SynoAPIEX_DLL Function Manual

v3.035, 2011.11

hangzhou Synochip Technologies Co.,Ltd

http://www.synochip.com

Addr: Rm 103, Building 17, No.176 Tianmushan Road, Hangzhou 310012, P.R.China

Statement

Following documents contain hangzhou Synochip Technologies Co.,Ltd (for short SYNO) private information. This information is accurate, reliable, without permission of the management of the Company, the third party shall not use or disclose freely; of course, without the authorization of any special conditions, limitations or inform the case of a copy of this information and unauthorized Changes are violations.

At any time, without telling any of the parties in the case, SYNO shall have the right about products and services of the company to make changes, add, delete, improvements and other changes. In the use of our products, SYNO does not carry any responsibility or obligation; and third parties shall not use any patents or other intellectual property infringement.

All products sold are subject to the recognition of the book in order of sales terms and conditions. The Company uses the test, tools, quality control and other technical means to support the products meet the required performance specifications related to a certain degree of assurance. In addition to the written requirements of government clearly is not necessary to perform all the parameters of each product tested.

In addition to SYNO's logo design, all other trademarks or registered trademarks are the property of their respective owners.

Version history

Ver.	Date	Modify		
		capter	page	content
3.031	2011-04-14			Initial version
3.035	2011-11-16			40~42,Note1~4

Index

1 API Function Description

1.1 Define

```
// error code
#define PS_OK                   0x00   // success
#define PS_COMM_ERR             0x01   // package error
#define PS_NO_FINGER            0x02   // There is no finger on the sensor
#define PS_GET_IMG_ERR          0x03   // fail to get image
#define PS_FP_TOO_DRY           0x04   // the finger too dry
#define PS_FP_TOO_WET           0x05   // the finger too wet
#define PS_FP_DISORDER          0x06   // Too disorderly fingerprint"
#define PS_LITTLE_FEATURE       0x07   // too little feature
#define PS_NOT_MATCH            0x08   // No matching fingerprint
#define PS_NOT_SEARCHED         0x09   // No fingerprint searched
#define PS_MERGE_ERR            0x0a   // Combine character error
#define PS_ADDRESS_OVER         0x0b   // Address number is out of fingerprint range
#define PS_READ_ERR             0x0c   // Read templet from fingerprint library error
#define PS_UP_TEMP_ERR          0x0d   // Upload character failed
#define PS_RECV_ERR             0x0e   // Recv package error
#define PS_UP_IMG_ERR           0x0f   // Upload image failed"
#define PS_DEL_TEMP_ERR         0x10   // Del Template error
#define PS_CLEAR_TEMP_ERR       0x11   // Clear Template error
#define PS_SLEEP_ERR            0x12   // Can't go into Sleep mode
#define PS_INVALID_PASSWORD     0x13   // Password is not correct
#define PS_RESET_ERR            0x14   // reset system error
#define PS_INVALID_IMAGE        0x15   // Invalid fingerprint image
#define PS_HANGOVER_UNREMOVE      0X17   // Must move finger

// buffer
#define CHAR_BUFFER_A           0x01
#define CHAR_BUFFER_B           0x02
#define MODEL_BUFFER            0x03

// Device type
#define DEVICE_USB      0
#define DEVICE_COM      1
#define DEVICE_UDisk    2

// Baud
```

```
#define BAUD_RATE_9600          0x01
#define BAUD_RATE_19200         0x02
#define BAUD_RATE_38400         0x04
#define BAUD_RATE_57600         0x06    //default
#define BAUD_RATE_115200        0x0C

#define CHAR_LEN_AES1711        1024  // 512->1024 [2009.11.12] AES1711 1024 size of the template
#define CHAR_LEN_NORMAL         512   // 512  Generic version of the template using the 512 size
```

1.2 Parameter Introduction

hHandle: device handle, user of Multi-device operation,get bf function PSOpenDeviceEx.
nAddr: device addr. Default is 0xFFFFFFFF
pPassword: device communicate key
iBufferID: used to store temporary fingerprint.It's value is 1,2
iStartPage: the start page of flash about fingerprint library
iPageNum: page number about fingerprint library
iMbAddress: get the page index which find the right finger

