第33卷 第20期 计 算 机 工 程 2007年10月
Vol.33 No.20 Computer Engineering October 2007
·开发研究与设计技术· 文章编号：1000—3428(2007)20—0279—04 文献标识码：A 中图分类号：TP391

基于嵌入式系统的中文输入法的设计与实现

胡明星，李双全，张 激
(华东计算技术研究所，上海 200233)

摘 要：中文输入是中文应用软件进行人机交互必不可少的部分，目前在嵌入式系统中广泛使用的图形系统都不支持中文输入。该文提出了一种在嵌入式图形系统中实现中文输入的方案，介绍了目前几种流行的嵌入式图形系统，阐述了中文输入法实现的基本原理和算法，给出了该中文输入法在不同嵌入式图形系统中的实现方法和示例。

关键词：中文输入法；嵌入式图形系统；QPE；NANOX

Design and Implementation of Chinese Input Method in Embedded System

HU Ming-xing, LI Shuang-quan, ZHANG Ji
(East China Research Institute of Computer Technology, Shanghai 200233)

【Abstract】Chinese input method is an important part of human-computer interaction in Chinese software applications. Currently, the Chinese input method is not supported by the widely used embedded GUI systems. This paper introduces several most popular embedded GUI systems and details about the principle and the core algorithm of the Chinese input method, and introduces how to implement the Chinese input method in different embedded GUI systems.

【Key words】 Chinese input method; embedded GUI system; QPE; NANOX

1 概述

随着嵌入式系统在家电、娱乐、通信等领域的应用不断发展，嵌入式系统越来越需要一个界面友好、支持中文的图形系统。目前在嵌入式系统中广泛使用的图形系统对中文显示已有很好的支持，但都不支持中文的输入，因此中文输入法在嵌入式图形系统中的实现，对于嵌入式相关产品的应用具有极强的现实意义[1]。

在嵌入式系统下实现中文输入主要涉及2个方面：汉字显示和中文输入。汉字显示由带中文字库且支持中文显示算法的嵌入式图形系统支持。中文输入与图形系统的关系如图1所示。

图1 中文处理模块与图形系统的关系

2 嵌入式图形系统

图形系统提供了一种良好的用户与应用程序之间的交互机制，用户可以用鼠标、键盘等输入设备对屏幕上显示的窗口、按钮等界面进行直接操作。实现一个图形系统通常涉及到几方面的问题：事件处理，窗口管理，对图形设备的快速操作，对输入设备的支持等。

下面介绍二款比较常用并对中文显示有着很好支持的图形库系统：Microwindows，Qt/Embedded，笔者已在其上实现了中文输入功能。

Microwindows是由Century Software开发的一个开源(LGPL)项目，主要为一些小型设备和平台提供图形窗口环境。它支持许多硬件平台，具有很强的移植性，可以运行在嵌入式Linux上，并且提供了基于Win32/nano-X的两套API接口。

Qt/Embedded是一个跨平台的C++图形系统，由Trolltech公司出品。Qt/Embedded对于各种硬件接口到GUI工具包提供了完整的图形包。Qt/Embedded是开源(LGPL)项目。它提供了丰富的窗口部件集，具有面向对象、易于扩展、组件编程的特点[2]。

3 中文输入模块的设计与实现

中文输入法的实质是建立一种按键组合到汉字编码的映射关系，它首先完成从汉字外码到汉字内码的转换，再根据内码计算出其对应汉字的字模地址，然后通过该地址从字库文件中读取相应汉字的字模信息，最后汉字显示模块调用直接描画函数将汉字输出到屏幕。

汉字外码指用户从键盘上键入汉字时所用的汉字编码，通常用户使用拼音或五笔等。汉字内码(也称汉字机内码)指计算机内部存储、处理加工汉字时所用的代码。对于西文字符的机内码采用ASCII码来表示。而每个汉字的内码用2个字节的二进制数表示，为了与ASCII相区别，最高位为1。汉字字库指存放在存储器中的常用汉字和符号的点阵信息的集合。每个汉字的点阵信息叫作该字的字形，也称字模。字模地址指汉字在汉字字库中的偏移地址。字模信息指汉字的

