Overview

The USB HID generic application is a simple demonstration program that uses the KSDK software. It is enumerated as a HID-compliant device and users can use PC's application to exchange data with the device.

System Requirement

Hardware requirements

- J-Link ARM
- P&E Micro Multi-link universal
- Mini/micro USB cable
- USB A to micro AB cable
- Hardware (tower/base board, ...) for specific device
- Personal Computer(PC)

Software requirements

- The project files, for lite version example, the path is: <SDK_Install>/boards/<board>/usb/usb_device_hid_generic_lite/<RTOS>/<toolchain>. For non-lite version example, the path is: <SDK_Install>/boards/<board>/usb/usb_device_hid_generic/<RTOS>/<toolchain>.

> **Note**
>
> The RTOS is BM or FreeRTOS.

- The PC's test tool. This tool is generated based on Wimar's USB HID Component for C#, the source code link is: http://www.codeproject.com/Articles/18099/A-USB-HID-Component-for-C Due to support the USB HID generic device, there are some changes in file "Sniffer.cs".

1. Function `usb_OnSpecifiedDeviceArrived`

   This function is changed to :

```
private void usb_OnSpecifiedDeviceArrived(object sender, EventArgs e)
{
    this.lb_message.Items.Add("My device was found");
    this.tb_send.Text = "12 34 56 78 90 ab cd ef";
}
```

2. Function `btn_send_Click`

   This function is changed to :

```
private void btn_send_Click(object sender, EventArgs e)
{
    try
    {
        string text = this.tb_send.Text + " ";
        int length = 0;
        text = new System.Text.RegularExpressions.Regex("[\\s]+").Replace(text, " ");
        text.Trim();
        string[] arrText = text.Split(' ');
        byte[] data = new byte[arrText.Length];
        for (int i = 0; i < (arrText.Length); i++)
        {
            if (arrText[i] != "")
            {
                int value = Int32.Parse(arrText[i],
       System.Globalization.NumberStyles.AllowHexSpecifier);
                data[i] = (byte)Convert.ToByte(value);
                length++;
            }
        }

        for (int i = 0;i < length;i = i+this.usb.SpecifiedDevice.OutputReportLength-1)
        {
            byte[] send_buffer = new byte[this.usb.SpecifiedDevice.OutputReportLength];
            send_buffer[0] = (byte)0;
            for(int j = 1; (j < (data.Length - i + 1)) && (j <
       this.usb.SpecifiedDevice.OutputReportLength);j++)
            {
                send_buffer[j] = data[j + i - 1];
            }
            if (this.usb.SpecifiedDevice != null)
            {
                this.usb.SpecifiedDevice.SendData(send_buffer);
            }
            else
            {
                MessageBox.Show("Sorry but your device is not present. Plug it in!! ");
            }
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.ToString());
    }
}
```

3. Function usb_OnDataRecieved

   This function is changed to :

```
private void usb_OnDataRecieved(object sender, DataRecievedEventArgs args)
{
    if (InvokeRequired)
    {
        try
        {
            Invoke(new DataRecievedEventHandler(usb_OnDataRecieved), new object[] {
       sender, args });
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.ToString());
        }
    }
    else
    {
        string rec_data = "Recv:";
        int is_report_id = 1;
        foreach (byte myData in args.data)
        {
            if (is_report_id > 0)
            {
                is_report_id = 0;
            }
            else
            {
                rec_data += myData.ToString("X2") + " ";
            }
        }
        this.lb_read.Items.Insert(0, rec_data);
    }
}
```

Note

This tool is only tested in Microsoft Windows 7 Enterprise X64. For others Operation System, please download the source code, apply the changes and rebuild the test tool projects.

Getting Started

Hardware Settings

- The Jumper settings:
  JP12 connected .

Prepare the example

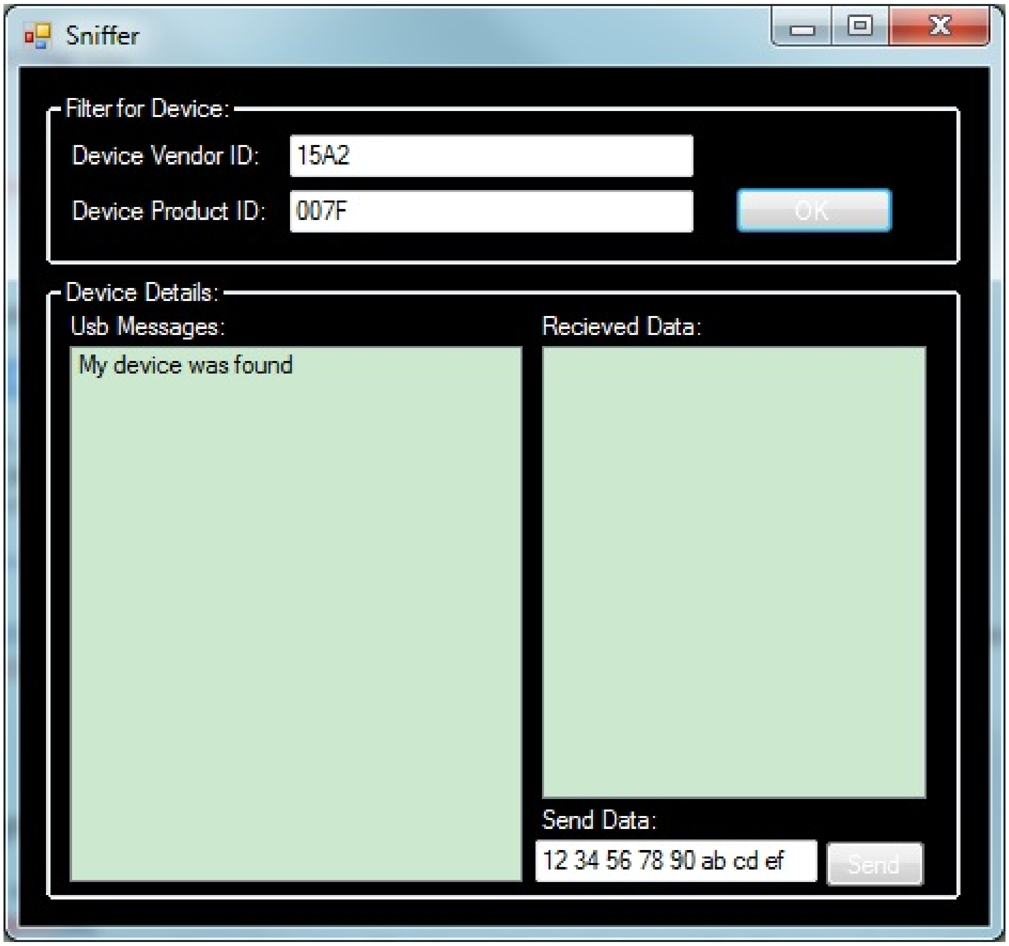
1. Download the program to the target board.
2. Power off the target board. And then power on again.
3. Connect a USB cable between the PC and the USB device port of the board.

