Panasonic
ideas for life

SS Design

Mac Hanin Software
HaninLib API Specification

Version: 0.05

Date: 2008/2/4

TM	PM	Prepared
Youko Ohiwa	Wu Jia-Kwan	Wu Jia-Kwan

Panasonic Taiwan Laboratories Co., Ltd. (PTL)

[Change History]

Date	Ver.	Category	Section	Changes	Changed by
2007/12/14	0.01	-	-	First edition	Wu Jia-Kwan
2007/12/18	0.02	U	1.1	modify the description and add a structure diagram to clarify the basic concept	Wu Jia-Kwan
		U	1.2	remove the terms not be referred in this document	
		U	3	modify the block diagram to clarify the relationship between UI and API	
2008/01/07	0.03	U	2.3	added synopsis of data structure used in HaninLib	Wu Jia-Kwan
		U	2.4	added detailed description of HaninLib global variables	
2008/01/22	0.04	U	2.3	Table 7, modified the size of ulenmark from 4 to 1 in structure CORE	Wu Jia-Kwan
2008/02/04	0.05	D		remove ISM stamp on cover page	Wu Jia-Kwan

Category [A: Added, U: Updated, D: Deleted]

[Table of Contents]

1. Introduction

1.1 Outline

Hanin input system is a popular Chinese input method both on Windows and Mac OS. Hanin consists of three portions: Hanin library (thereafter, HaninLib), Hanin dictionary (thereafter, HaninDic) and user interface (UI). Figure 1 depicts overview of Hanin input system.

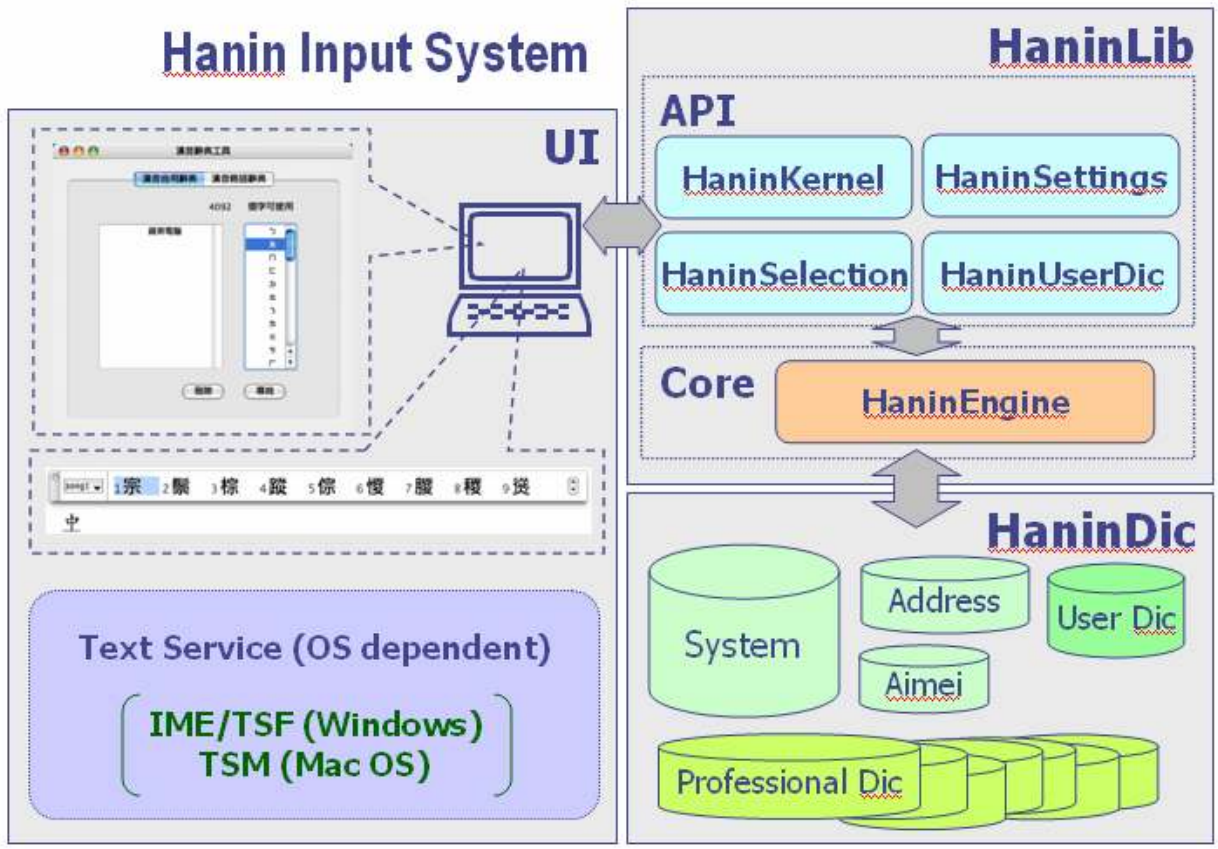


Figure 1: Overview of Hanin input system

The functional features of Hanin input system are as follows.

- Input mode (keyboard layout): Zhuyin, MPS II, Pinyin
  (Zhuyin keyboard layout: Standard, E-Ten, IBM, MiTAC)
- Conversion mode: converted by character, continuously converted by word
- Conversion algorithms: Long word preemption（長詞優先）
  Word with high usage frequency preemption（高頻詞優先）
  Preceding word preemption（前詞優先）
  Continual word preemption（連續變換優先）
  No conversion for Conjunction or Preposition character（連接詞不變換）
- Length of input buffer: 20 characters (2-byte character)

- System character dictionary: 17,005 characters (encoding: BIG5E)
  13,053 characters (encoding: BIG5)
- 60,000 words
- User defined word: maximum 4,096 characters
- Abbreviations: maximum 256 entries
- Learning dictionary: 100 character/word
- Professional dictionary: 14 categories, total 121,300 words
- Address dictionary provided
- Configuration settings of input status
- Fuzzy conversion function (called Aimei function)
- Symbol input function

This document mainly introduces the specification of Application Programming Interface (API) for HaninLib. The Hanin API has been constructed to allow developers to develop a traditional Chinese input method application/editor by means of utilizing Hanin core engine (HaninEngine) and HaninDic on popular OS such as Mac OS, Windows, and Linux.

HaninAPI consists of 4 categories of functions such as HaninKernel, HaninSelection, HaninUserDic and HaninSettings. HaninKernel functions allocate sufficient memory of working area to start up Hanin application, deal with users input keys to conduct the Chinese character or word conversion, and report the conversion result. The function to register user defined word is also included in HaninKernel. HaninSelection provides functions to get/select the same pronounced characters/words and a group of functions to handle the aimei conversion to choose character/word pronounced similar to user input. HaninSeletion also provides an abbreviation function to get the short/long form text regarding base text. HaninUserDic functions deal with the maintenance of user dictionary which includes user defined words and abbreviation text. HaninUseDic allows users to search/delete user defined word, and add/delete/modify abbreviations. HaninSettings provides a series of functions to allow user to set the current input status such as input mode, display mode, state of address/input.

1.2 Definition of Terms

[1]	MPS II	Mandarin Phonetic Symbols II （國語注音符號第二式）
[2]	Big5E	Big5 Extension Character Set
[3]	Zhuyin Fuhao	A phonetic system for transcribing Chinese for people learning to read, write or speak Mandarin（注音符號）
[4]	Aimei function	Fuzzy phonetic feature, slightly incorrect pronunciations similar to user's input（曖昧功能）

1.3 References

The following documents are inputs of this specification. When changes are made to the following documents, these changes will impact this specification.

[1] The Conversion Algorithm of Hanin Input Method (V3.x), Chen Chi-Zhang, Chang

Chia-Hwa, MITT, Jun 29, 1993
[2] Hanin System Specification, Lin Chi-Hsuan, MITT, May 3, 1997
[3] Hanin5 Library Reference , Liu Guei-Ji, PTL, Nov 1, 2002
[4] http://www.yale.edu/chinesemac/pages/hanin_5.html
[5] http://www.yale.edu/chinesemac/pages/tcim_x3.html

2. HaninLib API

2.1 Description of Function Block

HaninLib API consists of 4 groups of functions such as HaninKernel, HaninSelection, HaninUserDic, and HaninSettings. The function groups of HaninLib API are depicted in Figure 2.

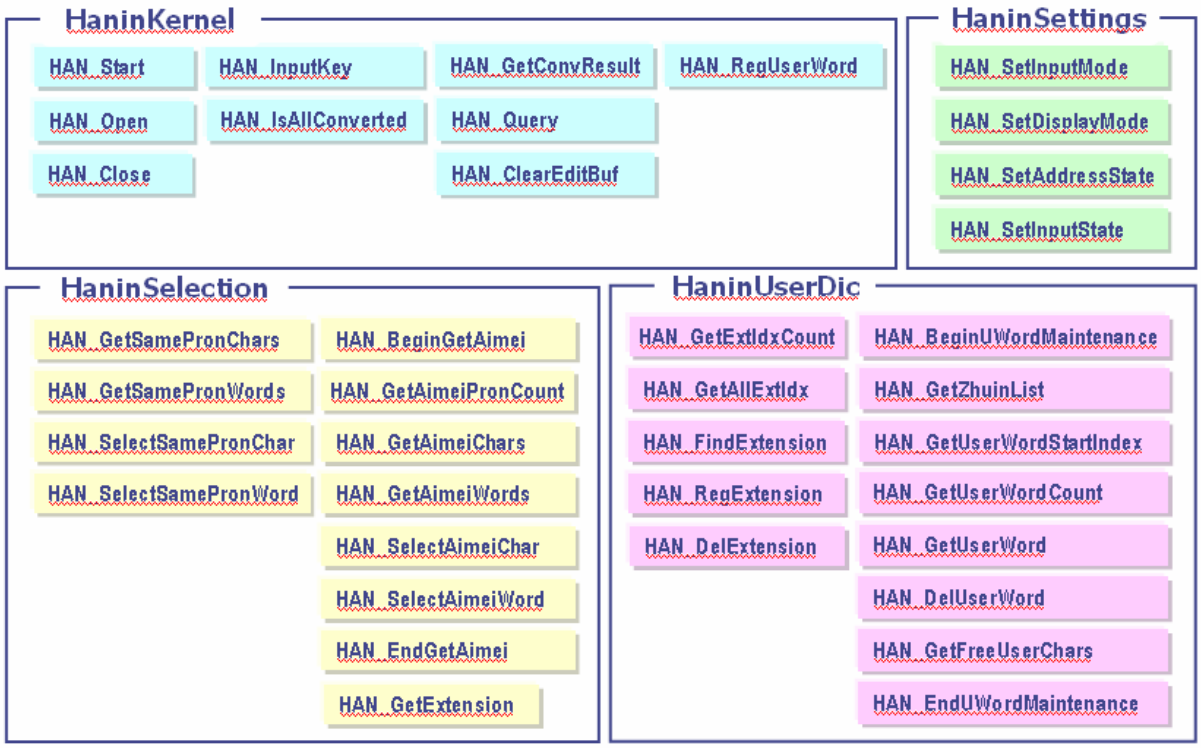


Figure 2: Function Groups of HaninLib APIs

HaninKernel starts the Chinese character conversion process by allocating adequate memory to store the intermediate conversion result. HaninKernel provides APIs to initialize the settings of current input conditions and operation when inputs a pressed key symbol and reports the up-to-date conversion result. When ends of Hanin input method, HaninKernel releases the allocated memory area.

HaninSelection provides functions to convert a series of input key-pressed symbols into Chinese character. The functions to get all the same pronounced Chinese characters and words are included in HaninSelection. Moreover, it has Aimei functions to allow fuzzy conversion for an incorrect pronounce (zhuyin) of Chinese character/word. HaninSelection

has abbreviation functions to deal with the maintenance of abbreviation extension.

HaninSettings consists of setting functions to set the Hanin input profile at the beginning or/and to change the status of Hanin input profile during HaninLib is active. HaninUserDic provides a suit of functions to maintain user-defined words.

2.2 Predifined Identifiers

HaninLib provides predefined identifiers for configuration setting of Hanin input method and the limitation of string length. These predefined identifiers are shown in Table 1.

