/\*BEGIN\_FILE\_HDR

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* NOTICE:

\* Copyright @ kingsignal.cn. All Rights Reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* FILE : main.c

\* Description :

\*

\* (Requirements, pseudo code and etc.)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Version Initials Date CR# Descriptions

\* 1.0 mufeng.li@kingsignal.cn 2020/10/12 N/A Initial Version

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*END\_FILE\_HDR\*/

#include <ti/sysbios/knl/Swi.h>

#include "esc\_can/EscCanDrv.h"

#include <shipant\_config.h>

/\*16 priority \*/

#define TASK\_STABILIZATION\_PRIORITY 15

#define TASK\_STABILIZATION\_DELAY 5

#define TASK\_STABILIZATIONDATREAD\_PRIORITY 14

#define TASK\_STABILIZATIONDATREAD\_DELAY 1/6

#define TASK\_COMMUNICATION\_PRIORITY 10

#define TASK\_COMMUNICATION\_DELAY 5

//天线状态管理任务优先级

#define TASK\_STATEMANAGEMENT\_PRIORITY 9

#define TASK\_STATEMANAGEMENT\_DELAY 20

#define TASK\_MONITOR\_FAST\_PRIORITY 5

#define TASK\_MONITOR\_FAST\_DELAY 20

#define TASK\_MONITOR\_SLOW\_PRIORITY 4

#define TASK\_MONITOR\_SLOW\_DELAY 50

#define TASK\_MAX 10

#define NUMPERSEC 10;

/\*用于统计CPU使用状况\*/

#define TASK\_ALL 0

#define TASK\_ID\_STATEMANAGEMENT 1

#define TASK\_ID\_STABILIZATION 2

#define TASK\_ID\_COMMUNICATION 3

#define TASK\_ID\_MONITOR\_SLOW 4

#define TASK\_ID\_STABILIZATIONDATREAD 8

#define TASK\_ID\_QEP 9

#define LENGTH\_RUNTIME 200

#define ATS01\_GPIO\_NULL 100

/\*目前空闲，可以兼容\*/

#define LAUNCHPAD\_C28379D\_BLUE\_LED\_GPIO 31

#define LAUNCHPAD\_C28379D\_RED\_LED\_GPIO 34

/\*只是为初始化GPIO的定义一个宏\*/

#define ATS01\_PUL\_FY\_GPIO 12

#define ATS01\_DIR\_FY\_GPIO 13

#define ATS01\_ENA\_FY\_GPIO 60

#define ATS01\_PUL\_JH\_GPIO 14

#define ATS01\_DIR\_JH\_GPIO 15

#define ATS01\_ENA\_JH\_GPIO 58

//限位

#define ATS01\_XW\_FWA\_GPIO 59

#define ATS01\_XW\_FWB\_GPIO 61

#define ATS01\_XW\_FYA\_GPIO 64

#define ATS01\_XW\_FYB\_GPIO 62

#define ATS01\_XW\_JH\_GPIO 65

//按键

#define ATS01\_KEYA\_GPIO 69

#define ATS01\_KEYB\_GPIO 78

#define ATS01\_LED\_HEARTBEAT\_GPIO 66

//gpio100不存在

#define ATS01\_LED\_STATE\_GPIO ATS01\_GPIO\_NULL

#define ATS01\_LED\_A\_GPIO 3

#define ATS01\_LED\_B\_GPIO 73

#define ATS01\_DVB\_RESET\_GPIO 41

//电磁阀

#define ATS01\_BRAKE\_GPIO 60

#define MOTOR\_BREAK\_OFF GPIO\_WritePin(ATS01\_BRAKE\_GPIO, 1)

#define MOTOR\_BREAK\_ON GPIO\_WritePin(ATS01\_BRAKE\_GPIO, 0)

#define ATS01\_BZ\_CLOSE 0

#define ATS01\_BZ\_OPEN 1

#define DEBUG\_PRINTF debug\_printf

//#else

//#define DEBUG\_PRINTF(\_X\_) do{;}while(0)

//#define DEBUG\_PRINTF(format, ...) do{;}while(0)

//#endif

extern ULong cpuload[TASK\_MAX];

extern FP64 run\_time[TASK\_MAX];

extern FP64 run\_time\_interval[TASK\_MAX];

extern FP64 run\_time\_stabilization[LENGTH\_RUNTIME];

extern xdc\_runtime\_Types\_FreqHz freq;

extern void debug\_printf(const char \*const fmt, ...);

extern void SetFlashParameter(INT16U nPara);

extern void SetOTPParameter(INT16U Para);

extern void init\_emif1();

/\*BEGIN\_FILE\_HDR

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* NOTICE:

\* Copyright @ atsing.cn. All Rights Reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* FILE : main.c

\* Description :

\*

\* (Requirements, pseudo code and etc.)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Version Initials Date CR# Descriptions

\* 1.0 hjdai@atsing.cn 2013/11/14 N/A Initial Version

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*END\_FILE\_HDR\*/

/\* ----------------------------------- XDC module Headers \*/

#include <xdc/std.h>

#include <xdc/runtime/Log.h>

#include <xdc/runtime/Types.h>

#include <xdc/runtime/Diags.h>

#include <xdc/runtime/System.h>

#include <xdc/runtime/Error.h>

#include <xdc/runtime/Timestamp.h>

//#define DSP28\_DATA\_TYPES

#include "F28x\_Project.h" // DSP2833x Headerfile Include File

/\* ----------------------------------- BIOS module Headers \*/

#include <ti/sysbios/BIOS.h>

#include <ti/sysbios/knl/Semaphore.h>

#include <ti/sysbios/knl/Swi.h>

#include <xdc/cfg/global.h>

#include <ti/sysbios/knl/Task.h>

#include <ti/sysbios/knl/Event.h>

#include <ti/sysbios/knl/Queue.h>

#include <ti/sysbios/knl/Clock.h>

#include <ti/sysbios/hal/Hwi.h>

#include <ti/sysbios/family/c28/Timer.h>

#include <ti/sysbios/knl/Swi.h>

#include <string.h>

#include <stdarg.h>

#include "SpiExt.h"

#include "I2CExt.h"

#include "SciExt.h"

#include "AdcExt.h"

#include "TimerExt.h"

#include "PwmExt.h"

#include "EepromExt.h"

#include "MonitorExt.h"

#include "StateManagementExt.h"

#include "CommunicationExt.h"

#include "qep/QEPAppExt.h"

#include "main.h"

#include "task\_stabilization/task\_stabilization.h"

#include "stepper\_motor\_driver.h"

#include "esc\_can/EscCanDrv.h"

#ifdef \_\_cplusplus

#pragma DATA\_SECTION("BlockBufferFile")

#else

#pragma DATA\_SECTION(run\_time\_stabilization,"BlockBufferFile");

#endif

ULong cpuload[TASK\_MAX];

FP64 run\_time[TASK\_MAX];

FP64 run\_time\_interval[TASK\_MAX];

FP64 run\_time\_stabilization[LENGTH\_RUNTIME];

xdc\_runtime\_Types\_FreqHz freq;

static void Gpio\_Init();

void SystemInit();

Void cpuLoadInit(Void);

Void main()

{

Error\_Block eb;

Error\_init(&eb);

SystemInit();

/\* Set up the Hwi, Swi, and Task thread load tables \*/

cpuLoadInit();

Timestamp\_getFreq(&freq);

/\*Dead Loop\*/

Task\_Params taskParams;

Task\_Params\_init(&taskParams);

////////////////////////////////////////////////////

//任务初始化

////////////////////////////////////////////////////

task\_StateManagementInit();

// 20ms & priority 9

taskParams.priority = TASK\_STATEMANAGEMENT\_PRIORITY;

taskParams.stackSize = 3072;

g\_AntCtrlIntBlock.m\_task\_statemanagement = Task\_create(task\_StateManagement, &taskParams, NULL);

if (g\_AntCtrlIntBlock.m\_task\_statemanagement == NULL)

{

System\_abort("Task create failed");

}

task\_stabilization\_init();

Clock\_Params clockParams;

Clock\_Params\_init(&clockParams);

clockParams.period = TASK\_STABILIZATION\_DELAY;

clockParams.startFlag = FALSE;

clockParams.arg = (UArg)0x0;

g\_AntCtrlIntBlock.m\_task\_stabilization = Clock\_create(task\_stabilization\_control, 1, &clockParams, &eb);

if ( g\_AntCtrlIntBlock.m\_task\_stabilization == NULL) {

System\_abort("Clock create failed");

}

Clock\_start( g\_AntCtrlIntBlock.m\_task\_stabilization);

// 20ms & priority 10

task\_CommunicationInit();

taskParams.priority = TASK\_COMMUNICATION\_PRIORITY;

taskParams.stackSize = 2048;

g\_AntCtrlIntBlock.m\_task\_communication = Task\_create(task\_Communication, &taskParams, NULL);

if (g\_AntCtrlIntBlock.m\_task\_communication == NULL)

{

System\_abort("Task create failed");

}

task\_MonitorInit();

//50ms & priority 4

taskParams.priority = TASK\_MONITOR\_SLOW\_PRIORITY;

taskParams.stackSize = 2048;

g\_AntCtrlIntBlock.m\_taskMonitorSlowControlBlock = Task\_create(task\_MonitorSlow, &taskParams, NULL);

if (g\_AntCtrlIntBlock.m\_taskMonitorSlowControlBlock == NULL)

{

System\_abort("Task create failed");

}

//看门狗初始化

wakeint\_init();

BIOS\_start();

}

void SystemInit()

{

//开启外设时钟

//

// Step 1. Initialize System Control:

// PLL, WatchDog, enable Peripheral Clocks

// This example function is found in the F2837xS\_SysCtrl.c file.

//

InitSysCtrl();

DELAY\_US(200000); //2000000

InitGpio();

Gpio\_Init();

watchdog\_service\_flag = 1;

////////////////////////////////////////////////////

//外设初始化

////////////////////////////////////////////////////

SciDriver\_Init();

I2CA\_Init();

//码盘计数初始化

// QepDriver\_Init();

AdcDriver\_Init();

timer\_init();

//PWM溢出中断，1/6ms中断一次

Pwm\_Init(NULL, NULL);

//初始化zone6 无外扩ram

//init\_emif1();

EscCan\_init();

}

// FILE: emif1\_16bit\_asram.c

// TITLE: EMIF1 module accessing 16bit ASRAM.

extern void setup\_emif1\_pinmux\_async\_16bit(Uint16);

//

// mem\_read\_write - This function performs simple read/write word accesses

// to memory.

//

char

mem\_read\_write(Uint32 start\_addr, Uint32 mem\_size)

{

unsigned short mem\_rds;

unsigned short mem\_wds;

unsigned short \*XMEM\_ps;

unsigned int i;

//

//Write data

//

XMEM\_ps = (unsigned short \*)start\_addr;

//

//Fill memory

//

mem\_wds = 0x0123;

for (i=0; i < mem\_size; i++)

{

\*XMEM\_ps++ = mem\_wds;

mem\_wds += 0x1111;

}

//

//Verify memory

//

mem\_wds = 0x0123;

XMEM\_ps = (unsigned short \*)start\_addr;

for (i=0; i < mem\_size; i++)

{

mem\_rds = \*XMEM\_ps;

if( mem\_rds != mem\_wds)

{

return(1);

}

XMEM\_ps++;

mem\_wds += 0x1111;

}

return(0);

}

#define ASRAM\_CS2\_START\_ADDR 0x100000

#define ASRAM\_CS2\_SIZE 0x8000

void init\_emif1(void)

{

Uint16 ErrCount = 0;

char ErrCount\_local;

//

//Configure to run EMIF1 on full Rate (EMIF1CLK = CPU1SYSCLK)

//

EALLOW;

ClkCfgRegs.PERCLKDIVSEL.bit.EMIF1CLKDIV = 0x1;

EDIS;

EALLOW;

//

//Disable Access Protection (CPU\_FETCH/CPU\_WR/DMA\_WR)

//

Emif1ConfigRegs.EMIF1ACCPROT0.all = 0x0;

if(Emif1ConfigRegs.EMIF1ACCPROT0.all != 0x0)

{

ErrCount++;

}

//

// Commit the configuration related to protection. Till this bit remains set

// content of EMIF1ACCPROT0 register can't be changed.

//

Emif1ConfigRegs.EMIF1COMMIT.all = 0x1;

if(Emif1ConfigRegs.EMIF1COMMIT.all != 0x1)

{

ErrCount++;

}

//

// Lock the configuration so that EMIF1COMMIT register can't be

// changed any more.

//

Emif1ConfigRegs.EMIF1LOCK.all = 0x1;

if(Emif1ConfigRegs.EMIF1LOCK.all != 1)

{

ErrCount++;

}

EDIS;

//

//Configure GPIO pins for EMIF1

//

setup\_emif1\_pinmux\_async\_16bit(0);

//

//Configure the access timing for CS2 space

//

Emif1Regs.ASYNC\_CS2\_CR.all = (EMIF\_ASYNC\_ASIZE\_16 | // 16Bit Memory

// Interface

EMIF\_ASYNC\_TA\_4 | // Turn Around time

// of 2 Emif Clock

EMIF\_ASYNC\_RHOLD\_2 | // Read Hold time

// of 1 Emif Clock

EMIF\_ASYNC\_RSTROBE\_7 | // Read Strobe time

// of 4 Emif Clock

EMIF\_ASYNC\_RSETUP\_2 | // Read Setup time

// of 1 Emif Clock

EMIF\_ASYNC\_WHOLD\_2 | // Write Hold time

// of 1 Emif Clock

EMIF\_ASYNC\_WSTROBE\_7 | // Write Strobe time

// of 1 Emif Clock

EMIF\_ASYNC\_WSETUP\_2 | // Write Setup time

// of 1 Emif Clock

EMIF\_ASYNC\_EW\_DISABLE | // Extended Wait

// Disable.

EMIF\_ASYNC\_SS\_DISABLE // Strobe Select Mode

// Disable.

);

//

//Check basic RD/WR access to CS2 space

//

ErrCount\_local = mem\_read\_write(ASRAM\_CS2\_START\_ADDR, ASRAM\_CS2\_SIZE);

ErrCount = ErrCount + ErrCount\_local;

if(ErrCount == 0)

{

DEBUG\_PRINTF("EMIF normal\r\n");

}

else

{

DEBUG\_PRINTF("EMIF error \r\n");

}

}

void Gpio\_Init()

{

//初始化步进电机

GPIO\_SetupPinMux(ATS01\_PUL\_FY\_GPIO, GPIO\_MUX\_CPU1, 0);

GPIO\_SetupPinOptions(ATS01\_PUL\_FY\_GPIO, GPIO\_OUTPUT, GPIO\_PUSHPULL);

GPIO\_SetupPinMux(ATS01\_DIR\_FY\_GPIO, GPIO\_MUX\_CPU1, 0);

GPIO\_SetupPinOptions(ATS01\_DIR\_FY\_GPIO, GPIO\_OUTPUT, GPIO\_PUSHPULL);

GPIO\_SetupPinMux(ATS01\_ENA\_FY\_GPIO, GPIO\_MUX\_CPU1, 0);

GPIO\_SetupPinOptions(ATS01\_ENA\_FY\_GPIO, GPIO\_OUTPUT, GPIO\_PUSHPULL);

GPIO\_SetupPinMux(ATS01\_PUL\_JH\_GPIO, GPIO\_MUX\_CPU1, 0);

GPIO\_SetupPinOptions(ATS01\_PUL\_JH\_GPIO, GPIO\_OUTPUT, GPIO\_PUSHPULL);

GPIO\_SetupPinMux(ATS01\_DIR\_JH\_GPIO, GPIO\_MUX\_CPU1, 0);

GPIO\_SetupPinOptions(ATS01\_DIR\_JH\_GPIO, GPIO\_OUTPUT, GPIO\_PUSHPULL);

GPIO\_SetupPinMux(ATS01\_ENA\_JH\_GPIO, GPIO\_MUX\_CPU1, 0);

GPIO\_SetupPinOptions(ATS01\_ENA\_JH\_GPIO, GPIO\_OUTPUT, GPIO\_PUSHPULL);

//初始化限位

GPIO\_SetupPinMux(ATS01\_XW\_FWA\_GPIO, GPIO\_MUX\_CPU1, 0);

GPIO\_SetupPinOptions(ATS01\_XW\_FWA\_GPIO, GPIO\_INPUT, GPIO\_PUSHPULL);

GPIO\_SetupPinMux(ATS01\_XW\_FWB\_GPIO, GPIO\_MUX\_CPU1, 0);

GPIO\_SetupPinOptions(ATS01\_XW\_FWB\_GPIO, GPIO\_INPUT, GPIO\_PUSHPULL);

GPIO\_SetupPinMux(ATS01\_XW\_FYA\_GPIO, GPIO\_MUX\_CPU1, 0);

GPIO\_SetupPinOptions(ATS01\_XW\_FYA\_GPIO, GPIO\_INPUT, GPIO\_PUSHPULL);

GPIO\_SetupPinMux(ATS01\_XW\_FYB\_GPIO, GPIO\_MUX\_CPU1, 0);

GPIO\_SetupPinOptions(ATS01\_XW\_FYB\_GPIO, GPIO\_INPUT, GPIO\_PUSHPULL);

GPIO\_SetupPinMux(ATS01\_XW\_JH\_GPIO, GPIO\_MUX\_CPU1, 0);

GPIO\_SetupPinOptions(ATS01\_XW\_JH\_GPIO, GPIO\_INPUT, GPIO\_PUSHPULL);

//按键

GPIO\_SetupPinMux(ATS01\_KEYA\_GPIO, GPIO\_MUX\_CPU1, 0);

GPIO\_SetupPinOptions(ATS01\_KEYA\_GPIO, GPIO\_INPUT, GPIO\_PUSHPULL);

GPIO\_SetupPinMux(ATS01\_KEYB\_GPIO, GPIO\_MUX\_CPU1, 0);

GPIO\_SetupPinOptions(ATS01\_KEYB\_GPIO, GPIO\_INPUT, GPIO\_PUSHPULL);

//LED

GPIO\_SetupPinMux(ATS01\_LED\_HEARTBEAT\_GPIO, GPIO\_MUX\_CPU1, 0);

GPIO\_SetupPinOptions(ATS01\_LED\_HEARTBEAT\_GPIO, GPIO\_OUTPUT, GPIO\_PUSHPULL);

GPIO\_SetupPinMux(ATS01\_LED\_STATE\_GPIO, GPIO\_MUX\_CPU1, 0);

GPIO\_SetupPinOptions(ATS01\_LED\_STATE\_GPIO, GPIO\_OUTPUT, GPIO\_PUSHPULL);

GPIO\_SetupPinMux(ATS01\_LED\_A\_GPIO, GPIO\_MUX\_CPU1, 0);

GPIO\_SetupPinOptions(ATS01\_LED\_A\_GPIO, GPIO\_OUTPUT, GPIO\_PUSHPULL);

GPIO\_SetupPinMux(ATS01\_LED\_B\_GPIO, GPIO\_MUX\_CPU1, 0);

GPIO\_SetupPinOptions(ATS01\_LED\_B\_GPIO, GPIO\_OUTPUT, GPIO\_PUSHPULL);

//DVB

GPIO\_SetupPinMux(ATS01\_DVB\_RESET\_GPIO, GPIO\_MUX\_CPU1, 0);

GPIO\_SetupPinOptions(ATS01\_DVB\_RESET\_GPIO, GPIO\_OUTPUT, GPIO\_PUSHPULL);

//BREAK

GPIO\_SetupPinMux(ATS01\_BRAKE\_GPIO, GPIO\_MUX\_CPU1, 0);

GPIO\_SetupPinOptions(ATS01\_BRAKE\_GPIO, GPIO\_OUTPUT, GPIO\_OPENDRAIN);

GPIO\_WritePin(ATS01\_BRAKE\_GPIO, 0);

}

Void cpuLoadInit(Void)

{

Types\_FreqHz freq;

ULong maxLoad;

/\* freq is maximum timestamp counts per second (100% cpuload) \*/

Timestamp\_getFreq(&freq);

maxLoad = freq.lo / NUMPERSEC; /\* since we run load NUMPERSEC times \*/

memset(cpuload,0,TASK\_MAX\*sizeof(INT32U));

cpuload[TASK\_MAX-1] = maxLoad;

}

/\*==================================================================

\* Function : debug\_printf

\* Description :

\* Input Para :

\* Output Para :

\* Return Value:

==================================================================\*/

static char line[128];

void debug\_printf(const char \*const fmt, ...)

