Panasonic
ideas for life

PS Design

Mac Hanin Software
HaninLib Modules
Specification

Version: 0.02

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
2008/02/01	0.01	-	-	First edition	Wu Jia-Kwan
2008/02/04	0.02	D		remove the ISM stamp on cover page	Wu Jia-Kwan

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

[Table of Contents]

1. Introduction

1.1 Abstract

The HaninLib, which is the core of Hanin input method, consists of 2 main portions: API and Core. HaninLib API is a collection of user interface for developers to utilize the functionality provided by Hanin. HaninLib Core is the core engine of Hanin. It conducts algorithms of Chinese word/character conversion, retrieves data from high compressed ratio of Hanin dictionaries, and maintain the user/learning dictionary.

This document mainly describes the core functions and global variables in HaninLib Core. Also, the calling sequence of HaninLib API is detailed described in this document. Regarding HaninLib API introduction, please refer to [1] for details.

1.2 Definition of Terms

[1]	Zhuyin Fuhao	A phonetic system for transcribing Chinese
[2]	Measure Word	(or Classifier) is used along with numerals to define the quantity of given object
[3]	Big5E	Big5 Extension Character Set

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] Wu Jia-Kwan, "HaninLib API Specification and Usage", Technical Report PTL9641, Panasonic Taiwan Laboratories Co., Ltd., Dec 17, 2007

[2] Liu Guei-Ji, "Hanin5 Library release version 1", Source Program MM92026, Panasonic Taiwan Laboratories Co., Ltd., Jul 29, 2003

2. HaninLib Core Overview

2.1 HaninLib Core Architecture

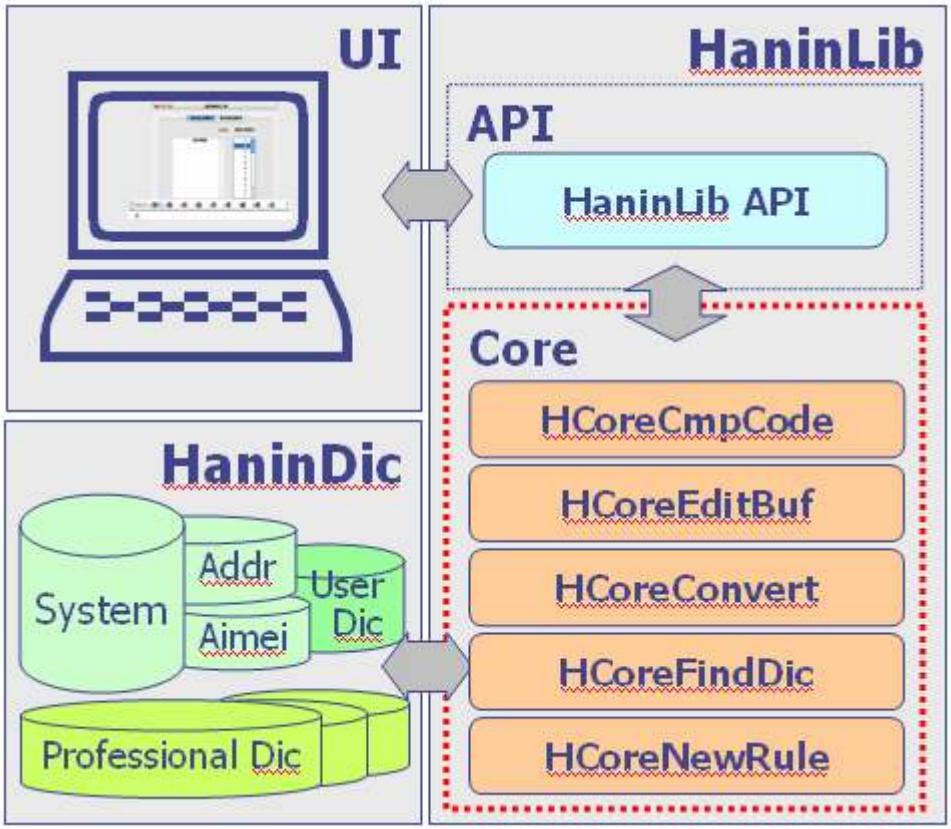

Figure 1: Overview of Hanin input system

Figure 1 depicts a overview of Hanin input system. Mainly, the red dashed rectangle highlights the role of HaninLib Core and its components. HaninLib Core consists of 5 components: HCoreCmpCode, HCoreEditBuf, HCoreConvert, HCoreFindDic, and HCoreNewRule. HCoreCmpCode converts continuous key symbols into Hanin individual compressed code and recovers zhuyin fuhao or pinyin alphabet from compressed code. HCoreEditBuf deals with the text code and its attribute of characters/words in Hanin edit buffer. HCoreConvert executes Hanin particular patent algorithms to do the Chinese conversion. HCoreFindDic deals with Hanin dictionaries which includes system dictionary, address dictionary, professional dictionaries, and user dictionary. HCoreNewRule conducts several special conversion rules for quantifier, measure word, surname to improve the Chinese conversion ratio.

Table 1: Files related to 5 components of HaninCore.

Component of HaninLib Core	Filename
HCoreCmpCode	chuin.c cmp2pin.c

	compress.c pinin.c
HCoreEditBuf	esc.c os2api.c real.c scren.c show_pin.c up_down.c
HCoreConvert	match.c
HCoreFindDic	allget.c big5e.c choice.c ld_fun.c md_fun.c ud_fun.c
HCoreNewRule	rule.c unit.c

2.2 File Description

Here is a list of all documented files with brief description.

Table 2: List of HaninLib Core files

Filename	Description
allget.c	retrieve all the homophone characters and words from Hanin dictionary
big5e.c	load/get BIG5+ characters
choice.c	load/find aimei characters/words, convert compressed code into Zhuyin fuhao, get all the homophone characters/words, choose a specified character/word
chuin.c	convert continuous keys into string of zhuyin fuhao
cmp2pin.c	convert compresses code into pinyin alphabet
compress.c	compress zhuyin fuhao into Hanin individual compressed code, recover compressed code to zhuyin fuhao
esc.c	automatically send a character in edit buffer to the caller application
hanin.h	declaration of definitions, data structures, global variables, and functions
juitable.h	declare and initialize 4 kinds of zhuyin mapping tables to represent compressed code for each key button
ld_fun.c	operations regarding learning dictionary
match.c	conduct Hanin algorithms for Chinese conversion
md_fun.c	search characters and words in Hanin dictionaries, deal with repetition of words

os2api.c	operation of edit buffer, load/check Hanin system dictionary, professional dictionaries, learning dictionary
pinin.c	convert continuous key symbols into Hanin specified compressed code
real.c	entrance of Hanin Core
rule.c	perform rules for quantifiers, surnames, and specified rules
scren.c	delete or backspace a character in edit buffer
show_pin.c	display zhuyin fuhao srting in edit buffer
ud_fun.c	maintain Hanin user dictionary
unit.c	deal with rules of measure words
up_down.c	initialize the structure Core, remove the identical character/word with edit buffer from selection buffer

2.3 List of HaninLib Modules

Here is a list of all the HaninLib modules and module IDs.

Table 3: List of Modules

Filename	Module ID	Module Name	Description
HaninLib Core component: **HCoreCmpCode**			
chuin.c	0001	ZhuinMatch	convert input key code into zhuyin fuhao
cmp2pin.c	0002	cmp2pinin	convert compressed code to pinyin alphabet string
	0003	append	
compress.c	0004	CompressZhuin	convert zhuyin fuhao to compressed code
	0005	RecoverZhuin	recover zhuyin fuhao from compressed code
pinin.c	0006	PininMatch	convert input keycode to pinyin alphabet string
	0007	PininDelete	delete/backspace a pinyin symbol
	0008	Code2Level	get the level of zhuyin code
HaninLib Core component: **HCoreEditBuf**			
esc.c	1001	scroll	send back the first character in edit buffer to caller application
	1002	fillback	no operation for Big5 charset
os2api.c	1003	Clr_line	clear edit buffer
	1004	bell	no source code
	1005	words_dis	insert length of characters to display buffer
	1006	put_ch	insert a character to display buffer
	1007	convert_attr	convert display attribute symbol to pseudo code
	1008	CheckDictionary	load and check all the Hanin dictionaries
	1009	MDLoad	load system or address dictionary and

			retrieve the IDX1 table
	1010	ProfDicLoad	load professional dictionary and retrieve the IDX1 table
	1011	ckUDLoad	load and check learning words and abbreviations retrieved from user dictionary
	1012	LDCheck	check the availability of learning dictionary
	1013	LDCreate	initialize the structure of learning word
real.c	1014	realtime	entrance function of HaninLib Core
	1015	others	deal with normal input keys
	1016	UnusedKey	check current input symbol key is used regarding current input mode
	1017	init_vars	initialize the global variables
	1018	perform	send back the characters in edit buffer to caller application
	1019	initial	initialize the global variables and edit buffer
	1020	enter	process when pressed “Enter” key
	1021	put_back	sent character in edit buffer to caller application
	1022	modify	process when pressed function button to recover zhuyin fuhao
	1023	recover	recover the change of word by Hanin algorithms
	1024	keypad	check current input key is keypad key
	1025	chg_key	change capital letter into small letter, and vice versa
scren.c	1026	Handelete	delete a character on cursor position in edit buffer
	1027	backsp	process when pressed “Backspace” key
	1028	shift_cursor	move cursor in edit buffer
show_pin.c	1029	show_pinin	display zhuyin fuhao in edit buffer
up_down.c	1030	big5cmp	compare current input character code with specified characters
	1031	init_len_mark	initialize the length mark in structure CORE
	1032	AppendToBuffer	append specified length of specified characters into *chbuf*
HaninLib Core component: **HCoreConvert**			
match.c	2001	match	perform Hanin patent algorithms
	2002	SymbolMatch	check of input symbol

	2003	rshift_data	shift right one record of data
	2004	lshift_data	shift left one record of data
HaninLib Core component: **HCoreFindDic**			
allget.c	3001	SearchAllChars	search all homophone characters
	3002	SearchAllWords	search all homophone words
big5e.c	3003	ReadBig5E	retrieve Big5E characters from Big5E dictionary
	3004	GetBig5EChar	get Big5E characters from candidate buffer
	3005	AddACmp	convert specified zhuyin fuhao into compressed code and insert it to structure of index of Big5E characters
	3006	CompareCmp	compare 2 compressed code buffers
choice.c	3007	sear_aimei	load aimei pronounced data from aimei dictionary
	3008	cmp2zhuin	recover zhuyin fuhao from compressed code
	3009	GetSameProns	retrieve all the same pronounced characters or words
	3010	SelectSamePron	select one same pronounced word
	3011	SelectAimeiPron	select one aimei word
ld_fun.c	3012	LDWrite	write learning words from memory of structure LDRec into user dictionary
	3013	LDSearchWord	search learning word in memory of structure LDRec
	3014	LDInsertWord	insert a learning word into memory of structure LDRec
	3015	LDMoveWord	move learning word to top of memory of structure LDRec
	3016	InsertLearnCode	insert a new learning word into learning dictionary
md_fun.c	3017	SearchOneWord	search a word
	3018	SearchOneChar	search a character
	3019	MDSearchFChar	search the first 5 characters from specified dictionary
	3020	MDSearchChar	search characters in system dictionary or professional dictionaries
	3021	MDSearchWord	search words in system dictionary, professional dictionaries
	3022	MDGetCharAddress	get character address in dictionary by the 1st

			and 2nd compressed codes
	3023	MDGetWordAddress	get word address in dictionary by the 3rd compressed code
	3024	BinarySearch	binary search for finding an item in table
	3025	CompareCmpcode	compare 2 compressed codes
	3026	ReadBuffer	read data from specified file
	3027	RepSearchWord	conduct a rule of repeated character
	3028	two_invert	swap the 1st and 2nd entries of char buffer
ud_fun.c	3029	UDGetHashIndex	get hash index according to compressed code
	3030	UDSearchWord	search words in user dictionary
	3031	UDInsertWord	insert a word into user dictionary
	3032	UDWrite	write the specified user defined words into user dictionary
	3033	q_extend	retrieves long form text and short form text according input base text
	3034	UDLoad	load user defined words from user dictionary
	3035	hash_to_sort	copy user defined words in hash buffer into sorted list
	3036	insert_list	insert a new defined word to memory of structure SORTEDLIST by ascending order
	3037	MK2_Word	assign the sorted records to hash buffer
	3038	GetUWordIndex	get the nearest index to input zhuyin fuhao
	3039	SaveUserWord	write user defined words and index into user dictionary
	3040	GetFreeWords	count the number of available rooms for user defined words
	3041	do_omit	omit the specified record in memory of structure SORTEDLIST
	3042	GetJuinList	give the zhuyin list
	3043	strcmp2	compare 2 strings
	3044	DelExt	remove the specified abbreviation from user dictionary
	3045	RegExt	register the specified abbreviation into user dictionary
	3046	FindExtID	get the index of specified abbreviation
	3047	GetExt	retrieve short form text and long form text of specified abbreviation

	3048	GetExtVacancy	get the first index for registering a new abbreviation
HaninLib Core component: **HCoreNewRule**			
rule.c	4001	SpecialRuleMatch	conduct special conversion rules
	4002	MatchRuleA	conduct 2 special rules for matching unit, adverb regarding specific noun
	4003	MatchRuleB	conduct a rule for matching specific verb and noun
	4004	MatchRuleC	conduct a special rule for matching title and surname
unit.c	4005	UnitSearchWord	conduct algorithms of special rules
	4006	fdan	find the same compressed code in records for special rule
	4007	fndx	retrieve matched adverb character and its frequency
	4008	chk_unit	check the status of special rule for measure words
	4009	Spatch	execute special rules of adverb, measure words,etc
	4010	ReDisplay	update edit buffer
	4011	do_rep	conduct repeated Chinese character algorithm
	4012	AddrCharMatch	revise matching character when address mode is ON

2.4 Data Structure

This section introduces the data structures used in HaninLib Core. Most of them are the same structures that used in HaninLib API.

Table 4: Structure of LDRec

Member	Type	Description	Size (byte)
last	unsigned char	index of previous learning word record	1
next	unsigned char	index of next learning word record	1
cmpCode	unsigned char [4]	compressed code of current learning word	4
bigCode	unsigned char [4]	Big5 code of current learning word	4
Total Size			10

[Synopsis of LDRec]

The structure LDRec is for maintaining records of learning words, up to 100 records. The head of learning word records is the last record of LDRec array allocated for learning words. The contents of LDRec array will be updated according to input Chinese character and word when HaninLib is active.