Table 1: HaninLib predefined identifiers

Category	Identifier	Default	Note
Scan code	DEL	0x53	delete a 2-byte character on the right of line cursor in edit window, no operation during converting input key to zhuyin
	HOME	0x47	move cursor to the top of edit window, no operation during converting input key to zhuyin
	END	0x4F	move cursor to the bottom of edit window, no operation during converting input key to zhuyin
	RIGHT	0x4D	move cursor right a 2-byte character in edit window, no operation during converting input key to zhuyin
	LEFT	0x4B	move cursor left a 2-byte character in edit window, no operation during converting input key to zhuyin
	UP	0x48	change converted 2-byte character into zhuyin, no operation during converting input key to zuyin
	ESC	0x01	cancel the latest word conversion in edit window, no operation during converting input key to zhuyin
	BCKSP	0x0E	delete a 2-byte character or converting zhuyin in front of line cursor in edit window
	RET	0x1C	send back the converted 2-byte characters in edit window
	TAB	0x0F	extend short form of abbreviation into long form

Return value	RC_SUCCESS	0	return value for HAN_RegUserWord() and HAN_RegExtension()
	RC_OTHERERROR	-1	
	RC_WORDEXIST	-2	the registering user defined word already existed
	RC_USERWORDFULL	-3	no room to register user defined word in user dictionary
	RC_INVALIDWORD	-4	the registering user defined word is invalid
	RC_INUWMAINTENANCE	-5	current state is in maintenance operation of user dictionary
	RC_ERRORUSERWORDLEN	-6	the word length of user defined word is incorrect
	RC_INVALIDEXTID	-2	return value for HAN_RegExtension()
	RC_NOLONGEXT	-3	the abbreviation has no long form text
	RC_ALREADYEXIST	-4	the abbreviation already existed
	RC_EXTENSIONFULL	-5	no room to register abbreviation into user dictionary
	RC_NOTFOUND	-6	can't find indicated abbreviation in user dictionary
Input mode	STDKBD	1	standard zhuyin keyboard layout
	IBMKBD	2	IBM zhuyin keyboard layout
	ETKBD	3	Eten zhuyin keyboard layout
	MITECKBD	4	MiTAC zhuyin keyboard layout
	JUIN2	5	MPS II pinyin
	ROMAN	6	Roman pinyin
Input State	IS_CHINESE	1	Chinese conversion
	IS_PASS	2	bypass alphabet
	IS_FULLABC	3	change alphabet into 2-byte character
Display mode	DISPLAY_ZHUIN	0	Zhuyin display in input buffer
	DISPLAY_ALPHA	1	alphabet display in input buffer
Converted status	ATTRIB_INPUT	1	converting characters in input buffer
	ATTRIB_CONVERTED	2	converted in input buffer
Restriction	MAX_HDIC_PATH	256	maximum length of dictionary path
	MAXPROFDICOPENED	3	maximum number of available professional dictionaries
	MAX_CHAR_INPUTBUF	96	maximum number of characters in input buffer
	MAX_CHAR_RESULTBUF	82	maximum number of characters in converted result buffer

	MAX_CHAR_LONGFORM	42	maximum number of characters in long form of abbreviation
	MAX_CHAR_SHORTFORM	26	maximum number of characters in short form of abbreviation
	MAX_CHAR_UDICZHUYIN	42	maximum length of zhuyin buffer for user defined word
	MAX_CHAR_UDICWORD	12	maximum length of word buffer for user defined word

2.3 Data Structure

This section describes the data structures for Chinese character conversion in HaninLib. The data structures are shown in Table 2 ~ Table 11.

Table 2: Structure of HANININIT

Member	Type	Description	Size (byte)
hanin_dic	CHAR	path of Hanin system dictionary	256
aimei_dic	CHAR	path of Aimei dictionary	256
add_dic	CHAR	path of address dictionary	256
user_dic	CHAR	path of user dictionary	256
prof_dic	CHAR	path of 3 professional dictionaries	768
AddressState	CHAR	flag of condition of using address dictionary	1
InputMode	CHAR	flag of input zhuyin/pinyin mode	1
DisplayMode	CHAR	flag to indicate display language	1
Total			1,795

[Synopsis of HANININIT]

The structure HANININIT is for initializing the HaninLib. HaninLib needs full path and filename of Hanin dictionaries (includes system dictionaries, professional dictionaries, user dictionaries) and preference setting for Chinese input to convert input keys into Chinese characters.

Table 3: Structure of INPUTKEY

Member	Type	Description	Size (byte)
inkey	UCHAR	ASCII code of input key	1
scancode	UCHAR	scan code of input key	1
KB_Caps	CHAR	condition of CapsLock is pressed	1
KB_Shift	CHAR	condition of Shift key is pressed	1
Total			4

[Synopsis of INPUTKEY]

The structure INPUTKEY is used for replacing pressed key code into ASCII code and Hanin predefined scancode. HaninLib deals with the zhuyin fuhao according to input ASCII code, scancode, and status of CapsLock key and Shift key.

Table 4: Structure of CONVRESULT

Member	Type	Description	Size (byte)
szEdit	UCHAR	input buffer, the maximum number of 2-byte characters is 40	96
szAttr	UCHAR	attribute of each byte in input buffer ATTRIB_INPUT: zhuyin/pinyin (converting) ATTRIB_CONVERTED: Chinese 2-bye character	96
EditLength	SHORT	length of characters in input buffer	2
szSendBack	UCHAR	buffer storing characters will be sent back to application	82
SendBackLength	SHORT	length of characters in send back buffer	2
CursorPos	SHORT	current cursor position	2
Total			280

[Synopsis of CONVRESULT]

The structure CONVRESULT contains Chinese conversion result. Developers can get the latest conversion result from structure CONVRESULT during HaninLib is active.

Table 5: Structure of EXTENSION

Member	Type	Description	Size (byte)
szLong	UCHAR	long form of abbreviation	84
LongExtLength	SHORT	character length of long form of abbreviation	2
szShort	UCHAR	short form of abbreviation	52
ShortExtLength	SHORT	character length of short form of abbreviation	2
Total			140

[Synopsis of EXTENSION]

The structure EXTENSION is used for replacing base text by short form text or long form text for abbreviation function.

Table 6: Structure of USERWORD

Member	Type	Description	Size (byte)
szZhuin	UCHAR	zhuyin string of user defined word	42
szUserWord	UCHAR	user defined word	12
Total			54

[Synopsis of USERWORD]

The structure USERWORD is used for storing Chinese word and its zhuyin fuhao in order to register user defined word into user dictionary.

Table 7: Structure of CORE

Member	Type	Description	Size (byte)
clen	SHORT	length of character	8
wlen	SHORT	length of word	8
cidx	LONG	index of character	16
widx	LONG	index of word	16
wrange	SHORT	range for word	8
mlenmark	CHAR	length mark for MD	4
ulenmark	CHAR	length mark for UD	1
nowkey	CHAR	current input keys	10
fchar	UCHAR	the first found 5 Chinese characters	10
flag	CHAR	flag to indicate change of character	1
flag_bak	CHAR	backup for flag	1
Total			83

[Synopsis of CORE]

The structure CORE keeps the index information of input Chinese character and word during conducting Chinese conversion. The searching Chinese character or word in Hanin dictionaries can be accelerated by using structure CORE.

Table 8: Structure of IDX1

Member	Type	Description	Size (byte)
schar	CHAR	not used in HaninLib	1
charoff2	UCHAR	low byte of character position in dictionary	1
idxoff2	UCHAR	low byte of index in dictionary	1
charoff1	UCHAR:4	high byte of character position in dictionary	0.5
idxoff1	UCHAR:4	high byte if index in dictionary	0.5
Total			4

[Synopsis of IDX1]

The structure IDX1 stores the index and position in Hanin dictionaries regarding input Chinese character. Developer can retrieve those available Chinese characters by structure IDX1.

Table 9: Structure of LDRec

Member	Type	Description	Size (byte)
last	UCHAR	index of previous record in learning dictionary	1
next	UCHAR	index of next record in learning dictionary	1
cmpCode	UCHAR	compressed code of character/word	4
bigCode	UCHAR	Big5 code of character/word	4
Total			10

[Synopsis of LDRec]

The structure LDRec is used for storing learning dictionary (in user dictionary). The contents of LDRec will be updated according to input Chinese character and word when HaninLib is active.

Table 10: Structure of WORKAREA

Member	Type	Description	Size (byte)
cchResult	USHORT	length of result Chinese characters	2
chResult	CHAR	Big5 code of result Chinese characters	64
chMode	CHAR	mode for displaying character	64
attrMode	CHAR	attribute of displaying character	64
chDisplay	CHAR	displaying characters	64
attrDisplay	CHAR	display attribute of characters	64
PininPos	CHAR	length of pinyin symbol in input buffer, default is -1	1
InputState	CHAR	current input state	1
AddrState	CHAR	flag of usage of Address dictionary	1
ENDSEL	CHAR	flag to indicate whether the cursor is at the end of pinyin symbol in input buffer	1
born_flag	CHAR	flag to recover zhuyin	1
f_want	CHAR	cursor position (in byte) in input buffer	1
j_off	CHAR	editing string length (in byte) in input buffer	1
ownoff	CHAR	length of zhuyin symbol in input buffer	1
freq_fg	CHAR	frequency for comparing with previous/next characters	1
word_fg	CHAR	word frequency	1
distmp	UCHAR	zhuyin fuhao of current input character	6
youwant	UCHAR	Big5 code of Chinese character	40
youwant_bak	UCHAR	backup of youwant	40
cmpkigo	UCHAR	compressed index of zhuyin fuhao	40

nowkey_buf	CHAR	current input keys	10
Core	CORE	character information for each input Chinese character	1,660
liang	UCHAR	measure word （量詞） and its frequency	3
chbuf	UCHAR	choice buffer for same pronounced characters and words	600
undoStart	CHAR	start position of same pronounced character/word	1
undoLen	CHAR	length of same pronounced character/word	1
rule_buf	UCHAR	conversion rule of pinyin or zhuyin fuhao	10
pos_buf	UCHAR	position of pinyin symbol	10
outoff	ULONG	offset of character address in dictionaries	4
oldfreq	CHAR	previous frequency	1
specChoice	CHAR	flag to select 2-byte symbols	1
Total			2,759

[Synopsis of WORKAREA]

The structure WORKAREA contains the necessary information for conducting Chinese conversion in HaninLib. The information in structure WORKAREA is mainly used for dealing with the edit buffer for input keys, display zhuyin fuhao and conversion result, operation when receiving scancode.

Table 11: Structure of AIMEISTRUCT

Member	Type	Description	Size (byte)
nPos	SHORT	start position of aimei character/word	2
nCount	SHORT	amount of aimei character/word	2
aimei_cmp	UCHAR	compressed code of aimei character/word	2
szAimeiPron	UCHAR	aimei characters/words	80
Total			86

[Synopsis of AIMEISTRUCT]

In aimei function, the structure AIMEISTRUCT is used for storing those Chinese characters and words which pronunciations similar to input Chinese character or word.

2.4 Memory Requirement for Global Variables

The memory requirement of HaninLib consists of working area for Hanin conversion algorithm, state variables for indicating input status and flags, index buffer for operating dictionaries, and system/user/professional dictionaries paths. Table 12 lists the necessary memory required by HaninLib.

Table 12: Memory requirements of HaninLib

Variable	Type	Description	Size (byte)
pWA	WORKAREA *	pointer to structure WORKAREA	4
hanin_dic	CHAR	path of system dictionary	256
add_dic	CHAR	path of address dictionary	256
aimei_dic	CHAR	path of aimei dictionary	256
user_dic	CHAR	path of user dictionary	256
big5e_dic	CHAR	path of BIG5-E dictionary	256
prof_dic	CHAR	paths of 3 professional dictionaries	768
mdIdxTable1	IDX1	index information of system dictionary, profrssional dictionaries	1,160
ldWord	LDRec	learning information of learning dictionary	1,010
udHashIdx	SHORT	hash index of user dictionary	96
udHashBuf	UCHAR	hash buffer of user dictionary	16,384
inkey	UCHAR	current input key	1
inkey_bak	UCHAR	backup of inkey	1
chlen	SHORT	current size of choice buffer	2
oldchlen	SHORT	previous size of choice buffer	2
idxbuff	UCHAR	index buffer of abbreviation	1,536
toutbuf	UCHAR	character buffer for retrieving character/word from dictionaries	1,536
jui_iner	CHAR *	pointer to indicate keyboard layout of specified zhuyin keyboard	4
PininMode	CHAR	current pinyin mode	1
DisplayMode	CHAR	current display mode	1
InputMode	CHAR	current input mode	1
AddressState	CHAR	current address mode	1
ldDirty	CHAR	flag to indicate the change of learning dictionary	1
ldWordHead	UCHAR	the start index of learning character/word	1
g_bReady	SHORT	flag to indicate HaninLib is ready	2
g_bInUWMaintenance	SHORT	flag to indicate the state of maintenance of user dictionary	2
Total			23,794

[Description of HaninLib Global Variables]

SYNTAX **WORKAREA *pWA**

SYNOPSIS *pWA* is a pointer to represent the allocated working area when initializing HaninLib.

SYNTAX **char hanin_dic[256]**

SYNOPSIS *hanin_dic* is the filename of Hanin system dictionary. The filename of dictionary should contain full path of this dictionary.

SYNTAX **char add_dic[256]**

SYNOPSIS *add_dic* is the filename of Hanin address dictionary. The filename of dictionary should contain full path of this dictionary.

SYNTAX **char aimei_dic[256]**

SYNOPSIS *aimei_dic* is the filename of Hanin aimei dictionary. The filename of dictionary should contain full path of this dictionary.

SYNTAX **char user_dic[256]**

SYNOPSIS *user_dic* is the filename of Hanin user dictionary. The filename of dictionary should contain full path of this dictionary.

SYNTAX **char big5e_dic[256]**

SYNOPSIS *big5e_dic* is the filename of Big5E character dictionary. The filename of dictionary should contain full path of this dictionary. This Big5E character dictionary is mainly used in Hongkong area.

SYNTAX **char prof_dic[3][256]**

SYNOPSIS *prof_dic[3]* are the filenames of user selected Hanin professional dictionaries. The filename of dictionary should contain full path of this dictionary.

SYNTAX **IDX1 mdIdxTable1[MAXPROFDICOPEND+2][MD_MAX_IDX]**

SYNOPSIS *mdIdxTable1* is the index table and position table of system dictionary or specified professional dictionary. *mdIdxTable1* can assist in finding available Chinese characters from Hanin dictionaries.

SYNTAX **LDRec ldWord[LD_MAX_WORD+1]**

SYNOPSIS *ldWord* contains all the learning information of characters and pronunciations. The *ldWord* data is loaded when HaninLib initialization, restored to user dictionary when HaninLib is terminated.

SYNTAX **short udHashIdx[UD_MAX_IDX+1]**

SYNOPSIS *udHashIdx* is the index table used for indicating user defined words.

SYNTAX **unsigned char udHashBuf[UD_MAX_SIZE]**

SYNOPSIS *udHashBuf* recorded user defined words and their pronunciations.

