

飞思卡尔杯第六届设计应用大奖赛 决赛设计论文

设计名称 : 手势识别器
比赛编号 : 1000142
通讯地址 : 深圳市南山区西丽大学城哈工大校区C栋102B
邮编 : 518055
电子邮箱 : hitpir@gmail.com
电话 : 0755-26033990 13632542544
推荐单位名称 : 哈尔滨工业大学深圳研究生院学生科学与技术协会
单位地址 : 深圳市南山区西丽大学城

小组成员

姓名:	<u>刘钊</u>	(中文)	<u>Liu Zhao</u>	(英文)	职业:	<u>硕士研究生</u>
姓名:	<u>原志杰</u>	(中文)	<u>Yuan Zhi Jie</u>	(英文)	职业:	<u>硕士研究生</u>
姓名:	<u>徐维昌</u>	(中文)	<u>Xu Wei Chang</u>	(英文)	职业:	<u>硕士研究生</u>
姓名:	<u>宋翔</u>	(中文)	<u>Song Xiang</u>	(英文)	职业:	<u>硕士研究生</u>
姓名:	<u>张世琦</u>	(中文)	<u>Zhang Shi Qi</u>	(英文)	职业:	<u>硕士研究生</u>
姓名:	<u>黄棋波</u>	(中文)	<u>Huang Qi Bo</u>	(英文)	职业:	<u>硕士研究生</u>

摘要.....- 2 -

引言.....- 2 -

一、.....- 2 - 手势识别的控制原理

二、.....- 3 - 系统描述

1.....- 3 - 系统组成及功能

2.....- 4 - 预定义的动作

3.....- 5 - 演示终端

三、.....- 6 - 硬件描述

四、.....- 7 - 软件描述

1.	- 8 -	单片机硬件初始化
2.	- 9 -	数据采集
3.	- 10 -	动作检测
4.	- 10 -	SCI通信
五、	- 11 -	演示终端
1.	- 11 -	J2ME平台开发环境配置
2.	- 11 -	软件应用程序的流程
3.	- 12 -	软件应用程序的组织结构
六、	- 17 -	结论

手势识别器

宋翔,原志杰,徐维昌,张世琦,黄棋波,刘钊
哈尔滨工业大学深圳研究生院

摘要

本文介绍了一种利用加速度传感器对手持设备进行导航控制的小型系统。本系统使用Freescale MC9S08QG单片机为核心,利用MMA7260Q加速度传感器的采样值,设计了一种运动判定方案来识别一些预定义的动作,并将控制指令发送到手持设备上使其产生相应的动作。测试表明,该控制系统的姿势识别率较高。

引言

现在消费电子设备已近普及到了每个人,从手机、MP3播放器、硬盘播放器到数码相机、PDA,每种设备都拥有导航键对其进行控制。目前比较流行的导航键控制方式有四维键、摇杆,这是最常见的两种导航键,此外还有一些手机上有很新颖的导航键设计比如:LG-KG70的滚轴键、LG-KE608的转盘设计、索爱W830的触摸式、多普达D802的飞梭滚轮等等。总之导航键的设计趋势是取代传统的四维键和摇杆,使其更新颖、时尚。

这里，我们用加速度传感器设计一种看不见的导航键来代替四维键的功能，这种方案更能满足消费者的好奇心，满足消费者追逐时尚的需求。此外，在需要导航键控制的任何消费电子设备上都可以使用这种看不见的导航键，其应用市场广阔，并且设备成本极低。

手势识别器的主要核心部分由加速度传感器和单片机两部分组成。其中，加速度传感器负责加速度值的采集，单片机的主要功能包括A/D数模变换、抽样数据处理、发送控制指令。其中抽样数据的处理是系统的关键。

一、 手势识别的控制原理

本系统利用3轴加速度的值来判定对物体运动预定义的6种姿势。首先分别对3个轴采样，每个轴各获得50个数据，然后分别对每个轴上的数据进行处理来判定是否发生了预定义的动作。动作定义在下面的部分说明，这里仅用Y轴来说明判定的原理。

获得Y轴上的50个数据存放到数组y_data[N]中，将这50个数求和取平均值。若y_data[N]中数据最大值与最小值之差在一个设定的阈值之内，则认为物体在Y轴向上是没有动作的、静止的，此时更新y_init值为y_data[N]求得的平均值；否则，y_data[N]中数据最大值与最小值之差超出一个设定的阈值，则认为物体是运动的，y_init值不变，仍然为上一次静止状态时的值。

下面两个图是Y轴分别向左、向右摇动时采样得到的加速度抽样值y_data[N]。图中红线代表上一次静止时的采样值，蓝线代表运动时的采样值。图1为向左摇动时的值，可以明显看出加速度的值较静止时有明显的变化，向着增大的方向变

