



实验四、TinyOS实验

一、实验目的

- 1、了解典型nesC的程序结构及语法
- 2、了解tinyos执行机制，实现程序异步处理的方法
- 3、了解tinyos中task抽象及其使用
- 4、在Blink程序中使用printf输出信息，使用task实现计算和外部设备操作的并发



二、预备知识

- 1、课堂讲义
- 2、TinyOS Programming
- 3、TinyOS Blink源程序
- 4、TinyOS printf组件的使用

http://tinyos.stanford.edu/tinyos-wiki/index.php/The_TinyOS_Printf_Library



三、实验内容

1、(1) Blink程序的编译和下载

(2) 给Blink程序加入printf，在每次定时器事件触发
点亮LED的同时通过串口显示信息

(3) 修改BLink程序，只使用一个Timer，三个LED灯作
为3位的二进制数表示（亮灯为1，不亮为0），按照0-7的
顺序循环显示，同时将数值显示在终端上。

三、实验内容

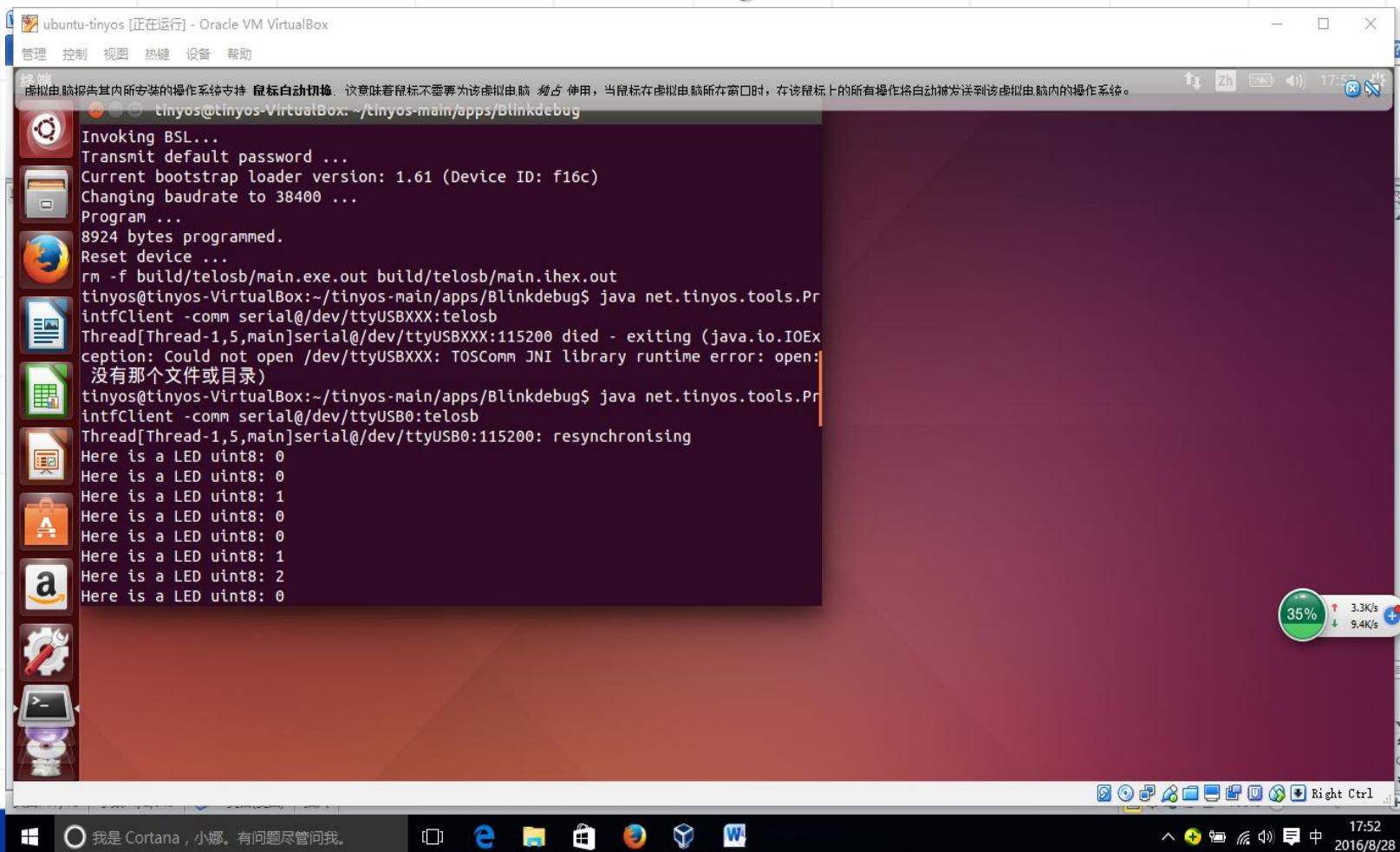
- Tinyos printf

- tos/lib/printf
- 修改Makefile
 - CFLAGS += -I\$(TOSDIR)/lib/printf
 - PFLAGS += -DNEW_PRINTF_SEMATICS
 - CFLAGS += -DPRINTF_BUFFER_SIZE=XXX
- 在组件实现文件加入
 - #include "printf.h "
 - components PrintfC;
 - components SerialStartC;
- printf("Here is a uint8: %u\n", dummyVar1);
- printfflush();