作者简介：胡明星(1978－)，女，助理工程师，主研方向：系统软件，嵌入式图形系统及应用；李双全，工程师；张 激，研究员
收稿日期：2006-10-25 **E-mail：**humingxing@ecict.com.cn

点阵信息。

因此实现中文输入法的核心是要完成两方面的工作：输入法字典的设计和输入算法的设计。从某种意义上讲，编码字典设计是关键，输入算法设计是保障。

3.1 输入法字典的结构

输入法字典文件一般分为头部信息，汉字编码信息，索引表信息，编码对应的汉字内容等。例如，全拼输入法字典的结构如 *hz_input_table* 所述。

```
typedef struct {
        char magic_number[sizeof(MAGIC_NUMBER)];
        char ename[CIN_ENAME_LENGTH];
        char cname[CIN_CNAME_LENGTH];
        char selkey[SELECT_KEY_LENGTH];
        char last_full;
        int TotalKey;
        int MaxPress;
        int MaxDupSel;
        int TotalChar;
        unsigned char KeyMap[128];
        unsigned char KeyName[64];
        unsigned short KeyIndex[64];
        int phrase_num;
        char phr[4];
        ITEM *item;
} hz_input_table;
```

这个数据结构是全拼输入法中最重要的一个信息表，其中，*magic_number* 为表头内容，如 ECIGB；*ename* 为输入法的英文名称如 pinyin；*cname* 为输入法的中文名称如全拼；*selkey* 存储用户选择汉字时输入的有效选择字符“1234567890”；*last_full* 表示半全角状态；*MaxPress* 表示一个汉字最多可以输入 10 个拼音序列；*MaxDupSel* 表示用户选择重码的汉字时，一页最多可以让用户选择的汉字个数；*TotalChar* 表示本输入法支持汉字和数组总数目；*KeyMap*[128]是将全拼输入法用户输入的字母序列(a～z)的 ASCII 码为下标，其对应的值为 1～26，其他的值为 0；*KeyName*[64]是 *keyMap*[128]的逆表，即 1～26 被转换成“a～z”；*KeyIndex*[64]是输入法内部查找汉字的首要参考表，它是存放 26 个字母位于输入法字典中开始的索引号；*phrase_num* 表示本输入法支持的词组的数目；*phr* 是词组数组的指针。*item* 指向一个 ITEM 结构类型的数组：

```
struct ITEM {
unsigned long key1；
unsigned long key2；
unsigned short ch；
unsigned short frequency；
}；
```

item 数组共有 *TotalChar* 个项，每个 item 对应一个汉字或者一个词组。*item*[*TotalChar*]数组是所有合法的字母排列对应的所有汉字或词组的集合。集合中的所有元素首先按照首字母的不同分为 26 个集合，并且按照首字母 ASCII 码从小到大的顺序从前到后排列这 26 个集合。在这 26 个集合中再按照第 2 个字母的不同又分为 26 个集合，并且按照第 2 个字母 ASCII 码从小到大的顺序从前到后排列，依此类推。如果一个合法的字母排列对应多个汉字或词组，就按照使用频率来排列，常用的排在前面，不常用的排在后面。ITEM 结构中的 *frequency* 就是汉字或词组的使用频率。

如果这个 *item* 是汉字，那么 *ch* 就是这个汉字的 GB2312 编码；如果这个 *item* 是词组，那么 *ch* 就是这个词组在 *phr* 数组中的索引。ITEM 结构中的 *key*1 和 *key*2 的值是根据输入字母序列计算出来的，反映了一个完整合法的字母序列和一个 *item* 的对应关系。

