USB 共有デバイスと USB 単体デバイスの相違点

デスクリプタについて

【デバイスデスクリプタの構成】

USBのデスクリプタは下図のような構成になっています。USBフレームワークでは
これらを usbdsc.h と usbdsc.c で生成しています。
従って新しい USB デバイスを構成するときには、必ずこのデスクリプタを作成する必要
があります。以下の例題では、RS232C over USB のアプリの CDC クラスを例として説明していきま
す。

【デバイスデスクリプタ】

バイト#	フィールド名	内　　容
0	bLength	このディスクリプタのサイズ（0x12 の固定値）
1	bDescritorType	ディスクリプタの種別（0x01 の固定値）
2	bcdUSB	BCD 表現の USB バージョン(下位マイナー)
3	〃	〃　　(上位メジャー)
4	bDeviceClass	クラスコード　　(通常 0xFF) 0:クラスなし　0xFF:ベンダー　1～0xFE:特定
5	bDeviceSubClass	サブクラスコード　(自由　通常 0)
6	bDeviceProtocol	プロトコル指定 0:固有なし　　0xFF:ベンダ固有
7	bMaxPacketSizeo	エンドポイント 0 の最大パケットサイズ
8,9	idVendoor	ベンダ ID（２バイト下位、上位順）
10,11	idProduct	プロダクト ID（２バイト下位、上位順）
12	bcdDevice	BCD 表現のデバイスバージョン(下位)
13	〃	〃　　(上位)
14	iManufacturer	製造者のストリングへのインデックス番号
15	iProduct	製品のストリングへのインデックス番号
16	iSerialNumber	製造番号のストリングへのインデックス番号
17	bNumConfiguration	構成可能なデバイス数（通常１）

デバイスの特性を決める基本のデスクリプタで、下記の内容とします。
下記が CDC クラスの場合の例です。
ベンダーID、プロダクト ID を変更する場合には、ここで変更します。

またストリングを変更したときはインデックス番号にも注意が必要です。

```
/* Device Descriptor */
rom USB_DEV_DSC device_dsc=
{
    sizeof(USB_DEV_DSC),    // Size of this descriptor in bytes
    DSC_DEV,                // DEVICE descriptor type
    0x0200,                 // USB Spec Release Number in BCD format
    CDC_DEVICE,             // Class Code
    0x00,                   // Subclass code
    0x00,                   // Protocol code
    EP0_BUFF_SIZE,          // Max packet size for EP0, see usbcfg.h
    0x04D8,                 // Vendor ID
    0x000A,                 // Product ID: CDC RS-232 Emulation Demo
    0x0000,                 // Device release number in BCD format
    0x01,                   // Manufacturer string index
    0x02,                   // Product string index
    0x00,                   // Device serial number string index
    0x01                    // Number of possible configurations
};
```

IDの変更はこの2行

【コンフィギュレーション関連デスクリプタ】

コンフィギュレーションは最低1個必要で、デバイスの中に含まれる機能単位の
特性を指定します。このコンフィギュレーションに関係するデスクリプタには
インターフェースとエンドポイントがあります。

ますコンフィギュレーションデスクリプタの内容は下記とします。
ここでは電源の特性と最大電流の設定が必要です。インターフェースデスクリプタの内容は下記となります。

```
Private Sub Command1_Click()
'
'   汎用USBドライバ  USB接続制御
'
    hUSB = Uusbd_Open_mask(UU_MASK_VENDOR + UU_MASK_PRODUCT, 0, 0, Vendor, Product, 0)
    If hUSB = -1 Then
        MsgBox "USB接続ができません"
        End
    End If
    hCMD = Uusbd_OpenPipe(hUSB, 0, 0)
    If hCMD = -1 Then
        MsgBox "出力パイプ0を開けませんでした"
        End
    End If
    hSTA = Uusbd_OpenPipe(hUSB, 0, 1)
    If hSTA = -1 Then
        MsgBox "入力パイプ1を開けませんでした"
        End
    End If
    Text1.Text = "Ok!"
End Sub
```

