# USB Development Kit (UsbDk) Software Development Manual

Revision 1.1

03/25/2015

# Table of Contents

# Introduction

USB Development Kit is a set of software modules meant to provide Windows user mode applications with direct and exclusive access to USB devices. Basic principle of UsbDk operation is detaching given USB device from Windows PNP manager and device drivers stack and providing user mode applications with API for USB-specific operations on the device.

Some distinctive UsbDk properties are:

- Device capture process is totally dynamic, i.e. no INF files and no self-signing needed, any device can be captured.
- UsbDk co-exists with original device driver, when the device is not captured original driver is loaded by the system automatically.
- If user mode client terminates unexpectedly for any reason system reverts to original device driver immediately.
- UsbDk supports all types of devices and interfaces - bulk, isochronous, composite, HID etc.
- Being USB filter driver UsbDk doesn't require WHQL-ing as per Microsoft requirements.

UsbDk supports all Windows versions starting from XP, i.e. XP, Vista, 7, 8, 8.1, 2003, 2008, 2008R2, 2012, 2012R2, both 32 and 64 bits architectures and as far as we concerned should support future versions of Windows without architectural changes.

# UsbDk Software Modules

USB Development Kit software consists of following modules:

- **UsbDk.sys/UsbDk.inf** – UsbDk kernel mode engine.
- **UsbDkHelper.dll** - Helper DLL that provides user mode API for system deployment and usage
- **UsbDkController.exe** - Command line application for system API usage demonstration and verification.

# UsbDk Build Procedure

UsbDk build procedure requires **Visual Studio 2013 Update 4**, **WDK 8.1** and **WinDDK 7.1**. MSI packages creation requires **WiX Toolset 3.8**.

All tools needed to build are free and may be downloaded from the Internet. Following table summarizes corresponding download locations.

| | |
|---|---|
| **Visual Studio 2013** | http://www.visualstudio.com/downloads/download-visual-studio-vs |
| **WDK 8.1** | https://msdn.microsoft.com/en-us/windows/hardware/gg454513.aspx |
| **WDK 7.1** | http://www.microsoft.com/en-us/download/details.aspx?id=11800 |
| **Wix Toolset 3.8** | https://wix.codeplex.com/releases/view/115492 |

It is important to note that WDK 8.1 must be installed after Visual studio in order to function properly. Besides that current build system assumes that all components installed at their default locations on disk C:.

Having all components properly installed UsbDk driver and user mode components may be build from Visual Studio using solution file from root of the source tree.

There is also a batch build script **buildAll.bat** that compiles all UsbDk components for all supported operating systems and creates all MSI packages.

# Installation

## Installation from MSI package

UsbDk binaries are supplied to end user in form of MSI package. There are separate MSI packages for Intel x86 and AMD x64 architectures. Install process follows standard MSI practices, i.e. binaries put to target system by MSI installation and deleted by un-install initiated via control panel's uninstall program applet.

## Installation with UsbDkController

While MSI packages are good for end user developers frequently need more flexible way for package installation. Required functionality is provided by UsbDkController application that is part of UsbDk source code base and compiled together with other UsbDk components.

UsbDk binaries may be installed by running "`UsbDkController.exe -i`" from folder that contains output of UsbDk build system. In some cases installation may require system reboot, in that case UsbDkController will print corresponding message to the console.

Similarly uninstallation of UsbDk is done by running "`UsbDkController.exe -u`" command. Uninstallation of UsbDk never requires system reboot.

Never mix MSI and UsbDkController installations because they are incompatible. MSI package will fail to install if there is previous UsbDk version installed by UsbDkController. Version installed by UsbDkController must be uninstalled by UsbDkController and vise versa.

# Diagnostic Tools

## UsbDkController

**UsbDkController.exe** is a simple command line utility for UsbDk diagnostics and API usage demonstration. **UsbDkController.exe** is a part of UsbDk source tree and compiled by UsbDk build procedure.

