# 嵌入式手扎八 ADC/DAC（传感器）

说明：因为这次ADC和DAC模块做的比较晚，所以这次的实验报告就结合了本次期末考试做的实验第41题来进行书写；（因为有一些普通的传感器主要用到了AD模块接收模拟量的通道输入进行处理，和温度传感器原理类似。所以这次手札主要围绕如何再程序里写时序来接收传感器发送过来的数据的这个问题）

## 实验目的：

1、通过ADC模块进行传感器温度读取显示

2、通过DAC模块进行三角波的C#端显示

3、通过编写温湿度传感器DHT11和MCU之间的数据传输时的时序程序来控制温度数据的输入到MCU

## 实验环境：

1）PC机一台，KL25开发板一片，导线若干

2) KDS开发环境，VS2017或更高平台

## 实验内容：

## 实验一： 1、通过ADC模块进行传感器温度读取显示

### 1）设计AD模块

#### #1目的：通过设计AD模块来将MCU的AD模块的接收温度模拟量的通道值发送给C#,让C#在接收到了模拟量后进行数字量转换

#2问题：

##### 问题1）如何正确将温度模拟量发送给C#

@方法：通过AD模块中的函数 adc\_read（通道号）来采集每个AD模块的输入通道的值，存放到数组ADCResult[]里面;读取温度模拟量对应的通道的值存放到数组的ADCResult[15]里面。

@源码：

//A组初始化（通道组、单端输入，采样精度，硬件均值）

adc\_init(MUXSEL\_A, 0, 16, SAMPLE32);

//加头标志

ADCResult[0] = 0x1122;

//采集数据

ADCResult[1] = adc\_read(0);

**for** (i = 2; i <= 8; i++)

ADCResult[i] = adc\_read(i + 1);

**for** (i = 9; i <= 13; i++)

ADCResult[i] = adc\_read(i + 2);

ADCResult[14] = adc\_read(23);

ADCResult[15] = adc\_read(26); //芯片温度采集通道

//B组初始化（通道组、单端输入，采样精度，硬件均值）

adc\_init(MUXSEL\_B, 0, 16, SAMPLE32);

//采样

**for** (i = 16; i <= 20; i++)

ADCResult[i] = adc\_read(i - 12);

//加末尾标志

ADCResult[20] = 0x8899;

//将采集的A/D值通过串口发送到PC

uart\_sendN(UART\_TEST, 42, (uint\_8\*) ADCResult);

Delay\_ms(50);

### 2）设计C#模块

#### #目的：串口接收MCU传递过来的模拟量进行处理转成温度的数字量，进行字符转换输出到对应的text控件里面

#### #问题：

##### 1）如何接收对应的温度通道的模拟量并处理成数字量；

@方法：

因为在MCU发送每个通道的模拟量组成的模拟量数组时，对应通道的模拟量存放在对应位置的数组元素里，位置已经一一确定，所以C#只要按照正确的数组元素的顺序来接受传递的模拟量，则可以按照温度模拟量存放的位置来正确的接收模拟量；在进行处理即可；

2）

@源码：

private void SCIPort\_DataReceived(object sender, System.IO.Ports.SerialDataReceivedEventArgs e)

{

byte len = 0; //标记接收的数据的长度

Int32 i;

Thread.Sleep(50);

sci.SCIReceiveData(SCIPort, ref PublicVar.g\_ReceiveByteArray, ref len);

if ((len == 42) && (PublicVar.g\_ReceiveByteArray[0] == 0x22) && (PublicVar.g\_ReceiveByteArray[41] == 0x88))

{

//将串口的数据显示到文本

//通道A

i = PublicVar.g\_ReceiveByteArray[3] \* 256 + PublicVar.g\_ReceiveByteArray[2];

SCIUpdateRevtxtbox(txtchannel0, Convert.ToString(i, 16));

。.

.

.

.

i = PublicVar.g\_ReceiveByteArray[17] \* 256 + PublicVar.g\_ReceiveByteArray[16];

SCIUpdateRevtxtbox(txtchannel9, Convert.ToString(i, 16));

//接受电压值

double temp = Convert.ToInt32(i \* 3300) / 65535.0;//按照比例转换成数字量

.

.

.

