
ncs51\8052.h(22):   some of which are compiled with FTCC, to produce an executable,
ncs51\ADuC84x.h(23):   some of which are compiled with FTCC, to produce an executable
```

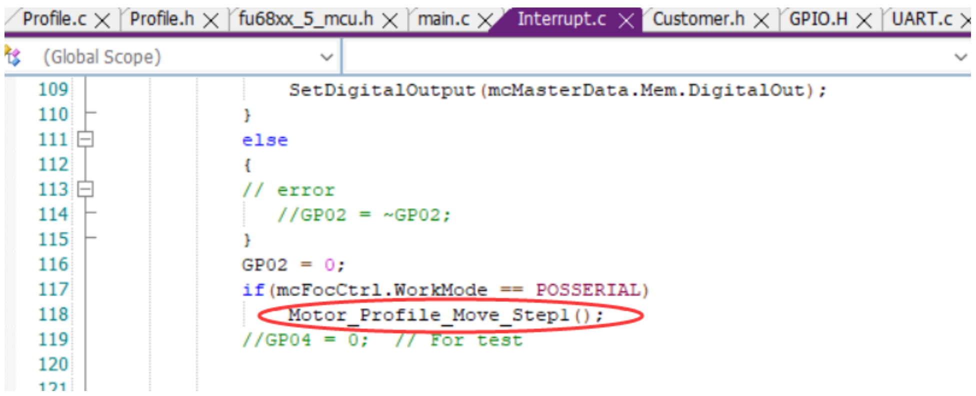
5 Project Porting

5.1 Timing Problem 1

A project (such as servo motor) may run overtime after it is ported and complied successfully. After code analysis, it is found that the timing for DMA to receive or transmit the interrupts is tight, as shown below.

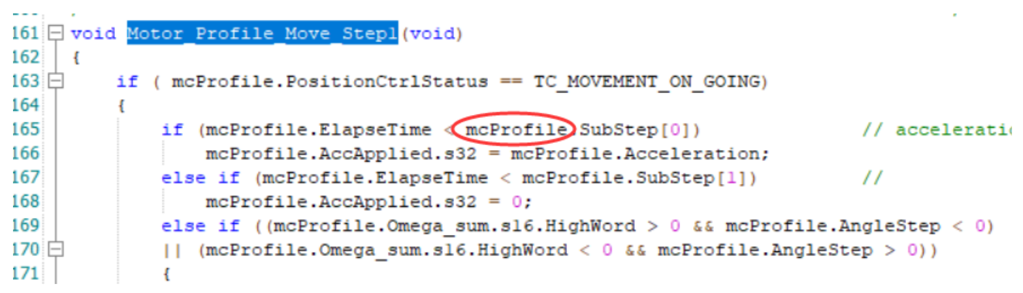


Cause: IDE always runs overtime after **Start** is clicked to process the data received by MCU2, because it runs to the following function.



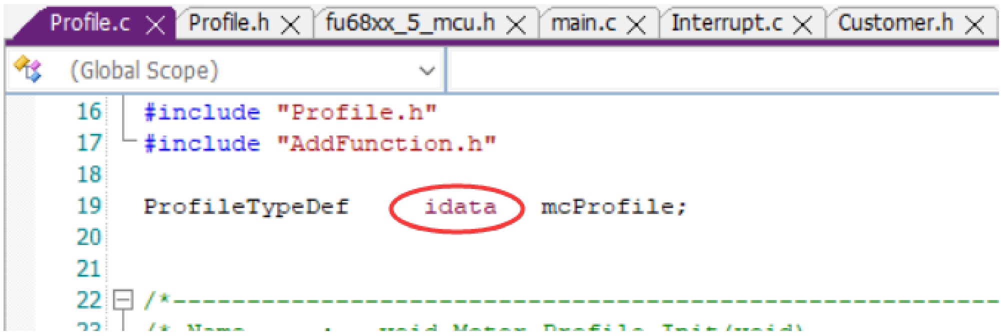
```
        SetDigitalOutput(mcMasterData.Mem.DigitalOut);
    }
    else
    {
    // error
        //GP02 = ~GP02;
    }
    GP02 = 0;
    if(mcFocCtrl.WorkMode == POSSERIAL)
        Motor_Profile_Move_Step1();
    //GP04 = 0;   // For test
```

This function contains mcProfie structure, and this variable is defined in xdata. It takes more time than usual for the compiler to access xdata. In this case, we change the memory type of this variable to idata, so that IDE processes this function with fewer instructions to ensure the received interrupt is processed in time, reducing MCU execution time.



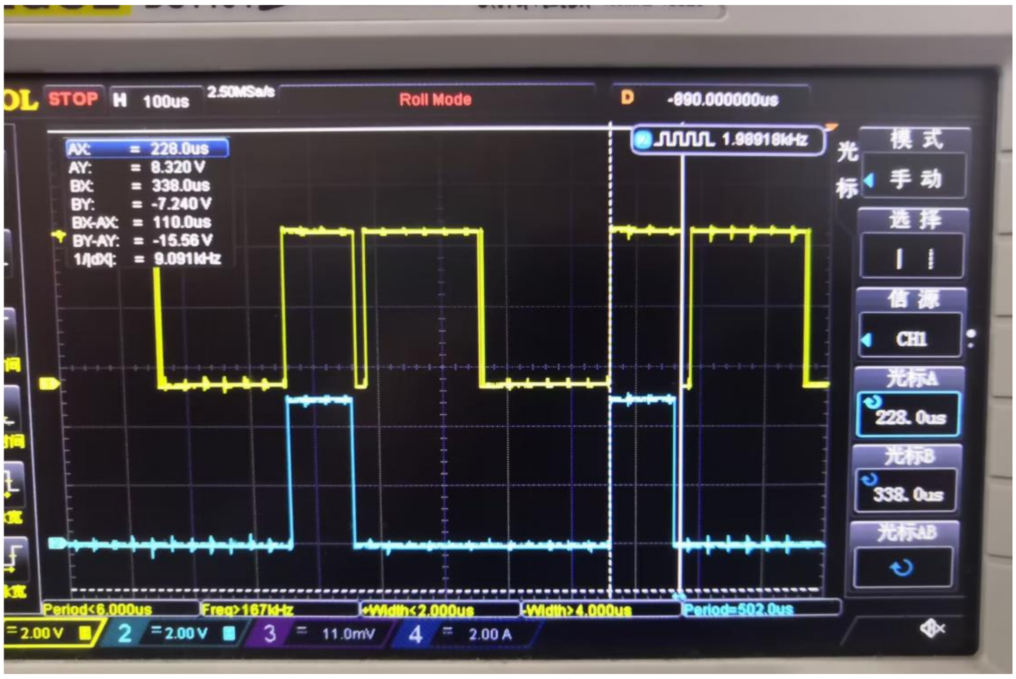
```
void Motor_Profile_Move_Step1(void)
{
    if ( mcProfile.PositionCtrlStatus == TC_MOVEMENT_ON_GOING)
    {
        if (mcProfile.ElapseTime < mcProfile.SubStep[0])            // accelerati
            mcProfile.AccApplied.s32 = mcProfile.Acceleration;
        else if (mcProfile.ElapseTime < mcProfile.SubStep[1])       //
            mcProfile.AccApplied.s32 = 0;
        else if ((mcProfile.Omega_sum.s16.HighWord > 0 && mcProfile.AngleStep < 0)
        || (mcProfile.Omega_sum.s16.HighWord < 0 && mcProfile.AngleStep > 0))
        {
```

Solution: Find the structure nodes where timeout occurs and optimize the number of instructions to reduce execution time. For example, due to the complier, it is delayed by more than 10us to process the service. After the structure variable mcProfile in the function is optimized, the interrupt flag bit is executed in time and the program runs normally.



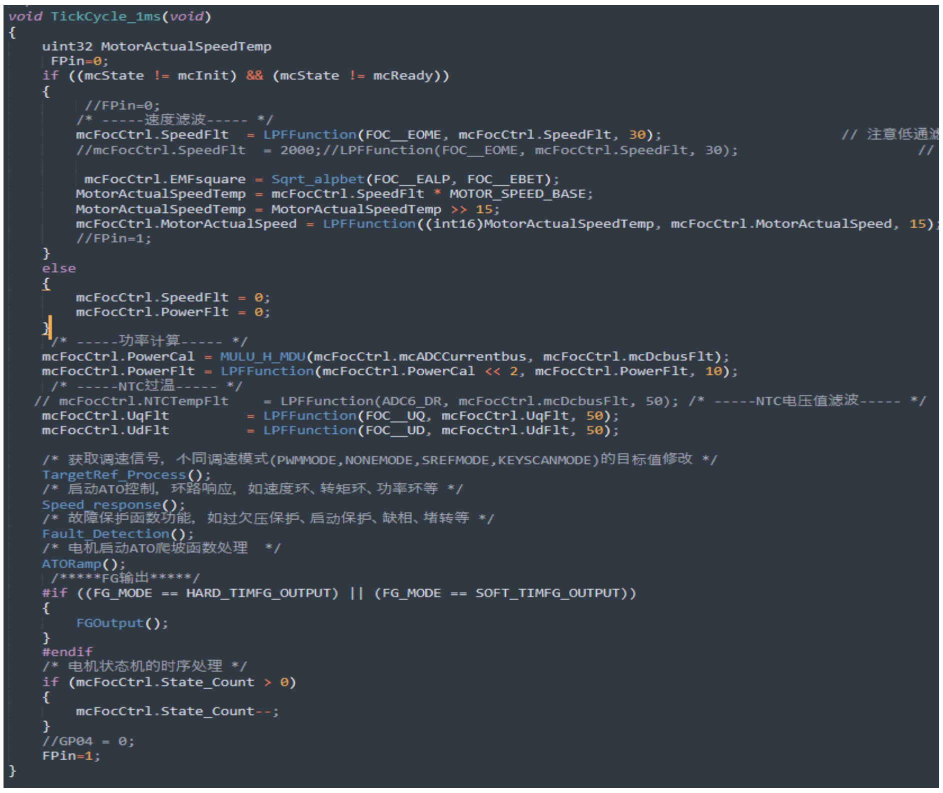
```
#include "Profile.h"
#include "AddFunction.h"

