Programmer's Guide for Fingerprint's SDK

Wison Technology Corp.

Addr: 11F-2, No. 289, Sec. 2, Guang-Fu Rd., Hsin-Chu 300, Taiwan, R.O.C.
Tel: 886-3-5163339
Fax: 886-3-5163679
Email : raymond@wison.com.tw

1. File needed to run the program

A. OR100 (Optical CMOS Reader)

I. *WIS_API.Dll & WISCMS12.Dll*

II. *WISCMS12.INF & WISCMS12.SYS (Driver for OR100)*

B. OR200 (Optical CMOS Reader)

I. *WIS_API.Dll & WISCMOS2.Dll*

II. *WISCMOS2.INF & WISCMOS2.SYS (Driver for OR200)*

C. OR100-R (Optical CMOS Reader with RoHS)

I. *WIS_API.Dll & WISCMS1R.Dll*

II. *WISCMS1R.INF & WISCMS1R.SYS (Driver for OR100-R)*

2. Function List

***Please load the WIS_API.dll and include the WIS_API.h to use the API below*:**

Functions of Device Control	
WIS_InitDriver	WIS_TerminateDriver
Functions of Device Diagnosis	
WIS_TestDevice	WIS_CheckNoFinger
Functions of Fingerprint Capture and Feature Extraction	
WIS_Snap	WIS_Capture
WIS_InitCapture	WIS_EndCapture
WIS_CreateTemplate	
Functions of Fingerprint Image Access	
WIS_GetImage	WIS_GetImageSize
WIS_SaveImage	WIS_DisplayImage
WIS_SetParameter	
Functions of Enrollment	
WIS_Enroll	WIS_ReleaseEnroll
WIS_SetEnrollMode	
Functions of Verification	
WIS_VerifyTemplate	WIS_VerifyTemplateAllAngle (*New)
WIS_StartIdentify (*New)	WIS_Identify (*New)
WIS_IdentifyResult (*New)	WIS_ReleaseIdentify (*New)

Note:

- ***For Borland C++ Builder developers,***
  1. *Please type in the command mode "implib WIS_Api.lib WIS_Api.dll" to get the library needed for BCB.*
  2. *Add the generated WIS_Api.lib to your project.*
  3. *To run the program, please make sure that CP3240MT.DLL and VCL35.BPL have existed.*

- ***For VB6 .Net developers,***

  All the code is compatible with VB except the graphics that is used to display the fingerprint image. Please refer to the code below:

```
Dim m_hDC As Integer
Dim g_pic_graphics As Graphics
Dim g_intptr_hdc As IntPtr

g_pic_graphics = Picture1.CreateGraphics
g_intptr_hdc = g_pic_graphics.GetHdc()
m_hDC = g_intptr_hdc.ToInt32
```

  whererrem_hDC *will be used later in WIS_DisplayImage as the device context for displaying the image.*

3. Function Description

WIS_InitDriver

Synopsis

HANDLE WINAPI WIS_InitDriver(int device)

Description

The **WIS_InitDriver()** connects the capture driver of the fingerprint device. Please connect the capture driver when your program is initialized, and disconnect the capture driver before terminating your program.

Parameter

device

1. OR100 : CMOS (defined in “WIS_API.h”)
2. OR200 : CMOS2 (defined in “WIS_API.h”)
3. OR100-R : CMOSROHS (defined in “WIS_API.h”)

Return Value

i. **Handle of the driver** : if the connection succeeds.
ii. **NULL :** if connection failed.

Remarks

This function must be called before the other API is used. Please disconnect the capture driver when program is finished.

WIS_TerminateDriver

Synopsis

void WINAPI WIS_TerminateDriver(HANDLE hInit)

Parameter

hInit the handle returned by **WIS_InitDriver()**

Description

The **WIS_TerminateDriver()** disconnects the capture driver of the fingerprint device.

Return Value

None.

WIS_TestDevice

Synopsis

int WINAPI WIS_TestDevice(HANDLE hInit)

Description

Test if the fingerprint device is OK.