{

va\_list Argument;

INT8U motor\_mode = get\_moto\_mode(MOTOR\_AZ);

INT8U motor\_mode\_pol = get\_moto\_mode(MOTOR\_POL);

Hwi\_disable();

Hwi\_enableInterrupt(DCANA\_1\_INTERRUPT);

set\_moto\_mode(MOTOR\_1234, FALSE);

va\_start(Argument, fmt);

#ifdef DEBUG\_PRINTF\_CTRL

System\_vprintf((xdc\_String)fmt, Argument);

#endif

vsprintf(line, fmt, Argument);

strcat(m\_gSndCommDbgBuffer, line);

System\_flush();

va\_end(Argument);

set\_moto\_mode(MOTOR\_123, motor\_mode);

set\_moto\_mode(MOTOR\_POL, motor\_mode\_pol);

Hwi\_enable();

}

/\*BEGIN\_FILE\_HDR

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* NOTICE:

\* Copyright @ atsing.cn. All Rights Reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* FILE : AdcAppExt.h

\* Description :

\*

\* (Requirements, pseudo code and etc.)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Version Initials Date CR# Descriptions

\* 1.0 hjdai@atsing.cn 2013/12/02 N/A Initial Version

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*END\_FILE\_HDR\*/

#ifndef ADCAPPEXT\_H

#define ADCAPPEXT\_H

/\*Include files\*/

#include "atsingdef.h"

#include "F28x\_Project.h"

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* ALL definition and declaration CAN be used ,but can NOT revised \*

\* outside this driver \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* macro definition \*/

#define ADC\_CHANNEL\_CNT 16

/\* structure definiton \*/

/\* variable definition \*/

extern INT16U adc\_result[ADC\_CHANNEL\_CNT];

/\* function declaration \*/

//ADC驱动初始化

extern void AdcDriver\_Init(void);

//ADC启动

extern void AdcDriver\_Start(void);

extern Uint16 adca\_dri\_get\_value(char flag);

extern void AdcDriver\_ValueGet(void);

#endif /\* end of header file \*/

/\*BEGIN\_FILE\_HDR

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* NOTICE:

\* Copyright (c) 2013 by @ atsing.cn. All Rights Reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* FILE : ADCCALCAppExt.h

\* Description :

\*

\* (Requirements, pseudo code and etc.)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Version Initials Date CR# Descriptions

\* 1.0 hjdai@atsing.cn 2013/02/07 N/A Initial Version

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*END\_FILE\_HDR\*/

#ifndef ADCCALCAPPEXT\_H

#define ADCCALCAPPEXT\_H

/\*Include files\*/

#include "atsingdef.h"

#include "../FilterAlgorithm/FilterAlgorithmDrv.h"

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* ALL definition and declaration CAN be used ,but can NOT revised \*

\* outside this driver \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* macro definition \*/

/\* structure definiton \*/

typedef enum

{

ADC\_OK,

ADC\_UPEXCEED,

ADC\_DOWNEXCEED,

ADC\_NOCHANGE

}e\_ADCCalc\_err;

typedef struct str\_ADCCalc

{

FP64 value\_zero; /\*parameters, ad zero point\*/

FP64 p\_ad2physical; /\*parameters, proportion, k\*/

INT16S limit\_max; /\*parameters, sample value normal limit\*/

INT16S limit\_min;

INT16S value\_adc; /\*input, ad sample value\*/

FP64 value\_off; /\*internal\*/

FP64 value\_sat; /\*internal\*/

FP64 value; /\*output \*/

FP64 value\_last;

e\_ADCCalc\_err status; /\*output \*/

t\_DJitterFilterStr dfilter; /\*internal\*/

}t\_ADCCalc;

/\* variable definition \*/

/\* function declaration \*/

extern void ADCCalc\_init(t\_ADCCalc \*p, INT16U cnt);

extern void ADCCalc\_calc(t\_ADCCalc \*p, INT16S aNewDat);

#endif /\* end of header file \*/

/\*BEGIN\_FILE\_HDR

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* NOTICE:

\* Copyright (c) 2013 by @ atsing.cn. All Rights Reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* FILE : ADCCALCCfgExt.h

\* Description :

\*

\* (Requirements, pseudo code and etc.)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Version Initials Date CR# Descriptions

\* 1.0 hjdai@atsing.cn 2013-7-16 N/A Initial Version

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*END\_FILE\_HDR\*/

#ifndef ADCCALCCFGEXT\_H

#define ADCCALCCFGEXT\_H

/\*Include files\*/

#include "atsingdef.h"

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* ALL definition and declaration CAN be used and revised outside this driver \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* macro definition \*/

#define ADCCALC\_DFILTERCNT 10

#endif /\* end of header file \*/

/\*BEGIN\_FILE\_HDR

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* NOTICE:

\* Copyright (c) 2013 by @ atsing.cn. All Rights Reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* FILE : ADCCALCDrv.h

\* Description :

\*

\* (Requirements, pseudo code and etc.)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Version Initials Date CR# Descriptions

\* 1.0 hjdai@atsing.cn 2013/07/17 N/A Initial Version

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*END\_FILE\_HDR\*/

#ifndef ADCCALCDRV\_H

#define ADCCALCDRV\_H

/\*Include files\*/

#include "ADCCALCExt.h"

/\*\*\*\*following definitions CAN NOT be used outside this driver\*\*\*\*\*\*/

/\* structure definiton \*/

/\* macro definition \*/

#ifndef FA\_CFG\_DSWFILTER\_EN

#define FA\_CFG\_DSWFILTER\_EN

#endif

#endif /\* end of header file \*/

/\*BEGIN\_FILE\_HDR

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* NOTICE:

\* Copyright (c) 2013 by @ atsing.cn. All Rights Reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* FILE : ADCCALCExt.h

\* Description :

\*

\* (Requirements, pseudo code and etc.)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Version Initials Date CR# Descriptions

\* 1.0 hjdai@atsing.cn 2013/02/07 N/A Initial Version

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*END\_FILE\_HDR\*/

#ifndef ADCCALCEXT\_H

#define ADCCALCEXT\_H

/\*Include files\*/

#include "ADCCALCHWExt.h"

#include "ADCCALCAppExt.h"

#include "ADCCALCCfgExt.h"

#endif /\* end of header file \*/

/\*BEGIN\_FILE\_HDR

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* NOTICE:

\* Copyright (c) 2013 by @ atsing.cn. All Rights Reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* FILE : ADCCALCHWExt.h

\* Description :

\*

\* (Requirements, pseudo code and etc.)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Version Initials Date CR# Descriptions

\* 1.0 hjdai@atsing.cn 2013/02/07 N/A Initial Version

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*END\_FILE\_HDR\*/

#ifndef ADCCALCHWEXT\_H

#define ADCCALCHWEXT\_H

/\*Include files\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This file include defination related with MCU \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* macro definition \*/

#endif /\* end of header file \*/

/\*BEGIN\_FILE\_HDR

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* NOTICE:

\* Copyright (c) 2013 by @ atsing.cn. All Rights Reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* FILE : XXXCfgExt.h

\* Description :

\*

\* (Requirements, pseudo code and etc.)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Version Initials Date CR# Descriptions

\* 1.0 ygfeng@atsing.cn 2013/06/14 N/A Initial Version

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*END\_FILE\_HDR\*/

#ifndef ADCCFGEXT\_H

#define ADCCFGEXT\_H

/\*Include files\*/

//#include "atsingdef.h"

//#include " "

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* ALL definition and declaration CAN be used and revised outside this driver \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* macro definition \*/

#endif /\* end of header file \*/

/\*BEGIN\_FILE\_HDR

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* NOTICE:

\* Copyright (c) 2013 by @ atsing.cn. All Rights Reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* FILE : ADCCALCDrv.c

\* Description :

\*

\* (Requirements, pseudo code and etc.)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Version Initials Date CR# Descriptions

\* 1.0 hjdai@atsing.cn 2013/02/07 N/A Initial Version

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*END\_FILE\_HDR\*/

#ifndef ADCCALCDRV\_C

#define ADCCALCDRV\_C

#endif

/\* include files \*/

#include "ADCCALCDrv.h"

/\* static variable declaration \*/

/\* static function declaration \*/

/\* static function definition \*/

/\* public function declaration \*/

/\*==================================================================

\* Function : ADCCalc\_init

\* Description : 函数初始化

\* Input Para :

\* Output Para :

\* Return Value:

==================================================================\*/

void ADCCalc\_init(t\_ADCCalc \*p, INT16U cnt)

{

DJitterFilterInit(DSW\_FILTER\_MODE\_DIRECT, &p->dfilter, FALSE, ADCCALC\_DFILTERCNT);

}

/\*==================================================================

\* Function : ADCCalc\_calc

\* Description :

\* Input Para :

\* Output Para :

\* Return Value:

==================================================================\*/

void ADCCalc\_calc(t\_ADCCalc \*p, INT16S aNewDat)

{

p->value\_adc = aNewDat;

/\*sat protect\*/

if(p->value\_adc > p->limit\_max)

{

p->value\_sat = p->limit\_max;

p->status = ADC\_UPEXCEED;

}

else if(p->value\_adc < p->limit\_min)

{

p->value\_sat = p->limit\_min;

p->status = ADC\_DOWNEXCEED;

}

else

{

p->value\_sat = p->value\_adc;

p->status = ADC\_OK;

}

p->value\_off = p->value\_sat - p->value\_zero;

DJitterFilter(aNewDat, &p->dfilter);

if(DSW\_FILTER\_CNT\_RESET == p->dfilter.m\_Counter)

{

p->status = ADC\_NOCHANGE;

}

p->value = p->value\_off \* p->p\_ad2physical;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*BEGIN\_FILE\_HDR

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* NOTICE:

\* Copyright (c) 2013 by @ atsing.cn. All Rights Reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* FILE : XXXHWExt.h

\* Description :

\*

\* (Requirements, pseudo code and etc.)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Version Initials Date CR# Descriptions

\* 1.0 ygfeng@atsing.cn 2013/06/14 N/A Initial Version

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*END\_FILE\_HDR\*/

#ifndef ADCHWEXT\_H

#define ADCHWEXT\_H

/\*Include files\*/

//#include " "

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This file include defination related with MCU \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* macro definition \*/

#define ADC\_CHANNEL\_COUNT 3u

/\*J1.38 ADCINB4 B4 信标接收机AD

J1.39 ADCINB7 b7 DVB

J1.40 ADCINB6 b6 电源5v

\*/

#define ADC\_CHANNEL\_BEACON\_AGC 0x0c

#define ADC\_CHANNEL\_DVB\_AGC 0x0f

#define ADC\_CHANNEL\_POWER\_VOLTAGE\_5V 0x0e

#endif /\* end of header file \*/

/\*BEGIN\_FILE\_HDR

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* NOTICE:

\* Copyright (c) 2013 by @ atsing.cn. All Rights Reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* FILE : AdcExt.h

\* Description :

\*

\* (Requirements, pseudo code and etc.)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Version Initials Date CR# Descriptions

\* 1.0 ygfeng@atsing.cn 2013/06/14 N/A Initial Version

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*END\_FILE\_HDR\*/

#ifndef ADCEXT\_H

#define ADCEXT\_H

/\*Include files\*/

#include "../ADC/AdcHWExt.h"

#include "../ADC/AdcAppExt.h"

#include "../ADC/AdcCfgExt.h"

#endif /\* end of header file \*/

/\*BEGIN\_FILE\_HDR

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* NOTICE:

\* Copyright (c) 2013 by @ atsing.cn. All Rights Reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* FILE : XXXDrv.h

\* Description :

\*

\* (Requirements, pseudo code and etc.)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Version Initials Date CR# Descriptions

\* 1.0 ygfeng@atsing.cn 2013/06/14 N/A Initial Version

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*END\_FILE\_HDR\*/

#ifndef ADCDRV\_H

#define ADCDRV\_H

#include "F28x\_Project.h"

/\*\*\*\*following definitions CAN NOT be used outside this driver\*\*\*\*\*\*/

/\* structure definiton \*/

/\* macro definition \*/

// ADC start parameters

#if (CPU\_FRQ\_150MHZ) // Default - 150 MHz SYSCLKOUT

#define ADC\_MODCLK 0x3 // HSPCLK = SYSCLKOUT/2\*ADC\_MODCLK2 = 150/(2\*3) = 25.0 MHz

#endif

#if (CPU\_FRQ\_100MHZ)

#define ADC\_MODCLK 0x2 // HSPCLK = SYSCLKOUT/2\*ADC\_MODCLK2 = 100/(2\*2) = 25.0 MHz

#endif

#define ADC\_CKPS 0x1 // ADC module clock = HSPCLK/2\*ADC\_CKPS = 25.0MHz/(1\*2) = 12.5MHz

#define ADC\_SHCLK 0xf // S/H width in ADC module periods = 16 ADC clocks

//#define SYSBIOS 1

#define CPU\_ADCa\_1\_INTERRUPT 32

#endif /\* end of header file \*/

/\*BEGIN\_FILE\_HDR

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* NOTICE:

\* Copyright (c) 2013 by @ atsing.cn. All Rights Reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* FILE : AdcDrv.c

\* Description :

\*

\* (Requirements, pseudo code and etc.)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Version Initials Date CR# Descriptions

\* 1.0 ygfeng@atsing.cn 2013/06/14 N/A Initial Version

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*END\_FILE\_HDR\*/

#ifndef ADCDRV\_C

#define ADCDRV\_C

#endif

/\* include files \*/

#include "AdcDrv.h"

#include "AdcExt.h"

#if (SYSBIOS)

#include <xdc/runtime/System.h>

#include <xdc/runtime/Error.h>

#include <ti/sysbios/hal/Hwi.h>

#include <ti/sysbios/knl/Swi.h>

#include <atsingdef.h>

#include <ti/sysbios/family/c28/Hwi.h>

//#define DSP28\_DATA\_TYPES

#endif

/\* static variable declaration \*/

//static

/\* static function declaration \*/

// Prototype statements for functions found within this file.

interrupt void adc\_isr(void);

INT16U adc\_result[16];

INT16U adc\_result\_new = FALSE;

/\* static function definition \*/

//

// ConfigureADC - Write ADC configurations and power up the ADC for both

// ADC A and ADC B

//

void ConfigureADC(void)

{

EALLOW;

//

//write configurations

//

AdcaRegs.ADCCTL2.bit.PRESCALE = 6; //set ADCCLK divider to /4

// AdcbRegs.ADCCTL2.bit.PRESCALE = 6; //set ADCCLK divider to /4

AdcSetMode(ADC\_ADCA, ADC\_RESOLUTION\_12BIT, ADC\_SIGNALMODE\_SINGLE);

// AdcSetMode(ADC\_ADCB, ADC\_RESOLUTION\_12BIT, ADC\_SIGNALMODE\_SINGLE);

//

//Set pulse positions to late

//

AdcaRegs.ADCCTL1.bit.INTPULSEPOS = 1;

// AdcbRegs.ADCCTL1.bit.INTPULSEPOS = 1;

//

//power up the ADCs

//

AdcaRegs.ADCCTL1.bit.ADCPWDNZ = 1;

// AdcbRegs.ADCCTL1.bit.ADCPWDNZ = 1;

//

//delay for 1ms to allow ADC time to power up

//

DELAY\_US(1000);

EDIS;

}

//

// SetupADCSoftware - Setup ADC channels and acquisition window

//

void SetupADCSoftware(void)

{

Uint16 acqps;

Uint16 tempsensor\_acqps;

tempsensor\_acqps = 139; //temperature sensor needs at least 700ns

//acquisition time

//

//determine minimum acquisition window (in SYSCLKS) based on resolution

//

if(ADC\_RESOLUTION\_12BIT == AdcaRegs.ADCCTL2.bit.RESOLUTION)

{

acqps = 14; //75ns

}

else //resolution is 16-bit

{

acqps = 63; //320ns

}

//

//Select the channels to convert and end of conversion flag

//ADCA

//

EALLOW;

AdcaRegs.ADCSOC0CTL.bit.CHSEL = 0; //SOC0 will convert pin A0

AdcaRegs.ADCSOC0CTL.bit.ACQPS = acqps; //sample window is acqps +

//1 SYSCLK cycles

AdcaRegs.ADCSOC1CTL.bit.CHSEL = 1; //SOC1 will convert pin A1

AdcaRegs.ADCSOC1CTL.bit.ACQPS = acqps; //sample window is acqps +

//1 SYSCLK cycles

AdcaRegs.ADCSOC2CTL.bit.CHSEL = 2; //SOC1 will convert pin A2

AdcaRegs.ADCSOC2CTL.bit.ACQPS = acqps; //sample window is acqps +

//1 SYSCLK cycles

AdcaRegs.ADCSOC3CTL.bit.CHSEL = 13; //SOC0 will convert internal //connection A13 3; // //SOC1 will convert pin A3

AdcaRegs.ADCSOC3CTL.bit.ACQPS = tempsensor\_acqps; //acqps; //sample window is acqps +

//1 SYSCLK cycles

AdcaRegs.ADCSOC4CTL.bit.CHSEL = 5; //SOC1 will convert pin A2

AdcaRegs.ADCSOC4CTL.bit.ACQPS = acqps; //sample window is acqps +

//1 SYSCLK cycles

AdcaRegs.ADCSOC5CTL.bit.CHSEL = 5; //SOC1 will convert pin A2

AdcaRegs.ADCSOC5CTL.bit.ACQPS = acqps; //sample window is acqps +

//1 SYSCLK cycles

AdcaRegs.ADCINTSEL1N2.bit.INT1SEL = 2; //end of SOC1 will set INT1 flag

AdcaRegs.ADCINTSEL1N2.bit.INT1E = 1; //enable INT1 flag

AdcaRegs.ADCINTFLGCLR.bit.ADCINT1 = 1; //make sure INT1 flag is cleared

/\*

//ADCB

AdcbRegs.ADCSOC0CTL.bit.CHSEL = 0; //SOC0 will convert pin B0

AdcbRegs.ADCSOC0CTL.bit.ACQPS = acqps; //sample window is acqps +

//1 SYSCLK cycles

AdcbRegs.ADCSOC1CTL.bit.CHSEL = 1; //SOC1 will convert pin B1

AdcbRegs.ADCSOC1CTL.bit.ACQPS = acqps; //sample window is acqps +

//1 SYSCLK cycles

AdcbRegs.ADCINTSEL1N2.bit.INT1SEL = 1; //end of SOC1 will set INT1 flag

AdcbRegs.ADCINTSEL1N2.bit.INT1E = 1; //enable INT1 flag

AdcbRegs.ADCINTFLGCLR.bit.ADCINT1 = 1; //make sure INT1 flag is cleared

\*/

EDIS;