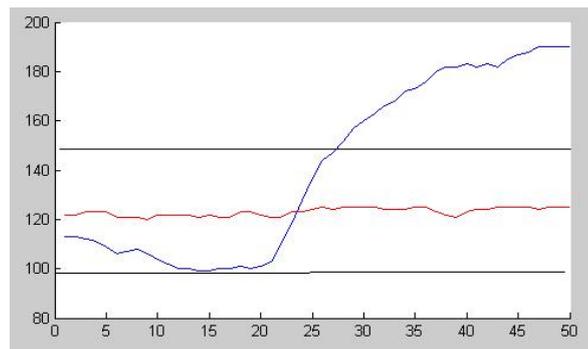


图1 向左摇

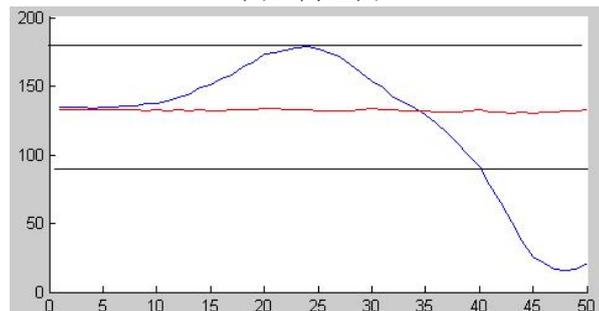


图2 向右摇

从以上两个图中可以看出，两条黑线之间的数据很难断定是那个动作产生的，因为两个动作都可能产生这样的值。所以，利用黑线之外的数据来判定是向右还是向左摇。同时对于这两个动作，黑线之外的数据有明显的差异，数据相差很多。对于图1这些数据大于150，图2中这些数据小于90。因此这些数据至少相差60，可以很容易的将向左、向右的两个动作给区分开。

本系统就是根据这样的原理来实现的，首先分别设定向左、向右摇时的阈值和两个计数器；然后，将新采样得到的50个值存放在y_data[N]中，将每个值与静止时的值y_init比较，如果数据超出某个阈值，则在相应的计数器上累加，直到所有50个数全部比较完

毕；最后根据计数器值的大小来判定执行了某个动作。

同理，利用Z轴的采样值来判定向上、向下摇，利用X、Y两个轴来判定执行图片向左翻转还是向右翻转，这里略去。

二、 系统描述

1.

系统组成及功能

系统由三个部分组成：CT-298单片机控制部分、MM-2860加速度传感器部分、演示终端。系统框图如图3所示。

CT-298单片机控制部分主要负责传感器数据的A/D变换、动作检测、与演示终端进行数据通信。MM-2860加速度传感器部分负责加速度值的测量并将其转换为电压值。演示终端负责处理从单片机传来的动作类型并执行相应的动作，同时向单片机返回进出二级菜单的指令。简单来说就是CT-298采集MM-2860上传来的数据，处理后判定是哪一种动作姿态，并将这种动作代码发送给演示终端

端，演示终端根据不同的动作代码执行不同的演示动作。对于CT-298单片机系统更详细的设计将在下面的部分介绍，这里首先介绍一下本系统设计的动作姿势及代码，以及与演示系统交互的指令。

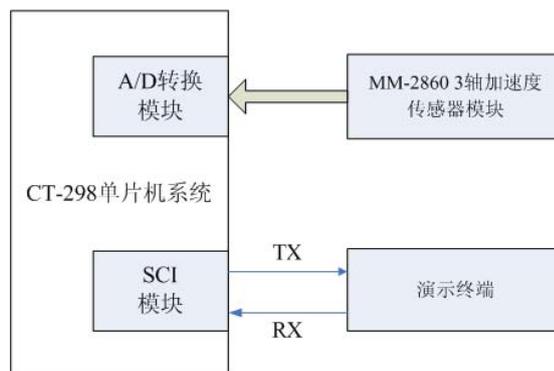


图3 系统框图

2.

预定义的动作

系统设计了六种动作，他们分别是左摇、右摇、下摇、上摇、图片左转、图片右转。下面两幅图形象的说明了其中的两个动作，其它的动作与此类似。



图4 右摇

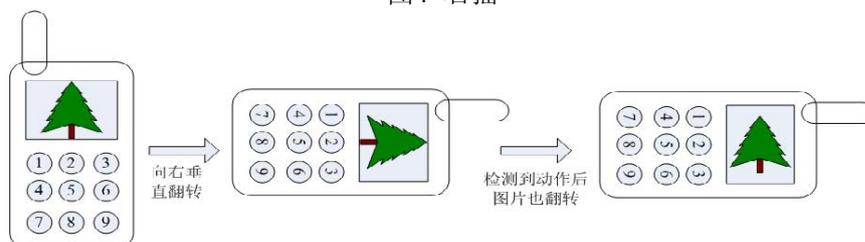


图5 图片向左翻转

系统还为每个动作定义了相应的动作代码，见表1：

表1 动作定义

动作定义	动作代码	动作宏定义
左摇	0x01	MOVE_LEFT
右摇	0x02	MOVE_RIGHT
下摇	0x04	MOVE_UP
上摇	0x08	MOVE_DOWN
图片左转	0x10	MOVE_TURN_LEFT
图片右转	0x20	MOVE_TURN_RIGHT
无动作	0x00	NO_ACTION

