汎用 USB-IO for MATLAB

ここでは、HID デバイスとして認識するモルフィー企画の USB-IO を MATLAB 上で使うための手順について説明する。
モルフィー企画の USB-IO は、HID デバイスとして認識するため、特別なドライバソフトはいらない。
そのため、USB - IO とそれの入出力を行うための usbg.dll（このドキュメント内で作成する。）さえあれば、8 ビット + 4 ビットのデータの入出力が可能になる。

MATLAB 用の MEX プログラミング手順

STEP.1
外部ハードウェアとの通信を確実にしておくとどちらが間違えているのかわからないため、まず、DOS 環境で実行できる C プログラムを作成する。（main 関数を含め形で作成する。）

STEP.2
初期化関数、入力関数、出力関数の 3 つの機能に分け関数化する。

STEP.3
main 関数をコメントアウトし mexFunction 関数から引数に応じ適切に入出力関数を呼び出せるようにプログラミングしていく。

USB-IO を使うためのプログラミング環境設定

汎用 USB のドライバを作成するためには VisualC++ 6.0 に Win2000DDK をインストールする必要がある。また、ライブラリとして setupapi.lib, hid.lib を使用するため、環境変数は以下のものを追加する。

PATH
C:\Program Files\Microsoft Visual Studio\Common\Tools\WinNT; 
C:\Program Files\Microsoft Visual Studio\Common\MSDev98\Bin; 
C:\Program Files\Microsoft Visual Studio\Common\Tools; 
C:\Program Files\Microsoft Visual Studio\VC98\bin

LIB
C:\Program Files\Microsoft Visual Studio\VC98\mfc\lib; 
C:\Program Files\Microsoft Visual Studio\VC98\lib; 
c:\ntddk\lib; 
c:\ntddk\libchk\i386

INCLUDE
C:\Program Files\Microsoft Visual Studio\VC98\atl\include; 
C:\Program Files\Microsoft Visual Studio\VC98\mfc\include; 
C:\Program Files\Microsoft Visual Studio\VC98\include; 
c:\ntddk\inc

DOS プログラムの作成

Visual C++での環境が整ったら、usbest1.c をコンパイル実行してみよう。

コンパイル実行は、
ci usbest1.c

すれはばよい。このプログラムでは、HID インターフェースの GUID を取り出すプログラムである。
なお、ここで使っている pragma ディレクティブは、コンパイルする際の外部ライブラリをリンクするためのコードである。usbest1.c

```
#include <stdio.h>
#include <stdlib.h>
#include <stddef.h>
#include <string.h>
#include <windows.h>
#include <setupapi.h>
#include <hidsdi.h>
#pragma comment(lib, "setupapi.lib")
#pragma comment(lib, "hid.lib")

void main(void){
   GUID hidGuid;
   HidD_GetHidGuid(&hidGuid);
   printf("%x %x %x
", (unsigned long)hidGuid.Data1,(unsigned long)hidGuid.Data2,(unsigned long)hidGuid.Data3);
}
```

コンパイル実行結果

C:\>ci usbest1.c
Microsoft (R) 32-bit C/C++ Optimizing Compiler Version 12.00.8804 for 80x86
Copyright (C) Microsoft Corp 1984-1998. All rights reserved.
usbest1.c
Microsoft (R) Incremental Linker Version 6.00.8447
Copyright (C) Microsoft Corp 1992-1998. All rights reserved.
/out:usbest1.exe
usbest1.obj
C:\>usbest1 4d1e55b2 f16f 11cf
この番号が GUID の一部である、実は、hidGuid.Data4[8]もあるが、これは MAC アドレスなどがマシンの特定に使われるものらしい。（詳しくは未確認）なお、GUID はマシンによりユニークな番号が割り当てられるためここで示した番号とは異なるかもしれない。GUID が取得できなかったのでさらにプログラムを拡張していく。

