I.J.Modern Education and Computer Science, **2011, 1, 16-23**
Published Online February 2011 in MECS (http://www.mecs-press.org/)

A Chinese Character Text Editor for Embedded System Education

Ye Junjie Yu Jianxin
Department of Computer Science and Technology
Nanjing University
Nanjing, China
ye020510625@126.com yujianxn@nju.edu.cn

Abstract—**A course project experiment plays a very important role in mastering a student embedded system software education. The paper gives the design and implementation solution to a Chinese character text editor for embedded system (CCTE for short), including requirement analysis, function design, task division, main process flow, Chinese character process, man-machine interface layout and test results. This course project can be used as a template for embedded system software experiment.**

Index Terms—embedded system; text editing; Chinese character processing; VxWorks; WindML; course project

I. FOREWORD

One important aspect of an embedded system (ES for short) education is the course project experiment. It requires the students to make a product prototype with some design concept in a lab, which is between verification experiment in school and industry project at scale and technology level [1]. But the current situation in China is that many college students lack enough knowledge about product development in the IT industry. This situation is especially worse for ES students. Even though they have studied many courses in the institute, it is still a puzzle for them to make a course project. If the teacher and the assistant of ES course have prepared several course project prototypes in advance, it is convenient for them to help students to quickly get familiar with ES software development and smoothly complete the assigned course projects. This kind of prototype examples should possess characteristic of wide applicability and adequate development scale.

Out of this consideration, we developed an ES Chinese character text editor (CCTE for short), which is a teaching course project used in our ES course teaching.

CCTE can be considered a course project experiment paradigm embedded in the software development curricula for master students. It is a text editor utility based on VxWorks operating system and various ES experiment board, capable of ASCII text editing and Chinese character text editing. In this paper, we introduce design and implementation of CCTE.

II. CONFIGURE DEVELOPMENT ENVIRONMENT

More often than not, the development of an embedded system is accomplished in a cross development environment, that is, the code is written and compiled on the host machine, while the executable image program will be running on the target machine. In our experiment, the host machine is a PC running Windows XP and the integrated development environment is Tornado 2.2 donated by Wind River Corporation. The operating system on the target machine is VxWorks. Because CCTE will have operations on files, we established the DOS file system in the flash memory. The interface between VxWorks and flash memory is TrueFFS (Tffs for short) block device, which is not a file system itself and thus entails the use of DOS file system.

In this experiment, the configuration of Tffs device and DOS file system is accomplished in the initializing code as below:

```
sysTffsFormat();          // formats the Tffs block device

usrTffsConfig(0,0, ROOT); // installs the DOS file system
```

In addition, the Chinese character library HZK16, in which the dot matrix information of common Chinese characters in GB2312 is stored, and the check list file of Chinese character with Pinyin code, also named Pinyin key tree, are used.

Further more; the user interface is programmed with the graphics library WindML from Wind River Co., so we need to compile the WindML library separately for individual target experiment board.

III. REQUIREMENT ANALYSIS

We analyzed some documents of a well-made text editor with graphic interface, which runs in the real ES environment [2][3]. After the analysis, we considered that an ES editor should have the following basic functions [4].

(1) It can create a new text file with graphic interface; (2) User can input uppercase and lowercase, ASCII symbols, numbers, and Chinese characters through a small keyboard with only 4*4 keys and a touch screen; (3) It can write edited text into a file stored on flash memory; (4) It can open a text

file in flash memory, and the characters in file can be well displayed on editing window in pages and lines sequence at LCD screen of ARM9 processor experiment board; (5) Page down and page up display function; (6) Find a specific string in text and highlight it. (7) By pressing keys on small keyboard, user can move focus cursor to operating location in text file. (8) It can insert or delete an ASCII character or a Chinese character at the cursor location.

IV. TASK DIVISION

The embedded text editor program is divided into 7 tasks. Detailed information of those tasks are given as follows:

tEntry task: This is the entry function of CCTE. It is also the main control function of the editor. The task firstly initializes global variables and arrays, secondly creates some semaphores and message queues. Then the task draws picture frame of Man-Machine Interface (MMI for short) on LCD, gets edited text file name, builds up some fonts for text drawing, sets up Pinyin lookup tree for Chinese character input. After that, it creates 6 tasks. Finally it enters an infinite loop.

