

Heng Yu KB Statistics API

Edition 1.1

Contents

Keymap/Firmware Update API	3
int UpdateKeyMap(char * sExePath, char *sKeymapPath, char *sLogPath);	3
int UpdateFirmware(char * sExePath, char *sFirmwarePath, char *sLogPath);	4
int GetResultOfUpdate(char *sLogPath, char *sResultMsg, time_t * latestUpdateTime);	7
Keyboard Statistics API.....	8
Model Property Structure	8
BOOL IsUsbConnected();	8
int FlushKbEEPROM();	9
int GetKbModel(model_prop *pModelProp);	10
int GetPressedCount(int nKeyIndex, int *pCount);	11
int GetMaxPressedCount(int *pKeyPos, int *pCount)	12
int GetSwipeCount(int *pTotalCount, int *pSuccessCount);	13
int GetFirmware(char *sVer)	14

Keymap/Firmware Update API

```
int UpdateKeyMap(char * sExePath, char *sKeymapPath, char *sLogPath);
```

Description:

Update Key map of the keyboard

Input parameter:

sExePath – execute file name and its absolute path

sKeymapPath – keymap file name and its absolute path

sLogPath – log file name and its absolute path

Return:

0x00 => success

0x01 => The system is out of memory source

0x2F => Keyboard model is not found

0xE6 => Invalid input parameter

0x62 => Invalid exe file(format invalid)

0x63 => exe file cannot be found

0x64 => exe path is not found

```
int UpdateFirmware(char * sExePath, char *sFirmwarePath, char *sLogPath);
```

Description:

Update the firmware to the keyboard using firmware downloader software with a USB cable or PS2 cable according to exe file.

Input parameter:

sExePath – execute file name and its absolute path

sFirmwarePath– firmware file name and its absolute path

sLogPath – log file name and its absolute path

Return:

0x00 => success

0x01 => The system is out of memory source

0xE6 => Invalid input parameter

0x62 => Invalid exe file(format invalid)

0x63 => exe file cannot be found

0x64 => exe path is not found

Example, USB update

```
int ret=0,retry=0;
char content[2048]={0};
time_t time;
if(!UpdateFirmware("HY_USB_FD.exe","L2326V152.bin","FirmwareLog.txt"))
{
    //check the log file has been created or not
    do
    {
        ret = GetResultofUpdate("FirmwareLog.txt",content,&time);
        if(ret ==0)
        {
            //get the content of the log file
            str.Format("Content : %s Latest Mod Time: %s",content,ctime(&time));
            AfxMessageBox(str);
            break;
        }
        //delay 1s if not found
        Sleep(1000);
        retry++;
    }while(ret != 0 || retry < 120); // usb within 2 min or ps2 within 5 min
}else
    AfxMessageBox("Update Firmware through USB fail");
```

Example, PS2 update

```
int ret=0,retry=0;
char content[2048]={0};
time_t time;
if(!UpdateFirmware("HY_PS2_FD.exe","L2326V152.bin","FirmwareLog.txt"))
{
    //check the log file has been created or not
    do
    {
        ret = GetResultofUpdate("FirmwareLog.txt",content,&time);
        if(ret ==0)
        {
            //get the content of the log file
            str.Format("Content : %s Latest Mod Time: %s",content,ctime(&time));
            AfxMessageBox(str);
            break;
        }
        //delay 1s if not found
        Sleep(1000);
        retry++;
    }while(ret != 0 || retry < 420); // ps2 within 5 min, so 420/60 = 7 min timeout
}else
    AfxMessageBox("Update Firmware through PS2 fail");
```

```
int GetResultOfUpdate(char *sLogPath, char *sResultMsg, time_t * latestUpdateTime);
```

Description:

Get the result of update from a given log file

Input parameter:

sLogPath - the absolute path of the log file

sResultMsg– [out] The buffer supplied by the application layer where the message returned from the update is to be stored

latestUpdateTime– [out] The buffer supplied by the application layer where the latest update time is to be stored

Return:

0x00 => Success

0xE6 => Invalid input parameter

0x60 => No file can be read

0x61 => cannot open file

Example:

```
CString str;
time_t time;
char content[2048]={0};
if (!GetResultofUpdate("log.txt",content,&time)) {
    str.Format("Content : %s Latest Mod Time: %s ",content,ctime(&time));
    AfxMessageBox (str);
}
else
    AfxMessageBox("Get result of update fail");
```

Keyboard Statistics API

Model Property Structure

```
typedef struct model_prop {  
    int nPID;  
    int nTotalKeys; // number of keys in this model  
    BOOL bMsrSupport; // is MSR supported?  
    char sModelName[16]; // model name, e.g. S66A  
    byte pbKeyIndex[256];  
} model_prop;
```

Note:

Each key is assigned with a key index.

pbKeyIndex stores indices for an array of the keys, starting from the top to the right and the left to the right of the keyboard.

For examples,

pbKeyIndex[0] is key index of the leftmost key at the top row of the keyboard.

pbKeyIndex[1] is key index of the 2nd leftmost key at the top row of the keyboard.

pbKeyIndex[nTotalKeys-1] is key index of the rightmost key at the bottom row of the keyboard.

