**電工實驗（四）**

**微處理器實驗六**

**RS232**

**第十四組**

**許銘森B063011058**

主程式

**UART標頭檔**

#ifndef \_\_UARTAGENT\_H\_\_

#define \_\_UARTAGENT\_H\_\_

#include "M451Series.h"

extern uint32\_t timecount; //用來計時的

typedef struct{ //宣告成CQ\_t struct

uint8\_t data[64]; //包含64大小的buf

uint8\_t head; //包含head tail

uint8\_t tail;

}CQ\_t;

void UART0\_Initial(void);

void UART0\_TxTask(void);

void UART0\_IRQHandler(void);

uint8\_t UART0\_IsRxDataReady(void);

uint8\_t UART0\_IsTxBufFull(void);

void UART0\_WriteByte(uint8\_t Data);

uint8\_t UART0\_ReadByte(void);

static void CQ\_EnQ(CQ\_t \*ptrBuf, uint8\_t Data);

static uint8\_t CQ\_DeQ(CQ\_t \*ptrBuf);

static uint8\_t CQ\_IsFull(CQ\_t \*ptrBuf);

static uint8\_t CQ\_IsEmpty(CQ\_t \*ptrBuf);

static void CQ\_Clear(CQ\_t \*ptrBuf);

#endif /\* \_\_UARTAGENT\_H\_\_ \*/

**UART 主程式**

#include "UARTAgent.h"

static CQ\_t TxBuf; //宣告txbuffer

static CQ\_t RxBuf; //宣告rxbuffer

void UART0\_Initial(void){

CLK->APBCLK0 |= CLK\_APBCLK0\_UART0CKEN\_Msk; //UART0 Clock Enable

CLK->CLKSEL1 &= ~CLK\_CLKSEL1\_UARTSEL\_Msk; //UART0 Clock Source Selection

CLK->CLKSEL1 |= 0x3 << CLK\_CLKSEL1\_UARTSEL\_Pos; //22.1184MHz

CLK->CLKDIV0 &= ~CLK\_CLKDIV0\_UARTDIV\_Msk; //UART 除頻

CLK->CLKDIV0 |= 0 << CLK\_CLKDIV0\_UARTDIV\_Pos; //22.1184MHz / (0 + 1) = 22.1184MHz

UART0->BAUD &= ~UART\_BAUD\_BAUDM0\_Msk; //Set Baud Rate Divide Mode

UART0->BAUD &= ~UART\_BAUD\_BAUDM1\_Msk; //0 = UART\_CLK / [16 \* (BRD+2)]

UART0->BAUD &= ~UART\_BAUD\_BRD\_Msk;

UART0->BAUD |= 34 << UART\_BAUD\_BRD\_Pos; // 22118400 / (16 \* (34 + 2)) = 38400

UART0->FUNCSEL &= ~UART\_FUNCSEL\_FUNCSEL\_Msk; //UART Function Selection

UART0->FUNCSEL |= 0 << UART\_FUNCSEL\_FUNCSEL\_Pos; //00 = UART function.

UART0->LINE &= ~UART\_LINE\_WLS\_Msk; //Word Length Selection

UART0->LINE |= 0x3 << UART\_LINE\_WLS\_Pos; //11 = 8 bits.

UART0->LINE &= ~UART\_LINE\_NSB\_Msk; //Number Of "STOP Bit"

UART0->LINE |= 0x0 << UART\_LINE\_NSB\_Pos; //0 = One "STOP bit"

UART0->LINE &= ~UART\_LINE\_PBE\_Msk; //Parity Bit Enable Bit

UART0->LINE |= 0x0 << UART\_LINE\_PBE\_Pos; //0 = No parity bit generated.

UART0->INTEN = 0;

UART0->INTEN |= UART\_INTEN\_RDAIEN\_Msk;//Receive Data Available Interrupt Enable Bit

NVIC->ISER[1] |= 0x1 << (36 - 32); //enable NVIC UART0 interrupt(36)

/\* Set pin function as Tx Rx pin \*/

SYS->GPD\_MFPL &= ~(SYS\_GPD\_MFPL\_PD6MFP\_Msk | SYS\_GPD\_MFPL\_PD1MFP\_Msk);

SYS->GPD\_MFPL |= (SYS\_GPD\_MFPL\_PD6MFP\_UART0\_RXD | SYS\_GPD\_MFPL\_PD1MFP\_UART0\_TXD);

CQ\_Clear(&TxBuf);

CQ\_Clear(&RxBuf);

}

void UART0\_TxTask(void){

/\* If transmitter holding register(THR) is empty and Tx buffer is not empty \*/

if((UART0->INTSTS & UART\_INTSTS\_THREIF\_Msk) && !CQ\_IsEmpty(&TxBuf)){

UART0->DAT = CQ\_DeQ(&TxBuf);