In the infinite loop, it calls function task—input ( ) to do an input job. That function waits to receive a touch screen input or keyboard input. Once input is received, it sends corresponding message to related task.

tTextArea task: This task is responsible for text's content display. CCTE displays such contents as ASCII number, ASCII symbol, ASCII letters, and Chinese characters. At the beginning stage, it opens a text file from flash memory, reading all data and loading them into memory. Then it sets up index for the text pages and calculates total page number. After that, it displays the first page content in editing window of CCTE. It is always in a waiting state. If this task received messages from other tasks, including cursor moving, content inserting, content deleting, content appending, page up or page down displaying, file saving, etc., it processes them at once according to concrete content of received message.

tNumArea task: Main function of this task is to respond to message trigged by number input buttons. It extracts ASCII code from the message which is sent from tEntry, and sends them to tTextArea task, which will insert the ASCII code into cursor place and display the text page including that code.

tEngArea task: This task responds to message trigged by English input buttons. It processes message sent from tEntry, and extracts ASCII letter code. Then this task sends ASCII code to tTextArea task, which will insert and display the code.

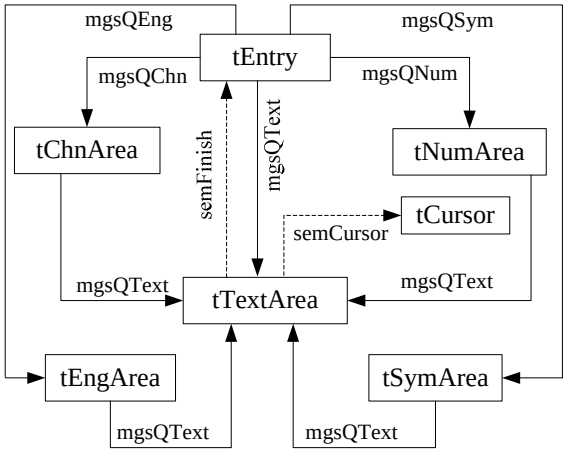

Figure 1. IPC flow chart for the main 7 tasks of CCTE

tChnArea task: It responds to message trigged by Chinese character input buttons. It processes Chinese phonetic code messages sent from tEntry, completing Chinese character letter code search and input through phonetic retrieval function. Then this task sends Chinese character GB2312 code to tTextArea task, which will insert and display that Chinese character at current cursor place.

tSymArea task: It responds to messages trigged by symbol input buttons. It processes key scan code message sent from tEntry and obtains ASCII symbol code within symbol conversion table. Then it sends ASCII symbol code to tTextArea, so as to insert and display the symbol code.

Fig. 1 shows the inner task communication (IPC for short) among the main 6 tasks. In this program, the communication is mostly implemented by message queues. It also involves a semaphore so as to realize the synchronization between two tasks. In the Fig. 1, the semaphore is marked by dashed line, while the message queues are marked by real lines.

V. DATA STRUCTURE

The main data structure in CCTE is character array FTEXT, which is used to save the total text data read from .txt file from flash memory. User's any operation on the text, such as appending, deleting or inserting will change the date elements in this array. The declaration statement of FTEXT array is as follows:

```
char FTEXT[262144]={0}; /* total space 256KB */
```

Another main structure is an integer array PINDEX. It is used to record the first character byte location of a page in array FTEXT, and the location is expressed by subscript number in FTEXT array. The declaration statement of PINDEX array is as follows:

```
long PINDEX[163];    /* Max. page number 163 */
```

Before CCTE begins to display the content of a test file, the data in the buffer FTEXT requires preprocessing. Firstly, it needs to calculate the number of pages concerning the whole text file. And store this number as the global variable

TOTAL_PAGE. Secondly, it needs to establish an index for each page. CCTE use array to represent the index, and PINDEX [N] records the offset of the first byte in page N.

In Fig. 2, three main data structure variables are illustrated by UML class diagram. They are char_msg, tree_node, and find—list.

The data type char—msg is used in message queue among tasks by send—msg( ) and recv—msg ( ) functions to deliver current input information which is from keyboard or touch screen, or to deliver what will be inserted in text. Character codes to be inserted into text are transmitted by this kind of message too. In data structure char—msg, the start field is used to record a type of message; the value field is used to record ASCII character code when ASCII character is transmitted. Once transmitting Chinese characters, both value and word fields are used.