ProfileTypeDef   idata   mcProfile;


/*-----------------------------------------------------------
```

The waveform after code optimization is shown as below:



5.2 Timing Problem 2

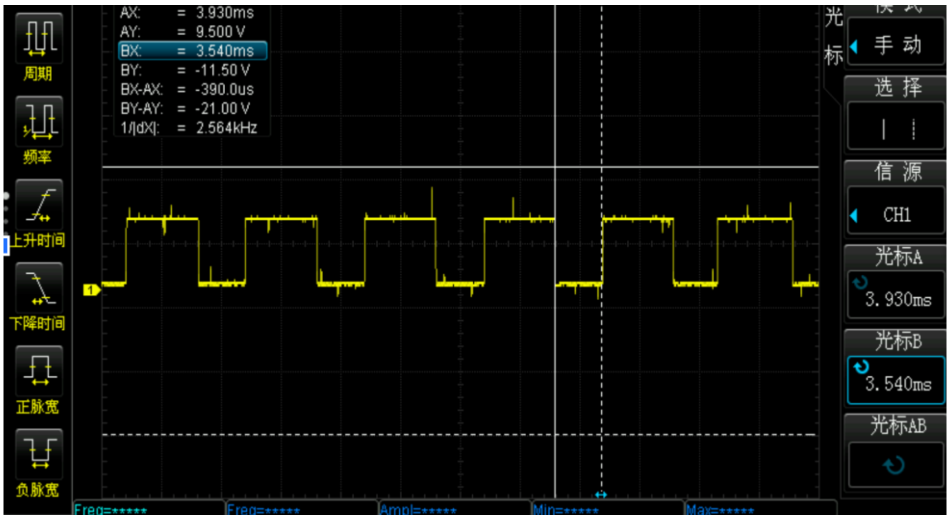
Here is an example to describe how the IDE processes 6572 project. The entire function is executed in 1ms, and the test code is shown as below.



```
void TickCycle_1ms(void)
{
    uint32 MotorActualSpeedTemp
     FPin=0;
    if ((mcState != mcInit) && (mcState != mcReady))
    {
         //FPin=0;
        /* -----速度滤波----- */
        mcFocCtrl.SpeedFlt  = LPFFunction(FOC__EOME, mcFocCtrl.SpeedFlt, 30);                // 注意低通滤
        //mcFocCtrl.SpeedFlt  = 2000;//LPFFunction(FOC__EOME, mcFocCtrl.SpeedFlt, 30);                //

         mcFocCtrl.EMFsquare = Sqrt_alpbet(FOC__EALP, FOC__EBET);
        MotorActualSpeedTemp = mcFocCtrl.SpeedFlt * MOTOR_SPEED_BASE;
        MotorActualSpeedTemp = MotorActualSpeedTemp >> 15;
        mcFocCtrl.MotorActualSpeed = LPFFunction((int16)MotorActualSpeedTemp, mcFocCtrl.MotorActualSpeed, 15);
        //FPin=1;
    }
    else
    {
        mcFocCtrl.SpeedFlt = 0;
        mcFocCtrl.PowerFlt = 0;
    }
     /* -----功率计算----- */
    mcFocCtrl.PowerCal = MULU_H_MDU(mcFocCtrl.mcADCCurrentbus, mcFocCtrl.mcDcbusFlt);
    mcFocCtrl.PowerFlt = LPFFunction(mcFocCtrl.PowerCal << 2, mcFocCtrl.PowerFlt, 10);
     /* -----NTC过温----- */
   // mcFocCtrl.NTCTempFlt    = LPFFunction(ADC6_DR, mcFocCtrl.mcDcbusFlt, 50); /* -----NTC电压值滤波----- */
    mcFocCtrl.UqFlt          = LPFFunction(FOC__UQ, mcFocCtrl.UqFlt, 50);
    mcFocCtrl.UdFlt          = LPFFunction(FOC__UD, mcFocCtrl.UdFlt, 50);

    /* 获取调速信号, 不同调速模式(PWMMODE,NONEMODE,SREFMODE,KEYSCANMODE)的目标值修改 */
    TargetRef_Process();
    /* 启动ATO控制, 环路响应, 如速度环、转矩环、功率环等 */
    Speed_response();
    /* 故障保护函数功能, 如过欠压保护、启动保护、缺相、堵转等 */
    Fault_Detection();
    /* 电机启动ATO爬坡函数处理  */
    ATORamp();
     /*****FG输出*****/
    #if ((FG_MODE == HARD_TIMFG_OUTPUT) || (FG_MODE == SOFT_TIMFG_OUTPUT))
    {
        FGOutput();
    }
    #endif
    /* 电机状态机的时序处理 */
    if (mcFocCtrl.State_Count > 0)
    {
        mcFocCtrl.State_Count--;
    }
    //GP04 = 0;
    FPin=1;
}
```

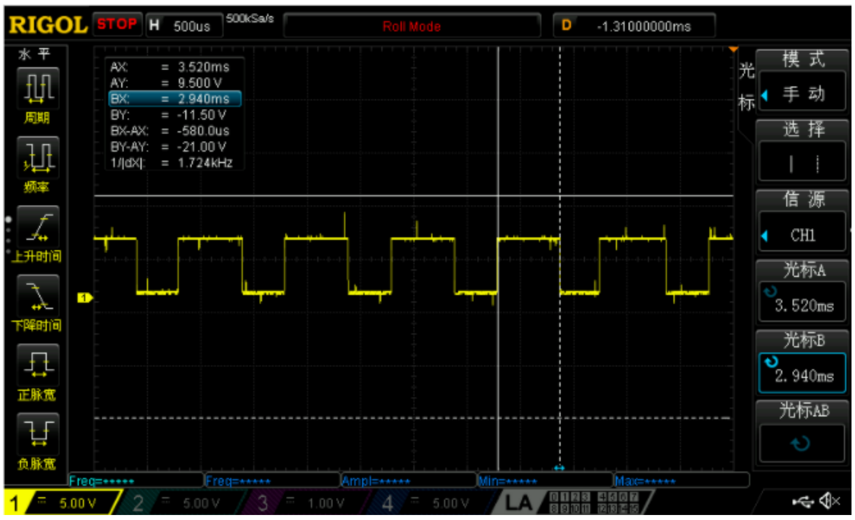
Figure 5-1 Test Code

The waveform during normal operation is shown in Figure 2 and Figure with L: 390us and H: 580us.



