实验报告

2051374 吴雨阳 2052225 张勤杭 2052134 刘治华

1IAR实验开发环境

1.1 课内实验内容描述

蓝牙 4.0(CC2540)模块实验部分，使用的软件开发环境为 IAR Embedded Wordbench for MCS-51。 本节将介绍如何使用该 IAR 环境搭建配套实验工程。后续实验的工程建立方法参照本节设置建立。关于IAR的详细说明文档请浏览 IAR 官方网站或软件安装文件夹下 8051\doc 里的支持文档。

本次实验通过一个简单的 LED 闪灯测试程序工程带领用户逐步熟悉 IAR for 51 实验开发环境。

实验主要过程阐述

1. 安装IAR环境
2. 建立模版工程样例
3. 配置工程选项

   需要配置目标传感器部分的选项，使得开发环境与CC2540能够配合工作。
4. 连接开发板与计算机

   开发板端为Micro-USB口，计算机端为USB串口。另需一根RS232串口线连接开发板与计算机，进行串口通信的操作。注意要操作换位置的按钮，选中当前使用的传感器。由于该实验对传感器没有特殊要求，因此我们随便选择蓝牙传感器中有两个LED灯的即可。

1.2 实验现象观测

下载成功后，程序便开始运行。可以看到传感器上的LED灯有规律地闪烁。我们修改了`Delay`函数的时间，发现数值越大，闪烁频率越小。

1.3 实验原理分析

- 所有程序都需要在计算机上编译好之后，将二进制程序流烧写进传感器的存储部件中（根据TI的文档，应当是一个可编程的Flash，大小在128KB或256KB）
- 传感器在上电后会运行这个程序的主函数（ `void main` ），一般来讲在主函数内写一个`while(1)`的循环就能使程序一直运行（在上电的情况下）。

1.4 实验总结和思考

1. 破解软件花了很久，学校在这方面要加油。
2. 学会了控制硬件，链接芯片。

2 LED灯控制实验

2.1 课内实验内容描述

使用 IAR 开发环境设计程序，利用 CC2540 的 IO 控制 LED 外设的闪烁。程序通过配置 CC2540 IO 寄存器的高低电平来控制 LED 灯的状态，用循环语句来实现程序的不间断运行。

2.2 实验现象观测

绿色LED灯等时闪烁，红色LED灯常亮。更改LED灯的`Delay(interval)`时间，闪烁频率下降。

2.3 实验原理分析

- `#define LED0 P1_1 //定义 LED0 为 P11 口控制`
- `P1DIR |= 0x2; //P1.1 输出`

此处，P1DIR是IO输入输出寄存器。使用该寄存器，保存管理灯的状态之变量。

2.4 实验总结和思考

本次实验是最为简单的传感器实验。几乎所有传感器都有LED部件。可以使用LED灯当作Debug方法进行简单的调试。

3 定时控制实验

3.1 课内实验内容描述

使用 IAR 开发环境设计程序，利用 CC2540 的 Timer1、Timer2 定时器控制 LED 外设的闪烁。程序通过配置 CC2540 处理器的 Timer1 定时器进行自动装载计数，通过查询 IRCON 中断标志来检查 Timer1 定时器T1CTL计数溢出中断状态，从而控制 LED 灯的闪烁状态。

3.2 实验现象观测

LED灯等时快速闪烁，闪烁若干次后熄灭，随后又开始等时快速闪烁。

3.3 实验原理分析

`T1CTL` 是一个变量寄存器，在程序初始化程序中，通过给这个变量赋值来设定中断的具体配置。在本程序中，`T1CTL = 0x0d`表示：中断无效、128分频、自动重装模式。

T1CTL(0XE4)	**Timer1控制寄存器**
BIT3，BIT2：定时器分频倍数选择	00：不分；01：8分频；10：32分频；11：128分频
BIT1，BIT0：定时器模式选择	00：暂停；01：自动重装：0X0000–0XFFFF；10：比较计数：0X0000–T1CC0；11：PWM方式

