

KEYLINK™ II/KEYLINK™ III MIGRATION GUIDELINE FOR LEGACY APPLICATIONS

Introduction.....	2
KeyLink™ II vs. KeyLink™ III.....	2
DLL function calls/returns.....	2
Function Summary.....	3
Example	4

Introduction

This application note serves as a guideline to help developers migrate KeyLink™ II based software applications to KeyLink™ III units. The term “KeyLink” will be used to refer to any of the KeyLink family Reader/Writers (SlimLink™, Keytroller™ etc.)

KeyLink™ II vs. KeyLink™ III

The KeyLink III readers are very similar to the KeyLink II readers. The KeyLink III/SlimLink™ III has the following differences:

- Significantly higher data transfer rates and faster memory erase cycles. This is particularly important when working with larger SPI-Flash type memories.
- Supports block protect of SPI and SPI Flash devices.
- The KeyLink III supports a USB interface only
- LED stays lit:
 - green indicates device is connected and idle (safe to insert or remove key or token)
 - red indicates memory device is being accessed

DLL function calls/returns

In general, functions in both sets return an integer value of either 0 to indicate successful command execution or a non-zero value if an exception occurred. When mapping existing KeyLink II based application software to work with the KeyLink III, it is probably recommended that you rely on only success or non-success values of the function return. In other words, you may have to modify your application code in areas that execute conditional code based on exception return values. See the following example:

```
/******  
*   sample  
*  
#define DK_ERROR_NO_HANDLE 1  
*****/  
int nReturn;  
nReturn = DkIsKeyInserted(hPortHandle)  
if(nReturn != 0)  
switch(nReturn)  
{  
  case 1 :  
    DisplayError( DK_ERROR_NO_HANDLE);  
    break;  
  
  case 7:  
    DisplayError(DK_ERROR_KEY_NOT_INSERTED);  
  
}
```

****** WARNING ******
KeyLink™ III version of this function will only:

- 0 (SUCCESS)
- 7 (Key not inserted)

review KeyLinkIII DevKit Manual for error codes.

Function Summary

The KeyLink™ III function exports can be viewed as a superset of the KeyLink™ II functions. Refer to the KeyLink III Serial Development Kit Manual for specific error return values.

KeyLink™ II
DkOpenSerial(int com,HANDLE& handle, int baud)
DkReadKey(HANDLE handle,char* buffer,int size,long offset, int cycles)
DkWriteKey(HANDLE handle,char* data,int size,long offset, int cycles)
DkWriteHexKey(HANDLE handle,char* data,int size,long offset, int cycles)
DkCloseSerial(HANDLE handle)
DkIsKeyInserted(HANDLE handle)
DkReadConfig(HANDLE handle,char* buffer)
DkWriteConfig(HANDLE handle,char* data)
DkEraseKey(HANDLE handle)
DkReadFWRev(HANDLE handle, unsigned char* buffer)
DkReadSerialNum(HANDLE handle,unsigned char* buffer)

KeyLink™ III (shaded are legacy support only)
DkIsReaderReady()
DkResetDevice(void)
Dk3ReadFWRev(unsigned char* buffer)
Dk3ReadConfig(char* buffer)
Dk3WriteConfig(char* data)
Dk3ReadKeySize(unsigned char* buffer)
Dk3IsKeyInserted()
Dk3ReadKeyStatus(unsigned char* packet)
Dk3EraseKey()
Dk3ReadKey(char* buffer,int size,long offset, int cycles)
Dk3WriteKey(char* data,int size,long offset, int cycles)
Dk3WriteHexKey(char* data,int size,long offset, int cycles)
Dk3ReadSerialNum(unsigned char* buffer)
DkEraseSector(char* data)
DkReadElectronicSignature(char* buffer)
DkBlockProtect(char* buffer)
DkReadBlockProtectBits(unsigned char* buffer)
DkOpenSerial(int com,HANDLE& handle, int baud)
DkCloseSerial(HANDLE handle)
DkIsKeyInserted(HANDLE handle)
DkReadKeyStatus(HANDLE handle,unsigned char* packet)
DkReadKeySize(HANDLE handle,unsigned char* buffer)
DkReadConfig(HANDLE handle,char* buffer)
DkWriteConfig(HANDLE handle,char* data)
DkEraseKey(HANDLE handle);
DkReadKey(HANDLE handle,char* buffer,int size,long offset, int cycles)
DkWriteKey(HANDLE handle,char* data,int size,long offset, int cycles)
DkWriteHexKey(HANDLE handle,char* data,int size,long offset, int cycles)
DkWritePageKey(HANDLE handle,char* data,int size,long offset, int cycles)
DkReadFWRev(HANDLE handle, unsigned char* buffer)
DkReadSerialNum(HANDLE handle,unsigned char* buffer)

