AVR153：標準HIDｸﾗｽに基づくUSB PCﾄﾞﾗｲﾊﾞ

要点

- Windows® 98SEまたはそれ以降によって支援
- 全二重通信
- EP0を通す命令送出
- どのｺﾝﾊﾟｲﾗ(VC++、JAVA、VB、～)によっても支援される動的ﾘﾝｸ ﾗｲﾌﾞﾗﾘ(DLL)
- VC++応用に対する装置の自動検出
- 点対点通信

1. 序説

この応用記述はあなたの応用でUSB HID DLLの統合方法を記述します。提供される例はVC++とJAVAのｺﾝﾊﾟｲﾗに基づきますが、DLLはどのｺﾝﾊﾟｲﾗ(VB、Delphi、LabView、～)でも使うことができます。

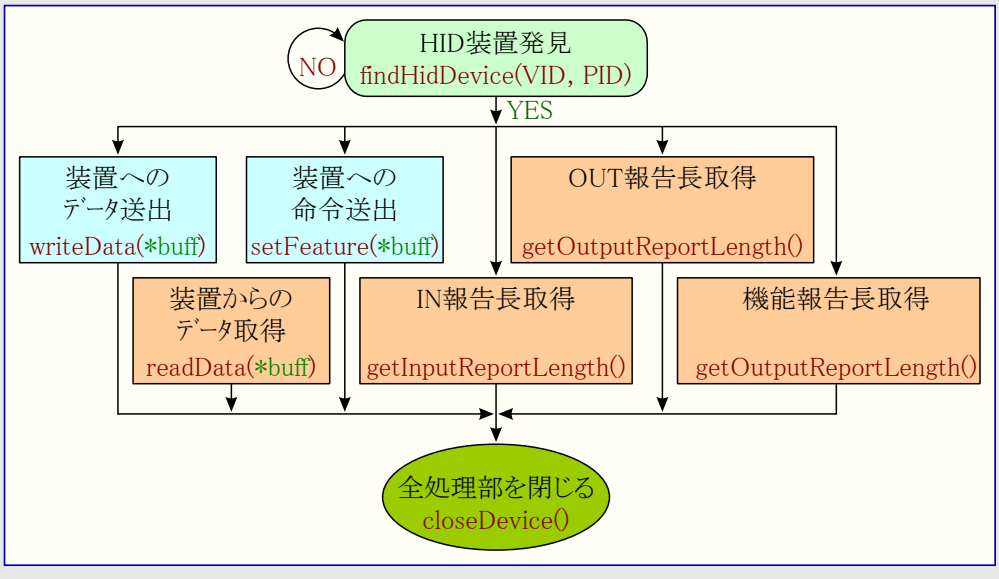
各種形式の実装を実演する簡素なｺｰﾄﾞ例が与えられます。

2. DLL関数

USB HID仕様で詳述されるように、標準HID応用はﾃﾞｰﾀを送受信するのに2つの報告(IN/OUT)を使います。これらの報告の長さはﾌｧｰﾑｳｪｱで割り当てられ、ﾌｧｰﾑｳｪｱ設定に従ってDLLによって自動的に検知されます(必要な場合にこれらの値を変更する方法を知るにはUSB標準HID実装応用を参照してください)。

このDLLが同時に1つの標準HID装置と1つの装置だけとの通信をあなたに許すことに注意してください。このDLLを使って同時に多数の装置を管理することはできません。

図2-1. DLL関数



2.1. findHidDevice

この関数(BOOLEAN)は装置が接続された場合に供給者ID(VID)/製品ID(PID)を用いて標準HID装置を発見してﾊﾝﾄﾞﾙを開きます。

入力

const UINT VendorID ：これは供給者IDです。
const UINT ProductID：これは製品IDです。

出力

FALSE：装置が見つからない場合。より多くの情報はGetLastError()で得られます。
GetLastErrorは以下を返します。
装置が見つからなかった場合、ERROR_USB_DEVICE_NOT_FOUND
装置発見、しかし素性取得不能の場合、ERROR_USB_DEVICE_NO_CAPABILITIES
TRUE：接続が成功してﾊﾝﾄﾞﾙが開かれた場合。