传感器的定时器有一个工作频率。系统在不配职工作频率时 默认为2分频，即$32M/2=16M$，所以定时器每次溢出时$T=1/(16M/128)*65536=0.5s$

3.4 实验总结和思考

本次实验对于中断有了初步的认识。认识中断、了解中断应该是嵌入式系统的必要功课。

4 片上温度AD实验

4.1 课内实验内容描述

使用 IAR 开发环境设计程序，利用 CC2540 的内部温度传感器作为 A/D 输入源，将转换后的温度数值利用串口发送给 PC 机终端。

1. 配置串口终端，波特率 57600、8 位、无奇偶校验、无硬件流模式
2. 编译调试程序，将程序下载到传感器上
3. 观察屏幕输出，用手捂盖住传感器查看结果有无变化

4.2 实验现象观测

屏幕上输出`temperature:  xx`的字样，间隔时间相等

4.3 实验原理分析

- 模数转换之电压数值：系统的ADC（A-D Convertion）完成了大部分工作，容易得到期间内部的工作电压（Vdd）数值，借此与基准值比较进行计算即可得到电压。

- 模数转换之温度数值：采用当前热敏电阻的电压与温度在22摄氏度的电压相对比，采用代码 `temp = 22 + ( (value - voltageAtTemp22) / TEMP_COEFFICIENT );` 直接返回当前温度。

4.4 实验总结和思考

本次实验涉及到了A/D转换。这应该是物联网中最核心的功能之一。这是一个连续的世界，但是计算机永远都是离散值。A/D转换应该是连接计算机与现实的桥梁。

5 串口收发数据实验

5.1 课内实验内容描述

使用 IAR 开发环境设计程序，利用CC2540的串口0进行数据收发通讯。也即向传感器串口发送指令，控制LED灯的开灭。

1. 配置~~超级终端~~串口调试助手（超级终端遇到Bug，无法向传感器发送指令）
2. 打开串口，设置波特率 57600、8 位、无奇偶校验、无硬件流模式
3. 下载并运行程序

5.2 实验现象观测

预期现象：

发送	现象
11	LED1开（红灯开）
10	LED1关
21	LED2开（绿灯开）
20	LED2关

注意到有时灯光并不是像我们预测的那样运行。反复查找错误后，发现这样的误命令发生在没有输回车，发送了多个连续命令的时候。串口会把上一个字符串没有发完的内容当作下一个字符串的第一位，造成错误。

实验原理分析

维持串口通信的程序中包括 CLKCONCMD 和 CLKCONSTA 控制寄存器，用来控制系统时钟源和状态，SLEEPCMD 和 SLEEPSTA 寄存器用来控制各种时钟源的开关和状态。PERCFG 寄存器为外设功能控制寄存器，用来控制外设功能模式。 U0CSR、 U0GCR、 U0BUF、U0BAUD等为串口相关寄存器。

可以看到串口通信是非常依赖时钟的。

5.3 实验总结和思考

串口通信看似简单实则复杂。在更底层有包括校验位、硬件控制等等约束进行封装。在我们程序的C代码中，需要处理的只有发生I/O的中断方法。

6 声响/光敏传感器实验

6.1 课内实验内容描述

使用 IAR 开发环境设计程序，利用 CC2540 的 I/O 中断来监测声响/光敏传感器的状态。

1. 打开串口，设置波特率 57600、8 位、无奇偶校验、无硬件流模式
2. 下载并运行程序

6.2 实验现象观测

1. 通过声音和光照来控制同一个灯。
2. 实验室里比较嘈杂注释掉，控制声音的代码可以更清楚看到光照的变化。
3. 每次光照有变化的时候，灯会改变状态。

6.3 实验原理分析

本实验主要在中断的控制。基本上逻辑都在中断里。

```
/*************************************************************************
//中断服务程序(P1_2、P1_3  端口)
*************************************************************************/
#pragma vector = P1INT_VECTOR
__interrupt void P1_ISR(void)
{
    if((P1IFG&0x01) > 0) //中断
    {
        prints("Sound Switch!\r\n");
        P1IFG &= ~(0x01);
        LED1 = !LED1;
        Delay(1000);
    }
    if((P0IFG&0x02) > 0)
    {
        prints("Light Switch!\r\n");
```

```
            P0IFG &= ~(0x02);
            LED1 = !LED1;
            Delay(1000);
        }
        P0IF = 0; //清中断标志
        P1IF = 0; //清中断标志
}
```