```
AX:    =  3.930ms
AY:    =  9.500 V
BX:    =  3.540ms
BY:    =  -11.50 V
BX-AX: =  -390.0us
BY-AY: =  -21.00 V
1/|dX|: = 2.564kHz
```

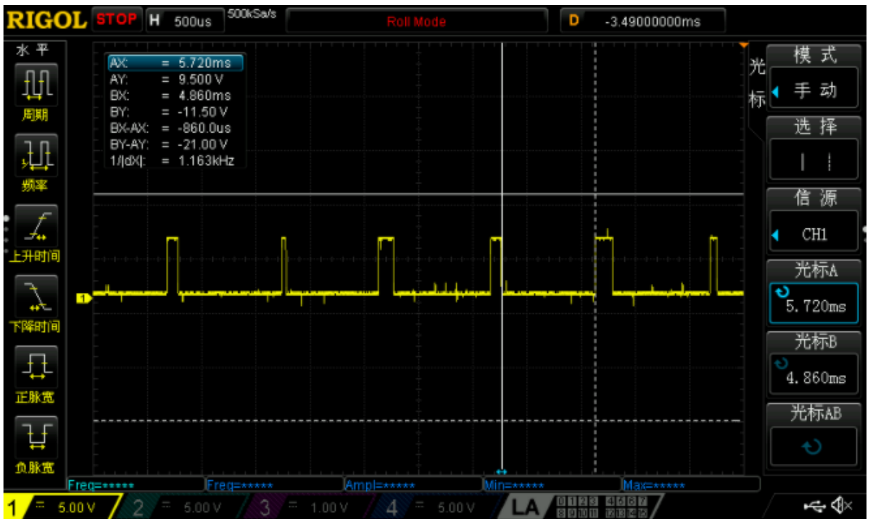
Figure 5-2 Waveform during Normal Operation



```
AX:     =  3.520ms
AY:     =  9.500 V
BX:     =  2.940ms
BY:     =  -11.50 V
BX-AX:  =  -580.0us
BY-AY:  =  -21.00 V
1/|dX|: =  1.724kHz
```

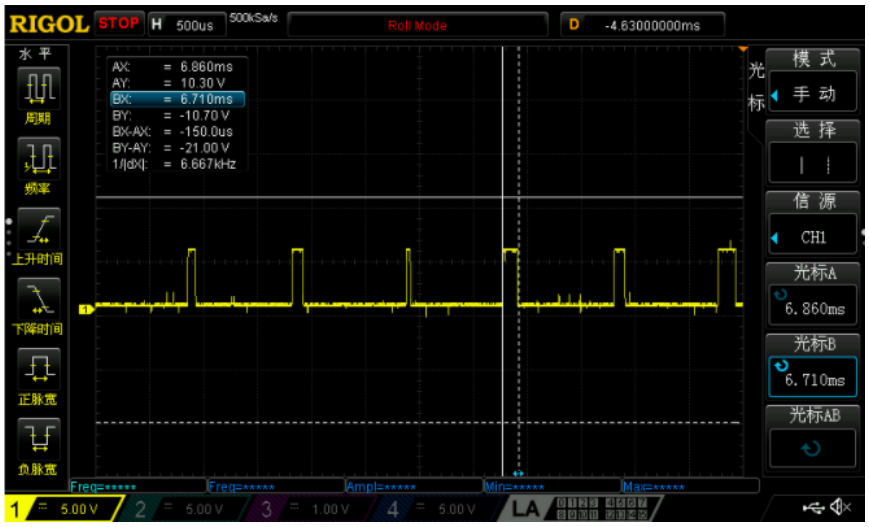
Figure 5-3 Waveform during Normal Operation

The waveform during IDE operation is shown in Figure 4 and Figure 5 with L: 860us and H: 150us. The timing is in disorder.



```
AX:     =  5.720ms
AY:     =  9.500 V
BX:     =  4.860ms
BY:     =  -11.50 V
BX-AX:  =  -860.0us
BY-AY:  =  -21.00 V
1/|dX|: =  1.163kHz
```

Figure 5-4 Waveform during IDE Operation

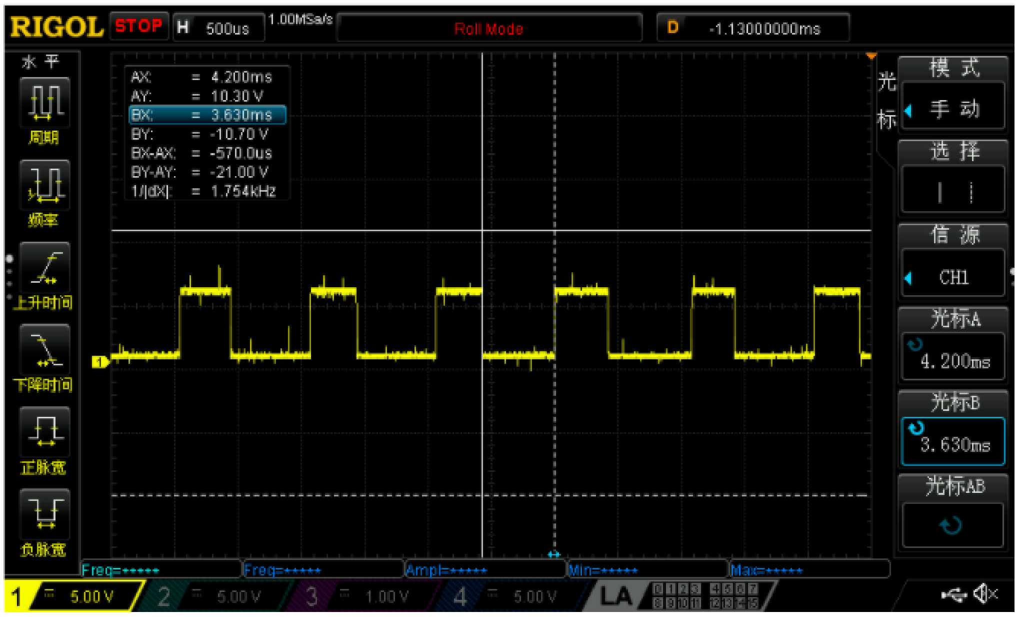


