How to Communicate with its USB Devices using HID Protocol

wqaxs36
24 Jul 2020 CPOL

This article will help you to understand how to communicate with the USB devices using WinAPI in C#.

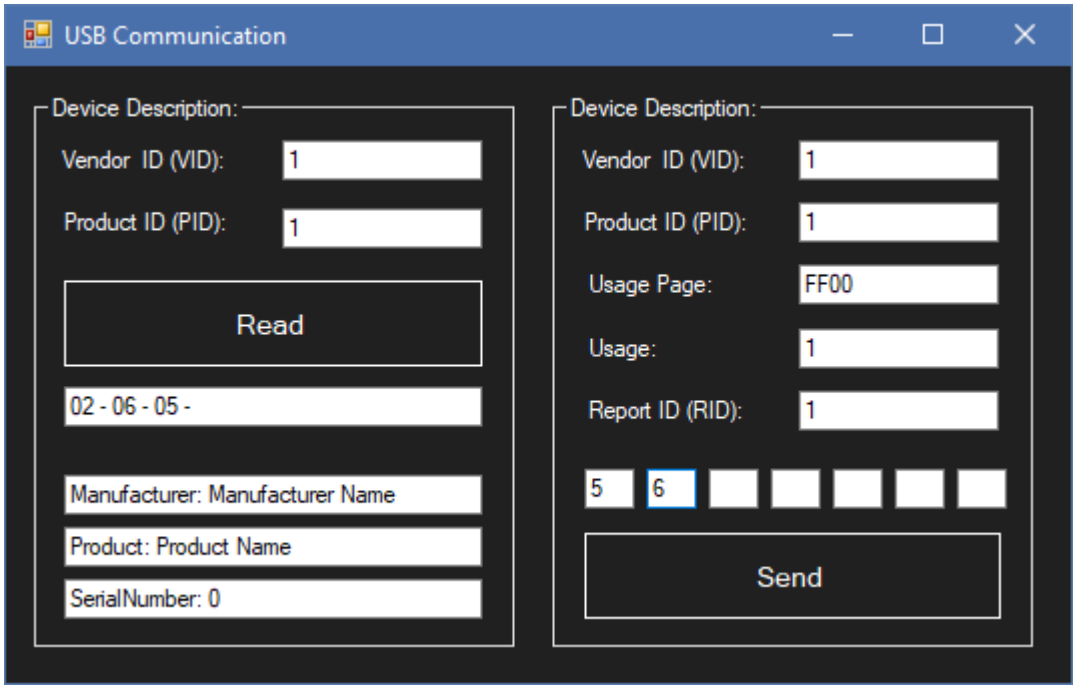
This article shows you how to use the USB/HID protocol under Windows to be able to send/receive USB packets from any USB devices connected to the PC. And without using DLL, just an application is needed.

Download demo - 19.4 KB

(Visual Studio 2022 project)

Warning

This USB sniffer, because of its user mode method access to hardware, cannot read HID packets with RID at 0, it's due to Windows protection level to prevent keyloggers/spying software.



(Do not add 0x, else the application will crash, I haven't added 0x prefix support)

Introduction

This article shows you how to use the USB/HID protocol under Windows to be able to send/receive USB packets from any USB devices connected to the PC.

And without using DLL, just an application is needed.

Background

This article was possible with this WDK sample:

> **https://github.com/Microsoft/Windows-driver-samples/tree/master/hid/hclient**

Basically, it's just a rewrite of this sample, but in a simple form.

Using the Code

Main code:

```
void    Update()
{
    while (true)
```

```
    {
        CheckHIDRead();
        CheckHIDWrite();

        Thread.Sleep(1);
    }
}
```

`CheckHIDRead()` and `CheckHIDWrite()` are checking if we have press Read or Start button and if entered data (VID-PID-Usa***) correspond to a connected USB Device.

This function returns the number of USB devices in order to scan them.