**UsbDkController.exe** command line options:

| | |
|---|---|
| ***UsbDkController.exe –h*** | - show this help screen and exit |
| ***UsbDkController.exe –i*** | - install UsbDk driver |
| ***UsbDkController.exe –u*** | - uninstall UsbDk driver |
| ***UsbDkController.exe –n*** | - enumerate USB devices |
| ***UsbDkController.exe -r ID SN*** | - redirect device by ID and serial number |

**Hider API:**

| | |
|---|---|
| ***UsbDkController -H VID PID BCD Class Hide*** | - add dynamic hide rule |
| ***UsbDkController -P VID PID BCD Class Hide*** | - add persistent hide rule |
| ***UsbDkController -D VID PID BCD Class Hide*** | - delete persistent hide rule |
| ***<VID PID BCD Class>*** | May be specific value or -1 to match all |
| ***<Hide>*** | Should be 0 or 1, if 0, the rule is terminal |

## Tracing Scripts

UsbDk driver uses Windows WPP facilities for execution tracing. UsbDk user mode components use OutputDebugString Win32 API for the same purpose. All UsbDk traces may be viewed by DbgView tool.

There is a script in UsbDk source tree (Tools\Trace\UsbDkTrace.bat) that configures WPP tracing engine and enables output of driver's WPP traces to DbgView.

# Accessing UsbDk from user mode applications

## User Mode API

UsbDk provides C user mode API exported by UsbDkHelper.dll.
API is defined in 4 files summarized in the table below.

| | |
|---|---|
| UsbDkHelper.h | Installation and device access functions declaration |
| UsbDkData.h | Installation and device access related data structures |
| UsbDkHelperHider.h | Device hiding functions declaration |
| UsbDkDataHider.h | Device hiding related data structures |

See corresponding headers in UsbDk source tree for full API documentation.

### Code samples

UsbDkController.exe application is a primary source of UsbDk code samples. It is a part of UsbDk source tree. Another good source of sample code is libusb backend for UsbDk, see libusb-devel archives for corresponding patches.

## System architecture and operation

### Overall System Architecture

UsbDk.sys is both USB filter driver and generic USB device driver. On installation it is being registered as USB filter driver and system invokes it for each new USB device being discovered including USB hubs. On invocation UsbDk.sys checks type of underlying device and creates filter instances for USB hubs only.

Being a filter of USB hub UsbDk.sys receives all requests from upper part of USB stack including enumeration requests that originated by PNP manager (IRP_MJ_PNP/IRP_MN_QUERY_DEVICE_RELATIONS).

Upon enumeration request completion by USB hub driver UsbDk.sys scans array of child devices returned and in case there are devices to be redirected (according to current configuration) it attaches as filter to those devices as well.