}

/\* public function declaration \*/

/\*==================================================================

\* Function : ADC驱动初始化

\* Description :

\* Input Para :

\* Output Para :

\* Return Value:

==================================================================\*/

void AdcDriver\_Init()

{

ConfigureADC();

//

// Initialize the temperature sensor

// Note: The argument needs to change if using a VREFHI voltage other than 3.0V

//

InitTempSensor(3.0);

#if (SYSBIOS)

Hwi\_Handle myHwi;

Error\_Block eb;

Error\_init(&eb);

myHwi = Hwi\_create(CPU\_ADCa\_1\_INTERRUPT, adc\_isr, NULL, &eb);

if (myHwi == NULL)

{

System\_abort("Hwi create failed");

}

// Hwi\_disableInterrupt(CPU\_TIMER\_1\_INTERRUPT);

#else

EALLOW; // This is needed to write to EALLOW protected registers

PieVectTable.ADCA1\_INT = &adc\_isr;

EDIS; // This is needed to disable write to EALLOW protected registers

IER |= M\_INT1;

#endif

SetupADCSoftware();

//开启第一次AD数据采集

AdcDriver\_Start();

}

/\*==================================================================

\* Function : ADC启动

\* Description :

\* Input Para :

\* Output Para :

\* Return Value:

==================================================================\*/

void AdcDriver\_Start()

{

//开启a的4个通道 0、1、2、3

AdcaRegs.ADCSOCFRC1.all = 0x003f; //SOC0 to SOC3

}

/\*==================================================================

\* Function : 读取adca 通道flag测量值

\* Description :

\* Input Para :

\* Output Para :

\* Return Value:

==================================================================\*/

Uint16 adca\_dri\_get\_value(char flag)

{

Uint16 nValue;

//

//wait for ADCA to complete, then acknowledge flag

//

while(AdcaRegs.ADCINTFLG.bit.ADCINT1 == 0);

AdcaRegs.ADCINTFLGCLR.bit.ADCINT1 = 1;

switch(flag)

{

case 0:

{

nValue = AdcaResultRegs.ADCRESULT0;

break;

}

case 1:

{

nValue = AdcaResultRegs.ADCRESULT1;

break;

}

case 2:

{

nValue = AdcaResultRegs.ADCRESULT2;

break;

}

case 3:

{

nValue = AdcaResultRegs.ADCRESULT3;

break;

}

case 4:

{

nValue = AdcaResultRegs.ADCRESULT4;

break;

}

case 5:

{

nValue = AdcaResultRegs.ADCRESULT5;

break;

}

default:

break;

}

return nValue;

}

/\*==================================================================

\* Function : 读取5V电源电压测量值

\* Description :

\* Input Para :

\* Output Para :

\* Return Value:

==================================================================\*/

void AdcDriver\_ValueGet(void)

{

//

//wait for ADCA to complete, then acknowledge flag

//

while(AdcaRegs.ADCINTFLG.bit.ADCINT1 == 0);

AdcaRegs.ADCINTFLGCLR.bit.ADCINT1 = 1;

adc\_result[0] = AdcaResultRegs.ADCRESULT0;

adc\_result[1] = AdcaResultRegs.ADCRESULT1;

adc\_result[2] = AdcaResultRegs.ADCRESULT2;

adc\_result[3] = AdcaResultRegs.ADCRESULT3;

adc\_result[5] = AdcaResultRegs.ADCRESULT5;

adc\_result\_new = TRUE;

}

void adc\_isr(void)

{

/\* adc\_result[0] = AdcaResultRegs.ADCRESULT0 >>4;

adc\_result[1] = AdcaResultRegs.ADCRESULT1 >>4;

adc\_result[2] = AdcaResultRegs.ADCRESULT2 >>4;

adc\_result\_new = TRUE;

// Reinitialize for next ADC sequence

AdcaResultRegs.ADCST.bit.INT\_SEQ1\_CLR = 1; // Clear INT SEQ1 bit

PieCtrlRegs.PIEACK.all = PIEACK\_GROUP1; // Acknowledge interrupt to PIE

return;

\*/

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*BEGIN\_FILE\_HDR

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* NOTICE:

\* Copyright (c) 2013 by @ atsing.cn. All Rights Reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* FILE : atsingdef.h

\* Description :

\*

\* used for ALL MCU

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Version Initials Date CR# Descriptions

\* 1.0 hjdai@atsing.cn 2013/03/01 N/A Initial Version

\* 1.1 hjdai@atsing.cn 2013/04/01 N/A same data type def with ucos-II, delete xdata data type

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*END\_FILE\_HDR\*/

#ifndef \_ATSINGDEF\_H\_

#define \_ATSINGDEF\_H\_

#define SET\_BIT0 0x01

#define SET\_BIT1 0x02

#define SET\_BIT2 0x04

#define SET\_BIT3 0x08

#define SET\_BIT4 0x10

#define SET\_BIT5 0x20

#define SET\_BIT6 0x40

#define SET\_BIT7 0x80

#define CLR\_BIT0 0xFE

#define CLR\_BIT1 0xFD

#define CLR\_BIT2 0xFB

#define CLR\_BIT3 0xF7

#define CLR\_BIT4 0xEF

#define CLR\_BIT5 0xDF

#define CLR\_BIT6 0xBF

#define CLR\_BIT7 0x7F

#ifndef BIT0

#define BIT0 0x00000001

#endif

#ifndef BIT1

#define BIT1 0x00000002

#endif

#ifndef BIT2

#define BIT2 0x00000004

#endif

#ifndef BIT3

#define BIT3 0x00000008

#endif

#ifndef BIT4

#define BIT4 0x00000010

#endif

#ifndef BIT5

#define BIT5 0x00000020

#endif

#ifndef BIT6

#define BIT6 0x00000040

#endif

#ifndef BIT7

#define BIT7 0x00000080

#endif

#ifndef BIT8

#define BIT8 0x00000100

#endif

#ifndef BIT9

#define BIT9 0x00000200

#endif

#ifndef BIT10

#define BIT10 0x00000400

#endif

#ifndef BIT11

#define BIT11 0x00000800

#endif

#ifndef BIT12

#define BIT12 0x00001000

#endif

#ifndef BIT13

#define BIT13 0x00002000

#endif

#ifndef BIT14

#define BIT14 0x00004000

#endif

#ifndef BIT15

#define BIT15 0x00008000

#endif

#ifndef BIT16

#define BIT16 0x00010000

#endif

#define SET\_BITX (0x01U)

#define CLR\_BITX (0x00U)

//used to configure in or configure out a specific function

#define FUN\_EN 1

#define FUN\_DIS 0

#ifndef NULL

#define NULL 0

#endif

#ifndef FALSE

#define FALSE 0

#endif

#ifndef TRUE

#define TRUE 1

#endif

#ifndef INVALID

#define INVALID (2U)

#endif

#define CPU\_C8051 0x01u

#define CPU\_TMS320C28 0x02u

#define CPUTYPE CPU\_TMS320C28

#if CPUTYPE == CPU\_C8051

/\* data type defintion \*/

typedef unsigned char BOOLEAN;

typedef unsigned char INT8U; /\* Unsigned 8 bit quantity \*/

typedef signed char INT8S; /\* Signed 8 bit quantity \*/

typedef unsigned int INT16U; /\* Unsigned 16 bit quantity \*/

typedef signed int INT16S; /\* Signed 16 bit quantity \*/

typedef unsigned long INT32U; /\* Unsigned 32 bit quantity \*/

typedef signed long INT32S; /\* Signed 32 bit quantity \*/

typedef float FP32; /\* Single precision floating point \*/

typedef double FP64; /\* Double precision floating point \*/

/\*storage type defintion \*/

#define XDATA xdata

/\* NOP () macro support \*/

extern void \_nop\_ (void);

#define NOP() \_nop\_()

#endif

#if CPUTYPE == CPU\_TMS320C28

/\* data type defintion \*/

typedef unsigned char BOOLEAN;

typedef unsigned char INT8U; /\* Unsigned 8 bit quantity \*/

typedef signed char INT8S; /\* Signed 8 bit quantity \*/

typedef unsigned int INT16U; /\* Unsigned 16 bit quantity \*/

typedef signed int INT16S; /\* Signed 16 bit quantity \*/

typedef unsigned long INT32U; /\* Unsigned 32 bit quantity \*/

typedef signed long INT32S; /\* Signed 32 bit quantity \*/

typedef unsigned long long INT64U; /\* Unsigned 32 bit quantity \*/

typedef signed long long INT64S; /\* Signed 32 bit quantity \*/

typedef float FP32; /\* Single precision floating point \*/

typedef double FP64; /\* Double precision floating point \*/

/\*storage type defintion \*/

#define XDATA

/\* NOP () macro support \*/

#define NOP() asm(" nop")

#endif

#endif /\* \*/

//###########################################################################

//

// FILE: can.h

//

// TITLE: Defines and Macros for the CAN controller.

//

//###########################################################################

// $TI Release: F2837xS Support Library v3.00.00.00 $

// $Release Date: Wed Jan 25 16:06:35 CST 2017 $

// $Copyright:

// Copyright (C) 2014-2017 Texas Instruments Incorporated - http://www.ti.com/

//

// Redistribution and use in source and binary forms, with or without

// modification, are permitted provided that the following conditions

// are met:

//

// Redistributions of source code must retain the above copyright

// notice, this list of conditions and the following disclaimer.

//

// Redistributions in binary form must reproduce the above copyright

// notice, this list of conditions and the following disclaimer in the

// documentation and/or other materials provided with the

// distribution.

//

// Neither the name of Texas Instruments Incorporated nor the names of

// its contributors may be used to endorse or promote products derived

// from this software without specific prior written permission.

//

// THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS

// "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT

// LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR

// A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT

// OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,

// SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT

// LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,

// DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY

// THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT

// (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE

// OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

// $

//###########################################################################

#ifndef \_\_CAN\_H\_\_

#define \_\_CAN\_H\_\_

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//! \addtogroup can\_api

//! @{

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// If building with a C++ compiler, make all of the definitions in this header

// have a C binding.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#ifdef \_\_cplusplus

extern "C"

{

#endif

#define CAN\_INDEX\_TO\_BASE(idx) ((idx == 0) ? CAN\_A\_BASE : CAN\_B\_BASE)

#define CAN\_INDEX\_TO\_MSG\_RAM\_BASE(idx) ((idx == 0) ? CAN\_A\_MSG\_RAM : CAN\_B\_MSG\_RAM)

#define CAN\_REG\_WORD\_MASK (0xFFFFU)

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// These are the Defines to select CAN pin muxing when calling the functions

// ConfigCanPinMuxing(), ConfigGpioCanA() & ConfigGpioCanB() in F2837x\_Can.c

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#define CAN\_A\_GPIO4\_GPIO5 1 //switch case 1

#define CAN\_A\_GPIO19\_GPIO18 2 //switch case 2

#define CAN\_A\_GPIO31\_GPIO30 3 //switch case 3

#define CAN\_A\_GPIO37\_GPIO36 4 //switch case 4

#define CAN\_A\_GPIO63\_GPIO62 5 //switch case 5

#define CAN\_A\_GPIO71\_GPIO70 6 //switch case 6

#define CAN\_B\_GPIO6\_GPIO7 1 //switch case 1

#define CAN\_B\_GPIO8\_GPIO10 2 //switch case 2

#define CAN\_B\_GPIO12\_GPIO13 3 //switch case 3

#define CAN\_B\_GPIO16\_GPIO17 4 //switch case 4

#define CAN\_B\_GPIO20\_GPIO21 5 //switch case 5

#define CAN\_B\_GPIO38\_GPIO39 6 //switch case 6

#define CAN\_B\_GPIO72\_GPIO73 7 //switch case 7

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Miscellaneous defines for Message ID Types

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// These are the flags used by the tCANMsgObject.ui32Flags value when calling the

// CANMessageSet() and CANMessageGet() functions.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//! This definition is used with the tCANMsgObject ui32Flags value and indicates

//! that transmit interrupts should be enabled, or are enabled.

#define MSG\_OBJ\_TX\_INT\_ENABLE 0x00000001

//! This indicates that receive interrupts should be enabled, or are

//! enabled.

#define MSG\_OBJ\_RX\_INT\_ENABLE 0x00000002

//! This indicates that a message object will use or is using an extended

//! identifier.

#define MSG\_OBJ\_EXTENDED\_ID 0x00000004

//! This indicates that a message object will use or is using filtering

//! based on the object's message identifier.

#define MSG\_OBJ\_USE\_ID\_FILTER 0x00000008

//! This indicates that new data was available in the message object.

#define MSG\_OBJ\_NEW\_DATA 0x00000080

//! This indicates that data was lost since this message object was last

//! read.

#define MSG\_OBJ\_DATA\_LOST 0x00000100

//! This indicates that a message object will use or is using filtering

//! based on the direction of the transfer. If the direction filtering is

//! used, then ID filtering must also be enabled.

#define MSG\_OBJ\_USE\_DIR\_FILTER (0x00000010 | MSG\_OBJ\_USE\_ID\_FILTER)

//! This indicates that a message object will use or is using message

//! identifier filtering based on the extended identifier. If the extended

//! identifier filtering is used, then ID filtering must also be enabled.

#define MSG\_OBJ\_USE\_EXT\_FILTER (0x00000020 | MSG\_OBJ\_USE\_ID\_FILTER)

//! This indicates that a message object is a remote frame.

#define MSG\_OBJ\_REMOTE\_FRAME 0x00000040

//! This indicates that this message object is part of a FIFO structure and

//! not the final message object in a FIFO.

#define MSG\_OBJ\_FIFO 0x00000200

//! This indicates that a message object has no flags set.

#define MSG\_OBJ\_NO\_FLAGS 0x00000000

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//! This define is used with the flag values to allow checking only status

//! flags and not configuration flags.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#define MSG\_OBJ\_STATUS\_MASK (MSG\_OBJ\_NEW\_DATA | MSG\_OBJ\_DATA\_LOST)

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//! The structure used for encapsulating all the items associated with a CAN

//! message object in the CAN controller.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

typedef struct

{

//! The CAN message identifier used for 11 or 29 bit identifiers.

uint32\_t ui32MsgID;

//! The message identifier mask used when identifier filtering is enabled.

uint32\_t ui32MsgIDMask;

//! This value holds various status flags and settings specified by

//! tCANObjFlags.

uint32\_t ui32Flags;

//! This value is the number of bytes of data in the message object.

uint32\_t ui32MsgLen;

//! This is a pointer to the message object's data.

unsigned char \*pucMsgData;

}

tCANMsgObject;

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//! This structure is used for encapsulating the values associated with setting

//! up the bit timing for a CAN controller. The structure is used when calling

//! the CANGetBitTiming and CANSetBitTiming functions.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

typedef struct

{

//! This value holds the sum of the Synchronization, Propagation, and Phase

//! Buffer 1 segments, measured in time quanta. The valid values for this

//! setting range from 2 to 16.

uint16\_t uSyncPropPhase1Seg;

//! This value holds the Phase Buffer 2 segment in time quanta. The valid

//! values for this setting range from 1 to 8.

uint16\_t uPhase2Seg;

//! This value holds the Resynchronization Jump Width in time quanta. The

//! valid values for this setting range from 1 to 4.

uint16\_t uSJW;

//! This value holds the CAN\_CLK divider used to determine time quanta.

//! The valid values for this setting range from 1 to 1023.

uint16\_t uQuantumPrescaler;

}

tCANBitClkParms;

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//! This data type is used to identify the interrupt status register. This is

//! used when calling the CANIntStatus() function.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

typedef enum

{

//! Read the CAN interrupt status information.

CAN\_INT\_STS\_CAUSE,

//! Read a message object's interrupt status.

CAN\_INT\_STS\_OBJECT

}

tCANIntStsReg;

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//! This data type is used to identify which of several status registers to

//! read when calling the CANStatusGet() function.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

typedef enum

{

//! Read the full CAN controller status.

CAN\_STS\_CONTROL,

//! Read the full 32-bit mask of message objects with a transmit request

//! set.

CAN\_STS\_TXREQUEST,

//! Read the full 32-bit mask of message objects with new data available.

CAN\_STS\_NEWDAT,

//! Read the full 32-bit mask of message objects that are enabled.

CAN\_STS\_MSGVAL

}

tCANStsReg;

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// These definitions are used to specify interrupt sources to CANIntEnable()

// and CANIntDisable().

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//! This flag is used to allow a CAN controller to generate error

//! interrupts.

#define CAN\_INT\_ERROR 0x00000008

//! This flag is used to allow a CAN controller to generate status

//! interrupts.

#define CAN\_INT\_STATUS 0x00000004

//! This flag is used to allow a CAN controller to generate interrupts

//! on interrupt line 0

#define CAN\_INT\_IE0 0x00000002

//! This flag is used to allow a CAN controller to generate interrupts

//! on interrupt line 1

#define CAN\_INT\_IE1 0x00020000

// Defined to maintain compatibility with Stellaris Examples

#define CAN\_INT\_MASTER CAN\_INT\_IE0

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// These definitions are used to specify the clock source to

// CANClkSourceSelect()

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//! This flag is used to clock the CAN controller Selected CPU SYSCLKOUT

//! (CPU1.Sysclk or CPU2.Sysclk).

#define CAN\_CLK\_CPU\_SYSCLKOUT 0 // PERx.SYSCLK (default on reset)

//! This flag is used to clock the CAN controller with the X1/X2 oscillator

//! clock.

#define CAN\_CLK\_EXT\_OSC 1 // External Oscillator (XTAL)

//! This flag is used to clock the CAN controller with the clock from

//! AUXCLKIN (from GPIO)

#define CAN\_CLK\_AUXCLKIN 2 // AUXCLKIN (from GPIO)

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//! This definition is used to determine the type of message object that will

//! be set up via a call to the CANMessageSet() API.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

typedef enum

{

//! Transmit message object.

MSG\_OBJ\_TYPE\_TX,

//! Transmit remote request message object

MSG\_OBJ\_TYPE\_TX\_REMOTE,

//! Receive message object.

MSG\_OBJ\_TYPE\_RX,

//! Receive remote request message object.

MSG\_OBJ\_TYPE\_RX\_REMOTE,

//! Remote frame receive remote, with auto-transmit message object.

MSG\_OBJ\_TYPE\_RXTX\_REMOTE

}

tMsgObjType;

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// The following enumeration contains all error or status indicators that can

// be returned when calling the CANStatusGet() function.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//! CAN controller is in local power down mode.

#define CAN\_STATUS\_PDA 0x00000400

//! CAN controller has initiated a system wakeup.

#define CAN\_STATUS\_WAKE\_UP 0x00000200

//! CAN controller has detected a parity error.

#define CAN\_STATUS\_PERR 0x00000100

//! CAN controller has entered a Bus Off state.

#define CAN\_STATUS\_BUS\_OFF 0x00000080

//! CAN controller error level has reached warning level.

#define CAN\_STATUS\_EWARN 0x00000040

//! CAN controller error level has reached error passive level.

#define CAN\_STATUS\_EPASS 0x00000020

//! A message was received successfully since the last read of this status.

#define CAN\_STATUS\_RXOK 0x00000010

//! A message was transmitted successfully since the last read of this

//! status.

#define CAN\_STATUS\_TXOK 0x00000008

//! This is the mask for the last error code field.

#define CAN\_STATUS\_LEC\_MSK 0x00000007

//! There was no error.