Note

For detailed instructions, see the appropriate board User's Guide.

Run the example

1. Plug-in the device which is running HID generic example into PC. You will see a HID-compliant device enumerated in Device Manager.
2. Open the test tool. Set the Vendr ID(VID) to "15A2", set Product ID(PID) to "007F". And then click the butten "OK".
   If the device is found and opened, the message "My device was found" will be showed in textbox "USB Messages".



Device Vendor ID: 15A2

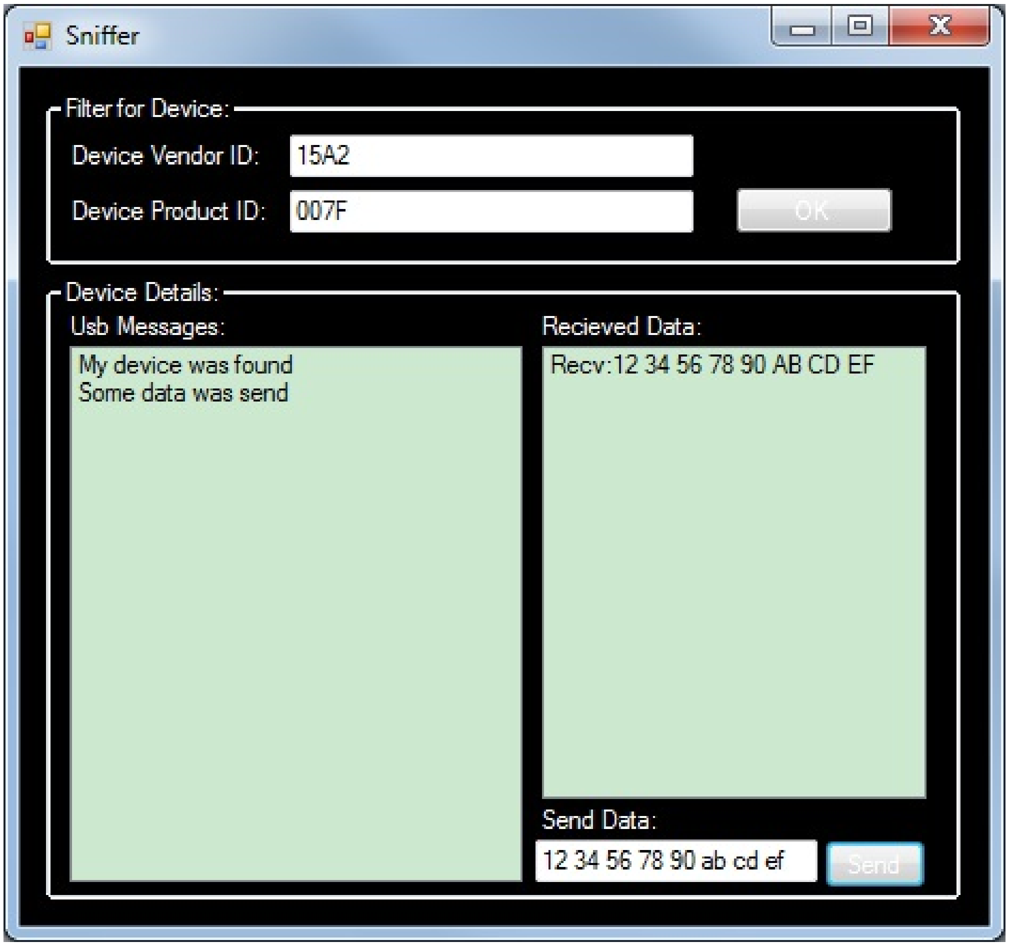
Device Product ID: 007F

My device was found

Send Data: 12 34 56 78 90 ab cd ef

> **Note**
>
> The VID and PID is Hex format.

3. Enter the data to the textbox "Send data", and then click the butten "Send". If data is sent, the message "Some data was send" will be showed in textbox "USB Messages". And then the device will send back the data. When the data is received, the received data will be showed in textbox "receive data".
   For example, data "12 34 56 78 90 ab cd ef" is sent to the device, then the message "Recv:12 34 56 78 90 ab cd ef" will be showed when the test tool receives the data.



Filter for Device:

Device Vendor ID: 15A2

Device Product ID: 007F

Device Details:

Usb Messages:

My device was found
Some data was send

Recieved Data:

Recv:12 34 56 78 90 AB CD EF

Send Data: 12 34 56 78 90 ab cd ef

Note

The data is Hex format.