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <windows.h>
#include <setupapi.h>
#include <hidsdi.h>
#pragma comment(lib, "setupapi.lib")
#pragma comment(lib, "hid.lib")

void main(void){
    int i, j;
    int count;
    GUID hidGuid;
    HDEVINFO dis;
    BOOL r;
    DWORD size;
    SP_DEVICE_INTERFACE_DATA sdid;
    PSP_DEVICE_INTERFACE_DETAIL_DATA_A pdidd = NULL;
    size_t didd_size = 0;
    HANDLE dh, fd;
    HIDD_ATTRIBUTES attr;
    size_t len;
    char buffer[255];
    HidD_GetHidGuid(&hidGuid);
    dis = SetupDiGetClassDevsA(&hidGuid, NULL, 0, DIGCF_PRESENT | DIGCF_DEVICEINTERFACE);
    count = 0;
    for (i = 0; ; i++) {
        memset(&sdid, 0, sizeof(sdid));
        sdid.cbSize = sizeof(SP_DEVICE_INTERFACE_DATA);
        r = SetupDiEnumDeviceInterfaces(dis, NULL, &hidGuid, i, &sdid);
        if (!r) break;
        size = 0;
        r = SetupDiGetDeviceInterfaceDetailA(dis, &sdid, NULL, 0, &size, NULL);
        if (r || GetLastError() != ERROR_INSUFFICIENT_BUFFER) continue;
        if (size > didd_size) {
            didd_size = (size + 15) & ~15;
            pdidd = malloc(didd_size);
        }
        memset(pdidd, 0, didd_size);
        pdidd->cbSize = sizeof(SP_DEVICE_INTERFACE_DETAIL_DATA_A);
        r = SetupDiGetDeviceInterfaceDetailA(dis, &sdid, pdidd, size, NULL, NULL);
        if (!r) continue;
        dh = CreateFileA((LPCSTR)pdidd->DevicePath, GENERIC_READ | GENERIC_WRITE, FILE_SHARE_READ, NULL, OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, NULL);
        if (dh == INVALID_HANDLE_VALUE) continue;
        memset(&attr, 0, sizeof(HIDD_ATTRIBUTES));
        attr.Size = sizeof(HIDD_ATTRIBUTES);
        HidD_GetAttributes(dh, &attr);
        CloseHandle(dh);
        print("%s
", (LPCSTR)pdidd->DevicePath);
        print("%s %s %x
", attr.VendorID, attr.ProductID, attr.VersionNumber);
        count++;
    }
    SetupDiDestroyDeviceInfoList(dis);
}
```

このプログラムでは、USB-ID へのファイルハンドル名とベンダ ID、プロダクト ID を検出、表示している。ここで、HID タイプの USB デバイスが複数接続されていた場合、これらの出力も複数出るようにプログラミングしてある。（プログラム途中のため malloc したまま free はしていない。）
ハンドルのファイル名は、