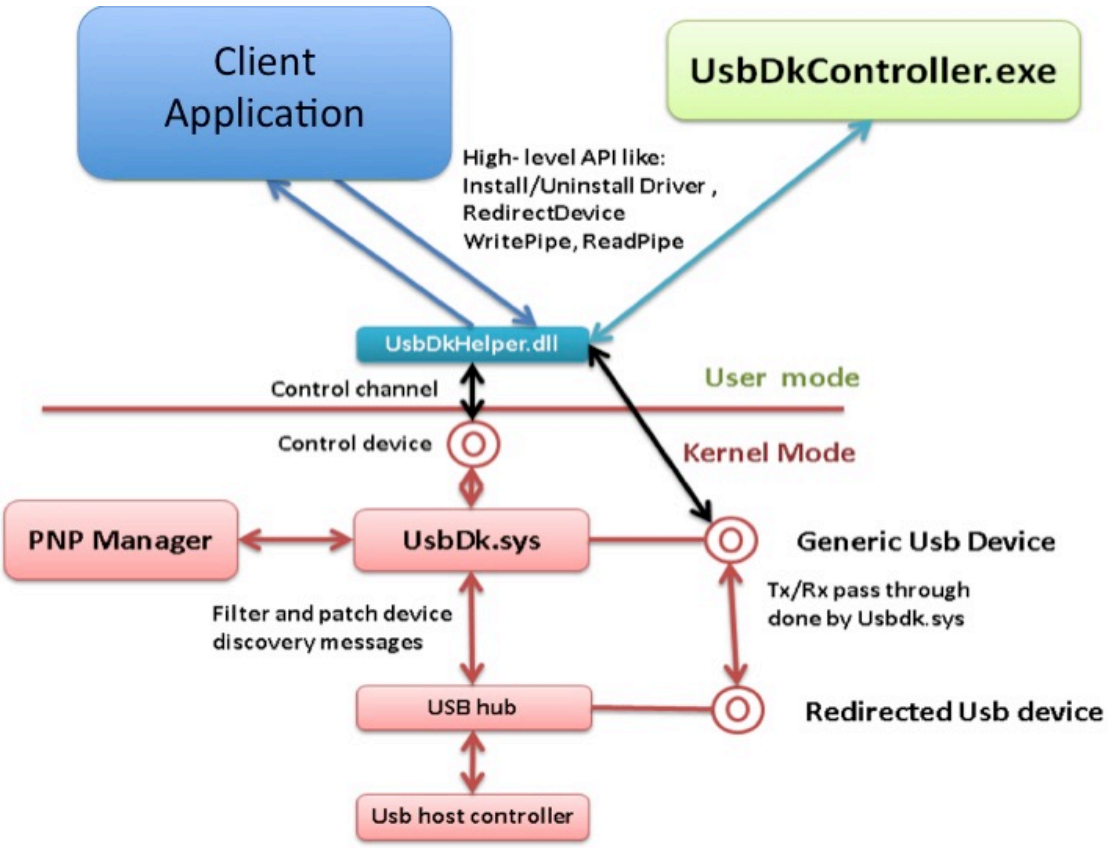
As a result all PNP manager requests pass via UsbDk.sys callbacks and the latter patches device ID properties as needed to make PNP manager recognize the device as a generic USB device. Besides that UsbDk.sys marks underlying device object as raw PDO so the system assigns the driver who created it (UsbDk.sys) to be the device driver as well.

At this stage any request to this vendor specific device will be forwarded to UsbDk.sys which in turn will patch and pass requests to/from original device created by USB hub as needed.

There is also a separate API provided by UsbDkHelper.dll intended to detach USB device from USB stack without acquiring actual access to the device (hider API). This API may be useful for security applications that need to deny access to USB devices according to security policy or to prevent windows from showing “New Hardware” pop-ups for devices solely managed by UsbDk. When UsbDk.sys discovers device to be hidden during enumeration phase (IRP_MJ_PNP/IRP_MN_QUERY_DEVICE_RELATIONS) it wipes corresponding PDO from device array returned by underlying USB bus driver, this effectively hides given device from PNP manager and OS driver stack.

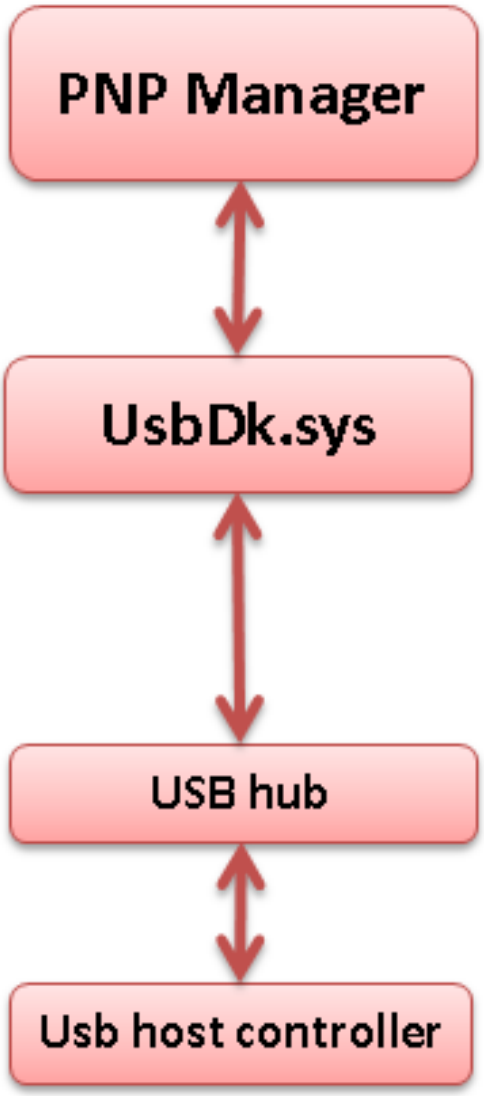
UsbDk.sys configuration originates from UsbDkHelper.dll and passed via internal device IO controls (IOCTLs).