Example

The following example shows how an existing KeyLink™ II application could be converted to work with a KeyLink™ III reader. The code demonstrates how you could use either the existing KeyLink II function name or use a similar KeyLink III function call.

```
/******  
*      TestFunc  
*  
*      Description:  
*      Sample function to show behavior of KeyLink II vs. KeyLink III commands.  
*      Code checks for presence of reader - then checks firmware rev  
*  
*      Parameters:  
*      bUseKL3      - bool to tell function to use new style KeyLink III command  
*  
*      Return value:  
*****/  
#define SUCCESS  
void TestFunc(bUseKL3)  
{  
    int nBaud = 9600;  
    HANDLE hPortHandle;  
  
    int numbytes = 32;      //Set maximum number of bytes to 32 const  
    int myoffset = 0;      //Set offset  
    int mycycles = 1;      //Set number of 32 byte cycles to read  
  
    int nReturn;  
  
    // ***** Check if reader is ready *****  
    if(!bUseKL3)  
    {  
        // Check for Device on Com port 1  
        // KL3 version of DkOpenSerial will ignore all 3 params  
        nReturn = DkOpenSerial(intPort, hPortHandle, nBaud);  
    }  
    else // check with new KL3 function  
        nReturn = DkIsReaderReady();  
  
    if nReturn != 0 // DK_SUCCESS  
    {  
        m_MainConsole.AddString("KeyLinkIII not detected");  
        return;  
    }  
  
    // ***** Check firmware version of reader *****  
  
    CString csString;  
    unsigned char* data = new unsigned char[43];  
  
    if(!bUseKL3)  
    {  
        // KL3 DLL version will ignore hPortHandle parameter  
        nReturn = DkReadFWRev(hPortHandle,data);  
    }  
  
    else  
    {  
        nReturn = Dk3ReadFWRev(data);  
    }  
}
```

```

}

// ***** Check if Key is present *****
if(!bUseKL3)
{
    // KL3 DLL version will ignore hPortHandle parameter
    nReturn = DkIsKeyInserted(HANDLE handle);
}

else
{
    nReturn = Dk3IsKeyInserted();
}

if(nReturn !=0)
{
    m_MainConsole.AddString("Key NOT detected");
    delete [] data;
    return;
}

// ***** Erase Key *****
if(!bUseKL3)
{
    // KL3 DLL version will ignore hPortHandle parameter
    nReturn = DkEraseKey(hPortHandle);
}

else
{
    nReturn = Dk3EraseKey();
}
}

```



Datakey[®] ELECTRONICS

12730 Creek View Avenue
Savage, MN 55378-2618 USA

Phone: 952-746-4066

Fax: 952-746-4061

Toll-free Phone: 800-328-8828

Toll-free Fax: 866-289-4212

info@datakeyelectronics.com

www.datakey.com



©2007 Datakey Electronics, Inc. All Rights Reserved. (01/07) Datakey Electronics, Inc. products, images and marketing materials are protected by various patents, copyrights and/or trademarks. Datakey Electronics, Inc. assumes no responsibility for any errors which may appear in this document, reserves the right to change devices or specifications detailed herein at any time without notice, and does not make any commitment to update the information contained herein. No licenses to patents or other intellectual property of Datakey Electronics, Inc. are granted by the Company in connection with the sale of Datakey Electronics, Inc. products, expressly or by implication.