Parameter

hInit The handle returned by **WIS_InitDriver()**

Return Value

i. **OK** The fingerprint device is OK.
ii. **FAIL** There is problem with your fingerprint system.

Remarks

This function diagnoses your fingerprint device. Before testing, please clean the capture area and **make sure that there is no finger on the reader**.

WIS_CheckNoFinger

Synopsis

int WINAPI WIS_CheckNoFinger(HANDLE hInit)

Description

To check if there is any fingerprint on the reader.

Parameter

hInit	The handle returned by **WIS_InitDriver()**

Return Value

i.	**OK**	There is no fingerprint on the reader.
ii.	**FAIL**	There is a fingerprint on the reader.
iii.	**OUT_OF_MEMORY**	Failed to allocate memory.

Remarks

This function is mainly used in the enrollment process. To get the stable and real features of a fingerprint during the enrollment, the user must remove his finger from the reader once a fingerprint has been snapped and put it down again on the reader after **WIS_EnrolI** has successfully been processed for this snapped fingerprint image. You can check if a fingerprint has actually been lifted off the reader by using this function.

WIS_InitCapture

Synopsis

int WINAPI WIS_InitCapture(HANDLE hInit)

Parameter

hInit	The handle returned by **WIS_InitDriver()**

Description

This function MUST BE called prior to **WIS_Capture()** to snap a fingerprint from the fingerprint device to the main memory by a fingerprint image quality control process. Call **WIS_EndCapture()** to free the resource when the capture process is completed.

Return Value

OK	Succeeded.
FAIL	Unable to intialize the capture

Remarks

This function is to allocate the requireded resource for th ecapture process and MUST BE called prior to **WIS_Capture()** to snap a fingerprint from the fingerprint device to the main memory by a fingerprint image quality control process. Call **WIS_EndCapture()** to free the resource when the capture process is completed.

WIS_EndCapture

Synopsis

int WINAPI WIS_EndCapture(HANDLE hInit)

Parameter

hInit	The handle returned by **WIS_InitDriver()**

Description

This function MUST BE called when the capture process is completed. The function is used together with **WIS_InitCapture() and WIS_Capture()**.

Return Value

OK	Succeeded.
FAIL	Unable to intialize the capture process.

Remarks

This function MUST BE called when the capture process is completed. The function is used together with **WIS_InitCapture() and WIS_Capture()**.

WIS_Capture

Synopsis

int WINAPI WIS_Capture(HANDLE hInit, int *rCount)

Parameter

hInit	The handle returned by **WIS_InitDriver()**
rCount	A value used internally by the function. The developer MUST initial this value to 0 before use.

Description

To snap a fingerprint from the fingerprint device to the main memory by a fingerprint image quality control process. The fingerprint quality control cycle needs several frames of images and will continuously return the status of the fingerprint after each frame of image captured.

Return Value

OK	a valid fingerprint has successfully been snapped.
FAIL_GET_VERSION	**The driver** is found invalid.

Remarks

This function snaps a fingerprint image from the fingerprint device to the main memory. You should use a while loop to run this function and stop if a valid fingerprint has successfully been grabbed.

WIS_Snap

Synopsis

int WINAPI WIS_Snap(HANDLE hInit)

Parameter

hInit	The handle returned by **WIS_InitDriver()**

Description

To snap a fingerprint from the fingerprint device to the main memory by fingerprint image quality control process. The fingerprint quality control cycle needs several frames of images to judge the quality of the fingerprint. This function will return status of the fingerprint after a cycle of quality judgment.

Return Value

OK	a valid fingerprint has successfully been snapped.
FAIL_GET_VERSION	**The driver** is found invalid.

Remarks

This function snaps a good-enough fingerprint image from the fingerprint device to the main memory. You should use a while loop to run this function and stop if a valid fingerprint has successfully been grabbed.

WIS_CreateTemplate

Synopsis

int WINAPI WIS_CreateTemplate(HANDLE hInit, unsigned char *rRawTemplate)

Parameter

hInit	The handle returned by **WIS_InitDriver()**
rRawTemplate	The template, which is the extracted minutia of the fingerprint from the image of main memory.