Table 5: Structure of IDX1

Member	Type	Description	Size (byte)
schar	char	compressed code (not used in current version of HaninLib)	1
charoff2	unsigned char	low byte of character position in Hanin dictionary	1
idxoff2	unsigned char	low byte of index in Hanin dictionary	1
charoff1	unsigned char : 4	high byte of character position in Hanin dictionary	½
idxoff1	unsigned char : 4	high byte of index in Hanin dictionary	½
Total Size			4

[Synopsis of IDX1]

The structure IDX1 is the first layer of index for finding the second layer index of Hanin dictionary. IDX1 indicates the index and offset in Hanin dictionary regarding input Chinese character.

Table 6: Structure of IDX2

Member	Type	Description	Size (byte)
schar	unsigned char	compressed code (not used in current version of HaninLib)	1
fseg	char	high byte of offset in Hanin dictionary	1
offset	unsigned short	offset of character in Hanin dictionary	2
clen	unsigned short	character length	2
Total Size			6

[Synopsis of IDX2]

The structure IDX2 is the second layer index of Hanin dictionary to get all the characters regarding specified compressed code.

Table 7: Structure of CORE

Member	Type	Description	Size (byte)
clen	short [4]	character length for specified compressed code	8
wlen	short [4]	length of word located in dictionaries	8

cidx	long [4]	index of character in Hanin dictionaries	16
widx	long [4]	index of word in Hanin dictionaries	16
wrange	short [4]	word range in Hanin dictionaries	8
mlenmark	char [4]	length mark for system dictionary	4
ulenmark	char	length mark for user dictionary	1
nowkey	char [10]	input key codes	10
fchar	unsigned char [10]	the first found 5 characters	10
flag	char	flag to represent change of character	1
flag_bak	char	backup for flag	1
Total Size			83

[Synopsis of CORE]

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

Table 8: Structure of WORKAREA

Member	Type	Description	Size (byte)
cchResult	unsigned short	length of result Chinese characters	2
chResult	char [64]	result Chinese characters in Big5 code	64
chMode	char [64]	code for each character	64
attrMode	char [64]	attribute for each characters	64
chDisplay	char [64]	displaying characters	64
attrDisplay	char [64]	displaying attribute of characters	64
PininPos	char	length of pinyin symbol in edit buffer	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 symbols in edit buffer	1
born_flag	char	flag to recover zhuyin fuhao	1
f_want	char	current position (in byte) in edit buffer	1
j_off	char	length of Chinese characters iin edit buffer	1
ownoff	char	length of zhuyin symbol in edit buffer	1
freq_fg	char	frequency for comparing with previous/next character	1
word_fg	char	word frequency	1
distmp	unsigned char [6]	zhuyin fuhao of current input	6
youwant	unsigned char [40]	Big5 code of Chinese characters	40

youwant_bak	unsigned char [40]	backup of youwant	40
cmpkigo	unsigned char [40]	compressed code of each zhuyin fuhao	40
nowkey_buf	char [10]	current input key codes	10
Core	CORE [20]	character information for each input Chinese character	1,660
liang	unsigned char [3]	measure word and its frequency	3
chbuf	unsigned char [600]	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	unsigned char [10]	conversion rule of pinyin or zhuyin fuhao	10
pos_buf	unsigned char [10]	position of pinyin symbol	10
outoff	unsigned long	offset of character address in Hanin dictionary	4
oldfreq	char	previous frequency	1
specChoice	char	flag to select 2-byte symbols	1
Total Size			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 available scancode.

Table 9: 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	unsigned char [2]	compressed code of aimei character/word	2
szAimeiPron	unsigned char [80]	aimei characters/words	80
Total Size			86

[Synopsis of AIMEISTRUCT]

In aimei function, the structure AIMEISTRUCT is used for storing Chinese characters and

words which pronunciations similar to current input Chinese character or word.

Table 10: Structure of SORTEDLIST

Member	Type	Description	Size (byte)
strSize	int	size of current user defined words	4
cmp_code	unsigned char *	pointer to compressed code of user word	4
char_code	unsigned char *	pointer to Big5 code of user word	4
Total Size			12

[Synopsis of SORTEDLIST]

The structure SORTEDLIST stores all the current user defined words retrieved from user dictionary by ascend order of zhuyin fuhao.

3. Module Specification

This section introduces all the modules in HaninLib Core.

[0001 ZhuinMatch]

This function converts input key code into zhuyin fuhao.

---

short ZhuinMatch (*void*)

ARGUMENTS

void

SYNOPSIS

(1) get the index of input key code and its code level
(2) get the zhuyin fuhao
(3) update display buffer

GLOBAL VARIABLES IN USE

unsigned char *inkey*
    [IN] current input key
char *nowkey_buf[]*
    [OUT] character buffer of current input keys
unsigned char *distmp[]*
    [OUT] character buffer for current zhuyin fuhao
char *ownoff*
    [IN] length of zhuyin symbol in edit buffer

RETURN VALUE

short *result* 1~4: level of zhuyin fuhao, 5: error occurred

MACRO IN USE

```
#define SP_INSERT          0x10
#define SP_REPLACE         0x40
```

CALLER FUNCTIONS

match

[0002 cmp2pinin]

This function converts compressed code into pinyin symbols.

short cmp2pinin (unsigned char **cmpCode*, unsigned char **pinin*)

ARGUMENTS

unsigned char **cmpCode*

[IN] buffer of compressed code

unsigned char **pinin*

[OUT] buffer of converted pinyin symbols

SYNOPSIS

(1) convert compressed code into pinyin symbols according to specified conversion rules.

GLOBAL VARIABLES IN USE

char *PininMode*

[IN] pinyin mode

RETURN VALUE

void

MACRO IN USE

```
#define TT                  0x80        /* MPS II */
#define MM                  0x40        /* roman pinyin */
#define YY                  0x20        /* with consonant */
#define NN                  0x10        /* no consonant */
#define TC                  6
```

CALLER FUNCTIONS

cmp2pinin

[0003 append]

This function appends pinyin alphabet during the process to convert compressed code into pinyin symbols.

void append (short *code*, short *flag*, unsigned char **pinin*, short **pc*)

ARGUMENTS

short *code*

[IN] index of compressed code

short *flag*

[IN] flag of rules for converting compressed code to pinyin symbols

unsigned char **pinin*

[OUT] buffer of converted pinyin symbols

short **pc*

[IN/OUT] index of pinyin symbol buffer

SYNOPSIS

(1) convert compressed code into pinyin symbols according to specified conversion rules.

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

void

MACRO IN USE

```
#define TT                  0x80        /* MPS II */
#define MM                  0x40        /* roman pinyin */
#define YY                  0x20        /* with consonant */
#define NN                  0x10        /* no consonant */
#define RR                  0x08        /* replace */
```

CALLER FUNCTIONS

cmp2pinin

[0004 CompressZhuin]

This function compresses zhuyin fuhao of character at current cursor position in edit buffer into compressed code.

---

void CompressZhuin (PBYTE *zCode*, PBYTE *cmpCode*)

ARGUMENTS

PBYTE *zCode*

[IN] buffer of zhuyin code

PBYTE *cmpCode*

[OUT] buffer of compressed code

SYNOPSIS

(1) covert zhuyin fuhao codes into compressed code

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

void

MACRO IN USE

```
#define TONE_COUNT          6
```

CALLER FUNCTIONS

AddACmp, match

[0005 RecoverZhuin]

This function recovers zhuyin fuhao from character at current cursor position in edit buffer.

---

void RecoverZhuin (short *loc*)

ARGUMENTS

short *loc*

[IN] position of character to be recovered

SYNOPSIS

(1) duplicate input keys for this character from *Core[loc].nowkey* to *nowkey_buf*

(2) delete tone

(3) conduct zhuyin/pinyin match

GLOBAL VARIABLES IN USE

char *nowkey_buf[]*

[OUT] character buffer of current input keys

char *Core[].nowkey*

[IN] input keys of specified location of character

char *PininPos*

[OUT] current position of pinyin symbol in edit buffer

char *ownoff*

[OUT] length of zhuyin symbol in edit buffer

char *PininMode*

[IN] pinyin mode

RETURN VALUE

void

MACRO IN USE

```
#define SPEC_CHAR          '\\'
```

CALLER FUNCTIONS

modify

[0006 PininMatch]

This function converts a series of input keys into pinyin alphabet string.

---

short PininMatch (*void*)

ARGUMENTS

void

SYNOPSIS

(1) convert a series of input key code into pinyin alphabet string by referring *pinin[][]* table

GLOBAL VARIABLES IN USE

unsigned char *inkey*

[IN] current input key

char *PininMode*

[IN] pinyin mode

char *PininPos*

[IN] current position of pinyin symbol in edit buffer

char *ownoff*

[IN] length of zhuyin symbol in edit buffer

unsigned char *rule_buf[]*

[IN] conversion rule of zhuyin/pinyin

unsigned char *pos_buf[]*

[IN] position of zhiyin/pinyin symbol

char *nowkey_buf[]*

[IN] character buffer of current input keys

RETURN VALUE

short *result* 5: error occurred, 4: completed, 1: uncompleted

MACRO IN USE

```
#define TONE_COUNT          6
#define SP_INSERT           0x10
#define SP_REPLACE          0x40
#define i_code              21
#define v_code              23
#define any_char            31
#define en_code             33
```

CALLER FUNCTIONS

match

[0007 PininDelete]

This function deletes a pinyin symbol

---

void PininDelete (*void*)

ARGUMENTS

void

SYNOPSIS

(1) delete a pinyin symbol

GLOBAL VARIABLES IN USE

char *DisplayMode*

[IN] display mode for zhuyin fuhao

char *PininPos*

[IN] current position of pinyin symbol in edit buffer

char *ownoff*

[IN] length of zhuyin symbol in edit buffer

unsigned char *rule_buf[]*

[IN] conversion rule of zhuyin/pinyin

unsigned char *pos_buf[]*

[IN] position of zhiyin/pinyin symbol

RETURN VALUE

void

MACRO IN USE

```
#define DISPLAY_ALPHA      1
#define SP_DELETE          0x20
#define SP_REPLACE         0x40
#define i_code             21
#define o_code             25
```

CALLER FUNCTIONS

backsp

[0008 Code2Level]

This function returns the level of specified zhuyin code

---

short Code2Level (char *code*)

ARGUMENTS

char *code*

[IN] zhuyin code

SYNOPSIS

(1) return the level of specified zhuyin code

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

short *level* 1: zhuyin code < *i_code*
2: *i_code* <= zhuyin code < *a_code*
3: *a_code* <= zhuyin code < *t1_code*
4 : others

MACRO IN USE

```
#define i_code          21
#define a_code          24
#define t1_code         37
```

CALLER FUNCTIONS

ZhuinMatch

[1001 scroll]

This function send the first character in edit buffer to caller application, and shift left one record of data regarding edit buffer

---

void scroll (*void*)

ARGUMENTS

void

SYNOPSIS

(1) assign the first character code and length to result buffer
(2) shift left one record of data
(3) update the values of global variables

GLOBAL VARIABLES IN USE

unsigned short *cchResult*
[OUT] character length of result Chinese characters in edit buffer
char *chResult[]*
[OUT] character buffer of result Chinese characters in edit buffer
char *j_off*
[OUT] character length of editing string in edit buffer
unsigned char *youwant[]*
[IN/OUT] Big5 code of Chinese characters in edit buffer
char *f_want*

[IN] the cursor position in edit buffer

char *undoLen*

[OUT] character length of same pronounce character/word

char *undoStart*

[IN/OUT] start position of homophone character/word

RETURN VALUE

void

MACRO IN USE

```
#define W_START            21
#define NOR                0
```

CALLER FUNCTIONS

others

[1002 fillback]

This function changes character code from Big5 code into TCA code. Currently, this function is not available.

---

void fillback (short *j*, short *i*)

ARGUMENTS

short *j*

[??] not referred in this function

short *i*

[IN] specified index of *youwant*

SYNOPSIS

(1) if TCAFLAG is 0, change the specified index of character from Big5 code into TCA code

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

void

MACRO IN USE

```
#define TCAFLAG             1
```

CALLER FUNCTIONS

scroll, perform

[1003 Clr_line]

This function clear specified range of edit buffer

---

void Clr_line (short *x1*, short *x2*)

ARGUMENTS

short *x1*

[IN] start index to be cleared

short *x2*

[IN] end index

SYNOPSIS

(1) insert character of space into display buffer from *x1* to *x2*

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

void

MACRO IN USE

```
#define NOR                 0
```

CALLER FUNCTIONS

scroll, match, initial, Handelete, show_pinin

[1004 bell]

No source

void bell (*void*)

ARGUMENTS

void

SYNOPSIS

(1) N/A

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

void

MACRO IN USE

N/A

CALLER FUNCTIONS

N/A

[1005 words_dis]

This function inserts length of characters into display buffer

void words_dis (short *loc*, unsigned char **s*, short *len*, short *mode*)

ARGUMENTS

short *loc*

[IN] display position of character

unsigned char **s*

[IN] character buffer to be inserted

short *len*

[IN] character length to be inserted

short *mode*

[IN] display attribute for characters

SYNOPSIS

(1) insert characters in character buffer into display buffer one by one

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

void

MACRO IN USE

N/A

CALLER FUNCTIONS

scroll, match, recover, MatchRuleA, MatchRuleB, MatchRuleC, Handelete, shift_cursor, show_pinin, Spatch, ReDisplay, do_rep

[1006 put_ch]

This function inserts a character into display buffer, and set its display attribute according its position.