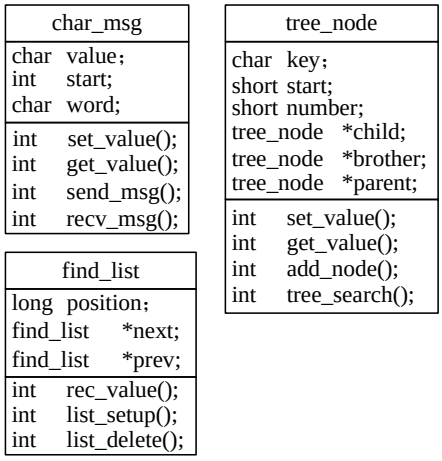

Figure 2. UML class diagram

Data structure variable tree—node is used to express nodes that make up Chinese characters in a Pinyin lookup tree. Any Pinyin letter in a valid Pinyin sequence string corresponds to a unique node of the lookup tree, and this Pinyin letter is stored in key field of the node. In order to build that tree, function add—node ( ) is called. When user inputs a Chinese character by Pinyin method, function tree—search ( ) will be called to search the tree and obtain several Chinese characters. In this case, start field records GB2312 code of the first Chinese character corresponding to that Pinyin sequence, while number field records the number of Chinese characters corresponding to that Pinyin sequence.

Data structure variable find—list used to create a double linked list in which the node records the searched string position. In search state, when a string pattern is found within text, then the searched string position in text is recorded in a newly added node of double linked list. This store operation is done for all matched position for each search action. Once work state returns to edit state from search state, it is necessary to delete the double linked list by calling list—delete ( ) function.

VI. MAIN PROCESS FLOW

Here we introduce pseudo-code and process flow chart of CCTE entry function. The pseudo-code is listed as follows.

```
void Entry ( ){
Initialize global variable.
Initialize WindML.
Create font.
Set up key tree for Chinese character Pinyin input method.
Input text file name for editing.
Draw MMI picture frame.
While(1)
{
  Create message queue and semaphore.
  Spawn 6 tasks.
  Wait for input until reset editor.
  Delete all message queues, semaphores and 6 tasks.
  }
}
```

The most important function of CCTE is TextArea( ). That task is responsible for such operations as text display, page down or page up, insert, append, find, etc. All the other tasks will send messages to this function by message queue, besides, the function also needs to use semaphore for implementing synchronization and mutual exclusion with other tasks while accessing the public resources. Main operation flow chart of task function TextArea( ) is illustrated by Fig. 3.

The Fig.3 gives a brief flow chart of tTextArea task, and its implementation process is more complicated. At each time a user adds one character to the edited text; CCTE should first recalculate the number of pages about the whole text file and reestablish the index for each page. Then determine which lines need to be redrawn. After that, CCTE clears these lines and redraw the lines with modified data. If the character added at the end of one page, and then the CCTE's display will show the content of next page.

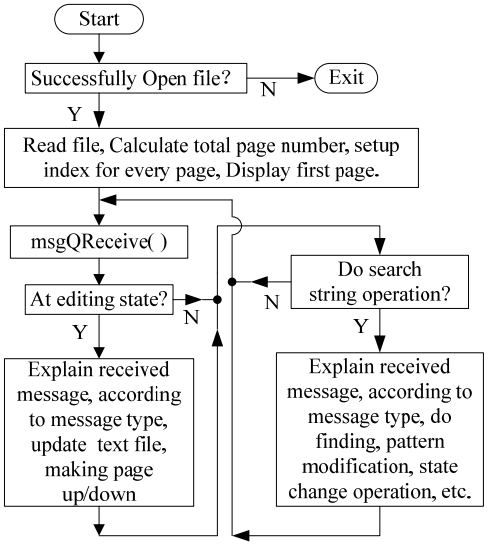

Figure 3. tTextArea task flow chart

In Fig.3, if search string operation is executed, tTextArea will call rec—value ( ) to create a double linked list. If a matched string is found, rec—value ( ) will add a new node to double linked list, and write the match positions in the text into that node. This kind of adding node and writing position action will do many times until search operation is to go to the end of the edited text. The pseudo-code of function rec—value ( ) is listed as follows.

```
rec—value ( ){
  Initialize the double linked list.
  for(i=0; FILE_TEXT[i]!=0xFF; i++)
  {  /* 0xFF is defined as a end character of the text */
    if( at byte i, one matched string is found  )
    { Add a new node & write the position i to the node. }
  }
}
```