SYNTAX **unsigned char inkey**

SYNOPSIS *inkey* is the key code of current input key

SYNTAX **unsigned char inkey_bak**

SYNOPSIS *inkey_bak* is a duplication of *inkey* for backup of input key code.

SYNTAX **short chlen**

SYNOPSIS *chlen* is the size (in byte) of current choice buffer for selecting same pronounced character or word.

SYNTAX **short oldchlen**

SYNOPSIS *oldchlen* is the previous size (in byte) of choice buffer. The *oldchlen* is mainly used when the size of choice buffer is bigger than 255.

SYNTAX **unsigned char idxbuff[AD_IDX_SIZE]**

SYNOPSIS *idxbuff* is the index table for abbreviation.

SYNTAX **unsigned char toutbuf[AD_IDX_SIZE]**

SYNOPSIS *toutbuf* is a character buffer for retrieving Chinese characters or words from Hanin dictionaries.

SYNTAX **char *jui_iner**

SYNOPSIS *jui_iner* is an pointer to indicate selected zhuyin keyboard mapping. This pointer is used when the input mode is not pinyin mode.

SYNTAX **char PininMode**

SYNOPSIS *PininMode* represents current pinyin mode. *PininMode* is 0 if current input mode is zhuyin.

SYNTAX **char DisplayMode**

SYNOPSIS *DisplayMode* represents the pronounce symbol for displaying in Hanin edit buffer. The *DisplayMode* can't be alphabet character if current input mode is zhuyin.

SYNTAX **char InputMode**

SYNOPSIS *InputMode* represents current input mode is zhuyin with specified keyboard mapping or pinyin.

SYNTAX **char AddressState**

SYNOPSIS *AddressState* indicates the usage of address dictionary. If *AddressState* is active, HaninLib will retrieve character or word from address dictionary first.

SYNTAX **char ldDirty**

SYNOPSIS *ldDirty* is a flag to indicate the modification status of learning dictionary. The learning dictionary is updated if *ldDirty* is not 0.

SYNTAX **unsigned char ldWordHead**

SYNOPSIS *ldWordHead* represents the head record of learning dictionary.

SYNTAX **short g_bReady**

SYNOPSIS *g_bReady* is a flag to represent current status of HaninLib. HaninLib is ready if *g_bReady* is TRUE.

SYNTAX **short g_bInUWMaintenance**

SYNOPSIS *g_bInUWMaintenance* is a flag to indicate current status of maintenance of user dictionary. When maintaining user defined words, the *g_bInUWMaintenance* will be TRUE.

2.5 API Description

HaninLib API consists of core functions of Hanin Chinese input algorithm to assist developer in development of Hanin input method on various platforms such as Windows, Mac OS, and Linux. Table 13 lists the HaninLib API functions.

Table 13: List of HaninLib API functions

API Name	Description
HaninKernel	
HAN_Start	HAN_Start initializes HaninLib. It's necessary for a caller function to prepare a data structure of HANININIT for initializing global variables in HaninLib
HAN_Open	Hanin_Open creates an instance for Hanin input method and allocates an adequate amount of memory for using in HaninLib.
HAN_InputKey	do the operation when input a pressed key
HAN_IsAllConverted	return TRUE if no converting zhuyin/pinyin in input buffer, otherwise, return FALSE
HAN_GetConvResult	get the converted result into structure CONVRESULT
HAN_Query	get current settings of paths of Hanin dictionaries and conditions of input status
HAN_ClearEditBuf	clear the input buffer and get the converted result
HAN_RegUserWord	register selected word in input buffer to be an user-defined word stored in user dictionary
HAN_Close	free the allocated memory by HAN_Open function and re-new the learning dictionary
HaninSelection	
HAN_GetSamePronChars	retrieve all the same pronounced characters as the selected Chinese character at current cursor
HAN_GetSamePronWords	retrieve all the same pronounced words as the selected Chinese word at current cursor

HAN_SelectSamePronChar	choice Chinese character at current cursor in input buffer into selected same pronounced character
HAN_SelectSamePronWord	choice Chinese word at current cursor in input buffer into selected same pronounced word
HAN_BeginGetAimei	allocate memory for structure AIMEISTRUCT, and start to find aimei characters
HAN_GetAimeiPronCount	return the number of aimei pronounced characters
HAN_GetAimeiChars	retrieve all the aimei pronounced characters of the Chinese character at current cursor
HAN_GetAimeiWords	retrieve all the aimei pronounced words of the Chinese word at current cursor in input buffer
HAN_SelectAimeiChar	change Chinese character at current cursor in input buffer into selected aimei pronounced character
HAN_SelectAimeiWord	change Chinese word at current cursor in input buffer into selected aimei pronounced word
HAN_EndGetAimei	free the allocated memory for structure AIMEISTRUCT
HAN_GetExtension	get the long form and short form of abbreviation for base text in input buffer
HaninUserDic	
HAN_GetExtIdxCount	get the number of registered abbreviations
HAN_GetAllExtIdx	retrieve all the registered abbreviations
HAN_FindExtension	retrieve the long form text and short form text of indicated base text
HAN_RegExtension	register an abbreviation
HAN_DelExtension	delete an abbreviation
HAN_BeginUWordMaintenance	start to maintain user defined words by loading user dictionary and allocates memory area for structure SORTEDLIST
HAN_GetZhuinList	get the list of zhuyin fuhao
HAN_GetUserWordStartIndex	get the index in user dictionary for user defined word first using indicated zhuyin fuhao
HAN_GetUserWordCount	get the amount of user defined words in user dictionary
HAN_GetUserWord	get the user defined word and its zhuyin fuhao
HAN_DelUserWord	remove one user defined word from user dictionary
HAN_GetFreeUserChars	get the remainder number of room to register user defined word
HAN_EndUWordMaintenance	end the maintenance of user defined word by storing updated user dictionary and free the allocated memory area
HaninSettings	
HAN_SetInputMode	set Hanin input mode and change the display mode based on new input mode

	moreover, load the keyboard mapping if input mode is zhuyin
HAN_SetDisplayMode	set display language in input buffer
HAN_SetAddressState	set the flag of using address dictionary when converting
HAN_SetInputState	set the input status

The following sections describe all the HaninLib API functions in detail.

2.5.1 HAN_Start

The initialization process when develop Chinese input method using HaninLib. The caller function should prepare certain amount of memory for structure HANININIT in order to initialize global variables using in HaninLib.

---

short HAN_Start (PHANININIT *pHaninInit*)

ARGUMENTS

pHaninInit

[IN] a pointer to structure HANININIT, HaninLib can get the initializing data of dictionaries paths and input status from it, refer to Table 2 for detail of structure HANININIT
if not specifies the path of user dictionary, HaninLib will automatically create a user dictionary

SYNOPSIS

(1) duplicate dictionaries paths from HANININIT into global variables
(2) check the availability of system dictionary, address dictionary, professional dictionaries, and user dictionary
(3) if Big5E character dictionary exists, load Big5E characters
(4) set input mode according to values of input mode and display mode in HANININIT
(5) set *g_bReady* to be True

VARIABLES IN USE

g_bReady

[OUT] a flag to represent HaninLib is ready

RETURN VALUE

True: success

RELATED FUNCTIONS

CheckDictionary, ReadBig5E, SetInputMode

2.5.2 HAN_Open

This function creates a Hanin instance for each application calling it. When terminate a Hanin instance, it's necessary to free allocated memory by this function.

PHANIN HAN_Open (void)

ARGUMENTS

void

SYNOPSIS

(1) allocate memory for structure WORKAREA
(2) initialize the allocated working area
(3) return pointer *pWA* if it's available

VARIABLES IN USE

N/A

RETURN VALUE

NULL:	*g_bReady* is False, or *pWA* is NULL
pWA:	a pointer to structure WORKAREA which is working area for Hanin input method

RELATED FUNCTIONS

reset_work_area, Encrypt

2.5.3 HAN_InputKey

This function conducts the operation of Chinese conversion based on user pressed keys and gets the result of conversion.

---

short HAN_InputKey (PHANIN *pHanin*, PINPUTKEY *pInputKey*,
PCONVRESULT *pConvResult*)

ARGUMENTS

pHanin

[IN] a pointer to structure HANIN, it will be redirected to be the pointer of Hanin working area

pInputKey

[IN] a pointer to structure INPUTKEY which consists of ASCII code, scan code of input key and status of CapsLock/Shift key

pConvResult

[OUT] a pointer to structure CONVRESULT, it's used to retrieve Chinese characters in input buffer for sending back to application

SYNOPSIS

(1) get ASCII code and scan code of input key from *pInputKey*
(2) set *KB_STATUS* by the status of CapsLock and Shift
(3) call Hanin core function to conduct the Chinese conversion process
(4) retrieve the conversion result into *pConvResult*

VARIABLES IN USE

KB_STATUS

[OUT] a flag to represent pressed status of CapsLock key and Shift key

scancode

[OUT] a flag to represent scan code of input key

RETURN VALUE

False:	*g_bReady* is False, or *pHanin* is NULL
True:	success

RELATED FUNCTIONS

Decrypt, realtime, Encrypt, HAN_GetConvResult

2.5.4 HAN_IsAllConverted

This function checks the status of completion of Chinese conversion in Hanin input buffer

short HAN_IsAllConverted (PHANIN *pHanin*)

ARGUMENTS

pHanin

[IN] a pointer to structure HANIN, it will be redirected to be the pointer of HaninLib working area

SYNOPSIS

(1) return the status of conversion depends on the value of *ownoff*

VARIABLES IN USE

ownoff

[OUT] to indicate the length of zhuyin fuhao (or pinyin alphabet)

RETURN VALUE

False:	*g_bReady* is False, or *pHanin* is NULL, or *ownoff* is not 0
True:	*ownoff* is 0, i.e. the Chinese character conversion is completed

RELATED FUNCTIONS

Decrypt, Encrypt

2.5.5 HAN_GetConvResult

This function retrieves the result of Chinese conversion in Hanin input buffer

---

void HAN_GetConvResult (PHANIN *pHanin*, PCONVRESULT *pConvResult*)

ARGUMENTS

pHanin

[IN] a pointer to structure HANIN, it will be redirected to be the pointer of Hanin working area

pConvResult

[OUT] a pointer to structure CONVRESULT, it's used to retrieve Chinese characters in input buffer for sending back to application

SYNOPSIS

(1) get the length of characters in input buffer, and copy the characters in input buffer into structure CONVRESULT
(2) give the attribute of character in structure CONVRESULT to be ATTRIB_CONVERTED if conversion is complete
(3) give the character attribute to be ATTRIB_INPUT if character is zhuyin fuhao or pinyin alphabet
(4) duplicate the conversion result into structure CONVRESULT

VARIABLES IN USE

pWA->chResult

[OUT] clear character buffer after copy conversion result into structure CONVRESULT

pWA->cchResult

[OUT] reset length of character buffer after copy conversion result into structure CONVRESULT

RETURN VALUE

void

RELATED FUNCTIONS

Decrypt, Encrypt

2.5.6 HAN_Query

This function retrieves current setting of dictionaries paths and conditions of input status

---

short HAN_Query (PHANININIT *pHaninInit*)

ARGUMENTS

pHaninInit

[OUT] a pointer to structure HANININIT, retrieve current HaninLib setting of dictionaries paths and conditions of input status

SYNOPSIS

(1) duplicate paths of system dictionary, address dictionary, aimei dictionary, user dictionary, Big5E dictionary, and professional dictionaries into structure HANININIT

(2) copy display mode, input mode, and address mode into structure HANININIT

VARIABLES IN USE

N/A

RETURN VALUE

False:	*g_bReady* is False
True:	success

RELATED FUNCTIONS

GetInputMode

2.5.7 HAN_ClearEditBuf

This function initializes the HaninLib input buffer and retrieves conversion result

void HAN_ClearEditBuf (PHANIN *pHanin*, PCONVRESULT *pConvResult*)

ARGUMENTS

pHanin

[IN] a pointer to structure HANIN, it will be redirected to be the pointer of HaninLib working area

pConvResult

[OUT] a pointer to structure CONVRESULT, it's used to retrieve Chinese characters in input buffer for sending back to application

SYNOPSIS

(1) initialize the HaninLib working area

(2) retrieve conversion result into structure CONVRESULT

VARIABLES IN USE

N/A

RETURN VALUE

void

RELATED FUNCTIONS

Decrypt, initial, Encrypt, HAN_GetConvResult

2.5.8 HAN_RegUserWord

The user defined word should be registered during Chinese conversion in input buffer. User moves the cursor to the left of Chinese word (2~4 Chinese characters) in input buffer, and presses function key to online register user defined word. This function registers the selected Chinese word into user dictionary.

short HAN_RegUserWord (PHANIN *pHanin*, short *nLength*)

ARGUMENTS

pHanin

[IN] a pointer to structure HANIN, it will be redirected to be the pointer of HaninLib working area

nLength

[IN] the length (in byte) of user defined word

SYNOPSIS

(1) if *nLength* is not specified, set the length of user defined word is difference in character length between end of input buffer and cursor position

(2) call UDInsertWord to register Chinese word into user dictionary

VARIABLES IN USE

g_bReady

[IN] a flag to represent HaninLib is ready

g_bInUWMaintenance

[IN] a flag to represent current operation is on maintenance of user dictionary

RETURN VALUE

RC_OTHERERROR	*g_bReady* is False or *pHanin* is NULL or zhuyin fuhao in input buffer or user dictionary doesn't exist
RC_INUWMAINTENANCE	*g_bInUWMaintenance* is True
RC_ERRORUSERWORDLEN	length of user defined word is not correct
RC_INVALIDWORD	length of user defined word is not 2~4 Chinese characters or 2-byte symbol exists in user defined word
RC_USERWORDFULL	user dictionary overflow
RC_WORDEXIST	user defined word already exists in user dictionary
RC_SUCCESS	success

RELATED FUNCTIONS

Decrypt, Encrypt, UDInsertWord

2.5.9 HAN_Close

This function closes the HaninLib operation. It frees the allocated memory area by HAN_Open and updates the learning dictionary.