```
AX:     =  6.860ms
AY:     =  10.30 V
BX:     =  6.710ms
BY:     =  -10.70 V
BX-AX:  =  -150.0us
BY-AY:  =  -21.00 V
1/|dX|: =  6.667kHz
```

Figure 5-5 Waveform during IDE Operation

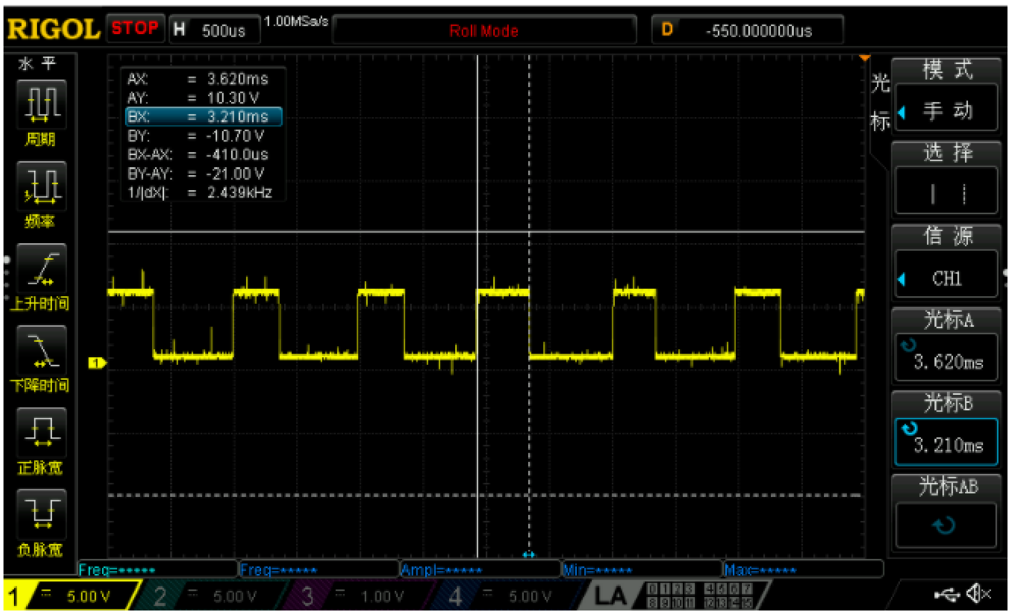
After code analysis and code optimization, since some sampling time is implemented for carrier interrupt, a global variable is used to store power data and the variable type xdata is changed to idata. The test results are as follows:

After optimization, the waveform during IDE operation is shown in Figure 6 and Figure 7 with L: 570us and H: 410us.



```
RIGOL  STOP  H 500us  1.00MSa/s  Roll Mode  D -1.13000000ms

AX:      = 4.200ms
AY:      = 10.30 V
BX:      = 3.630ms
BY:      = -10.70 V
BX-AX:   = -570.0us
BY-AY:   = -21.00 V
1/|dX|:  = 1.754kHz

模 式   手 动
选 择
信 源   CH1
光标A   4.200ms
光标B   3.630ms
光标AB

Freq=*****  Freq=*****  Ampl=*****  Min=*****  Max=*****
1 = 5.00 V   2 = 5.00 V   3 = 1.00 V   4 = 5.00 V   LA
```

Figure 5-6 Waveform during IDE Operation after Modifications



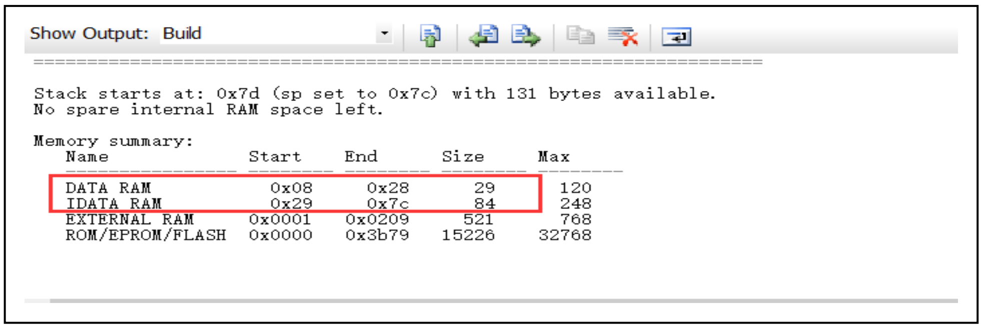
```
RIGOL  STOP  H 500us  1.00MSa/s  Roll Mode  D -550.000000us

AX:      = 3.620ms
AY:      = 10.30 V
BX:      = 3.210ms
BY:      = -10.70 V
BX-AX:   = -410.0us
BY-AY:   = -21.00 V
1/|dX|:  = 2.439kHz

模 式   手 动
选 择
信 源   CH1
光标A   3.620ms
光标B   3.210ms
光标AB

Freq=*****  Freq=*****  Ampl=*****  Min=*****  Max=*****
1 = 5.00 V   2 = 5.00 V   3 = 1.00 V   4 = 5.00 V   LA
```

Figure 5-7 Waveform during IDE Operation after Modifications

Conclusion: The variables that affect the timing most are the global variables xdata and idata.

6 Stack Overflow

Check compilation results for this problem.

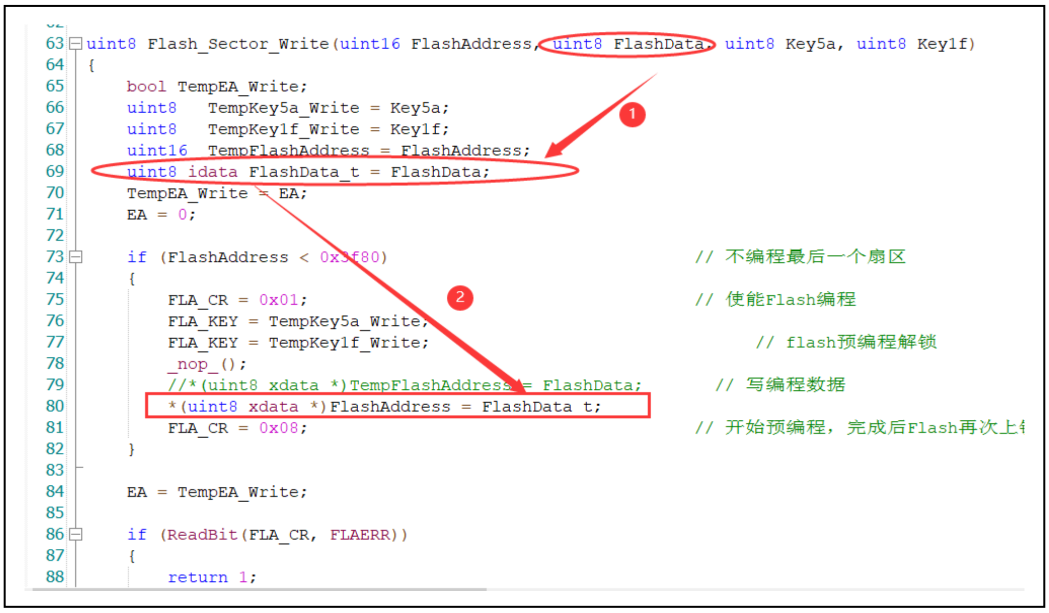


```
=====================================================================

Stack starts at: 0x7d (sp set to 0x7c) with 131 bytes available.
No spare internal RAM space left.

Memory summary:
   Name             Start    End      Size     Max
   ---------------- -------- -------- -------- --------
   DATA RAM           0x08     0x28       29      120
   IDATA RAM          0x29     0x7c       84      248
   EXTERNAL RAM     0x0001   0x0209      521      768
   ROM/EPROM/FLASH  0x0000   0x3b79    15226    32768
```

Stack overflow may occurs if the size occupied by idata RAM and data RMB is too large, generally more than 160. Moreover, the interrupt function shall not be processed in a very complex manner and shall not be nested too deeply. In this case, the function parameters shall be as few as possible. A return value or a pointer shall not be defined if the return result can be obtained a global variable, as it invisibly increases the capacity of the stack, or even worse, it may cause stack corruption for those functions with deep and complicated procedure layers. Therefore, if a program resets or runs abnormally for no reason, you can check whether the stack size is too small, or decrease data and idata and assign some variables to xdata memory region.

7 Flash Self-programming

If it fails to read or write the Fash in IDE project, you can solve the problem by following the steps as below.