#define CAN\_STATUS\_LEC\_NONE 0x00000000

//! A bit stuffing error has occurred.

#define CAN\_STATUS\_LEC\_STUFF 0x00000001

//! A formatting error has occurred.

#define CAN\_STATUS\_LEC\_FORM 0x00000002

//! An acknowledge error has occurred.

#define CAN\_STATUS\_LEC\_ACK 0x00000003

//! The bus remained a bit level of 1 for longer than is allowed.

#define CAN\_STATUS\_LEC\_BIT1 0x00000004

//! The bus remained a bit level of 0 for longer than is allowed.

#define CAN\_STATUS\_LEC\_BIT0 0x00000005

//! A CRC error has occurred.

#define CAN\_STATUS\_LEC\_CRC 0x00000006

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// The following macros are added for the new Global Interrupt EN/FLG/CLR

// register

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//CANINT0 global interrupt bit

#define CAN\_GLOBAL\_INT\_CANINT0 0x00000001

//CANINT1 global interrupt bit

#define CAN\_GLOBAL\_INT\_CANINT1 0x00000002

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// The following macros are missing in hw\_can.h because of scripting

// but driverlib can.c needs them

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#define CAN\_INT\_INT0ID\_STATUS 0x8000

#define CAN\_IF1ARB\_STD\_ID\_S 18

#define CAN\_IF1ARB\_STD\_ID\_M 0x1FFC0000 // Standard Message Identifier

#define CAN\_IF2ARB\_STD\_ID\_S 18

#define CAN\_IF2ARB\_STD\_ID\_M 0x1FFC0000 // Standard Message Identifier

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// API Function prototypes

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

extern void CANClkSourceSelect(uint32\_t ui32Base, uint16\_t ucSource);

extern void CANBitTimingGet(uint32\_t ui32Base, tCANBitClkParms \*pClkParms);

extern void CANBitTimingSet(uint32\_t ui32Base, tCANBitClkParms \*pClkParms);

extern uint32\_t CANBitRateSet(uint32\_t ui32Base, uint32\_t ui32SourceClock,

uint32\_t ui32BitRate);

extern void CANDisable(uint32\_t ui32Base);

extern void CANEnable(uint32\_t ui32Base);

extern bool CANErrCntrGet(uint32\_t ui32Base, uint32\_t \*pui32RxCount,

uint32\_t \*pui32TxCount);

extern void CANInit(uint32\_t ui32Base);

extern void CANIntClear(uint32\_t ui32Base, uint32\_t ui32IntClr);

extern void CANIntDisable(uint32\_t ui32Base, uint32\_t ui32IntFlags);

extern void CANIntEnable(uint32\_t ui32Base, uint32\_t ui32IntFlags);

extern void CANIntRegister(uint32\_t ui32Base, unsigned char ucIntNumber,

void (\*pfnHandler)(void));

extern uint32\_t CANIntStatus(uint32\_t ui32Base, tCANIntStsReg eIntStsReg);

extern void CANIntUnregister(uint32\_t ui32Base, unsigned char ucIntNumber);

extern void CANMessageClear(uint32\_t ui32Base, uint32\_t ui32ObjID);

extern void CANMessageGet(uint32\_t ui32Base, uint32\_t ui32ObjID,

tCANMsgObject \*pMsgObject, bool bClrPendingInt);

extern void CANMessageSet(uint32\_t ui32Base, uint32\_t ui32ObjID,

tCANMsgObject \*pMsgObject, tMsgObjType eMsgType);

extern bool CANRetryGet(uint32\_t ui32Base);

extern void CANRetrySet(uint32\_t ui32Base, bool bAutoRetry);

extern uint32\_t CANStatusGet(uint32\_t ui32Base, tCANStsReg eStatusReg);

extern void CANGlobalIntEnable(uint32\_t ui32Base, uint32\_t ui32IntFlags);

extern void CANGlobalIntDisable(uint32\_t ui32Base, uint32\_t ui32IntFlags);

extern void CANGlobalIntClear(uint32\_t ui32Base, uint32\_t ui32IntFlags);

extern bool CANGlobalIntstatusGet(uint32\_t ui32Base, uint32\_t ui32IntFlags);

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Mark the end of the C bindings section for C++ compilers.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#ifdef \_\_cplusplus

}

#endif

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Close the Doxygen group.

//! @}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#endif // \_\_CAN\_H\_\_

/\*BEGIN\_FILE\_HDR

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* NOTICE:

\* Copyright (c) 2020 by @ kingsigna.cn. All Rights Reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* FILE : XXXDrv.h

\* Description :

\*

\* (Requirements, pseudo code and etc.)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Version Initials Date CR# Descriptions

\* 1.0 mufeng.li@kingsigna.cn 2020/10/12 N/A Initial Version

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*END\_FILE\_HDR\*/

#ifndef COMMDRV\_H

#define COMMDRV\_H

/\*Include files\*/

#include "atsingdef.h"

#include "../state\_management/StateManagementAppExt.h"

/\*\*\*\*following definitions CAN NOT be used outside this driver\*\*\*\*\*\*/

typedef INT8S (\*RCV\_FUN\_DEF)(void \* in\_s, void \* out\_s);

typedef INT8S (\*SND\_FUN\_DEF)(void \* in\_s, void \* out\_s);

/\* structure definiton \*/

struct ParserBlock{

INT16U m\_nFunction;

RCV\_FUN\_DEF m\_pRcvFun;

SND\_FUN\_DEF m\_pSndFun;

void \* recv\_out;

void \* send\_in;

};

struct ParserBlock\_ZTE{

char m\_nFunction[30];

RCV\_FUN\_DEF m\_pRcvFun;

SND\_FUN\_DEF m\_pSndFun;

void \* recv\_out;

void \* send\_in;

};

enum Comm\_Src{

UNKNOWN = 0x00,

ZTE = 0x10,

ACU = 0x20

};

/\* macro definition \*/

//#define COMM\_PROTOCOL\_PELCOD

#define COMM\_PROTOCOL\_ATSING

#define COMM\_PROTOCOL\_ATSING\_NMEA

#define BROADCASTCNT 10

struct broadcastTimerCnt{

INT16U thr;

INT16U cnt;

};

/\* 0 :info

\* 1:rawimu

\* 2:gpfpd

\* 3:gpgga

\* \*/

#define BROADCAST\_INDEX\_INFO 0

#define BROADCAST\_INDEX\_RAWIMU 1

#define BROADCAST\_INDEX\_GPFPD 2

#define BROADCAST\_INDEX\_GPGGA 3

#define BROADCAST\_INDEX\_GPRMC 4

#define BROADCAST\_INDEX\_GINS 5

#define BROADCAST\_INDEX\_REPORT 6

#define BROADCAST\_INDEX\_DBG 7

extern struct broadcastTimerCnt gbroadcastTimerCnt[BROADCASTCNT];

#define MAX\_BUFFER 512

extern char m\_gSndCommGnssBuffer[MAX\_BUFFER];

extern char m\_gSndCommDbgBuffer[MAX\_BUFFER];

/\* function declaration \*/

extern Void task\_CommunicationInit();

extern Void task\_Communication(UArg arg0, UArg arg1);

#endif /\* end of header file \*/

/\*BEGIN\_FILE\_HDR

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* NOTICE:

\* Copyright (c) 2013 by @ atsing.cn. All Rights Reserved.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* FILE : AdcDrv.c

\* Description :

\*

\* (Requirements, pseudo code and etc.)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Version Initials Date CR# Descriptions

\* 1.0 ygfeng@atsing.cn 2013/06/14 N/A Initial Version

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*END\_FILE\_HDR\*/

#ifndef COMMDRV\_C

#define COMMDRV\_C

#endif

#include "sys\_bios\_include.h"

#include "drivers\_headers.h"

#include "interaction\_modules.h"

#include "application\_include.h"

#include "RecvFun.h"

#include "SendFun.h"

#include "setdatafunc.h"

//#include "pelcod.h"

//#include "pelcodfun.h"

/\* include files

#include "atsingdef.h"

#include <xdc/runtime/System.h>

#include <xdc/runtime/Error.h>

#include <ti/sysbios/hal/Hwi.h>

//#define DSP28\_DATA\_TYPES

#include "F28x\_Project.h"

#include <ti/sysbios/knl/Task.h>

#include <ti/sysbios/knl/Queue.h>

#include <ti/sysbios/knl/Event.h>

#include <ti/sysbios/knl/Semaphore.h>

#include <xdc/runtime/Timestamp.h>

#include <stdio.h>

#include "common.h"

#include "COMMDrv.h"

#include "RecvFun.h"

#include "SendFun.h"

#include "setdatafunc.h"

#include "string.h"

#include "SciExt.h"

#include "AdcExt.h"

#include "main.h"

#include "ADCCalc/ADCCalcAppExt.h"

#include "motor\_control.h"

#include "MonitorExt.h"

#include "StateManagementExt.h"

#include "task\_stabilization/task\_stabilization.h"

#include "qep/QEPAppExt.h"

#include "esc\_can/EscCanDrv.h"

#include "pelcod.h"

#include "pelcodfun.h"

#include "GPS/nmea.h"

#include "timer/timerAppExt.h"

\*/

/\* static variable declaration \*/

//static

#ifdef \_\_cplusplus

#pragma DATA\_SECTION("BlockBufferFile")

#else

#pragma DATA\_SECTION(m\_gRcvCommBuffer,"BlockBufferFile");

#endif

//接收缓冲区

INT8S m\_gRcvCommBuffer[MAX\_BUFFER];

INT8S m\_gRcvCommBuffer2[MAX\_BUFFER];

#ifdef \_\_cplusplus

#pragma DATA\_SECTION("BlockBufferFile")

#else

#pragma DATA\_SECTION(m\_gSndCommBuffer,"BlockBufferFile");

#endif

//发送缓冲区

INT8S m\_gSndCommBuffer[MAX\_BUFFER] = {NULL};

INT8S m\_gSndCommBuffer2[MAX\_BUFFER] = {NULL};

char m\_gSndCommGnssBuffer[MAX\_BUFFER] = {NULL};

char m\_gSndCommDbgBuffer[MAX\_BUFFER] = {NULL};

static INT16U m\_gRcvIndex = 0;

static INT16U m\_gSndIndex = 0;

static INT16U m\_gRcvIndex2 = 0;

static INT16U m\_gSndIndex2 = 0;

enum Comm\_Src comm\_src;

INT16U g\_nBroadcastTimer\_fun62 = 0;

INT16U g\_nBroadcastTimer\_fun62\_cnt = 100; // 100; // 100 \* 20ms = 2s 5:输出4次/s，CPU 30%。 10:2次/s

#define BROADCAST\_COUNTER 1

#ifdef \_\_cplusplus

#pragma DATA\_SECTION("BlockBufferFile")

#else

#pragma DATA\_SECTION(s\_buf,"BlockBufferFile");

#endif

STRUCT\_xx\_S\_R s\_buf = {{0},0};

#ifdef \_\_cplusplus

#pragma DATA\_SECTION("BlockBufferFile")

#else

#pragma DATA\_SECTION(r\_buf,"BlockBufferFile");

#endif

STRUCT\_xx\_S\_R r\_buf = {{0}, 0};

#ifdef \_\_cplusplus

#pragma DATA\_SECTION("BlockBufferFile")

#else

#pragma DATA\_SECTION(s\_buf\_broadcast,"BlockBufferFile");

#endif

STRUCT\_xx\_S\_R s\_buf\_broadcast = {{0},0};

//报文解析函数映射

static struct ParserBlock g\_pCommParsers[] = {

{0x10, (RCV\_FUN\_DEF)do\_rcv\_fun\_10, (SND\_FUN\_DEF)do\_snd\_fun\_10, (void \*)&out\_10\_r,(void \*)&in\_10\_s},

{0x11, (RCV\_FUN\_DEF)do\_rcv\_fun\_11, (SND\_FUN\_DEF)do\_snd\_fun\_11,(void \*)&out\_11\_r, (void \*)&in\_11\_s},

{0x12, (RCV\_FUN\_DEF)do\_rcv\_fun\_12, (SND\_FUN\_DEF)do\_snd\_fun\_12, (void \*)&out\_12\_r,(void \*)&in\_12\_s},

{0x13, (RCV\_FUN\_DEF)do\_rcv\_fun\_13, (SND\_FUN\_DEF)do\_snd\_fun\_13,(void \*)&out\_13\_r, (void \*)&in\_13\_s},

{0x14, (RCV\_FUN\_DEF)do\_rcv\_fun\_14, (SND\_FUN\_DEF)do\_snd\_fun\_14, (void \*)&out\_14\_r,(void \*)&in\_14\_s},

{0x15, (RCV\_FUN\_DEF)do\_rcv\_fun\_15, (SND\_FUN\_DEF)do\_snd\_fun\_15, (void \*)&out\_15\_r,(void \*)&in\_15\_s},

{0x16, (RCV\_FUN\_DEF)do\_rcv\_fun\_16, (SND\_FUN\_DEF)do\_snd\_fun\_16,(void \*)&out\_16\_r, (void \*)&in\_16\_s},

{0x17, (RCV\_FUN\_DEF)do\_rcv\_fun\_17, (SND\_FUN\_DEF)do\_snd\_fun\_17, (void \*)&out\_17\_r,(void \*)&in\_17\_s},

{0x18, (RCV\_FUN\_DEF)do\_rcv\_fun\_18, (SND\_FUN\_DEF)do\_snd\_fun\_18,(void \*)&out\_18\_r, (void \*)&in\_18\_s},

{0x19, (RCV\_FUN\_DEF)do\_rcv\_fun\_19, (SND\_FUN\_DEF)do\_snd\_fun\_19, (void \*)&out\_19\_r,(void \*)&in\_19\_s},

{0x1b, (RCV\_FUN\_DEF)do\_rcv\_fun\_1b, (SND\_FUN\_DEF)do\_snd\_fun\_1b, (void \*)&out\_1b\_r,(void \*)&in\_1b\_s},

{0x20, (RCV\_FUN\_DEF)do\_rcv\_fun\_20, (SND\_FUN\_DEF)do\_snd\_fun\_20,(void \*)&out\_20\_r, (void \*)&in\_20\_s},

{0x21, (RCV\_FUN\_DEF)do\_rcv\_fun\_21, (SND\_FUN\_DEF)do\_snd\_fun\_21,(void \*)&out\_21\_r, (void \*)&in\_21\_s},

{0x22, (RCV\_FUN\_DEF)do\_rcv\_fun\_22, (SND\_FUN\_DEF)do\_snd\_fun\_22,(void \*)&out\_22\_r, (void \*)&in\_22\_s},

{0x23, (RCV\_FUN\_DEF)do\_rcv\_fun\_23, (SND\_FUN\_DEF)do\_snd\_fun\_23,(void \*)&out\_23\_r, (void \*)&in\_23\_s},

{0x24, (RCV\_FUN\_DEF)do\_rcv\_fun\_24, (SND\_FUN\_DEF)do\_snd\_fun\_24,(void \*)&out\_24\_r, (void \*)&in\_24\_s},

{0x25, (RCV\_FUN\_DEF)do\_rcv\_fun\_25, (SND\_FUN\_DEF)do\_snd\_fun\_25, (void \*)&out\_25\_r,(void \*)&in\_25\_s},

{0x26, (RCV\_FUN\_DEF)do\_rcv\_fun\_26, (SND\_FUN\_DEF)do\_snd\_fun\_26,(void \*)&out\_26\_r, (void \*)&in\_26\_s},

{0x27, (RCV\_FUN\_DEF)do\_rcv\_fun\_27, (SND\_FUN\_DEF)do\_snd\_fun\_27,(void \*)&out\_27\_r, (void \*)&in\_27\_s},

{0x28, (RCV\_FUN\_DEF)do\_rcv\_fun\_28, (SND\_FUN\_DEF)do\_snd\_fun\_28,(void \*)&out\_28\_r, (void \*)&in\_28\_s},

{0x29, (RCV\_FUN\_DEF)do\_rcv\_fun\_29, (SND\_FUN\_DEF)do\_snd\_fun\_29,(void \*)&out\_29\_r, (void \*)&in\_29\_s},

{0x2A, (RCV\_FUN\_DEF)do\_rcv\_fun\_2A, (SND\_FUN\_DEF)do\_snd\_fun\_2A,(void \*)&out\_2a\_r, (void \*)&in\_2a\_s},

{0x30, (RCV\_FUN\_DEF)do\_rcv\_fun\_30, (SND\_FUN\_DEF)do\_snd\_fun\_30,(void \*)&out\_30\_r, (void \*)&in\_30\_s},

{0x31, (RCV\_FUN\_DEF)do\_rcv\_fun\_31, (SND\_FUN\_DEF)do\_snd\_fun\_31,(void \*)&out\_31\_r, (void \*)&in\_31\_s},

{0x32, (RCV\_FUN\_DEF)do\_rcv\_fun\_32, (SND\_FUN\_DEF)do\_snd\_fun\_32,(void \*)&out\_32\_r, (void \*)&in\_32\_s},

{0x33, (RCV\_FUN\_DEF)do\_rcv\_fun\_33, (SND\_FUN\_DEF)do\_snd\_fun\_33,(void \*)&out\_33\_r, (void \*)&in\_33\_s},

{0x34, (RCV\_FUN\_DEF)do\_rcv\_fun\_34, (SND\_FUN\_DEF)do\_snd\_fun\_34,(void \*)&out\_34\_r, (void \*)&in\_34\_s},

{0x35, (RCV\_FUN\_DEF)do\_rcv\_fun\_35, (SND\_FUN\_DEF)do\_snd\_fun\_35,(void \*)&out\_35\_r, (void \*)&in\_35\_s},

{0x36, (RCV\_FUN\_DEF)do\_rcv\_fun\_36, (SND\_FUN\_DEF)do\_snd\_fun\_36,(void \*)&out\_36\_r, (void \*)&in\_36\_s},

{0x37, (RCV\_FUN\_DEF)do\_rcv\_fun\_37, (SND\_FUN\_DEF)do\_snd\_fun\_37,(void \*)&out\_37\_r, (void \*)&in\_37\_s},

{0x38, (RCV\_FUN\_DEF)do\_rcv\_fun\_38, (SND\_FUN\_DEF)do\_snd\_fun\_38,(void \*)&out\_38\_r, (void \*)&in\_38\_s},

{0x39, (RCV\_FUN\_DEF)do\_rcv\_fun\_39, (SND\_FUN\_DEF)do\_snd\_fun\_39,(void \*)&out\_39\_r, (void \*)&in\_39\_s},

{0x3A, (RCV\_FUN\_DEF)do\_rcv\_fun\_3A, (SND\_FUN\_DEF)do\_snd\_fun\_3A,(void \*)&out\_3a\_r, (void \*)&in\_3a\_s},

{0x40, (RCV\_FUN\_DEF)do\_rcv\_fun\_40, (SND\_FUN\_DEF)do\_snd\_fun\_40,(void \*)&out\_40\_r, (void \*)&in\_40\_s},

{0x41, (RCV\_FUN\_DEF)do\_rcv\_fun\_41, (SND\_FUN\_DEF)do\_snd\_fun\_41,(void \*)&out\_41\_r, (void \*)&in\_41\_s},

{0x42, (RCV\_FUN\_DEF)do\_rcv\_fun\_42, (SND\_FUN\_DEF)do\_snd\_fun\_42,(void \*)&out\_42\_r, (void \*)&in\_42\_s},