如前所述，首先将拼音字母的 ASCII 值 *key* 借助 *KeyMap* 表转换成输入法内部使用的值 *inkey*(*inkey=KeyMap*[*key*])存入数组 *InpKey*[*MaxPress*]中，不足 *MaxPress* 个的位置补 0，然后由这 10 个值计算 *key*1=*InpKey*[4]|(*InpKey*[3]<<6)|(*InpKey*[2]<<12)|(*InpKey*[1]<<18)| (*InpKey*[0]<<24) 和 *key*2 = *InpKey*[9] |(*InpKey*[8]<<6) | (*InpKey*[7]<<12) | (*InpKey*[6] <<18) | (*InpKey*[5]<<24)，这两个值 *key*1 和 *key*2 就是 ITEM 中的值。对任意不同的 10 个字母排列，*key*1 和 *key*2 都是不同的。

全拼输入过程就是一个遍历查找过程，首先根据用户输入字母计算出 *val*1 和 *val*2，找到满足((*val*1==*key*1)&&(*val*2==*key*2))的 *item*(可能有多个)，对找到的每一个 *item*，如果它的 *ch*>0xA1A1，那么这个 *item* 就对应一个汉字，*ch* 就是它的 GB2312 编码。否则是词组。但是如果采用这样简单的匹配查找，那么只有在用户输入全部的拼音字母序列之后才能找到匹配的汉字或词组。在用户输入的过程中，使用下面这样一个掩码表即可只输入部分字母就能找出可能的汉字或词组：

```
unsigned long mask[18]=
{0x00000000, 0x00000000, 0x00000000, 0x00000000, 0x00000000,
0x00000000, 0x3f000000, 0x3ffc0000, 0x3ffff000, 0x3fffffc0, 0x3fffffff,
0x3fffffff, 0x3fffffff, 0x3fffffff, 0x3fffffff, 0x3fffffff, 0x3fffffff, 0x3fffffff  }
```

输入法根据用户输入的字母序列计算出 *val*1 和 *val*2 查找匹配的 *item* 时，将改为((*val*1==(*key*1&*mask*[*InputCount*+5]))&&(*val*2==(*key*2&*mask*[*InputCount*]))),其中 *input_count* 是用户这时输入的字母个数。其实也就是在比较前将 ITEM 中的 *key*1 和 *key*2 值与上合适的掩码，屏蔽用户还没输入的后面的值，这样就找出了用户输入字母序列等于开始字母序列而不是全部字母序列的所有汉字或词组。前面提到的 *KeyIndex*[64]数组是为了提高查找汉字或词组的效率，通过第 1 个字母把搜索范围先缩小了 26 倍，然后在这范围内查找。

3.2 中文输入法的实现

以主要的全局变量、接口函数和查找算法为例，中文输入法的流程使用 UML 活动图(图 2)，描述如下：

unsigned short *seltab*[16]； 存放当前查找到的汉字的 GB2312 编码值。

int *cur_sel_num*； 存放当前查找到的匹配汉字的数目。

unsigned char *inp_key*[*MAX_PRESS*]； 记录当前用户输入的字母。

int *input_count*； 记录当前用户输入的字母个数。

int *char_index*[15]；*char_index* [*i*]记录和用户输入的前 *i* 个字母匹配的第 1 个 ITEM 的序号(*i* 从 1 开始)。

int *start_key* , *end_key*, *next_pg_index*, *curr_pg_index*；

start_key 记录和当前输入的字母序列匹配的第一个 ITEM 的序号，*end_key* 记录和当前输入的第一个字母匹配的最后一个 ITEM 的序号。*curr_pg_index* 记录当前输出汉字的 ITEM 序号，即给用户输出汉字的当前位置。*next_pg_index* 记录下一屏输出汉字的 ITEM 第 1 个序号。*next_pg_index*, *curr_pg_index* 都必须在 *start_key* 和

end_key 之间，记录上下翻页时输出汉字的序号。

void *load_imt*(void);