---

void put_ch (short *x*, unsigned char *ch*, short *mode*)

ARGUMENTS

short *x*

[IN] display position of character

unsigned char *ch*

[IN] character code to be inserted

short *mode*

[IN] display attribute for this character

SYNOPSIS

(1) assign character code to display buffer

(2) assign display attribute to attribute buffer

GLOBAL VARIABLES IN USE

char *chDisplay[]*

[OUT] display buffer of current characters

char *attrDisplay[]*

[OUT] display attribute buffer of current characters

RETURN VALUE

void

MACRO IN USE

```
#define W_START            21
#define W_END              (W_START+L_STNCE+7+3)
#define NOR                0
#define BLUE               4
```

CALLER FUNCTIONS

Clr_line, words_dis, show_pinin

[1007 convert_attr]

This function returns the pseudo code of specified display attribute

---

char convert_attr (short *attr*)

ARGUMENTS

short *attr*

[IN] display attribute of characters in edit buffer

SYNOPSIS

(1) return pseudo code of specified display attribute

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

short *pcode* pseudo code of display attribute

MACRO IN USE

```
#define NOR                0
#define REV                1
#define GREEN              2
#define RED                3
```

```
#define BLUE                4
#define HILITE              5
```

CALLER FUNCTIONS

reset_word_area, put_ch

[1008 CheckDictionary]

This function opens Hanin dictionaries, loads their index table, and checks the availability of indexes

---

short CheckDictionary (*void*)

ARGUMENTS

void

SYNOPSIS

(1) load and check system/address dictionary
(2) load and check professional dictionaries
(3) load and check user dictionary

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

short *result* 0: failed, 1: success

MACRO IN USE

```
#define MAXPROFDICOPENED   3
```

CALLER FUNCTIONS

HAN_Start

[1009 MDLoad]

This function opens system/address dictionary and retrieves its IDX1 table.

---

short MDLoad (short *i*)

ARGUMENTS

short *i*

[IN] flag to indicate address dictionary

SYNOPSIS

(1) if *i* is 0 then open system dictionary, else open address dictionary

(2) retrieve its IDX1 table

GLOBAL VARIABLES IN USE

IDX1 *mdIdxTable1[]*

[OUT] index table in each dictionaries

RETURN VALUE

short *result* 0: failed to open the specified dictionary, 1: success

MACRO IN USE

```
#define MD_IDX_SIZE        (MD_MAX_IDX*sizeof(IDX1))
```

CALLER FUNCTIONS

CheckDictionary

[1010 ProfDicLoad]

This function opens specified professional dictionary and retrieves IDX1 table.

---

short ProfDicLoad (short *n*)

ARGUMENTS

short *n*

[IN] specified index of professional dictionary

SYNOPSIS

(1) open the specified professional dictionary

(2) retrieve IDX1 table

GLOBAL VARIABLES IN USE

IDX1 *mdIdxTable1[]*

[OUT] index table in each dictionaries

RETURN VALUE

short *result* 0: failed to open the professional dictionary, 1: success

MACRO IN USE

```
#define MD_IDX_SIZE        (MD_MAX_IDX*sizeof(IDX1))
```

CALLER FUNCTIONS

CheckDictionary

[1011 ckUDLoad]

This function loads index of learning words, index of user defined words, and index of abbreviations from user dictionary, and checks the availability of these indexes.

---

short ckUDLoad (*void*)

ARGUMENTS

void

SYNOPSIS

(1) if user dictionary doesn't exist, create a new file of user dictionary, initialize contents in file
(2) retrieve index of records of learning words, retrieve index table of user defined words, retrieve index of abbreviations
(3) check these indexes

GLOBAL VARIABLES IN USE

char *ldWordHead*
[IN/OUT] top record of learning word in *ldWord*
LDRec *ldWord[]*
[IN/OUT] records of learning word
short *udHashIdx[]*
[IN/OUT] index of hash table for user defined words
unsigned char *toutbuf[]*
[IN/OUT] character buffer for retrieving characters/words from Hanin dictionaries
unsigned char *idxbuff[]*

[IN/OUT] index buffer for abbreviations

RETURN VALUE

short *result* 0: error occurred, 1: success

MACRO IN USE

```
#define LD_MAX_WORD        100
#define UD_MAX_IDX         47
#define AD_IDX_SIZE        (AD_MAX_WORD*6)
#define USR_ID_LEN         12
#define USR_LAB_LEN        12
#define USR_LD_LEN         1024
#define UD_IDX_SIZE        ((UD_MAX_IDX+1)*2)
#define USR_AD_LEN         (AD_IDX_SIZE+4+64*AD_MAX_WORD)
#define OFF_QDSTART        17528
#define QD_IDXTABLE_SIZE   1536
#define QD_IDX_SIZE        6
```

CALLER FUNCTIONS

CheckDictionary

[1012 LDCheck]

This function checks the availability of linked index of structure for learning words.

---

short LDCheck (*void*)

ARGUMENTS

void

SYNOPSIS

(1) check the linked list of member *last* in struct LDRec
(2) check the linked list of member *next* in struct LDRec

GLOBAL VARIABLES IN USE

char *ldWordHead*
[IN] top record of learning word in *ldWord*
LDRec *ldWord[]*

[IN] records of learning word

RETURN VALUE

short *result* 0: OK, 1: wrong linked list

MACRO IN USE

```
#define LD_MAX_WORD        100
```

CALLER FUNCTIONS

ckUDLoad

[1013 LDCreate]

This function initializes the records of structure for learning word.

---

void LDCreate (*void*)

ARGUMENTS

void

SYNOPSIS

(1) assign linked index to each member
(2) initialize the values of members and top index of records

GLOBAL VARIABLES IN USE

char *ldWordHead*
[IN] top record of learning word in *ldWord*
LDRec *ldWord[]*
[IN] records of learning word

RETURN VALUE

void

MACRO IN USE

```
#define LD_MAX_WORD        100
```

CALLER FUNCTIONS

ckUDLoad

[1014 realtime]

This function is the entrance module of Hanin library. It parses the input keys, conducts processes by function keys, and executes Hanin conversion algorithms.

---

short realtime (*void*)

ARGUMENTS

void

SYNOPSIS

(1) parse the input key based on *InputState*, *PininMode*
(2) conduct the functional process when current input key is a function key
(3) execute Hanin conversion algorithm when input key is available to conversion

GLOBAL VARIABLES IN USE

unsigned char *inkey_bak*
    [OUT/IN] backup of current input key
unsigned char *inkey*
    [IN/OUT] current input key
unsigned char *scancode*
    [IN] scancode of current input key
char *InputState*
    [IN] current input state
unsigned int *KB_STATUS*
    [IN] status of keyboard
char *PininMode*
    [IN] pinyin mode
char *ownoff*
    [IN] length of zhuyin symbol in edit buffer
char *nowkey_buf[]*
    [IN] character buffer of current input keys
char *j_off*
    [IN] character length of editing string in edit buffer
char *f_want*
    [IN] the cursor position in edit buffer
char *ENDSEL*
    [OUT] flag to indicate whether the cursor is at the end of zhuyin fuhao or pinyin

symbol in edit buffer

char *undoLen*

[IN] character length of same pronounce character/word

RETURN VALUE

short *result* 0: no process, 1: conversion completed

MACRO IN USE

```
#define IS_PASS           2
#define IS_FULLABC        3
#define CAPS_LOCK         0x40
#define SHIFT             0x03
#define SPEC_CHAR         '\\'
#define UP                0x48
#define HOME              0x47
#define END               0x4F
#define LEFT              0x4B
#define RIGHT             0X4D
#define DEL               0x53
#define BCKSP             0x0E
#define ESC               0x01
#define RET               0x1C
#define L_STNCE           40
```

CALLER FUNCTIONS

HAN_InputKey

[1015 others]

This function handles the process when input non-function keys

---

short UnusedKey (*void*)

ARGUMENTS

void

SYNOPSIS

(1) shift one character to caller application when character length of edit buffer exceeds the

limitation

(2) conduct Hanin conversion algorithms

GLOBAL VARIABLES IN USE

char *ownoff*

[IN] length of zhuyin symbol in edit buffer

char *j_off*

[IN] character length of editing string in edit buffer

char *f_want*

[IN] the cursor position in edit buffer

RETURN VALUE

short *result* 0: conversion incompleted, 1: conversion completed

MACRO IN USE

```
#define L_STNCE            40
```

CALLER FUNCTIONS

realtime

[1016 UnusedKey]

This function checks current input symbol key is available regarding current input mode

---

short UnusedKey (*void*)

ARGUMENTS

void

SYNOPSIS

(1) set the flag of keyboard mapping regarding current input mode

(2) check availability of current input symbol key

GLOBAL VARIABLES IN USE

char *InputMode*

[IN] input mode, current keyboard mapping for zhuyin fuhao

struct spec_key *spec_key*

[IN] symbol key and its flag regarding 4 kinds of keyboard mapping

char *PininMode*

[IN] pinyin mode

RETURN VALUE

short *result* 0: unused symbol, 1: available symbol

MACRO IN USE

```
#define IBMKBD              2
#define ETKBD               3
#define MITECKBD            4
#define STD_BIT             0x01
#define IBM_BIT             0x02
#define ET_BIT              0x04
#define MITEC_BIT           0x08
```

CALLER FUNCTIONS

realtime

[1017 init_vars]

This function initializes global variables related to Hanin edit buffer

---

void init_vars (*void*)

ARGUMENTS

void

SYNOPSIS

(1) initialize the global variables

GLOBAL VARIABLES IN USE

char *f_want*

[OUT] the cursor position in edit buffer

char *ownoff*

[OUT] length of zhuyin symbol in edit buffer

char *j_off*

[OUT] character length of editing string in edit buffer

char *PininPos*

[OUT] current position of pinyin symbol in edit buffer

char *ENDSEL*

[OUT] flag to indicate whether the cursor is at the end of zhuyin fuhao or pinyin symbol in edit buffer

char *undoLen*

[OUT] character length of same pronounce character/word

char *ldDirty*

[OUT] the flag to indicate the change status of learning dictionary

char *born_flag*

[OUT] flag to indicate the status of recovering zhuyin fuhao

char *nowkey_buf[]*

[OUT] character buffer of current input keys

unsigned char *rule_buf[]*

[OUT] conversion rule of zhuyin/pinyin

unsigned char *pos_buf[]*

[OUT] position of zhiyin/pinyin symbol

RETURN VALUE

void

MACRO IN USE

```
#define L_STNCE            40
```

CALLER FUNCTIONS

initial

[1018 perform]

This function sends characters in edit buffer back to caller application

---

void perform (*void*)

ARGUMENTS

void

SYNOPSIS

(1) assign result length of characters in edit buffer to *cchResult*

(2) duplicate characters in edit buffer to *ccResult* buffer

GLOBAL VARIABLES IN USE

unsigned short *cchResult*

[OUT] character length of result Chinese characters in edit buffer

char *chResult[]*

[OUT] character buffer of result Chinese characters in edit buffer

char *j_off*

[IN] character length of editing string in edit buffer

RETURN VALUE

void

MACRO IN USE

N/A

CALLER FUNCTIONS

enter, put_back

[1019 initial]

This function initializes the global variables related to Hanin edit buffer

---

void initial (*void*)

ARGUMENTS

void

SYNOPSIS

(1) initialize global variables

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

void

MACRO IN USE

```
#define W_START            21
#define W_END              (W_START+L_STNCE+7+3)
```

CALLER FUNCTIONS

HAN_ClearEditBuf, reset_word_area, enter, put_back

[1020 enter]

This function executes the process when the Enter key pressed

---

void put_back (*void*)

ARGUMENTS

void

SYNOPSIS

(1) if zhuyin/pinyin matching is complete, send characters in edit buffer to caller application and update learning dictionary
(2) initialize the edit buffer

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

void

MACRO IN USE

N/A

CALLER FUNCTIONS

realtime

[1021 put_back]

This function sends current characters in edit buffer to caller application when input 1-byte alphabet

---

void put_back (*void*)

ARGUMENTS

void

SYNOPSIS

(1) send characters in edit buffer back to caller application
(2) update learning dictionary
(3) add alphabet code of current input key into result buffer
(4) initialize the data buffers related to edit buffer

GLOBAL VARIABLES IN USE

unsigned char *inkey*
[IN] current input key

RETURN VALUE

void

MACRO IN USE

N/A

CALLER FUNCTIONS

realtime

[1022 modify]

This function recovers zhuyin fuhao from character at current cursor in edit buffer

---

void modify (*void*)

ARGUMENTS

void

SYNOPSIS

(1) set recover flag to be ON
(2) recover zhuyin fuhao or pinyin alphabet by calling RecoverZhuin
(3) set recover flag to be OFF

GLOBAL VARIABLES IN USE

char *born_flag*
[IN/OUT] flag to indicate the status of recovering zhuyin fuhao

RETURN VALUE

void

MACRO IN USE

N/A

CALLER FUNCTIONS

realtime

[1023 recover]

This function recovers the change of word by matching Hanin algorithms

---

void recover (*void*)

ARGUMENTS

void

SYNOPSIS

(1) recover the change of converted word which conducted Hanin algorithms by swapping *undoLen* bytes of *youwant* for *youwant_bak*

GLOBAL VARIABLES IN USE

char *undoLen*
[IN/OUT] character length of same pronounce character/word
char *undoStart*
[IN/OUT] start position of homophone character/word
char *Core[].flag*
[IN/OUT] change flag for specified character
char *Core[].flag_bak*
[IN/OUT] backup of *Core[].flag*
unsigned char *youwant[]*
[IN/OUT] Big5 code of Chinese characters in edit buffer
unsigned char *youwant_bak[]*
[IN/OUT] backup of *youwant[]*
unsigned char *chbuf[]*
[IN] choice buffer for selecting homophone characters/words

RETURN VALUE

void

MACRO IN USE

```
#define W_START            21
#define NOR                0
```

CALLER FUNCTIONS

match, Handelete

[1024 keypad]

This function checks current input key is pressing keypad key or not

---

short keypad (*void*)

ARGUMENTS

void

SYNOPSIS

(1) check current input key is a keypad key

GLOBAL VARIABLES IN USE

unsigned char *scancode*
[IN] scancode of current input key
unsigned char *inkey*
[IN] current input key

RETURN VALUE

short *result* 0: not a keypad key, 1: keypad key

MACRO IN USE

N/A

CALLER FUNCTIONS

N/A

[1025 chg_key]

This function change capital letter of *inkey* into small letter, and vice versa

---

void chg_key (*void*)

ARGUMENTS

void

SYNOPSIS

(1) change capital letter into small letter, and vice versa

GLOBAL VARIABLES IN USE

unsigned char *inkey*
[IN/OUT] current input key

RETURN VALUE

void

MACRO IN USE

N/A

CALLER FUNCTIONS

realtime

[1026 Handelete]

This function deletes a character on current cursor position in edit buffer.

---

short Handelete (short *isBackSpace*)

ARGUMENTS

short *isBackSpace*
[IN] flag to indicate the status of Backspace key pressed

SYNOPSIS

(1) delete a character and its related data

GLOBAL VARIABLES IN USE

char *j_off*

[IN] character length of editing string in edit buffer

char *f_want*

[IN] the cursor position in edit buffer

char *undoLen*

[IN/OUT] character length of same pronounce character/word

char *undoStart*

[IN/OUT] start position of homophone character/word

char *Core[].flag*

[IN] change flag for specified character

unsigned char *youwant[]*

[IN] Big5 code of Chinese characters in edit buffer

RETURN VALUE

short *status* -1: failed, 0: success

MACRO IN USE

```
#define W_START            21
#define NOR                0
```

CALLER FUNCTIONS

RecoverZhuin, realtime, backsp

[1027 backsp]

This function deletes data related to edit buffer when Backspace key was pressed.