{0x43, (RCV\_FUN\_DEF)do\_rcv\_fun\_43, (SND\_FUN\_DEF)do\_snd\_fun\_43,(void \*)&out\_43\_r, (void \*)&in\_43\_s},

// {0x44, (RCV\_FUN\_DEF)do\_rcv\_fun\_44, (SND\_FUN\_DEF)do\_snd\_fun\_44,(void \*)&out\_44\_r, (void \*)&in\_44\_s},

{0x45, (RCV\_FUN\_DEF)do\_rcv\_fun\_45, (SND\_FUN\_DEF)do\_snd\_fun\_45,(void \*)&out\_45\_r, (void \*)&in\_45\_s},

{0x46, (RCV\_FUN\_DEF)do\_rcv\_fun\_46, (SND\_FUN\_DEF)do\_snd\_fun\_46,(void \*)&out\_46\_r, (void \*)&in\_46\_s},

// {0x47, (RCV\_FUN\_DEF)do\_rcv\_fun\_47, (SND\_FUN\_DEF)do\_snd\_fun\_47,(void \*)&out\_47\_r, (void \*)&in\_47\_s},

{0x48, (RCV\_FUN\_DEF)do\_rcv\_fun\_48, (SND\_FUN\_DEF)do\_snd\_fun\_48,(void \*)&out\_48\_r, (void \*)&in\_48\_s},

// {0x49, (RCV\_FUN\_DEF)do\_rcv\_fun\_49, (SND\_FUN\_DEF)do\_snd\_fun\_49,(void \*)&out\_49\_r, (void \*)&in\_49\_s},

// {0x4A, (RCV\_FUN\_DEF)do\_rcv\_fun\_4A, (SND\_FUN\_DEF)do\_snd\_fun\_4A,(void \*)&out\_4a\_r, (void \*)&in\_4a\_s},

{0x50, (RCV\_FUN\_DEF)do\_rcv\_fun\_50, (SND\_FUN\_DEF)do\_snd\_fun\_50,(void \*)&out\_50\_r, (void \*)&in\_50\_s},

{0x51, (RCV\_FUN\_DEF)do\_rcv\_fun\_51, (SND\_FUN\_DEF)do\_snd\_fun\_51,(void \*)&out\_51\_r, (void \*)&in\_51\_s},

{0x52, (RCV\_FUN\_DEF)do\_rcv\_fun\_52, (SND\_FUN\_DEF)do\_snd\_fun\_52,(void \*)&out\_52\_r, (void \*)&in\_52\_s},

{0x53, (RCV\_FUN\_DEF)do\_rcv\_fun\_53, (SND\_FUN\_DEF)do\_snd\_fun\_53,(void \*)&out\_53\_r, (void \*)&in\_53\_s},

{0x54, (RCV\_FUN\_DEF)do\_rcv\_fun\_54, (SND\_FUN\_DEF)do\_snd\_fun\_54,(void \*)&out\_54\_r, (void \*)&in\_54\_s},

{0x55, (RCV\_FUN\_DEF)do\_rcv\_fun\_55, (SND\_FUN\_DEF)do\_snd\_fun\_55,(void \*)&out\_55\_r, (void \*)&in\_55\_s},

{0x56, (RCV\_FUN\_DEF)do\_rcv\_fun\_56, (SND\_FUN\_DEF)do\_snd\_fun\_56,(void \*)&out\_56\_r, (void \*)&in\_56\_s},

{0x57, (RCV\_FUN\_DEF)do\_rcv\_fun\_57, (SND\_FUN\_DEF)do\_snd\_fun\_57,(void \*)&out\_57\_r, (void \*)&in\_57\_s},

{0x58, (RCV\_FUN\_DEF)do\_rcv\_fun\_58, (SND\_FUN\_DEF)do\_snd\_fun\_58,(void \*)&out\_58\_r, (void \*)&in\_58\_s},

{0x59, (RCV\_FUN\_DEF)do\_rcv\_fun\_59, (SND\_FUN\_DEF)do\_snd\_fun\_59,(void \*)&out\_59\_r, (void \*)&in\_59\_s},

{0x5A, (RCV\_FUN\_DEF)do\_rcv\_fun\_5A, (SND\_FUN\_DEF)do\_snd\_fun\_5A,(void \*)&out\_5a\_r, (void \*)&in\_5a\_s},

{0x60, (RCV\_FUN\_DEF)do\_rcv\_fun\_60, (SND\_FUN\_DEF)do\_snd\_fun\_60,(void \*)&out\_60\_r, (void \*)&in\_60\_s},

// {0x80, (RCV\_FUN\_DEF)do\_rcv\_fun\_80, (SND\_FUN\_DEF)do\_snd\_fun\_80,(void \*)&out\_80\_r, (void \*)&in\_80\_s}

};

struct broadcastTimerCnt gbroadcastTimerCnt[BROADCASTCNT];

GPS\_DATA\_STRUCT GPS\_Data;

/\* static function definition \*/

struct ParserBlock \*FindParserBlock(INT8U nFunction)

{

int i;

struct ParserBlock \*pFind = NULL;

// IDE(("nfun:%x",nFunction));

for (i = 0; i < ARR\_LEN(g\_pCommParsers); i++)

{

pFind = &g\_pCommParsers[i];

if (pFind->m\_nFunction == nFunction)

{

// IDE(("i:%d", i));

return pFind;

}

}

return NULL;

}

INT16U ParsePacket(STRUCT\_xx\_S\_R \*buf\_r, STRUCT\_xx\_S\_R \*buf\_s)

{

INT16U rtn = TRUE;

struct ParserBlock \*pPaser;

pPaser = FindParserBlock(buf\_r->out\_buf[2]);

if (pPaser != NULL)

{

buf\_r->eff\_len = recv\_packet\_trans\_mean((INT8S \*)buf\_r->out\_buf, (INT8U)buf\_r->eff\_len);//转义

pPaser->m\_pRcvFun(buf\_r, pPaser->recv\_out);

pPaser->m\_pSndFun(pPaser->send\_in, buf\_s);

buf\_s->eff\_len = send\_packet\_trans\_mean((INT8S \*)buf\_s->out\_buf, (INT8U)buf\_s->eff\_len);//发送转译

rtn = TRUE;

}

else

{

rtn = FALSE;

}

return rtn;

}

/\*通讯波特率改为115200\*/

Void task\_CommunicationInit()

{

INT16U i;

//串口A用于与ARM通信、程序升级

g\_AntCtrlIntBlock.m\_commu = SciDriver\_Open(SCIPORT\_A, SCIBAUD\_57600, SCIDATA\_8, SCISTOPBIT\_1, SCIPARITY\_NONE);

//串口C用于读取IMU数据

//串口B用于扩展，如读取信标机数据，或外围调试

#ifndef BEACON\_KC

// g\_AntCtrlIntBlock.m\_commu\_baudrate = SCIBAUD\_115200;

// g\_AntCtrlIntBlock.m\_commu\_test = SciDriver\_Open(SCIPORT\_B, g\_AntCtrlIntBlock.m\_commu\_baudrate, SCIDATA\_8, SCISTOPBIT\_1, SCIPARITY\_NONE);

#endif

for (i=0; i<BROADCASTCNT; i++ )

{

gbroadcastTimerCnt[i].cnt=0;

//gbroadcastTimerCnt[i].thr = 10;

}

}

#define feedback\_ok() { \

sprintf((char\*)s\_buf.out\_buf,"ok\r\n"); \

s\_buf.eff\_len = strlen((char\*)s\_buf.out\_buf); \

memcpy(&m\_gSndCommBuffer2[m\_gSndIndex], s\_buf.out\_buf, s\_buf.eff\_len); \

m\_gSndIndex2 += s\_buf.eff\_len;\

}

#define feedback\_fail() { \

sprintf((char\*)s\_buf.out\_buf,"fail\r\n"); \

s\_buf.eff\_len = strlen((char\*)s\_buf.out\_buf); \

memcpy(&m\_gSndCommBuffer2[m\_gSndIndex], s\_buf.out\_buf, s\_buf.eff\_len); \

m\_gSndIndex2 += s\_buf.eff\_len;\

}

static char line[MINMEA\_MAX\_LENGTH];

Void task\_Communication(UArg arg0, UArg arg1)

{

INT16U nRcvCount = 0,i = 0,j = 0,tmp = 0;

INT16U nIndex = 0;

INT16U send\_flag\_ack = FALSE;

INT16U send\_flag\_broadcast\_flag = FALSE;

ULong start = 0;

ULong end = 0;

INT16U com\_test\_recv = 0;

INT16U com\_test\_send = 0;

comm\_src=UNKNOWN; //ACU; //

for(;;)

{

start = Timestamp\_get32();

#ifdef COMM\_PROTOCOL\_ATSING

tmp = 0;

nRcvCount = SciDriver\_Read(g\_AntCtrlIntBlock.m\_commu, &m\_gRcvCommBuffer[m\_gRcvIndex]);

if(com\_test\_recv)

{

memcpy(m\_gSndCommBuffer2,m\_gRcvCommBuffer, nRcvCount);

SciDriver\_Write(g\_AntCtrlIntBlock.m\_commu\_test, m\_gSndCommBuffer2, nRcvCount);

}

m\_gRcvIndex += nRcvCount;

//最短的协议长度为7个 ,51协议的

if ((m\_gRcvCommBuffer[0] == COMMAND\_HEAD) && (m\_gRcvIndex >= 7))

{

nIndex = 5;

for (; nIndex < m\_gRcvIndex; nIndex++)

{

//(1)刚好够一条报文长度，(2)大于一条报文的长度

if (m\_gRcvCommBuffer[nIndex - 1] == ENTER && m\_gRcvCommBuffer[nIndex] == NEWLINE)

{

//报文协议长度

nRcvCount = nIndex + 1;

memcpy(r\_buf.out\_buf,m\_gRcvCommBuffer, nRcvCount);

r\_buf.eff\_len = nRcvCount;

send\_flag\_ack = ParsePacket(&r\_buf,&s\_buf);

//用于检测到符合报文协议的，但没有匹配的，接收到什么就返回什么2015.7.15

//如果有多余的报文，存入缓冲区中

for (nIndex = nRcvCount; nIndex < m\_gRcvIndex; nIndex++)

{

m\_gRcvCommBuffer[nIndex - nRcvCount] = m\_gRcvCommBuffer[nIndex];

}

m\_gRcvIndex -= nRcvCount;

break;

}

}

}

if(m\_gRcvIndex > 0 && (m\_gRcvCommBuffer[0] != COMMAND\_HEAD))

{

for(i=0;i<m\_gRcvIndex;i++)

{

if(m\_gRcvCommBuffer[i] == COMMAND\_HEAD)

{

tmp = i;

//第一次找到的剩余数据存入到缓冲区中

for(j=tmp;j<m\_gRcvIndex;j++)

m\_gRcvCommBuffer[j-tmp] = m\_gRcvCommBuffer[j];

break;

}

}

if(tmp == i)

m\_gRcvIndex = m\_gRcvIndex - tmp;

else

m\_gRcvIndex = 0;

}

//防止ACU持续发送非法报文，造成接收缓冲区溢出

if(m\_gRcvIndex >= MAX\_BUFFER-20)

m\_gRcvIndex = 0;

/\*每次只发送一个报文\*/

m\_gSndIndex = 0;

if(TRUE == send\_flag\_ack)//发送正确报文协议

{

send\_flag\_ack = FALSE;

memcpy(&m\_gSndCommBuffer[m\_gSndIndex], s\_buf.out\_buf, s\_buf.eff\_len\*sizeof(s\_buf.out\_buf[0]));

m\_gSndIndex += s\_buf.eff\_len\*sizeof(s\_buf.out\_buf[0]);

}

if(TRUE == send\_flag\_broadcast\_flag)//用于发送62 、63协议

{

send\_flag\_broadcast\_flag = FALSE;

memcpy(&m\_gSndCommBuffer[m\_gSndIndex], s\_buf\_broadcast.out\_buf, s\_buf\_broadcast.eff\_len\*sizeof(s\_buf.out\_buf[0]));

m\_gSndIndex += s\_buf\_broadcast.eff\_len\*sizeof(s\_buf\_broadcast.out\_buf[0]);

}

SciDriver\_Write(g\_AntCtrlIntBlock.m\_commu, m\_gSndCommBuffer, m\_gSndIndex);

if(com\_test\_send)

SciDriver\_Write(g\_AntCtrlIntBlock.m\_commu\_test, m\_gSndCommBuffer, m\_gSndIndex);

#endif

#ifdef COMM\_PROTOCOL\_ATSING\_NMEA

if(!com\_test\_send && !com\_test\_recv)

{

//nmea类似协议发送

INT8S \*pHead = 0, \*pEnd = 0;

INT16S msg\_start = FALSE, msg\_end = FALSE, msg\_valid = FALSE;

INT16U msg\_counter = 0, msg\_err\_counter=0; //用于记录处理过多少条完整的GPS信息

static INT16U msg\_odd\_num2=0;

tmp = 0;

nRcvCount = SciDriver\_Read(g\_AntCtrlIntBlock.m\_commu\_test, &m\_gRcvCommBuffer2[msg\_odd\_num2]);

m\_gRcvIndex2 =msg\_odd\_num2 + nRcvCount;

msg\_odd\_num2 = 0;

m\_gSndIndex2 = 0;

nIndex = 0;

for (; nIndex < m\_gRcvIndex2; nIndex++)

{

if(m\_gRcvCommBuffer2[nIndex] == '$' || m\_gRcvCommBuffer2[nIndex] == '#' )

{

pHead = &m\_gRcvCommBuffer2[nIndex];

msg\_start = TRUE;

}

if((m\_gRcvCommBuffer2[nIndex] == '\r') && (m\_gRcvCommBuffer2[nIndex+1] == '\n'))

{

if(msg\_start) //只有结尾，继续寻找；

{

pEnd = &m\_gRcvCommBuffer2[nIndex+1];

msg\_end = TRUE;

//break;

}

}

if(msg\_end)

{

memcpy(line, pHead,pEnd-pHead+1);

line[pEnd-pHead+1] = NULL;

//SciDriver\_Write(g\_AntCtrlIntBlock.m\_commu\_test, (INT8S\*)line, pEnd-pHead+1);

msg\_start = 0;

msg\_end = 0;

msg\_valid = TRUE;

}

if(msg\_valid)

{

msg\_counter++;

msg\_valid = FALSE;

switch (minmea\_sentence\_id(line, false)) {

case MINMEA\_SENTENCE\_SAVE:{

struct minmea\_sentence\_save frame;

if (minmea\_parse\_saveconfig(&frame, line)) {

if(frame.save\_type == EEPROM\_MAIN)

{

DEBUG\_PRINTF("saveconfig main\r\n");

if(WriteConfigData(EEPROM\_MAIN))

{

feedback\_ok();

}

else

{

feedback\_fail();

}

}

else if(frame.save\_type == EEPROM\_BAK)

{

DEBUG\_PRINTF("saveconfig bak\r\n");

if(WriteConfigData(EEPROM\_BAK))

{

feedback\_ok();

}

else

{

feedback\_fail();

}

}

}

} break;

case MINMEA\_SENTENCE\_FRESET:{

struct minmea\_sentence\_save frame;

if (minmea\_parse\_freset(&frame,line))

{

if(frame.save\_type == EEPROM\_DEFAULT)

{

DEBUG\_PRINTF("use default data!\r\n");

use\_default\_data();

if(!WriteConfigData(EEPROM\_MAIN))

{

g\_AntCtrlIntBlock.m\_eeprom\_status\_flag=0;

DEBUG\_PRINTF("Write config data main Fail!\r\n");

}

else

{

feedback\_ok();

}

}

else if(frame.save\_type == EEPROM\_BAK)

{

if(ReadConfigData(EEPROM\_BAK))

{

if(WriteConfigData(EEPROM\_BAK))

{

feedback\_ok();

}

else

{

feedback\_fail();

}

DEBUG\_PRINTF("use default data!\r\n");

}

else

DEBUG\_PRINTF("read bak data fail !\r\n");

}

}

} break;

case MINMEA\_SENTENCE\_RESET:{

if (minmea\_parse\_reset(line)) {

DEBUG\_PRINTF("reset\r\n");

feedback\_ok();

watchdog\_service\_flag = 0;

//复位

//g\_AntCtrlIntBlock.m\_enumCmdStatusAnt\_Commu = ANT\_INIT;

}

} break;

case MINMEA\_SENTENCE\_COM:

{

struct minmea\_sentence\_com frame;

if (minmea\_parse\_com(&frame, line)) {

feedback\_ok();

g\_AntCtrlIntBlock.m\_commu\_baudrate = frame.badurate;

g\_AntCtrlIntBlock.m\_commu\_test = SciDriver\_Open(SCIPORT\_B, g\_AntCtrlIntBlock.m\_commu\_baudrate, SCIDATA\_8, SCISTOPBIT\_1, SCIPARITY\_NONE);

}

} break;

case MINMEA\_SENTENCE\_LOG:{

if (minmea\_parse\_log(&s\_buf, line)) {

s\_buf.eff\_len = strlen((char\*)s\_buf.out\_buf);

memcpy(&m\_gSndCommBuffer2[m\_gSndIndex], s\_buf.out\_buf, s\_buf.eff\_len);

m\_gSndIndex2 += s\_buf.eff\_len;

}

} break;

case MINMEA\_SENTENCE\_UNLOG:{

if (minmea\_parse\_unlog(line)) {

;

}

} break;

case MINMEA\_SENTENCE\_CONFIG:

{

if (minmea\_parse\_config(line)) {

;

}

} break;

case MINMEA\_SENTENCE\_SET:

{

if (minmea\_parse\_set(line)) {

;

}

}

break;

case MINMEA\_SENTENCE\_CMD:

{

if (minmea\_parse\_cmd(line)) {

feedback\_ok();

}

}

break;

case MINMEA\_SENTENCE\_GGA: {

struct minmea\_sentence\_gga frame;

if (minmea\_parse\_gga(&frame, line)) {

if(frame.fix\_quality>0)

{

GPS\_Data.Gps\_Sat\_num = frame.satellites\_tracked;

GPS\_Data.m\_fAltitude = minmea\_tofloat(&frame.altitude);

GPS\_Data.m\_fLatitude = minmea\_tocoord(&frame.latitude);

GPS\_Data.m\_fLongitude = minmea\_tocoord(&frame.longitude);

GPS\_Data.m\_nAVFlag = 'A';

}

else

GPS\_Data.m\_nAVFlag = 'V';

// msg\_rtn |=1;

}

if(gbroadcastTimerCnt[BROADCAST\_INDEX\_GPGGA].thr>0)

strcat(m\_gSndCommGnssBuffer, line);

} break;

case MINMEA\_SENTENCE\_RMC: {

struct minmea\_sentence\_rmc frame;

if (minmea\_parse\_rmc(&frame, line)) {

static INT8U msg\_err\_cnt = 0;

if(frame.valid)

{

GPS\_Data.m\_fLatitude = minmea\_tocoord(&frame.latitude);

GPS\_Data.m\_fLongitude = minmea\_tocoord(&frame.longitude);

GPS\_Data.m\_nAVFlag = 'A';

}

else

GPS\_Data.m\_nAVFlag = 'V';

GPS\_Data.m\_oTime.m\_year = frame.date.year;

GPS\_Data.m\_oTime.m\_month = frame.date.month;

GPS\_Data.m\_oTime.m\_day = frame.date.day;

GPS\_Data.m\_oTime.m\_nHour = frame.time.hours;

GPS\_Data.m\_oTime.m\_nMin = frame.time.minutes;