Description

This function converts the fingerprint image in main memory to a 160 bytes raw fingerprint template that can roughly represent the feature of a fingerprint.

Return Value

i. **OK** : input image has been processed successfully.
ii. **OUT_OF_MEMORY** : insufficient memory for processing.
iii. **FAIL_GET_VERSION** : the driver is invalid.

Remarks

This function converts the fingerprint image in main memory to a 160 bytes raw fingerprint template that can roughly represent the feature of a fingerprint.

i. You should first snap a fingerprint to the main memory.
ii. You should allocate 160 bytes memory for the raw template
iii. 130K run time memory is required for this function.

WIS_GetImage

Synopsis

int WINAPI WIS_GetImage(HANDLE hInit, unsigned char Mode, unsigned char Size,unsigned char *lpImage)

Parameter

hInit	The handle returned by **WIS_InitDriver()**
Mode	GRAY or BINARY image.
Size	**LARGE** or **SMALL**.
lpImage	A pointer to the bufffer to save the raw image.

Description

Load the fingerprint image from the main memory to the buffer.

Return Value

i. **OK** : Get a fingerprint image successfully.
ii. **OUT_OF_MEMORY** : Unable to allocate memory while processing.
iii. **FAIL_GET_VERSION** : Driver is found invalid.

Remarks

This function gets a raw fingerprint image buffer. One must allocate the memory needed for the image.

Item	Memory Needed	Memory Needed
OR100	256 x 256	128 x 128
OR200	256 x 256	128 x 128
OR100-R	256 x 256	128 x 128

Please note:

i. You should first snap a fingerprint to the main memory.
ii. You should allocate the memory needed.
iii. You should free the memory when **WIS_GetImage()** is no longer in use.

WIS_GetImageSize

Synopsis

int WINAPI WIS_GetImageSize(HANDLE hInit, unsigned char SizeFlag, int *Width, int *Height, unsigned long *Size)

Parameter

hInit	The handle returned by **WIS_InitDriver()**
SizeFlag	A **LARGE** or **SMALL** image.
Width	The width of the image depending on the SizeFlag.
Height	The height of the image depending on the SizeFlag.
Size	Equal to Width * Height., can be NULL.

Description

Return the dimension of the image of LARGE or SMALL size.

Return Value

i. **OK** : Get the dimension successfully.
ii. **otherwise** : failed.

Remarks

This function return the dimension of the image of **LARGE** or **SMALL** size. One may allocate the memory needed for the image using the dimension. The memory need is Width * Height.

Item	LARGE	SMALL
OR100	256 x 256	128 x 128
OR200	256 x 256	128 x 128
OR100-R	256 x 256	128 x 128

WIS_SetEnrollMode

Synopsis

int WINAPI WIS_SetEnrollMode(HANDLE hInit, unsigned char Mode)

Parameter

hInit	The handle returned by **WIS_InitDriver()**
Mode	The mode of the enrollment. 1: 160 bytes 3: 480 bytes 4: 320 bytes

Description

The enrollment will generate a final fingerprint code of 160/320/480 bytes depending of the setting mode.

Return Value

OK: always succeed.

Remarks

These three modes will all give the high performance of matching. However, larger template size will keep more information of the fingerprint and thus give a higher accuracy but lower speed. The user may use the different mode depending of the applications and capture device.

For smaller area of capture device or 1-1 verification or 1-Little of identification, the 480-byte mode is recommended. For identification of a lot of persons that speed is the main concern, the 160-byte or 320-byte mode is recommended.

WIS_Enroll

Synopsis

int WINAPI WIS_Enroll(HANDLE hInit, unsigned char *rEnrlTemplate)

Parameter

hInit	The handle returned by **WIS_InitDriver()**
rEnrlTemplate	The final fingerprint code to represent the feature of a fingerprint if the enrollment is successful.

Description

Generate a final fingerprint code of 160/320/480 bytes.