两个中断控制相同的LED灯，那么同一时刻只有一个中断在接手程序。

6.4 实验总结和思考

中断用得好的话，可以很容易地做到一些仅凭线性程序很难做到的事情。比如楼道里的声控开关。怎么让开关在白天的时候不会开启？在每次声控中断被触发时，判断光感即可。如果是线性编程，由于冬夏两季白天时间不同，很难做到准确控制。

7 磁场强度传感器实验

7.1 课内实验内容描述

阅读磁场强度传感器芯片文档，熟悉该传感器的使用及时序操作。使用 IAR 开发环境设计程序，利用 CC2540 的IO口来模拟I2C读取三轴加速度传感器的状态。

1. 打开串口，设置波特率 57600、8 位、无奇偶校验、无硬件流模式
2. 下载并运行程序

7.2 实验现象观测

置磁铁于不同方位，三轴磁场强度发生变化

7.3 实验原理分析

系统配套的三轴加速度传感器，与蓝牙 4.0 模块的 P0_1、P1_0 管脚相连，这样我们可以知道，三轴加速度传感器模块的时钟线与蓝牙 4.0 模块的 P0_1 IO 引脚相连，数据线与 P1_0 IO 引脚相连。因此我们需要在代码中将相应引脚进行输入输出控制模拟该传感器时序，来读取磁场强度传感器状态。

程序序通过配置 CC2540 处理器的 IO P1_2、P1_3 引脚来模拟磁场强度传感器 IIC 时序，进而取得传 感器的状态，如果顺利采集到磁场强度值，则在串口输出相应的磁场强度数据

```
X = buf[0]<<8 | buf[1];
Y = buf[2]<<8 | buf[3];
Z = buf[4]<<8 | buf[5];
fx = (float)(X > 0x7ff?(X | 0xF800):X)/1090;
fy = (float)(Y > 0x7ff?(Y | 0xF800):Y)/1090;
fz = (float)(Z > 0x7ff?(Z | 0xF800):Z)/1090;
prints("Hx=");
memset(mfs,0,10);
sprintf(mfs, "%5.2f",fx);
prints(mfs);
prints("\r\n");
```

其中，`buf` 由`mfsread()`与总线交互进行更新。

7.4 实验总结和思考

本次实验现象明显，接触到了总线的相关知识，总线运行起来性能良好。但是物理知识有些忘记，无法解释这个磁铁与传感器位置对数值的影响。不过日常生活中用得比较频繁的功能就是手机的横屏竖屏转换。不得不说苹果公司的创新能力和嵌入式系统设计能力了得，小小一个传感器，带来了多广阔的使用场景。

8 干簧门磁/霍尔开关传感器实验

8.1 课内实验内容描述

使用IAR开发环境设计程序，利用 CC2540 的 IO 中断来监测干簧门磁/霍尔开关传感器的状态。

> 干簧门磁-霍尔开关模块主要由干簧管和霍尔传感器组成。
>
> 干簧管（Reed Switch）也称舌簧管或磁簧开关，是一种磁敏的特殊开关。干簧管的工作原理非常简单，两片端点处重叠的可磁化的簧片、密封于一玻璃管中，两簧片分隔的距离仅约几个微米，玻璃管中装填有高纯度的惰性气体，在尚未操作时，两片簧片并未接触、外加的磁场使两片簧片端点位置附近产生不同的极性，结果两片不同极性的簧片将互相吸引并闭合。依此技术可做成非常小尺寸体积的切换组件，并且切换速度非常快速、且具有非常优异的信赖性。永久磁铁的方位和方向确定何时以及多少次开关打开和关闭。
>
> 霍尔传感器（Hall sensor）是根据霍尔效应制作的一种磁场传感器。霍尔效应是磁电效应的一种，这一现象是霍尔在研究金属的导电机构时发现的。霍尔器件是一种采用半导体材料制成的磁电转换器件。如果在输入端通入控制电流，当有一磁场穿过该器件磁感面，则在输出端出现霍尔电势。在磁场力作用下，在金属或通电半导体中将产生霍耳效应，其输出电压与磁场强度成正比