GPS\_Data.m\_oTime.m\_nSecond = frame.time.seconds;

GPS\_Data.m\_oTime.m\_nMillisecond = frame.time.microseconds;

// msg\_rtn |=1;

if(GPS\_Data.m\_nAVFlag == 'A'){

msg\_err\_cnt = 0;

GPS\_Data.gps\_status = GPS\_STATUS\_LOCATED;//已经定位

}

else{

if(msg\_err\_cnt++>10)

GPS\_Data.gps\_status = GPS\_STATUS\_UNLOCATED;

}

}

if(gbroadcastTimerCnt[BROADCAST\_INDEX\_GPRMC].thr>0)

strcat(m\_gSndCommGnssBuffer, line);

} break;

case MINMEA\_SENTENCE\_HDT: {

struct minmea\_sentence\_hdt frame;

if (minmea\_parse\_hdt(&frame, line)) {

if(frame.heading.scale>0)

{

GPS\_Data.m\_fHeading = minmea\_tofloat(&frame.heading);

GPS\_Data.m\_HeadingFlag.bit.HDT = TRUE;

}

// msg\_rtn |=1;

}

} break;

case MINMEA\_SENTENCE\_HEADINGA: {

struct minmea\_sentence\_headinga frame;

if (minmea\_parse\_headinga(&frame, line)) {

if(frame.sol\_stat)

{

GPS\_Data.m\_fHeading = minmea\_tofloat(&frame.heading);

GPS\_Data.m\_length = minmea\_tofloat(&frame.length);

GPS\_Data.m\_HeadingFlag.bit.HEADINGA = TRUE;

}

// msg\_rtn |=1;

//GPS\_Data.m\_fHeading = minmea\_tofloat(&frame.heading);

}

} break;

case MINMEA\_SENTENCE\_GINS: {

struct minmea\_sentence\_gins frame;

if (minmea\_parse\_gins(&frame, line)) {

GPS\_Data.m\_roll = minmea\_tofloat(&frame.m\_roll);

GPS\_Data.m\_pitch = minmea\_tofloat(&frame.m\_pitch);

GPS\_Data.m\_yaw = minmea\_tofloat(&frame.m\_yaw);

GPS\_Data.m\_fHeading = minmea\_tofloat(&frame.m\_heading);

GPS\_Data.Gps\_Stat = frame.m\_sat\_nums;

GPS\_Data.m\_acc[0] = minmea\_tofloat(&frame.m\_acc[0]);

GPS\_Data.m\_acc[1] = minmea\_tofloat(&frame.m\_acc[1]);

GPS\_Data.m\_acc[2] = minmea\_tofloat(&frame.m\_acc[2]);

GPS\_Data.m\_gyro[0] = minmea\_tofloat(&frame.m\_gyro[0]);

GPS\_Data.m\_gyro[1] = minmea\_tofloat(&frame.m\_gyro[1]);

GPS\_Data.m\_gyro[2] = minmea\_tofloat(&frame.m\_gyro[2]);

GPS\_Data.m\_mag[0] = minmea\_tofloat(&frame.m\_mag[0]);

GPS\_Data.m\_mag[1] = minmea\_tofloat(&frame.m\_mag[1]);

GPS\_Data.m\_mag[2] = minmea\_tofloat(&frame.m\_mag[2]);

GPS\_Data.m\_temp = minmea\_tofloat(&frame.m\_temp);

GPS\_Data.m\_interval = frame.m\_interval;

/\*GPS\_Data.m\_fLatitude = minmea\_tocoord(&frame.latitude);

GPS\_Data.m\_fLongitude = minmea\_tocoord(&frame.longitude);

GPS\_Data.m\_fAltitude = minmea\_tofloat(&frame.altitude);

if(GPS\_Data.m\_fLatitude <0)

GPS\_Data.m\_LatitudeFlag = SOUTH;

else

GPS\_Data.m\_LatitudeFlag = NORTH;

if(GPS\_Data.m\_fLongitude <0)

GPS\_Data.m\_LongitudeFalg = WEST;

else

GPS\_Data.m\_LongitudeFalg = EAST;

if(GPS\_Data.Gps\_Sat\_num <=2 )

GPS\_Data.m\_nAVFlag = 'V';

else

GPS\_Data.m\_nAVFlag = 'A';

\*/

GPS\_Data.m\_HeadingFlag.bit.MAGN = TRUE;

GPS\_Data.m\_ins\_heartbeat = TRUE;

// msg\_rtn |=1;

}

if(gbroadcastTimerCnt[BROADCAST\_INDEX\_GINS].thr>0)

strcat(m\_gSndCommGnssBuffer, line);

} break;

case MINMEA\_INVALID:

case MINMEA\_UNKNOWN:

msg\_err\_counter++;

break;

default: {

msg\_err\_counter++;

} break;

}

}

}

if(msg\_start) //有头没尾，保存

{

msg\_start = 0;

msg\_odd\_num2 = &m\_gRcvCommBuffer2[m\_gRcvIndex2] - pHead; //包括前面的'$'

if(msg\_odd\_num2>MAX\_BUFFER/2)

msg\_odd\_num2 = 0;

memcpy(m\_gRcvCommBuffer2, pHead, msg\_odd\_num2);

}

if((gbroadcastTimerCnt[BROADCAST\_INDEX\_INFO].thr > 0) &&(gbroadcastTimerCnt[BROADCAST\_INDEX\_INFO].cnt++ >= gbroadcastTimerCnt[BROADCAST\_INDEX\_INFO].thr))

{

gbroadcastTimerCnt[BROADCAST\_INDEX\_INFO].cnt = 0;

minmea\_generate\_info((char\*)s\_buf\_broadcast.out\_buf);

s\_buf\_broadcast.eff\_len = strlen((char\*)s\_buf\_broadcast.out\_buf);

// do\_snd\_fun\_62(&s\_buf\_broadcast);

// s\_buf\_broadcast.eff\_len = send\_packet\_trans\_mean((INT8S \*)s\_buf\_broadcast.out\_buf, (INT8U)s\_buf\_broadcast.eff\_len);//发送转译

memcpy(&m\_gSndCommBuffer2[m\_gSndIndex2], s\_buf\_broadcast.out\_buf, s\_buf\_broadcast.eff\_len\*sizeof(s\_buf.out\_buf[0]));

m\_gSndIndex2 += s\_buf\_broadcast.eff\_len\*sizeof(s\_buf\_broadcast.out\_buf[0]);

}

if((gbroadcastTimerCnt[BROADCAST\_INDEX\_RAWIMU].thr > 0) &&(gbroadcastTimerCnt[BROADCAST\_INDEX\_RAWIMU].cnt++ >= gbroadcastTimerCnt[BROADCAST\_INDEX\_RAWIMU].thr))

{

gbroadcastTimerCnt[BROADCAST\_INDEX\_RAWIMU].cnt = 0;

minmea\_generate\_rawimu((char\*)s\_buf\_broadcast.out\_buf);

s\_buf\_broadcast.eff\_len = strlen((char\*)s\_buf\_broadcast.out\_buf);

// do\_snd\_fun\_62(&s\_buf\_broadcast);

// s\_buf\_broadcast.eff\_len = send\_packet\_trans\_mean((INT8S \*)s\_buf\_broadcast.out\_buf, (INT8U)s\_buf\_broadcast.eff\_len);//发送转译

memcpy(&m\_gSndCommBuffer2[m\_gSndIndex2], s\_buf\_broadcast.out\_buf, s\_buf\_broadcast.eff\_len\*sizeof(s\_buf.out\_buf[0]));

m\_gSndIndex2 += s\_buf\_broadcast.eff\_len\*sizeof(s\_buf\_broadcast.out\_buf[0]);

}

if((gbroadcastTimerCnt[BROADCAST\_INDEX\_GPFPD].thr > 0) &&(gbroadcastTimerCnt[BROADCAST\_INDEX\_GPFPD].cnt++ >= gbroadcastTimerCnt[BROADCAST\_INDEX\_GPFPD].thr))

{

gbroadcastTimerCnt[BROADCAST\_INDEX\_GPFPD].cnt = 0;

minmea\_generate\_gpfpd((char\*)s\_buf\_broadcast.out\_buf);

s\_buf\_broadcast.eff\_len = strlen((char\*)s\_buf\_broadcast.out\_buf);

// do\_snd\_fun\_62(&s\_buf\_broadcast);

// s\_buf\_broadcast.eff\_len = send\_packet\_trans\_mean((INT8S \*)s\_buf\_broadcast.out\_buf, (INT8U)s\_buf\_broadcast.eff\_len);//发送转译

memcpy(&m\_gSndCommBuffer2[m\_gSndIndex2], s\_buf\_broadcast.out\_buf, s\_buf\_broadcast.eff\_len\*sizeof(s\_buf.out\_buf[0]));

m\_gSndIndex2 += s\_buf\_broadcast.eff\_len\*sizeof(s\_buf\_broadcast.out\_buf[0]);

}

if(m\_gSndCommGnssBuffer[0] != NULL)

{

INT16U len = strlen(m\_gSndCommGnssBuffer);

memcpy(&m\_gSndCommBuffer2[m\_gSndIndex2], m\_gSndCommGnssBuffer, len);

m\_gSndIndex2 += len;

m\_gSndCommGnssBuffer[0] = NULL;

}

if(m\_gSndCommDbgBuffer[0] != NULL)

{

INT16U len = strlen(m\_gSndCommDbgBuffer);

memcpy(&m\_gSndCommBuffer2[m\_gSndIndex2], m\_gSndCommDbgBuffer, len);

m\_gSndIndex2 += len;

m\_gSndCommDbgBuffer[0] = NULL;

}

SciDriver\_Write(g\_AntCtrlIntBlock.m\_commu\_test, m\_gSndCommBuffer2, m\_gSndIndex2);

#endif

}

#if 0

nRcvCount = SciDriver\_Read(g\_AntCtrlIntBlock.m\_commu\_test, &m\_gRcvCommBuffer2[m\_gRcvIndex]);

m\_gRcvIndex2 +=nRcvCount;

//最短的协议长度为7个 ,51协议的

if ((m\_gRcvCommBuffer2[0] == COMMAND\_HEAD) && (m\_gRcvIndex2 >= 7))

{

nIndex = 5;

for (; nIndex < m\_gRcvIndex2; nIndex++)

{

//(1)刚好够一条报文长度，(2)大于一条报文的长度

if (m\_gRcvCommBuffer2[nIndex - 1] == ENTER && m\_gRcvCommBuffer2[nIndex] == NEWLINE)

{

//报文协议长度

nRcvCount = nIndex + 1;

memcpy(r\_buf.out\_buf,m\_gRcvCommBuffer2, nRcvCount);

r\_buf.eff\_len = nRcvCount;

send\_flag\_ack = ParsePacket(&r\_buf,&s\_buf);

//用于检测到符合报文协议的，但没有匹配的，接收到什么就返回什么2015.7.15

//如果有多余的报文，存入缓冲区中

for (nIndex = nRcvCount; nIndex < m\_gRcvIndex2; nIndex++)

{

m\_gRcvCommBuffer2[nIndex - nRcvCount] = m\_gRcvCommBuffer2[nIndex];

}

m\_gRcvIndex2 -= nRcvCount;

break;

}

}

}

if(m\_gRcvIndex2 > 0 && (m\_gRcvCommBuffer2[0] != COMMAND\_HEAD))

{

for(i=0;i<m\_gRcvIndex2;i++)

{

if(m\_gRcvCommBuffer2[i] == COMMAND\_HEAD)

{

tmp = i;

//第一次找到的剩余数据存入到缓冲区中

for(j=tmp;j<m\_gRcvIndex2;j++)

m\_gRcvCommBuffer2[j-tmp] = m\_gRcvCommBuffer2[j];

break;

}

}

if(tmp == i)

m\_gRcvIndex2 = m\_gRcvIndex2 - tmp;

else

m\_gRcvIndex2 = 0;

}

send\_flag\_broadcast\_flag =FALSE;

if((g\_nBroadcastTimer\_fun62\_cnt > 0) &&(g\_nBroadcastTimer\_fun62++ >= g\_nBroadcastTimer\_fun62\_cnt))

{

g\_nBroadcastTimer\_fun62 = 0;

do\_snd\_fun\_62(&s\_buf\_broadcast);

s\_buf\_broadcast.eff\_len = send\_packet\_trans\_mean((INT8S \*)s\_buf\_broadcast.out\_buf, (INT8U)s\_buf\_broadcast.eff\_len);//发送转译

//sprintf((char\*)m\_gSndch, "%d\t%.1f\t%d\t%d\t%d\t%d\r\n",

// moto\_chassis[0].speed\_rpm,canmotor\_set\_spd[0], moto\_chassis[0].given\_current, moto\_chassis[0].real\_current,

// (INT16S)moto\_chassis[0].round\_cnt, moto\_chassis[0].angle); //

//SciDriver\_WriteStr(g\_AntCtrlIntBlock.m\_commu, m\_gSndch);

send\_flag\_broadcast\_flag = TRUE;

}

//防止ACU持续发送非法报文，造成接收缓冲区溢出

if(m\_gRcvIndex2 >= MAX\_BUFFER-20)

m\_gRcvIndex2 = 0;

/\*每次只发送一个报文\*/

m\_gSndIndex2 = 0;

if(TRUE == send\_flag\_ack)//发送正确报文协议

{

send\_flag\_ack = FALSE;

memcpy(&m\_gSndCommBuffer[m\_gSndIndex2], s\_buf.out\_buf, s\_buf.eff\_len\*sizeof(s\_buf.out\_buf[0]));

m\_gSndIndex2 += s\_buf.eff\_len\*sizeof(s\_buf.out\_buf[0]);

}

if(TRUE == send\_flag\_broadcast\_flag)//用于发送62 、63协议

{

send\_flag\_broadcast\_flag = FALSE;

memcpy(&m\_gSndCommBuffer[m\_gSndIndex2], s\_buf\_broadcast.out\_buf, s\_buf\_broadcast.eff\_len\*sizeof(s\_buf.out\_buf[0]));

m\_gSndIndex2 += s\_buf\_broadcast.eff\_len\*sizeof(s\_buf\_broadcast.out\_buf[0]);

}

#ifndef BEACON\_KC

SciDriver\_Write(g\_AntCtrlIntBlock.m\_commu\_test, m\_gSndCommBuffer, m\_gSndIndex2);

#endif

#endif

end = Timestamp\_get32();

cpuload[TASK\_ID\_COMMUNICATION] = end - start;

run\_time[TASK\_ID\_COMMUNICATION] = (end - start)/(FP64)freq.lo \* 1000;//ms

Task\_sleep(TASK\_COMMUNICATION\_DELAY);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef \_\_GET\_MSG\_QUE\_H\_\_

#define \_\_GET\_MSG\_QUE\_H\_\_

//0x10

int TianXianZhuangTai(char cData);

//0x13

int GpsWeiZhiPeiZhi(char \* cStatus, float \* fJingDu, float \* WeiDu, char \* cType);

//0x15

int GpsShuChuPeiZhi(char cOutput, char cXieYi, char cTime);

//0x16

int ModemPeiZhi(char cType, char cPort, char cProtol, char cTime);

//0x17

int PoShuPeiZhi(char cAutoColor, char cAutoRf);

//0x18

int LuoJingPeiZhi(char cType, char \* pName);

//0x19

int BoShuYanSe(char cSubCode, char \* pYanSe);

//0x1A

int ShengJi();

//0x20

int FangWeiJiaoZheng(char cMod, float \* fData);

//0x21

int FangXiangBianMaQi(short \* pZongZhuanShu, int \* pMaPanJiaoDu);

//0x22

int FangXiangTuoLuo(char cData, short sAngle, short \* pRetAngle);

//0x23

int FangXiangDianJiQuDongQi(char cData, short sZhanKongBi, short \* pZhanKongBi);

//0x24

int FuYangBianMaQi(char cData, short \* pZongZhuanShu, int \* pMaPanJiaoDu);

//0x25

int FuYangDianJiQuDongQi(char cData, short sZhanKongBi, short \* pZhanKongBi);

//0x26

int HengGunBianMaQi(char cData, short \* pZongZhuanShu, int \* pMaPanJiaoDu);

//0x27

int HengGunDianJiQuDongQi(char cData, short sZhanKongBi, short \* pZhanKongBi);

//0x28

int BaoZhaKongZhi(char cHengGunBaoZha, char cFuYangBaoZha, char \* pHengGunBaoZha, char \* pFuYangBaoZha);

//0x29

int XianWeiZhuangTai(char cStatus, char \*pStatus, char \* pFuYangZheng, char \*pFuYangFu,

char \*pHengGunZheng, char \* pHengGunFu);

//0x2A

int JiHuaJiaoDu(char cStatus, float \* fJiHuaJiaoDu);

//0x30

int QingXieYiLingDian(char cHengGunStatus, short sHengGun, short \* pHengGun,

char cFuYangStatus, short sFuYang, short \*pFuYang,

char cFangWeiSgtatus, short sFangWei, short \*pFangWei);

//0x31

int TuoLuoLingDian(char cHengGunStatus, short sHengGun, short \* pHengGun,

char cFuYangStatus, short sFuYang, short \*pFuYang,

char cFangWeiSgtatus, short sFangWei, short \*pFangWei);

//0x32

int FuYangJiaoZheng(char cStatus, float \* pData);

//0x33

int QingXieYiYuanShiShuJu(char cHengGunStatus, short sHengGun, short \* pHengGun,

char cFuYangStatus, short sFuYang, short \*pFuYang,

char cFangWeiSgtatus, short sFangWei, short \*pFangWei);

//0x34

int TuoLuoYiYuanShiShuJu(char cHengGunStatus, short sHengGun, short \* pHengGun,

char cFuYangStatus, short sFuYang, short \*pFuYang,

char cFangWeiSgtatus, short sFangWei, short \*pFangWei);

//0x35

int LuoJingShuJu(char cData, char \*pLuoJingType, short \* pData);

//0x36

int get\_compass\_angle(int angle, char flag);

int LuoJingShuJuLuoPan(char cData, char \*pLuoJingType, short \* pData);

//0x37

int XinBiaoJieShouJi(char cData, char \*pXinBiaoJieShouJiStatus, char \*pXinBiaoShiBieStatus, short \* pXinHaoQiangDu);

//0x38

int DVBJieShouJi(char cData,char \*pDVBJieShouJiStatus, char \*pXinBiaoShiBieStatus, short \* pXinHaoQiangDu);

//0x39

int GongFangKongZhi(char cSubCode, char \* pFaSheYunXu);

//0x3A

int LNBKongZhi(char cSubCode, float \* pPinLv13v0k, float \* pPinLv13v22k, float \* pPinLv18v0k, float \* pPinLv18v22k, char \* cDianYa);

//0x40

int WenDuChuanGanQi(char \*pStatus, char \*pWenDu);

//0x41

int PowerMonitor(char \* p5vStatus, short \*p5vValue, char \* p24vStatus, short \*p24vValue);

//0x42

int ConfigData(char cData, short sDataAddr, short sData,

short \* pDataAddr, short \* pData);

//0x43

int GetVersion(char cData, short \* pVersion);

//0x45

int FangWeiShouCang(char cData, float \* pData);

//0x46

int FuYangShouCang(char cData, float \* pData);

//0x47

int tel\_acu\_47(char cOutput, char flag);

//0x50

int MuBiaoWeiXing(char cFunSelect,

float \* pWeiXingJiingDu, char \* pJiHuaLeiXing, char \* pJiHuaFangShi,

char \* pJieShouJiLeiXing,float \* pXinBiaoPinLv, float \* pDvbPingLv,

int \* pDVBFuHaoLv, float \* pJiHuaPianZhiJiao, float \* pMenXian,

char \* cQingYong);