```c
\out:usbtest2.exe
usbtest2.obj
C:¥ >usbtest
4d1e55b2 f16f 11cf
¥¥?¥hid#vid_0bfe&pid_1003#6&99bfe71&0&0000#{4d1e55b2-f16f-11cf-88cb-001111000030 }
```

VenderID は 0xbfe, ProductID が 0x1003, VersionNumber が 1 であることがわかる。ファイルのハンドル名がわからないのは、HID デバイス情報をいらないので SetupDiDestroyDeviceInfoList により処理する。
このプログラムでは、最初に検出した HID デバイスを使うようになっている。もししっかり検出、処理したいのであれば、VenderID, ProductID などをチェックしてやると良い。（ここでは、表示をしているだけ）

ここまでくれば、後は、ファイルの出入力関数によりデータをやり取りすることができる。
以下のプログラムでは、CreateFile, ReadFile, WriteFile によりデータのやり取りを行っている。このUSB-IO では、1 回に送るデータは 9 パイトであり、その中に命令、ポート指定 Read, Write, そのパラメータを入力し読み書きするようになっている。

usbtest3.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <stddef.h>
#include <string.h>
#include <windows.h>
#include <setupapi.h>
#include <hidsdi.h>
#pragma comment(lib, "setupapi.lib")
#pragma comment(lib, "hid.lib")

void main(void){
    int i,j;
    int count;
    GUID hidGuid;
    HDEVINFO dis;
    BOOL r;
    DWORD size;
    SP_DEVICE_INTERFACE_DATA sdid;
    PSP_DEVICE_INTERFACE_DETAIL_DATA_A pdidd = NULL;
    size_t didd_size = 0;
    HANDLE dh,fd;
    HIDD_ATTRIBUTES attr;
    size_t len;
    char buffer[255];
    HidD_GetHidGuid(&hidGuid);
    printf("%x %x %x\n", (unsigned long)hidGuid.Data1, (unsigned long)hidGuid.Data2, (unsigned long)hidGuid.Data3);
    dis = SetupDiGetClassDevsA(&hidGuid, NULL, 0, DIGCF_PRESENT | DIGCF_DEVICEINTERFACE );
    count = 0;
    for (i = 0;;i++){
        memset(&sdid,0,sizeof(sdid));
        sdid.cbSize = sizeof( SP_DEVICE_INTERFACE_DATA );
        r = SetupDiEnumDeviceInterfaces(dis,NULL,&hidGuid,i,&sdid);
        if (!r) break;
        size = 0;
        r = SetupDiGetDeviceInterfaceDetailA(dis,&sdid,NULL,0,0, &size, NULL);
        if ( r ) break;
        if ( r || GetLastError() == ERROR_INSUFFICIENT_BUFFER ) continue;
        if ( size > didd_size ) {
            didd_size = ( size + 15 ) & ~15;
            pdidd = malloc(didd_size);
            memset(pdidd,0,didd_size);
            pdidd->cbSize = sizeof(SP_DEVICE_INTERFACE_DETAIL_DATA_A);
            r = SetupDiGetDeviceInterfaceDetailA(dis,&sdid,0,&size,NULL,0, &size, NULL);
            if ( !r ) continue;
            dh = CreateFileA( ( LPCSTR )pdidd->DevicePath,GENERIC_READ | GENERIC_WRITE,FILE_SHARE_READ, NULL, OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, NULL);
            if (dh == INVALID_HANDLE_VALUE) continue;
            memset(&attr,0,sizeof(HIDD_ATTRIBUTES));
            ```
以下に簡単な表を示す。buffer[1]の内容によりモードが変わる。

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>02</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>03</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>04</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

以上の注意点を元にMEX 関数化を考えていけばよい。
この場合、プログラム名はusbg.cとするなど、仕様は、以下のように定義する。

```c
#define usbg(0,255); % ポート0に255を書き込み
data=usbg(0) % 変数dataにポート0の内容を読み込む。
```

まず、このようなプログラムを作成するために、
```c
int usb_write(int port,int data),
int usb_read(int port)
int usb_init(void);
int usb_close(void);
```
の4つの機能に分けてプログラムを作成を行っていく。