.

i = PublicVar.g\_ReceiveByteArray[39] \* 256 + PublicVar.g\_ReceiveByteArray[38];

SCIUpdateRevtxtbox(txtchannel7b, Convert.ToString(i, 16));

//温度（26通道）数据处理

i = PublicVar.g\_ReceiveByteArray[31] \* 256 + PublicVar.g\_ReceiveByteArray[30];

double VTemp = Convert.ToInt32(i \* 3300) / 65535.0;

//将电压值转换为温度值

//719为25度时的标准电压采样值，1.715为温度传感器的电压/温度曲线（近似线性关系）的斜率

double t = 25 - (VTemp - 719) / 1.715;

//绘制坐标

double x = (double)new XDate(DateTime.Now);

double y = t;

list.Add(x, t);

String strT = t.ToString().Substring(0, 5);

SCIUpdateRevtxtbox\_2(textBoxTemperature, strT);//显示温度

Refresh(zedGraphControl1);

this.TSSLState.Text = "过程提示:数据接收!";

}

//接收数据失败

else

{

//sci.SCIReceInt(SCIPort, 1);//设置产生接收中断的字节数

this.TSSLState.Text = "过程提示:数据接收!";

}

//timer1.Enabled = true;

}

## 实验二： 2、通过DAC模块进行三角波的C#端显示

### 1）设计DA/AD模块

#### #目的：通过编写三角波的数字量变化代码，每次for循环将变化后的的数字量通过DA模块转化为模拟量（电压值从PTE\_30端口输出），选取和时AD模块的通道进行模拟量接受，存放到数组ADCResult[]内部通过MCU传递给C#

#### #问题：

##### 问题1）如何正确的将每次变化的数字量转化为模拟量

@方法：使用DA模块的dac\_convert（）函数进行转换，从唯一的PTE\_30输出，选取AD模块的通道9，进行采集对应的模拟量输出（在C#端体现）

@源码：

for(; ;){

dac\_convert(VReference);

x++;

//根据标志位，设置小灯慢慢点亮或慢慢熄灭

//三角波

**if** (light\_flag == 1)

VReference += 10;

**else** **if** (light\_flag == 0)

VReference -= 10;

//VReference限幅，并反转灯点亮或熄灭的标志位

**if** (VReference >= 2600)

light\_flag = 0;

**if** (VReference <= 2200)

light\_flag = 1;

//A组初始化（通道组、单端输入，采样精度，硬件均值）

adc\_init(MUXSEL\_A, 0, 16, SAMPLE32);

//加头标志

ADCResult[0] = 0x1122;

//采集数据

ADCResult[1] = adc\_read(0);

**for** (i = 2; i <= 8; i++)

ADCResult[i] = adc\_read(i + 1);

**for** (i = 9; i <= 13; i++)

ADCResult[i] = adc\_read(i + 2);

ADCResult[14] = adc\_read(23);

ADCResult[15] = adc\_read(26); //芯片温度采集通道

//B组初始化（通道组、单端输入，采样精度，硬件均值）

adc\_init(MUXSEL\_B, 0, 16, SAMPLE32);

//采样

**for** (i = 16; i <= 20; i++)

ADCResult[i] = adc\_read(i - 12);

//加末尾标志

ADCResult[20] = 0x8899;

//将采集的A/D值通过串口发送到PC

uart\_sendN(UART\_TEST, 42, (uint\_8\*) ADCResult);

Delay\_ms(50);

}

### 2)C#模块的设计

@方法：和上面的步骤类似，只是使用的时我们和PTE\_30相连的通道引脚的模拟量，直接用作画图函数list(x,y)的纵坐标的值；

@源码：

i = PublicVar.g\_ReceiveByteArray[17] \* 256 + PublicVar.g\_ReceiveByteArray[16];

SCIUpdateRevtxtbox(txtchannel9, Convert.ToString(i, 16));

//接受电压值

double temp = Convert.ToInt32(i \* 3300) / 65535.0;//按照比例转换成数字量

..

.

.