//0x51

int LiLunDuiXingShuJu(float \* pFangXiang, float \* pFuYang, float \*pJiHua);

//0x52

int MenXianPeiZhi(char cData, char \* pData);

//0x53还没写

int JieShouJiXinHaoYouXianJi(char cSubFunCode);

//0x54

int ZheDangPeiZhi(char cSubCode, float \* pFangWeiQiShi,

float \*pFangWeiJieShu, float \* pFuYangQiShi, float \* pFuYangJieShu);

//0x55

int ant\_serch\_angle(char cSubFunCode,float \*az\_search\_left, float \*az\_search\_right, float \*el\_search\_down, float \*el\_search\_up);

int SouSuoFanWei(char cSubFunCode, float \* pFwss\_fw, float \* pFyss\_fw, float \* pFybj\_jd,

float \* pFwss\_sd, float \* pFyss\_sd);

//0x56

int ZaoShengDianPing(char \* pZaoShengDianPingZhuangTai, short \* pZaoShengDianPingShuJu,

char \* pGenZongDianPingZhuangTai, short \* pGenZongDianPingShuJu);

//0x58

int TianXianMianSaoMian(char cSubFun,

float \* pFangWeiJIaoDu, float \* pFuYangJIaoDu,

float \* pMenXian, short \* pTime);

//0x59

int ShouDongKongZhiBuChangSuDu(char cType1, char cType2, float \* data);

//0x5A

int ShouDongKongZhiSanZhou(char cSubFunCode, char cStep);

#endif

#include <stdio.h>

#include "message\_queues.h"

union

{

struct{

unsigned char c0;

unsigned char c1;

unsigned char c2;

unsigned char c3;

}s4;

unsigned short usData;

short sData;

unsigned int unData;

int nData;

float fData;

}uData;

void ushort2char(unsigned short data, char \* pData)

{

uData.usData = data;

\* pData++ = uData.s4.c0;

\* pData++ = uData.s4.c1;

}

unsigned short char2ushort(char \* pData)

{

uData.s4.c0 = \* pData++;

uData.s4.c1 = \* pData++;

return uData.usData;

}

void short2char(short data, char \* pData)

{

uData.sData = data;

\* pData++ = uData.s4.c0;

\* pData++ = uData.s4.c1;

}

short char2short(char \* pData)

{

uData.s4.c0 = \* pData++;

uData.s4.c1 = \* pData++;

return uData.sData;

}

void uint2char(unsigned int data, char \* pData)

{

uData.unData = data;

\* pData++ = uData.s4.c0;

\* pData++ = uData.s4.c1;

\* pData++ = uData.s4.c2;

\* pData++ = uData.s4.c3;

}

unsigned short char2uint(char \* pData)

{

uData.s4.c0 = \* pData++;

uData.s4.c1 = \* pData++;

uData.s4.c2 = \* pData++;

uData.s4.c3 = \* pData++;

return uData.unData;

}

void int2char(int data, char \* pData)

{

uData.nData = data;

\* pData++ = uData.s4.c0;

\* pData++ = uData.s4.c1;

\* pData++ = uData.s4.c2;

\* pData++ = uData.s4.c3;

}

unsigned short char2int(char \* pData)

{

uData.s4.c0 = \* pData++;

uData.s4.c1 = \* pData++;

uData.s4.c2 = \* pData++;

uData.s4.c3 = \* pData++;

return uData.nData;

}

void float2char(float data, char \* pData)

{

uData.fData = data;

\* pData++ = uData.s4.c0;

\* pData++ = uData.s4.c1;

\* pData++ = uData.s4.c2;

\* pData++ = uData.s4.c3;

}

float char2float(char \* pData)

{

uData.s4.c0 = \* pData++;

uData.s4.c1 = \* pData++;

uData.s4.c2 = \* pData++;

uData.s4.c3 = \* pData++;

return uData.fData;

}

/\*

\* 发送时，查找是否有需要转义，有转义返回转义后增加长度，无转义返回0

\* pbuf,转义前后的协议buf，nLen转义前后的协议长度

\* return 转义后协议pbuf长度

\*/

int send\_packet\_trans\_mean(char \*pbuf, int nLen)

{

int i = 0, i\_tmp = 0, loop = 0;

for(i = 3; i < ((nLen-2)+loop); i++)

{

if(pbuf[i] == 0x0D)

{

pbuf[i] = 0x0E;

for(i\_tmp = (nLen+loop); i\_tmp > i; i\_tmp--){

pbuf[i\_tmp + 1] = pbuf[i\_tmp];

}

pbuf[i+1] = 0x2D;

loop++;

}else if((pbuf[i] == 0x0E)){

for(i\_tmp = (nLen+loop); i\_tmp > i; i\_tmp--){

pbuf[i\_tmp + 1] = pbuf[i\_tmp];

}

pbuf[i+1] = 0x2E;

loop++;

}else

;

}

return (loop+nLen);

}

/\*

\* 功能说明:1，转义

\* 用在接收处理数据

\* pbuf,转义前后的协议buf，nLen转义前后的协议长度

\* return 转义后协议pbuf长度

\*/

int recv\_packet\_trans\_mean(char \* pbuf, int nLen)

{

int i = 0, i\_tmp = 0, loop = 0;

for(i = 3; i < nLen-2; i++)

{

if(pbuf[i] == 0x0E){

if(pbuf[i+1] == 0x2D){

pbuf[i] = 0x0D;

for(i\_tmp = (i+1); i\_tmp < (nLen-loop); i\_tmp++){

pbuf[i\_tmp] = pbuf[i\_tmp + 1];

}

loop++;

}else if(pbuf[i+1] == 0x2E){

for(i\_tmp = (i+1); i\_tmp < (nLen-loop); i\_tmp++){

pbuf[i\_tmp] = pbuf[i\_tmp + 1];

}

loop++;

}else

;

}else

;

}

return (nLen-loop);

}

int MsgQueSendAndRead(char \* pSendData, const int nSendDataLen,

char \*pReadData, const int nReadDataLen)

{

static int nInit = 0;

int nSendLen = -1;

int nReadLen = -1;

int nRet = 0;

int nTemp = 0;

if (nInit == 0)

{

nRet = MsgQueInit();

if (nRet != 0)

{

TraceLog("MsgQueInit error\n");

return -1;

}

nInit = 1;

}

if ( (pSendData == NULL) || (nSendDataLen <= 0)

|| (pReadData == NULL) || (nReadDataLen <= 0) )

{

TraceLog("MsgQueSendAndRead param error\n");

return -1;

}

nTemp = send\_packet\_trans\_mean(pSendData, nSendDataLen);

nSendLen = MsgQueSend(pSendData, nTemp, 11);

if (nSendLen != nTemp)

{

TraceLog("MsgQueSend error\n");

return -1;

}

if(pSendData[2] == 0x50 || pSendData[2] == 0x59){

nReadLen = MsgQueRead(pReadData, nReadDataLen, 0x90 +pSendData[4]);

}else

nReadLen = MsgQueRead(pReadData, nReadDataLen, pSendData[2]);

if (nSendLen <= 0)

{

TraceLog("msg\_queue\_read error\n");

return -1;

}

recv\_packet\_trans\_mean(pReadData, nReadLen);

return 0;

}

static int DelMeg(char \*pMsgSendData)

{

int nDataFieldLen = 0;

unsigned char nSum = 0;

int i = 0;

if (pMsgSendData == NULL)

{

return -1;

}

pMsgSendData[0] = ':';

pMsgSendData[1] = 0X30;

nDataFieldLen = pMsgSendData[3];

for(i=0; i<nDataFieldLen+3; i++)

{

nSum += pMsgSendData[i+1];

}

pMsgSendData[nDataFieldLen+4] = nSum;

pMsgSendData[nDataFieldLen+5] = 0X0D;

pMsgSendData[nDataFieldLen+6] = 0X0A;

return 0;

}

//ok

int TianXianZhuangTai(char cData)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

short sTemp = 0;

int nRet = 0;

szMsgSendData[2] = 0x10; //功能码

szMsgSendData[3] = 0x01; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cData;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

#else

#endif

return 0;

}

int GpsWeiZhiPeiZhi(char \* cStatus, float \* fJingDu, float \* WeiDu, char \* cType)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

short sTemp = 0;

int nRet = 0;

szMsgSendData[2] = 0x13; //功能码

szMsgSendData[3] = 14; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = \*cStatus;

short2char((\* fJingDu) \* 100, szMsgSendData + 5);

short2char((\* WeiDu) \* 100, szMsgSendData + 7);

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\*cStatus = szMsgReadData[4];

sTemp = char2short(szMsgReadData + 5);

\* fJingDu = (float)(sTemp / 100.0);

sTemp = char2short(szMsgReadData + 7);

\* WeiDu = (float)(sTemp / 100.0);

\* cType = szMsgReadData[9];

#else

#endif

return 0;

}

//船首向校正配置与查询

int ChuanShouXiang(char cMod, short nAngle, short \* pRealAngle)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x20; //功能码

szMsgSendData[3] = 0x03; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cMod;

szMsgSendData[5] = nAngle>>8;

szMsgSendData[6] = nAngle;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pRealAngle = szMsgReadData[5]<<8 + szMsgReadData[5];

#else

\* pRealAngle = 1122;

#endif

return 0;

}

int FangXiangBianMaQi(short \* pZongZhuanShu, int \* pMaPanJiaoDu)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x21; //功能码

szMsgSendData[3] = 0x00; // 长度，不包括lcr，也不包括自己

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pZongZhuanShu = 0x00;

\* pMaPanJiaoDu = 0x00;

#else

\* pZongZhuanShu = 0x00;

\* pMaPanJiaoDu = 0x00;

#endif

return 0;

}

int FangXiangTuoLuo(char cData, short sAngle, short \* pRetAngle)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x20; //功能码

szMsgSendData[3] = 0x03; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cData;

szMsgSendData[5] = sAngle>>8;

szMsgSendData[6] = sAngle;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pRetAngle = 0x00;

#else

\* pRetAngle = 0x00;

#endif

return 0;

}

int FangXiangDianJiQuDongQi(char cData, short sZhanKongBi, short \* pZhanKongBi)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x23; //功能码

szMsgSendData[3] = 0x03; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cData;

szMsgSendData[5] = sZhanKongBi>>8;

szMsgSendData[6] = sZhanKongBi;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pZhanKongBi = 0x00;

#else

\* pZhanKongBi = 0x00;

#endif

return 0;

}

int FuYangBianMaQi(char cData, short \* pZongZhuanShu, int \* pMaPanJiaoDu)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x24; //功能码

szMsgSendData[3] = 0x01; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cData;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pZongZhuanShu = 0x00;

\* pMaPanJiaoDu = 0x00;

#else

\* pZongZhuanShu = 0x00;

\* pMaPanJiaoDu = 0x00;

#endif

return 0;

}

int FuYangDianJiQuDongQi(char cData, short sZhanKongBi, short \* pZhanKongBi)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x23; //功能码

szMsgSendData[3] = 0x03; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cData;

szMsgSendData[5] = sZhanKongBi>>8;

szMsgSendData[6] = sZhanKongBi;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pZhanKongBi = 0x00;

#else

\* pZhanKongBi = 0x00;

#endif

return 0;

}

int HengGunBianMaQi(char cData, short \* pZongZhuanShu, int \* pMaPanJiaoDu)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x26; //功能码

szMsgSendData[3] = 0x01; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cData;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pZongZhuanShu = 0x00;

\* pMaPanJiaoDu = 0x00;

#else

\* pZongZhuanShu = 0x00;

\* pMaPanJiaoDu = 0x00;

#endif

return 0;

}

int HengGunDianJiQuDongQi(char cData, short sZhanKongBi, short \* pZhanKongBi)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x27; //功能码

szMsgSendData[3] = 0x03; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cData;

szMsgSendData[5] = sZhanKongBi>>8;

szMsgSendData[6] = sZhanKongBi;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pZhanKongBi = 0x00;

#else

\* pZhanKongBi = 0x00;

#endif

return 0;

}

int BaoZhaKongZhi(char cHengGunBaoZha, char cFuYangBaoZha, char \* pHengGunBaoZha, char \* pFuYangBaoZha)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x28; //功能码

szMsgSendData[3] = 0x02; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cHengGunBaoZha;

szMsgSendData[5] = cFuYangBaoZha;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pHengGunBaoZha = 0x00;

\* pFuYangBaoZha = 0x00;

#else

\* pHengGunBaoZha = 0x00;

\* pFuYangBaoZha = 0x00;

#endif

return 0;

}

int XianWeiZhuangTai(char cStatus, char \*pStatus, char \* pFuYangZheng, char \*pFuYangFu,

char \*pHengGunZheng, char \* pHengGunFu)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x29; //功能码

szMsgSendData[3] = 0x01; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cStatus;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pStatus = 0x00;

\* pFuYangZheng = 0x00;

\* pFuYangFu = 0x00;

\* pHengGunZheng = 0x00;

\* pHengGunFu = 0x00;

#else

\* pStatus = 0x00;

\* pFuYangZheng = 0x00;

\* pFuYangFu = 0x00;

\* pHengGunZheng = 0x00;

\* pHengGunFu = 0x00;

#endif

return 0;

}

int JiHuaJiaoDu(char cStatus, float \* fJiHuaJiaoDu)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

int i =0;

szMsgSendData[2] = 0x2A; //功能码

szMsgSendData[3] = 0x03; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cStatus;

short2char(\*fJiHuaJiaoDu\*100, szMsgSendData+5);

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

#if 0

TraceLog("write:");

for(i=0; i<64; i++)

{

TraceLog("[%x]", szMsgSendData[i]);

}

TraceLog("\n");

#endif

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

#if 0

TraceLog("read:");

for(i=0; i<64; i++)

{

TraceLog("[%x]", szMsgReadData[i]);

}

TraceLog("\n");

#endif

\* fJiHuaJiaoDu = char2short(szMsgReadData+5)/100.0;

#else

\* fJiHuaJiaoDu = 4.4;

#endif

return 0;

}

int QingXieYiLingDian(char cHengGunStatus, short sHengGun, short \* pHengGun,

char cFuYangStatus, short sFuYang, short \*pFuYang,

char cFangWeiSgtatus, short sFangWei, short \*pFangWei)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x30; //功能码

szMsgSendData[3] = 0x09; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cHengGunStatus;

szMsgSendData[5] = sHengGun>>8;

szMsgSendData[6] = sHengGun;

szMsgSendData[7] = cFuYangStatus;

szMsgSendData[8] = sFuYang>>8;

szMsgSendData[9] = sFuYang;

szMsgSendData[10] = cFangWeiSgtatus;

szMsgSendData[11] = sFangWei>>8;

szMsgSendData[12] = sFangWei;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pHengGun = 0x00;

\* pFuYang = 0x00;

\* pFangWei = 0x00;

#else

\* pHengGun = 0x00;

\* pFuYang = 0x00;

\* pFangWei = 0x00;

#endif

return 0;

}

int TuoLuoLingDian(char cHengGunStatus, short sHengGun, short \* pHengGun,

char cFuYangStatus, short sFuYang, short \*pFuYang,

char cFangWeiSgtatus, short sFangWei, short \*pFangWei)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x31; //功能码

szMsgSendData[3] = 0x09; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cHengGunStatus;

szMsgSendData[5] = sHengGun>>8;

szMsgSendData[6] = sHengGun;

szMsgSendData[7] = cFuYangStatus;

szMsgSendData[8] = sFuYang>>8;

szMsgSendData[9] = sFuYang;

szMsgSendData[10] = cFangWeiSgtatus;

szMsgSendData[11] = sFangWei>>8;

szMsgSendData[12] = sFangWei;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, sizeof(szMsgSendData),

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pHengGun = 0x00;

\* pFuYang = 0x00;

\* pFangWei = 0x00;

#else

\* pHengGun = 0x00;

\* pFuYang = 0x00;

\* pFangWei = 0x00;

#endif

return 0;

}

int FuYangJiaoZheng(char cStatus, float \* pData)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

int i = 0;

szMsgSendData[2] = 0x32; //功能码

szMsgSendData[3] = 0x03; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cStatus;

short2char(\*pData\*100, szMsgSendData+5);

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

#if 0

TraceLog("write:");

for(i=0; i<64; i++)

{

TraceLog("[%x]", szMsgSendData[i]);

}

TraceLog("\n");

#endif

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

#if 0

TraceLog("read:");

for(i=0; i<64; i++)

{

TraceLog("[%x]", szMsgReadData[i]);

}

TraceLog("\n");

#endif

\* pData = char2short(szMsgReadData+5)/100.0;

#else

#endif

return 0;

}

int QingXieYiYuanShiShuJu(char cHengGunStatus, short sHengGun, short \* pHengGun,

char cFuYangStatus, short sFuYang, short \*pFuYang,

char cFangWeiSgtatus, short sFangWei, short \*pFangWei)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x31; //功能码

szMsgSendData[3] = 0x09; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cHengGunStatus;

szMsgSendData[5] = sHengGun>>8;

szMsgSendData[6] = sHengGun;

szMsgSendData[7] = cFuYangStatus;

szMsgSendData[8] = sFuYang>>8;

szMsgSendData[9] = sFuYang;

szMsgSendData[10] = cFangWeiSgtatus;

szMsgSendData[11] = sFangWei>>8;

szMsgSendData[12] = sFangWei;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pHengGun = 0x00;

\* pFuYang = 0x00;

\* pFangWei = 0x00;

#else

\* pHengGun = 0x00;

\* pFuYang = 0x00;

\* pFangWei = 0x00;

#endif

return 0;

}

int TuoLuoYiYuanShiShuJu(char cHengGunStatus, short sHengGun, short \* pHengGun,

char cFuYangStatus, short sFuYang, short \*pFuYang,

char cFangWeiSgtatus, short sFangWei, short \*pFangWei)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x31; //功能码

szMsgSendData[3] = 0x09; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cHengGunStatus;

szMsgSendData[5] = sHengGun>>8;

szMsgSendData[6] = sHengGun;

szMsgSendData[7] = cFuYangStatus;

szMsgSendData[8] = sFuYang>>8;

szMsgSendData[9] = sFuYang;

szMsgSendData[10] = cFangWeiSgtatus;

szMsgSendData[11] = sFangWei>>8;

szMsgSendData[12] = sFangWei;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pHengGun = 0x00;

\* pFuYang = 0x00;

\* pFangWei = 0x00;

#else

\* pHengGun = 0x00;

\* pFuYang = 0x00;

\* pFangWei = 0x00;

#endif

return 0;

}

int LuoJingShuJu(char cData, char \*pLuoJingType, short \* pData)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x32; //功能码

szMsgSendData[3] = 0x01; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cData;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pLuoJingType = 0x00;

\* pData = 0x00;

#else

\* pLuoJingType = 0x00;

\* pData = 0x00;

#endif

return 0;

}

int LuoJingShuJuLuoPan(char cData, char \*pLuoJingType, short \* pData)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x32; //功能码

szMsgSendData[3] = 0x01; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cData;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pLuoJingType = 0x00;

\* pData = 0x00;

#else

\* pLuoJingType = 0x00;

\* pData = 0x00;

#endif

return 0;

}

int XinBiaoJieShouJi(char cData, char \*pXinBiaoJieShouJiStatus, char \*pXinBiaoShiBieStatus, short \* pXinHaoQiangDu)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x32; //功能码

szMsgSendData[3] = 0x01; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cData;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pXinBiaoJieShouJiStatus = 0x00;