当单片机检测到某个动作时会将相应的值赋给变量type_action，之后通过SCI将type_action的值发送到演示终端。如果用户执行的动作不在这6个动作之中，则将NO_ACTION赋给type_action，表示演示终端不执行任何动作。

系统还定义了单片机与演示终端交互的指令，用于系统在主菜单和二级菜单之间的界面切换，指令从演示系统通过SCI传给单片机，单片机接收到之后进入或则退出二级菜单。见表2：

表2 指令定义

指令定义	指令代码	指令宏定义
进入图片二级菜单	0xF1	PICTURE_ENTER
退出图片二级菜单	0xF2	PICTURE_OUT

这里仅定义了一个二级菜单，即图片菜单，系统还可以定义更多的二级菜单，并且还可以定义三级菜单。

此外，需要说明动作MOVE_TURN_LEFT，MOVE_TURN_RIGHT用于图片翻转，仅在图片菜单可用，在主菜单不可用。动作下摇执行的是换下一幅图片，动作上摇执行的是退出图片菜单。

3.

演示终端

本系统目标定位在便携式消费类电子产品上，因此在软件应用程序上必须选择具有广泛基础的应用平台。由于近年来J2ME在便携式终端中的应用非常广泛，因此本系统也采用了J2ME平台进行开发。通过比较，选择了J2ME的WTK开发包，它是专门针对移动无线设备而设计的开发包，并提供了一个统一的平台。在WTK的框架下开发出来的java程序可以被众多的移动设备所支持，所以能够有效解决兼容性的问题。

在WTK下我们利用默认的一个手机样机的仿真器DefaultColorPhone进行了开发，DefaultColorPhone的样子如图6。



图6 手机界面

仿真器的外观和操作类似一部移动电话，但是并不代表某个特定的设备，而是提供对其所支持的 API 的正确实现，每个命令按钮对应着相应的API函数。从图中可以看到手机的导航键，我们更改了导航键的API函数，使其从鼠标单击触发的方式转化为串口动作代码控制。首先，我们编写了J2ME的串口界接收程序，用于接收从单片机传来的单字节的type_action值；然后将type_action以参数的形式传给API,根据不同的type_action值手机执行不同的动作。这些动作包括菜单上下翻、进出二级子菜单、图片翻转。手机动作与type_action的对应关系如表3：

表3 目录与动作代码的对应关系

主菜单目录		图片目录	
手机动作	type_action	手机动作	type_action
上一个菜单	0x04	下一个图片	0x04
下一个菜单	0x08	退出	0x08
进入	0x02	向左翻转	0x10
退出	0x01	向右翻转	0x20

根据type_action的值，在手机界面上产生相应的动作，手机界面发送不同的变化。演示终端的具体实现在下面的内容详细介绍。

三、

硬件描述

系统硬件分为两个部分：CT-298和MM-2860。

CT-298是安装有型号为MC9S08QG的单片机、由USB总线电源供电的小型评估板。CT-298上安装有连个按钮开关、LED灯、蜂鸣器，作为开发的输入输出器件。同时，USB-COM转换电路采用了FTDI公司制造的FT232R，容许单片机MC9S08QG与电脑之间通过USB接口进行串行通信。BDM用于代码的烧写及系统的调试。CT-298的结构如下图7所示：

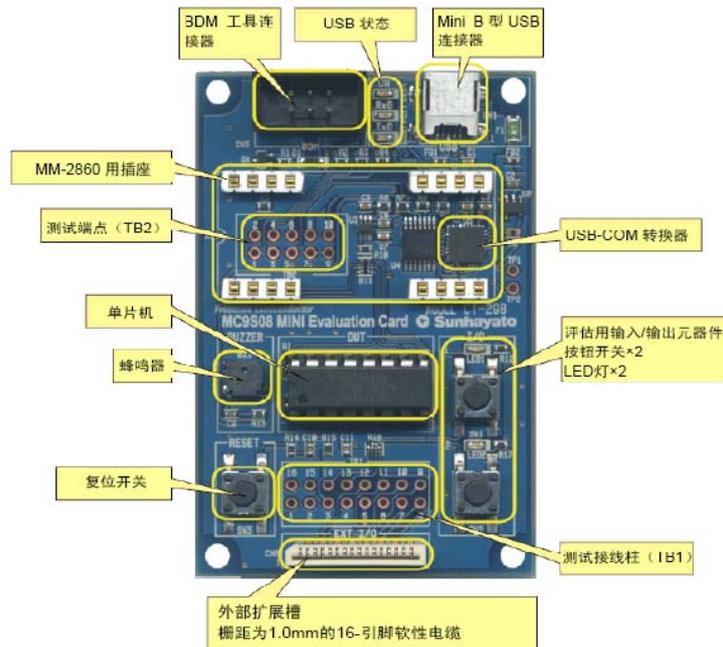


图7 CT-298硬件结构图

MM-2860是含有Freescale公司制造的MMA7260Q型3轴小量程加速度传感器的模块。它可以直接安装在CT-298为其设计的插口上。MM-2860的结构图如图8所示：