8.2 实验现象观测

1. 用磁铁接近干簧管，干簧管连通，串口打印`Reed Switch`
2. 用磁铁接近霍尔开关，霍尔开关连通，串口打印`Hall Switch`

靠近左边的黑色霍尔元件，可以见到输出`Hall  Switch`的字样。

程序开始运行后持续输出`Reed  Switch` ，不难观测到干簧门磁的灵敏度要高于霍尔元件。

8.3 实验原理分析

```
#pragma vector = P1INT_VECTOR
__interrupt void P1_ISR(void)
{
    if((P1IFG&0X04) > 0) //中断
    {
        prints("SWITCH1 Warning!\r\n");
        P1IFG &= ~(0x0c);
        LED0 = !LED0;
        Delay(1000);
    }
    if((P1IFG&0X08) > 0)
    {
        prints("SWITCH2 Warning!\r\n");
        P1IFG &= ~(0x0c);
        LED0 = !LED0;
        Delay(1000);
    }
    P1IF = 0; //清中断标志
```

实际运行的代码中看到的是`Hall  Switch`和`Reed  Switch`的字符串。

8.4 实验总结和思考

不同的传感器器件灵敏度不一样，要根据适合的使用场景进行选择。如果是要监测地球磁场判断设备方向，那么干簧门磁器件可能更加合适，但如果是磁铁这样场强较大的磁场，则霍尔元件应该更加合适。

本实验的中断处理和之前的声响/光照相同。

9 红外对射传感器实验

9.1 课内实验内容描述

- 阅读 UP-MobNet-II型系统 ZIGBEE 模块硬件部分文档，熟悉 ZIGBEE 模块硬件接口。学习红外对射传感器检测原理和使用方法
- 使用 IAR 开发环境设计程序，利用 CC2530 的IO中断来监测红外对射传感器的状态。

1. 使用 Select_Col和Select_Row 按键选择 ZIGBEE 仿真器要连接的 ZIGBEE 设备模块(接红外对射传感器)(根据 LED 指示灯判断)。
2. 启动 IAR 开发环境，打开\Sensor\红外对射传感器实验路径下的Exp2.eww。
3. 在 IAR 开发环境中编译、运行、调试程序。
4. 使用 PC 机自带的超级终端连接串口，将超级终端设置为串口波特率 57600、8 位、无奇偶校验、无硬件流模式

9.2 实验现象观测

- 使用单片卡片：无现象。红外传感器没有反应。
- 使用两张卡片叠加：现象明显。在遮盖红外传感器时，LED灯会熄灭。终
- 端收到字符串`IRDS warning!`

9.3 实验原理分析

系统配套的红外对射传感器，与 SENSOR A/D 排针相连，这样我们可以知道，红外对射传感器模块的信号线与 ZigBee 模块的 P0_0 IO 引脚相连。因此我们需要在代码中将该引脚配置成中断输入模式，来监测红外对射传感器状态。

```
/*************************************************************************
//中断服务程序(P1_2 端口)
*************************************************************************/
#pragma vector = P0INT_VECTOR
__interrupt void P0_ISR(void)
{
    prints("IRDS Warning!\r\n");
    if((P0IFG&0x01) == 0x01) //中断
    {
        P0IFG &= ~(0x01);
        LED1 = !LED1;
        Delay(1000);
    }
    P0IF = 0; //清中断标志
}
```