}

}

void UART0\_IRQHandler(void){

uint8\_t c;

if(UART0->INTSTS & UART\_INTSTS\_RDAIF\_Msk){

c = UART0->DAT;

if(!CQ\_IsFull(&RxBuf)){

CQ\_EnQ(&RxBuf, c);

}

}

}

uint8\_t UART0\_IsRxDataReady(void){ //確認準備傳資料

return !CQ\_IsEmpty(&RxBuf); //RXbuf不為空

}

uint8\_t UART0\_IsTxBufFull(void){ //確認Txbuf不是滿

return CQ\_IsFull(&TxBuf);

}

void UART0\_WriteByte(uint8\_t Data){ //寫入資料

CQ\_EnQ(&TxBuf, Data);

}

uint8\_t UART0\_ReadByte(void){ //讀取資料

return CQ\_DeQ(&RxBuf);

}

static void CQ\_EnQ(CQ\_t \*ptrBuf, uint8\_t Data){ //寫入資料到CQ

ptrBuf->data[ptrBuf->head++] = Data;

ptrBuf->head &= 0x3F;

}

static uint8\_t CQ\_DeQ(CQ\_t \*ptrBuf){ //讀取資料到CQ

uint8\_t ret;

ret = ptrBuf->data[ptrBuf->tail++];

ptrBuf->tail &= 0x3F;

return ret;

}

static uint8\_t CQ\_IsFull(CQ\_t \*ptrBuf){

return (((ptrBuf->head + 1) & 0x3F) == ptrBuf->tail);

}

static uint8\_t CQ\_IsEmpty(CQ\_t \*ptrBuf){

return (ptrBuf->head == ptrBuf->tail);

}

static void CQ\_Clear(CQ\_t \*ptrBuf){

ptrBuf->head = 0;

ptrBuf->tail = 0;

}

**Main**

#include <stdio.h> //導入各標頭檔

#include "M451Series.h"

#include "TimebaseAgent.h"

#include "SevenSegmentAgent.h"

#include "ButtonAgent.h"

#include "StepMotorAgent.h"

#include "ADCAgent.h"

#include "UARTAgent.h"

char const strHello[] = "\nHello!\n";

char sendbuf[64];

uint32\_t timecount;

uint8\_t \_7seg\_lo;

uint8\_t \_7seg\_hi;

uint8\_t mtr\_speed; //rpm

uint8\_t mtr\_oldspeed;

uint8\_t mtr\_dir;

uint32\_t mtr\_d;

int8\_t clk\_sec;

int8\_t clk\_min;

int8\_t clk\_hr;

void DisplayTask(uint8\_t hi, uint8\_t lo);

void ClockTick(void);

uint8\_t StrPush(char \*str);

uint32\_t StepMtr\_RPMtoD(uint8\_t rmp);

int \_\_main(){

char c;

Timebase\_Initial(); //呼叫各初始化涵式

\_7Seg\_Initial();

Btn\_Initial();

StepMtr\_Initial();

ADC\_Initial();

UART0\_Initial();

clk\_sec = 10;

clk\_min = 16;

clk\_hr = 20;

\_7seg\_hi = 0;

\_7seg\_lo = 0;

mtr\_dir = 1;

mtr\_speed = 253;

mtr\_d = StepMtr\_RPMtoD(mtr\_speed);

StrPush((char\*)strHello);

while(1){//前面實驗有關

if(Btn\_IsOneShot(0x01) == 0x01){

StrPush("This is BTN1\n");

Btn\_OneShotClear(0x01);

}

if(Btn\_IsOneShot(0x02) == 0x02){

Btn\_OneShotClear(0x02);

StrPush("This is BTN2\n");

}

if(Btn\_IsOneShot(0x04) == 0x04){

Btn\_OneShotClear(0x04);

StrPush("This is BTN3\n");

}

if(Btn\_IsOneShot(0x08) == 0x08){

Btn\_OneShotClear(0x08);

StrPush("This is BTN4\n");

}

if(UART0\_IsRxDataReady()){

c = UART0\_ReadByte();

switch(c){//電腦控制

case '+':

if(mtr\_speed == 255){

StrPush("Max speed\n");

}else{

mtr\_speed++;

mtr\_d = StepMtr\_RPMtoD(mtr\_speed);

StrPush("Speed Up\n");

}

break;

case '-':

if(mtr\_speed == 0){

StrPush("Min speed\n");

}else{

mtr\_speed--;

mtr\_d = StepMtr\_RPMtoD(mtr\_speed);

StrPush("Speed Down\n");