将打开输入法字典 *pinyin.tab* 读入内存空间，构造 *hz_input_table* 数据结构

void *find_match_key* (void)；

根据 *inp_key* [*MAX_PRESS*]中用户当前输入的字母序列查找匹配的第一个 ITEM 的序号，存入 *start_key* 和 *char_index*[*input_count*]中，把最后一个 ITEM 的序号存入 *end_key*，并把 *start_key* 置为 *curr_pg_index*。

void *fill_match_chars* (int *num*)；

从 *curr_pg_index* 位置开始向后查找和当前输入字母序列匹配的 ITEM，最多 *num* 个，将找到的 ITEM 结构中的 *ch* 即汉字编码填入 *seltab*[16]中，把实际显示候选字的数目记录在 *cur_sel_num*，把下一个要输出的 ITEM 的序号记录在 *next_pg_index* 中。

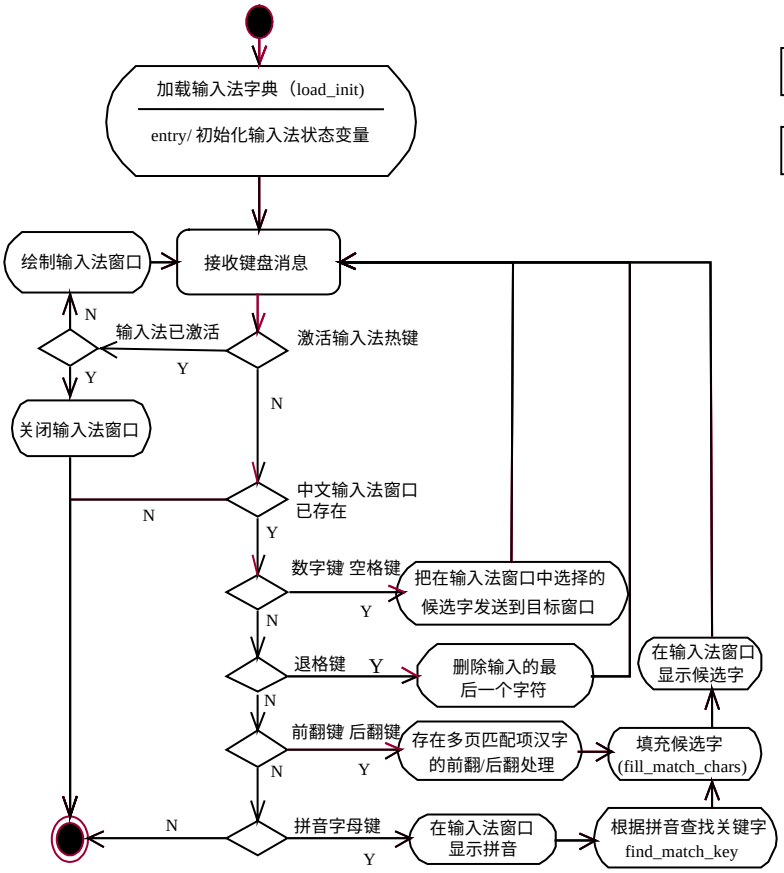

图 2　中文输入法的活动图

了解了上述主要的接口函数，下面说明一下汉字查找的算法：

根据全拼输入法字典 ITEM 的排列原理，先要查找关键字即根据输入的拼音计算出 *val*1 和 *val*2，然后与 ITEM 的 *key*1 和 *key*2 比较，比较之前根据用户输入的字母数量对 *key*1 和 *key*2 进行处理，屏蔽用户没输入的部分找出第 1 个匹配的 ITEM 序号 *start_key* 和最后一个 ITME 序号 *end_key*。这样在 *start_key* 和 *end_key* 之间的所有 ITEM 就是当前匹配的用户输入的汉字和词组。接着填充候选字即在刚才匹配的范围内，以当前序号位置为起点向前和向后拷贝 ITEM 的 *ch* 即汉字内码到 *seltab*[16]中。最后由汉字显示模块将汉字内码显示到输入法候选窗口上。