---

void HAN_Close (PHANIN *pHanin*)

ARGUMENTS

pHanin

[IN] a pointer to structure HANIN, it will be redirected to be the pointer of HaninLib working area

SYNOPSIS

(1) update learning dictionary
(2) free the memory area of *pHanin*

VARIABLES IN USE

N/A

RETURN VALUE

void

RELATED FUNCTIONS

Decrypt, LDWrite

2.5.10 HAN_GetSamePronChars

This function retrieves all the Chinese characters which the zhuyin fuhao is identical to specified Chinese character. If the cursor is at the end of input buffer, the latest character is the specified Chinese character. If the cursor is within the input buffer, the right character of the cursor is the specified Chinese character.

---

short HAN_GetSamePronChars (PHANIN *pHanin*, PBYTE *pCandBuf*, short *CandBufSize*, short *bSpecCharOnly*)

ARGUMENTS

pHanin

[IN] a pointer to structure HANIN, it will be redirected to be the pointer of HaninLib working area

pCandBuf

[OUT] a pointer to a character buffer which receives all the Chinese characters which the zhuyin fuhao is identical

CandBufSize

[IN] indicates the size (in byte) of *pCandBuf*

bSpecCharOnly

[IN] a flag to represent that only retrieve the 2-byte symbol characters

SYNOPSIS

(1) if the specified Chinese character is a 2-byte symbol, then retrieves all the 2-byte symbols
otherwise, retrieves all the same pronounced Chinese characters from learning dictionary, professional dictionaries, and system dictionary

VARIABLES IN USE

N/A

RETURN VALUE

length: the character length (in byte) of *pCandbuf*

RELATED FUNCTIONS

GetAllSpecChars, GetSameProns

2.5.11 HAN_GetSamePronWords

This function retrieves all the Chinese words which the zhuyin fuhao is identical to specified Chinese word. If the cursor is at the end of input buffer, the latest Chinese word is the specified Chinese word. If the cursor is within the input buffer, the right Chinese word of the cursor is the specified Chinese word.

HAN_GetSamePronWords can only retrieve Chinese word consists of 2 Chinese characters.

---

short HAN_GetSamePronWords (PHANIN *pHanin*, PBYTE *pCandBuf*, short *CandBufSize*)

ARGUMENTS

pHanin

[IN] a pointer to structure HANIN, it will be redirected to be the pointer of HaninLib working area

pCandBuf

[OUT] a pointer to a character buffer which receives all the Chinese words which the zhuyin fuhao is identical to specified Chinese word

CandBufSize

[IN] indicates the size (in byte) of *pCandBuf*

SYNOPSIS

(1) call GetSameProns to retrieve all the same pronounced Chinese words from learning dictionary, user dictionary, professional dictionaries, and system dictionary

VARIABLES IN USE

N/A

RETURN VALUE

length: the character length (in byte) of *pCandbuf*

RELATED FUNCTIONS

GetSameProns

2.5.12 HAN_SelectSamePronChar

This function choices one of the Chinese character which the zhuyin fuhao is identical to specified Chinese character. If the cursor is at the end of input buffer, the latest character is the specified Chinese character. If the cursor is within the input buffer, the right character of the cursor is the specified Chinese character.

---

short HAN_SelectSamePronChar (PHANIN *pHanin*, PBYTE *pSelBuf*, PCONVRESULT *pConvResult*)

ARGUMENTS

pHanin

[IN] a pointer to structure HANIN, it will be redirected to be the pointer of HaninLib working area

pSelBuf

[IN] a pointer to a character buffer which includes all the Chinese characters which the zhuyin fuhao is identical

pConvResult

[OUT] a pointer to structure CONVRESULT, it's used to retrieve Chinese characters in input buffer for sending back to application

SYNOPSIS

(1) choice one of the same pronounced Chinese character or 2-byte symbol from *pSelBuf*

VARIABLES IN USE

N/A

RETURN VALUE

False: *g_bReady* is False or *pHanin* is NULL or zhuyin fuhao exists in input buffer or cursor is at the leftmost position in input buffer

True: success

RELATED FUNCTIONS

DoSelectSpecChar, SelectSamePron

2.5.13 HAN_SelectSamePronWord

This function choices one of the Chinese word which the zhuyin fuhao is identical to specified Chinese word. If the cursor is at the end of input buffer, the latest Chinese word is the specified Chinese word. If the cursor is within the input buffer, the right Chinese word of the cursor is the specified Chinese word.

---

short HAN_SelectSamePronWord (PHANIN *pHanin*, PBYTE *pSelBuf*, PCONVRESULT *pConvResult*)

ARGUMENTS

pHanin

[IN] a pointer to structure HANIN, it will be redirected to be the pointer of HaninLib working area

pSelBuf

[IN] a pointer to a character buffer which includes all the Chinese characters which the zhuyin fuhao is identical

pConvResult

[OUT] a pointer to structure CONVRESULT, it's used to retrieve Chinese characters in input buffer for sending back to application

SYNOPSIS

(1) choices one of the same pronounced Chinese word from *pSelBuf*

VARIABLES IN USE

N/A

RETURN VALUE

False: *g_bReady* is False or *pHanin* is NULL or zhuyin fuhao exists in input buffer or the specified Chinese word is less than 2 Chinese

characters

True: success

RELATED FUNCTIONS

SelectSamePron

2.5.14 HAN_BeginGetAimei

This function starts the fuzzy function (called aimei function in HaninLib) to find all the probable Chinese character or word which its zhuyin fuhao is similar to specified Chinese character or word according to aimei rules defined in HaninLib.

If the cursor is at the end of input buffer, the latest Chinese character or word is the specified Chinese character or word. If the cursor is within the input buffer, the right Chinese character or word of the cursor is the specified Chinese character or word.

HAN_BeginGetAimei allocates a memory area for structure AIMEISTRUCT. This allocated memory area needs to be used in all the related aimei functions.

PAIMEI HAN_BeginGetAimei (PHANIN *pHanin*)

ARGUMENTS

pHanin

[IN] a pointer to structure HANIN, it will be redirected to be the pointer of HaninLib working area

SYNOPSIS

(1) allocate memory area for working area of aimei function, and assign it to a pointer *pAimeiStruct*
(2) duplicate cursor position and compressed zhuyin code to *pAimeiStruct*
(3) get the number of aimei zhuyin related to compressed zhuyin code from aimei dictionary

VARIABLES IN USE

N/A

RETURN VALUE

False: *g_bReady* is False or *pHanin* is NULL

NULL: zhuyin fuhao exists in input buffer or input buffer is empty or *pAimeiStruct* can't be allocated or the specified Chinese character is

a 2-byte symbol

pAimeiStruct: a pointer to working area for aimei functions

RELATED FUNCTIONS

Decrypt, Encrypt, shift_cursor, sear_aimei

2.5.15 HAN_GetAimeiPronCount

This function returns the number of aimei zhuyin characters regarding specified Chinese character in input buffer.

short HAN_GetAimeiPronCount (PHANIN *pHanin*, PAIMEI *pAimei*)

ARGUMENTS

pHanin

[IN] a pointer to structure HANIN, it will be redirected to be the pointer of HaninLib working area

pAimei

[IN] a pointer to structure AIMEI, it's a pointer to working area of aimei functions

SYNOPSIS

(1) get the number of aimei zhuyin characters from *pAimei*

VARIABLES IN USE

N/A

RETURN VALUE

0: *g_bReady* is False or *pHanin* is NULL or *pAimei* is NULL

count: number of aimei zhuyin characters

RELATED FUNCTIONS

Decrypt, Encrypt

2.5.16 HAN_GetAimeiChars

This function retrieves all the aimei zhuyin characters regarding specified Chinese character in input buffer.

short HAN_GetAimeiChars (PHANIN *pHanin*, PAIMEI *pAimei*, short

***idx*, PBYTE *pCandBuf*, short *CandBufSize*,**
PBYTE *pZhuin*, short **pZhuinLength*)

ARGUMENTS

pHanin

[IN] a pointer to structure HANIN, it will be redirected to be the pointer of HaninLib working area

pAimei

[IN] a pointer to structure AIMEI, it's a pointer to working area of aimei functions

idx

[IN] the index of compressed zhuyin code in *pAimeiStruct* (count from 0)

pCandBuf

[OUT] a pointer to a character buffer which receives all the aimei zhuyin Chinese characters

CandBufSize

[IN] indicates the size (in byte) of *pCandBuf*

pZhuin

[OUT] a pointer to a character buffer which receives all the aimei zhuyin fuhao regarding aimei zhuyin Chinese characters
the format of zhuyin fuhao depends on the setting of Display mode

pZhuinLength

[OUT] indicates the size (in byte) of *pZhuin*

SYNOPSIS

(1) duplicate compressed zhuyin fuhao from *pAimeiStruct* to *cmpkigo*
(2) initialize the content of structure *Core*, and character buffer *chbuf*
(3) if *pZhuin* is not NULL, then get the string of zhuyin fuhao or pinyin alphabet according to *DisplayMode*
(4) search all the aimei Chinese character from learning dictionary, professional dictionaries, user dictionary, system dictionary, and Big5E dictionary
(5) copy aimei Chinese characters from *chbuf* to *pCandBuf*

VARIABLES IN USE

cmpkigo

[OUT] a character buffer to copy compressed zhuyin fuhao from *pAimeiStruct*

pWA->Core

[OUT/IN] a structure of CORE to keep the information for searching aimei characters in Hanin dictionaries

pWA->chbuf

[OUT/IN] a character buffer to store all the aimei characters in Hanin dictionaries

chlen

[OUT/IN] character length (in byte) of *chbuf*

RETURN VALUE

0:	*g_bReady* is False or *pHanin* is NULL or *pAimei* is NULL or *idx* is not available in [0, *pAimeiStruct->nCount*]
length:	length of character buffer (*pCandBuf*) of aimei zhuyin characters

RELATED FUNCTIONS

Decrypt, Encrypt, init_len_mark, shift_cursor, cmp2pinin, cmp2zhuin

2.5.17 HAN_GetAimeiWords

This function retrieves all the Chinese words which have aimei zhuyin fuhao regarding specified Chinese word in input buffer.

---

short HAN_GetAimeiWords (PHANIN *pHanin*, PAIMEI *pAimei*, short *idx*, PBYTE *pCandBuf*, short *CandBufSize*, PBYTE *pZhuin*, short **pZhuinLength*)

ARGUMENTS

pHanin

[IN] a pointer to structure HANIN, it will be redirected to be the pointer of HaninLib working area

pAimei

[IN] a pointer to structure AIMEI, it's a pointer to working area of aimei functions

idx

[IN] the index of compressed zhuyin code in *pAimeiStruct* (count from 0)

pCandBuf

[OUT] a pointer to a character buffer which receives all the aimei zhuyin Chinese words

CandBufSize

[IN] indicates the size (in byte) of *pCandBuf*

pZhuin

[OUT] a pointer to a character buffer which receives all the aimei zhuyin fuhao regarding aimei zhuyin Chinese word
the format of zhuyin fuhao depends on the setting of Display mode

pZhuinLength

[OUT] indicates the size (in byte) of *pZhuin*

SYNOPSIS

(1) duplicate compressed zhuyin fuhao from *pAimeiStruct* to *cmpkigo*
(2) initialize the content of structure *Core*, and character buffer *chbuf*
(3) if *pZhuin* is not NULL, then get the string of zhuyin fuhao or pinyin alphabet according to *DisplayMode*
(4) search all the aimei Chinese character from learning dictionary, professional dictionaries, user dictionary, system dictionary, and Big5E dictionary
(5) copy aimei Chinese characters from *chbuf* to *pCandBuf*

VARIABLES IN USE

cmpkigo

[OUT] a character buffer to copy compressed zhuyin fuhao from *pAimeiStruct*

pWA->Core

[OUT/IN] a structure of CORE to keep the information for searching aimei words in Hanin dictionaries

pWA->chbuf

[OUT/IN] a character buffer to store all the aimei Chinese words in Hanin dictionaries

chlen

[OUT/IN] character length (in byte) of *chbuf*

RETURN VALUE

0:	*g_bReady* is False or *pHanin* is NULL or *pAimei* is NULL or *idx* is not available in [0, *pAimeiStruct->nCount*] or specified Chinese word less than 2 Chinese characters
length:	length of character buffer (*pCandBuf*) of aimei zhuyin characters

RELATED FUNCTIONS

Decrypt, Encrypt, init_len_mark, shift_cursor, cmp2pinin, cmp2zhuin

2.5.18 HAN_SelectAimeiChar

This function chooses one of the Chinese characters from structure *pAimei*.

short HAN_SelectAimeiChar (PHANIN *pHanin*, PAIMEI *pAimei*, short *idx*, PBYTE *pSelBuf*, PCONVRESULT

***pConvResult*)**

ARGUMENTS

pHanin

[IN] a pointer to structure HANIN, it will be redirected to be the pointer of HaninLib working area

pAimei

[IN] a pointer to structure AIMEI, it's a pointer to working area of aimei functions

idx

[IN] the index of compressed zhuyin code in *pAimeiStruct* (count from 0)

pSelBuf

[IN] a pointer to a character buffer which includes all the Chinese characters of aimei zhuyin fuhao

pConvResult

[OUT] a pointer to structure CONVRESULT, it's used to retrieve Chinese characters in input buffer for sending back to application