8ﾋﾞｯﾄ AVR® マイクロ コントローラ

応用記述

本書は一般の方々の便宜のため有志により作成されたもので、Atmel社とは無関係であることを御承知ください。しおりのはじめにでの内容にご注意ください。

Rev. 7645B-07/08, 7645BJ2-05/21

2.2. closeDevice

USB装置との通信と関連する全てのﾊﾝﾄﾞﾙを閉じます。

2.3. writeData

この関数(BOOLEAN)は装置にﾃﾞｰﾀを送ります(ﾃﾞｰﾀOUT)。この関数によって支援される最大ﾃﾞｰﾀ長はgetOutputReportLength関数(3頁の2.9.項をご覧ください)によって与えられる値と等しいか、またはより小さくなければなりません。

ﾃﾞｰﾀ長がﾌｧｰﾑｳｪｱで指定された最大長を超える場合、使用者はﾃﾞｰﾀを複数ﾊﾟｹｯﾄで送らなければなりません。

ﾃﾞｰﾀ長が最大長よりも小さい時に、この関数は0($00)の残りﾊﾞｲﾄで完全にします。

入力

UCHAR* buffer ：送るべきﾊﾟｹｯﾄのﾎﾟｲﾝﾀ

2.3.1. 出力

FALSE：ﾃﾞｰﾀ転送失敗の場合。GetLastError()はERROR_WRITE_FAULT符号を返すでしょう。
TRUE ：ﾊﾟｹｯﾄが成功裏に転送された場合。

2.4. readData

この関数(BOOLEAN)は装置によって送られたﾃﾞｰﾀ ﾊﾟｹｯﾄを読みます(ﾃﾞｰﾀIN)。ﾃﾞｰﾀ喪失を避けるため、この関数は(ｽﾚｯﾄﾞまたは計時器を用いて)継続的動作形態で呼び出されるべきです。

入力

UCHAR* buffer ：受信したﾊﾟｹｯﾄを含む緩衝部へのﾎﾟｲﾝﾀ

緩衝部はgetInputReportLength関数(3頁の2.10.項をご覧ください)によって与えられたIN報告の長さを持たなければなりません。

2.4.1. 出力

FALSE：利用可能なﾃﾞｰﾀがない場合。
TRUE ：ﾃﾞｰﾀが受信されて緩衝部に格納された場合。

2.5. setFeature

この関数(BOOLEAN)はHID装置を制御するための命令ﾃﾞｰﾀ送信(換言すると、ﾌﾞｰﾄﾛｰﾀﾞ開始、新規作業開始、～)を使用者に許します。ﾃﾞｰﾀは"SetReport"要求としてｴﾝﾄﾞﾎﾟｲﾝﾄ0上に送出されます(更なる情報についてはHID仕様を参照してください)。INとOUTのｴﾝﾄﾞﾎﾟｲﾝﾄは適用できる生ﾃﾞｰﾀ転送にだけ使われます。

ﾃﾞｰﾀ長はﾌｧｰﾑｳｪｱによって固定化され、getFeatureReportLength関数を用いて得られます(3頁の2.11.項を参照してください)。ﾃﾞｰﾀ長はgetFeatureReportLength関数によって返された長さを超えてはなりません。

入力

UCHAR* buffer ：受信したﾊﾟｹｯﾄを含む緩衝部へのﾎﾟｲﾝﾀ

出力

FALSE：ﾃﾞｰﾀ転送失敗の場合。
TRUE ：ﾃﾞｰﾀが上手く転送された場合。

2.6. hidRegisterDeviceNotification

この関数がVC++ﾌﾟﾛｼﾞｪｸﾄだけで使うことができることに注意してください。

この関数は新しいﾌﾟﾗｸﾞ&ﾌﾟﾚｲ装置が接続または切断されたかを応用に通知します。

入力

HWND hWnd – ｳｨﾝﾄﾞｳへのﾊﾝﾄﾞﾙ

出力

FALSE：関数が失敗の場合。拡張異常情報を得るにはGetLastErrorを呼んでください。
TRUE ：関数が成功した場合。

2.7. hidUnregisterDeviceNotification

この関数がVC++ﾌﾟﾛｼﾞｪｸﾄだけで使うことができることに注意してください。

この関数は指定された装置通知ﾊﾝﾄﾞﾙを閉じます。

入力

HWND hWnd – ｳｨﾝﾄﾞｳへのﾊﾝﾄﾞﾙ

出力

FALSE ：関数が失敗の場合。拡張異常情報を得るにはGetLastErrorを呼んでください。
TRUE ：関数が成功した場合。

2.8. isMyDeviceNotification

この関数がVC++ﾌﾟﾛｼﾞｪｸﾄだけで使うことができることに注意してください。

この関数は"hidRegisterDeviceNotification"によって通知された(接続された、または切断された)新しい装置が使用済みHID装置か否かの調査を許します。