Return Value

i. **QUALITY_A, QUALITY B, QUALITY C, QUALITY D**: The quality of enrolled fingerprint.
ii. **QUALITY_NOT_YET** : Enrollment is not completed yet.
ii. **Others < 0** : Image quality is not good enough.

Remarks

This function generates the final fingerprint code r***EnrlTemplate*** from several input ***RawTemplate*** by collecting their common features. The purpose of enrollment is to get enough stable characteristics to represent the corresponding fingerprint.

you should call **WIS_ReleaseEnroll()** to release the system resource. Basically, the kernel process of enrollment works in a continuous loop as following:

1. **Use WIS_Snap()** or **WIS_Capture** to get a good-enough fingerprint.
2. call **WIS_Enroll()**.
3. If the return value is not one of the qualities defined, repeat step 1 and step 2 until the ***quality*** of the fingerprint is derived.
4. Trials for more than 5 times and still cannot get the ***quality*** of the finger, that means the finger to enroll may not be good enough. You should change to another finger and restart the enrollment.
5. If you want to improve the enrolled quality, you can continue executing step 1 to step 3 to get a better final fingerprint code with better ***quality***.
6. If you have tried to ***enhance*** the enrolled quality more than 3 times but the ***quality*** still remains in a certain quality without any improvement, it seems that the enrolled ***quality*** has been stable. Any attempt to enhancement may be in vain. You should stop the enrollment with the stable enrolled ***quality***. If you are not satisfied with the current enrolled ***quality***, choose another finger and restart the enrollment.
7. call **WIS_ReleaseEnroll()** to free the resource.

WIS_ ReleaseEnroll

Synopsis

int WINAPI WIS_ReleaseEnroll (HANDLE hInit)

Parameter

hInit The handle returned by **WIS_InitDriver().**

Description

To release the all the internal resource created during the enrollment process.

Return Value

i. **>0** Resource is released successfully.

Remarks

This function releases all the internal resource created during the enrollment process. Call this function only if **WIS_Enroll()** is no longer in use.

WIS_VerifyTemplate

Synopsis

int WINAPI WIS_VerifyTemplate(HANDLE hInit, unsigned char *RawTemplate,
unsigned char *EnrlTemplate, int security, int *rScore)

Parameter

hInit	The handle returned by **WIS_InitDriver()**
RawTemplate	The fingerprint code generated through **WIS_CreateTemplate().**
EnrlTemplate	The final fingerprint template generated through **WIS_Enroll()**.
security	A parameter to set the threshold that determines where the verification can be passed.
rScore	The similarity of two fingerprints to be compared,ranged from 0 ~100. A higher score means a higher similarity.

SECURITY_A	Verification passes as long as the minutiae matching score is over the threshold. The FAR of security A is 1/100,000.
SECURITY_B	The FAR of security A is 1/10,000.
SECURITY_C	The FAR of security A is 3/10,000.
SECURITY_D	The FAR of security A is 1/1,000.
SECURITY_E	The FAR of security A is 1/100.

Description

To verify two fingerprint templates, while one is generated through **WIS_CreateTemplate()** and the other through the **WIS_Enroll()**.

Return Value

i. **OK** : The verification of fingerprint image with final fingerprint code meets the requirement of ***security*.**
ii. **FAIL** : The fingerprint image is not identical with the final fingerprint code on the required security.
iii. **OUT_OF_MEMORY :** Insufficient memory for image processing.
iv. **INVALID_TEMPLATE :** the input ***EnrlTemplate*** is illegal.
v. **INVALID_SECURITY :** improper security level setting.

Remarks

This function verifies two fingerprint templates, while one is generated through **WIS_CreateTemplate()** and the other through the **WIS_Enroll()**.

The argument ***security*** sets the threshold that determines whether this verification can be passed.