4　中文输入法在嵌入式图形系统的实现

根据不同图形系统的特点，笔者已利用封装技术将上述中文输入法在下面两种图形系统中实现。在嵌入式图形系统下实现中文输入的原理是：图形系统服务器进程接收到“键盘事件”后，首先将“键盘事件”送到中文服务器的键盘事件过滤器中进行“过滤”；然后由中文输入服务器的事件处理模块进行处理，根据核心算法输出中文字符的编码，同时在中文处理界面上(包括拼音输入区域、汉字候选区域)显示相应的信息；最后将转换后的中文编码发送到拥有焦点的窗口，并显示相应的汉字。

中文输入服务器的设计框图见图 3。

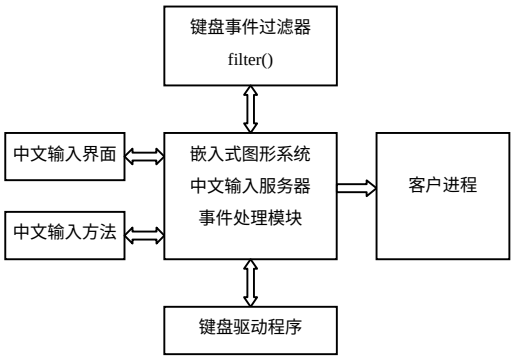

图 3　中文输入服务器的设计框图

4.1　中文输入在 QPE 上的实现

QPE是在Qt/Embedded库的基础上，专门针对PDA等嵌入式Linux的移动手持设备开发的开放源码的应用程序包和开发库[3]。但是QPE软件包没有提供中文输入，笔者利用QPE图形开发包提供的输入法编程接口和QPE所提供的插件技术，将上述中文输入法核心算法在QPE系统中实现，由于QPE图形系统采用C++编写，因此该中文输入法的实现也采用面向对象程序设计思想进行设计 实现。

由 Qt/Embedded 的键盘事件的分析可知，键盘过滤器是 QWSServer:: KeyboardFilter 类的一个虚函数 *filter*()。当 *filter*()返回 false，键盘事件直接发送到客户进程，它的发送由 QWSServer::sendKeyEvent()完成；当 *filter*() 返回 true，键盘事件由实现的中文输入的核心函数 *filter*()完成。键盘事件过滤器的安装过程：

```
SimpleIM    *ime = new SimpleIM(QString(getenv ("QPEDIR"))+
"/single/pinyin.tab");
QWSServer::setKeyboardFilter(ime);
```

键盘事件过滤器的安装以后，当产生键盘事件时会进入到以下函数：

```
bool SimpleIM::filter(int unicode, int keycode, int modifiers, bool
isPress, bool autoRepeat)
```

需要注意的是：由于输入法字典是由 GB2312 编码，而在 Qt/Embedded 库内部都采用 Unicode 编码，因此在将汉字发送到程序前必须转码。Qt/Embedded 的 QTextCode 类提供了 GB2312 向 Unicode 转换的功能。

实现后的具体示例见图 4。

图 4　中文输入在 QPE 上的实现

4.2　中文输入在 nano-X 上的实现

nano-X 图形系统没有像 Qt/Embedded 那样自带了事件过滤的接口，并且它采用了 Client/Server 的架构。所以 nano-X 上的输入法设计成：nano-X server 提供了对键盘和鼠标的处理，它接收键盘和鼠标的中断信息，并将该信息封装成事件，发送给 nano-X 的客户程序。当在中文状态时，nano-X 服务器截获到键盘的输入，并将键码转发到输入法进程。输入法进程负责加载拼音输入法字典，绘制输入法窗口，当截获到用户输入的拼音序列后，通过查输入法字典、列出候选字。用户可以通过空格键或者数字键选取待输入的字，并将该汉字的 GB2312 编码发送到当前的焦点窗口上。