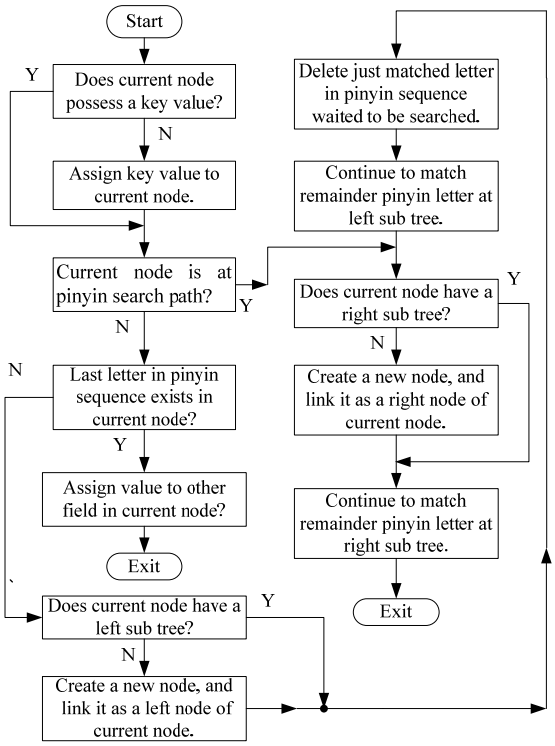

Figure 4. add_node function flow chart

During CCTE initialization stage, function Entry( ) calls function add—node( ) to build a Chinese character Pinyin lookup tree. Function add—node ( ) use recursive-descent method to do this. The add—node ( ) needs to read data from the file named input.txt which is stored in the flash memory. After the top add—node function return, the Pinyin lookup tree has been completed. The flow chart of recursive function add—node ( ) is shown in Fig. 4.

After using add—node ( ) routine to establish Pinyin lookup tree, we query this tree by routine tree—search( ). This routine is also using recursive functions, just like the routine add—node ( ). When realize the input method of Pinyin, we need to call this routine. The flow chart of recursive function tree—search ( ) is shown in Fig. 5

As shown on the Fig.5, because not all of the letter sequences correspond to Chinese character, if the return value of the top function tree—search ( ) is 0, that means the phonetic sequence can find the corresponding Chinese characters. On the other hand, if the return value is 1, it means the phonetic sequence can not find any corresponding Chinese characters.

To display text on the LCD screen, the function display—one( ) is invoked, whose parameters include the position of read pointer of the target character in the text file, the color and size of the text and so on. Besides, because a Chinese character is coded in two bytes while an English character is coded in one byte, a return value is needed to inform the caller whether the displayed character is Chinese or English. The flow chart of this function is shown in the Fig.6.