中断控制程序与之前的实验差别不大。

9.4 实验总结和思考

本次实验首次采用了CC2530传感器进行实验，感觉和CC2540差别不大.....也可能是并没有涉及到底层协议的实验之原因。

红外传感器在生活中也有很大的用处。比如汽车计算里程时，使用红外传感器能够快速准确地记录轮胎旋转周数得到距离；一些需要判断有无活物经过的场所也需要红外传感器。

10 温湿度传感器实验

10.1 课内实验内容描述

阅读SHTX0温湿度传感器芯片文档，熟悉该传感器的使用及时序操作。使用IAR开发环境设计程序，利用CC2530的IO口来监测温湿度传感器的状态。

10.2 实验现象观测

10.3 实验原理分析

```
void s_connectionreset(void);
char s_measure(unsigned char *p_value, unsigned char *p_checksum, unsigned char
mode);
void calc_sth11(float *p_humidity ,float *p_temperature);
float calc_dewpoint(float h,float t);
char s_read_byte(unsigned char ack);
char s_write_byte(unsigned char value);
extern void Sht11Delay(uint n);
extern void Sht11Init(void);
//extern void calc_sth11(float *p_humidity ,float *p_temperature);
//extern float calc_dewpoint(float h,float t);
extern char GetHumiAndTemp(float *humi, float *temp);
```

主要的温湿度采集由`sht11.c` 的模拟时序代码完成。此处进行了函数定义。

10.4 实验总结和思考

这是Zigbee传感器的总线总线时序模拟实验。总线的工作很大程度上依赖于时序。如果要获取一个传感器的状态，应当模拟时序编程。

11 TI-BLE 4.0 协议栈入门实验

11.1 课内实验内容描述

协议定义的是一系列通信标准，通信双方需要共同按照这一标准进行正常的数据收发；协议栈是协议的具体实现形式，通俗的理解为代码实现的函数库，以便开发人员调用。

BLE4.0 协议栈就是将各个层定义的协议都集合在一起，以函数的形式实现，并提供一些应用层 API，供用户调用

要实现蓝牙这个协议，需要非常复杂的代码库。这和操作系统编程很像，都是偏向于内核的东西。

1. 安装`TI BLE-CC254x-1.4.0`协议栈
2. 打开工程文件，下载调试

11.2 实验现象观测

下载后，传感器上的灯等时闪烁。

11.3 实验原理分析

和协议栈有关的程序相当于给开发者展示协议栈的实现源码，在此基础上进行编程。这体现了传感器编程的灵活性。

11.4 实验总结和思考

打开工程文件后，左边的文件浏览器就出现了很复杂的目录结构，包括各种库函数和`HAL`等实现基本操作的函数。

12 基于BLE4.0的使能从机notify实验

12.1 课内实验内容描述

使用 IAR 开发环境设计程序，BLE-CC254x-1.4.0 协议栈源码例程 SimpleBLECentral 和 SimpleBLEPeripheral 工程基础上，实现无线连接，从机通知被使能，每隔三秒给集中器发送一次数据。

1. 使用 USB 线连接 PC 机和移动互联网教学科研平台的 CC-debug 接口，将系统配套串口线一端连接 PC机，一端连接移动互联网平台的串口 2(com2)上
2. 打开 exp\BLE4.0\Basic Examples\Projects\ble
3. 编译 EnableNotiFy_Central\CC2540 工程和EnableNotiFy_Peripheral\CC2540DB 下工程，分别下载到两个蓝牙 4.0 模块。
4. 将蓝牙 4.0Central 的串口连接到 PC 机上，打开串口终端，设置波特率为 57600、8 位、无奇偶校验、无硬件流模式。给模块上电，当红色 LED 由闪烁变为常量时，表示连接成功，可以在串口终

端上看到`Notify Value: 3`，其中 3 就是特性值 4 的值 0x03，默认定义的特性值 3 的值为 3。

12.2 实验现象观测

12.3 实验原理分析

- 主机代码的状态管理部分是一个大循环，引入了状态码和`switch`语句让程序能够有状态的切换
- 蓝牙扫描到的设备信息，被存储到专门的数据结构里。
- 主机代码的事件管理是一个条件语句，完成各种事件发生后的程序行为。
- 从机代码主要是一个一直可见的信号发送

