www.x-telcom.com

Function interface description of Fingerprint Module X-Telcom AGF200

Table of contents

1. Initialize the collector

Function prototype : int __stdcall LIVESCAN_Init ( ) ; Parameters : None.

Return value : The call is successful and returns 1

2. Release Collector

Function prototype ; int _ stdcall LIVESCAN_Close ( ) ; Parameters : None.

Return value ; if the call is successful, return 1

3. Get the number of collector channels

Function prototype : int __stdcall LIVESCAN_GetChannelCount ( ) ; Parameters : None.

Return value : If the call is successful, the number of channels returned (>0)

4. Setting Brightness

Function prototype : int __stdcall LIVESCAN_SetBright (int nChannel , int nBright); Parameters : int nChannel Channel number. Input parameter.

int nBright Brightness , ranging from 0 to 255, where 0 means all black. 255 means all white. Input parameter. Return value : Call successful , return 1

5. Setting the contrast

Function prototype : int __stdcall LIVESCAN_SetContrast (int nChannel , int nContrast); Parameters : int nChannel Channel number. Input parameter.

int nContrast Contrast ratio , range is 0~255 . Input parameters. Return value : Call successful , return 1

6. Get brightness

Function prototype : int __stdcall LIVESCAN_GetBright (int nChannel , int * nBright); Parameters : int nChannel Channel number. Input parameter.

int * pnBright Integer pointer storing the current brightness. Output parameter. Return value : Call successful , return 1

7. Get Contrast

Function prototype : int __stdcall LIVESCAN_GetContrast (int nChannel , int * nContrast); Parameters : int nChannel Channel number. Input parameter.

int * pnContrast Integer pointer storing the current contrast. Output parameter. Return value : Call successful, return 1

8. Get the maximum width and height of the image captured by the acquisition device

Function prototype : int __stdcall LIVESCAN_GetMaxImageSize (int nChannel , int * pnWidth , int *p nHeight);

Parameters : int nChannel Channel number. Input parameter.

int * pnWidth Integer pointer storing the image width. Output parameter

int * pnHeight Integer pointer storing the image height. Output parameter Return value : Call successful, return 1

9. Get the acquisition position, width and height of the current image

Function prototype : int __stdcall LIVESCAN_GetCaptWindow (int nChannel , int * pnOriginX , int * pnOriginY , int * pnWidth , int * pnHeight);

Parameters : int nChannel Channel number. Input parameter.

int * pnOriginX The X value of the acquisition origin coordinate of the image acquisition window . Output parameters int * pnOriginY The Y value of the acquisition origin coordinate of the image acquisition window . Output parameters int * pnWidth Integer pointer to store the width of the acquired image. Output parameter.

int * pnHeight Integer pointer storing the height of the acquired image. Output parameter Return value : Call successful , return 1

10. Set the acquisition position, width and height of the current image

Function prototype : int __stdcall LIVESCAN_SetCaptWindow (int nChannel , int nOriginX , int nOriginY , int nWidth , int nHeight);

Parameter: int nChannel Channel number. Input parameter.

int nOriginX The X value of the acquisition origin coordinate of the image acquisition window . Input parameter.

int nOriginY The Y coordinate of the acquisition origin of the image acquisition window Value. Input parameter. int nWidth An integer pointer storing the width of the captured image. Input parameters

int nHeight Integer pointer to store the height of the acquired image. Input parameter. Return value : Call successful, return 1

11. Call the property setting dialog box of the acquisition device

Function prototype : int __stdcall LIVESCAN_Setup ( ) ; Parameters : None.

Return value : Call successful, return 1

12. Prepare to capture an image frame

Function prototype : int __stdcall LIVESCAN_BeginCapture (int nChannel);

Parameters : int nChannel Channel number. Input parameter.

Return value : Call successful , return 1

13. Capture an image frame

Function prototype : int __stdcall LIVESCAN_GetFPRawData (int nChannel , BYTE * pRawData); Parameters : int nChannel Channel number. Input parameters

BYTE * pRawData Points to the memory block where the collected data is stored. Output parameter. Return value : Call successful, return 1

14. End the acquisition of one frame of image

Function prototype : int __stdcall LIVESCAN_EndCapture (int nChannel);

Parameters : int nChannel Channel number. Input parameter.

Return value : Call successful , return 1

15. Determine whether the acquisition device supports acquisition window settings

Function prototype : int __stdcall LIVESCAN_IsSupportCaptWindow (int nChannel);

Parameters : int nChannel Channel number. Input parameter.