---

short backsp (*void*)

ARGUMENTS

void

SYNOPSIS

(1) delete a character when zhuyin/pinyin matching is completed

(2) delete a zhuyin fuhao or pinyin alphabet during zhuyin/pinyin matching

GLOBAL VARIABLES IN USE

char *f_want*

[IN] the cursor position in edit buffer

char *ownoff*

[IN/OUT] length of zhuyin symbol in edit buffer

char *PininMode*

[IN] pinyin mode

char *nowkey_buf[]*

[IN/OUT] character buffer of current input keys

RETURN VALUE

short *status* -1: failed, 0: success

MACRO IN USE

```
#define SPEC_CHAR          '\\'
#define SP_DELETE          0x20
```

CALLER FUNCTIONS

realtime

[1028 shift_cursor]

This function moves current cursor in edit buffer

---

void shift_cursor (short *count*, short *show*)

ARGUMENTS

short *count*

[IN] offset which cursor moves

short *show*

not be referred in this function

SYNOPSIS

(1) if current cursor is able to be moved, shifts *count* position in edit buffer

GLOBAL VARIABLES IN USE

char *f_want*

[IN] the cursor position in edit buffer

char *ownoff*

[IN] length of zhuyin symbol in edit buffer

char *j_off*

[IN] character length of editing string in edit buffer

unsigned char *youwant[]*

[IN] Big5 code of Chinese characters in edit buffer

RETURN VALUE

void

MACRO IN USE

```
#define W_START            21
#define NOR                0
```

CALLER FUNCTIONS

GetSameProns, SelectSamePron, SelectAimeiPron, HAN_BeginGetAimei, HAN_GetAimeiChars, HAN_GetAimeiWords, GetAllSpecChars, IsAllSpecByte, realtime, backsp

[1029 show_pinin]

This function display zhuyin fuhao to the output buffer

---

void show_pinin (short *mode*, char *code*)

ARGUMENTS

short *mode*

[IN] display mode, 0: append zhuyin after current position, 1: replace zhuyin at current position, 2: replace zhuyin at previous position

char *code*

[IN] code of zhuyin fuhao

SYNOPSIS

(1) append or replace zhuyin fuhao for output in edit buffer according to *mode* and *DisplayMode*

GLOBAL VARIABLES IN USE

unsigned char *distmp[]*

[IN] character buffer for current zhuyin fuhao

char *DisplayMode*

[IN] display mode for zhuyin fuhao

char *nowkey_buf[]*

[IN] character buffer of current input keys

char *f_want*

[IN] the cursor position in edit buffer

char *j_off*

[IN] character length of editing string in edit buffer

char *PininPos*

[IN] current position of pinyin symbol in edit buffer

RETURN VALUE

void

MACRO IN USE

```
#define SP_INSERT          0x10
#define SP_REPLACE         0x40
#define SP_POS             0x0F
#define W_START            21
#define HILITE             5
#define NOR                0
```

CALLER FUNCTIONS

ZhuinMatch, SymbolMatch, PininMatch, PininDelete, backsp

[1030 big5cmp]

This function compares character codes in edit buffer with specified characters.

---

short big5cmp (unsigned char **s*, short *start*, short *len*)

ARGUMENTS

unsigned char **s*

[IN] character buffer to be compared

short *start*

[IN] specified index in edit buffer

short *len*

[IN] character length

SYNOPSIS

(1) compare characters of *(*youwant+start)* with **s*

GLOBAL VARIABLES IN USE

char *Core[].flag*

[IN] change flag for specified character

unsigned char *youwant[]*

[IN] Big5 code of Chinese characters in edit buffer

RETURN VALUE

short *result* 0: identical, -1: not the same

MACRO IN USE

```
#define CHG_FG              0x80
#define HANIN_CRYPT         0x2A
```

CALLER FUNCTIONS

LDSearchWord, MDSearchWord, UDSearchWord

[1031 init_len_mark]

This function initializes the specified record of struct CORE

---

void init_len_mark (short *i*)

ARGUMENTS

short *i*

[IN] index of records in *Core*

SYNOPSIS

(1) initialize the value of each member of *Core[i]*

GLOBAL VARIABLES IN USE

CORE *Core[]*

[OUT] records of character information for each input characters

RETURN VALUE

void

MACRO IN USE

```
#define MAXPROFDICOPENED   3
```

CALLER FUNCTIONS

HAN_SetAddressState, HAN_GetAimeiChars, HAN_GetAimeiWords, match, init_vars, Handelete

[1032 AppendToBuffer]

This function append specified length of specified characters into character buffer *chbuf*

---

void AppendToBuffer (unsigned char **code*, short *count*, short *st*, short *len*, short *mode*)

ARGUMENTS

unsigned char **code*

[IN] character buffer to be appended

short *count*

[IN] amount of characters to be appended

short *st*

[IN] specified index in edit buffer

short *len*

[IN] character length for each word

short *mode*

[IN] search mode

SYNOPSIS

(1) append *len*-byte of characters from **code* to *chbuf*

(2) if *mode* is SEARCH_ALL, skip the same characters already exist in *chbuf*

GLOBAL VARIABLES IN USE

short *chlen*

[IN/OUT] current character length of *chbuf*

unsigned char *chbuf[]*

[IN/OUT] choice buffer for selecting homophone characters/words

unsigned char *youwant[]*

[IN/OUT] Big5 code of Chinese characters in edit buffer

RETURN VALUE

void

MACRO IN USE

```
#define HANIN_CYYPT        0x2A
#define SEARCH_ALL         1
```

CALLER FUNCTIONS

LDSearchWord, MDSearchChar, MDSearchWord, UDSearchWord, UnitSearchWord, AddrCharMatch

[2001 match]

This function performs Hanin algorithms for converting input keys to zhuyin fuhao, and converts zhuyin to Chinese character

---

short match (*void*)

ARGUMENTS

void

SYNOPSIS

(1) check the state of zhuyin/pinyin match or symbol match
(2) shift right one record of data related to edit buffer
(3) compress completed zhuyin/pinyin into compressed code
(4) execute Hanin algorithms to convert zhuyin/pinyin into Chinese character/word

GLOBAL VARIABLES IN USE

char *ownoff*
    [IN] length of zhuyin symbol in edit buffer
unsigned char *inkey*
    [IN] current input key
char *nowkey_buf[]*
    [IN] character buffer of current input keys
char *PininMode*
    [IN] pinyin mode
char *born_flag*
    [IN] flag to indicate the status of recovering zhuyin fuhao
unsigned char *cmpkigo[]*

[IN/OUT] the compressed code buffer

char *f_want*

[IN] the cursor position in edit buffer

char *Core[].nowkey*

[OUT] current input keys

unsigned char *distmp[]*

[IN] character buffer for current zhuyin fuhao

char *j_off*

[IN] character length of editing string in edit buffer

char *AddrState*

[IN] flag to represent current use of address dictionary

CORE *Core[]*

[IN/OUT] records of character information for each input characters

char *word_fg*

[OUT/IN] flag to indicate the status of Chinese word conversion

char *oldfreq*

[OUT/IN] previous frequency

unsigned char *liang[3]*

[OUT/IN] character buffer to store adverb character and its frequency

char *undoLen*

[IN/OUT] character length of same pronounce character/word

char *undoStart*

[IN/OUT] start position of homophone character/word

char *freq_fg*

[IN] word frequency

unsigned char *youwant_bak[]*

[IN/OUT] backup of *youwant[]*

RETURN VALUE

short *state* -1: error occurred, 0: uncompleted match, 1: zhuyin fuhao recovered, 2: completed match

MACRO IN USE

```
#define SPEC_CHAR          '\\'
#define WRD_FG             0x08
#define LEN_FG             0x07
#define FRQ_FG             0x70
#define W_START            21
#define NOR                0
```

CALLER FUNCTIONS

RecoverZhuin, others

[2002 SymbolMatch]

This function checks match of input symbol

---

short SymbolMatch (*void*)

ARGUMENTS

void

SYNOPSIS

(1) if *ownoff* is 0, set *SPEC_CHAR* to *nowkey_buf* and return uncompleted state
(2) set *inkey* to *nowkey_buf*, and return completed match state

GLOBAL VARIABLES IN USE

char *ownoff*
[IN] length of zhuyin symbol in edit buffer
char *nowkey_buf[]*
[OUT] character buffer of current input keys
unsigned char *inkey_bak*
[IN] backup of current input key
unsigned char *inkey*
[IN] current input key

RETURN VALUE

short *state* 1: uncompleted match, 4: completed match

MACRO IN USE

```
#define SPEC_CHAR          '\\'
#define SP_INSERT          0x10
```

CALLER FUNCTIONS

match

[2003 rshift_data]

This function shifts right related records and buffers regarding current cursor in edit buffer.

void lshift_data (*void*)

ARGUMENTS

void

SYNOPSIS

(1) shift right the data of *youwant*, *youwant_bak*, and *cmpkigo*
(2) shift right one record of *Core[]*

GLOBAL VARIABLES IN USE

char *j_off*
[IN] character length of editing string in edit buffer
char *f_want*
[IN] the cursor position in edit buffer
unsigned char *youwant[]*
[IN/OUT] Big5 code of Chinese characters in edit buffer
unsigned char *youwant_bak[]*
[IN/OUT] backup of *youwant[]*
unsigned char *cmpkigo[]*
[IN/OUT] the compressed code buffer
CORE *Core[]*
[IN/OUT] records of character information for each input characters

RETURN VALUE

void

MACRO IN USE

N/A

CALLER FUNCTIONS

match

[2004 lshift_data]

This function shifts left related records and buffers regarding current cursor in edit buffer.

void lshift_data (short *start*)

ARGUMENTS

short *start*

[IN] index to start shifting data

SYNOPSIS

(1) shift left the data of *youwant*, *youwant_bak*, and *cmpkigo*

(2) shift left one record of *Core[]*

GLOBAL VARIABLES IN USE

char *j_off*

[IN] character length of editing string in edit buffer

unsigned char *youwant[]*

[IN/OUT] Big5 code of Chinese characters in edit buffer

unsigned char *youwant_bak[]*

[IN/OUT] backup of *youwant[]*

unsigned char *cmpkigo[]*

[IN/OUT] the compressed code buffer

CORE *Core[]*

[IN/OUT] records of character information for each input characters

RETURN VALUE

void

MACRO IN USE

N/A

CALLER FUNCTIONS

scroll, match, Handelete

[3001 SearchAllChars]

This function searches all the homophone characters.

short SearchAllChars (*void*)

ARGUMENTS

void

SYNOPSIS

(1) search same pronounced characters in learning dictionary
(2) search same pronounced characters in professional dictionaries
(3) search same pronounced characters in system dictionary
(4) if *specChoice* is ON, filter out the 2-byte symbols, else get the Big5E characters

GLOBAL VARIABLES IN USE

unsigned char *cmpkigo[]*
    [IN] the compressed code buffer
char *f_want*
    [IN] the cursor position in edit buffer
short *chlen*
    [OUT] current character length of *chbuf*
unsigned char *chbuf[]*
    [OUT] choice buffer for selecting homophone characters/words
char *specChoice*
    [IN] flag to indicate the selection of 2-byte fuhao characters

RETURN VALUE

short *length*: number of bytes in searching buffer

MACRO IN USE

```
#define SEARCH_ALL         1
#define MAXPROFDICOPENED   3
```

CALLER FUNCTIONS

GetSameProns, HAN_GetAimeiChars

[3002 SearchAllWords]

This function searches all the homophone words

---

short SearchAllWords (*void*)

ARGUMENTS

void

SYNOPSIS

(1) search same pronounced words in learning dictionary
(2) search same pronounced words in user dictionary
(3) search same pronounced words in professional dictionaries
(4) search same pronounced words in system dictionary
(5) conduct special rules for input word

GLOBAL VARIABLES IN USE

unsigned char *cmpkigo[]*
    [IN] the compressed code buffer
char *f_want*
    [IN] the cursor position in edit buffer
short *chlen*
    [OUT] current character length of *chbuf*

RETURN VALUE

short *length*: number of bytes in searching buffer

MACRO IN USE

```
#define SEARCH_ALL         1
#define MAXPROFDICOPENED   3
```

CALLER FUNCTIONS

GetSameProns, HAN_GetAimeiWords

[3003 ReadBig5E]

This function retrieves Big5E characters from Big5E dictionary and calculates the index of Big5E characters in candidate buffer.

---

short ReadBig5E (char **big5e_dic*)

ARGUMENTS

char **big5e_dic*
    [IN] full path and filename of Big5E dictionary

SYNOPSIS

(1) retrieve Big5E characters into candidate buffer

(2) calculate index of Big5E characters in candidate buffer

GLOBAL VARIABLES IN USE

BIG5EIDX *big5eidx[]*

[IN] structure of index for Big5E characters

char *big5echar[]*

[IN] candidate buffer for Big5E characters

char *szTone[]*

[IN] character buffer of tones

RETURN VALUE

short *result*: 0: failed, 1: success

MACRO IN USE

```
#define MAXEXTIDX          1100
```

CALLER FUNCTIONS

SearchAllChars, SearchOneChar

[3004 GetBig5EChar]

This function gets the Big5E characters from candidate buffer.

---

short GetBig5EChar (char **pCmp*, char **pCharBuf*, short *num*)

ARGUMENTS

char **pCmp*

[IN] compressed code to be searched

char **pCharBuf*

[IN] candidate buffer for Big5E characters

short *num*

[IN] number of Big5E character to be retrieved, *num* = 1 for retrieving one character, others for all characters

SYNOPSIS

(1) compare specified compressed code with compressed code in *big5eidx*

(2) if found, retrieve Big5E characters from candidate buffer

GLOBAL VARIABLES IN USE

BIG5EIDX *big5eidx[]*

[IN] structure of index for Big5E characters

char *big5echar[]*

[IN] candidate buffer for Big5E characters

RETURN VALUE

short *length*: character length of retrieved Big5E characters

MACRO IN USE

N/A

CALLER FUNCTIONS

SearchAllChars, SearchOneChar

[3005 AddACmp]

This function compresses specified zhuyin fuhao string into compressed code and inserts it into structure of index for Big5E characters.