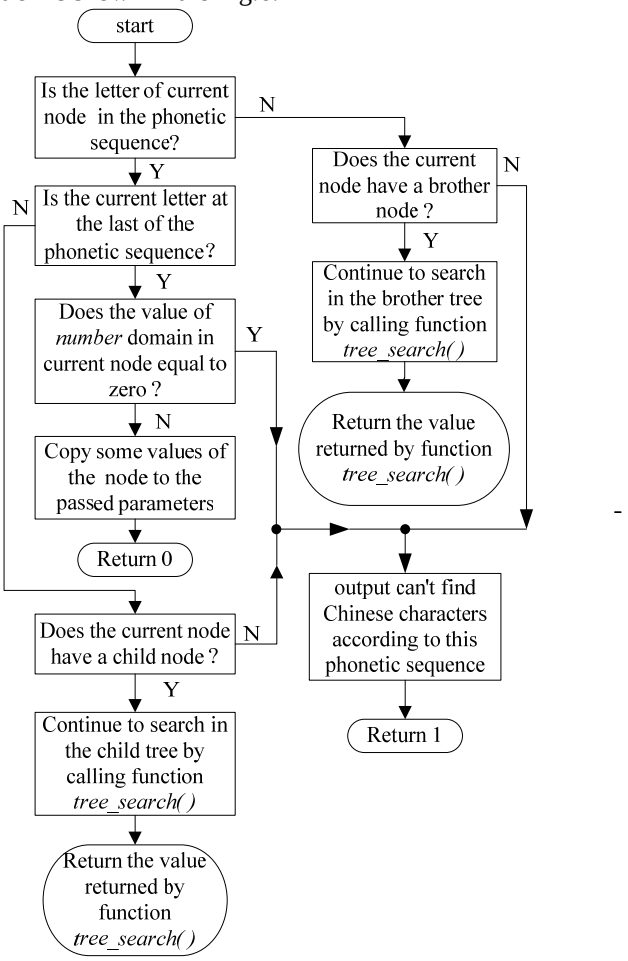

Figure 5. tree_search function flow chart

Function HZK_out( ) gets the Chinese character dot matrix information from HZK16 file. This function can draw Chinese character on the screen based on input parameters, which represent Chinese character code, size, color, coordinate of

drawing this character. We use pseudo-code to describe process flow of HZK_out function as follows.

```
HZK_out( )
{
    Declare some local variables and give primary value.
    open file HZK16
    Find out the displacement of the current Chinese character in the HZK16 file.
    for(i=0;i<16;i++)
    {
      for(j=0;j<2;j++)
      {
       Read in one byte information of dot matrix.
       Draw points on the screen with that byte info.
      }
    }
    Close the file.
}
```

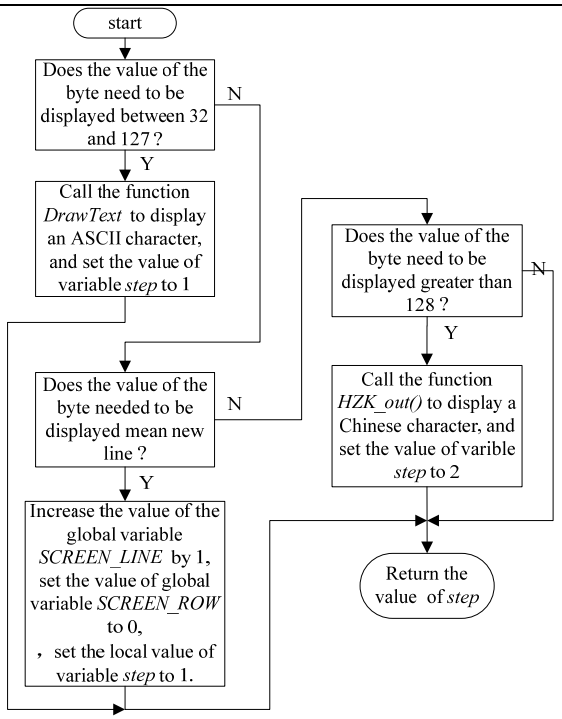

Figure 6. display_one function flow chart

VII. CHINESE CHARACTER PROCESSING

In this course project, Chinese character matrix library HZK16 is used for Chinese character display within the edited text. In HZK16 library, every simplified Chinese character code is represented in two bytes and accordant dot array matrix has 16 rows and 16 columns. HZK16 library is national simplified Chinese code standard published at 1980, supporting 6763 Chinese characters and 682 symbols, in which level one has 3755 Chinese characters sorted by phonetic sequence, level two has 3008 Chinese characters sorted by simplified radicals[5]. Each Chinese character in HZK16 library, which is of size 16 * 16, needs 256 dots to be displayed. That is to say, 32 bytes are needed to display a common Chinese character. It is known that a GB2312 Chinese character is coded in 2 bytes, ranging from 0xA1A1 to 0xFEFE. Here, A1-A9 is the range of punctuations and B0-F7 is the range of Chinese characters, each range consisting of 94 characters.

Apart from the display, the editor program provides a Pinyin input method. For this reason, we construct a key tree[6]. Every node in key tree corresponds to a key value. Nodes path that is from root node to any node along the key tree constitutes a Pinyin code of Chinese character. If a node possesses the last letter of one Pinyin sequence, the first Chinese character's GB2312 code of that Pinyin sequence in HZK 16 is saved in that node. Additionally, the number of this kind of Chinese characters is also saved in that node. After a Pinyin key tree has been built up, the first GB2312 code of a group Chinese character which has the same input Pinyin sequence can be retrieved by recursive-descent searching routine. The number of Chinese characters to this Pinyin sequence can be retrieved as well.