.

list.Add(x, t);//画图

## 实验三：通过编写温湿度传感器DHT11和MCU之间的数据传输时的时序程序来控制温度数据的输入到MCU

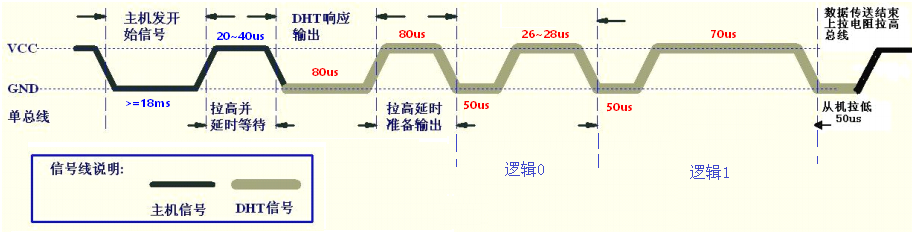
### 1）设计DHT11模块

#### #目的：初始化I/O引脚，并设计MCU和传感器传输数据的时序，已完成一个完整的数据传输周期

#### #问题

##### 问题1）：如何设计时序

@方法：通过上网查询DHT11的读取时序图，学习DHT和主机进行通信是的时序，从而在MCU端进行编程完成相关的引脚的输入输出功能转换，并控制相应引脚功能转换的时间；



@源码：

/\*

\* dht11.c

\*

\* Created on: 2019年12月25日

\* Author: 刘乾

\*/

//file:dht11.c

//author:lulipro

//date:2019-5-5

#include "dht11.h"

//包含引脚设置模块，延时模块

#include "gpio.h"

#include "delay.h"

#include "adc.h"

#include "common.h"

#include <string.h> //for memset()

static uint\_16 DHT11\_dataPin\_; //驱动DHT11的数据线引脚

static uint8\_t DHT11\_recvData\_[5]; //存放 从DHT11读取的数据的缓冲数组，40个bit

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

\*作用：初始化

\*参数：pin，Arduino驱动DHT11的总线使用的IO引脚

\*返回：无

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

void DHT11\_init(uint\_16 pin)

{

//初始IO引脚号赋给静态全局变量DHT11\_dataPin\_

DHT11\_dataPin\_ = pin;

}

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

\*作用：读取一次DHT11的传感器数据

\*参数：temperature，指向存放温度数据的指针；humidity，指向存放湿度数据的指针

\*返回：函数的执行状态：DHT11\_OK，DHT11\_TIMEOUT,DHT11\_CHECKEROR

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

int DHT11\_read(uint8\_t\* temperature,uint8\_t\* humidity )

{

uint8\_t time\_cnt;

uint8\_t i;

uint8\_t bit\_position;

memset(DHT11\_recvData\_,0,5); //缓冲数组内容清0

/\*--------------主机发送起始信号----------------\*/

gpio\_init(DHT11\_dataPin\_,1,0);

gpio\_set(DHT11\_dataPin\_, 0); // 主机将总线拉低（时间>=18ms），使得DHT11能够接收到起始信号。

delay\_us(18000); //至少 18 ms

gpio\_set(DHT11\_dataPin\_, 1); // 主机将总线拉高，代表起始信号结束。

delay\_us(35); //延时20~40us

/\*--------------引脚配置为输入模式，准备接收传感器回传的数据-------------------\*/

gpio\_init(DHT11\_dataPin\_,0,0); //配置为输入模式

//DHT11将总线拉低至少80us，作为DHT11的响应信号（ACK）。

time\_cnt=0;

while(0 == gpio\_get(DHT11\_dataPin\_))

{

delay\_us(5);

++time\_cnt;

if(time\_cnt > 16) return DHT11\_TIMEOUT;

}

//DHT11将总线拉高至少80us，为发送传感器数据做准备。

time\_cnt=0;

while(1 == gpio\_get(DHT11\_dataPin\_))

{

delay\_us(5);

++time\_cnt;

if(time\_cnt > 16) return DHT11\_TIMEOUT;

}

/\*-------------------DHT11数据帧的接收和解析------------------\*/

for( i=0 ; i < 40 ; ++i )

{

time\_cnt = 0;

while( 0 == gpio\_get(DHT11\_dataPin\_) ) //拉低50us作为bit信号的起始标志

{

time\_cnt++;

delay\_us(5);

if(time\_cnt>10) return DHT11\_TIMEOUT;