入力

DWORD dwData – OnDeviceChangeの第2ﾊﾟﾗﾒｰﾀによって与えられる値

出力

FALSE ：接続/切断された装置が使用済みHID装置の場合。
TRUE ：それが別の装置の場合。

2.9. getOutputReportLength

本関数はOUT報告(PCから装置へ送られたﾃﾞｰﾀ ﾊﾟｹｯﾄ)の長さの取得を使用者に許します。この値はﾌｧｰﾑｳｪｱで指定されます。

2.10. getInputReportLength

本関数はIN報告(装置からPCへ送られたﾃﾞｰﾀ ﾊﾟｹｯﾄ)の長さの取得を使用者に許します。この値はﾌｧｰﾑｳｪｱで指定されます。

2.11. getFeatureReportLength

本関数は機能報告(PCから装置へ送られた制御ﾃﾞｰﾀ ﾊﾟｹｯﾄ)長の取得を使用者に許します。この値はﾌｧｰﾑｳｪｱで指定されます。

3. PC実演

3.1. VC++実演

VC++実演はﾌﾟﾛｼﾞｪｸﾄにAtUsbHid.Dllを設定する方法、それとﾌﾟﾗｸﾞ&ﾌﾟﾚｲ通知の使用方法をも使用者に見せることを許します。

3.1.1. Visual C++応用でのDLL設定

AtUsbHid.hﾌｧｲﾙはAtmelのUSB HID DLLに存在する関数の設定と使用の手助けをするﾏｸﾛを提供します。

DLLを用いて応用を設計する時に以下を行う必要があります。

- DLL用のﾊﾝﾄﾞﾗｰ(HINSTANCE hLIb=NUL;)を作成してください。
- 関数hLib=LoadLibrary(AT_USB_HID_DLL);を用いてDLLを設定してください。
- loadFuncPointers(hlib)を用いて各DLL関数を設定してください。

一旦、異常なしでそれらの段階が実行されてしまうと、DLLとそれの関数はあなたの応用に設定され、DYNCALL(DLL関数())ﾏｸﾛを用いて呼び出すことができます。

応用が停止される時にﾒﾓﾘからDLLを開放するのにFreeLibrary(hLib)関数の使用が便利です。

ﾒﾓﾘからDLLを開放する前にUSB装置ﾊﾝﾄﾞﾙが閉じられていることを確実にしなければなりません。

3.1.2. 自動装置接続/切断機能の使用

DLLは装置の接続/切断の検知を使用者に許す関数を提供します。

この機能を実行するには以下の活動を行わなければなりません。

以下を用いて装置変更通知を得るためにあなたの応用を登録してください。
DYNCALL(hidRegisterDeviceNotification)((m_hWnd))

あなたのﾒｯｾｰｼﾞ配置応用にON_WM_DEVICECHANGE()関数を追加してください。

装置状況変更の度毎に呼び出される、OnDeviceChange(UINT nEventType, DWORD dwData)関数呼び出しを作成してください。

OnDeviceChange関数であなたの装置の状態が変更(接続/切断)されたかを知るためにDYNCALL(isMyDeviceNotification(dwData))関数を呼び出してください(UsbHidDemoCodeDlg.cpp内の実演ｺｰﾄﾞをご覧ください)。

応用を抜け出す時に、DYNCALL(hidUnregisterDeviceNotification(m_hWnd))関数の使用が登録解除に便利です。

3.1.3. readDataの使い方

データが装置によって継続的に送出され得るため、計時器に基づく関数を用いてデータを読むことが重要です。これはreadData関数を継続的にポーリングすることを許します。