Following picture provides high-level architecture overview of UsbDk and its clients.


Client Application
UsbDkController.exe
High- level API like:
Install/Uninstall Driver ,
RedirectDevice
WritePipe, ReadPipe
UsbDkHelper.dll
User mode
Control channel
Control device
Kernel Mode
PNP Manager
UsbDk.sys
Generic Usb Device
Tx/Rx pass through
done by Usbdk.sys
Filter and patch device
discovery messages
USB hub
Redirected Usb device
Usb host controller

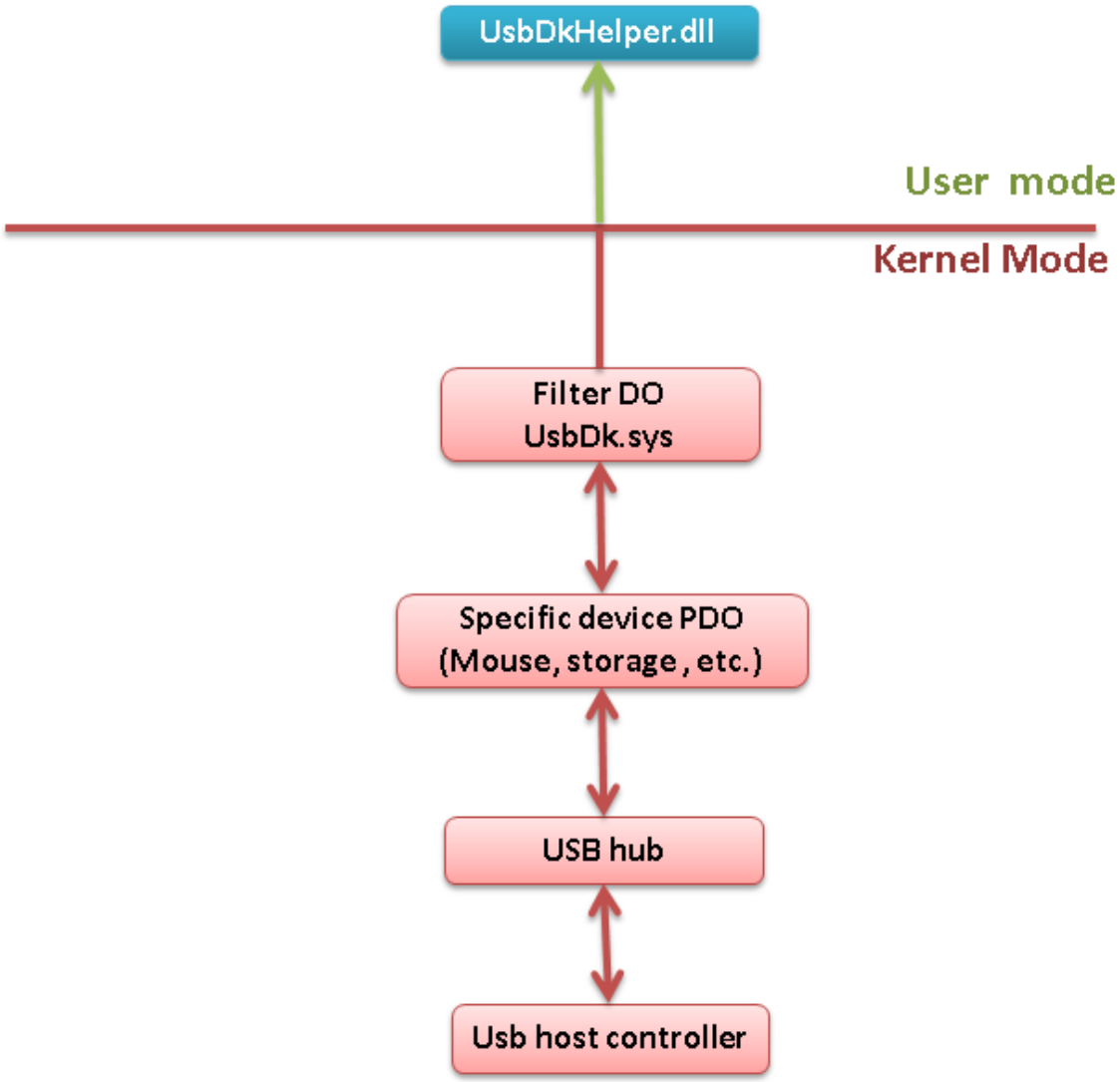
## Filtering models

### USB Hubs Filtering Model



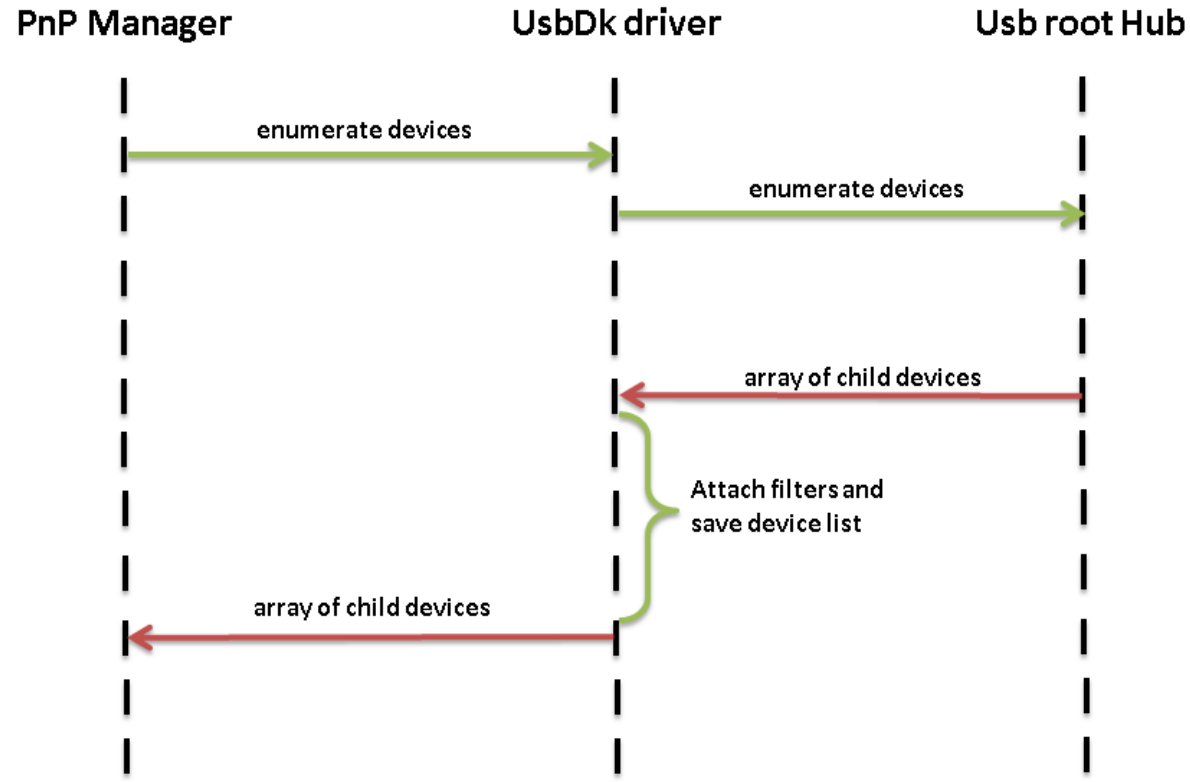