On common PC, with a plenty of system resources, input methods is implemented with the main focus on full performance and convenient operation. In embedded system, however, because of the strictly limited system resource, the main efforts will be spent on the efficiency of the algorithm and storage space. The input methods applied in the embedded system should satisfy such requirements as structure compact, easy to port and with small size of memory. Besides, the input methods should need low computation, because the limited clock frequency of embedded processors. Association of input is desirable which can speed up user input, however, this will swell the codes of input method which makes it undesirable in embedded system, so it all depends on user's demand.

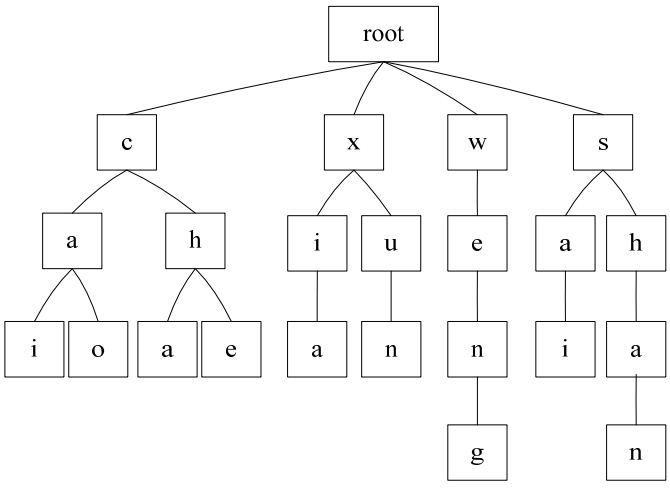

Figure 7. Pinyin keyword lookup path in the Pinyin tree

If CCTE uses a static array to store the Pinyin sequence, it will take up more memory resources. In this experimental project, we use a keyword tree to store the check list of Chinese character versus its Pinyin keyword sequence, to save memory. We consider this data structure as an appropriate storage solution, as in this structure if Pinyin sequences which have the same prefix, keyword

tree just store the same prefix only one time. Keyword tree is a special kind of search tree, in which each node has a character composing the keyword. This tree is different from a key tree in the conventional sense. Corresponding to a Pinyin keyword, it has a path from the root node to a leaf node. Please see Fig 7. Moreover, in order to save the memory resource, Chinese characters codes are not stored in the nodes of the tree.

Here we introduce keyword tree in detail. Consider a Pinyin keyword set{ cai, cao, cha, xia, xun, weng, sai, shan, chen} which can be partitioned by first letters as { (cai, cao, cha, chen), (xia, xun), (weng)，(sai, shan) }. Any set whose number of keywords exceeds one can be further partitioned by second letters as {{cai, cao), (cha, chen) } and so on. Obviously the sets generated in this way can easily compose an ordered tree, in which the characters contained in nodes of the same level are ordered from left to right. In CCTE, this ordered tree is named as Pinyin lookup tree, which is shown in the Fig.7.

A text file input.txt is used to implement the Pinyin input method. Information about Pinyin keyword sequence is stored in this file, and let's take one line as an example: "bai 白 8", the string "bai" stands for a Pinyin keyword sequence, and subsequent Chinese character "白" implies the first Chinese character which this Pinyin keyword sequence corresponds to, while the number "8" means this Pinyin sequence corresponds to eight Chinese characters. This input.txt file consists of many such lines.

VIII. INPUT TEXT

There are two different input methods in the CCTE; they are the touch screen and the key board. We use the keyboard to achieve the input and deleting of numbers, letters, symbols and Chinese characters. We use the touch screen to locate the position of cursor and select input method.

Function task—input( ) is responsible for receiving inputs from touch screen and keyboard. The flow chart of this function is shown in the Fig.8.