SYNOPSIS

(1) select one aimei Chinese character from *pAimei*

VARIABLES IN USE

N/A

RETURN VALUE

False:	*g_bReady* is False or *pHanin* is NULL or *pAimei* is NULL or *idx* is not available in [0, *pAimeiStruct->nCount*] or zhuyin fuhao exists in input buffer or cursor is at the leftmost position in input buffer
0:	*idx* is not available in [0, *pAimeiStruct->nCount*]
True:	success

RELATED FUNCTIONS

SelectAimeiPron

2.5.19 HAN_SelectAimeiWord

This function chooses one of the Chinese words from structure *pAimei*.

short HAN_SelectAimeiWord (PHANIN *pHanin*, PAIMEI *pAimei*, short

***idx*, PBYTE *pSelBuf*, PCONVRESULT *pConvResult*)**

ARGUMENTS

pHanin

[IN] a pointer to structure HANIN, it will be redirected to be the pointer of HaninLib working area

pAimei

[IN] a pointer to structure AIMEI, it's a pointer to working area of aimei functions

idx

[IN] the index of compressed zhuyin code in *pAimeiStruct* (count from 0)

pSelBuf

[IN] a pointer to a character buffer which includes all the Chinese words of aimei zhuyin fuhao

pConvResult

[OUT] a pointer to structure CONVRESULT, it's used to retrieve Chinese words in input buffer for sending back to application

SYNOPSIS

(1) select one aimei Chinese word from *pAimei*

VARIABLES IN USE

N/A

RETURN VALUE

False:	*g_bReady* is False or *pHanin* is NULL or *pAimei* is NULL or *idx* is not available in [0, *pAimeiStruct->nCount*] or zhuyin fuhao exists in input buffer or Chinese word has less than 2 Chinese characters
0:	*idx* is not available in [0, *pAimeiStruct->nCount*]
True:	success

RELATED FUNCTIONS

SelectAimeiPron

2.5.20 HAN_EndGetAimei

This function terminates the aimei function by releasing allocated memory area for working area of aimei function.

short HAN_EndGetAimei (PHANIN *pHanin*, PAIMEI *pAimei*)

ARGUMENTS

pHanin

[IN] a pointer to structure HANIN, it will be redirected to be the pointer of HaninLib working area

pAimei

[IN] a pointer to structure AIMEI, it's a pointer to working area of aimei functions

SYNOPSIS

(1) release the memory area of *pAimei*

VARIABLES IN USE

N/A

RETURN VALUE

False:	*g_bReady* is False or *pHanin* is NULL or *pAimei* is NULL
True:	success

RELATED FUNCTIONS

N/A

2.5.21 HAN_GetExtension

This function retrieves the long form text or short form text according to the specified abbreviation (base text) in input buffer.

short HAN_GetExtension (PHANIN *pHanin*, PEXTENSION *pExtension*)

ARGUMENTS

pHanin

[IN] a pointer to structure HANIN, it will be redirected to be the pointer of HaninLib working area

pExtension

[OUT] a pointer to structure EXTENSION, it's a pointer to working area for storing long form text and short form text

SYNOPSIS

(1) retrieve long form text, short form text regarding specified abbreviation base text from user dictionary

VARIABLES IN USE

pWA->chbuf

[IN] a character buffer to store the long form text and short form text

RETURN VALUE

False:	*g_bReady* is False or *pHanin* is NULL or length of specified abbreviation is invalid or user dictionary can't be opened or abbreviation is not registered
True:	success

RELATED FUNCTIONS

q_extend

2.5.22 HAN_GetExtIdxCount

This function gets the number of registered abbreviations in user dictionary.

short HAN_GetExtIdxCount (void)

ARGUMENTS

void

SYNOPSIS

(1) count the number of abbreviations in user dictionary

VARIABLES IN USE

idxbuff

[IN] an index buffer to indicate the index of abbreviation in user dictionary

RETURN VALUE

0:	*g_bReady* is False
count:	number of abbreviations

RELATED FUNCTIONS

N/A

2.5.23 HAN_GetAllExtIdx

This function gets all the registered abbreviations from user dictionary. An abbreviation has 3 Chinese characters (i.e. 6-byte code). HAN_GetAllExtIdx saves 6-byte code into character buffer for each registered abbreviation. User can register at most 256 abbreviations in HaninLib.

---

short HAN_GetAllExtIdx (PBYTE *pExtIdx*)

ARGUMENTS

pExtIdx

[OUT] a pointer to character buffer for receiving all the abbreviations registered in user dictionary.

SYNOPSIS

(1) retrieves registered abbreviations from user dictionary

VARIABLES IN USE

idxbuff

[IN] an index buffer to indicate the index of abbreviation in user dictionary

RETURN VALUE

0: *g_bReady* is False

count: number of registered abbreviations

RELATED FUNCTIONS

N/A

2.5.24 HAN_FindExtension

This function retrieves the long form text and short form text according to specified abbreviation (base text). The long form text contains at most 40 bytes (20 Chinese characters), the short form text contains at most 24 bytes (12 Chinese characters).

---

short HAN_FindExtension (PBYTE *pExtIdx*, PEXTENSION *pExtension*)

ARGUMENTS

pExtIdx

[IN] a pointer to character buffer for receiving all the abbreviations registered in user dictionary

pExtension

[OUT] a pointer to structure EXTENSION, it's a pointer to working area for storing long form text and short form text

SYNOPSIS

(1) find the index of specified abbreviation

(2) retrieve long form text and short form text from user dictionary

VARIABLES IN USE

N/A

RETURN VALUE

False:	*g_bReady* is False or the specified abbreviation does not exist or fail to get long/short form text
True:	success

RELATED FUNCTIONS

FindExtID, GetExt

2.5.25 HAN_RegExtension

This function registers the long form text and short form text and the specified abbreviation (base text) into user dictionary.

---

short HAN_RegExtension (PBYTE *pExtIdx*, PEXTENSION *pExtension*)

ARGUMENTS

pExtIdx

[IN] a pointer to character buffer for receiving all the abbreviations registered in user dictionary

pExtension

[OUT] a pointer to structure EXTENSION, it's a pointer to working area for storing long form text and short form text

SYNOPSIS

(1) register long form text, short form text and abbreviation into user dictionary

VARIABLES IN USE

N/A

RETURN VALUE

RC_OTHERERROR:	*g_bReady* is False or index of abbreviation is wrong or user dictionary can't be opened
RC_INVALIDEXTID:	character length of abbreviation is invalid or abbreviation contains 2-byte space code (i.e. 0xA140)
RC_NOLONGEXT:	long form text does not exist
RC_ALREADYEXIST:	abbreviation already exists
RC_EXTENSIONFULL:	no room for registering abbreviation
RC_SUCCESS:	success

RELATED FUNCTIONS

RegExt

2.5.26 HAN_DelExtension

This function deletes the specified abbreviation and its long form text and short form text from user dictionary.

---

short HAN_DelExtension (PBYTE *pExtIdx*)

ARGUMENTS

pExtIdx

[IN] a pointer to character buffer for receiving all the abbreviations registered in user dictionary

SYNOPSIS

(1) delete the specified abbreviation and its long form text and short form text from user dictionary

VARIABLES IN USE

N/A

RETURN VALUE

False:	*g_bReady* is False or fail to delete the specified abbreviation
True:	success

RELATED FUNCTIONS

DelExt

2.5.27 HAN_BeginUWordMaintenance

This function starts the maintenance process of user dictionary. Users can't register any user defined word during maintenance process.

PUWM HAN_BeginUWordMaintension (void)

ARGUMENTS

void

SYNOPSIS

(1) allocate memory area for a sort list used in maintenance process by calling *hash_to_sort*
(2) set *g_bInUWMaintenance* to be True

VARIABLES IN USE

g_bInUWMaintenance
[OUT] a flag to represent execution status of the maintenance process of user defined word

RETURN VALUE

NULL:	*g_bReady* is False or maintenance process already started or fails to open user dictionary
100:	success

RELATED FUNCTIONS

UDLoad, hash_to_sort

2.5.28 HAN_GetZhuinList

This function gets the list of 37 zhuyin fuhao. The index of zhuyin fuhao operated in user word maintenance process is based on order of this zhuyin list.

short HAN_GetZhuinList (PBYTE *pZhuin*)

ARGUMENTS

pZhuin

[OUT] a pointer to character buffer to get the zhuyin fuhao list

SYNOPSIS

(1) save the 37 zhuyin fuhao into *pZhuin*

VARIABLES IN USE

N/A

RETURN VALUE

0:	*g_bReady* is False
length:	length of zhuyin fuhao list (in byte)

RELATED FUNCTIONS

GetJuinList

2.5.29 HAN_GetUserWordStartIndex

This function gets the start position regarding the specified zhuyin fuhao for finding user defined word in user dictionary. If can't find the user defined word start from specified zhuyin fuhao, it will find the user defined word start by next zhuyin fuhao till user word is found.

short HAN_GetUserWordStartIndex (PUWM *pUwm*, short *iZhuin*)

ARGUMENTS

pUwm

[IDLE] *pUwm* is not used in HAN_GetUserWordStartIndex

iZhuin

[IN] specified index of zhuyin fuhao

SYNOPSIS

(1) get the start position of user defined word regarding specified zhuyin fuhao

VARIABLES IN USE

N/A

RETURN VALUE

0:	*g_bInUWMaintenance* is False
position:	start position of user defined word in user dictionary regarding the specified zhuyin fuhao

RELATED FUNCTIONS

GetUWordIndex

2.5.30 HAN_GetUserWordCount

This function gets the current number of user defined words registered in user dictionary.

short HAN_GetUserWordCount (PUWM *pUwm*)

ARGUMENTS

pUwm

[IDLE] *pUwm* is not used in HAN_GetUserWordCount

SYNOPSIS

(1) return *sort_num*

VARIABLES IN USE

N/A

RETURN VALUE

0:	*g_bInUWMaintenance* is False
sort_num:	number of user defined words in user dictionary

RELATED FUNCTIONS

N/A

2.5.31 HAN_GetUserWord

This function gets the specified user defined word and its zhuyin fuhao.

short HAN_GetUserWord (PUWM *pUwm*, short *idx*, PUSERWORD *pUserWord*)

ARGUMENTS

pUwm

[IDLE] *pUwm* is not used in HAN_GetUserWord

idx

[IN] specified index of user defined word

pUserWord

[OUT] a pointer to data structure USERWORD to receive a user defined word and its zhuyin fuhao

SYNOPSIS

(1) duplicate the specified index of user defined word from *sort_list* into *pUserWord*

(2) convert compressed zhuyin code into zhuyin fuhao string

VARIABLES IN USE

sort_num

[IN] amount of user defined words

sort_list

[IN] data structure of SORTEDLIST to store the numbers and user word list

RETURN VALUE

0:	*g_bInUWMaintenance* is False or *idx* is invalid
strSize:	length of user defined word (in byte)

RELATED FUNCTIONS

cmp2zhuin

2.5.32 HAN_DelUserWord

This function removes the specified user defined word and its zhuyin fuhao from user dictionary.

short HAN_DelUserWord (PUWM *pUwm*, short *idx*)

ARGUMENTS

pUwm

[IDLE] *pUwm* is not used in HAN_DelUserWord

idx

[IN] specified index of user defined word

SYNOPSIS

(1) delete the specified index of user defined word from user dictionary

VARIABLES IN USE

N/A

RETURN VALUE

False:	*g_bInUWMaintenance* is False or *idx* is invalid
True:	success

RELATED FUNCTIONS

do_omit

2.5.33 HAN_GetFreeUserChars

This function can get the number of user defined words which is available to be registered into user dictionary. User can register at most 4,096 Chinese characters in HaninLib. Each Chinese word contains 2 to 5 Chinese characters.

---

short HAN_GetFreeUserChars (PUWM *pUwm*)

ARGUMENTS

pUwm

[IDLE] *pUwm* is not used in HAN_GetFreeUserChars

SYNOPSIS

(1) return the number of available user defined words in user dictionary

VARIABLES IN USE

N/A

RETURN VALUE

0:	*g_bInUWMaintenance* is False
num:	the number of available user defined words

RELATED FUNCTIONS

GetFreeWord

2.5.34 HAN_EndUWordMaintenance

This function frees the allocated memory by HAN_BeginUWordMaintenance and terminates the user defined word maintenance process.

void HAN_EndUWordMaintenance (PUWM *pUwm*)

ARGUMENTS

pUwm

[IDLE] *pUwm* is not used in HAN_EndUWordMaintenance

SYNOPSIS

(1) free allocated memory area and update the latest user defined word into user dictionay by calling SaveUerWord
(2) set *g_bInUWordMaintenance* is False

VARIABLES IN USE

g_bInUWMaintenance

[OUT] a flag to represent execution status of the maintenance process of user defined word

RETURN VALUE

void

RELATED FUNCTIONS

SaveUerWord

2.5.35 HAN_SetInputMode

This function set the input mode of HaninLib. The input mode can be changed when Hanin input method is active.

char HAN_SetInputMode (char *cInputMode*)

ARGUMENTS

cInputMode

[IN] input mode to be set. HaninLib supports 6 kinds of input mode as below.

STDKBD	zhuyin input using standard keyboard layout
IBMKBD	zhuyin input using IBM defined keyboard layout
ETKBD	zhuyin input using E-TEN defined keyboard layout
MITECKBD	zhuyin input using MiTAC defined keyboard layout
JUIN2	pinyin input using MPS2 romanization system
ROMAN	pinyin input using hanyu pinyin

SYNOPSIS

(1) set input mode according to previous input mode and display mode by calling SetInputMode

VARIABLES IN USE

N/A

RETURN VALUE

-1:	*g_bReady* is False
oldInputMode:	the previous input mode

RELATED FUNCTIONS

SetInputMode

2.5.36 HAN_SetDisplayMode

This function set the display mode in input buffer of HaninLib. If input mode is zhuyin input, the display mode can be only set to zhuyin display. If input mode is pinyin input, the display mode can be set to zhuyin display or alphabet display.