```c
usbtest4.c
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <stddef.h>
#include <string.h>
#include <windows.h>
#include <setupapi.h>
#include <hidapi.h>
#pragma comment(lib, "setupapi.lib")
#pragma comment(lib, "hid.lib")
int usb_write(int port,int data,HANDLE fd){
  char buffer[9];
  int ch = 2;
  int count = 0;
  if (port == 0) ch = 1;
  ```
buffer[0]=0;
buffer[1]=(char)ch;
buffer[2]=(char)data;
buffer[3]=0xff;
buffer[4]=0xff;
buffer[5]=0xff;
buffer[6]=0xff;
buffer[7]=0xff;
buffer[8]=0xff;
return WriteFile((HANDLE)fd,buffer,9,&count,0);
}

int usb_read(int port,HANDLE fd){
    char buffer[9];
    int ch = 4;
    int count = 9;
    if (port == 0) ch = 3;
    usb_write(port,0xff,fd);
    buffer[0]=0;
    buffer[1]=(char)ch;
    buffer[2]=0xff;
    buffer[3]=0xff;
    buffer[4]=0xff;
    buffer[5]=0xff;
    buffer[6]=0xff;
    buffer[7]=0xff;
    buffer[8]=0xff;
    WriteFile((HANDLE)fd,buffer,9,&count,0);
    ReadFile((HANDLE)fd,buffer,9,&count,0);
    return buffer[2];
}

HANDLE usb_init(void){
    GUID hidGuid;
    HDEVINFO dis;
    int count,i;
    BOOL r;
    DWORD size;
    SP_DEVICE_INTERFACE_DATA sdid;
    PSP_DEVICE_INTERFACE_DETAIL_DATA_A pdidd = NULL;
    size_t didd_size = 0;
    HANDLE dh,fd;
    HIDD_ATTRIBUTES attr;
    size_t len;
    char buffer[255];
    HidD_GetHidGuid(&hidGuid);
    // printf("%x %x %x\n",(unsigned long)hidGuid.Data1,(unsigned long)hidGuid.Data2,(unsigned long)hidGuid.Data3);
    dis = SetupDiGetClassDevsA(&hidGuid, NULL, 0, DIGCF_PRESENT | DIGCF_DEVICEINTERFACE );
    count = 0;
    for (i = 0;;i++){
        memset(&sdid,0,sizeof(sdid));
        sdid.cbSize = sizeof( SP_DEVICE_INTERFACE_DATA );
        r = SetupDiEnumDeviceInterfaces(dis,NULL,&hidGuid,i,&sdid);
        if (!r) break;
        size = 0;
        r = SetupDiGetDeviceInterfaceDetailA(dis,&sdid,NULL,0,&size,NULL);
        if (r || GetLastError() != ERROR_INSUFFICIENT_BUFFER ) continue;
        if ( size > didd_size ) {
            didd_size = ( size + 15 ) & ~15;
            pdidd = malloc(didd_size);
            memset(pdidd,0,didd_size);
            pdidd->cbSize = sizeof(SP_DEVICE_INTERFACE_DETAIL_DATA_A);
            r = SetupDiGetDeviceInterfaceDetailA(dis,&sdid,pdidd,size,NULL,NULL);
            if (r) continue;
            dh = CreateFileA( ( LPCSTR )pdidd->DevicePath,GENERIC_READ | GENERIC_WRITE,FILE_SHARE_READ,0,0,FILE_ATTRIBUTE_NORMAL,NULL);
            if (dh == INVALID_HANDLE_VALUE ) continue;
            memset(&attr,0,sizeof(HIDD_ATTRIBUTES));
            attr.Size = sizeof(HIDD_ATTRIBUTES);
            HidD_GetAttributes(dh,&attr);
            CloseHandle(dh);
        }
    }
    return dh;
}
```c
void main(void) {
    int i;
    HANDLE fd = usb_init();
    for (i = 0; i < 20; i++) {
        printf("%x",usb_read(0,fd));
        usb_write(0,i,fd);
        getch();
    }
    usb_close(fd);
}
```

```c
int usb_write(int port, int data, HANDLE fd) {
    char buffer[9];
    int ch = 2;
    int count = 9;
    if (port == 0) ch = 1;
    buffer[0] = (char)0;
    buffer[1] = (char)ch;
    buffer[2] = (char)data;
    buffer[3] = (char)0xff;
    buffer[4] = (char)0xff;
    buffer[5] = (char)0xff;
    buffer[6] = (char)0xff;
    buffer[7] = (char)0xff;
    buffer[8] = (char)0xff;
    return WriteFile((HANDLE)fd, buffer, 9, &count, 0);
}
```