\* pXinBiaoShiBieStatus = 0x00;

\* pXinHaoQiangDu = 0x00;

#else

\* pXinBiaoJieShouJiStatus = 0x00;

\* pXinBiaoShiBieStatus = 0x00;

\* pXinHaoQiangDu = 0x00;

#endif

return 0;

}

int DVBJieShouJi(char cData,char \*pDVBJieShouJiStatus, char \*pXinBiaoShiBieStatus, short \* pXinHaoQiangDu)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x32; //功能码

szMsgSendData[3] = 0x01; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cData;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pDVBJieShouJiStatus = 0x00;

\* pXinBiaoShiBieStatus = 0x00;

\* pXinHaoQiangDu = 0x00;

#else

\* pDVBJieShouJiStatus = 0x00;

\* pXinBiaoShiBieStatus = 0x00;

\* pXinHaoQiangDu = 0x00;

#endif

return 0;

}

int GongFangKongZhi(char cSubCode, char \* pFaSheYunXu)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x39; //功能码

szMsgSendData[3] = 0x03; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cSubCode;

szMsgSendData[5] = -1; //-1：无效

szMsgSendData[6] = \* pFaSheYunXu;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pFaSheYunXu = szMsgReadData[6];

#else

#endif

return 0;

}

int LNBKongZhi(char cSubCode, float \* pPinLv13v0k, float \* pPinLv13v22k, float \* pPinLv18v0k, float \* pPinLv18v22k, char \* cDianYa)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

int i = 0;

szMsgSendData[2] = 0x3A; //功能码

szMsgSendData[3] = 0x0a; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cSubCode;

short2char((\* pPinLv13v0k) \* 100, szMsgSendData + 5);

short2char((\* pPinLv13v22k) \* 100, szMsgSendData + 7);

short2char((\* pPinLv18v0k) \* 100, szMsgSendData + 9);

short2char((\* pPinLv18v22k) \* 100, szMsgSendData + 11);

szMsgSendData[13] = \* cDianYa;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

#if 0

TraceLog("read:");

for(i=0; i<64; i++)

{

TraceLog("[%x]", szMsgReadData[i]);

}

TraceLog("\n");

#endif

\* pPinLv13v0k = char2ushort(szMsgReadData+5)/100.0;

\* pPinLv13v22k = char2ushort(szMsgReadData+7)/100.0;

\* pPinLv18v0k = char2ushort(szMsgReadData+9)/100.0;

\* pPinLv18v22k = char2ushort(szMsgReadData+11)/100.0;

\* cDianYa = szMsgReadData[13];

#else

#endif

return 0;

}

//ok

int WenDuChuanGanQi(char \*pStatus, float \*pWenDu)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

short sTemp = 0;

int nRet = 0;

szMsgSendData[2] = 0x40; //功能码

szMsgSendData[3] = 0x00; // 长度，不包括lcr，也不包括自己

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pStatus = szMsgReadData[4];

sTemp = char2short(szMsgReadData+5);

\* pWenDu = (float)sTemp / 100.0;

#else

#endif

return 0;

}

int PowerMonitor(char \* p5vStatus, short \*p5vValue, char \* p24vStatus, short \*p24vValue)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x41; //功能码

szMsgSendData[3] = 0x00; // 长度，不包括lcr，也不包括自己

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* p5vStatus = 0x00;

\* p5vValue = 0x00;

\* p24vStatus = 0x00;

\* p24vValue = 0x00;

#else

\* p5vStatus = 0x00;

\* p5vValue = 0x00;

\* p24vStatus = 0x00;

\* p24vValue = 0x00;

#endif

return 0;

}

int ConfigData(char cData, short sDataAddr, short sData,

short \* pDataAddr, short \* pData)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x42; //功能码

szMsgSendData[3] = 0x05; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cData;

szMsgSendData[5] = sDataAddr>>8;

szMsgSendData[6] = sDataAddr;

szMsgSendData[7] = sData>>8;

szMsgSendData[8] = sData;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pData = 0x00;

\* pDataAddr = 0x00;

\* pData = 0x00;

#else

\* pData = 0x00;

\* pDataAddr = 0x00;

\* pData = 0x00;

#endif

return 0;

}

int GetVersion(char cData, short \* pVersion)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x43; //功能码

szMsgSendData[3] = 0x03; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cData;

szMsgSendData[5] = 0;

szMsgSendData[6] = 0;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pVersion = char2short(szMsgReadData+4);

#else

#endif

return 0;

}

int FangWeiShouCang(char cData, float \* pData)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

int i =0;

szMsgSendData[2] = 0x45; //功能码

szMsgSendData[3] = 0x03; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cData;

short2char(\*pData\*10, szMsgSendData+5);

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

#if 0

TraceLog("write:");

for(i=0; i<64; i++)

{

TraceLog("[%x]", szMsgSendData[i]);

}

TraceLog("\n");

#endif

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

#if 0

TraceLog("read:");

for(i=0; i<64; i++)

{

TraceLog("[%x]", szMsgReadData[i]);

}

TraceLog("\n");

#endif

\* pData = char2short(szMsgReadData+5)/10.0;

#else

\* pData = 2.2;

#endif

return 0;

}

int FuYangShouCang(char cData, float \* pData)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

int i =0;

szMsgSendData[2] = 0x46; //功能码

szMsgSendData[3] = 0x03; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cData;

short2char(\*pData\*10, szMsgSendData+5);

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

#if 0

TraceLog("write:");

for(i=0; i<64; i++)

{

TraceLog("[%x]", szMsgSendData[i]);

}

TraceLog("\n");

#endif

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

#if 0

TraceLog("read:");

for(i=0; i<64; i++)

{

TraceLog("[%x]", szMsgReadData[i]);

}

TraceLog("\n");

#endif

\* pData = char2short(szMsgReadData+5)/10.0;

#else

\* pData = 3.3;

#endif

return 0;

}

//ok

int MuBiaoWeiXing(char cFunSelect,

float \* pWeiXingJiingDu, char \* pJiHuaLeiXing, char \* pJiHuaFangShi,

char \* pJieShouJiLeiXing,float \* pXinBiaoPinLv, float \* pDvbPingLv,

int \* pDVBFuHaoLv, float \* pJiHuaPianZhiJiao, float \* pMenXian,

char \* cQingYong)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

short sTemp;

int nTemp;

int nRet = 0;

int i = 0;

szMsgSendData[2] = 0x50; //功能码

szMsgSendData[3] = 0x15; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cFunSelect;

short2char(\*pWeiXingJiingDu\*100, szMsgSendData+5);

szMsgSendData[7] = \* pJiHuaLeiXing;

szMsgSendData[8] = \* pJiHuaFangShi;

szMsgSendData[9] = \* pJieShouJiLeiXing;

float2char(\*pXinBiaoPinLv, szMsgSendData + 10);

float2char(\*pDvbPingLv, szMsgSendData + 14);

uint2char(\*pDVBFuHaoLv, szMsgSendData + 18);

short2char(\*pJiHuaPianZhiJiao\*100, szMsgSendData+22);

if( (cFunSelect == 0x01) || (cFunSelect == 0x11) )

{

szMsgSendData[24] = (\* pMenXian) \* 10;

}

else

{

szMsgSendData[24] = \* cQingYong;

}

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

#if 0

TraceLog("write:");

for(i=0; i<64; i++)

{

TraceLog("[%x]", szMsgSendData[i]);

}

TraceLog("\n");

#endif

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

#if 0

TraceLog("read:");

for(i=0; i<64; i++)

{

TraceLog("[%x]", szMsgReadData[i]);

}

TraceLog("\n");

#endif

sTemp = char2short(szMsgReadData + 5);

\* pWeiXingJiingDu = sTemp / 100.0;

\* pJiHuaLeiXing = szMsgReadData[7];

\* pJiHuaFangShi = szMsgReadData[8];

\* pJieShouJiLeiXing = szMsgReadData[9];

\* pXinBiaoPinLv = char2float(szMsgReadData + 10);

\* pDvbPingLv = char2float(szMsgReadData + 14);

\* pDVBFuHaoLv = char2uint(szMsgReadData + 18);

sTemp = char2ushort(szMsgReadData + 22);

// TraceLog("szMsgReadData = [%x][%x]\n", szMsgReadData[22], szMsgReadData[23]);

\* pJiHuaPianZhiJiao = sTemp / 100.0;

if( (cFunSelect == 0x01) || (cFunSelect == 0x11) )

{

\* pMenXian = (float)(szMsgReadData[24]/10.0);

}

else

{

\* cQingYong = szMsgReadData[24];

}

#else

\* pWeiXingJiingDu = 110.5;

#endif

return 0;

}

//ok

int LiLunDuiXingShuJu(float \* pFangXiang, float \* pFuYang, float \*pJiHua)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

short sTemp = 0;

int nRet = 0;

szMsgSendData[2] = 0x51; //功能码

szMsgSendData[3] = 0x00; // 长度，不包括lcr，也不包括自己

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

sTemp = char2short(szMsgReadData + 4);

\* pFangXiang = (float)sTemp / 100.0;

sTemp = char2short(szMsgReadData + 6);

\* pFuYang = (float)sTemp / 100.0;

sTemp = char2short(szMsgReadData + 8);

\* pJiHua = (float)sTemp / 100.0;

#else

\* pFangXiang = 1.1;

\* pFuYang = 2.2;

\* pJiHua = 3.3;

#endif

return 0;

}

int MenXianPeiZhi(char cData, char \* pData)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x46; //功能码

szMsgSendData[3] = 0x01; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cData;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pData = 0x00;

#else

\* pData = 0x00;

#endif

return 0;

}

int JieShouJiXinHaoYouXianJi(char cSubFunCode)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x53; //功能码

szMsgSendData[3] = 0x01; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cSubFunCode;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

// \* pData = 0x00;

#else

// \* pData = 0x00;

#endif

return 0;

}

int ZheDangPeiZhi(char cSubCode, float \* pFangWeiQiShi,

float \*pFangWeiJieShu, float \* pFuYangQiShi, float \* pFuYangJieShu)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

int i = 0;

szMsgSendData[2] = 0x54; //功能码

szMsgSendData[3] = 10; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cSubCode;

short2char(\*pFangWeiQiShi\*100, szMsgSendData+5);

short2char(\*pFangWeiJieShu\*100, szMsgSendData+7);

short2char(\*pFuYangQiShi\*100, szMsgSendData+9);

short2char(\*pFuYangJieShu\*100, szMsgSendData+11);

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

#if 0

TraceLog("write:");

for(i=0; i<64; i++)

{

TraceLog("[%x]", szMsgSendData[i]);

}

TraceLog("\n");

#endif

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

#if 0

TraceLog("read:");

for(i=0; i<64; i++)

{

TraceLog("[%x]", szMsgReadData[i]);

}

TraceLog("\n");

#endif

\* pFangWeiQiShi = char2short(szMsgReadData+5)/100.0;

\* pFangWeiJieShu = char2short(szMsgReadData+7)/100.0;

\* pFuYangQiShi = char2short(szMsgReadData+9)/100.0;

\* pFuYangJieShu = char2short(szMsgReadData+11)/100.0;

#else

#endif

return 0;

}

//ok

int ShouDongKongZhiSanZhou(char cSubFunCode, char cStep)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x5A; //功能码

szMsgSendData[3] = 0x03; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cSubFunCode; //子功能码

szMsgSendData[5] = cStep; //增加 减少一步

szMsgSendData[6] = 0X01; //cnt 定死为1

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

#else

#endif

return 0;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int update\_tel(char pData)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

short sTemp = 0;

int nRet = 0;

szMsgSendData[2] = 0x89; //功能码

szMsgSendData[3] = 0x01; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = pData;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

#else

#endif

return 0;

}

//ok

int ShouDongKongZhiBuChangSuDu(char cType1, char cType2, float \* pData)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

short sTemp = 0;

int nRet = 0;

szMsgSendData[2] = 0x59; //功能码

szMsgSendData[3] = 0x03; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = (cType1<<4)+cType2; //子功能码

short2char((short)(\* pData) \* 100, szMsgSendData + 5);

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

sTemp = char2short(szMsgReadData+5);

\* pData = (float)(sTemp / 100.0);

#else

#endif

return 0;

}

//得到罗盘角度

int get\_compass\_angle(int angle, char flag)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

short sTemp = 0;

int nRet = 0;

szMsgSendData[2] = 0x36; //功能码

szMsgSendData[3] = 0x03; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = flag; //子功能码

short2char((short)angle, szMsgSendData + 5);

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

sTemp = char2short(szMsgReadData+5);

return sTemp;

#else

return 0;

#endif

}

int SouSuoFanWei(char cSubFunCode, float \* pFwss\_fw, float \* pFyss\_fw, float \* pFybj\_jd,

float \* pFwss\_sd, float \* pFyss\_sd)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

int i =0;

szMsgSendData[2] = 0x55; //功能码

szMsgSendData[3] = 11; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cSubFunCode; //子功能码

short2char(\*pFwss\_fw\*100, szMsgSendData+5);

short2char(\*pFyss\_fw\*100, szMsgSendData+7);

short2char(\*pFybj\_jd\*100, szMsgSendData+9);

short2char(\*pFwss\_sd\*100, szMsgSendData+11);

short2char(\*pFyss\_sd\*100, szMsgSendData+13);

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

#if 0

TraceLog("write:");

for(i=0; i<64; i++)

{

TraceLog("[%x]", szMsgSendData[i]);

}

TraceLog("\n");

#endif

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

#if 0

TraceLog("read:");

for(i=0; i<64; i++)

{

TraceLog("[%x]", szMsgReadData[i]);

}

TraceLog("\n");

#endif

\* pFwss\_fw = char2short(szMsgReadData+5) / 100.0;

\* pFyss\_fw = char2short(szMsgReadData+7) / 100.0;

\* pFybj\_jd = char2short(szMsgReadData+9) / 100.0;

\* pFwss\_sd = char2short(szMsgReadData+11) / 100.0;

\* pFyss\_sd = char2short(szMsgReadData+13) / 100.0;

#else

\* pFwss\_fw = 1;

\* pFyss\_fw = 2;

\* pFybj\_jd = 3;

\* pFwss\_sd = 4;

\* pFyss\_sd = 5;

#endif

return 0;

}

int ant\_serch\_angle(char cSubFunCode,float \*az\_search\_left, float \*az\_search\_right, float \*el\_search\_down, float \*el\_search\_up)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

int i =0;

szMsgSendData[2] = 0x55; //功能码

szMsgSendData[3] = 0x0e; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cSubFunCode; //子功能码

short2char(\*az\_search\_left\*100, szMsgSendData+5);

short2char(\*az\_search\_right\*100, szMsgSendData+7);

short2char(\*el\_search\_down\*100, szMsgSendData+9);

short2char(\*el\_search\_up\*100, szMsgSendData+11);

szMsgSendData[13] = 0;

szMsgSendData[14] = 0;

szMsgSendData[15] = 0;

szMsgSendData[16] = 0;

szMsgSendData[17] = 0;

szMsgSendData[18] = 0;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

#if 0

TraceLog("write:");

for(i=0; i<64; i++)

{

TraceLog("[%x]", szMsgSendData[i]);

}

TraceLog("\n");

#endif

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

#if 0

TraceLog("read:");

for(i=0; i<64; i++)

{

TraceLog("[%x]", szMsgReadData[i]);

}

TraceLog("\n");

#endif

\*az\_search\_left = char2short(&szMsgReadData[5]) / 100.0;

\*az\_search\_right = char2short(&szMsgReadData[7]) / 100.0;

\*el\_search\_down = char2short(&szMsgReadData[9]) / 100.0;

\*el\_search\_up = char2short(&szMsgReadData[11]) / 100.0;

#else

\*az\_search\_left = 1;

\*az\_search\_right = 2;

\*el\_search\_down = 3;

\*el\_search\_up = 4;

#endif

return 0;

}

int ZaoShengDianPing(char \* pZaoShengDianPingZhuangTai, short \* pZaoShengDianPingShuJu,

char \* pGenZongDianPingZhuangTai, short \* pGenZongDianPingShuJu)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadDat[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x56; //功能码

szMsgSendData[3] = 0x00; // 长度，不包括lcr，也不包括自己

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadDat, sizeof(szMsgReadDat));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

#else

#endif

return 0;

}

int TianXianMianSaoMian(char cSubFun,

float \* pFangWeiJIaoDu, float \* pFuYangJIaoDu,

float \* pMenXian, short \* pTime)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x58; //功能码

szMsgSendData[3] = 0x09; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cSubFun;

short2char(\*pFangWeiJIaoDu\*100, szMsgSendData+5);

short2char(\*pFuYangJIaoDu\*100, szMsgSendData+7);

short2char(\*pMenXian\*100, szMsgSendData+9);

short2char(\*pTime\*10, szMsgSendData+11);

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pFangWeiJIaoDu = char2short(szMsgReadData+5) / 100.0;

\* pFuYangJIaoDu = char2short(szMsgReadData+7) / 100.0;

\* pMenXian = char2short(szMsgReadData+9) / 100.0;

\* pTime = char2short(szMsgReadData+11)/10.0;

#else

#endif

return 0;

}

int GpsShuChuPeiZhi(char cOutput, char cXieYi, char cTime)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x15; //功能码

szMsgSendData[3] = 0x03; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cOutput; //子功能码

szMsgSendData[5] = cXieYi;

szMsgSendData[6] = cTime;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

#else

#endif

return 0;

}

int tel\_acu\_47(char cOutput, char flag)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x47; //功能码

szMsgSendData[3] = 0x02; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cOutput; //子功能码

szMsgSendData[5] = flag;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

#else

#endif

return 0;

}

int ModemPeiZhi(char cType, char cPort, char cProtol, char cTime)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x16; //功能码

szMsgSendData[3] = 0x04; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cType;

szMsgSendData[5] = cPort;

szMsgSendData[6] = cProtol;

szMsgSendData[7] = cTime;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

#else

#endif

return 0;

}

int PoShuPeiZhi(char cAutoColor, char cAutoRf)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x17; //功能码

szMsgSendData[3] = 0x02; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cAutoColor;

szMsgSendData[5] = cAutoRf;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

#else

#endif

return 0;

}

int LuoJingPeiZhi(char cType, char \* pName)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x18; //功能码

szMsgSendData[3] = 0x02; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cType;

szMsgSendData[5] = \*pName;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pName = szMsgReadData[5];

#else

#endif

return 0;

}

int BoShuYanSe(char cSubCode, char \* pYanSe)

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

szMsgSendData[2] = 0x19; //功能码

szMsgSendData[3] = 0x02; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = cSubCode;

szMsgSendData[5] = \* pYanSe;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

\* pYanSe = szMsgReadData[5];

#else

#endif

return 0;

}

int ShengJi()

{

#ifndef TEST

char szMsgSendData[64] = {0};

char szMsgReadData[64] = {0};

int nRet = 0;

int i =0;

szMsgSendData[2] = 0x1A; //功能码

szMsgSendData[3] = 0x01; // 长度，不包括lcr，也不包括自己

szMsgSendData[4] = 1;

nRet = DelMeg(szMsgSendData);

if (nRet != 0)

{

TraceLog("DelMeg error\n");

return -1;

}

nRet = MsgQueSendAndRead(szMsgSendData, szMsgSendData[3] + 7,

szMsgReadData, sizeof(szMsgReadData));

if (nRet != 0)

{

TraceLog("MsgQueSendAndRead error\n");

return -1;

}

#else

#endif

return 0;

}