---

char HAN_SetDisplayMode (char *cDisplayMode*)

ARGUMENTS

cDisplayMode

[IN] display mode to be set.

SYNOPSIS

(1) set display mode according to the settings of *PininMode* and *cDisplayMode*

VARIABLES IN USE

N/A

RETURN VALUE

-1: *g_bReady* is False
oldDisplayMode: the previous display mode

RELATED FUNCTIONS

N/A

2.5.37 HAN_SetAddressState

This function set the status of using address dictionary. If address status is on, the Chinese conversion algorithm will mainly search address dictionary, otherwise, it will search system dictionary.

---

char HAN_SetAddressState (PHANIN *pHanin*, char *cAddresssState*)

ARGUMENTS

pHanin
[IN] a pointer to structure HANIN, it will be redirected to be the pointer of HaninLib working area
cAddressState
[IN] specify the address status

SYNOPSIS

(1) If it's different between current setting than previous setting, then initialize dictionary data in structure CORE

VARIABLES IN USE

N/A

RETURN VALUE

-1: *g_bReady* is False or *pHanin* is NULL
oldAddressState: the previous address state

RELATED FUNCTIONS

Decrypt, init_len_mark, Encrypt

2.5.38 HAN_SetInputState

This function set the input status of HaninLib. Hanin input method has 3 kinds of input status. Status IS_CHINESE is Chinese input status, status IS_PASS bypass the alphabet input, and status IS_FULL is to input 2-byte alphabet characters.

---

char HAN_SetInputState (PHANIN *pHanin*, char *cInputState*, PCONVRESULT *pConvResult*)

ARGUMENTS

pHanin

[IN] a pointer to structure HANIN, it will be redirected to be the pointer of HaninLib working area

cInputState

[IN] specify the input status

pConvResult

[OUT] a pointer to structure CONVRESULT, it's used to retrieve Chinese characters in input buffer for sending back to application

SYNOPSIS

(1) if *cInputState* is invalid then set *InputState* is IS_CHINESE, otherwise set *InputState* is *cInputState*

(2) retrieve the current Chinese conversion result form input buffer into *pConvResult*

VARIABLES IN USE

N/A

RETURN VALUE

-1:	*g_bReady* is False or *pHanin* is NULL
oldInputState:	the previous input state

RELATED FUNCTIONS

Decrypt, Encrypt, HAN_GetConvResult

3. Usage of HaninLib API

This section describes the usage of HaninLib APIs using a simple test tool to demonstrate how to let HaninLib work. Developers can follow the instructions and sample codes to implement a Chinese input method on their own working environment.

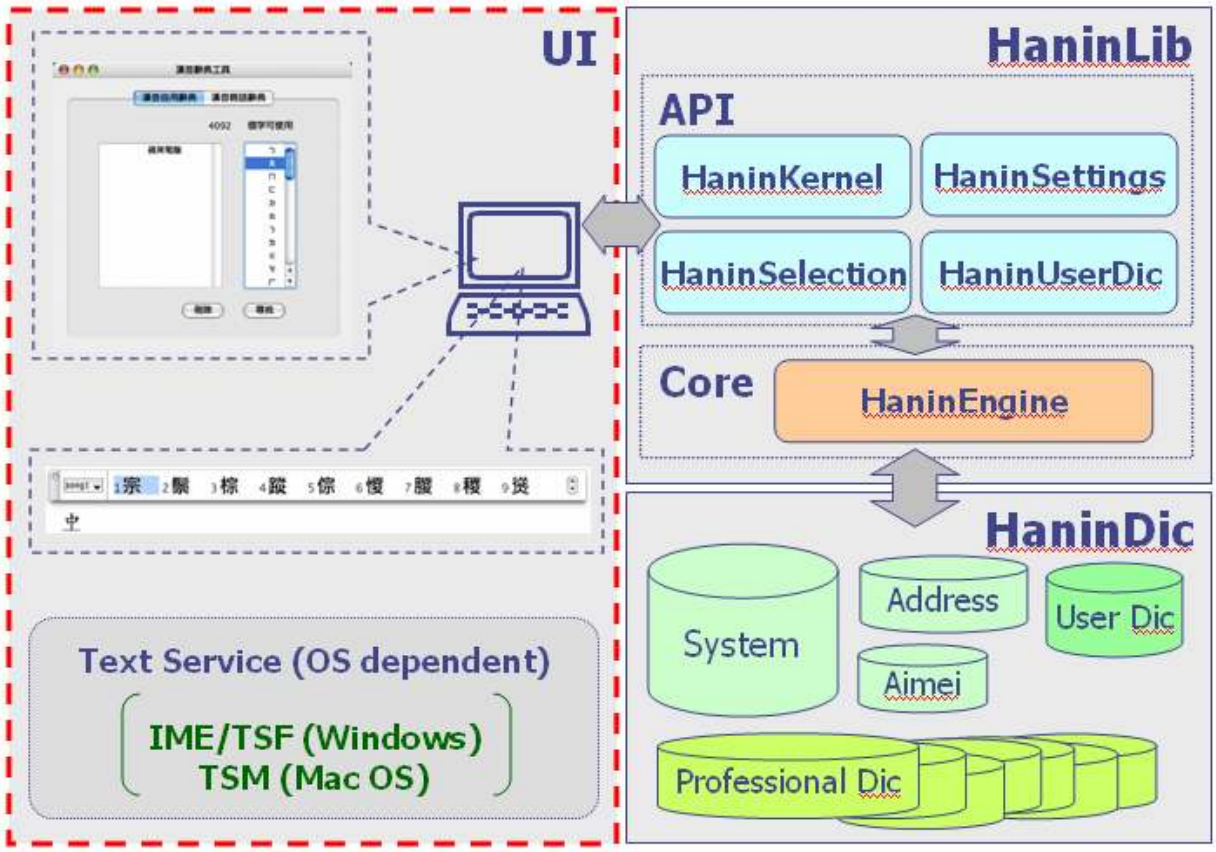


Figure 3: OS dependent portions in Hanin input system

When build a Chinese input method using HaninLib API, developers should implement at least 3 groups of functions which are configuration setting, user interface, and user dictionary maintenance. The red dashed rectangle in Figure 3 indicates the portions developers should implement on their own environment.

Function of configuration setting is to set the input conditions such as keyboard mapping for input key, display style of zhuyin fuhao, and default directory/filename of Hanin dictionaries. Function of user interface mainly provides the input/output mechanism for users to input characters from keyboard, touch panel, or soft keyboard, and see the Chinese conversion result on screen. Function of user dictionary maintenance is to remove those out-of-date user words. Figure 4-1, 4-2, 4-3 depict the connection between input functions and HaninLib API.

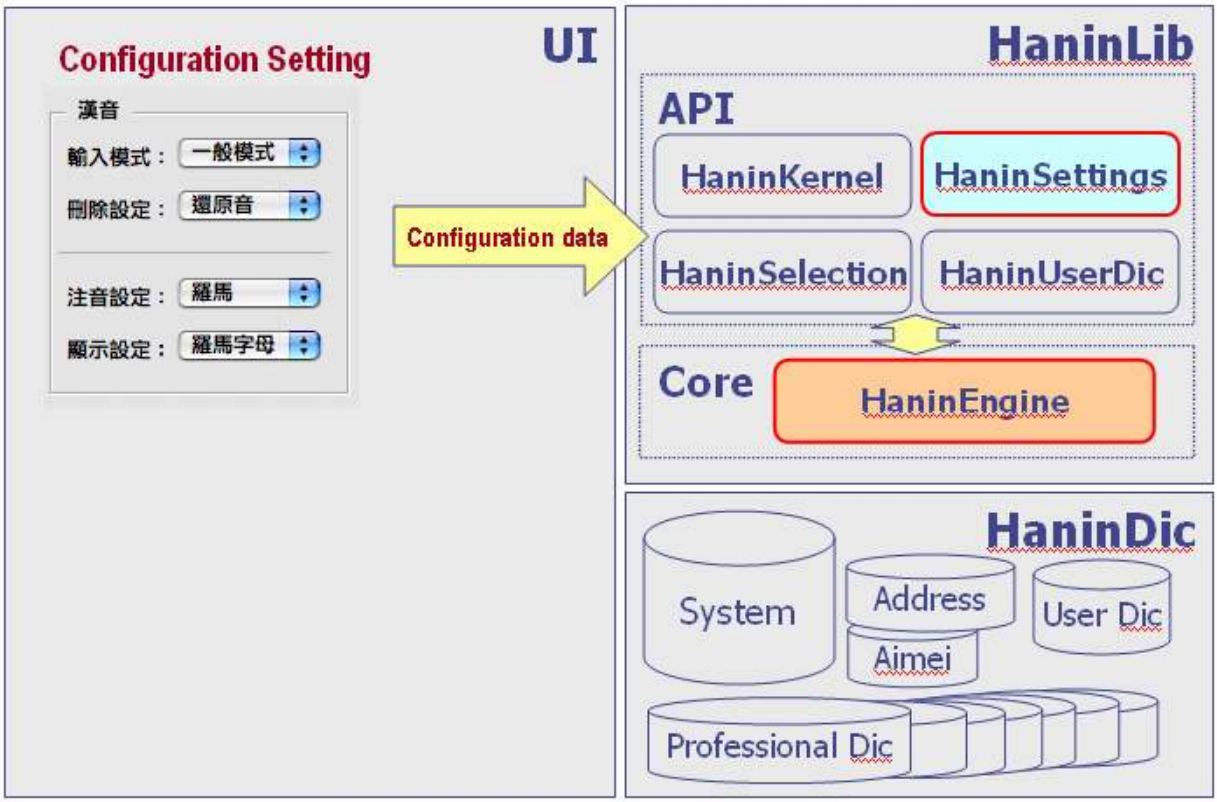


Figure 4-1: Block diagram of connection between configuration setting and HaninLib API