エンドポイントデスクリプタの内容は下記となります。

No	フィールド名	内　　容
0	bLength	このディスクリプタのサイズ(0x07 の固定値)
1	bDescriptorType	ディスクリプタの種別(0x05 の固定値)
2	bEndPointAddress	１バイトで下記表現 Bit7：方向　0：OUT　　1：IN Bit6～4：予約 Bit3～0：エンドポイント番号
3	bmAttributes	このエンドポイントの属性 Bit1,0 00：コントロール転送 01：アイソクロナス転送 10：バルク転送 11：インタラプト転送 Bit5-2 はアイソクロナス転送の時のみ使用
4,5	wMaxPacketSize	最大パケットサイズ Bit10-0　バイナリ値(Max1024) Bit12,11 アイソクロナス転送で使用
6	bInterval	ホストからのポーリング周期(msec 単位) バルク、コントロール転送の時は無視される アイソクロナス転送の時は１を指定

実際のCDCクラスの上記コンフィギュレーション関連のデスクリプタは

```
/* Configuration 1 Descriptor */
CFG01=
{
    /* Configuration Descriptor */
    sizeof(USB_CFG_DSC),    // Size of this descriptor in bytes
    DSC_CFG,                // CONFIGURATION descriptor type
    sizeof(cfg01),          // Total length of data for this cfg
    2,                      // Number of interfaces in this cfg
    1,                      // Index value of this configuration
    0,                      // Configuration string index
    _DEFAULT,               // Attributes, see usbdefs_std_dsc.h
    50,                     // Max power consumption (2X mA)

    /* Interface Descriptor */
    sizeof(USB_INTF_DSC),   // Size of this descriptor in bytes
    DSC_INTF,               // INTERFACE descriptor type
    0,                      // Interface Number
    0,                      // Alternate Setting Number
    1,                      // Number of endpoints in this intf
    COMM_INTF,              // Class code
    ABSTRACT_CONTROL_MODEL, // Subclass code
    V25TER,                 // Protocol code
    0,                      // Interface string index

    /* CDC Class-Specific Descriptors */
    sizeof(USB_CDC_HEADER_FN_DSC),CS_INTERFACE,DSC_FN_HEADER,0x0110,
    sizeof(USB_CDC_ACM_FN_DSC),CS_INTERFACE,DSC_FN_ACM,0x02,
    sizeof(USB_CDC_UNION_FN_DSC),CS_INTERFACE,DSC_FN_UNION,CDC_COMM_INTF_ID,CDC_DATA_INTF_ID,
    sizeof(USB_CDC_CALL_MGT_FN_DSC),CS_INTERFACE,DSC_FN_CALL_MGT,0x00,CDC_DATA_INTF_ID,

    /* Endpoint Descriptor */
    sizeof(USB_EP_DSC),DSC_EP,_EP02_IN,_INT,CDC_INT_EP_SIZE,0x02,

    /* Interface Descriptor */
    sizeof(USB_INTF_DSC),   // Size of this descriptor in bytes
    DSC_INTF,               // INTERFACE descriptor type
    1,                      // Interface Number
    0,                      // Alternate Setting Number
    2,                      // Number of endpoints in this intf
    DATA_INTF,              // Class code
    0,                      // Subclass code
    NO_PROTOCOL,            // Protocol code
    0,                      // Interface string index

    /* Endpoint Descriptors */
    sizeof(USB_EP_DSC),DSC_EP,_EP03_OUT,_BULK,CDC_BULK_OUT_EP_SIZE,0x00,
    sizeof(USB_EP_DSC),DSC_EP,_EP03_IN,_BULK,CDC_BULK_IN_EP_SIZE,0x00
};
```

下記となっています。

【デスクリプタストリング】

いろいろな文字情報をホストに提供するためのデータです。

内容は下記とします。

注意が必要なことは、文字コードは UNICODE でなければならないということです。

```
Private Sub Timer1_Timer()
'計測完了問い合わせと測定データ収集
    Dim K As Integer
    Dim N As Integer

    Command(0) = &H32                   '完了問い合わせコマンド
    'コマンド送信後折り返し状態受信
    State = WriteFile(hCMD, Command(0), 1, Result, 0)
    If State = 0 Then                   'エラー発生時
        MsgBox "出力パイプ0の書き込みでエラーが発生"
        End
    End If
    State = ReadFile(hSTA, RcvData(0), 64, Result, 0)
    If State = 0 Then                   'エラー発生時
        MsgBox "入力パイプ1の読み出しでエラーが発生"
        End
    End If

      (以下　省略)
```