---

short AddACmp (PBIG5EIDX *pIdx*, char **pZhuin*, short *len*)

ARGUMENTS

PBIG5EIDX *pIdx*

[OUT] pointer to structure of index for Big5E characters

char **pZhuin*

[IN] zhuyin fuhao string

short *len*

[IN] character length of *pZhuyin*

SYNOPSIS

(1) calculate replacement codes by specified zhuyin fuhao

(2) compress the replacement codes into compressed code

GLOBAL VARIABLES IN USE

char *szZhuin[]*

[IN] character buffer of zhuyin fuhao

char *szTone[]*

[IN] character buffer of tones

RETURN VALUE

short *result*: 0: failed, 1: success

MACRO IN USE

N/A

CALLER FUNCTIONS

ReadBig5E

[3006 CompareCmp]

This function compares two compressed code buffer.

---

short CompareCmp (char **pCmp1*, char **pCmp2*)

ARGUMENTS

char **pCmp1*

[IN] compressed code buffer to be compared

char **pCmp2*

[IN] another compressed code buffer to be compared

SYNOPSIS

(1) compare the first character of each string. If they are equal to each other, it continues with the following pairs until the characters differ or until a terminating null-character is reached

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

short *result*: 0: both strings are equal, positive number: *pCmp1* has a greater value than *pCmp2*, negative number: *pCmp2* has a greater value than *pCmp1*

MACRO IN USE

N/A

CALLER FUNCTIONS

ReadBig5E, GetBig5EChar

[3007 sear_aimei]

This function read information about aimei pronounced data regarding specified aimei zhuyin from aimei dictionary.

---

short sear_aimei (char **pron*, char **aimei_pron*)

ARGUMENTS

char **pron*

[IN] aimei zhuyin to be searched

char **aimei_pron*

[OUT] buffer of all candidated aimei zhuyin regarding **pron*

SYNOPSIS

(1) calculate the address location by specified aimei zhuyin **pron*

(2) get the value of offset from aimei dictionary

(3) retrieve the candidated aimei zhuyin from aimei dictionary

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

short *count*: number of candidated aimei zhuyin

MACRO IN USE

N/A

CALLER FUNCTIONS

HAN_BeginGetAimei

[3008 cmp2zhuin]

This function recovers the compressed code to zhuyin fuhao string

short cmp2zhuin (PBYTE *cmpCode*, PBYTE *zhuin*)

ARGUMENTS

PBYTE *cmpCode*

[IN] compressed code of user defined word

PBYTE *zhuin*

[OUT] character buffer for recovered zhuyin fuhao

SYNOPSIS

(1) recover zhuyin fuhao string regarding specified compressed code

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

short *length*: character length of zhuyin fuhao string

MACRO IN USE

```
#define TONE_COUNT          6
```

CALLER FUNCTIONS

SelectAimeiPron, HAN_GetAimeiChars, HAN_GetAimeiWords, HAN_GetUserWord

[3009 GetSameProns]

This function retrieves all the homophone characters or same pronounced words.

short GetSameProns (PHANIN *pHanin*, short *searchlen*, PBYTE *pCandBuf*, short *CandBufSize*, short *bSpecCharOnly*)

ARGUMENTS

PHANIN *pHanin*

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

short *searchlen*

[IN] character length of searched word

PBYTE *pCandBuf*

[OUT] pointer to a character buffer which includes all the same pronounced words

short *CandBufSize*

[IN] size of candidate buffer

short *bSpecCharOnly*

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

SYNOPSIS

(1) when the following status occurred, it can not select homophone word
g_bReady is False, *pHanin* is NULL, *ownoff* is not 0, character length in edit buffer is less than *searchlen*

(2) if *searchlen* is 4 then search all same pronounced words, else search all same pronounced characters

GLOBAL VARIABLES IN USE

short *g_bReady*

[IN] a flag to represent current status of HaninLib

char *j_off*

[IN] character length of editing string in edit buffer

char *ownoff*

[IN] length of zhuyin symbol in edit buffer

char *f_want*

[IN] the cursor position in edit buffer

short *chlen*

[OUT/IN] current character length of *chbuf*

short *oldchlen*

[OUT/IN] previous character length of *chbuf*

unsigned char *cmpkigo[]*

[IN] the compressed code buffer

char *ENDSEL*

[IN] flag to indicate whether the cursor is at the end of zhuyin fuhao or pinyin symbol in edit buffer

char *specChoice*

[OUT] flag to select 2-byte symbols

RETURN VALUE

short *length*: 0: failed to select same pronounced words, others: character length of candidate buffer

MACRO IN USE

```
#define HANIN_CRYPT5C      0x5C
```

CALLER FUNCTIONS

HAN_GetSamePronChars, HAN_GetSamePronWords

[3010 SelectSamePron]

This function chooses the specified index of homophone word among the candidates of homophone words.

---

short SelectSamePron (PHANIN *pHanin*, short *selectlen*, PBYTE *pSelBuf*, PCONVRESULT *pConvResult*)

ARGUMENTS

PHANIN *pHanin*

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

short *selectlen*

[IN] character length of selection word

PBYTE *pSelBuf*

[IN] pointer to a character buffer which includes all the same pronounced words

PCONVRESULT *pConvResult*

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

SYNOPSIS

(1) when the following status occurred, it can not select homophone word
g_bReady is False, *pHanin* is NULL, *ownoff* is not 0, character length in edit buffer is less than *selectlen*

(2) update character code in *youwant* by character code of selected same pronounced word

(3) insert learning word

(4) update the *pConvResult*

GLOBAL VARIABLES IN USE

short *g_bReady*

[IN] a flag to represent current status of HaninLib

char *j_off*

[IN] character length of editing string in edit buffer

char *ownoff*

[IN] length of zhuyin symbol in edit buffer

char *f_want*

[IN] the cursor position in edit buffer

unsigned char *youwant[]*

[IN] Big5 code of Chinese characters in edit buffer

char *ENDSEL*

[IN] flag to indicate whether the cursor is at the end of zhuyin fuhao or pinyin symbol in edit buffer

RETURN VALUE

short *result*: False: failed to select an same pronounced word, True: success

MACRO IN USE

```
#define HANIN_CRYPT5C      0x5C
```

CALLER FUNCTIONS

HAN_SelectSamePronChar, HAN_SelectSamePronWord, DoSelectSpecChar

[3011 SelectAimeiPron]

This function chooses the specified index of aimei word among the candidates of aimei words.

---

short SelectAimeiPron (PHANIN *pHanin*, PAIMEI *pAimei*, short *idx*,
short *selectlen*, PBYTE *pSelBuf*,
PCONVRESULT *pConvResult*)

ARGUMENTS

PHANIN *pHanin*

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

PAIMEI *pAimei*

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

short *idx*

[IN] index of compressed code in *pAimei*

short *selectlen*

[IN] character length of selection word

PBYTE *pSelBuf*

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

PCONVRESULT *pConvResult*

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

SYNOPSIS

(1) when the following status occurred, it can not select aimei word
g_bReady is False, *pHanin* is NULL, *pAimei* is NULL, *idx* is out of range, *ownoff* is not 0, character length in edit buffer is less than *selectlen*

(2) update compressed code in *cmpkigo* by compressed code of selected aimei word

(3) update character code in *youwant* by character code of selected aimei word

(4) insert learning word

(5) update the *pConvResult*

GLOBAL VARIABLES IN USE

short *g_bReady*

[IN] a flag to represent current status of HaninLib

char *j_off*

[IN] character length of editing string in edit buffer

char *ownoff*

[IN] length of zhuyin symbol in edit buffer

char *f_want*

[IN] the cursor position in edit buffer

unsigned char *cmpkigo[]*

[IN] the compressed code buffer

char *Core[].nowkey*

[OUT] current input keys

unsigned char *youwant[]*

[IN] Big5 code of Chinese characters in edit buffer

char *ENDSEL*

[IN] flag to indicate whether the cursor is at the end of zhuyin fuhao or pinyin symbol in edit buffer

RETURN VALUE

short *result*: False: failed to select an aimei word, True: success

MACRO IN USE

```
#define HANIN_CRYPT        0x2A
#define HANIN_CRYPT5C      0x5C
#define DISPLAY_ALPHA      1
```

CALLER FUNCTIONS

HAN_SelectAimeiChar, HAN_SelectAimeiWord

[3012 LDWrite]

This function writes records of learning words into learning dictionary.

---

short LDWrite (*void*)

ARGUMENTS

void

SYNOPSIS

(1) if *ldDirty* is ON, set the index of top record and write the records of learning words into learning dictionary

GLOBAL VARIABLES IN USE

char *ldDirty*

[IN/OUT] the flag to indicate the change status of learning dictionary

char *ldWordHead*

[IN] top record of learning word in *ldWord*

LDRec *ldWord[]*

[IN] records of learning word

RETURN VALUE

short *result*: 0: success

MACRO IN USE

```
#define LD_MAX_WORD        100
#define USR_LD_POS         (USR_LAB_POS+USR_LAB_LEN)
#define USR_LD_LEN         1024
```

CALLER FUNCTIONS

HAN_Close, enter, put_back

[3013 LDSearchWord]

This function searches the specified learning word in learning dictionary. The maximum character length for learning word is 4-byte.

---

short LDSearchWord (short *st*, short *len*, short *mode*)

ARGUMENTS

short *st*

[IN] specified index in edit buffer

short *len*

[IN] character length of word

short *mode*

[IN] search mode

SYNOPSIS

(1) If the specified character length is 2, search learning characters in records of learning words. If character length is 4, search learning word in records of learning words.

GLOBAL VARIABLES IN USE

short *chlen*

[IN] current character length of *chbuf*

short *oldchlen*

[OUT/IN] previous character length of *chbuf*

unsigned char *cmpkigo[]*

[IN] the compressed code buffer

char *ldWordHead*

[IN] top record of learning word in *ldWord*

LDRec *ldWord[]*

[IN] records of learning word

char *freq_fg*

[OUT] word frequency

RETURN VALUE

short *length*: number of searched bytes

MACRO IN USE

```
#define HANIN_CRYPT        0x2A
#define SEARCH_ALL         1
#define SEARCH_ONE         0
#define FRQ_FG             0x70
```

CALLER FUNCTIONS

SearchAllChars, SearchAllWords, SearchOneWord, SearchOneChar

[3014 LDInsertWord]

This function adds a new learning word into learning dictionary. If the input learning word already exists, moves this learning word to the top of records of learning words.

---

short LDInsertWord (PBYTE *cmpCode*, PBYTE *bigCode*)

ARGUMENTS

PBYTE *cmpCode*

[IN] compressed code of user defined word

PBYTE *bigChar*

[IN] character code of user defined word

SYNOPSIS

(1) if the input learning word already exists, move it to the top of records of *ldWord*, if it doesn't exist, insert the new learning word into the top record of *ldWord*

GLOBAL VARIABLES IN USE

char *ldDirty*

[OUT] the flag to indicate the change status of learning dictionary

char *ldWordHead*

[OUT/IN] top record of learning word in *ldWord*

LDRec *ldWord[]*

[IN/OUT] records of learning word

RETURN VALUE

short *result*: -1: learning word exists, 0: success

MACRO IN USE

```
#define HANIN_CRYPT        0x2A
```

CALLER FUNCTIONS

InsertLearnCode

[3015 LDMoveWord]

This function moves the specified learning word to the top of learning dictionary.

---

void LDMoveWord (unsigned char *ldWordCur*)

ARGUMENTS

unsigned char *ldWordCur*

[IN] index of learning word to be moved

SYNOPSIS

(1) change the linked index of struc LDRec *ldWord* by *ldWordCur*

GLOBAL VARIABLES IN USE

char *ldDirty*

[OUT] the flag to indicate the change status of learning dictionary

char *ldWordHead*

[OUT/IN] top record of learning word in *ldWord*

LDRec *ldWord[]*

[IN/OUT] records of learning word

RETURN VALUE

void

MACRO IN USE

N/A

CALLER FUNCTIONS

LDSearchWord, LDInsertWord

[3016 InsertLearnCode]

This function adds a new learning word or character into learning dictionary.

void InsertLearnCode (short *learnlen*)

ARGUMENTS

short *learnlen*

[IN] character length of learning word or character

SYNOPSIS

(1) add a new learning word or character into learning dictionary

GLOBAL VARIABLES IN USE

unsigned char *cmpkigo[]*

[IN] the compressed code buffer

unsigned char *youwant[]*

[IN] Big5 code of Chinese characters in edit buffer

char *f_want*

[IN] the cursor position in edit buffer

char *Core[].flag*

[OUT] change flag for specified character

char *undoLen*

[IN/OUT] character length of same pronounce character/word

char *undoStart*

[IN] start position of homophone character/word

RETURN VALUE

void

MACRO IN USE

```
#define CHG_FG              0x80
```

CALLER FUNCTIONS

SelectSamePron, SelectAimeiPron

[3017 SearchOneWord]

This function searches a word from dictionaries by the order of learning dictionary -> professional dictionaries -> system dictionary.

ARGUMENTS

short *st*

[IN] specified index in edit buffer

short *len*

[IN] character length of word

SYNOPSIS

(1) execute the rule of repeated word if word length is 6-byte
(2) if not found, search one word in learning dictionary
(3) if not found, search one word in user dictionary
(4) if not found, search one word in address dictionary when *AddrState* is ON. When *AddrState* is OFF, search one word in professional dictionaries
(5) if not found, search one word in system dictionary
(6) if not found, search one word in unit table

GLOBAL VARIABLES IN USE

short *chlen*

[OUT] current character length of *chbuf*

unsigned char *chbuf[]*

[OUT] choice buffer for selecting homophone characters/words

char *AddrState*

[IN] flag to represent current use of address dictionary

RETURN VALUE

short *result*: -1: not found, 0: success

MACRO IN USE

```
#define SEARCH_ONE         0
#define MAXPROFDICOPENED   3
```

CALLER FUNCTIONS

match

[3018 SearchOneChar]

This function searches a character from dictionaries by the order of learning dictionary ->

professional dictionaries -> system dictionary.