1.3 Function

1. **OpenDevice**
   int WINAPI PSOpenDeviceEx(HANDLE* pHandle, int nDeviceType,int iCom=1,int iBaud=1,int nPackageSize=2,int iDevNum=0);
   Parameter:
   [IN] nDeviceType, iCom,iBaud, nPackageSize, iDevNum-dev number of base 0
   [OUT] pHandle-get device handle
   Return: returns PS_OK if success,others see error code

2. **CloseDevice**
   int WINAPI PSCloseDeviceEx(HANDLE hHandle);
   Parameter:
   [IN] hHandle
   [OUT] null
   Return: returns PS_OK if success,others see error code

3. **Detect finger and Get Image**
   int WINAPI PSGetImage(HANDLE hHandle,int nAddr);
   int WINAPI PSGetImage_Enroll(HANDLE hHandle,int nAddr);
   Parameter:

[IN] hHandle,nAddr
[OUT] null
Return: returns PS_OK if success,others see error code

4. Generate Character file

int WINAPI PSGenChar(HANDLE hHandle,int nAddr,int iBufferID);
Parameter:
[IN] hHandle, nAddr, iBufferID
[OUT] null
Return: returns PS_OK if success,others see error code

5. Match two character file on chip

int WINAPI PSMatch(HANDLE hHandle,int nAddr,int* iScore);
Parameter:
[IN] hHandle, nAddr,
[OUT] iScore
Return: returns PS_OK if success,others see error code

6. Search a part or all of fingerprint libray

int WINAPI PSSearch(HANDLE hHandle,int nAddr,int iBufferID, int iStartPage, int iPageNum, int *iMbAddress,int *iscore=NULL);
Parameter:
[IN] hHandle, nAddr, iBufferID, iStartPage, iPageNum
[OUT] iMbAddress, iscore
Return: returns PS_OK if success,others see error code

7. Combine BufferA's character file with BufferB's character file and generate the templet

int WINAPI PSRegModule(HANDLE hHandle,int nAddr);
Parameter:
[IN] hHandle, nAddr
[OUT] null
Return: returns PS_OK if success,others see error code

8. Store BufferA or BufferB's character file to flash fingerprint library

int WINAPI PSStoreChar(HANDLE hHandle,int nAddr,int iBufferID, int iPageID);
Parameter:
[IN] hHandle, nAddr, iBufferID, iPageID
[OUT] null
Return: returns PS_OK if success,others see error code

9. Transfer a templet to BufferA or BufferB from flash fingerprint library

int WINAPI PSLoadChar(HANDLE hHandle,int nAddr,int iBufferID,int iPageID);
Parameter:
[IN] hHandle, nAddr, iBufferID, iPageID

[OUT] null

Return: returns PS_OK if success,others see error code

10. Transfer character file from BufferA or BufferB to PC

int WINAPI PSUpChar(HANDLE hHandle,int nAddr,int iBufferID, unsigned char* pTemplet, int* iTempletLength);

Parameter:

[IN] hHandle, nAddr, iBufferID

[OUT] pTemplet, iTempletLength

Return: returns PS_OK if success,others see error code

11. Download a character form form pc to BufferA or BufferB

int WINAPI PSDownChar(HANDLE hHandle,int nAddr,int iBufferID, unsigned char* pTemplet, int iTempletLength);

Parameter:

[IN] hHandle, nAddr, iBufferID, pTemplet, iTempletLength

[OUT] null

Return: returns PS_OK if success,others see error code

12. Upload Original Image

int WINAPI PSUpImage(HANDLE hHandle,int nAddr,unsigned char* pImageData, int* iImageLength);

Parameter:

[IN] hHandle, nAddr

[OUT] pImageData, iImageLength

Return: returns PS_OK if success,others see error code

13. Download original Image

int WINAPI PSDownImage(HANDLE hHandle,int nAddr,unsigned char *pImageData, int c);

Parameter:

[IN] hHandle, nAddr, pImageData, pImageData

[OUT] null

Return: returns PS_OK if success,others see error code

14. Genarate a BMP file from image data buffer

int WINAPI PSImgData2BMP(unsigned char* pImgData,const char* pImageFile);

Parameter:

[IN] pImgData, pImageFile-file name

[OUT] null

Return: returns PS_OK if success,others see error code

15. Get image data form bmp file

int WINAPI PSGetImgDataFromBMP(HANDLE hHandle,const char *pImageFile,unsigned char *pImageData,int *pnImageLen);

Parameter:

[IN] hHandle, pImageFile,
[OUT] pImageData, pnImageLen

Return: returns PS_OK if success,others see error code

16. Delete specify range of character file from flash fingerprint libaray

int WINAPI PSDelChar(HANDLE hHandle,int nAddr,int iStartPageID,int nDelPageNum);

Parameter:

[IN] hHandle, nAddr, iStartPageID, nDelPageNum
[OUT] null

Return: returns PS_OK if success,others see error code

17. Clear flash fingerprint libaray

int WINAPI PSEmpty(HANDLE hHandle,int nAddr);

Parameter:

[IN] hHandle, nAddr
[OUT] null

Return: returns PS_OK if success,others see error code

18. Read Parameter table

int WINAPI PSReadParTable(HANDLE hHandle,int nAddr,unsigned char* pParTable);

Parameter:

[IN] hHandle, nAddr
[OUT] pParTable

Return: returns PS_OK if success,others see error code

19. Read flash information page

int WINAPI PSReadInfPage(HANDLE hHandle,int nAddr, unsigned char* pInf);

Parameter:

[IN] hHandle, nAddr
[OUT] pInf-512 bytes data

Return: returns PS_OK if success,others see error code

20. Search a part or all of fingerprint library by high speed

int WINAPI PSHighSpeedSearch(HANDLE hHandle,int nAddr,int iBufferID, int iStartPage, int iPageNum, int *iMbAddress,int *iscore=NULL);

Parameter:

[IN] hHandle, nAddr, iBufferID, iStartPage, iPageNum
[OUT] iMbAddress, iscore

Return: returns PS_OK if success,others see error code

21. Get the fingerprint count in library

int WINAPI PSTemplateNum(HANDLE hHandle,int nAddr,int *iMbNum);

Parameter:

[IN] hHandle, nAddr

[OUT] iMbNum

Return: returns PS_OK if success,others see error code

22. Generate binary image

int WINAPI PSGenBinImage(HANDLE hHandle,int nAddr, int nImgType);

Parameter:

[IN] hHandle, nAddr, nImgType

[OUT] null

Return: returns PS_OK if success,others see error code

23. Set device communicate key

int WINAPI PSSetPwd(HANDLE hHandle,int nAddr,unsigned char* pPassword);

Parameter:

[IN] hHandle, nAddr

[OUT] null

Return: returns PS_OK if success,others see error code

24. Verify device communicate key

int WINAPI PSVfyPwd(HANDLE hHandle,int nAddr,unsigned char* pPassword);

Parameter:

[IN] hHandle, nAddr, pPassword

[OUT] null

Return: returns PS_OK if success,others see error code

25. Read Notepad

int WINAPI PSReadInfo(HANDLE hHandle,int nAddr,int nPage,unsigned char* UserContent);

Parameter:

[IN] hHandle, nAddr, nPage

[OUT] UserContent-512 bytes data, 32 bytes for each page

Return: returns PS_OK if success,others see error code

26. Write Notepad

int WINAPI PSWriteInfo(HANDLE hHandle,int nAddr,int nPage,unsigned char* UserContent);

Parameter:

[IN] hHandle, nAddr, nPage, UserContent

[OUT] null

Return: returns PS_OK if success,others see error code

27. Auto enroll

int WINAPI PSEnroll(HANDLE hHandle,int nAddr,int* nID);

Parameter:

[IN] hHandle, nAddr

[OUT] nID

Return: returns PS_OK if success,others see error code

28. Write register

int WINAPI PSWriteReg(HANDLE hHandle,int nAddr,int iRegAddr,int iRegValue);

Parameter:

[IN] hHandle, nAddr, iRegAddr, iRegValue

[OUT] null

Return: returns PS_OK if success,others see error code

29. Set baudrate

int WINAPI PSSetBaud(HANDLE hHandle,int nAddr,int nBaudNum);

Parameter:

[IN] hHandle, nAddr, nBaudNum

[OUT] null

Return: returns PS_OK if success,others see error code

30. Set security level

int WINAPI PSSetSecurLevel(HANDLE hHandle,int nAddr,int nLevel);

Parameter:

[IN] hHandle, nAddr, nLevel

[OUT] null

Return: returns PS_OK if success,others see error code

31. Set parket size

int WINAPI PSSetPacketSize(HANDLE hHandle,int nAddr,int nSize);

Parameter:

[IN] hHandle, nAddr, nSize

[OUT] null

Return: returns PS_OK if success,others see error code

32. Upload character file to PC

int WINAPI PSUpChar2File(HANDLE hHandle,int nAddr,int iBufferID, const char* pFileName);

Parameter:

[IN] hHandle, nAddr, iBufferID, pFileName

[OUT] null

Return: returns PS_OK if success,others see error code

33. Download a character file to BufferA or BufferB from PC

int WINAPI PSDownCharFromFile(HANDLE hHandle,int nAddr,int iBufferID, const char* pFileName);

Parameter:

[IN] hHandle, nAddr, iBufferID, pFileName

[OUT] null

Return: returns PS_OK if success,others see error code

34. Get random data generate by chip

int WINAPI PSGetRandomData(HANDLE hHandle,int nAddr,unsigned char* pRandom);

Parameter:

[IN] hHandle, nAddr

[OUT] pRandom-4 bytes

Return: returns PS_OK if success,others see error code

35. Set chip address

int WINAPI PSSetChipAddr(HANDLE hHandle,int nAddr,unsigned char* pChipAddr);

Parameter:

[IN] hHandle, nAddr

[OUT] pChipAddr-4 bytes

Return: returns PS_OK if success,others see error code

36. Auto identify finger

int WINAPI PSIdentify(HANDLE hHandle,int nAddr,int *iMbAddress);

Parameter:

[IN] hHandle, nAddr

[OUT] iMbAddress

Return: returns PS_OK if success,others see error code

37. Set GPIO state

int WINAPI PSDoUserDefine(HANDLE hHandle,int nAddr,int GPIO,int STATE);

Parameter:

[IN] hHandle, nAddr, GPIO, STATE

[OUT] null

Return: returns PS_OK if success,others see error code

38. Read flash index table of fingerprint library

int WINAPI PSReadIndexTable(HANDLE hHandle,int nAddr,int nPage,unsigned char* UserContent);

Parameter:

[IN] hHandle, nAddr, nPage-0,1,2,3

[OUT] UserContent-256 bytes,correspond nPage is 0~255 256~511 512~767 768~1023

Return: returns PS_OK if success,others see error code

39. Reservation interface for custom data packets

int WINAPI PS_SB(HANDLE hHandle,int nAddr,unsigned char *pSendData,int iLen,unsigned char *pRecvData,int *pRecvLen,int flag=1);

Parameter:

[IN] hHandle, nAddr, pSendData, iLen, flag-0,not receive data 1-receive data

[OUT] pRecvData, pRecvLen

Return: returns PS_OK if success,others see error code

40. Get Error message

char* WINAPI PSErr2Str(int nErrCode);

Parameter:

[IN] nErrCode：Error Code

[OUT] null

Return: Error message

41. Get the size of features

int WINAPI PSGetCharLen(int *pnLen);

Parameter:

[IN] null

[OUT] pnLen：The size of features.

Return: returns PS_OK if success,others see error code

42. Set the size of features

int WINAPI PSSetCharLen(int nLen = CHAR_LEN_NORMAL);

Parameter:

[IN] nLen：The size of features.

[OUT] null

Return: returns PS_OK if success,others see error code

2 Other

2.1 Note

1. AES1711 template size is 1024, but the default template size is 512. So the operation AES1711 device before you use the function PSSetCharLen set the template size to 1024 size
   You can use the function PSGetCharLen to see the size of the current template.
2. Expanded support for 10 or more serial ports.
3. Baud rate parameter iBaud, its value represents a multiple of the baud rate of 9600.
   For example, iBaud = 6, then the baud rate to 9600 * 6 = 57600
4. Remove restrictions on BufferID.BufferID values are based on 1