12.4 实验总结和思考

13 基于BLE4.0协议栈的点对点通信

13.1 课内实验内容描述

使用 IAR 开发环境设计程序，BLE-CC254x-1.4.0 协议栈源码例程 SimpleBLECentral 和 SimpleBLEPeripheral 工程基础上，实现无线连接，读写特性值 1，进行通信。

13.2 实验现象观测

13.3 实验原理分析

集中器发起建立连接的请求，外部设备响应后，两设备进入连接状态；集中器通过特定的 UUID 进行 GATT 数据服务的发现；发现 GATT 数据服务后，集中器发送要进行数据操作的“特性”的 UUID，节点设备
将这个“特性”的句柄返回给集中器，句柄特性值在属性表中的地址；集中器通过举并进行应用数据的读取和写入操作。

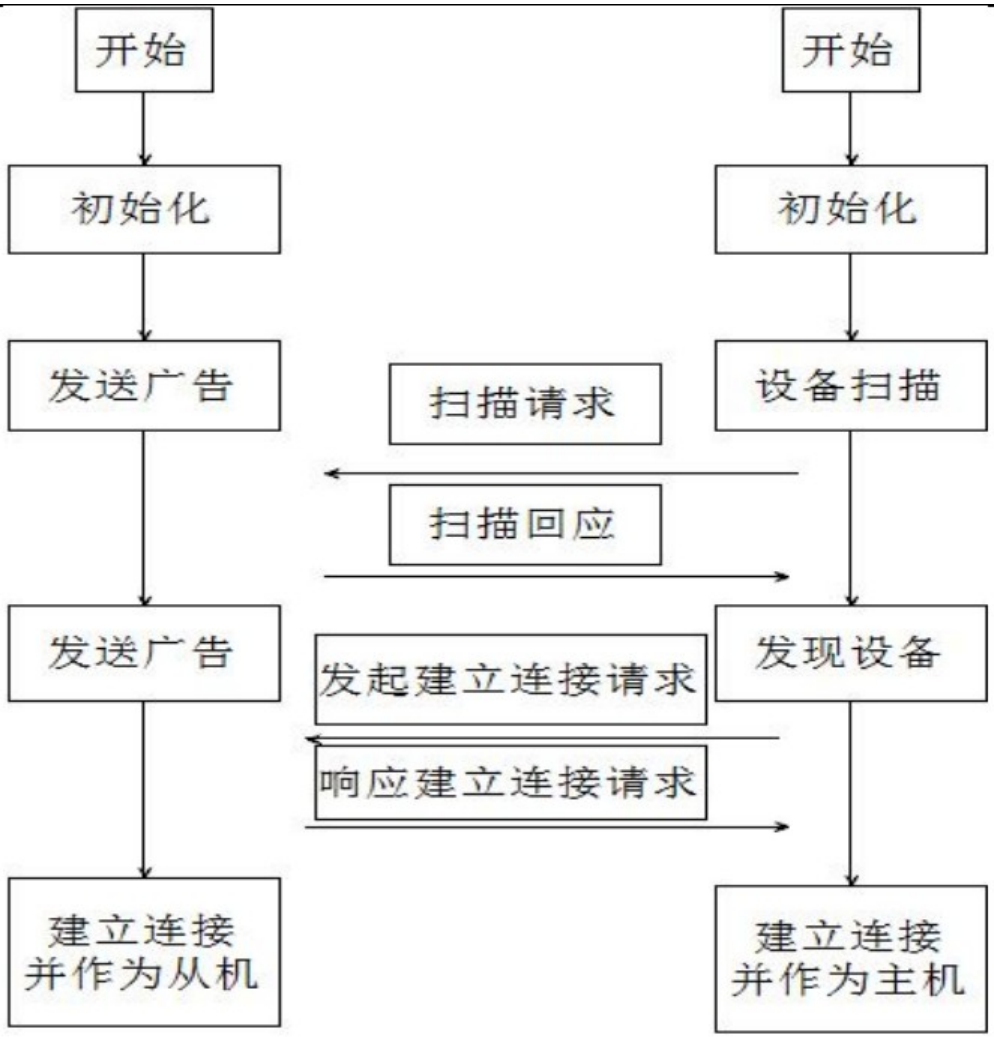