---

short SearchOneChar (short *st*)

ARGUMENTS

short *st*

[IN] specified index in edit buffer

SYNOPSIS

(1) search one character in learning dictionary
(2) if not found, search one character in address dictionary when *AddrState* is ON. When *AddrState* is OFF, search one character in professional dictionaries
(3) if not found, search one character in system dictionary
(4) if not found, search one character in Big5E dictionary

GLOBAL VARIABLES IN USE

short *chlen*

[OUT] current character length of *chbuf*

unsigned char *chbuf[]*

[OUT] choice buffer for selecting homophone characters/words

unsigned char *cmpkigo[]*

[IN] the compressed code buffer

char *AddrState*

[IN] flag to represent current use of address dictionary

RETURN VALUE

short *result*: -1: not found, 0: success

MACRO IN USE

```
#define SEARCH_ONE         0
#define MAXPROFDICOPENED   3
```

CALLER FUNCTIONS

match, recover

[3019 MDSearchFChar]

This function searches the first 5 character candidates from specified dictionary.

short MDSearchFChar (short *nDic*, short *st*)

ARGUMENTS

short *nDic*

[IN] index of available dictionaries

short *st*

[IN] specified index in edit buffer

SYNOPSIS

(1) return error code if current character is a 2-byte symbol

(2) get the index of compressed code **(cmpkugo+st)* in specified dictionary

(3) get the offset for first 5 characters area

(4) retrieves the first 5 characters from specified dictionary

GLOBAL VARIABLES IN USE

unsigned char *cmpkigo[]*

[IN] the compressed code buffer

unsigned char *toutbuf[]*

[OUT] character buffer for retrieving characters/words from Hanin dictionaries

IDX1 *mdIdxTable1[]*

[IN] index table in each dictionaries

char *AddrState*

[IN] flag to represent current use of address dictionary

unsigned char *Core[].fchar[]*

[OUT] the 5 first found characters

RETURN VALUE

short *result*: -1: not found, 0: success

MACRO IN USE

```
#define MD_MAX_IDX          58
#define MD_FCHAR_POS        23552
```

CALLER FUNCTIONS

MDSearchChar, MDSearchWord

[3020 MDSearchChar]

This function searches characters in system dictionary, address dictionary or professional dictionaries.

short MDSearchChar (short *nDic*, short *st*, short *mode*)

ARGUMENTS

short *nDic*

[IN] index of available dictionaries

short *st*

[IN] specified index in edit buffer

short *mode*

[IN] search mode, 0 for search one character and 1 for search all characters

SYNOPSIS

(1) retrieve one character using fast found 5 characters buffer if search mode is SEARCH_ONE
(2) get the value of offset indicator regarding specified dictionary
(3) if the length of characters is not 0, retrieve all the characters in dictionary
(4) if SEARCH_ALL, append all found characters into *toutbuf*

GLOBAL VARIABLES IN USE

unsigned char *cmpkigo[]*

[IN] the compressed code buffer

unsigned char *toutbuf[]*

[OUT] character buffer for retrieving characters/words from Hanin dictionaries

unsigned char *Core[].fchar[]*

[IN] the 5 first found characters

char *AddrState*

[IN] flag to represent current use of address dictionary

short *Core[].clen[]*

[IN] length of characters in dictionaries

char *PininMode*

[IN] pinyin mode

unsigned char *inkey_bak*

[IN] backup of current input key

struct spec_key *spec_key*

[IN] structure data of special keys for 2-byte symbols

unsigned char *youwant[]*

[IN] Big5 code of Chinese characters in edit buffer

short *chlen*

[IN] current character length of *chbuf*

short *oldchlen*

[OUT/IN] previous character length of *chbuf*

RETURN VALUE

short *result*: 0: failed to find a word, others: character length of found words

MACRO IN USE

```
#define HANIN_CRYPT        0x2A
#define SEARCH_ONE         0
#define SEARCH_CHAR        0
#define SEARCH_ALL         1
```

CALLER FUNCTIONS

SearchAllChars, SearchOneChar

[3021 MDSearchWord]

This function searches words in system dictionary, address dictionary or professional dictionaries.

---

short MDSearchWord (short *nDic*, short *st*, short *len*, short *mode*)

ARGUMENTS

short *nDic*

[IN] index of available dictionaries

short *st*

[IN] specified index in edit buffer

short *len*

[IN] word length

short *mode*

[IN] search mode, 0 for search one word and 1 for search all words

SYNOPSIS

(1) get the compressed code of word from input edit buffer

(2) get the value of offset indicator regarding specified dictionary

(3) if the index of word is not 0, retrieve all the words in word range in dictionary

GLOBAL VARIABLES IN USE

long *outoff*

[IN] current value of offset indicator

char *Core[].mlenmark[]*

[IN/OUT] length mark of characters

char *AddrState*

[IN] flag to represent current use of address dictionary

unsigned char *cmpkigo[]*

[IN] the compressed code buffer

short *chlen*

[IN] current character length of *chbuf*

short *oldchlen*

[OUT/IN] previous character length of *chbuf*

long *Core[].widx[]*

[IN] index of word in dictionaries

short *Core[].wrange[]*

[IN] range for words in dictionaries

unsigned char *toutbuf[]*

[OUT] character buffer for retrieving characters/words from Hanin dictionaries

char *freq_fg*

[IN] word frequency

RETURN VALUE

short *result*: 0: failed to find a word, others: character length of found words

MACRO IN USE

```
#define TONE_COUNT          6
#define FRQ_FG              0x70
#define SEARCH_CHAR         0
#define SEARCH_WORD         1
#define HANIN_CRYPT         0x2A
```

CALLER FUNCTIONS

SearchAllWords, SearchOneWord

[3022 MDGetCharAddress]

This function gets the address of character in specified dictionary using the 1st and 2nd compressed codes.

---

short MDGetCharAddress (short *nDic*, short *st*, FILE **wfp*, short *mode*)

ARGUMENTS

short *nDic*

[IN] index of available dictionaries

short *st*

[IN] specified index in edit buffer

FILE **wfp*

[IN] pointer to the FILE object where data are read from

short *mode*

[IN] search mode, 0 for search characters and 1 for search words

SYNOPSIS

(1) if current offset already exists, return current offset
(2) get the index table of compressed code in specified dictionary by the 1st compressed code
(3) find the index of 1st compressed code, and get the offset of index table for 2nd compressed code
(4) retrieve index table of 2nd compressed code
(5) get the offset and length of characters or words

GLOBAL VARIABLES IN USE

long *outoff*

[IN] current value of offset indicator

IDX1 *mdIdxTable1[]*

[IN] index table in each dictionaries

unsigned char *cmpkigo[]*

[IN] the compressed code buffer

short *Core[].clen[]*

[IN] length for characters in dictionaries

long *Core[].cidx[]*

[OUT] index of character in dictionaries

short *Core[].wlen[]*

[OUT] length for words in dictionaries

RETURN VALUE

short *result*: -1: failed to match the 3rd compressed code, 0: found word address

MACRO IN USE

```
#define MD_IDX2_POS        (MD_ID_POS+MD_ID_LEN)
#define MD_MAX_IDX         58
#define SEARCH_CHAR        0
#define SEARCH_WORD        1
```

CALLER FUNCTIONS

MDSearchChar, MDSearchWord

[3023 MDGetWordAddress]

This function gets the address of specified word in specified dictionary using the 3rd compressed code.

---

short MDGetWordAddress (short *nDic*, short *st*, FILE **wfp*)

ARGUMENTS

short *nDic*

[IN] index of available dictionaries

short *st*

[IN] specified index in edit buffer

FILE **wfp*

[IN] pointer to the FILE object where data are read from

SYNOPSIS

(1) if flag of word length is OFF, return error code
(2) retrieve size of word length data into *idxBuf*
(3) matching the 3rd compressed code in *idxBuf*

GLOBAL VARIABLES IN USE

long *outoff*

[IN] current value of offset indicator

unsigned char *cmpkigo[]*

[IN] the compressed code buffer

short *Core[].wlen[]*

[IN] character length for words in dictionaries

long *Core[].widx[]*

[OUT] index of word in dictionaries

short *Core[].wrange[]*

[OUT] range for words in dictionaries

RETURN VALUE

short *result*: -1: failed to match the 3rd compressed code, 0: found word address

MACRO IN USE

```
#define TONE_COUNT          6
```

CALLER FUNCTIONS

MDSearchWord

[3024 BinarySearch]

This function conducts the binary search to the specified item in specified memory area.

---

short BinarySearch (PBYTE *table*, BYTE *item*, short *low*, short *high*, short *size*)

ARGUMENTS

PBYTE *table*

[IN] memory area to be searched

BYTE *item*

[IN] target item to be searched

short *low*

[IN] left index for searching

short *high*

[IN] right index for searching

short *size*

[IN] the offset for item in *table*

SYNOPSIS

(1) do the binary search in *table* to find the same data as *item*.

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

short *result*: -1: failed to find *item* in *table*, others: index of *item* in *table*

MACRO IN USE

N/A

CALLER FUNCTIONS

MDSearchFChar, MDGetCharAddress

[3025 CompareCmpcode]

This function compares two compressed code buffer.

---

short CompareCmpcode (PBYTE *cmpCode1*, PBYTE *cmpCode2*, short *len*)

ARGUMENTS

PBYTE *cmpCode1*
[IN] compressed code buffer to be compared
PBYTE *cmpCode2*
[IN] another compressed code buffer to be compared
short *len*
[IN] character length to compare

SYNOPSIS

(1) compare the first character of each string. If they are equal to each other, it continues with the following pairs until the characters differ or until a terminating null-character is reached

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

short *result*: 0: both strings are equal, positive number: *cmpCode1* has a greater value than *cmpCode2*, negative number: *cmpCode2* has a greater value than *cmpCode1*

MACRO IN USE

```
#define TONE_COUNT          6
```

CALLER FUNCTIONS

MDSearchWord

[3026 ReadBuffer]

This function retrieves specified number of data from specified file.

---

short ReadBuffer (PBYTE *buf*, FILE **wfp*, short *num*)

ARGUMENTS

PBYTE *buf*

[IN] pointer to target character buffer

FILE **wfp*

[IN] pointer to the FILE object where characters are read from

short *num*

[IN] number of bytes to be read

SYNOPSIS

(1) read *num* bytes of data for *wfp* to character buffer *buf*

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

int *numRead*: character length to be read

MACRO IN USE

N/A

CALLER FUNCTIONS

MDSearchChar, MDSearchWord, MDGetWordAddress

[3027 RepSearchWord]

This function deals with conversion rule of repeated word.

short RepSearchWord (short *st*)

ARGUMENTS

short *st*

[IN] specified index in edit buffer

SYNOPSIS

(1) compare the compressed code in edit buffer with compressed code of specified repeated word

(2) update the character in edit buffer if it follows the rule of repeated word

GLOBAL VARIABLES IN USE

char *AddrState*

[IN] flag to represent current use of address dictionary

char *Core[].flag*

[IN] change flag for specified character

unsigned char *cmpkigo[]*

[IN] the compressed code buffer

unsigned char *chbuf[]*

[IN] choice buffer for selecting homophone characters/words

unsigned char *youwant[]*

[IN] Big5 code of Chinese characters in edit buffer

short *chlen*

[IN] current character length of *chbuf*

RETURN VALUE

int *chlen*: 0: *AddrState* is ON, or length flag is greater than 1
others: character length of *chbuf*

MACRO IN USE

```
#define LEN_FG              0x07
```

CALLER FUNCTIONS

SearchOneWord

[3028 two_invert]

This function swaps the first character and the second character of a character buffer.

void two_invert (unsigned char *s)

ARGUMENTS

unsigned char *s

[IN/OUT] character buffer

SYNOPSIS

(1) swap the first character and 2 second character in *s*

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

void

MACRO IN USE

N/A

CALLER FUNCTIONS

sear_aimei, MDGetCharAddress, ckUDLoad, UDWrite, UDLoad, SaveUerWord, ReadBig5E

[3029 UDGetHashIndex]

This function gets the index of hash table for specified compressed code.

short UDGetHashIndex (PBYTE *cmpCode*)

ARGUMENTS

PBYTE *cmpCode*

[IN] specified compressed code

SYNOPSIS

(1) calculate the hash index using compressed code

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

int *index*: hash index

MACRO IN USE

```
#define UD_MAX_IDX          47
```

CALLER FUNCTIONS

UDSearchWord, UDInsertWord, UDWrite

[3030 UDSearchWord]

This function searches specified index and length in edit buffer in user dictionary.

---

short UDSearchWord (short *st*, short *len*, short *mode*)

ARGUMENTS

short *st*
    [IN] specified index in edit buffer
short *len*
    [IN] specified character length in edit buffer
short *mode*
    [IN] search mode

SYNOPSIS

(1) get the compressed code and character code from edit buffer
(2) get the index of hash table regarding compressed code
(3) if the compressed code is identical to compressed code in user dictionary, set the flag of user word length mark
(4) append found user defined word into character buffer

GLOBAL VARIABLES IN USE

char *user_dic*
    [IN] full path filename of user dictionary
short *udHashIdx[]*
    [IN] index of hash table for user defined words
unsigned char *toutbuf[]*

[OUT] character buffer for retrieving characters/words from Hanin dictionaries

char *Core[].ulenmark*

[IN/OUT] length mark for user defined word

RETURN VALUE

int *length*: 0: length mark of user defined word is 0, or failed to open user dictionary, or range of user defined word overflow
others: changed length of character buffer

MACRO IN USE

```
#define USR_UD_POS          (USR_LD_POS+USR_LD_LEN)
#define UD_IDX_SIZE         ((UD_MAX_IDX+1)*2)
#define SEARCH_ALL          1
#define FRQ_FG              0x70
```

CALLER FUNCTIONS

SearchAllWords, SearchOneWord

[3031: UDInsertWord]

This function inserts an additional user defined word into user dictionary.

---

short UDInsertWord (PBYTE *cmpCode*, PBYTE *bigCode*, short *len*)

ARGUMENTS

PBYTE *cmpCode*

[IN] compressed code of user defined word

PBYTE *bigChar*

[IN] character code of user defined word

int *len*

[IN] character length of user defined word

SYNOPSIS

(1) check the availability of user defined word to be inserted
(2) check the existence of user defined word in user dictionary
(3) insert specified user defined word into user dictionary

GLOBAL VARIABLES IN USE

char *user_dic*

[IN] full path filename of user dictionary

short *udHashIdx[]*

[IN] index of hash table for user defined words

unsigned char *toutbuf[]*

[OUT] character buffer for retrieving characters/words from Hanin dictionaries

RETURN VALUE

int *result*: RC_SUCCESS: success
RC_INVALIDWORD: invalid Chinese characters
RC_USERWORDFULL: user words overflow
RC_OTHERERROR: failed to open user dictionary
RC_WORDEXIST: user word already exists in user dictionary

MACRO IN USE

```
#define RC_SUCCESS          0
#define RC_OTHERERROR       (-1)
#define RC_WORDEXIST        (-2)
#define RC_USERWORDFULL     (-3)
#define RC_INVALIDWORD      (-4)
#define UD_MAX_IDX          47
#define UD_MAX_SIZE         16384
#define USR_UD_POS          (USR_LD_POS+USR_LD_LEN)
#define UD_IDX_SIZE         ((UD_MAX_IDX+1)*2)
#define HANIN_CRYPT         0x2A
```

CALLER FUNCTIONS

HAN_RegUserWord

[3032: UDWrite]

This function writes the specified user defined word into user dictionary.