该实现方法修改了 nano-X server 端程序，在 srvmain.c 中创建共享内存，存取 nano-X server 的进程 id 号，输入法窗口 id 号，输入法窗口状态信息。在 nano-X 的键盘事件响应函数 GsCheckKeyboardEvent 收到键盘中断时，先判断当前键码是否是自定义的输入法控制热键，来决定键盘事件是分发给输入法进程，还是其他客户端进程。输入法进程收到键盘事件后，再调用输入法过滤函数 *hz_filter* 来完成从外码到汉字内码的转换。

实现后的具体示例见图 5。

图 5　中文输入在 nano-X 上的实现

5　结论

本文介绍了一个简单实用的中文输入法实现方案。该中文输入法的核心算法除了支持中文的拼音外，还可以随意地进行扩展，可支持五笔、笔画等输入法，并可以进行汉字、英文字母、数字符号等文本信息的输入。还支持词组和联想的功能。目前，本文所描述的解决方案在 PDA、数字电视接收机顶盒、上海电信家家易等项目中已得到广泛应用。同时，在多种操作系统上得到了实现，包括嵌入式 Linux 和华东计算技术研究所自主研制的 ReWorks 实时嵌入式操作系统。

参考文献

1 王　卓, 包　杰. 嵌入式 Linux 系统及其应用前景[J]. 单片机与嵌入式系统应用, 2004, (5).

2 徐广毅, 张晓林, 崔迎炜, 等. 嵌入式 Linux 系统中 GUI 系统的研究与移植[J]. 单片机与嵌入式系统应用, 2004, (10).

3 常　江. 嵌入式系统中文输入法的设计[J]. 电子产品世界, 2004, (17).

(上接第 278 页)

的 RFC 开发 PROXY 代理服务。经测试系统具备了较好的性能，达到了设计要求。

(1)安全性测试。对涉密区、核心区做攻击测试，这两个区的服务器抗一切已知攻击，包括网管的跨段入侵尝试。

(2)速度测试。实验平台为 10Mb/s 以太网，在可管区向外网群发邮件，与常规方案对比，测试结果如表 1。

表 1　速度测试对比结果

环境	单个 2MB 邮件/s	并发 50 个 40KB 邮件/s
通过 GAP 与 PROXY	<7	<70
不通过 GAP 与 PROXY	<6	<40

比较单个邮件发送，两者差别不大。群发时两者有明显差异，这主要是二级代理导致的速度下降。这是以速度换取安全的必然结果。

4　结束语

系统利用 GAP 技术，将校园网进行安全区隔离，较好地解决了校园网多安全层次和分布性的安全问题。不过本文的方案对 1 000Mb/s 主干校园网络明显带宽不够，改进办法可以采用其他的高速通信模块，目前 PC104PLUS 模块国内最高带宽已达到 400Mb/s，国外已有 800Mb/s 以上带宽的模块面市，基本可以满足需要。

参考文献

1 Michael B. (Un)Bridging the Gap[J]. Information Security, 2000, 3(7): 35-47.

2 PC/104 Embedded Consortium. PC104 Specification (Version2.5) [EB/OL]. (2005-06-08). http://www.winsystems.com/specs/pc104 spec.pdf.

3 陈　睿, 田忠和. 物理隔离网闸数据交换技术的研究[J]. 计算机与数字工程, 2005 ,33(2): 47-49.

4 岳红梅, 石冬雪. 基于嵌入式 LINUX 的网络隔离系统研究与实现[J]. 计算机工程与应用, 2005, 41(5): 141-143.

5 邹思轶. 嵌入式 Linux 设计与应用[M]. 北京: 清华大学出版社, 2002.

6 Gilbert D. The Linux SCSI HOWTO[EB/OL]. (2005-07-13). http://linux.cis.nctu.edu.tw/docs/woven/HOWTO/SCSI-HOWTO.html.