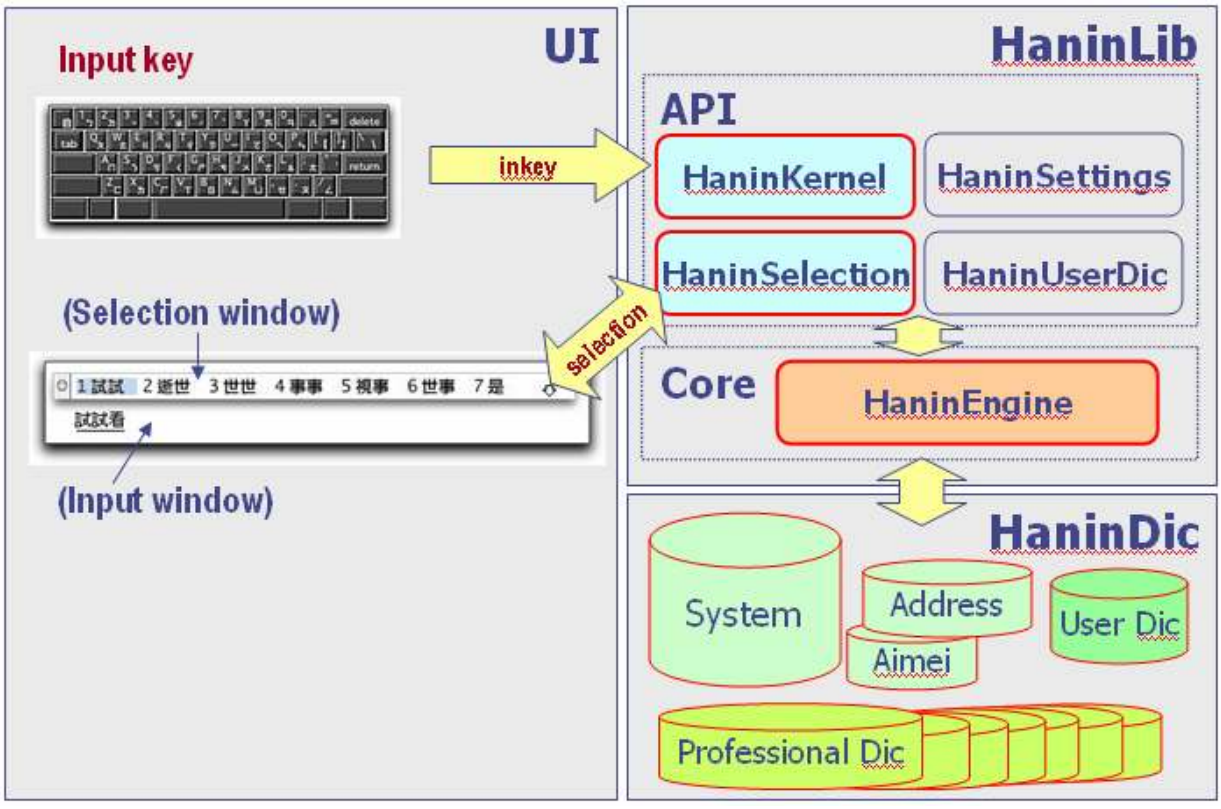


Figure 4-2: Block diagram of connection between user input and HaninLib API

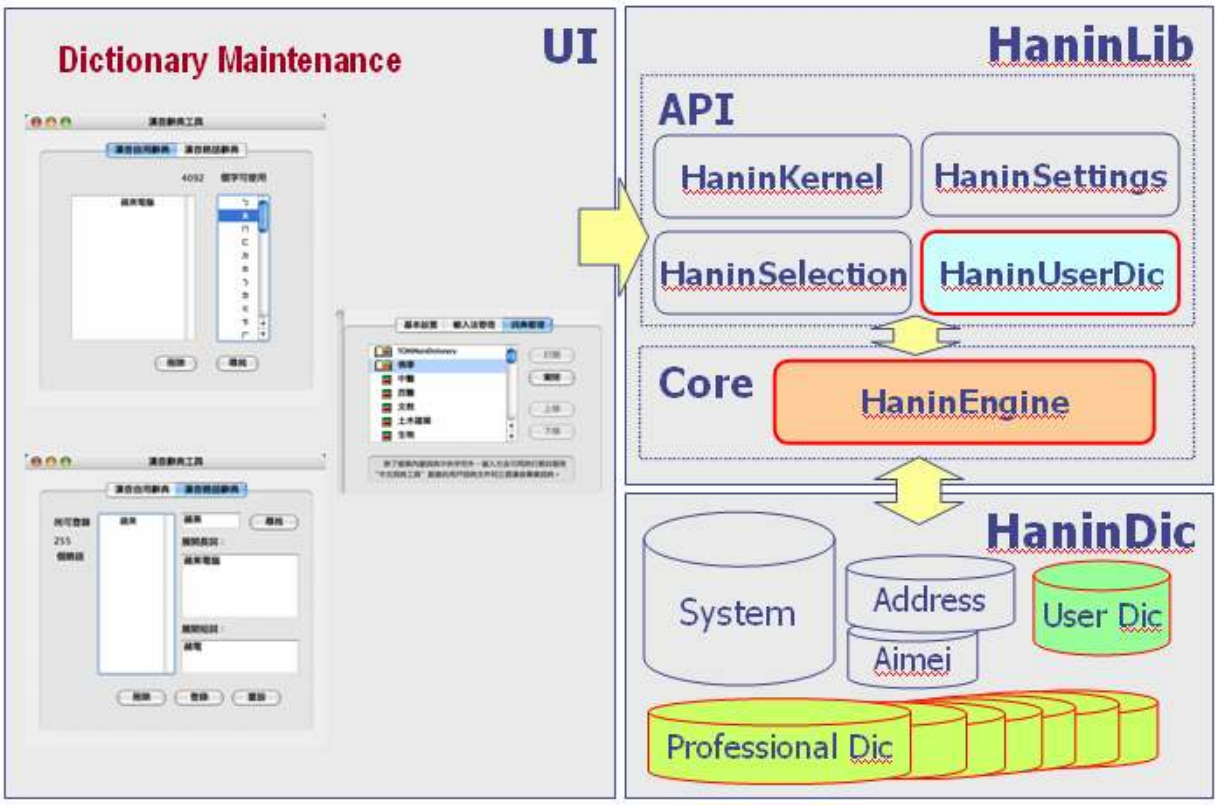


Figure 4-3: Block diagram of connection between dictionary maintenance and HaninLib API

3.1 Preparation

HaninLib is flexible to all the operation system on the market, therefore it has no key event interface and display functions. Developers should provide such input/output mechanism, and link with HaninLib to build a Chinese input method on his own environment.

The following conditions should be considered before using HaninLib.

Configuration Data

The configuration data means the input keyboard layout, output language of zhuyin fuhao, and directories of Hanin dictionaries. Due to HaninLib doesn't have the default configuration setting for configuration data, developer should prepare these configuration data based on his own environment before start using HaninLIB API (see Figure 4-1).

Input Key

Every working environment has characteristic mechanism to input keys. Whatever method user input a key, this key event should be transferred into Hanin's input key format before calling HaninLib API (see Figure 4-2).

Table 3, structure of INPUTKEY, shows the data structure of input key using in HaninLib. The ordinary input keys are mapping to zhuyin fuhao and then converted into Chinese character. The function key such as <Enter>, <Home>, <Tab>,…,etc. are necessary to be converted into predefined scancode symbols. HaninLib uses these scancode to conduct specific functions. The scancode is used in HaninLib can be referred in Table 1.

Necessary Memory

Basically, HaninLib has allocated sufficient memory area to execute the Hanin input method, it's not necessary to prepare extra memory area for HaninLib. However, for their

own needs, developers can design and prepare more memory area for building a Chinese input method.

3.2 Set Hanin Configuration

When start up the HaninLib, function HAN_Start will be called at first. HAN_Start executes the initialization process of HaninLib that initializes the global structure data by referring input HANININIT structure (see Table 2). This means developers should specify the HANININIT structure data before calling the function HAN_Start.

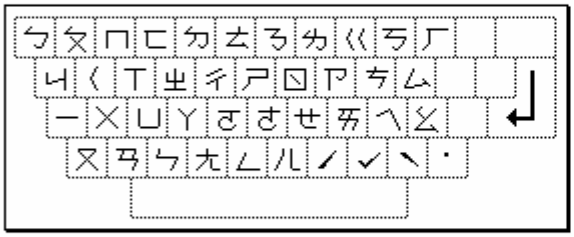
HANININIT structure includes 2 main configuration data, dictionary directory and input conditions. The dictionary directory should be full path and dictionary filename for each Hanin dictionary. Suppose developers don't give the dictionary directory, the function HAN_Start can still work normally without error, however, the Chinese conversion can't be completed due to HaninLib can't find any dictionary.

The input conditions include Input mode, Address mode and Display mode. The Address mode is for the reference order of system dictionary and address dictionary. The setting of Display mode will determine the returned string of zhuyin fuhao is bopomofo or alphabet. But, if the Input mode is zhuyin input (standard keyboard, IBM keyboard,..), the Display mode can't be set to alphabet. HaninLib will check the setting of Input mode and Display mode, and correct the setting of Display mode if necessary. Figure 5 shows the Input mode.

Zhuyin (standard keyboard)

Pinyin

Zhuyin (IBM keyboard)



MPS II

Zhuyin (E-Ten keyboard)

Figure 5: Sample of Input Mode

The sample code segment to set configuration data is as below.

```
#include "Haninif.h"

PHANIN pWork ;   /* pointer to Hanin working area */

void OnInitHaninLIB ( void )
{
    HANININIT stInitConf ;

    /* initialize Hanin configuration */
    stInitConf.InputMode = STDKBD ;
    stInitConf.DisplayMode = DISPLAY_ZHUIN ;
    stInitConf.AddressState = OFF ;
    /* dictionary directory setting */
    strcpy(stInitConf.hanin_dic, "C:\\HANIN\\HaninDic") ;
    strcpy(stInitConf.add_dic, "C:\\HANIN\\AdrDic") ;
    strcpy(stInitConf.aimei_dic, "C:\\HANIN\\AimeiDic") ;
    strcpy(stInitConf.user_dic,
            "C:\\HANIN\\USR_FILE\\$USR000.DIC") ;
    strcpy(stInitConf.big5e_dic, "C:\\HANIN\\Big5EDic" ) ;

    bRet = HAN_Start ( &stInitConf ) ;

    pWork = HAN_Open () ;
}
```

3.3 Input Chinese Characters

In order to input Chinese characters through continuously pressing keys, developers should design a procedure to deal with the key event on their own environment. The pressed key code should be changed into HaninLib input key format to convert a sequence of keys into Chinese character. This section gives a sample code to show how to conduct the Chinese character/word conversion using HaninLib. Figure 6 shows a sample appearance when input "松下電器".

```
#include "Haninif.h"

void OnKeyDown ( int nChar, int scancode, int KBstate )
{
    INPUTKEY stKey ;
    CONVRESULT stResult ;
    char outBuff[MAX_BUFFSIZE] ;     /* for display result */

    stKey.inkey = (unsigned char)nChar ;
    stKey.scancode = (unsigned char)scancode ;
    stKey.KB_Caps = (char)(KBstate & KB_CAPITAL ) ;
    stKey.KB_Shift = (char)(KBstate & KB_SHIFT ) ;

    if (HAN_InputKey (pWork, &stKey, &stResult) == False) {
        return ;
    }
    strcpy(outBuff, (char *)stResult.szEdit) ;
```

```
    /* display conversion result on screen */
    OutputResult (outBuff) ;
}
```

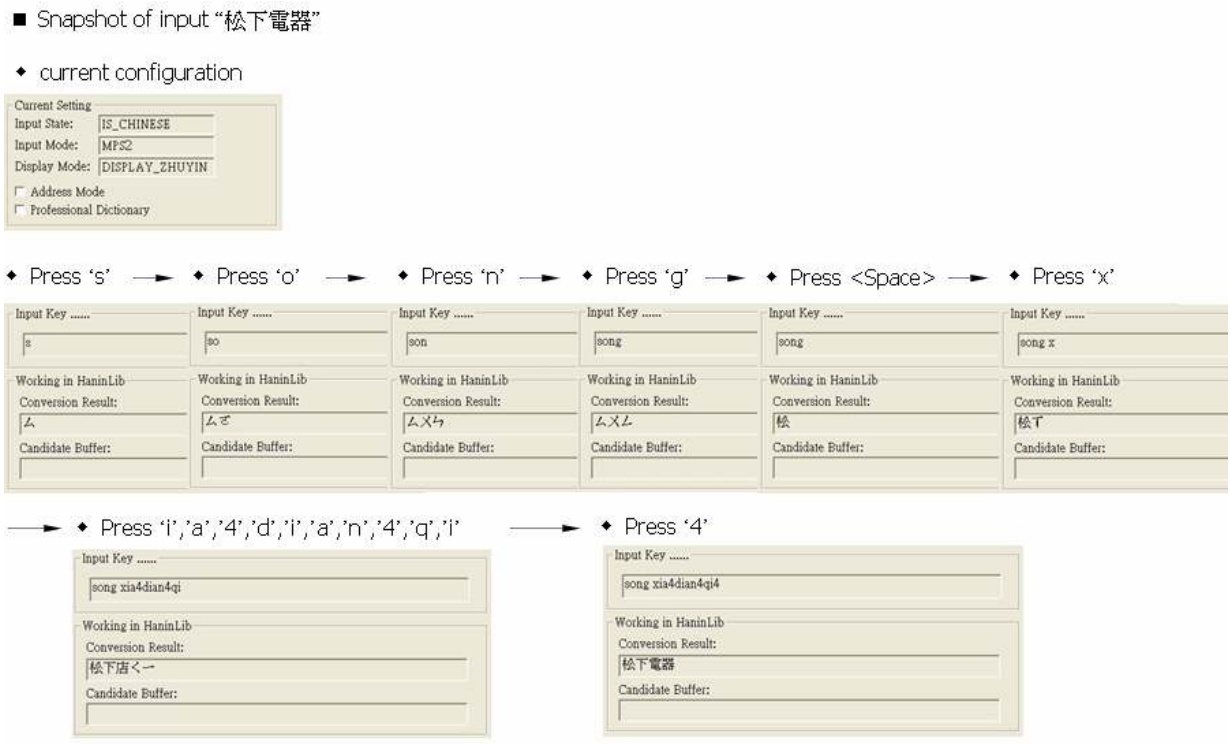


Figure 6: Sample of input Chinese word

3.4 Select Same Pronounced Characters

When the process of sequent key code converted into Chinese character or word, the same pronounced characters or words can be selected. HaninLib provides functions to retrieve same pronounced Chinese characters or words as the specified characters/words. Developers should implement a select window to allow users choose the character/word they want. Figure 7 shows a sample appearance for selection of same pronounced character/word.

```
#include "Haninif.h"

void OnSelection ()
{
    unsigned char ucCands[CANDSIZE] ;
    short sCount ;
    short sSelIdx, sPos ;
    CONVRESULT stResult ;

    if (HAN_IsAllConverted(pWork) == False) {
        return ;
    }
    switch (SELECTION_TYPE) {
    case CAND_WORD:
        sCount = HAN_GetSamePronWords(pWork, ucCands,
                                      CANDSIZE) ;
        break ;
    case CAND_CHAR:
```

```
            sCount = HAN_GetSamePronChars(pWork, ucCnads,
                                          CANDSIZE, NO ) ;
            break ;
        case CAND_SYMB:
            sCount = HAN_GetSamePronChars(pWork, ucCnads,
                                          CANDSIZE, YES ) ;
            break ;
        }
        /* choose the 1st candidate */
        sSelIdx = 1 ;
        switch (SELECTION_TYPE) {
        case CAND_WORD:
            sPos = (sSelIdx-1)*4 ;
            HAN_SelectSamePronWord(pWork, ucCands+sPos,
                                   &stResult) ;
            break ;
        case CAND_CHAR:
        case CAND_SYMB:
            sPos = (sSelIdx-1)*2 ;
            HAN_SelectSamePronChar(pWork, ucCands+sPos,
                                   &stResult) ;
            break ;
        }
        /* display conversion result on screen */
        OutputResult((char *)stResult.szEdit) ;
}
```