图8 MM-2860硬件结构图

MM-2860的电源供应是由CT-298上的MC9S08QG8单片机的PTB5端口来控制的，当PTB5端口为L时电源接通，此外，g-SELECT开关是选择传感器灵敏度的开关，使用时将MM-2860插入到CT-298的接口中即可。本系统采用加速度传感器的灵敏度选择为800 mV/g。

四、 软件描述

单片机主程序的流程如图9所示：

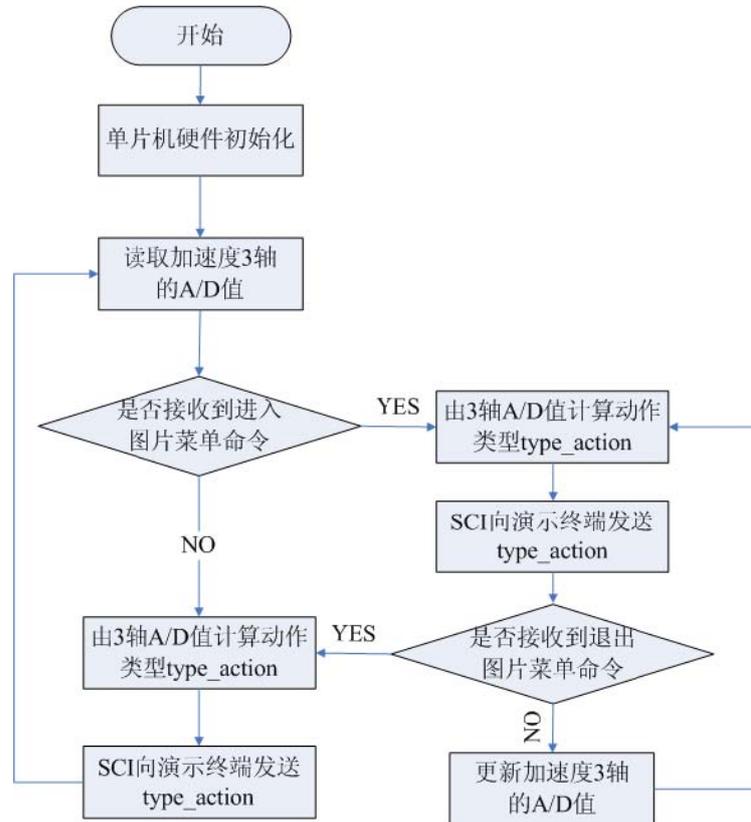


图9 主程序流程图

系统上电之后，首先要对单片机的硬件系统进行初始化、配置寄存器，之后才可以进行数据采集，将加速度3轴的值进行A/D变化，得到量化的值。图片菜单是二级菜单，如果没有接到演示系统传给单片机进入二级菜单的指令，则单片机一直在主程序运行，不断地采集A/D值、进行动作判定，并向演示系统发送type_action的指令。演示系统可根据接收到的type_action的类型采取相应的动作。如果单片机接收到进入二级菜单的命令，则进入图片菜单，同时也执行类似于主程序的动作判定程序，并不断更新A/D采样值，发送type_action，直到接收到演示系统的退出二级菜单指令才退出到主程序。下面详细介绍一下各个功能模块的具体设置。

1.

单片机硬件初始化

单片机系统主要的工作有将加速度传感器的模拟数据进行A/D变化、向演示系统发送type_action的动作类型、接收演示系统发来的进出二级菜单的指令、设置采样值，除此之外还需要对系统时钟、外部设备（灯，buzzer）进行配置。根据单片机的主要工作内容选择单片机内部的功能设备包括A/D变化器、模定时器、串行通信模块（SCI）、内部时钟源模块。下面分别介绍。

内部时钟源（Internal Clock Source）

ICSC1=0x04;

ICSC2=0x80;

ICSTRM=0xAF;

配置内部时钟源寄存器，选用内部时钟作为时钟源，使系统时钟为2MHZ。

A/D变换器

```

void adc_ini(void)    //ADC module initial
{

```

```

ADCSC1=0x5F;          //interrupt enable
ADCSC2=0x00;
ADCCFG=0x00;
APCTL1=0xC1;

```

```

}

```

使用中断方式进行A/D转换，使能端口PTA0、PTB2、PTB3为模拟输入端口，选择BUSCLK为A/D的转化时钟，采用8bits量化方式。中断函数ADC_ISR()负责读取量化以后的数据值。