As filter driver of USB root host

Captured Device Filtering Model

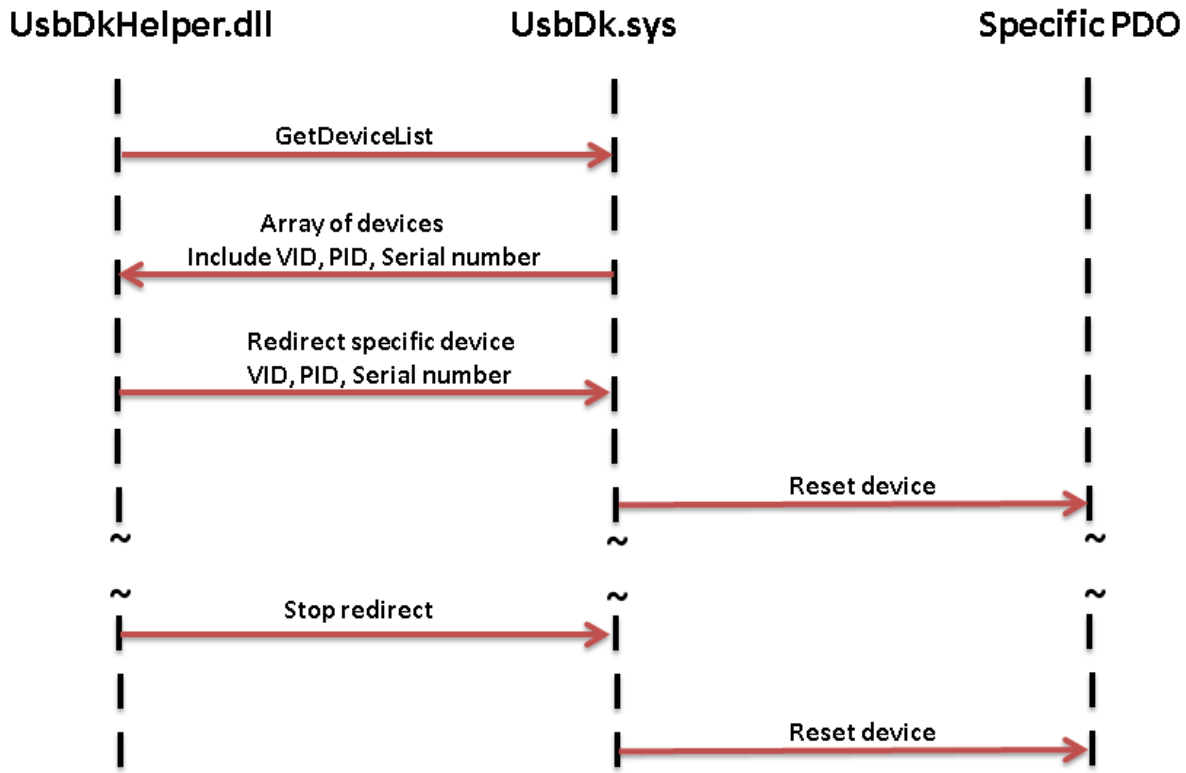


## Operation flows

### Filtering QUERY_DEVICE_RELATIONS for USB Hubs



## Acquisition of Exclusive Device Access



QUERY_DEVICE_RELATIONS flow with redirected device