```
63 uint8 Flash_Sector_Write(uint16 FlashAddress, uint8 FlashData, uint8 Key5a, uint8 Key1f)
64 {
65     bool TempEA_Write;
66     uint8   TempKey5a_Write = Key5a;
67     uint8   TempKey1f_Write = Key1f;
68     uint16  TempFlashAddress = FlashAddress;
69     uint8 idata FlashData_t = FlashData;
70     TempEA_Write = EA;
71     EA = 0;
72
73     if (FlashAddress < 0x3f80)                          // 不编程最后一个扇区
74     {
75         FLA_CR = 0x01;                                  // 使能Flash编程
76         FLA_KEY = TempKey5a_Write;
77         FLA_KEY = TempKey1f_Write;                           // flash预编程解锁
78         _nop_();
79         //*(uint8 xdata *)TempFlashAddress = FlashData;    // 写编程数据
80         *(uint8 xdata *)FlashAddress = FlashData_t;
81         FLA_CR = 0x08;                                  // 开始预编程，完成后Flash再次上
82     }
83
84     EA = TempEA_Write;
85
86     if (ReadBit(FLA_CR, FLAERR))
87     {
88         return 1;
```

Step 1: Assign this variable to a non-XRAM variable, as shown in the figure above, and then define a temporary variable FlashData_t in idata to receive data to be stored in the Flash;

Step 2: Comment out the previous write operations (line 79 in the above figure), and then rewrite one (line 80 in the figure), that is, the previous FlashData is replaced by the new temporary variable FlashData_t for the write operations. The codes on the left remain the same.

Generally, the byte data is written to the corresponding Flash address after above operations.

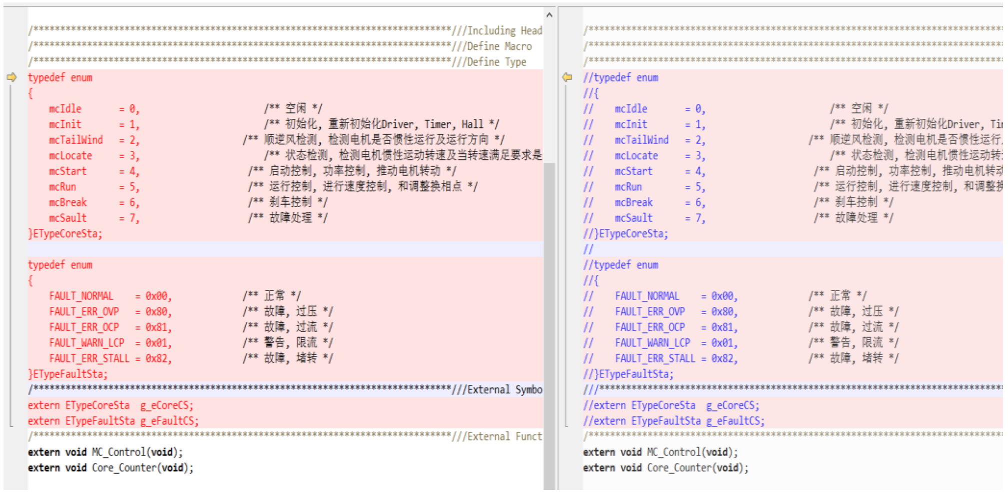
8 Code Size Reduction

The following uses FT5821 as an example.

8.1 Redundant Types and Variable Declarations

Comment out or delete types and variable declarations that are not used in the header file.

As shown below, the type definitions on the left are not used throughout the program, so you can comment them out (see the figure on the right).

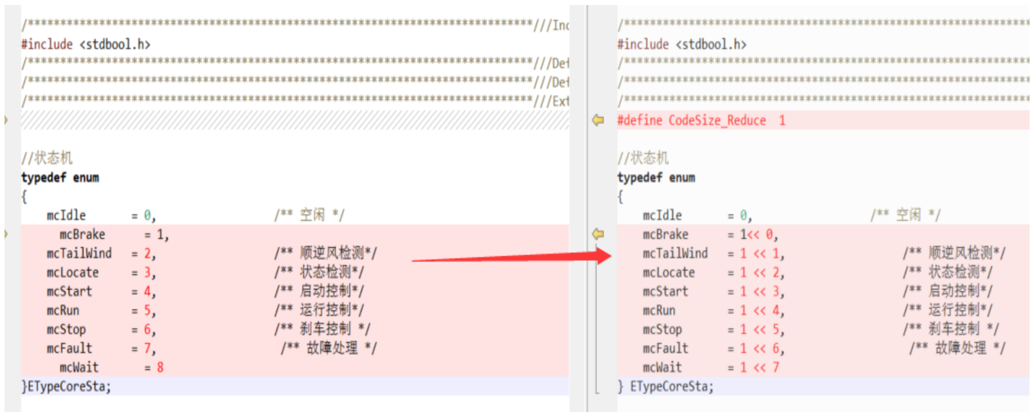


```
/*************************************************************************///Including Head
/*************************************************************************///Define Macro
/*************************************************************************///Define Type
typedef enum
{
    mcIdle        = 0,                    /** 空闲 */
    mcInit        = 1,                    /** 初始化，重新初始化Driver, Timer, Hall */
    mcTailWind    = 2,                /** 顺逆风检测，检测电机是否惯性运行及运行方向 */
    mcLocate      = 3,                    /** 状态检测，检测电机惯性运动转速及当转速满足要求是
    mcStart       = 4,                /** 启动控制，功率控制，推动电机转动 */
    mcRun         = 5,                /** 运行控制，进行速度控制，和调整换相点 */
    mcBreak       = 6,                /** 刹车控制 */
    mcSault       = 7,                /** 故障处理 */
}ETypeCoreSta;

typedef enum
{
    FAULT_NORMAL    = 0x00,           /** 正常 */
    FAULT_ERR_OVP   = 0x80,           /** 故障，过压 */
    FAULT_ERR_OCP   = 0x81,           /** 故障，过流 */
    FAULT_WARN_LCP  = 0x01,           /** 警告，限流 */
    FAULT_ERR_STALL = 0x82,           /** 故障，堵转 */
}ETypeFaultSta;
/*************************************************************************///External Symbo
extern ETypeCoreSta  g_eCoreCS;
extern ETypeFaultSta g_eFaultCS;
/*************************************************************************///External Funct
extern void MC_Control(void);
extern void Core_Counter(void);
```

```
/*************************************************************************
/*************************************************************************
/*************************************************************************
//typedef enum
//{
//    mcIdle        = 0,                    /** 空闲 */
//    mcInit        = 1,                    /** 初始化，重新初始化Driver, Ti
//    mcTailWind    = 2,                /** 顺逆风检测，检测电机是否惯性运行
//    mcLocate      = 3,                    /** 状态检测，检测电机惯性运动转
//    mcStart       = 4,                /** 启动控制，功率控制，推动电机转动
//    mcRun         = 5,                /** 运行控制，进行速度控制，和调整换
//    mcBreak       = 6,                /** 刹车控制 */
//    mcSault       = 7,                /** 故障处理 */
//}ETypeCoreSta;
//
//typedef enum
//{
//    FAULT_NORMAL    = 0x00,           /** 正常 */
//    FAULT_ERR_OVP   = 0x80,           /** 故障，过压 */
//    FAULT_ERR_OCP   = 0x81,           /** 故障，过流 */
//    FAULT_WARN_LCP  = 0x01,           /** 警告，限流 */
//    FAULT_ERR_STALL = 0x82,           /** 故障，堵转 */
//}ETypeFaultSta;
///************************************************************************
//extern ETypeCoreSta  g_eCoreCS;
//extern ETypeFaultSta g_eFaultCS;
/*************************************************************************
extern void MC_Control(void);
extern void Core_Counter(void);
```

8.2 Enumeration Type

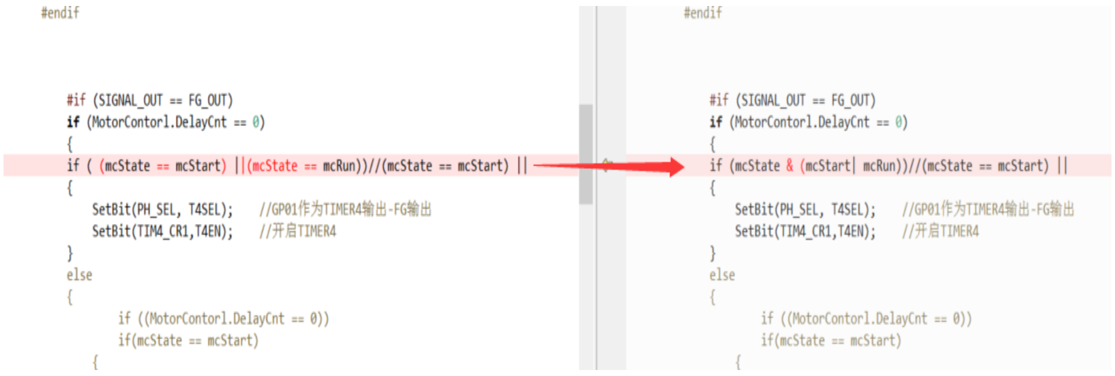
Change the member variables to 1 by left shifting n bits when an enumeration type such as a state machine is defined.