それを行うには以下を行わなければなりません。

あなたのメッセージ配置応用にON_WM_TIMER()関数を追加してください。

DYNCALL(readData(sbuffer))関数を呼ぶOnTimer(UINT nIDEvent)関数を作成してください。

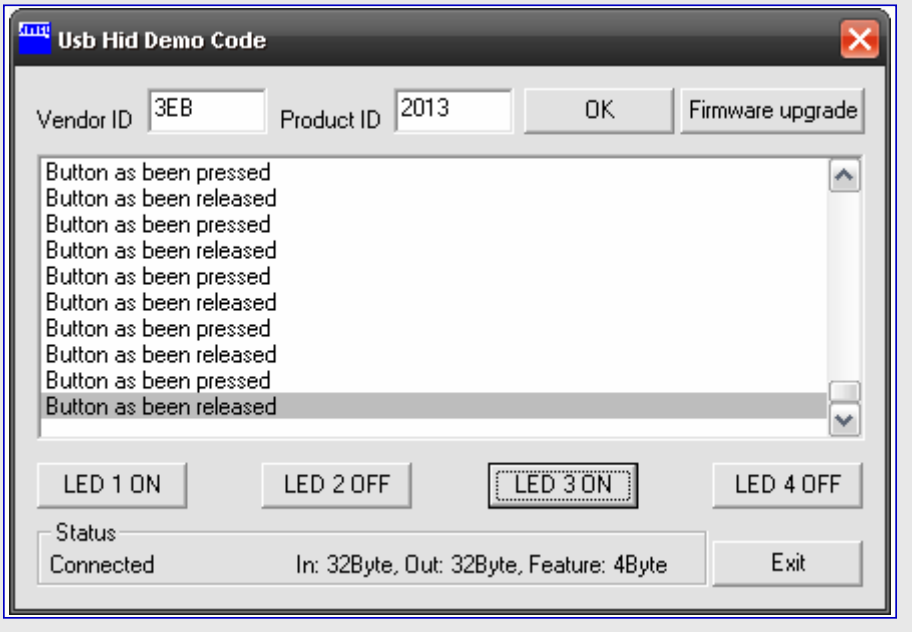
今やあなたの装置が接続された時にxms毎にreadData関数を呼び出すように、SetTimer(n,x,y);を用いて計時器を指定した間隔に設定することができます。

あなたの装置が切断された時にKilTimer(n)を用いて計時器を解除してください。

3.1.4. 使用者インターフェース

この下は提供される実演の画面描写です。既定PIDが1つの特定実演に関連付けされていることに注意してください(標準HIDインターフェースを持つAtmelの実演は同じPIDを持ちません)。あなたが使う装置と合わせるためにこのPIDパラメータを変更しなければならないかもしれません(あなたの実演によって使われるVID/PIDを得るにはファームウェアまたはデバイス マネージャを参照してください)。