BOOL IsUsbConnected();

Description:

Check the Heng Yu USB keyboard (VID: 0x0F39) is connected.

Return:

0 – Assume the PS2 keyboard is connected

1 – USB keyboard is connected


```
int FlushKbEEPROM();
```

Description:

Flush the statistic to keyboard EEPROM. It should be called when the POS machine is going to be shut down

Return :

0x00 => flush success

0xE1 => function initial fail

0xE2 => send command fail

0xE7 => received command fail

0xF0 => Checksum error

0xF1 => Invalid command

0xFB => Length error

```
int GetKbModel(model_prop *pModelProp);
```

Description:

Get the model from MODEL_PROP array.

Input parameter:

pModelProp – [out] The buffer supplied by the application layer where the keyboard property is to be stored keyboard model structure.

Return:

0x00 => success

0xE1 => function initial fail

0xE2 => send command fail

0xE6 => Invalid input parameter

0xE7 => received command fail

0xF0 => Checksum error

0xF1 => Invalid command

0xFB => Length error

Example:

```
model_prop *pModel;  
CString str;  
int ret=0;  
pModel = (model_prop *)calloc(1,sizeof(model_prop));  
if (pModel) {  
    ret = GetKbModel(pModel);  
    if (ret == 0) {  
        str.Format("Product ID : %s",pModel->sModelName);  
        AfxMessageBox(str.GetBuffer());  
    }  
    free(pModel);  
}
```

```
int GetPressedCount(int nKeyIndex, int *pCount);
```

Description:

Get the no. of accumulated pressed down for a given key index

Input parameters:

nKeyIndex – index of the key

pCount – [out] The buffer supplied by the application layer where the count of the pressed down of this key is to be stored

Return:

0x00 => success

0xE1 => function initial fail (e.g. pCount is NULL pointer or usb initialization fail)

0xE2 => send command fail

0xE6 => invalid input parameters

0xE7 => received command fail

0xF0 => Checksum error

0xF1 => Invalid command

0xFB => Length error

Example:

```
// When the user wants to get all keys' pressed-down count,
model_prop modelProp;
int nCount;
if (GetKbModel(model_prop &modelProp) == 0) {
    for (int i=0; i<modelProp.nTotalKeys; i++)
        GetPressedCount(model_prop.pbKeyIndex[i], &nCount);
    }
}
```

```
int GetMaxPressedCount(int *pKeyPos, int *pCount)
```

Description:

Get the max. no. of accumulated pressed down and the corresponding key position counting from the top row (from the left to the right) to the bottom row

Input parameter:

pKeyPos – [out] The buffer supplied by the application layer where the position of the key with the maximum count is to be stored.

pCount - [out] The buffer supplied by the application layer where the maximum count is to be stored

Return:

0x00 => success

0x2F => keyboard model is not found

0x2E => No Key Index found

0xE1 => function initial fail

0xE2 => send command fail

0xE6 => invalid input parameters

0xE7 => received command fail

0xF0 => Checksum error

0xF1 => Invalid command

0xFB => Length error

Example:

```
int nKeyPos=0,nCount=0,ret=0;
CString str;
model_prop modelProp;
//get keyboard model first
ret = GetKbModel(&modelProp);
if (ret == 0) {
    ret = GetMaxPressedCount(&nKeyPos,&nCount);
    if (ret == 0) {
        str.Format("Key position : %d, Max Count : %d", nKeyPos,nCount);
        AfxMessageBox(str.GetBuffer());
    }
}
```

```
int GetSwipeCount(int *pTotalCount, int *pSuccessCount);
```

Description:

Get the no. of accumulated swipe count and successful swipe count for the MSR reader

Input parameter:

pTotalCount– [out] The buffer supplied by the application layer where the total count of good and bad swipe is to be stored

pSuccessCount – [out] The buffer supplied by the application layer where the good swipe count is to be stored

Return:

0x00 => success

0xE1 => function initial fail (e.g. pTotalCount or pSuccessCount is NULL pointer or usb initialization fail)

0xE2 => send command fail

0xE6 => Invalid input parameter

0xE7 => received command fail

0xF0 => Checksum error

0xF1 => Invalid command

0xFB => Length error

Example

```
int nTotalCount=0,nSuccessCount=0;
int ret=0;
CString str;
ret = GetSwipeCount(&nTotalCount,&nSuccessCount);
if(ret ==0)
{
    str.Format("Total swipe : %d  \r\n Good swipe : %d  \r\n Bad swipe : %d",
        nTotalCount,nSuccessCount,(nTotalCount-nSuccessCount));
    AfxMessageBox(str);
}
```

```
int GetFirmware(char *sVer)
```

Description:

Get the firmware version of the keyboard

Input parameter:

sVer - [out] The buffer supplied by the application layer where the firmware version is to be stored

Return:

If the function succeeds, the return value is zero.

If the function fails, the return value is non-zero.

Example

```
int nCount=0,ret=0;
CString str,str1;
str.GetBuffer(25);
ret = GetFirmware((char *)str.GetBuffer());
if(ret == 0)
{
    str1.Format("Firmware version : %s",str.GetBuffer());
    GetDlgItem(IDC_OUTPUT)->SetWindowText(str1);
    str.ReleaseBuffer();
}
```