```
/*************************************************************************///Inc
#include <stdbool.h>
/*************************************************************************///Def
/*************************************************************************///Def
/*************************************************************************///Ext


//状态机
typedef enum
{
    mcIdle        = 0,                 /** 空闲 */
      mcBrake       = 1,
    mcTailWind    = 2,                 /** 顺逆风检测*/
    mcLocate      = 3,                 /** 状态检测*/
    mcStart       = 4,                 /** 启动控制*/
    mcRun         = 5,                 /** 运行控制*/
    mcStop        = 6,                 /** 刹车控制 */
    mcFault       = 7,                  /** 故障处理 */
      mcWait        = 8
}ETypeCoreSta;
```

```
/*************************************************************************
#include <stdbool.h>
/*************************************************************************
/*************************************************************************
/*************************************************************************
#define CodeSize_Reduce  1

//状态机
typedef enum
{
    mcIdle        = 0,                 /** 空闲 */
    mcBrake       = 1<< 0,
    mcTailWind    = 1 << 1,                 /** 顺逆风检测*/
    mcLocate      = 1 << 2,                 /** 状态检测*/
    mcStart       = 1 << 3,                 /** 启动控制*/
    mcRun         = 1 << 4,                 /** 运行控制*/
    mcStop        = 1 << 5,                 /** 刹车控制 */
    mcFault       = 1 << 6,                  /** 故障处理 */
    mcWait        = 1 << 7
} ETypeCoreSta;
```

You can also apply bits shift and mask (see the below figure on the right) when an enumeration type such as a state machine is defined. Code writing in the source file shall be modified as well.



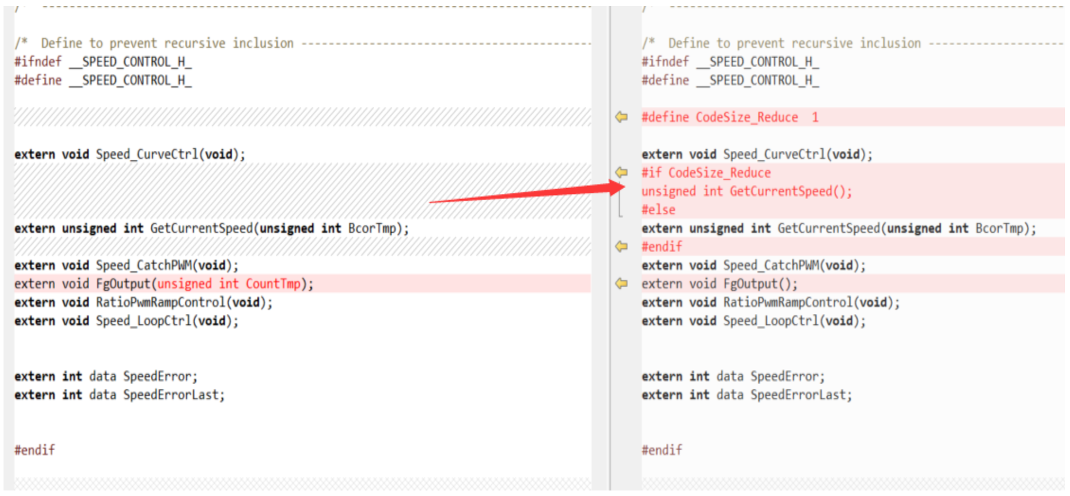
```
#endif


    #if (SIGNAL_OUT == FG_OUT)
    if (MotorContorl.DelayCnt == 0)
    {
    if ( (mcState == mcStart) ||(mcState == mcRun))//(mcState == mcStart) ||
    {
        SetBit(PH_SEL, T4SEL);    //GP01作为TIMER4输出-FG输出
        SetBit(TIM4_CR1,T4EN);    //开启TIMER4
    }
    else
    {
            if ((MotorContorl.DelayCnt == 0))
            if(mcState == mcStart)
        {
```

```
#endif


    #if (SIGNAL_OUT == FG_OUT)
    if (MotorContorl.DelayCnt == 0)
    {
    if (mcState & (mcStart| mcRun))//(mcState == mcStart) ||
    {
        SetBit(PH_SEL, T4SEL);    //GP01作为TIMER4输出-FG输出
        SetBit(TIM4_CR1,T4EN);    //开启TIMER4
    }
    else
    {
            if ((MotorContorl.DelayCnt == 0))
            if(mcState == mcStart)
        {
```

8.3 Function Declaration

Reduce the number of formal parameters in function declaration in the header file, and declare functions in the header file using global structure member variables.

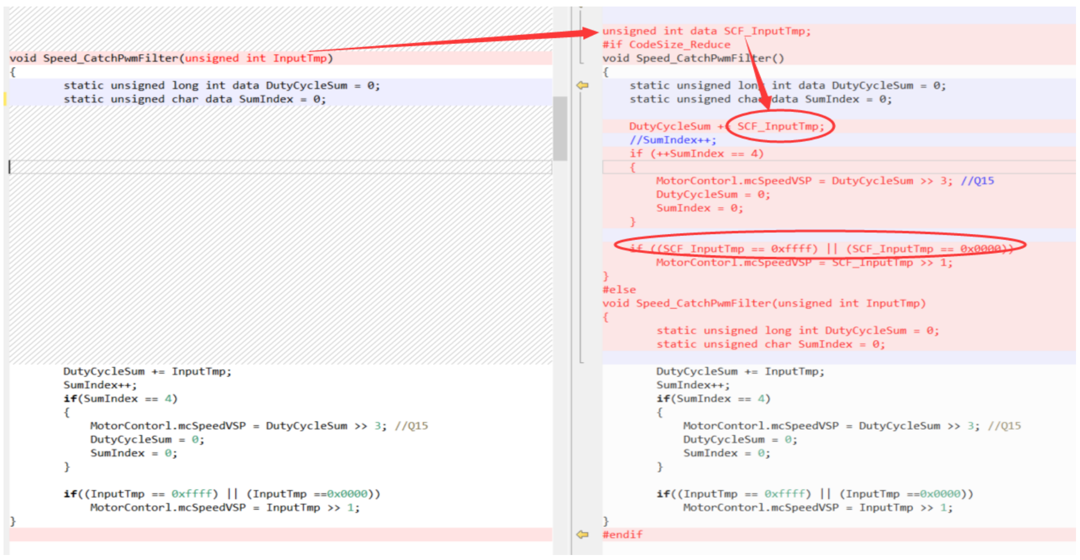
(1) Modify function declaration in the header file



```
/* Define to prevent recursive inclusion ----------------------------------------
#ifndef __SPEED_CONTROL_H_
#define __SPEED_CONTROL_H_


extern void Speed_CurveCtrl(void);



extern unsigned int GetCurrentSpeed(unsigned int BcorTmp);

extern void Speed_CatchPWM(void);
extern void FgOutput(unsigned int CountTmp);
extern void RatioPwmRampControl(void);
extern void Speed_LoopCtrl(void);


extern int data SpeedError;
extern int data SpeedErrorLast;


#endif
```

```
/* Define to prevent recursive inclusion --------------------
#ifndef __SPEED_CONTROL_H_
#define __SPEED_CONTROL_H_

#define CodeSize_Reduce  1

extern void Speed_CurveCtrl(void);
#if CodeSize_Reduce
unsigned int GetCurrentSpeed();
#else
extern unsigned int GetCurrentSpeed(unsigned int BcorTmp);
#endif
extern void Speed_CatchPWM(void);
extern void FgOutput();
extern void RatioPwmRampControl(void);
extern void Speed_LoopCtrl(void);


extern int data SpeedError;
extern int data SpeedErrorLast;