模定时器（MTIM）

```

MTIMCLK_PS=7;
MTIMCLK_CLKS=0;
MTIMMOD=75;
MTIMSC=0x20;

```

使用定时器用于产生延时程序delay()，时钟频率选择BUSCLK，分频系数取128分频，模值设置为75。完成设置之后可以获得延时为5ms的delay()，这样可以获得采样率为200的A/D值。

串行通信接口（SCI）

```

void sci_ini(word baud_div)          //SCI波特率设定
{
    SCIBD = baud_div;
    SCIC1 = 0x00;
    SCIC2 = 0x24;
}
sci_ini(2000000/16/9600);           //设定波特率为9600

```

SCI是单片机与演示系统进行通信的接口，负责发送动作类型给显示系通、接收从显示系统传来的指令。SCI设定数据传输的波特率为9600 baud/s，数据发送采用轮询方式，数据接收采用中断方式，无校验位。

函数SendChar()用于单片机向演示系统传递type_action的动作类型，中断函数SCIR_ISR(void)用于接收显示系统传来的数据并将其赋值给command，command单片机中定义接收指令的变量。

2.

数据采集

系统设置的采样率为200HZ，每秒钟分别对X、Y、Z 3轴采样200个数据，因此定义了三个大小为N的数组队数据进行缓存，他们是：

```

char x_data[N];
char y_data[N];
char z_data[N];

```

这里N取50，每0.25s存取一次，1s钟可以存取4次，保证采样率为200。函数void acce_meas(void) 负责将采样的数据分别放到这三个数组中，下面是程序的具体实现

```

for(j=0;j<N;j++)          //采集N个数据
{
    for(i=0;i<3;i++)
    {

```

```

if(i==0)
{
    adc_go(0);           //选择A/D信道0
    x_data[j]= ADC_val_L; //X轴
}
else if(i==1 )
{
    adc_go(6);           //选择A/D信道6
    y_data[j]= ADC_val_L; //Y轴
}
else
{
    adc_go(7);           //选择A/D信道7
    z_data[j]= ADC_val_L; //Z轴
}
}
delay();                //延时函数，用来设定采样率
}

```

3.

动作检测

动作检测主要是用获得的3组数据值 $x_data[N]$ 、 $y_data[N]$ 、 $z_data[N]$ 来判定当前动作是系统设定的动作的那一个，如果符合，则发送这个动作对应的预定义值给演示终端，否则发送NO_ACTION。函数`type_dectction()`用来实现动作检测，其中`type_move`为函数内部变量，用于记录动作代码。`type_dectction()`进行动作检测的流程如图10所示：

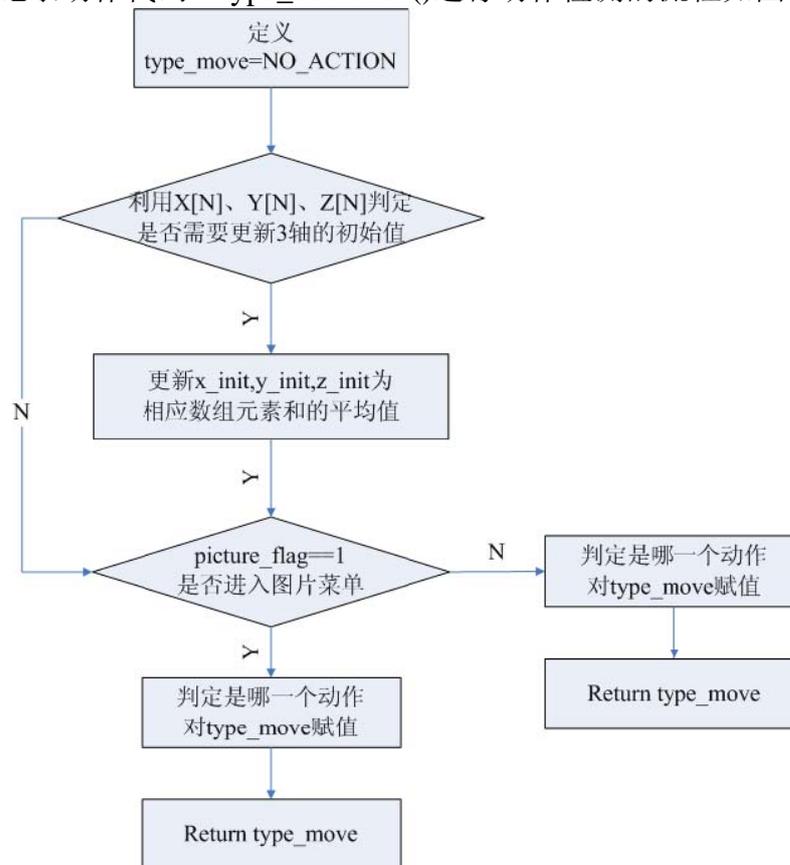


图10 type_dectction()流程图

4.