図3-1. VC++に基づく実演



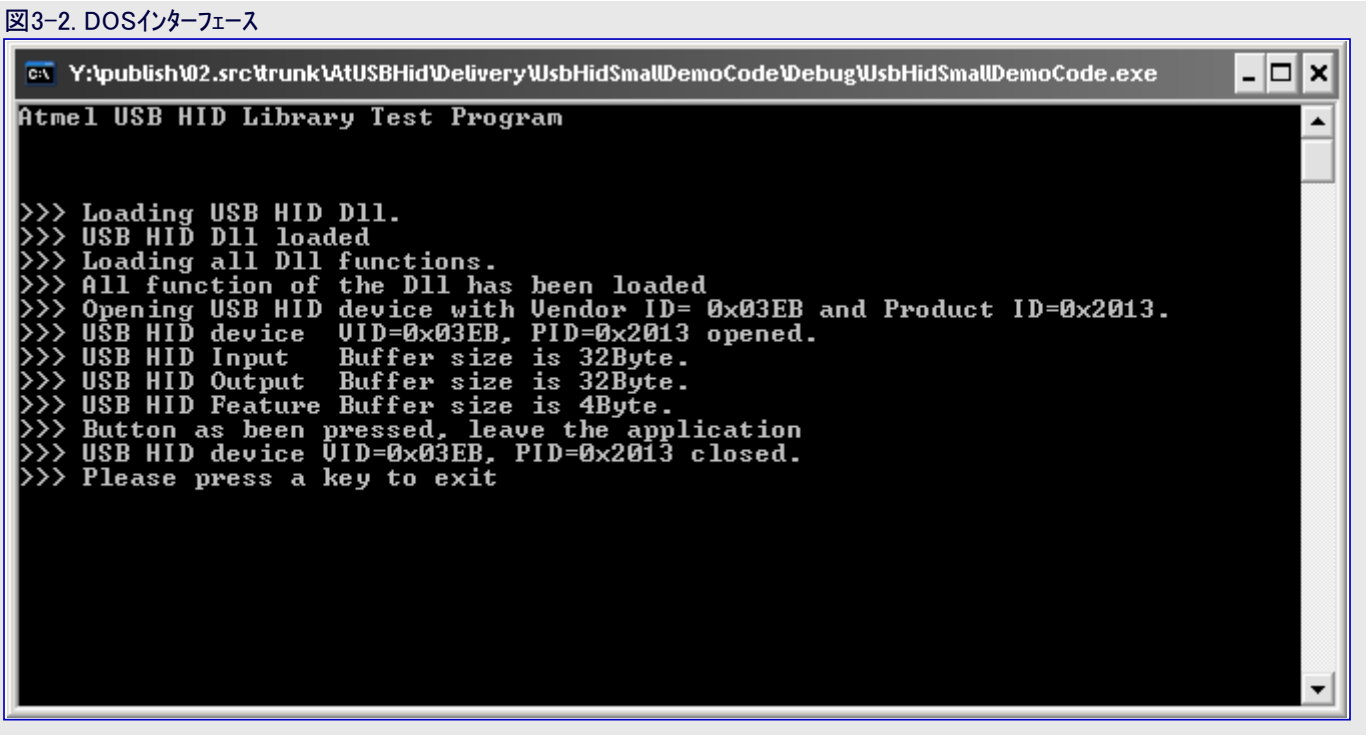
この下はGUI部品成分の記述です。

- Vendor ID、Product IDの枠は装置のVID/PIDを指定するのに使われます。
- 一旦VID/PIDが正しく設定されると、OK釦が押されるべきです。
- LED1～LED4釦は基板のLEDのON/OFF切り替えに使われます。
- Firmware Upgrade釦はUSBインターフェースを通してファームウェアを更新ためのブートローダの開始を使用者に許します(更なる詳細についてはブートローダのデータシートを参照してください)。
- Exit釦は応用を閉じます。
- Status領域は接続状況を与え、装置が接続された時のIN報告、OUT報告、機能報告の長さも与えます(これらのパラメータはデータを送受信するDLLによって自動的に使われます)。

3.1.5. DOS実演

この実演は簡単なｺﾝｿｰﾙ応用例を与えます。この実演は固定のVID/PIDを使い、これらのﾊﾟﾗﾒｰﾀを変更するには再ｺﾝﾊﾟｲﾙされなければなりません。このｺﾝｿｰﾙ応用を実行する前に、装置は接続されて標準HIDﾌｧｰﾑｳｪｱで走行しなければなりません。

図3-2. DOSｲﾝﾀｰﾌｪｰｽ



注: このﾌﾟﾛｼﾞｪｸﾄはMinGw(www.migw.org)を用いてｺﾝﾊﾟｲﾙすることができます。ｺﾏﾝﾄﾞ行は次のとおりです。

```
mingw32-g++ -O2 -Wall  UsbHidSmallDemoCode.cpp -o AtUsbHidMinGw.exe -I
```

3.2. JAVA実演

JAVA実演はJAVAﾌﾟﾛｼﾞｪｸﾄにAtUsbHid.dllを統合する方法を知ることを使用者に許します。

AtUsbHid.dllとJAVA間のｲﾝﾀｰﾌｪｰｽはAtUsbHidJni.jar一括を通して行われます。

3.2.1. AtUsbHid.dll統合

AtUsbHid.dllを統合するには下の段階に従わなければなりません。

・あなたのJAVAﾌｧｲﾙのｲﾝﾎﾟｰﾄ領域に以下のｺｰﾄﾞを追加してください。

```
import com.atmel.atusbhidjni.AtUsbHidJni
```

・DLLを使うために新しいｵﾌﾞｼﾞｪｸﾄを作成してください。

```
AtUsbHidJni usbDevice = new AtUsbHidJni();
```