#endif
```

(2) Modify function definitions in the source file

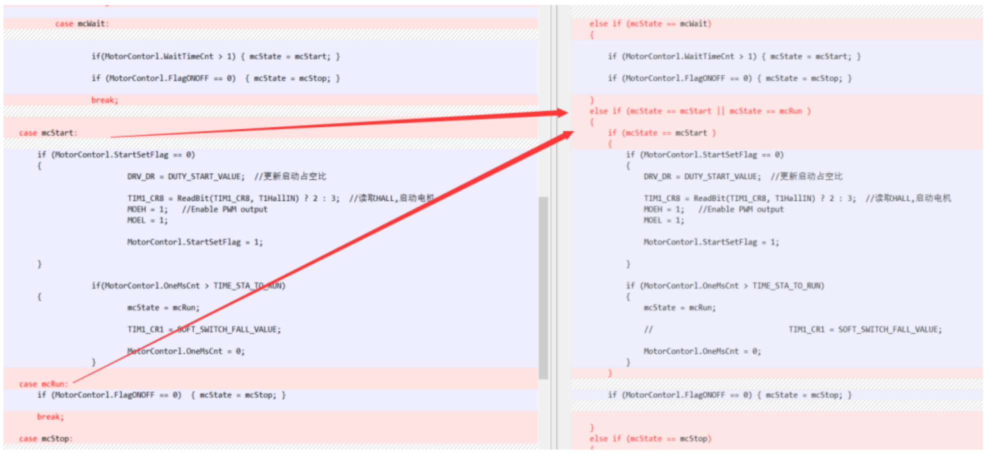


```
void Speed_CatchPwmFilter(unsigned int InputTmp)
{
        static unsigned long int data DutyCycleSum = 0;
        static unsigned char data SumIndex = 0;




        DutyCycleSum += InputTmp;
        SumIndex++;
        if(SumIndex == 4)
        {
            MotorContorl.mcSpeedVSP = DutyCycleSum >> 3; //Q15
            DutyCycleSum = 0;
            SumIndex = 0;
        }

        if((InputTmp == 0xffff) || (InputTmp ==0x0000))
            MotorContorl.mcSpeedVSP = InputTmp >> 1;
}
```

```
unsigned int data SCF_InputTmp;
#if CodeSize_Reduce
void Speed_CatchPwmFilter()
{
    static unsigned long int data DutyCycleSum = 0;
    static unsigned char data SumIndex = 0;

    DutyCycleSum += SCF_InputTmp;
    //SumIndex++;
    if (++SumIndex == 4)
    {
        MotorContorl.mcSpeedVSP = DutyCycleSum >> 3; //Q15
        DutyCycleSum = 0;
        SumIndex = 0;
    }

    if ((SCF_InputTmp == 0xffff) || (SCF_InputTmp == 0x0000))
        MotorContorl.mcSpeedVSP = SCF_InputTmp >> 1;
}
#else
void Speed_CatchPwmFilter(unsigned int InputTmp)
{
        static unsigned long int DutyCycleSum = 0;
        static unsigned char SumIndex = 0;

        DutyCycleSum += InputTmp;
        SumIndex++;
        if(SumIndex == 4)
        {
            MotorContorl.mcSpeedVSP = DutyCycleSum >> 3; //Q15
            DutyCycleSum = 0;
            SumIndex = 0;
        }

        if((InputTmp == 0xffff) || (InputTmp ==0x0000))
            MotorContorl.mcSpeedVSP = InputTmp >> 1;
}
#endif
```

8.4 Statement Selection

Replace the case statement with an if+else statement.

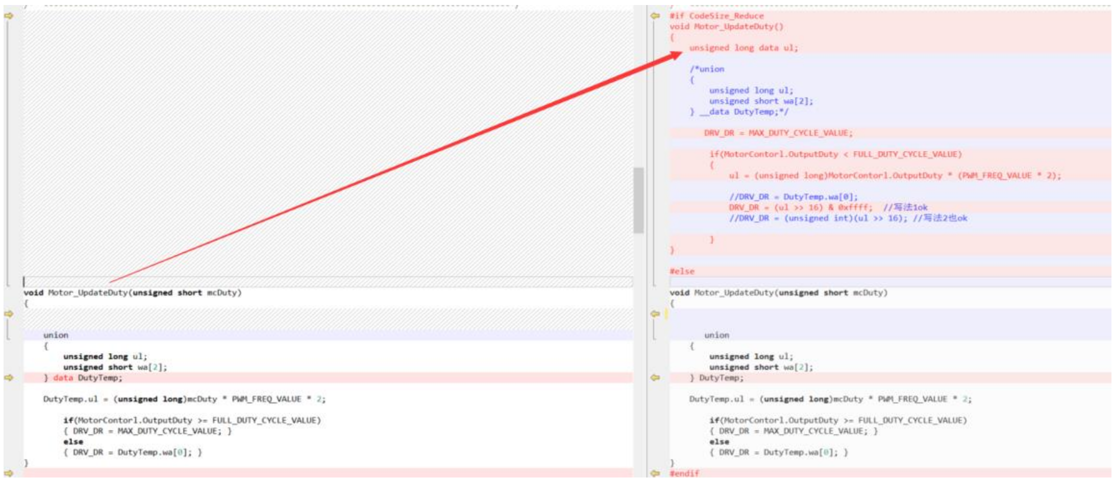


```
        case mcWait:

                if(MotorContorl.WaitTimeCnt > 1) { mcState = mcStart; }

                if (MotorContorl.FlagONOFF == 0)  { mcState = mcStop; }

                break;

    case mcStart:
        if (MotorContorl.StartSetFlag == 0)
        {
                        DRV_DR = DUTY_START_VALUE;  //更新启动占空比

                        TIM1_CR8 = ReadBit(TIM1_CR8, T1HallIN) ? 2 : 3;  //读取HALL,启动电机
                        MOEH = 1;   //Enable PWM output
                        MOEL = 1;

                        MotorContorl.StartSetFlag = 1;

        }

                if(MotorContorl.OneMsCnt > TIME_STA_TO_RUN)
        {
                        mcState = mcRun;

                        TIM1_CR1 = SOFT_SWITCH_FALL_VALUE;

                        MotorContorl.OneMsCnt = 0;
                }
    case mcRun:
        if (MotorContorl.FlagONOFF == 0)  { mcState = mcStop; }

        break;

    case mcStop:
```

```
else if (mcState == mcWait)
{

    if (MotorContorl.WaitTimeCnt > 1) { mcState = mcStart; }

    if (MotorContorl.FlagONOFF == 0) { mcState = mcStop; }

}
else if (mcState == mcStart || mcState == mcRun )
{
    if (mcState == mcStart )
    {
        if (MotorContorl.StartSetFlag == 0)
        {
            DRV_DR = DUTY_START_VALUE;  //更新启动占空比

            TIM1_CR8 = ReadBit(TIM1_CR8, T1HallIN) ? 2 : 3;  //读取HALL,启动电机
            MOEH = 1;   //Enable PWM output
            MOEL = 1;

            MotorContorl.StartSetFlag = 1;

        }

        if (MotorContorl.OneMsCnt > TIME_STA_TO_RUN)
        {
            mcState = mcRun;

            //                          TIM1_CR1 = SOFT_SWITCH_FALL_VALUE;