---

void UDWrite (PBYTE *cmpCode*, PBYTE *bigCode*, short *len*)

ARGUMENTS

PBYTE *cmpCode*

[IN] compressed code of user defined word

PBYTE *bigChar*

[IN] character code of user defined word

int *len*

[IN] character length of user defined word

SYNOPSIS

(1) open user dictionary, and retrieve index of hash table

(2) find the location to write the specified user defined word, and write compressed code and character code into dictionary

(3) update the index of hash table, and write index into dictionary

GLOBAL VARIABLES IN USE

char *user_dic*

[IN] full path filename of user dictionary

short *udHashIdx[]*

[IN] index of hash table for user defined words

unsigned char *toutbuf[]*

[OUT] character buffer for retrieving characters/words from Hanin dictionaries

RETURN VALUE

void

MACRO IN USE

```
#define USR_UD_POS          (USR_LD_POS+USR_LD_LEN)
#define UD_IDX_SIZE         ((UD_MAX_IDX+1)*2)
#define UD_MAX_IDX          47
#define HANIN_CRYPT         0x2A
```

CALLER FUNCTIONS

UDInsertWord

[3033: q_extend]

This function compares input base text. If it's equal to abbreviation records in user defined words, retrieves the long form text and short form text from user dictionary.

---

short q_extend (*void*)

ARGUMENTS

void

SYNOPSIS

(1) open user dictionary, and retrieve index of abbreviations
(2) compare input base text with index of abbreviations
(3) if it's identical, retrieve long form text and short form text

GLOBAL VARIABLES IN USE

char *user_dic*
[IN] full path filename of user dictionary
unsigned char *toutbuf[]*
[OUT] character buffer for retrieving characters/words from Hanin dictionaries
unsigned char *youwant[]*
[IN] Big5 code of Chinese characters in edit buffer
unsigned char *chbuf[]*
[IN] choice buffer for selecting homophone characters/words

RETURN VALUE

int *result*: 0: failed to open user dictionary, or indx overflow, or wrong index length
1: success

MACRO IN USE

```
#define USR_AD_POS          (USR_UD_POS+USR_UD_LEN)
#define AD_IDX_SIZE         (AD_MAX_WORD*6)
#define HANIN_CRYPT         0x2A
```

CALLER FUNCTIONS

HAN_GetExtension

[3034: UDLoad]

This function opens the user dictionary and retrieves hash index of user defined words and hash buffer of character code

---

short UDLoad (*void*)

ARGUMENTS

void

SYNOPSIS

(1) open user dictionary
(2) retrieve index into *udHashIdx*, and retrieve user defined words into *udHashBuf*

GLOBAL VARIABLES IN USE

char *user_dic*
[IN] full path filename of user dictionary
short *udHashIdx[]*
[IN] index of hash table for user defined words
unsigned char *udHashBuf[]*
[IN] hash buffer for user defined words

RETURN VALUE

int *count*: -1: failed to open user dictionary
0: success

MACRO IN USE

```
#define UD_IDX_SIZE        ((UD_MAX_IDX+1)*2)
#define UD_MAX_SIZE        16384
#define USR_UD_POS         (USR_LD_POS+USR_LD_LEN)
```

CALLER FUNCTIONS

HAN_BeginUWordMaintenance

[3035: hash_to_sort]

This function assigns the user defined words in hash buffer into a allocated sorted list.

---

int hash_to_sort (*void*)

ARGUMENTS

void

SYNOPSIS

(1) allocate memory area for *sort_list*
(2) copy compressed code and character code of user defined word into *sort_list*

GLOBAL VARIABLES IN USE

short *sort_num*

[IN] amount of records in *sort_list*

PSORTEDLIST *sort_list*

[IN/OUT] pointer to memory area of struct SORTEDLIST

short *udHashIdx[]*

[IN] index of hash table for user defined words

unsigned char *udHashBuf[]*

[IN] hash buffer for user defined words

RETURN VALUE

int *count*: amount of records in *sort_list*

MACRO IN USE

```
#define UD_MAX_IDX          47
#define UD_MAX_SIZE         16384
#define HANIN_CRYPT2A       0x2A
```

CALLER FUNCTIONS

HAN_BeginUWordMaintenance

[3036: insert_list]

This function inserts an additional record into *sort_list* by ascending order

---

void insert_list (PBYTE *in_code*, PBYTE *in_char*, int *strSize*)

ARGUMENTS

PBYTE *in_code*

[IN] compressed code of additional user defined word

PBYTE *in_char*

[IN] character code of additional user defined word

int *strSize*

[IN] character length of user defined word

SYNOPSIS

(1) compare compressed code of additional user defined word with compressed code in *sort_list*

(2) get the index to insert a additional user defined word

(3) copy compressed code and character code of additional user defined word into *sort_list*

GLOBAL VARIABLES IN USE

short *sort_num*

[IN] amount of records in *sort_list*

PSORTEDLIST *sort_list*

[IN/OUT] pointer to memory area of struct SORTEDLIST

RETURN VALUE

void

MACRO IN USE

N/A

CALLER FUNCTIONS

hash_to_sort

[3037: Mk2_Word]

This function assigns records in *sorted_list* to hash buffer *udHashBuf*.

---

void SaveUerWord (*void*)

ARGUMENTS

void

SYNOPSIS

(1) store sorted records of user defined words into hash buffer *udHashBuf*

GLOBAL VARIABLES IN USE

short *udHashIdx[]*

[IN] index of hash table for user defined words

unsigned char *udHashBuf[]*

[IN] hash buffer for user defined words

PSORTEDLIST *sort_list*

[OUT] pointer to memory area of struct SORTEDLIST

RETURN VALUE

void

MACRO IN USE

```
#define UD_MAX_IDX          47
#define HANIN_CryPT2A       0x2A
```

CALLER FUNCTIONS

SaveUerWord

[3038: GetUWordIndex]

This function get the nearest record of user defined word to the index of input zhuyin fuhao symbol.

---

short SaveUerWord (short *nn*)

ARGUMENTS

short *nn*

[IN] index of zhuyin fuhao

SYNOPSIS

(1) get the compressed code by zhuyin fuhao index *nn*
(2) find the index in sorted list which the value of compressed code is less than compressed code in sorted list

GLOBAL VARIABLES IN USE

PSORTEDLIST *sort_list*

[OUT] pointer to memory area of struct SORTEDLIST

RETURN VALUE

short *index*: nearest index to input zhuyin fuhao

MACRO IN USE

N/A

CALLER FUNCTIONS

HAN_GetUserWordStartIndex

[3039: SaveUerWord]

This function writes current records of user defined words and index hash table into user dictionary

---

void SaveUerWord (*void*)

ARGUMENTS

void

SYNOPSIS

(1) store sorted records of user defined words into hash buffer *udHashBuf*
(2) write index of hash buffer into user dictionary
(3) write hash buffer into user dictionary
(4) free allocated memory for sorted records *sort_list*

GLOBAL VARIABLES IN USE

char *user_dic*
[IN] full path filename of user dictionary
short *udHashIdx[]*
[IN] index of hash table for user defined words
unsigned char *udHashBuf[]*
[IN] hash buffer for user defined words
PSORTEDLIST *sort_list*
[OUT] pointer to memory area of struct SORTEDLIST

RETURN VALUE

void

MACRO IN USE

```
#define USR_UD_POS          (USR_LD_POS + USR_LD_LEN)
#define UD_IDX_SIZE         ((UD_IDX_SIZE+1)*2)
#define UD_MAX_IDX          47
```

CALLER FUNCTIONS

HAN_EndUWordMaintenance

[3040: GetFreeWords]

This function counts the number of available rooms for registering user defined words.

short GetFreeWords (*void*)

ARGUMENTS

void

SYNOPSIS

(1) count the number of vacancy area for user defined words

GLOBAL VARIABLES IN USE

short *udHashIdx[]*

[IN] index of hash table for user defined words

RETURN VALUE

short *count*: number of vacancy rooms

MACRO IN USE

```
#define UD_MAX_IDX          47
#define UD_MAX_SIZE         16384
```

CALLER FUNCTIONS

HAN_GetFreeUserChars

[3041: do_omit]

This function skips specified data in a record.

void do_omit (short *nn*)

ARGUMENTS

short *nn*

[IN] index of removing record

SYNOPSIS

(1) delete the specified record from *sort_list*

GLOBAL VARIABLES IN USE

PSORTEDLIST *sort_list*

[OUT] pointer to memory area of struct SORTEDLIST

RETURN VALUE

void

MACRO IN USE

N/A

CALLER FUNCTIONS

HAN_DelUserWord

[3042: GetJuinList]

This function gives a list of zhuyin character code to a character string.

---

short GetJuinList (PBYTE *pJuin*)

ARGUMENTS

PBYTE *pJuin*

[OUT] character string for zhuyin list

SYNOPSIS

(1) set the zhuyin Big5 code into *pJuin* one by one

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

short *len*: length of zhuyin character string

MACRO IN USE

N/A

CALLER FUNCTIONS

HAN_GetZhuinList

[3043: strcmp2]

This function compares two character strings.

int strcmp2 (char *s, char *t)

ARGUMENTS

char *s

[IN] character string to be compared

char *t

[IN] another character string to be compared

SYNOPSIS

(1) compare the first character of each string. If they are equal to each other, it continues with the following pairs until the characters differ or until a terminating null-character is reached

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

short *result*:	0: both strings are equal, positive number: *s* has a greater value than *t*, negative number: *t* has a greater value than *s*

MACRO IN USE

N/A

CALLER FUNCTIONS

insert_list, GetUWordIndex

[3044: DelExt]

This function removes the specified abbreviation from user dictionary

short DelExt (PBYTE *extbuf*)

ARGUMENTS

PBYTE *extbuf*

[IN] the input text

SYNOPSIS

(1) find the index of abbreviation regarding *extbuf*

(2) write an empty index into *idxbuff* and user dictionary

GLOBAL VARIABLES IN USE

char *user_dic*

[IN] full path filename of user dictionary

RETURN VALUE

short *ret*: 0: success, negative number: error occurred

MACRO IN USE

```
#define RC_SUCCESS         0
#define RC_OTHERERROR      (-1)
#define RC_INVALIDEXTID    (-2)
#define RC_NOLONGEXT       (-3)
#define RC_ALREADYEXIST    (-4)
#define RC_NOTFOUND        (-6)
#define OFF_QUICKIDX       17528
#define OFF_QUICK          19068
#define HANIN_CRYPT2A      0x2A
```

CALLER FUNCTIONS

HAN_RegExtension

[3045: RegExt]

This function registers a new abbreviation into user dictionary

---

short RegExt (PBYTE *extbuf*, PEXTENSION *pExtension*)

ARGUMENTS

PBYTE *extbuf*

[IN] the input text

PEXTENSION *pExtension*

[IN] new abbreviation to be registered

SYNOPSIS

(1) validate the base text, long form text, and short form text of registering abbreviation
(2) write *pExtension* into user dictionary if it's available

GLOBAL VARIABLES IN USE

char *user_dic*
[IN] full path filename of user dictionary
unsigned char *idxbuff[]*
[OUT] index buffer for abbreviations

RETURN VALUE

short *ret*: 0: success, negative number: error occurred

MACRO IN USE

```
#define RC_SUCCESS          0
#define RC_OTHERERROR       (-1)
#define RC_NOTFOUND         (-6)
#define OFF_QUICKIDX        17528
```

CALLER FUNCTIONS

HAN_DelExtension

[3046: FindExtID]

This function finds the index of abbreviation that has the same base text as input text.

---

short FindExtID (PBYTE *extbuf*)

ARGUMENTS

PBYTE *extbuf*
[IN] the input text

SYNOPSIS

(1) compare the input text with base text in *idxbuff*
(2) return the index if found

GLOBAL VARIABLES IN USE

unsigned char *idxbuff[]*
[IN] index buffer for abbreviations

RETURN VALUE

short *index*: if *index* > 0, found the abbreviation record
if *index* < 0, error occurred

MACRO IN USE

```
#define QD_IDXTABLE_SIZE   1536
#define QD_IDX_SIZE        6
#define RC_RANGEERROR      (-101)
#define RC_INVALIDEXTID    (-2)
#define RC_NOTFOUND        (-6)
```

CALLER FUNCTIONS

HAN_FindExtension, DelExt, RegExt

[3047: GetExt]

This function retrieves the specified index of abbreviation includes long form text and short form text from user dictionary.

---

short GetExt (short *idx*, PEXTENSION *pExtension*)

ARGUMENTS

short *idx*
[IN] specified index of abbreviation to be retrieved
PEXTENSION *pExtension*
[OUT] retrieved abbreviation record

SYNOPSIS

(1) set the start position of abbreviation area in user dictionary
(2) retrive long form text and short form text of specified abbreviation

GLOBAL VARIABLES IN USE

char *user_dic*
[IN] full path filename of user dictionary

RETURN VALUE

short *result*: 0: failed, 1: success

MACRO IN USE

```
#define AD_MAX_WORD        256
#define OFF_QUICK          19068
#define HANIN_CRYPT2A      0x2A
```

CALLER FUNCTIONS

HAN_FindExtension

[3048: GetExtVacancy]

This function gets the first available index for registering a new abbreviation.

---

short GetExtVacancy (*void*)

ARGUMENTS

void

SYNOPSIS

(1) check the available room for registering a new abbreviation

GLOBAL VARIABLES IN USE

unsigned char *idxbuff[]*

[IN] index buffer for abbreviations

RETURN VALUE

short *result*: (-1): no room for registering a new abbreviation, others: index of available room

MACRO IN USE

```
#define AD_MAX_WORD        256
```

CALLER FUNCTIONS

RegExt

[4001: SpecialRuleMatch]

This function executes the special rules one by one

short SpecialRuleMatch (*void*)

ARGUMENTS

void

SYNOPSIS

(1) One by one conduct Hanin special matching rules.