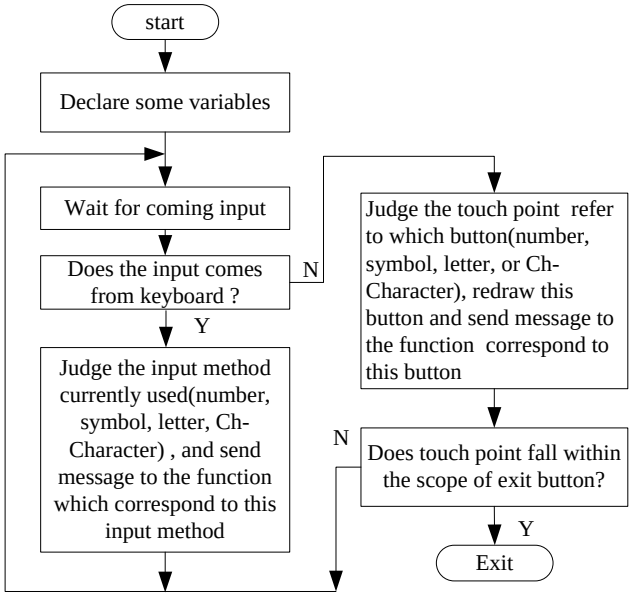

Figure 8. task_input function flow chart

We use components of WindML library to implement MMI for CCTE. The embedded experiment board has a 640*480 TFT type LCD. WindML supports this kind of LCD screen and touch screen[7]. The visual MMI interaction picture layout we designed has a convenient GUI for text operation. Fig. 9 gives the layout of MMI picture. And This MMI is based on WIMP（window, icon, menu, pointing device） whose interface is easy to introduce to novice users.

In this program, the Chinese character input is achieved by a Pinyin input method. And the Pinyin input method is divided into two stages.

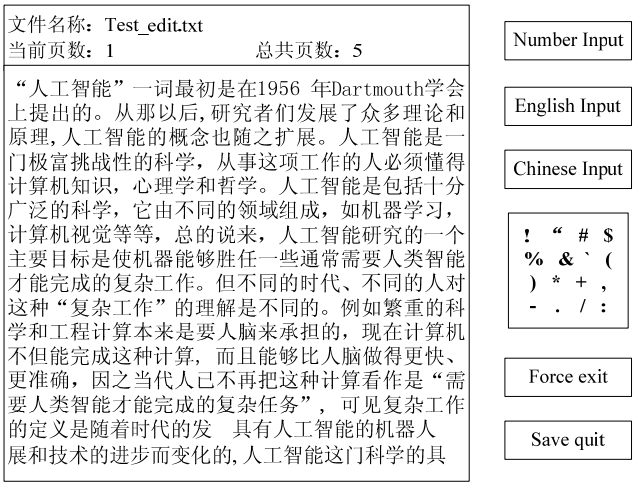

Figure 9. GUI picture of CCTE

The first stage is getting the input of phonetic alphabet. We use the keyboard on the experiment board to finish the input, however, this keyboard just owns 16 different keys and there are 26 different alphabets. Fig.10 illustrates our designed keyboard layout.

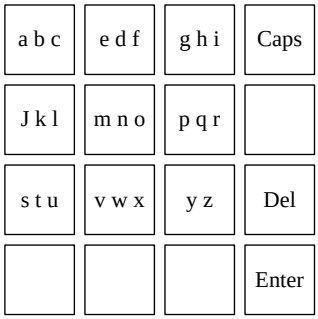

Figure 10. Pinyin input key arrangment in 4*4 key keyboard

After completing the input of phonetic alphabets, we enter the second stage.

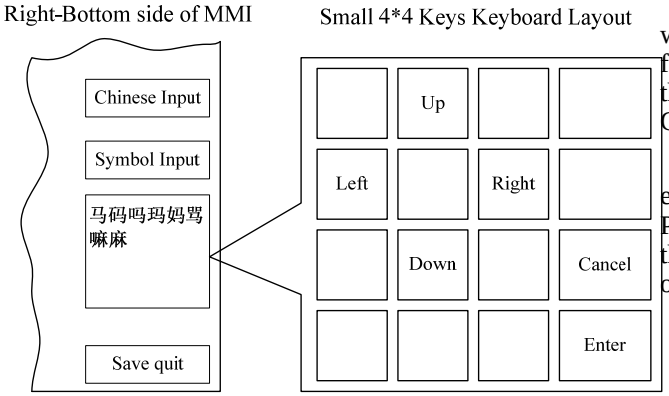

Figure 11. Pinyin search key arrangment in 4*4 key keyboard

Because a phonetic sequence may correspond to several (maybe none) Chinese characters, at this stage user should choose the Chinese character wanted within the displayed Chinese character set on the right-bottom side of the LCD screen. Again, we use the keyboard on the experiment board to do this operation and Fig.11 illustrates our designed keyboard layout.