WIS_VerifyTemplateAllAngle

Synopsis

int WINAPI WIS_VerifyTemplateAllAngle(HANDLE hInit, unsigned char *RawTemplate, unsigned char *EnrlTemplate, int security, int *rScore)

Parameter

hInit	The handle returned by **WIS_InitDriver()**
RawTemplate	The fingerprint code generated through **WIS_CreateTemplate().**
EnrlTemplate	The final fingerprint template generated through **WIS_Enroll()**.
security	A parameter to set the threshold that determines where the verification can be passed. See **WIS_VerifyTemplate()** for details.
rScore	The similarity of two fingerprints to be compared,ranged from 0 ~100. A higher score means a higher similarity.

Description

To verify two fingerprint templates, while one is generated through **WIS_CreateTemplate()** and the other through the **WIS_Enroll()**.

Return Value

i. **OK** : The verification of fingerprint image with final fingerprint code meets the requirement of ***security*.**
ii. **FAIL** : The fingerprint image is not identical with the final fingerprint code on the required security.
iii. **OUT_OF_MEMORY :** Insufficient memory for image processing.
iv. **INVALID_TEMPLATE :** the input ***EnrlTemplate*** is illegal.
v. **INVALID_SECURITY :** improper security level setting.

Remarks

This function verifies two fingerprint templates, while one is generated through **WIS_CreateTemplate()** and the other through the **WIS_Enroll()**.

The argument ***security*** sets the threshold that determines whether this verification can be passed.

*This function will have little difference with **WIS_VerifyTemplate().** It will match in a way of **360 degrees**, i.e. even the upside-down finger can be verified. However, the matching speed will be a little slower than **WIS_VerifyTemplate().***

WIS_StartIdentify

Synopsis

int WINAPI WIS_StartIdentify(unsigned char Mode, unsigned char Threshold, unsigned char *RawTemplate)

Parameter

Mode	IDENTIFY_MODE_0 ~ IDENTIFY_MODE_9, while Mode0 has the fastest speed but higher FRR.
Threshold	The score to identify successfully, 0~100, 65 as default.
RawTemplate	The fingerprint code generated through **WIS_CreateTemplate().**

Description

This function is to initial some parameters and to allocate the resource for identification.

Return Value

i. **OK** : Succeeded.
ii. **FAIL** : Failed
iii. **OUT_OF_MEMORY :** Insufficient memory.
iv. **INVALID_TEMPLATE :** the input ***RawTemplate*** is illegal.

Remarks

This function is to initial some parameters and to allocate the resource for identification. One MUST call this function before using **WIS_Identify()** for 1:N matching and call **WIS_ReleaseIdentify**() while no longer in use.

	IDENTIFY_MODE_0	**IDENTIFY_MODE_9**
Speed	3000 templates/second	2000 templates /second
FRR	1/50	1/100

See **WIS_Identify()** for details.

WIS_Identify

Synopsis

int WINAPI WIS_Identify(unsigned char *EnrlTemplate, int TemplateIndex , int *rScore)

Parameter

EnrlTemplate	The final fingerprint template generated through **WIS_Enroll()**.
TemplateIndex	The unique index created by the programmer. This index will uniquely represent each matching fingerprint template.
rScore	The similarity of two fingerprints to be compared, ranged from 0 ~100. A higher score means a higher similarity.

Description

The matching speed is very important for 1:N identification. This function is used to speed up the matching process.

Return Value

i. **OK** : The index is verified and no more subsequent matching needed.
ii. **FAIL** : the process is not yet and keep doing the matching.
iii. **OUT_OF_MEMORY :** Insufficient memory for processing.
iv. **INVALID_TEMPLATE :** the input ***EnrlTemplate*** is illegal.

Remarks

The identification functions will speed up the matching process. These process will somehow influence the FRR but not FAR. For faster speed, the FRR will be higher.

FAR = 1/100,000

	IDENTIFY_MODE_0	IDENTIFY_MODE_9
Speed	3000 templates/second	2000 templates /second
Threshold =65	FRR = 1/50	FRR = 1/100
Threshold =75	FRR = 1/30	FRR = 1/70
Threshold =85	FRR = 1/20	FRR = 1/50

The Speed and FRR of **IDENTIFY_MODE_1 ~ IDENTIFY_MODE_8** is just between Mode 0 & Mode 9.