■ Snapshot of character/word selection

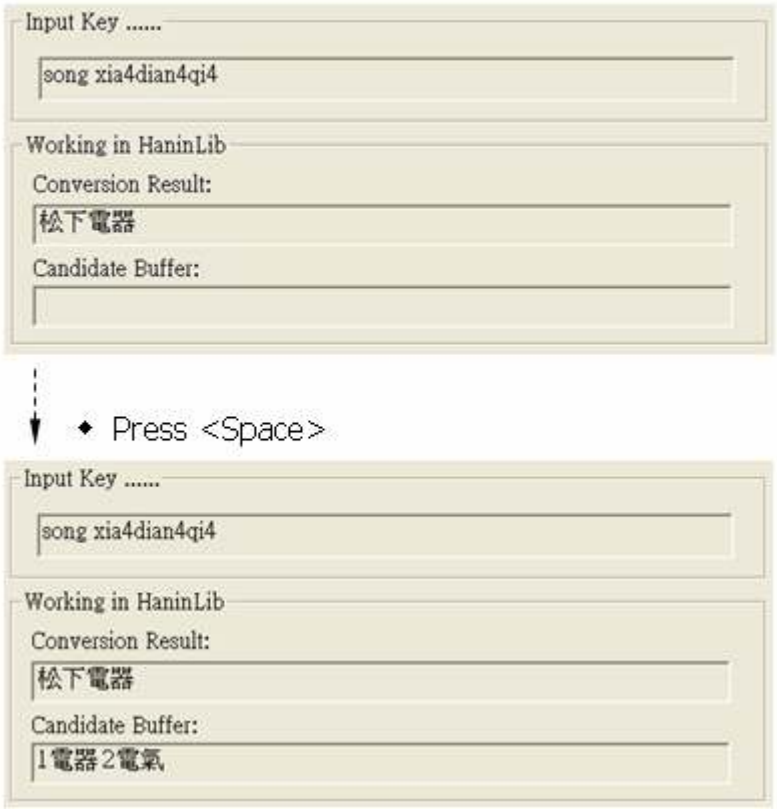


Figure 7: Sample of selection window

3.5 Register User Word

In order to improve the Chinese conversion performance, HaninLib provides a function to allow users register their own words or phrases during input characters using HaninLib. Users move the cursor to the head of text in Hanin edit window and then press a predefined hot key to add a new user word into user dictionary. The following code segment is an example to register a user defined word.

```
#include "Haninif.h"

void OnRegisterWord ()
{
     CONVRESULT stResult ;

     HAN_GetConvResult(pWork, &stResult) ;
     HAN_RegUserWord(pWork, stConvResult.EditLength) ;
}
```

3.6 Aimei Function

Aimei function allows users to select Chinese characters or words which pronounced similar to their input. The operation of aimei function is similar to selection of same pronounced characters/words, except it needs to load an extra aimei dictionary in the initialized process. After selection of aimei character/word, it's necessary to terminate the aimei function in order to free an allocated memory area for aimei structure in the aimei startup function. The following code segment is an example to select aimei words/characters, and

Figure 8 shows the screen snap.

```
#include "Haninif.h"

void OnSelAimeiWord ()
{
    PAIMEI pAimei ;
    CONVRESULT stResult ;
    short sCount, sYinIdx, sSelIdx ;
    unsigned char ucCand[BUFFSIZE] ;
    unsigned char ucZhuyin[YINSIZE] ;
    short sRetLen ;

    if (pAimei = HAN_BeginGetAimei(pWork) == NULL) {
        return ;    /* can't execute aimei function */
    }
    if (sCount = HAN_GetAimeiCount(pWork, pAimei) == 0) {
        return ;    /* no aimei yin */
    }
    sYinIdx = 0 ;   /* the 1st aimei yin (count from 0) */
    if (AIMEITYPE == ISWORD) {
        sRetLen = HAN_GetAimeiWords(pWork, pAimei, sYinIdx,
                    ucCand, BUFFSIZE, ucZhuyin, &sZhuyinLen) ;
    }
    else {
        sRetLen = HAN_GetAimeiChars(pWork, pAimei, sYinIdx,
                    ucCand, BUFFSIZE, ucZhuyin, &sZhuyinLen) ;
    }
    /* display aimei yin and words */
    DisplayResult(ucZhuyin, ucCand) ;

    /* choose the 1st candidate */
    sSelIdx = 1 ;
    if (AIMEITYPE == ISWORD) {
        HAN_SelectAimeiWord(pWork, pAimei, sSelIdx,
                ucCnad+(sSelIdx-1)*4, &stResult) ;
    }
    else {
        HAN_SelectAimeiChar(pWork, pAimei, sSelIdx,
                ucCnad+(sSelIdx-1)*4, &stResult) ;
    }
    /* display selection result */
    OutputResult((char *)stResult.szEdit) ;

    /* terminate aimei function */
    HAN_EndGetAimei(pWork, pAimei) ;
}
```

Figure 8: Sample of aimei function

3.7 Abbreviation Operation

HaninLib provides abbreviation function to speed up Chinese input by replacing base test into short/long form text in Hanin edit window. After input base text, users press a predefined hot key to replace base text into short form text, press hot key again to replace into long form text. Press hot key again and the base text will reappear. Developers can decide the replace order of short form text and long form text according their own design. The following code segment to illustrate the usage of abbreviation function. Figure 9 shows a example of abbreviation function.

```
#include "Haninif.h"

void OnExtension ()
{
```

```
    EXTENSION stAbbr ;
    unsigned char ucAbbrBuff[ABBRSIZE] ;

    if (HAN_GetExtension(pWork, &stAbbr) == False) {
        return ;    /* fail to find abbreviation */
    }
    if (ABBRTYPE == ABBR_SHORT) {
        strcpy(ucAbbrBuff, stAbbr.szLong) ;
        ABBRTYPE = ABBR_LONG ;
    }
    else {
        strcpy(ucAbbrBuff, stAbbr.szShort) ;
        ABBRTYPE = ABBR_SHORT ;
    }
    /* display abbreviation */
    OutputResult((char *)ucAbbrBuff) ;
}
```

■ Snapshot of abbreviation function

Figure 9: Sample of abbreviation function

3.8 User Words Maintenance

HaninLib provides 2 groups of functions to maintain sets of alternative abbreviations and inventory of user words in user dictionary. Developers design the user interface to conduct the operation of maintenance of user dictionary by means of these functions. The following code segment is a sample to list the registered user words in user dictionary, and Figure 10 shows the screen snap of maintenance interface.

```
#include "Haninif.h"

void OnUDmaintenance ()
{
    PUWM pUD ;
    USERWORD stUword ;
    short sUDcount, sUDwlen, i ;

    if (pUD=HAN_BeginUWordMaintenance() == NULL) {
        return ;
    }
    sUDcount = HAN_GetUserWordCount(pUD) ;
    for (i=0; i < <sUDcount; i++) {
        sUDwlen=HAN_GetUserWord(pUD, i, &stUword) ;
        strncpy((char *)ucZhuyin, (char *)stUword.szZhuin,
                strlen((char *) stUword.szZhuin)) ;
        strncpy((char *)ucUDword, (char *)stUword.szUserWord,
                sUDwlen) ;
        /* display user word */
        DisplayUserWord(ucZhuyin, ucUDword) ;
    }
    /* end of user word maintenance */
    HAN_EndUWordMaintenance(pUD) ;
}
```

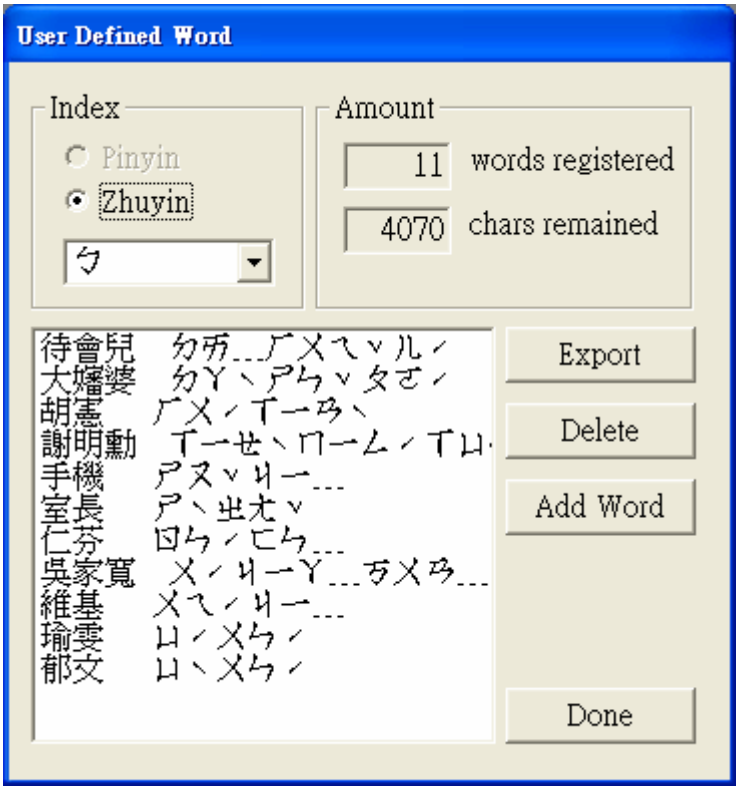


Figure 10: Screen snap of user dictionary maintenance

Regarding maintenance of alternative abbreviations, developers design their own user interface and add/delete/find abbreviation using HAN_RegExtension, HAN_DelExtension, and HAN_FindExtension functions.

Annex A. Table of Mandarin Phonetic Symbols II

On January 28, 1986, the Ministry of Education, upon completion of all the relevant revisions after the trial use period, formally announced the following, which includes the Table of comparison of MPS Ⅰ and MPS Ⅱ and a set of twelve rules for the use of MPS Ⅱ for the reference of all local and oversea users.

[Table of Mandarin Phonetic Symbols Ⅱ]

(1) Initials				
Labials	b	p	m	f
Apicals	d	t	n	l
Velars	g	k	h	
Palatals	j(i)	ch(i)	sh(i)	
Retroflexes	j	ch	sh	r
Dentals	tz	ts	s	
(2) Finals				
Single Finals	r, z			
	i, yi	u, wu	iu, yu	
	a	o	e	ê
Double Finals	ai	ei	au	ou
Nasal-end Finals	an	en	ang	eng
Retroflex Final	er			
(3) Compound Finals (Used with Initials)				
With i on-glide	ia	io	ie	
	iai	iau	iou	
	ian	in	iang	ing
With u on-glide	us	uo	uai	uei
	uan	uen	uang	ung
With iu on-glide	iue	iuan	iun	iung
(4) Tones				
First Tone	¯			
Second Tone	ˊ			
Third Tone	ˇ			
Fourth Tone	ˋ			
Neutral Tone	(unmarked)			

Note:1. In the Final series, the single vowels r and z refer to apical vowels; i, yi, u, wu, iu, yu are the high vowels that can function as glides and a, o, e, ê are vowels without glides.

2. For the use of the Compound Finals without initials, see Transliteration Rule 7.

[Table of Comparison: MPS I vs. MPS Ⅱ]

	MPS I	MPS II
(1) Initials		
Labials	ㄅ ㄆ ㄇ ㄈ	b p m f
Dentals	ㄉ ㄊ ㄋ ㄌ	d t n l
Velars	ㄍ ㄎ ㄏ	g k h
Palatals	ㄐ ㄑ ㄒ	j(i) ch(i) sh(i)
Retroflexes	ㄓ ㄔ ㄕ ㄖ	j ch sh r
Dentals	ㄗ ㄘ ㄙ	tz ts s
(2) Finals		
Single Final	(ㆠ)	r,z
	ㄧ ㄨ ㄩ	i,yi u,wu iu,yu
	ㄚ ㄛ ㄜ ㄝ	a o e ê
Double Finals	ㄞ ㄟ ㄠ ㄡ	ai ei au ou
Nasal-end Finals	ㄢ ㄣ ㄤ ㄥ	an en ang eng
Retroflex Final	ㄦ	er
(3) Compound Finals (Used with Initials)		
With i on-glide	ㄧㄚ ㄧㄛ ㄧㄝ ㄧㄞ	ia io ie iai
	ㄧㄠ ㄧㄡ ㄧㄢ ㄧㄣ	iau iou ian in
	ㄧㄤ ㄧㄥ	iang ing
With u on-glide	ㄨㄚ ㄨㄛ ㄨㄞ ㄨㄟ	ua uo uai uei
	ㄨㄢ ㄨㄣ ㄨㄤ ㄨㄥ	uan uen uang ung
With iu on-glide	ㄩㄝ ㄩㄢ ㄩㄣ ㄩㄥ	iue iuan iun iung
(4) Tones		
First Tone	(unmarked)	¯
Second Tone	ˊ	ˊ
Third Tone	ˇ	ˇ
Fourth Tone	ˋ	ˋ
Neutral Tone	·	(unmarked)

Annex B. Pinyin Table Used in HaninLib

聲母		韻母		結合韻母		聲調	
ㄅ	b	ㄧ	i, y	ㄧㄚ	ia, ya	一聲	空白, 1, 6
ㄆ	p	ㄨ	u, w	ㄧㄛ	io, yo	二聲	2, 7
ㄇ	m	ㄩ	yu, v	ㄧㄝ	ie, ye	三聲	3, 8
ㄈ	f	ㄚ	a	ㄧㄞ	iai, yai	四聲	4, 9
ㄉ	d	ㄛ	o	ㄧㄠ	iao, yao	輕聲	5, 0
ㄊ	t	ㄜ	e	ㄧㄡ	iu, you		
ㄋ	n	ㄝ	ê	ㄧㄢ	ian, yan		
ㄌ	l	ㄞ	ai	ㄧㄣ	in, yin		
ㄍ	g	ㄟ	ei	ㄧㄤ	iang, yang		
ㄎ	k	ㄠ	ao	ㄧㄥ	ing, ying		
ㄏ	h	ㄡ	ou	ㄨㄚ	ua, wa		
ㄐ	j	ㄢ	an	ㄨㄛ	uo, wo		
ㄑ	q	ㄣ	en	ㄨㄞ	uai, wai		
ㄒ	x	ㄤ	ang	ㄨㄟ	ui, uei, wei		
ㄓ	zh	ㄥ	eng	ㄨㄢ	uan, wan		
ㄔ	ch	ㄦ	er	ㄨㄣ	un, wen		
ㄕ	sh			ㄨㄤ	uang, wang		
ㄖ	r			ㄨㄥ	ong, ueng		
ㄗ	z			ㄩㄝ	ve, yue		
ㄘ	c			ㄩㄢ	van, yuan		
ㄙ	s			ㄩㄣ	vn, ven, yun		
				ㄩㄥ	yong, veng		