图 5.5.2 建立连接流程图（左侧为节点设备、右侧为集中器设备）

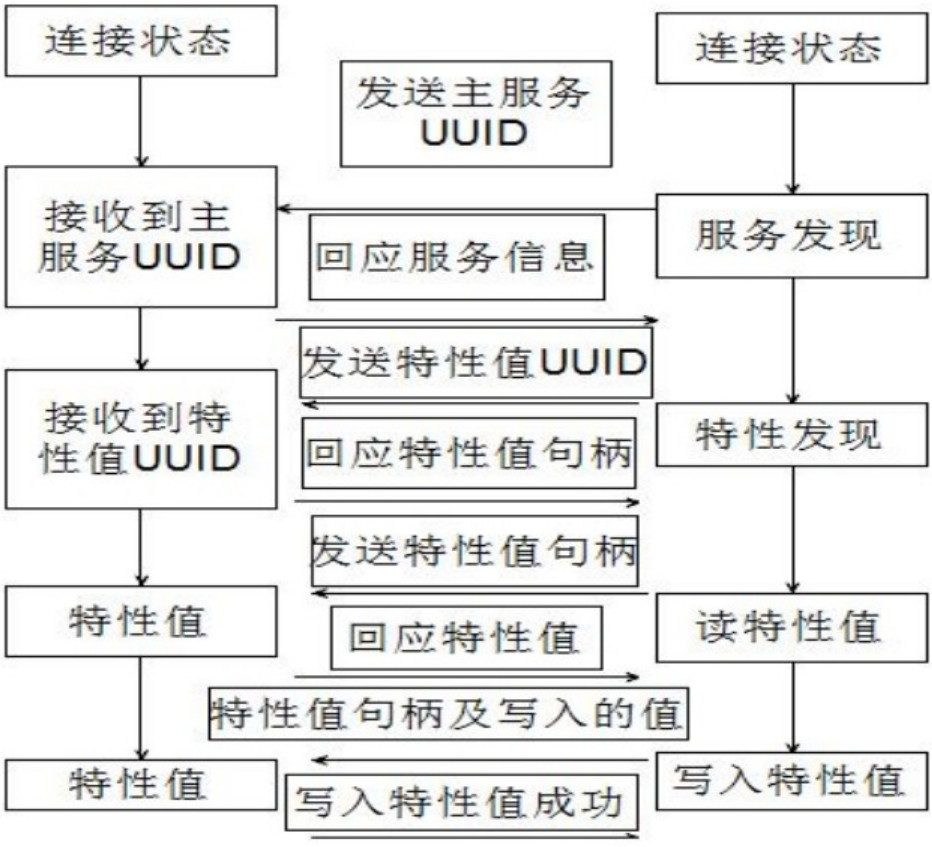

图 5.5.3 传输数据流程图（左侧为节点设备、右侧为集中器设备）

图 5.5.3 传输数据流程图（左侧为节点设备、右侧为集中器设备）

根据老师指导，此处的“广告”实则为“广播”。

不难发现通讯协议中状态的概念很重要。

13.4 实验总结和思考

本次实验对于蓝牙传感器的点对点通信有了基本了解。通过查看较为底层的如何使用蓝牙之方法，我们发现了蓝牙协议的复杂性和连接的可靠性。

联想到日常生活中，鼠标、耳机这些东西我们都不可或缺。但是总有这种情况，那就是都有连接、配对、忘记的操作。可以看出蓝牙协议主要在于状态管理了。相比起Wi-Fi，蓝牙的带宽较小，可靠性也更低。但是处理配件的连接、少量数据的传送，蓝牙很合适。