}

break;

case 's':

mtr\_speed = 0;

mtr\_d = StepMtr\_RPMtoD(mtr\_speed);

StrPush("Stop\n");

break;

case 'r':

mtr\_dir ^= 1;

StrPush("Reverse\n");

break;

case 'p':

sprintf(sendbuf, "Speed: %d rpm\nDirection: %s\n",

mtr\_speed, mtr\_dir ? "Clockwise":"Counterclockwise");

StrPush(sendbuf);

case 't':

sprintf(sendbuf, "Time: [%02d:%02d:%02d]\n", clk\_hr, clk\_min, clk\_sec);

StrPush(sendbuf);

break;

default:

StrPush("Unknown\n");

}

}

\_7seg\_hi = (mtr\_speed/10)%10;

\_7seg\_lo = mtr\_speed%10;

ClockTick();

UART0\_TxTask(); //傳送資料

DisplayTask(\_7seg\_hi, \_7seg\_lo); //10Hz update rate

\_7Seg\_Task(); //1kKz switch rate

ADC\_Task(); //ADC conversion

StepMtr\_Task(mtr\_dir, mtr\_d); //Step motor output

Btn\_Task(); //Button scan

}

}

void DisplayTask(uint8\_t hi, uint8\_t lo){

static uint32\_t DpOldCount = 0;

if((uint32\_t)(timecount - DpOldCount) < 1000)//100ms

return;

DpOldCount = timecount;

\_7Seg\_WriteBuf(hi, lo);

}

void ClockTick(void){

static uint32\_t ClockOldCount = 0;

if((uint32\_t)(timecount - ClockOldCount) < 10000)

return;

ClockOldCount = timecount;

UART0\_WriteByte(7);

if(++clk\_sec >= 60){

clk\_sec = 0;

if(++clk\_min >= 60){

clk\_min = 0;

if(++clk\_hr >= 24){

clk\_hr = 0;

}

}

}

}

uint8\_t StrPush(char \*str){

uint8\_t i = 0;

while(str[i] && !UART0\_IsTxBufFull()){

UART0\_WriteByte(str[i++]);

}

return i;

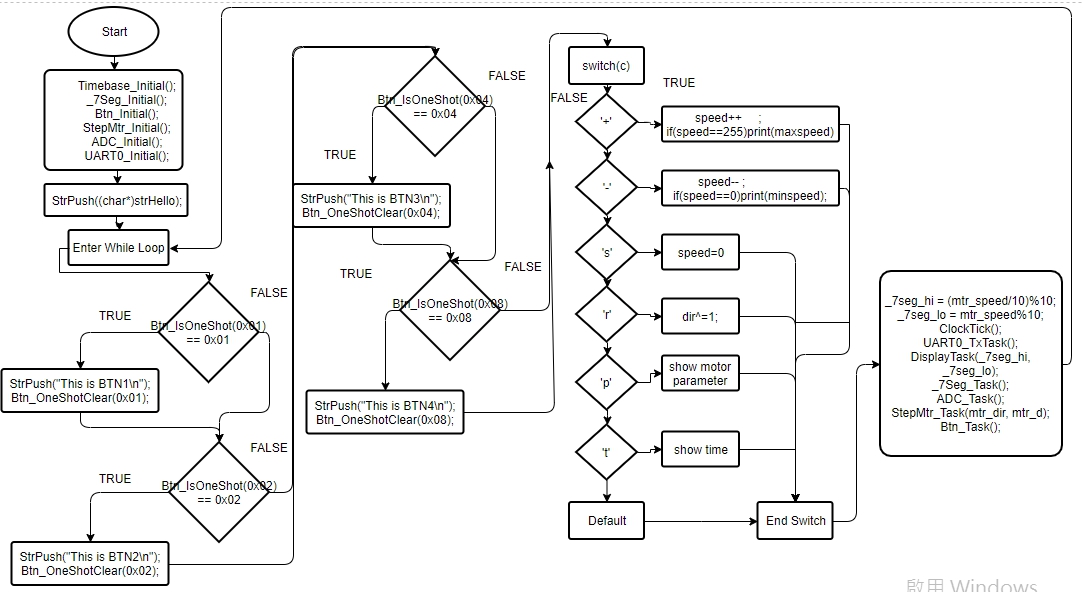
}

uint32\_t StepMtr\_RPMtoD(uint8\_t rmp)

{

return rmp ? 6000/rmp : 0;

}

流程圖

心得

這次實驗是第六次使用微控器，RS232為微控器和電腦連綫很重要的協定，Circular Queue buffer 是RS232的重點。微控器要和電腦傳資料才有意義，所以這次實驗可以結合前五次的内容，用電腦做控制。