IX. Test Results

(1) Every input method is tested and confirmed normal. (2) In text display state, operations such as page down display or page up display, cursor moving, character deleting, etc, all run normally. (3) Tests of cursor moving and locating upon touch screen are successful. (4) String pattern searching function runs well, all positions of test sample string in the tested text file can be searched out quickly. (5) When save button is pushed, edited text can be well saved into flash memory as a .txt file. After saving, if lab assistant or students opens this .txt file once again, the latest changed content can be normally read and shown. (6) In running state, if exit button of MMI is pushed, CCTE will soon stop running. At this occasion, the edited text will be saved, with latest changed content being abandoned.

X. Future Work

CCTE can be further improved, say, to select some successive words or text lines by highlight display method, cut and copy selected words or text lines operation, to save the file with another file name and so on. Functions like changing the word display color, size and font and so on, can also be added. Of course more font libraries should be added to support different Chinese character font shape display.

Pinyin input method can also be further improved, for instance, to add the association word input function. Once you input a Chinese character, a set of possible sequent Chinese character will be displayed in CCTE picture for user to choose. The first Chinese character and successive Chinese character will compose a frequently used Chinese word.

To achieve such a function, a table called association word code table is needed. In that table, the position of the first Chinese character chosen in the font library and that the subsequent Chinese characters with which can make a Chinese word are stored.

Moreover, Pinyin input self-learning function is extendible. That means, after using a period of time, the Pinyin input method will list the Chinese characters with the higher input frequency in the beginning part of optional window. See the left middle side of Fig. 11.

XI. Conclusion

A text editing tool is a basic software tool in handheld device, such as mobile phone or PDA, etc. CCTE course project possesses such functions within ES graphics interface as ASCII character input, Chinese character input, text edit, text display, etc. In our master course of embedded system software development, this course project can be used as a code paradigm in text editing tool education. The editor is programmed by ANCI C in Tornado IDE. Its total statements line number is about 2800. Education emphases of this course project are process oriented programming, multitask division, multitask scheduling by semaphores and messages, and Chinese character process. We can teach the editor's inner structure and process in class room. Apart from this, in lab, based on this sample editor we are able to make students write an ES text editing tool from the beginning, or expand functions at current version, so as to train their independent ES software development ability.

References

[1] Wilson P. Paula Filho, "Process Issues in Course Projects," St. Louis, Missouri, USA, pp. 629-630, May 2005.

[2] Andrew J. Ko, Htet Htet Aung, and Brad A. Myers, "Design Requirements for More Flexible Structured Editors from a Study of Programmers' Text Editing,", ACM New York, NY, USA, pp. 1557 – 1560, April 2005.

[3] Sharma, D. K. and Gruchacz, A. M., "The Display Text Editor TED: A Case Study in the Design and Implementation of Display-Oriented Interactive Human Interface," IEEE Trans. Comm. COM-30, Jan.1982.

[4] Teresa L. Roberts, Thomas P. Moran, "Evaluation of text editors,**"** ACM New York, NY, USA, pp. 136–141, March 1982.

GUO Hua, XU Long fei, and ZHANG Zhong, "Study on simplified and traditinoal Chinese processing and character libray

expansion technique in embedded system," computer Engineering and Design, vol.27, No. 3, Feb. 2006. (in Chinese)

[5] Li Fangjun, Jin Weidong, Xun Yonghong, and Liu Yong, "Realization the phonetic input method on embedded system," Education and Science, issue 2, June. 2006. (in Chinese)

[6] Wind River Co., VxWorks Programmer's Guide, Tornado Online Manuals, 2003.

Ye Junjie, male, born in 1987. Graduate Student in Computer Department of Nanjing University. His main research interests include Embedded System.

Yu Jianxin, male, 1953, bachelor's degree, senior engineer of CS department of Nanjing University. Recent education and research activity: Embedded System.