SCI通信

将函数type_dectction()返回值赋给type_action,利用函数SendChar()将其发送给演示终端, 其具体实现如下:

```

type_action=type_dectction();    //重新采样, 计算
switch(type_action)
{
    case MOVE_UP:
        PTBD_PTBD4=~PTBD_PTBD4;
        SendChar(MOVE_UP);        //发送动作代码MOVE_UP
        break;

    case MOVE_DOWN:
        LED1=~LED1;
        SendChar(MOVE_DOWN);      //发送动作代码MOVE_DOWN
        break;

    case MOVE_TURN_LEFT :
        LED1=~LED1;
        SendChar(MOVE_TURN_LEFT); //发送动作代码向左翻转
        break;

    case MOVE_TURN_RIGHT:
        PTBD_PTBD4=~PTBD_PTBD4;
        SendChar(MOVE_TURN_RIGHT); //发送动作代码向右翻转
        break;

    case NO_ACTION:
    default:
        SendChar(NO_ACTION);      //发送动作代码NO_ACTION
        break;
}

```

此外, 从演示终端传来的进出二级指令的中断接收程序如下:

```

interrupt 15 void SCIR_ISR(void)
{
    if(SCIS1_RDRF)
        command=SCID;
}

```

五、

演示终端

1.

J2ME平台开发环境配置

进行J2ME的开发需要从网络上下载免费的开发环境, 本系统软件部分的开发环境配置如表4所示。

表4 开发软件

名称	类别	开发包全名	版本
Java开发类库	SDK	jdk windows1_5_0_08 i586p.exe	1.5.0
J2ME无线工具开发	SDK	j2me_wireless_toolkit	2.2

包		2_2windows.exe	
JAVA IDE集成开发环境	IDE	eclipse3.2.1win32.zip	3.2.1

运行Eclipse后进行最后的配置：Window->preferences->J2ME->Platform Component 然后右键选择 Wireless Toolkil, 再选择弹出菜单中的 Add Wireless Toolkil, 选择刚刚安装的J2ME Toolkit的安装路径: C:\WTK22, 这样基本配置就配置好了。

J2ME的WTK开发包是专门针对移动无线设备而设计的开发包, 并提供了一个统一的平台。在WTK的框架下开发出来的java程序可以被众多的移动设备所支持, 所以能够有效解决兼容性的问题。

在配置好J2ME的开发环境后, 就可以进行软件开发工作了。

2.

软件应用程序的流程

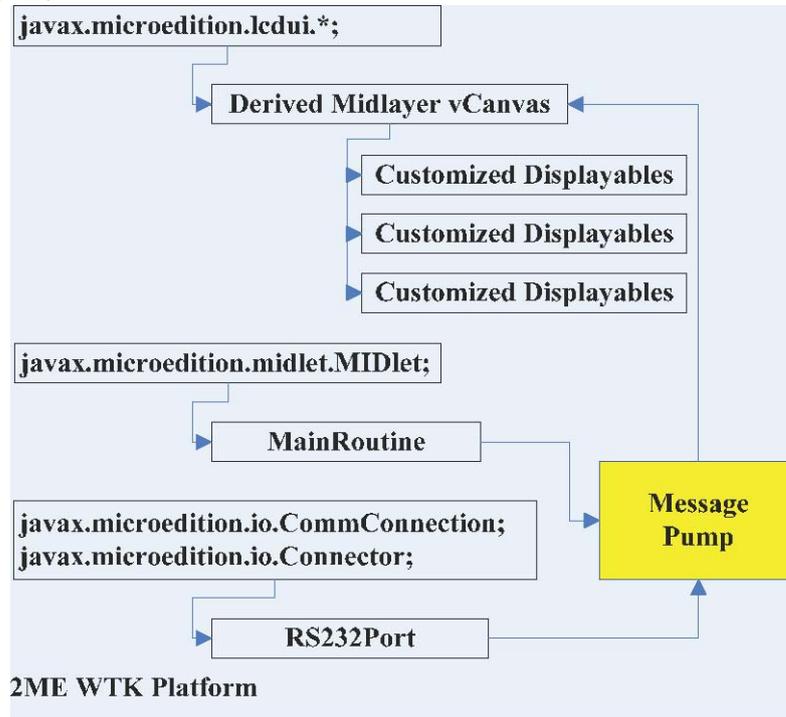


图11 程序流程结构

如图11所示, 程序分为主进程、串口监视模块和定制化用户界面三大部分, 分别基于J2ME不同的类库派生而成, 三部分之间通过消息机制相互联系, 共同构成整个程序的运行周期。具体每个模块所实现的功能参考下节内容。

3.