```c
int usb_read(int port, HANDLE fd) {
    char buffer[9];
    int ch = 4;
    int count = 9;
    if (port == 0) ch = 3;
    usb_write(port, 0xff, fd);
    buffer[0] = (char)0;
    buffer[1] = (char)ch;
    buffer[2] = (char)0xff;
    buffer[3] = (char)0xff;
    buffer[4] = (char)0xff;
    buffer[5] = (char)0xff;
    buffer[6] = (char)0xff;
    return 0;
}
```
HANDLE usb_init(void) {
    GUID hidGuid;
    HDEVINFO dis;
    int count;
    BOOL r;
    DWORD size;
    SP_DEVICE_INTERFACE_DATA sdid;
    PSP_DEVICE_INTERFACE_DETAIL_DATA_A pdidd = NULL;
    size_t didd_size = 0;
    HANDLE dh,fd;
    HIDD_ATTRIBUTES attr;
    HidD_GetHidGuid(&hidGuid);
    // printf("%x %x %x
",(unsigned long)hidGuid.Data1,(unsigned long)hidGuid.Data2,(unsigned long)hidGuid.Data3);
    dis = SetupDiGetClassDevsA(&hidGuid, NULL, 0, DIGCF_PRESENT | DIGCF_DEVICEINTERFACE);
    count = 0;
    for (i = 0;;i++){
        memset(&sdid,0,sizeof(sdid));
        sdid.cbSize = sizeof(SP_DEVICE_INTERFACE_DATA);
        r = SetupDiEnumDeviceInterfaces(dis,NULL,&hidGuid,i,&sdid);
        if (!r) break;
        size = 0;
        r = SetupDiGetDeviceInterfaceDetailA(dis,&sdid,NULL,0,&size,NULL);
        if (r || GetLastError() != ERROR_INSUFFICIENT_BUFFER) continue;
        if (size > didd_size ) {
            didd_size = (size + 15) & ~15;
            pdidd = malloc(didd_size);
        }
        memset(pdidd,0,didd_size);
        pdidd->cbSize = sizeof(SP_DEVICE_INTERFACE_DETAIL_DATA_A);
        r = SetupDiGetDeviceInterfaceDetailA(dis,&sdid,pdidd,size,NULL,NULL);
        if (r) continue;
        dh = CreateFileA( (LPCSTR)pdidd->DevicePath,GENERIC_READ | GENERIC_WRITE,FILE_SHARE_READ | FILE_SHARE_WRITE,NULL,OPEN_EXISTING,FILE_ATTRIBUTE_NORMAL,NULL);
        if (dh == INVALID_HANDLE_VALUE) continue;
        memset(&attr,0,sizeof(HIDD_ATTRIBUTES));
        HidD_GetAttributes(dh,&attr);
        CloseHandle(dh);
        mexPrintf("%s
",(LPCSTR)pdidd->DevicePath);
        mexPrintf("VendorID=%x
ProductID=%x
Version=%x
",attr.VendorID,attr.ProductID,attr.VersionNumber);
        count++;
    }
    SetupDiDestroyDeviceInfoList( dis );
    fd = CreateFile((LPCSTR)pdidd->DevicePath,GENERIC_READ | GENERIC_WRITE,FILE_SHARE_WRITE,NULL,OPEN_EXISTING,FILE_ATTRIBUTE_NORMAL,NULL);
    printf("%x
",fd);
    mexPrintf("Ready to use
");
    free(pdidd);
    return (HANDLE)fd;
}

void usb_close(HANDLE fd) {
    CloseHandle((HANDLE)fd);
}

#if 0
void main(void){
    int i;
    HANDLE fd = usb_init();
    for(i = 0;i < 20;i++) {
        printf("%x",usb_read(0,fd));
        usb_write(0,i,fd);
        getch();
    }
    usb_close(fd);
}
MATLABでの使用例
初めの起動のとき、以下のメッセージが出る。

```
>> usbg(0,255)
VendorID=bfe
ProductID=1003
Version=1
Ready to use
ans =
1
```
ここでは、ポート0番地に255を書き込んでいる例である。2回目以降は、ベンダIDなどは出力されない。

```
>> usbg(0,8)
ans =
1
```
なお、USBコネクタを抜くときは、必ずclear usbgを実行してから抜くこと、そうしないとメモリリークが起こりシステムが不安定となる原因となる。