C# Shrink ▲

```
Int32   FindDeviceNumber()
{
    var hidGuid        = new Guid();
    var deviceInfoData = new SP_DEVICE_INTERFACE_DATA();

    HidD_GetHidGuid(ref hidGuid);

    //
    // Open a handle to the plug and play dev node.
    //
    SetupDiDestroyDeviceInfoList(hardwareDeviceInfo);
    hardwareDeviceInfo    = SetupDiGetClassDevs(ref hidGuid, IntPtr.Zero, IntPtr.Zero,
DIGCF_PRESENT | DIGCF_DEVICEINTERFACE);
    deviceInfoData.cbSize = Marshal.SizeOf(typeof(SP_DEVICE_INTERFACE_DATA));

    var Index = 0;
    while (SetupDiEnumDeviceInterfaces(hardwareDeviceInfo, IntPtr.Zero, ref hidGuid, Index,
ref deviceInfoData))
    {
        Index++;
    }

    return (Index);
}
```

This function returns a data structure of each USB device needed for `Read()` and `Write()`.

C# Shrink ▲

```
Int32   FindKnownHIDDevices(ref HID_DEVICE[] HID_Devices)
{
    var hidGuid                 = new Guid();
    var deviceInfoData          = new SP_DEVICE_INTERFACE_DATA();
    var functionClassDeviceData = new SP_DEVICE_INTERFACE_DETAIL_DATA();

    HidD_GetHidGuid(ref hidGuid);

    //
    // Open a handle to the plug and play dev node.
    //
```

```
    SetupDiDestroyDeviceInfoList(hardwareDeviceInfo);
    hardwareDeviceInfo    = SetupDiGetClassDevs(ref hidGuid, IntPtr.Zero, IntPtr.Zero,
DIGCF_PRESENT | DIGCF_DEVICEINTERFACE);
    deviceInfoData.cbSize = Marshal.SizeOf(typeof(SP_DEVICE_INTERFACE_DATA));

    var iHIDD = 0;
    while (SetupDiEnumDeviceInterfaces(hardwareDeviceInfo, IntPtr.Zero, ref hidGuid, iHIDD,
ref deviceInfoData))
    {
        var RequiredLength = 0;

        //
        // Allocate a function class device data structure to receive the
        // goods about this particular device.
        //
        SetupDiGetDeviceInterfaceDetail(hardwareDeviceInfo, ref deviceInfoData,
IntPtr.Zero, 0, ref RequiredLength, IntPtr.Zero);

        if (IntPtr.Size == 8)
        {
            functionClassDeviceData.cbSize = 8;
        }
        else if (IntPtr.Size == 4)
        {
            functionClassDeviceData.cbSize = 5;
        }

        //
        // Retrieve the information from Plug and Play.
        //
        SetupDiGetDeviceInterfaceDetail(hardwareDeviceInfo, ref deviceInfoData, ref
functionClassDeviceData, RequiredLength, ref RequiredLength, IntPtr.Zero);

        //
        // Open device with just generic query abilities to begin with
        //
        OpenHidDevice(functionClassDeviceData.DevicePath, ref HID_Devices, iHIDD);

        iHIDD++;
    }

    return iHIDD;
}
```

This function extend `FindKnownHIDDevices()`.