软件应用程序的组织结构

本软件应用程序在WTK的手机仿真器下进行设计, 分为Base和display两个package。组织结构图如图12所示。



图12 文件层次结构

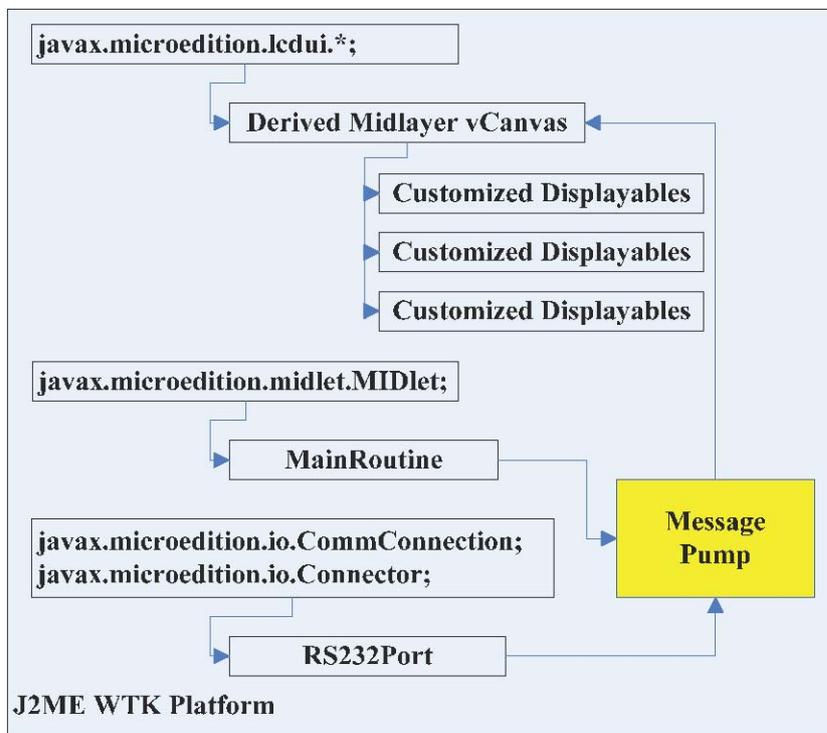


图 11 程序流程结构

所示。程序分为两个进程：串口监听程序和定制化用户界面。

Package: display

从图12可以看出文件系统分为两个Package，在Base Package中包含MainRoutine.java和RS232Port.java两个java文件；在display Package中的文件较多，主要实现了UI和基本的key响应。下面就按照文件的层次结构介绍软件部分的实现方法。

RS232Port类

RS232Port类继承Runnable接口，功能是作为一个串口程序用一个独立线程轮询串口数据，在接到命令后按照指定的规则解释成相应动作送到当前活动vCanvas(自定义画板)的externalMsg函数上。

该串口类线程的每次具体执行过程是：打开飞思卡尔硬件板所在的串口，以9600波特率进行连接并监听，代码如下所示。

```
CommConnection cc = (CommConnection) Connector.open("comm.:com2;baudrate=9600")
```

构造字节数据输入（输出）流，只要串口仍为打开状态，就读取一个字节的数据，然后对所读的数据按照既定规则进行派发 `dispatchcmd()`，传给`externalMsg`函数。

MainRoutine类：

`MainRoutine`为程序的入口类，它整合了程序中的所有的对象。`MainRoutine`派生于`MIDlet`类，重载实现了`MIDlet`中的`startApp`、`pauseApp`、`destroyApp`等方法，并且在`constructor`中加入了`exitCommand`命令，从而实现了手机的关机功能。

下面详细介绍各个方法的实现。

a.

`startApp`：该方法中定义了主程序（UI）的`title`——飞思卡尔嵌入式比赛手持应用程序，同时增加了电子书、图像浏览器、文件浏览器、关于、退出、功能5和功能6等七个项目，但是并没有现在实现这七个项目，而是留给`display`包中的类来实现。

b.

`pauseApp`：由于程序中并没有使用到“暂停”的功能，所以该方法中没有任何语句。

c.

`destroyApp`：当程序调用`destroyApp`的时候，程序将退出并向基本输出打印退出语句。

可以看出`MainRoutine`实现了程序的入口和退出，并同时标志了程序的基本框架，给具体的功能应用打下了坚实的基础。

vCanvas类：

`vCanvas`继承了`Canvas`类，并增加了`externalMsg`方法。之所以在其中增加`externalMsg`方法，是因为其超类`Canvas`不能动态地响应`key`，而我们对`Canvas`的要求是能够通过串口导入的数据（或按键）得到的`keyCode`做出相应的动作，比如图片的翻转、文本的滚动等等。于是，我们可以通过`vCanvas`派生出一系列的子类，实现我们的具体要求，为将来的开发了提供方便。

DisplayItem类：

`DisplayItem`是基础类，提供了通用的`string`数据组合。该类的构造方法中需要输入`shortText`、`longText`和`extra`三个`string`参数，这样就保证了每个`DisplayItem`都可以返回三个不同长度的文本信息，在不同的场合使用。其中`shortText`用于UI中的标题显示；`longText`为Item的主体内容；`extra`为Item的附加信息。