・DLLを設定してください。

```
usbDevice.loadLibraryUsbHid();
```

・今や、DLLは使用準備が整っています。DLL機能に関する更なる詳細については「DLL関数」章を参照してください。

・応用を抜け出す時にDLLを設定解除することが重要です。

```
usbDevice.UnloadloadLibraryUsbHid();
```

・ﾌﾟﾛｼﾞｪｸﾄをｺﾝﾊﾟｲﾙするにはAtUsbHidJni.jar一括のｸﾗｽ ﾊﾟｽに追加してください。

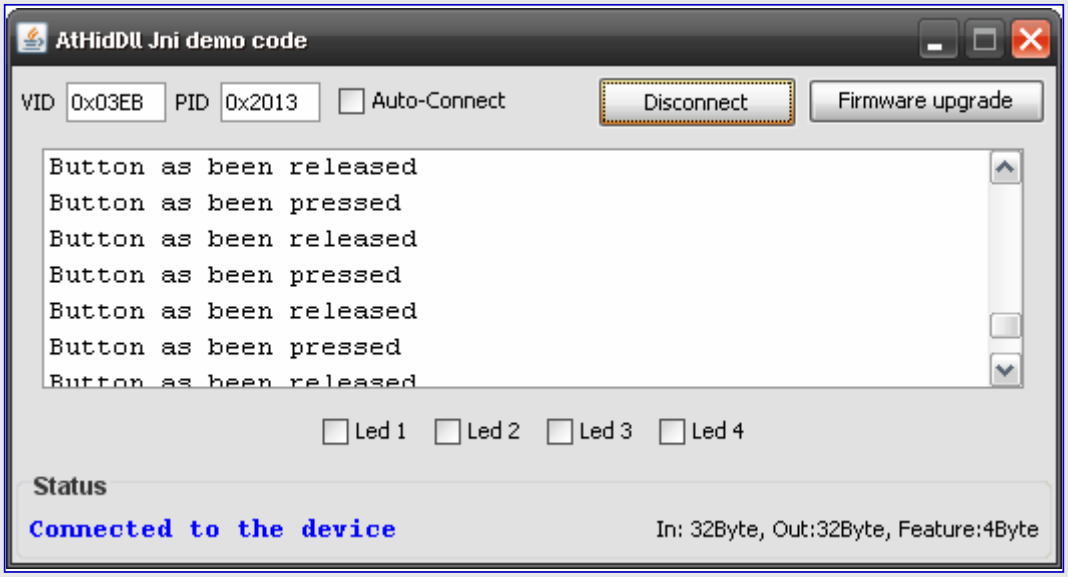
```
JAVAc userhid.JAVA -classpath AtUsbHidJni.jar
```