C# Shrink ▲

```
void    OpenHIDDevice(String DevicePath, ref HID_DEVICE[] HID_Device, Int32 iHIDD)
{
    //
    // RoutineDescription:
    // Given the HardwareDeviceInfo, representing a handle to the plug and
    // play information, and deviceInfoData, representing a specific hid device,
    // open that device and fill in all the relivant information in the given
    // HID_DEVICE structure.
```

```
    //
    HID_Device[iHIDD].DevicePath = DevicePath;

    //
    // The hid.dll api's do not pass the overlapped structure into deviceiocontrol
    // so to use them we must have a non overlapped device.  If the request is for
    // an overlapped device we will close the device below and get a handle to an
    // overlapped device
    //
    CloseHandle(HID_Device[iHIDD].Pointer);
    HID_Device[iHIDD].Pointer    = CreateFile(HID_Device[iHIDD].DevicePath, GENERIC_READ |
GENERIC_WRITE, FILE_SHARE_READ | FILE_SHARE_WRITE, 0, OPEN_EXISTING, 0, IntPtr.Zero);
    HID_Device[iHIDD].Caps       = new HIDP_CAPS();
    HID_Device[iHIDD].Attributes = new HIDD_ATTRIBUTES();

    //
    // If the device was not opened as overlapped, then fill in the rest of the
    // HID_Device structure.  However, if opened as overlapped, this handle cannot
    // be used in the calls to the HidD_ exported functions since each of these
    // functions does synchronous I/O.
    //
    HidD_FreePreparsedData(ref HID_Device[iHIDD].Ppd);
    HID_Device[iHIDD].Ppd = IntPtr.Zero;

    HidD_GetPreparsedData(HID_Device[iHIDD].Pointer, ref HID_Device[iHIDD].Ppd);
    HidD_GetAttributes(HID_Device[iHIDD].Pointer, ref HID_Device[iHIDD].Attributes);
    HidP_GetCaps(HID_Device[iHIDD].Ppd, ref HID_Device[iHIDD].Caps);

    var Buffer = Marshal.AllocHGlobal(126);
    {
        if (HidD_GetManufacturerString(HID_Device[iHIDD].Pointer, Buffer, 126))
        {
            HID_Device[iHIDD].Manufacturer = Marshal.PtrToStringAuto(Buffer);
        }
        if (HidD_GetProductString(HID_Device[iHIDD].Pointer, Buffer, 126))
        {
            HID_Device[iHIDD].Product = Marshal.PtrToStringAuto(Buffer);
        }
        if (HidD_GetSerialNumberString(HID_Device[iHIDD].Pointer, Buffer, 126))
        {
            Int32.TryParse(Marshal.PtrToStringAuto(Buffer), out
HID_Device[iHIDD].SerialNumber);
        }
    }
    Marshal.FreeHGlobal(Buffer);

    //
    // At this point the client has a choice.  It may chose to look at the
    // Usage and Page of the top level collection found in the HIDP_CAPS
    // structure.  In this way --------*it could just use the usages it knows about.
    // If either HidP_GetUsages or HidP_GetUsageValue return an error then
    // that particular usage does not exist in the report.
    // This is most likely the preferred method as the application can only
    // use usages of which it already knows.
    // In this case the app need not even call GetButtonCaps or GetValueCaps.
    //
    // In this example, however, wSendHID_PIDe will call FillDeviceInfo to look for all
```

```
    //    of the usages in the device.
    //
    //FillDeviceInfo(ref HID_Device, iHIDD);
}
```

Then come the two important functions that will make you able to read or write USB packets between a USB device and a PC.

C# Shrink ▲

```
void    HIDRead(HID_DEVICE HID_Device)
{
    ManufacturerName.Text = "Manufacturer: " + HID_Device.Manufacturer;
    ProductName.Text      = "Product: "      + HID_Device.Product;
    SerialNumber.Text     = "SerialNumber: " + HID_Device.SerialNumber.ToString();

    //
    // Read what the USB device has sent to the PC and store the result into HID_Report[]
    //
    var HID_Report = new Byte[HID_Device.Caps.InputReportByteLength];

    if (HID_Report.Length > 0)
    {
        var varA = 0U;
        ReadFile(HID_Device.Pointer, HID_Report, HID_Device.Caps.InputReportByteLength, ref
varA, IntPtr.Zero);

        Read_Output.Clear();

        for (var Index = 0; Index < HID_Device.Caps.InputReportByteLength; Index++)
        {
            Read_Output.Text += HID_Report[Index].ToString("X2");
            Read_Output.Text += " - ";
        }
    }
}
void    HIDWrite(HID_DEVICE HID_Device)
{
    //
    // Sent to the USB device what is stored in WriteData[]
    //
    var HID_Report = new Byte[HID_Device.Caps.OutputReportByteLength];

    if (HID_Report.Length > 0)
    {
        HID_Report[0] = HIDWriteData.ReportID;

        for (var Index = 0; Index < WriteData.Length; Index++)
        {
            if (Index + 1 < HID_Report.Length)
            {
                // Start at 1, as the first byte must be zero for HID report
                HID_Report[Index + 1] = WriteData[Index];
            }
        }
```

```
        var varA = 0U;
        WriteFile(HID_Device.Pointer, HID_Report, HID_Device.Caps.OutputReportByteLength,
ref varA, IntPtr.Zero);
    }
}
```

To be able to do that, you'd need to set following data before:

- `VendorID`
- `ProductID`
- `UsagePage`
- `Usage`
- `ReportID`

But be careful, you'd need to set the correct values for all those parameters, if one is `false`, you will not be able to send HID packets.