}

time\_cnt = 0;

while( 1 == gpio\_get(DHT11\_dataPin\_) ) //拉高。持续26~28us表示bit0，持续70us表示bit1

{

time\_cnt++;

delay\_us(5);

if(time\_cnt>14) return DHT11\_TIMEOUT;

}

if(time\_cnt>6){ //说明是bit1

bit\_position = 7 - i%8;

DHT11\_recvData\_[i/8] |= (uint8\_t)(1<<bit\_position);

}

}

//------------检验和的比对-------------

i = (uint8\_t)(DHT11\_recvData\_[0] + DHT11\_recvData\_[1] + DHT11\_recvData\_[2] + DHT11\_recvData\_[3]) ;

if(i != DHT11\_recvData\_[4] ) return DHT11\_CHECKERROR;

\*humidity = DHT11\_recvData\_[0]; //回传湿度数据

\*temperature = DHT11\_recvData\_[2]; //回传温度数据

return DHT11\_OK;

}

### 2）设计main函数

#### #目的：初始化相关功能模块，完成DHT11的循环调用

#### #问题：

##### 问题1）：如何通过保存每次读取的温度数据

@方法:设置全局变量 uint8\_t temperature; uint8\_t humidity; 当作参数，每次传入其指针给DHT11\_read(&temperature, &humidity);函数，每次读取后根据此函数的返回标志判断是都读取成功，若读取成功，则通过串口发送发送给C#端；

@源码：

//说明见工程文件夹下的Doc文件夹内Readme.txt文件

//=====================================================================

#include "includes.h" //包含总头文件

extern void Delay\_ms(uint16\_t u16ms);

//存放温湿度值的变量

uint8\_t temperature;

uint8\_t humidity;

uint\_8 Tem[5];

uint\_8 Hum[5];

int flag=0;

int main(void) {

//1. 声明主函数使用的变量

//2. 关总中断

DISABLE\_INTERRUPTS;

//3. 初始化外设模块

light\_init(LIGHT\_BLUE, LIGHT\_OFF); //蓝灯初始化

uart\_init(UART\_1, 115200); //使能串口1，波特率为9600

uart\_init(UART\_2, 9600); //使能串口2，波特率为9600

delay\_init(0);

DHT11\_init(DHT\_DATA); //初始化接口为PTB\_10

//4. 给有关变量赋初值

//5. 使能模块中断

uart\_enable\_re\_int(UART\_1); //使能串口1接收中断

uart\_enable\_re\_int(UART\_2); //使能串口2接收中断

//6. 开总中断

ENABLE\_INTERRUPTS;

//进入主循环

//主循环开始==================================================

while (1) {

flag=DHT11\_read(&temperature, &humidity);

switch (flag) {

case DHT11\_OK:

Tem[0] = temperature / 10 + '0';

Tem[1] = temperature % 10 + '0';

Hum[0] = humidity / 10 + '0';

Hum[1] = humidity % 10 + '0';

uart\_send\_string(UART\_1, "temperature ");

uart\_sendN(UART\_1, 2, Tem);

uart\_send\_string(UART\_1, "°C ");

uart\_send\_string(UART\_1, "humidity ");

uart\_sendN(UART\_1, 2, Hum);

uart\_send\_string(UART\_1, "%RF ");

break;

case DHT11\_TIMEOUT:

uart\_send\_string(UART\_1, "time out ");

break;

case DHT11\_CHECKERROR:

uart\_send\_string(UART\_1, "check error ");

break;

default:

uart\_send\_string(UART\_1, "unknown error ");

break;

}

Delay\_ms(1000); //每次读取后延时1s后下一次读取

} //主循环end\_for

//主循环结束==================================================

}

## 实验总结：

其实我认为这次抽到41题，和写40给我带来了极大的启示性意义！

假如我只抽到了前面几个简单的只用到ad通道的传感器，我估计我不是很有可能接触到自己写时序的这个小任务，也很难获得到宝贵的程序设计思想和MCU和传感器之间的简单的总线时序的编写经验。

一开始我认为这是一项很艰难的任务，但是通过这次我发现，问题不去尽心尽力解决之前，说难是没有太大意义的，把他学会了才是最重要的。

还有就是对待学习的心态，完成实验是一方面，更重要的是自己在完成实验当中获得了怎么样的成长。