注: 更なる情報についてはDLL一括と共に提供されるHTML資料を参照してください。

3.2.2. 使用者インターフェース

GUIソース コードはJNICodeForHIDDLLフォルダで利用可能です。この下はJAVA使用者インターフェースです。

図3-3. JAVA使用者インターフェース

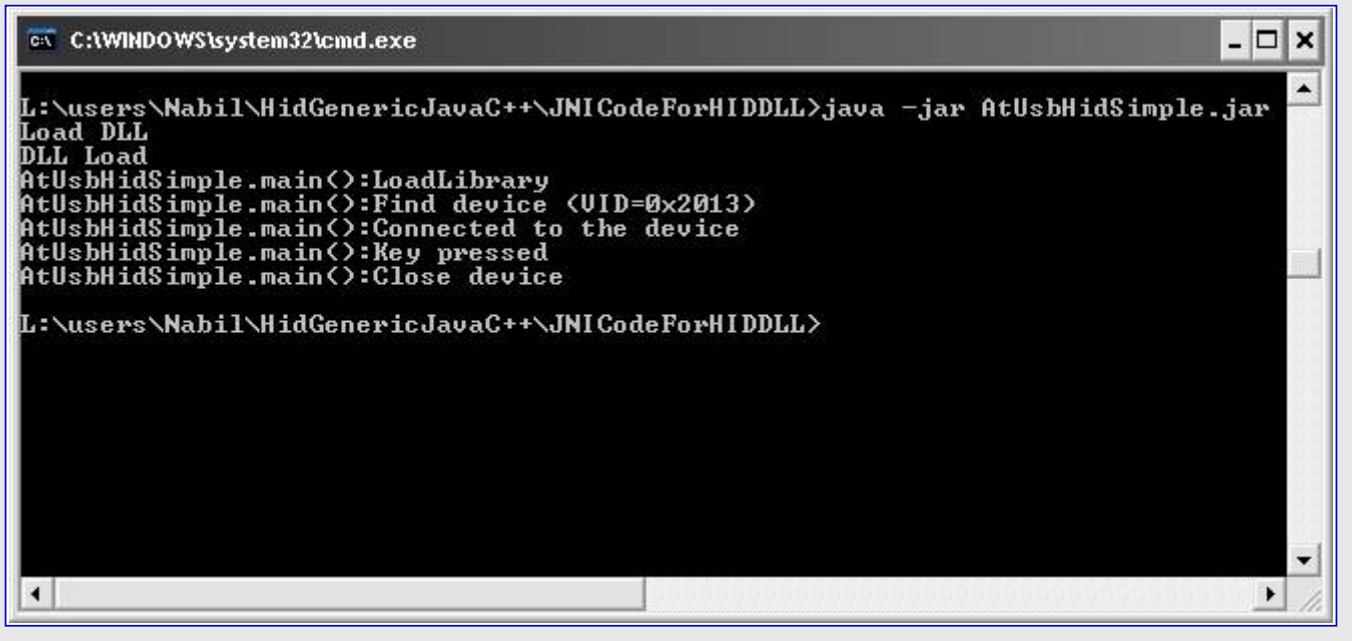


部品成分はVC++インターフェース用に記述されたものと同じ役割を持ちます(3.1.4.項参照)。Auto-Connect枠は装置の接続/切断の自動検出を応用に許すのに使われます。

3.2.3. DOS実演

この実演は簡単なコンソール応用例を与えます。この実演は固定のVID/PIDを使い、これらのパラメータを変更するには再コンパイルされなければなりません。このコンソール応用を実行する前に、装置は接続されて標準HIDファームウェアで走行しなければなりません。

図3-4. DOSインターフェース



4. 1式構造

DLL一括を解凍する時に多数のフォルダを見つけるでしょう。この下はその各々の内容です。

4.1. AtUsbHid

このフォルダはAtUsbHid.dllとAtUsbHid.hのファイルを含みます。

4.2. ExeDemo

このフォルダは各種の実行可能実演例を含みます。

4.3. JNICodeForHidDLL

このﾌｫﾙﾀﾞはJAVAﾌﾟﾛｼﾞｪｸﾄのｿｰｽ ｺｰﾄﾞを含みます。

4.4. UsbHidDemoCode

このﾌｫﾙﾀﾞはVC++ﾌﾟﾛｼﾞｪｸﾄのｿｰｽ ｺｰﾄﾞを含みます。

4.5. UsbHidSmallDemoCode

このﾌｫﾙﾀﾞはVC++小規模実演(DOS実演)のｿｰｽ ｺｰﾄﾞを含みます。

本社

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
TEL 1(408) 441-0311
FAX 1(408) 487-2600

国外営業拠点

Atmel Asia
Unit 1-5 & 16, 19/F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
Hong Kong
TEL (852) 2245-6100
FAX (852) 2722-1369

Atmel Europe
Le Krebs
8, Rue Jean-Pierre Timbaud
BP 309
78054 Saint-Quentin-en-
Yvelines Cedex
France
TEL (33) 1-30-60-70-00
FAX (33) 1-30-60-71-11

Atmel Japan
104-0033 東京都中央区
新川1-24-8
東熱新川ﾋﾞﾙ 9F
ｱﾄﾒﾙ ｼﾞｬﾊﾟﾝ株式会社
TEL (81) 03-3523-3551
FAX (81) 03-3523-7581

製品窓口

ｳｪﾌﾞｻｲﾄ
www.atmel.com

技術支援
avr@atmel.com

販売窓口
www.atmel.com/contacts

文献請求
www.atmel.com/literature

本応用記述はAtmelのAVR153応用記述(doc7645.pdf Rev.7645B-07/08)の翻訳日本語版です。日本語では不自然となる重複する形容表現は省略されている場合があります。日本語では難解となる表現は大幅に意訳されている部分もあります。必要に応じて一部加筆されています。頁割の変更により、原本より頁数が少なくなっています。

必要と思われる部分には( )内に英語表記や略称などを残す形で表記しています。

青字の部分はﾘﾝｸとなっています。一般的に赤字の0,1は論理0,1を表します。その他の赤字は重要な部分を表します。