WIS_IdentifyResult

Synopsis

int WINAPI WIS_IdentifyResult(int *CandidateIndex, int *rMaxScore)

Parameter

CandidateIndex	The index of the candidate that has the highest score.
rMaxScore	The returned highest score ranged from 0 ~100. A higher score means a higher similarity.

Description

This function is to get the final result (Candidate's Index and Score) of **WIS_Identify()**.

Return Value

i. **OK** : The score is higher than the threshold set in **WIS_StartIdentify()**.
ii. **FAIL** : The score is lower than the threshold set in **WIS_StartIdentify()**.

Remarks

This function is to get the matching result and thus return the possible candidate that has the highest score. If the returned score is higher than the threshold set in **WIS_StartIdentify()**, a qualified candidate will be found.

WIS_ReleaseIdentify

Synopsis

int WINAPI WIS_ReleaseIdentify(void)

Parameter

No

Description

To release the all the internal resource created during the enrollment process.

Return Value

i. **OK** : Resource is released successfully.

Remarks

This function releases all the internal resource created during the identification process. Call this function only if **WIS_Identify()** is no longer in use.

WIS_SaveImage

Synopsis

int WINAPI WIS_SaveImage(HANDLE hInit, unsigned char Mode, unsigned char Size, unsigned short FileType, char* Filename)

Parameter

hInit	The handle returned by **WIS_InitDriver()**
Mode	GRAY or BINARY image.
Size	**LARGE** or **SMALL**.
FileType	The image can be saved as a bitmap (**BMP**) file or a raw (**RAW**) file.
Filename	The filename to be saved as.

Description

Save the fingerprint image of required mode and size to a BMP or RAW file.

Return Value

i. **OK** : the image is saved successfully.
ii. **FAIL_OPEN_FILE** : failed to open the file.
iii. **OUT_OF_MEMORY** : failed to allocate memory.

Remarks

This function saves the image as a BMP or RAW file with the specified filename. The size and mode of the image must be determined.

Item	LARGE	SMALL
OR100	256 x 256	128 x 128
OR200	256 x 256	128 x 128
OR100-R	256 x 256	128 x 128

WIS_DisplayImage

Synopsis

int WINAPI WIS_DisplayImage(HANDLE hInit , HDC hDC, unsigned char Mode,
unsigned char Size, int nStartX, int nStartY ,
int nDestWidth, int nDestHeight)

Parameter

hInit	The handle returned by **WIS_InitDriver()**
hDC	Identifies the device context.
Mode	GRAY or BINARY image.
Size	**LARGE** or **SMALL**.
nStartX, nStartY	The start position of the image to be displayed
nDestWidth, nDestHeight	The size of the image to be displayed

Description

Display the fingerprint image of required mode and size on a device context with the specified position and size.

Return Value

i. **OK** If succeeds
ii. **FAIL** Otherwise.

Remarks

The function displays the fingerprint image of required mode and size on a device context with the specified position and size.

Item	LARGE	SMALL
OR100	256 x 256	128 x 128
OR200	256 x 256	128 x 128
OR100-R	256 x 256	128 x 128

WIS_SetParameter

Synopsis

BOOL WINAPI WIS_SetParameter(HANDLE hInit , unsigned char bBrightness, unsigned char bContrast, short sGamma)

Parameter

hInit	The handle returned by **WIS_InitDriver()**
bBrightness	To set the brightness of the output image, ranged from 0 ~ 255, default: 32
bContrast	To set the contrast of the output image, ranged from 0 ~ 31, default: 0
sGamma	To set the brightness of the output image, ranged from 0 ~ 10000, default: 1000.

Description

The programmer can tune the quality of the image depends on the environment or the status of the fingerprint. ***This function is valid only for OR100/207 series.***

Return Value

i. **TRUE** If succeeds
ii. **FALSE** Otherwise.

Remarks

The function let the programmer to tune the quality of the image depends on the environment or the status of the fingerprint.