`DisplayItem`类在本系统中会被例化来描述菜单和子功能的内容，是应用广泛的一个基础类。

FileCanvas类：

FileCanvas的超类为vCanvas，这样就使得其拥有响应key的入口，同时实现了Canvas的抽象方法paint()。该方法实现了对文件系统的浏览，结果如图13所示：



图13 文件浏览器效果图

从图中可以看出文件浏览器完成了文件系统的浏览功能，文件夹可以自由进入、退出，通过该类用户可以选择浏览具体的文件。FileCanvas被从上到下分割为Header、Body和Footer三个部分，Header部分显示该类的title，Body显示主体内容，Footer显示键盘的相应操作。相应的在程序中就实现了paintHeader、paintBody和paintFooter三个方法。

FileCanvas的主体部分实现了对文件系统内容的判断和显示，具体来说就是分辨根目录内容是文件夹还是文件。如果是文件夹的话，相应的就要实现它的进入和退出的功能，并且在其显示名称之后增加“/”以表示其可以进入。如果是文件的话，就可以实现其相应的打开操作。在当前的版本中仅提供浏览功能，并不能够直接打开对应的文件。

ImageCanvas类：

ImageCanvas同样继承于基础类vCanvas，它的主要功能是实现了对图片的浏览及旋转、镜像等基本操作。

考虑到目前使用手机浏览图片、拍摄图片的多方向性，如果能够通过加速度传感器自动校正图片的方向，使它自动满足我们想要方向，那么我们就免除了很多不必要的麻烦。于是我们对图片的浏览功能模块增加了向左或者向右旋转90度的功能。这样用户就会惊喜地发现无论他的手机怎样摆放，显示的图片将永远保持与地面垂直。



图14 原始图片与翻转后的图片

ItemCanvas类：

ItemCanvas同样继承了vCanvas类，实现了浏览文本的功能，构造方法的输入依次为Display、Displayable、Font和DisplayItem。从以上对DisplayItem的介绍中可以看出，文本内容来自DisplayItem的LongText，DisplayItem的ShortText部分为标题。Displayable保存着前一个正在显示的Display，Font为该文本浏览器显示的字体。

显示文本的时候，可以响应键盘的上、下、左、右方向键，也可以通过串口导入的手势来控制文本的滚动和退出等操作。

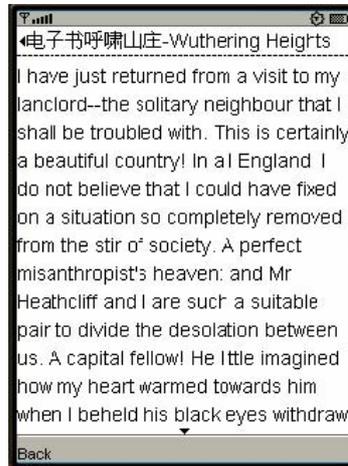


图15 电子书浏览器效果图

ListCanvas类:

ListCanvas提供了主画面的显示能力，并将用户操作派生到内部聚合成员上面去。具体功能的Item将包含在ListCanvas之中，每一个Item拥有显示在屏幕上的ShortItem，显示在detail screen中的LongText，还有不被显示的ExtraText，当然其信息我们也是可以得到的。在ListCanvas中，我们可以用“右键”显示Item的详细信息，也可以用SELECT键来标记每个Item。



图16 应用程序顶层界面效果图

如图16所示，我们的主菜单在ListCanvas中正确的显示了出来，其中的功能4和功能5目前尚未实现，在将来的工作中，我们会在其中实现音频和视频的播放。值得注意的是对主菜单的操作可以脱离键盘，而完全由串口导入的数据来控制，将加速度计与手机物理上绑定之后便可以通过对手机的摇晃、旋转等动作完成对菜单的全部操作。

ItemListCanvas类:

ItemListCanvas同样继承了vCanvas类，实现了多条目文本浏览的功能，构造方法的输入依次为Display、Displayable、Font、Title和ItemVector。将多条String构造为一个Vector传递给ItemListCanvas后，该类能够提供一个多条目文本实现的用户界面。本系统的“关于本系统”子功能就是通过实例化该类得到的。

CloseCanvas类：

该类提供了系统的关机logo画面



图17 关机logo效果图

六、

结论

本系统采用Freescale的单片机MC9S08QG和加速度传感器MMA7260Q实现了便携式手持设备的手势控制，并采用了开放的J2ME平台实现了终端应用程序。系统完整，实用性强，成本低廉，在满足用户基本操作需要的同时，增加了移动设备操作的趣味性和灵活性。同时，加速度传感器MMA7260Q较小的体积和独特的节电模式更使其在移动设备的应用中有着无可比拟的优势。可以预见，在未来的手持设备系统中会大量采用类似的手势识别功能，因此具有广阔的市场应用前景。