三、实验内容



java net.tinyos.tools.PrintfClient -comm serial@/dev/ttyUSBXXX:telosb





三、实验内容

2、修改Blink程序，在timer0的触发事件处理中加入计算

```
event void Timer0.fired()
{
    uint32_t i;
    dbg("BlinkC", "Timer 0 fired @ %s.\n",
sim_time_string());
    for(i=0;i<400001;i++)
        call Leds.led0Toggle();
}
```

观察LED亮灯的情况，分析其原因，将400001改为10001，再观察并进行分析，加入printf进行输出。

三、实验内容

3、采用task实现计算

```
uint32_t i;
task void computeTask()
{
    for (i=0;i < 400001; i++) {}
}
event void Timer0.fired()
{
    dbg("BlinkC", "Timer 0 fired @ %s.\n",
sim_time_string());
    call Leds.led0Toggle();
    post computeTask();
}
```

观察LED亮灯的情况，分析其原因，将400001改为10001，再观察并进行分析。



三、实验内容

4、请修改computetask的内容，将400001次计算分割成为若干小的部分，从而使得LED1和LED2的fire事件可以被正常调用，并通过printf输出。



加分、TinyOS传感器数据获取实验

一、实验目的

- 1、了解Telosb节点中传感器的类型与使用；
- 2、了解Telosb节点的传感器数据的获取；
- 3、获取的数据通过printf传输至电脑；
- 4、将节点的传感器数据传输到基站，并在电脑端解析显示，了解数据的采集过程。



二、预备知识

1、课堂讲义

2、TinyOS Programming

3、TinyOS Sense源程序

4、SensirionSht11C()组件的使用

5、HamamatsuS1087ParC()组件的使用



三、实验内容

1、传感器种类：

温度传感器、湿度传感器、光照传感器

telosb的温湿度集中在一个传感器上，该传感器名为SHT11，是Sensirion公司的产品，光照传感器使用的是Hamamatsu的S1087，两者的数据手册都可以从网上搜索到。

温湿度组件：SensirionSht11C()

光照组件：HamamatsuS1087ParC()

三、实验内容

2、温度、湿度、光照强度计算

温度：（摄氏度）

$$\text{temp} = -40.1 + 0.01 * \text{val};$$

相对湿度：（百分比）

$$\text{RH}_{\text{linear}} = c_1 + c_2 * \text{SO}_{\text{RH}} + c_3 * \text{SO}_{\text{RH}}^2$$

SO_{RH} 为12-bit的测量值 $c_1 = -4$, $c_2 = 0.0405$,
 $c_3 = -2.8 * 10E-6$

光照强度：（勒克斯Lux或Lx）

$$I = \text{AD} * 1.5 / 4096 / 10000$$

$$\text{LUX} = 0.625 * 1e6 * I * 1000 \quad (\text{S1087})$$

$$\text{LUX} = 0.769 * 1e6 * I * 1000 \quad (\text{S1087-01})$$

其中AD为12-bit测量值



三、实验内容

3、典型的办公室温度. 湿度. 光照强度测量值

serial@/dev/ttyUSB1:115200: resynchronising

received temperature:24.17°C

received humidity:55.86%

received photo:572.20Lux

received temperature:24.18°C

received humidity:55.89%

received photo:549.32Lux



三、实验内容

4、代码实现

头文件：

```
#include "TestSensor.h"  
#include "SensirionSht11.h"
```

配置文件：

```
components new SensirionSht11C();  
components new HamamatsuS1087ParC();
```

```
TestSensorC.readTemp -> SensirionSht11C.Temperature;  
TestSensorC.readHumidity -> SensirionSht11C.Humidity;  
TestSensorC.readPhoto -> HamamatsuS1087ParC;
```

```
uses interface Read<uint16_t> as readTemp;  
uses interface Read<uint16_t> as readHumidity;  
uses interface Read<uint16_t> as readPhoto;
```



三、实验内容

4、代码实现

读取传感器函数接口：

```
event void SampleTimer.fired()
{
    call readTemp.read() //读取温度值
    call readHumidity.read() //读取湿度值
    call readPhoto.read() //读取光照值
}
```



三、实验内容

4、代码实现

传感器读取实现：（三个传感器的读取类似）

```
event void readTemp.readDone(error_t result,  
uint16_t val) {  
  
    if (result == SUCCESS) {  
        // 在接收端进行解析  
        TempData = val;  
    }  
    else TempData = 0xffff;  
    sendData(TempData, 1);  
    或用printf输出  
}
```



三、实验内容

4、代码实现

接收端数据解析处理（java 类）

```
public void messageReceived(int to, Message message) {  
    TestSensorMsg msg = (TestSensorMsg)message;  
    int type = msg.get_nodeid();  
    double tempature;  
    double humidity;  
    double photo ;  
    switch(type){  
        case 1:  
            temperature = -40.1 + 0.01*msg.get_counter();  
            System.out.printf("received temperature:%.2f", tempature);  
            System.out.println( "°C");  
            break;  
        case 2:  
            humidity = -4 + 0.0405*msg.get_counter() + (-2.8/1000000) *msg.get_counter() *msg.get_counter();  
            System.out.printf("received humidity:%.2f" , humidity);  
            System.out.println("%");  
            break;  
        case 3:  
            photo = msg.get_counter() * 1.5 / 4096 / 10000;  
            photo = 0.625 * 1000000 * photo * 1000;  
            System.out.printf("received photo:%.2f" , photo);  
            System.out.println("Lux");  
            System.out.println();  
            break;  
        default:  
            System.out.println("received unknow data:" + msg.get_counter());  
            break;  
    }  
}
```