To read HID packets, you just need:

- `VendorID`
- `ProductID`

Also, you cannot read if the device cannot send data and you cannot write if the device cannot read data (defined by the HID Report Descriptor).

A device is defined by its `VendorID`:`ProductID` but shrunk into several functions defined by its `UsagePage`, `Usage` and `ReportID`.

As an example, the first function of a mouse is to send coordinate data, so you can read data from PC and the second function is to receive mouse button customization data, so you can send data from PC.

And to set those variables, you need to read the HID Descriptor of the USB devices that you target, it can be retrieved with a USB sniffer as https://github.com/djpnewton/busdog or http://www.usblyzer.com/usb-analysis-features.htm

The HID Descriptor usually start with 0x05, 0x01.

And to learn to read HID Descriptor, use this tool: http://www.usb.org/developers/hidpage#HID Descriptor Tool

Because this code is just a rewrite of an old C code from the 90s, it works on all Windows Versions.

History

- 2nd October, 2019: Initial version
- 5nd January, 2022: Improved version

License

Written By

wqaxs36

France

This member has not yet provided a Biography. Assume it's interesting and varied, and probably something to do with programming.

Watch

Comments and Discussions

Add a Comment or Question

Email Alerts

Search Comments

First Prev Next

Exception in read to run from VS IDE, but it's ok to run from command line
Felix_Shih 19-Jul-23 14:52

Reading incoming data
Member 7672236 13-Feb-23 22:01

Re: Reading incoming data
wqaxs36 14-Feb-23 0:37

Re: Reading incoming data
Member 15919362 14-Feb-23 5:13

Re: Reading incoming data
wqaxs36 14-Feb-23 14:11

Re: Reading incoming data
Member 7672236 14-Feb-23 21:40

Send data to microcontroller

Eduard Bumbu **12-Mar-22 15:09**

Re: Send data to microcontroller
wqaxs36 12-Mar-22 22:55

Re: Send data to microcontroller
don_ucw 13-May-22 20:56

Re: Send data to microcontroller
wqaxs36 13-May-22 23:45

what is Report ID (RID)
AliAhmadSabir **6-Jun-21 20:54**

Re: what is Report ID (RID)
wqaxs36 6-Jun-21 22:35

It does not work with HID barcode reader
Member 14635039 **5-Oct-20 11:33**

Re: It does not work with HID barcode reader
wqaxs36 5-Oct-20 22:25

Great tutorial about USB HID
Potter68 **31-Aug-20 0:24**

Controlling serial and keyboard emulated devices
uzayim **28-Jul-20 5:24**

Re: Controlling serial and keyboard emulated devices
wqaxs36 28-Jul-20 15:48

Thank you for this +5
honey the codewitch **24-Jul-20 14:12**

I think that I'm running into "cannot read HID packets with RID at 0"... can you please elaborate...?
crn114 **8-Jun-20 0:34**

Re: I think that I'm running into "cannot read HID packets with RID at 0"... can you please elaborate...?
wqaxs36 15-Jun-20 15:34

Re: I think that I'm running into "cannot read HID packets with RID at 0"... can you please elaborate...?
wqaxs36 15-Jun-20 15:52

I have a VID and a PID, I enter it and hit submit to read... only see zeros in window... what am I doing wrong?

crn114 **6-Jun-20 20:00**

Re: I have a VID and a PID, I enter it and hit submit to read... only see zeros in window... what am I doing wrong?
wqaxs36 15-Jun-20 15:29

Comminucate with KVM
vjaggi **12-May-20 22:33**

Re: Comminucate with KVM
wqaxs36 15-May-20 10:36

Refresh

1 2 3 Next ▷

General News Suggestion Question Bug Answer Joke Praise Rant Admin

Use Ctrl+Left/Right to switch messages, Ctrl+Up/Down to switch threads, Ctrl+Shift+Left/Right to switch pages.

Permalink
Advertise
Privacy
Cookies
Terms of Use

Layout: fixed | fluid

Web03 2.8:2023-08-14:1