Return value : Call successful , return 1

16. Determine whether the acquisition device supports the settings dialog box

Function prototype : int __stdcall
LIVESCAN_IsSupportSetup ( ) ; Parameters : None

Return value : Call successful , return 1

17. Get the interface version

Function prototype : int __stdcall
LIVESCAN_GetVersion ( ) ; Parameters : None

Return value : If the call is successful , the version number is returned (!=0)

18. Get interface description

Function prototype : int __stdcall LIVESCAN_GetDesc (char pszDesc [1024]);

Parameter : char pszDesc [1024] A memory block used to store interface descriptions. Output parameters.

Return value : Call successful, return 1

19. Get interface error information

Function prototype : int __stdcall LIVESCAN_GetErrorInfo (int nErrorNo , char pszErrorInfo [256]);

Parameter : char pszErrorInfo [256] Memory block used to store error information. Output parameter. ErrorNo Error code (<0) . Input parameters.

Return value : If it is a valid error number, it returns 1, at the same time pszErrInfo The information in the message is incorrect.

Description: pszErrorInfo Information encoding characters are used Characters specified in GB 2312 , characters not specified in GB 2312 Characters shall be those specified in GB 18030. Information shall be in the form of numerical values. Zero- terminated string.

20. codingWSQ

Function prototype:int _____ stdcall EncodeWSQ(unsigned char* wsqImageOut, int *wsqImageOutSi ze, float wsqBitRate, unsigned char* rawImage, int width, int height, int pixelDepth, int ppi);

Parameters : unsigned char * wsqImageOut wxya Image binary data stream. Output parameter.

int * wsqImageOutSize : wxya Image size. Output parameter. float wsqBitRate wxya Image compression ratio. Input parameter.

unsigned char * rawImage raw Image raw data. Input parameters. int width : Image width. Input parameter.

int height : Image height. Input parameter. int ppi : Image resolution. Input parameter.

Return value : Call successful, return 1

twenty one. decodingWSQ

Function prototype:int __stdcall DecodeWSQ(unsigned char*rawImageOut, int*width, int*height, unsigned char*wsqImage, int wsqImageLength);

Parameters : unsigned char * rawImageOut raw Image binary data stream. Input parameter.

int * width Image width. Output parameter. Int * height : Image height. Output parameter.

unsigned char * wsqImage wxya Image binary data stream. Input parameter. int wsqImageLength wxya Image length. Input parameter.

Return value : Call successful, return 1

twenty two. Get image quality

Function prototype : int __stdcall int GetImageQuality (unsigned char * image , int w , int h); Parameters : byte [] image image RAW Data. Input parameters.

int w Fingerprint image width. Input parameter. int h Fingerprint image height. Input parameter.

Return value : If the call is successful, the image quality is returned, and the quality range is [0, 1 00]

twenty three. Get touch status

Function prototype : int __stdcall int GetTouchStatus (unsigned char * status);

Parameters : unsigned char * status : Touch status. Output parameter.

Return value : Call successful, return 1

twenty four. Set the light status

Function prototype : int __stdcall int SetLedStatus (unsigned char led);

Parameters : unsigned char led . Light status. 0 turns off the light, 6 turns on the light. Input parameters.

Return value : Call successful, return 1

25. Image acquisition interface error code

Image acquisition interface error codes are shown in the table 1.1 .

surface 1.1 Error code description

Error Code	illustrate
-1	Parameter error. The parameters of the given function are wrong.
-2	Memory allocation failed. Not enough memory was allocated.
-3	Function not implemented. The function called is not implemented
-4	The device does not exist. During initialization, it was detected that the device does not exist.
-5	Device not initialized
-6	Illegal error number
-7	No authorization

26. Initialization algorithm

Function prototype : int __stdcall ZAZ_FpStd Lib_OpenDevice ( ) ;

Parameters: None.

Return value : If the call is successful, the device handle is returned; otherwise, 0

27. Release Algorithm

Function prototype : void __stdcall ZAZ_FpStd Lib_CloseDevice (int device);

Parameters: int device Device handle. Input parameter.

Return value : Call successful, return 1 , otherwise return 0

28. Compare fingerprint features

Function prototype: int __stdcall ZAZ_FpStd Lib_CompareTemplates(int device, unsigned char *sTemplate, unsigned char *fTemplate);

Parameters: int device Device handle. Input parameter.

unsigned char * sTemplate Fingerprint feature value.

Input parameter. unsigned char * fTemplate

Fingerprint feature value. Input parameter.

Return value : If the call is successful, the similarity score of the two fingerprint features is returned, and the score range is [0, 100]

29. Generate fingerprint features

letter number Original type : int __stdcall ZAZ_FpStd Lib_CreateISOTemplate (int device , unsigned char

*image, unsigned char *itemplate);

Parameters: int device Device handle. Input parameter.

unsigned char * image Fingerprint image data can only be entered Image size 256*360 bytes. Input parameters.

unsigned char * itemplate Fingerprint feature value, assigned before input 1024 The size in bytes. Output parameter. Return value : If the call is successful, the fingerprint feature size is returned, otherwise it is returned 0