GLOBAL VARIABLES IN USE

unsigned char *rule_A1[][6]*
[IN] records for special rule A1
short *RULE_A2_COUNT*
[IN] amount of records of *rule_A1* buffer
unsigned char *rule_A2[][8]*
[IN] records for special rule A2
short *RULE_A2_COUNT*
[IN] amount of records of *rule_A2* buffer
unsigned char *rule_B[][6]*
[IN] records for special rule B
short *RULE_B_COUNT*
[IN] amount of records of *rule_B* buffer
unsigned char *title[][4]*
[IN] records for titles
unsigned char *surname[][4]*
[IN] records for surnames

RETURN VALUE

short *result*: 0: failed to match, 1: success

MACRO IN USE

```
#define WRD_FG          0x08
#define CHG_FG          0x80
#define W_START         21
#define NOR             0
```

CALLER FUNCTIONS

realtime

[4002: MatchRuleA]

This function conducts a special rule for matching unit character regarding specific noun, and a special rule for matching adverb character regarding specific noun.

---

short MatchRuleB (PBYTE *rule*, short *count*, short *len*)

ARGUMENTS

PBYTE *rule*

[IN] pointer to character buffer of a special rule

short *count*

[IN] number of items in above *rule* buffer

short *len*

[IN] matching length

SYNOPSIS

(1) check the *len*-byte character in edit buffer is identical to the first *len*-byte character in specified *rule* buffer

(2) if it's identical, revise the character when the compressed code is identical

GLOBAL VARIABLES IN USE

char *f_want*

[IN] the cursor position in edit buffer

unsigned char *youwant[]*

[IN/OUT] Big5 code of Chinese characters in edit buffer

unsigned char *cmpkigo[]*

[IN] the compressed code buffer

char *Core[].flag*

[IN] change flag for specified character

RETURN VALUE

short *result*: 0: failed to match, 1: success

MACRO IN USE

```
#define WRD_FG          0x08
#define CHG_FG          0x80
#define W_START         21
#define NOR             0
```

CALLER FUNCTIONS

SpecialRuleMatch

[4003: MatchRuleB]

This function conducts the special rule for matching one character verb regarding specific noun

---

short MatchRuleB (PBYTE *rule*, short *count*, short *len*)

ARGUMENTS

PBYTE *rule*

[IN] character buffer of a rule for specific verb and noun

short *count*

[IN] number of items in *rule_B* buffer

short *len*

[IN] matching length

SYNOPSIS

(1) check the *len*-byte character in edit buffer is identical to the first *len*-byte character in *rule_B* buffer

(2) if it's identical, revise the character when the compressed code is identical

GLOBAL VARIABLES IN USE

char *f_want*

[IN] the cursor position in edit buffer

unsigned char *youwant[]*

[IN/OUT] Big5 code of Chinese characters in edit buffer

unsigned char *cmpkigo[]*

[IN] the compressed code buffer

char *Core[].flag*

[IN] change flag for specified character

RETURN VALUE

short *result*: 0: failed to match, 1: success

MACRO IN USE

```
#define W_START            21
```

```
#define NOR                 0
```

CALLER FUNCTIONS

SpecialRuleMatch

[4004: MatchRuleC]

This function conducts the special rule for matching title and surname.

---

short MatchRuleC (PBYTE *st*, PBYTE *ss*)

ARGUMENTS

PBYTE *st*

[IN] character buffer of position titles

PBYTE *ss*

[IN] character buffer of surname and its compressed code

SYNOPSIS

(1) conducts the special rule for matching title in *title* buffer

(2) search the Chinese character 1 character before the title in *surname* buffer

GLOBAL VARIABLES IN USE

char *f_want*

[IN] the cursor position in edit buffer

short *RULE_C1_COUNT*

[IN] number of titles in *title* buffer

unsigned char *youwant[]*

[IN/OUT] Big5 code of Chinese characters in edit buffer

short *RULE_C2_COUNT*

[IN] number of surnames in *surname* buffer

unsigned char *cmpkigo[]*

[IN] the compressed code buffer

char *Core[].flag*

[IN] change flag for specified character

RETURN VALUE

short *result*: 0: failed to match, 1: success

MACRO IN USE

```
#define WRD_FG              0x08
#define CHG_FG              0x80
#define W_START             21
#define NOR                 0
```

CALLER FUNCTIONS

SpecialRuleMatch

[4005: UnitSearchWord]

This function revises input Chinese characters by conducting algorithms to match special rules for adverb characters, measure words, unit characters, repeated characters.

---

short UnitSearchWord (short *st*, short *len*, short *mode*)

ARGUMENTS

short *st*

[IN] index to indicate the start position of *cmpkigo* buffer

short *len*

[IN] character length to be revised by special rules

short *mode*

[IN] search mode

SYNOPSIS

(1) find the same compressed code in records of struct Unit *fst_unit* in the first stage

(2) find the same following compressed code of *cmpkigo* buffer in records of struct Uint *fst_num_unit* and struct Unit *snd_num_unit*

(3) if address mode is ON, check the struct Unit *adr_snd_unit*; otherwise, check the struct Unit *sys_snd_unit* for unit characters

(4) set the flag *freq_fg* and *word_fg*

GLOBAL VARIABLES IN USE

short *chlen*

[IN] current character length of *chbuf*

short *oldchlen*

[OUT] previous character length of *chbuf*

struct Unit *fst_unit[]*

[IN] records of compressed code and character code for adverb

short *FST_UNIT_HI*

[IN] amount of records of structure *fst_unit*

unsigned char *cmpkigo[]*

[IN] the compressed code buffer

struct Unit *fst_num_unit[]*

[IN] compressed code and character code for 2-byte numerical symbols

short *FST_NUM_HI*

[IN] amount of records of structure *fst_num_unit*

struct Unit *snd_num_unit[]*

[IN] compressed code and character code for numerical unit characters such as “百”, “千”, “萬”

short *SND_NUM_HI*

[IN] amount of records of structure *snd_num_unit*

struct Unit *adr_snd_unit*

[IN] specific unit characters are used when `AddrState` is ON.

short *ADR_SND_UNIT_HI*

[IN] amount of unit records of *adr_snd_unit* which is used when `AddrState` is ON.

struct Unit *sys_snd_unit*

[IN] specific unit characters

short *SYS_SND_UNIT_HI*

[IN] amount of unit records of *sys_snd_unit*

char *freq_fg*

[IN] word frequency

char *word_fg*

[IN] flag to indicate the status of Chinese word conversion

RETURN VALUE

short *chglen*: changed length of *chbuf*

MACRO IN USE

```
#define HANIN_CRYPT        0x2A
#define NUMSPEC_A          1
#define NUMSPEC_B          2
#define NUMSPEC_C          16
#define TONE_COUNT         6
#define SEARCH_ALL         1
```

CALLER FUNCTIONS

SearchAllWords, SearchOneWord

[4006: fdan]

This function compares the cmpkigo of input character with specified records of strcut Unit.

---

short fdan (char *buf[]*, **struct Unit **unit_ptr*, **short** *hi*)**

ARGUMENTS

char *buf*

[IN] compressed code buffer to be compared

struct Unit **unit_ptr*

[IN] member of compressed code to be compared with *buf*

short *hi*

[IN] the maximum items in structure *unit_ptr* to be compared

SYNOPSIS

(1) find the identical compressed code in records of *unit_ptr* to compressed code of *buf*

GLOBAL VARIABLES IN USE

N/A

RETURN VALUE

short *index*: (-1): failed to find the same compressed code, others: index of matched record

MACRO IN USE

N/A

CALLER FUNCTIONS

UnitSearchWord, chk_unit, Spatch

[4007: fndx]

This function retrieves the matched adverb character and its frequency.

short fndx (short *st*)

ARGUMENTS

short *st*

[IN] index to indicate the start position of *cmpkigo* buffer

SYNOPSIS

(1) find the same cmpkigo in records of struct frestr *fre*

(2) get the code of found adverb and its frequency.

GLOBAL VARIABLES IN USE

char *AddrState*

[IN] flag to represent current use of address dictionary

unsigned char *liang[3]*

[OUT] character buffer to store adverb character and its frequency

struct frestr *fre[]*

[IN] compressed code, character code and frequency for measure words

short *FRE_HI*

[IN] amount of records of structure *fre*

unsigned char *cmpkigo[]*

[IN] the compressed code buffer

RETURN VALUE

short *index*: (-1): failed to match or *AddrState* is ON or negative index for *cmpkigo*, others: index of matched record

MACRO IN USE

```
#define FRE_LO              0
```

CALLER FUNCTIONS

match

[4008: chk_unit]

This function checks the status of special rule for measure words.

short chk_unit (short *k*)

ARGUMENTS

short *k*

[IN] index to indicate the start position of *cmpkigo* buffer

SYNOPSIS

(1) copy string **(ss+j)* of Chinese word into **(youwant+st)*

(2) set the word change flag to be ON

GLOBAL VARIABLES IN USE

struct Unit *fst_num_unit[]*

[IN] compressed code and character code for 2-byte numerical symbols

short *FST_NUM_HI*

[IN] amount of records of structure *fst_num_unit*

struct Unit *fst_unit[]*

[IN] compressed code and character code for some adverbs

short *FST_UNIT_HI*

[IN] amount of records of structure *fst_unit*

unsigned char *cmpkigo[]*

[IN] the compressed code buffer

RETURN VALUE

short *status*: (-1): failed to match, others: index of matched record

MACRO IN USE

N/A

CALLER FUNCTIONS

MatchRuleA, MatchRuleC

[4009: Spatch]

This function conducts special rules such as adverb, measure words regarding sentence in edit buffer.

---

short Spatch (*void*)

ARGUMENTS

void

SYNOPSIS

(1) conduct special rule for patching characters regarding 2-byte fuhao characters
(2) conduct special rule for repeated Chinese characters “又” and “越”
(3) deal with special rule for Chinese character “的”
(4) deal with special rule for measure words and quantifiers

GLOBAL VARIABLES IN USE

char *Core[].flag*
[IN] change flag for specified character
char *f_want*
[IN] the cursor position in edit buffer
unsigned char *youwant[]*
[OUT] Big5 code of Chinese characters in edit buffer
unsigned char *cmpkigo[]*
[IN] the compressed code buffer
unsigned char *spatch_char[][4]*
[IN] compressed code and character code for patching characters
short *SPATCH_CCHAR*
[IN] amount of *spatch_char* buffer
unsigned char *spatch_de[][4]*
[IN] compressed code and character code for patching characters
short *SPATCH_DE_COUNT*
[IN] amount of *spatch_de* buffer
unsigned char *spatch_data[][8]*
[IN] compressed code and character code for patching characters
short *SPATCH_COUNT*
[IN] amount of *spatch_data* buffer
struct Unit *fst_num_unit[]*
[IN] compressed code and character code for 2-byte numerical symbols
short *FST_NUM_HI*
[IN] amount of records of structure *fst_num_unit*
struct Unit *snd_num_unit[]*
[IN] compressed code and character code for numerical unit characters such as “百”, “千”, “萬”
short *SND_NUM_HI*
[IN] amount of records of structure *snd_num_unit*

RETURN VALUE

short *ret*: 0: failed to match, 1: success

MACRO IN USE

```
#define WRD_FG              0x08
#define CHG_FG              0x80
#define W_START             21
#define NOR                 0
#define NUMSPEC_A           1
#define NUMSPEC_B           2
#define NUMSPEC_C           16
#define TONE_COUNT          6
#define HANIN_CRYPT         0x2A
```

CALLER FUNCTIONS

SpecialRuleMatch

[4010: ReDisplay]

This function replaces the Chinese characters in *youwant* buffer when input sentence conform a special rule

---

void ReDisplay (short *st*, unsigned char *ss, short *j*)

ARGUMENTS

short *st*

[IN] index to indicate the start position of Chinese word to be replaced

unsigned char **ss*

[IN] pointer to the string from which characters will be copied

short *j*

[IN] index of **ss* to indicate the start position will be copied

SYNOPSIS

(1) copy string **(ss+j)* of Chinese word into **(youwant+st)*

(2) set the word change flag to be ON

GLOBAL VARIABLES IN USE

char *Core[].flag*

[OUT] change flag for specified character

unsigned char *youwant[]*

[OUT] Big5 code of Chinese characters in edit buffer

RETURN VALUE

void

MACRO IN USE

```
#define WRD_FG              0x08
#define W_START             21
#define NOR                 0
```

CALLER FUNCTIONS

Spatch

[4011: do_rep]

This function conducts special rule for repeated Chinese characters.

---

short do_rep (void)

ARGUMENTS

void

SYNOPSIS

(1) execute a special rule of repeated Chinese character in a sentence, such as 越飛越高.

(2) check 2 repeated Chinese characters: 又 (Big5: 0xA453), 越 (Big5: 0xB656)

GLOBAL VARIABLES IN USE

short *SREP_WCOUNT*

[IN] amount of records of *srep_word*

unsigned char *srep_word*

[IN] specified characters for rule of repeated Chinese character

char *f_want*

[IN] the cursor position in edit buffer

char *Core[].flag*

[IN] change flag for specified character

unsigned char *youwant[]*

[IN/OUT] Big5 code of Chinese characters in edit buffer

unsigned char *cmpkigo[]*

[IN] the compressed code buffer

RETURN VALUE

short *found*: 0: failed, 1: found

MACRO IN USE

```
#define CHG_FG              0x80
#define W_START             21
#define NOR                 0
```

CALLER FUNCTIONS

Spatch

[4012: AddCharMatch]

If address status is on, it looks up the identical compressed code to matching character. To append the found character to choice buffer when found the identical compressed code.

---

short AddrCharMatch (short *st*)

ARGUMENTS

st

[IN] index to indicate the position of the first compressed code for matching character

SYNOPSIS

(1) if the compressed code of matching character is identical to the a member in struct Unit *adr_snd_unit*, then append its character code to *chbuf*

GLOBAL VARIABLES IN USE

unsigned char *cmpkigo[]*

[IN] the compressed code buffer

short *ADR_SND_UNIT_HI*

[IN] amount of unit records of *adr_snd_unit* which is used when `AddrState` is ON.

struct Unit *adr_snd_unit*

[IN] specific unit characters are used when `AddrState` is ON.

RETURN VALUE

short *chlen*: character length (in byte) to be appended

MACRO IN USE

```
#define HANIN_CRYPT        0x2A
#define SEARCH_ONE         0
```

CALLER FUNCTIONS

SearchOneChar