CDC の例では下記となっています。 上からインデックスが0，1，2の文字列と
なります。

```
rom struct{byte bLength;byte bDscType;word string[1];}sd000={
sizeof(sd000),DSC_STR,0x0409};

rom struct{byte bLength;byte bDscType;word string[25];}sd001={
sizeof(sd001),DSC_STR,
'M','i','c','r','o','c','h','i','p',' ',
'T','e','c','h','n','o','l','o','g','y',' ','I','n','c','.'};

rom struct{byte bLength;byte bDscType;word string[25];}sd002={
sizeof(sd002),DSC_STR,
'C','D','C',' ','R','S','-','2','3','2',' ',
'E','m','u','l','a','t','i','o','n',' ','D','e','m','o'};
```

参考 URL　　http://www.picfun.com/usb20frame.html

USB 共有デバイスと USB 単体デバイスの相違点

Usb_descripter.c　（USB 共有デバイスの中の方）

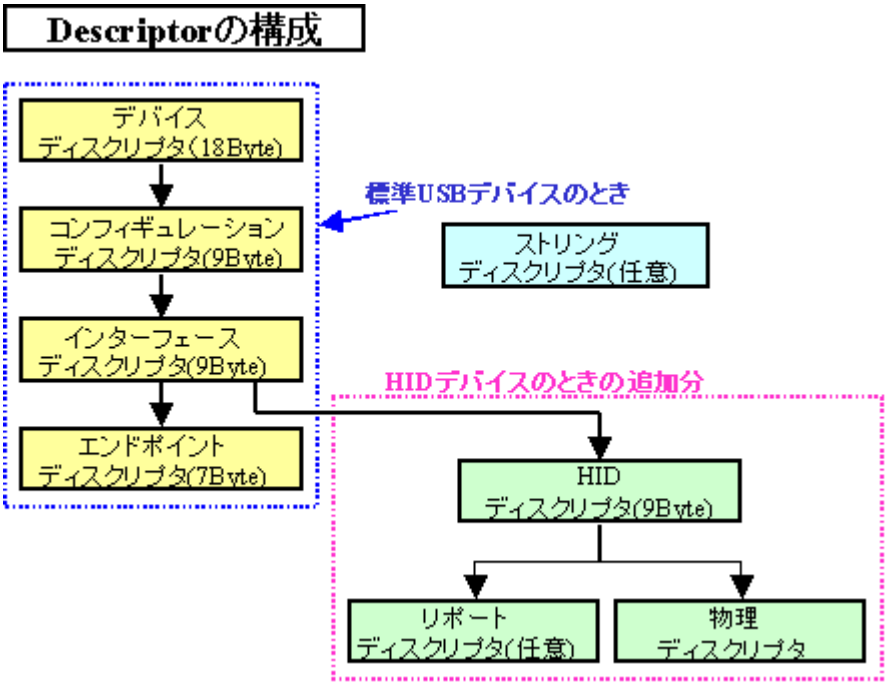
42 行目　　2,　　// Number of interfaces in this cfg

分かったこと

キーボードではエンドポイントが 2 つある

マウスではエンドポイントが 1 つである

まとめ



デバイスデスクリプタについて

デバイスの特性を決める基本のデスクリプタです

図の用に、1 つのマイコンに 2 つの機能（マウスとキーボードとか）を持たする時、まずデバイスデスクリプタは共有で 1 つそこで､1 つと 2 つのときの違いは基本的にないが、このとき使用するクラス（HID、,CDC など）ほ 1 つだけになる。つまり、1 のデバイスに rs232c 通信シュミレータとマウスを使うことは無理。逆に HID クラス同志のマウスとキーボードはいける。

コンフィギュレーションデスクリプタについて

コンフィギュレーションは最低 1 個必要で、デバイスの中に含まれる機能単位の特性を指定します。

コンフィギュレーションデスクリプタも共有で 1 つです。違いとしては

2,　//YTS　　// Number of interfaces in this cfg

のはじめの 1 と 2 の違うこれはインターフェイスの数（マウスとかキーボードの数）を表している。また

DESC_CONFIG_WORD(0x0042),　//YTS　// Total length of data for this cfg

の 0x0042 野値が違うがこれについては正確には分からない

インターフェイスデスクリプタについて

　インターフェイスデスクリプタはデバイスのインターフェイス（マウス、キーボード）の数だけある。ここでは違いはなくのデスクリプタがある。この中にはクラスのデスクリプタやエンドポイントがある

クラスデスクリプタ

　ここではっクラスのタイプなどを指定する

エンドポイントデスクリプタ

　DESC_CONFIG_WORD(3),　でユーザーが使うデータの長さを指定するこの場合だとマウスのデータで、3 バイトのデータ 1 バイト目にクリックのどのデータ 2，3 バイト目にマウスのカーソルの位置情報が入る。