            MotorContorl.OneMsCnt = 0;
        }
    }

    if (MotorContorl.FlagONOFF == 0) { mcState = mcStop; }

}
else if (mcState == mcStop)
```

8.5 Unions

Replace the unions with local variables.

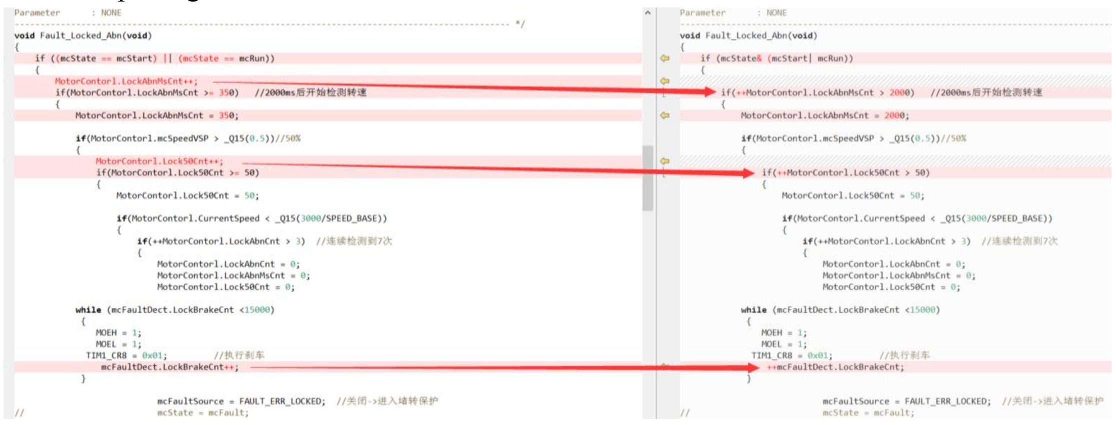


```
void Motor_UpdateDuty(unsigned short mcDuty)
{

    union
    {
        unsigned long ul;
        unsigned short wa[2];
    } data DutyTemp;

    DutyTemp.ul = (unsigned long)mcDuty * PWM_FREQ_VALUE * 2;

        if(MotorContorl.OutputDuty >= FULL_DUTY_CYCLE_VALUE)
        { DRV_DR = MAX_DUTY_CYCLE_VALUE; }
        else
        { DRV_DR = DutyTemp.wa[0]; }
}
```

```
#if CodeSize_Reduce
void Motor_UpdateDuty()
{
    unsigned long data ul;

    /*union
    {
        unsigned long ul;
        unsigned short wa[2];
    } __data DutyTemp;*/

       DRV_DR = MAX_DUTY_CYCLE_VALUE;

        if(MotorContorl.OutputDuty < FULL_DUTY_CYCLE_VALUE)
        {
            ul = (unsigned long)MotorContorl.OutputDuty * (PWM_FREQ_VALUE * 2);

            //DRV_DR = DutyTemp.wa[0];
            DRV_DR = (ul >> 16) & 0xffff;  //写法1ok
            //DRV_DR = (unsigned int)(ul >> 16); //写法2也ok

        }
}

#else
void Motor_UpdateDuty(unsigned short mcDuty)
{


       union
    {
        unsigned long ul;
        unsigned short wa[2];
    } DutyTemp;

    DutyTemp.ul = (unsigned long)mcDuty * PWM_FREQ_VALUE * 2;

        if(MotorContorl.OutputDuty >= FULL_DUTY_CYCLE_VALUE)
        { DRV_DR = MAX_DUTY_CYCLE_VALUE; }
        else
        { DRV_DR = DutyTemp.wa[0]; }
}
#endif
```

8.6 Logical Operators

Move the plus signs at the end to the front.

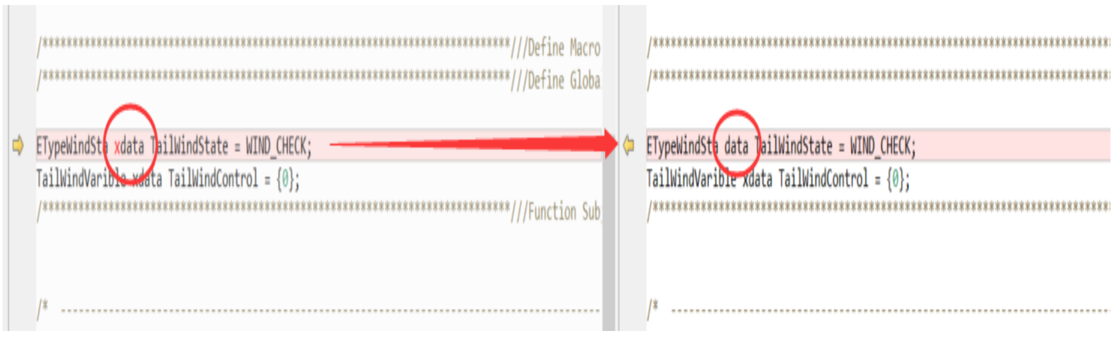


```
Parameter      : NONE
------------------------------------------------------------------------------------- */
void Fault_Locked_Abn(void)
{
    if ((mcState == mcStart) || (mcState == mcRun))
    {
        MotorContorl.LockAbnMsCnt++;
        if(MotorContorl.LockAbnMsCnt >= 350)   //2000ms后开始检测转速
        {
            MotorContorl.LockAbnMsCnt = 350;

            if(MotorContorl.mcSpeedVSP > _Q15(0.5))//50%
            {
                MotorContorl.Lock50Cnt++;
                if(MotorContorl.Lock50Cnt >= 50)
                {
                    MotorContorl.Lock50Cnt = 50;

                    if(MotorContorl.CurrentSpeed < _Q15(3000/SPEED_BASE))
                    {
                        if(++MotorContorl.LockAbnCnt > 3)  //连续检测到7次
                        {
                            MotorContorl.LockAbnCnt = 0;
                            MotorContorl.LockAbnMsCnt = 0;
                            MotorContorl.Lock50Cnt = 0;

            while (mcFaultDect.LockBrakeCnt <15000)
             {
                MOEH = 1;
                MOEL = 1;
              TIM1_CR8 = 0x01;          //执行刹车
                mcFaultDect.LockBrakeCnt++;
             }

                            mcFaultSource = FAULT_ERR_LOCKED;  //关闭->进入堵转保护
//                          mcState = mcFault;
```

```
Parameter      : NONE
------------------------------------------------------------------------------------
void Fault_Locked_Abn(void)
{
    if (mcState& (mcStart| mcRun))
    {

        if(++MotorContorl.LockAbnMsCnt > 2000)   //2000ms后开始检测转速
        {
            MotorContorl.LockAbnMsCnt = 2000;

            if(MotorContorl.mcSpeedVSP > _Q15(0.5))//50%
            {

                if(++MotorContorl.Lock50Cnt > 50)
                {
                    MotorContorl.Lock50Cnt = 50;

                    if(MotorContorl.CurrentSpeed < _Q15(3000/SPEED_BASE))
                    {
                        if(++MotorContorl.LockAbnCnt > 3)  //连续检测到7次
                        {
                            MotorContorl.LockAbnCnt = 0;
                            MotorContorl.LockAbnMsCnt = 0;
                            MotorContorl.Lock50Cnt = 0;

            while (mcFaultDect.LockBrakeCnt <15000)
             {
                MOEH = 1;
                MOEL = 1;
              TIM1_CR8 = 0x01;          //执行刹车
                ++mcFaultDect.LockBrakeCnt;
             }

                            mcFaultSource = FAULT_ERR_LOCKED;  //关闭->进入堵转保护
//                          mcState = mcFault;
```

8.7 Variable Storage Area

Change the variable storage area of enumeration type from xdata to data or idata. See the below figure.



```
/*************************************************************************///Define Macro
/*************************************************************************///Define Globa

ETypeWindSta xdata TailWindState = WIND_CHECK;
TailWindVarible xdata TailWindControl = {0};
/*************************************************************************///Function Sub


/* ----------------------------------------------------------------------------
```

```
/*****************************************************************************
/*****************************************************************************

ETypeWindSta data TailWindState = WIND_CHECK;
TailWindVarible xdata TailWindControl = {0};
/*****************************************************************************


/* ---------------------------------------------------------------------------
```

9 Revision History

Rev.	Descriptions	Date	Prepared By
V1.0	First release, translated from Chinese version 1.0.	2023/08/02	Eric Deng

Copyright Notice

Fortior Technology (Shenzhen) Co., Ltd.

Room 203, 2/F, Building No.11, Keji Central Road 2,

Software Park, High-Tech Industrial Park, Shenzhen, P.R. China 518057

Tel: 0755-26867710

Fax: 0755-26867715

URL: http://www.fortiortech.com

Contained herein