Microsoft OS 2.0 Descriptors Specification

May 19, 2013

Abstract

This document defines and describes the implementation of version 2.0 of the Microsoft OS Descriptors. The goal of Microsoft OS 2.0 Descriptors is to address the limitations and reliability problems with version 1.0 of OS descriptors and enable new Windows-specific functionality for USB devices.

This information applies to the following operating systems:
Windows 8.1 Preview

References and resources discussed here are listed at the end of this paper.

The current version of this paper is maintained on the Web at:
http://go.microsoft.com/fwlink/?LinkId=306681

Microsoft

Overview

Microsoft OS descriptors were originally defined to allow USB devices to return Windows-specific device properties that are not available to Windows through standard USB descriptors. The MS OS descriptors have enabled Windows to:

- Support USB device classes, which are not defined by the USB-IF,
- Support USB device containers through container IDs, and
- Allow devices to define registry properties for USB devices or composite devices.

While Microsoft OS 1.0 descriptors has enabled important USB scenarios for Windows and are considered an important Microsoft intellectual property, the implementation has been shown to have some weaknesses and limitations. These issues include exposing fragile USB devices, limitations in how MS OS descriptors can be scoped, and problems in defining new descriptors.

Microsoft OS 2.0 descriptors addresses those issues.

Use Standard BOS Descriptor

To determine whether a USB device supports Microsoft OS 1.0 descriptors, Windows issues a request for a USB string descriptor at the arbitrarily defined index of 0xEE. Some USB devices do not handle that unexpected string query correctly, causing them to hang and/or fail enumeration. To avoid such errors, Windows maintains an errata list of devices known to break when querying for the Microsoft OS string descriptor, and caches the response during initial enumeration of a device so that the query is not repeated on subsequent enumerations.

To avoid the issue with the MS OS 2.0 descriptors, a new BOS device capability descriptor has been defined to allow devices to return platform-specific properties. The BOS descriptor is a standard descriptor defined by the standard USB specification for USB versions 2.1 and greater, so its retrieval is a normal and expected enumeration step which should not lead to unintended device behavior.

Scoping of MS OS descriptors

With MS OS version 1.0 descriptors, Windows USB driver stack does not query for any MS OS descriptors if the device is a composite device, instead defers such queries to the USB Generic Parent Driver (Usbccgp.sys). The effect is that all MS OS descriptors are applied to specific composite functions, and none can be applied to the entire device itself.

To address that issue, in MS OS 2.0 descriptors allows MS OS descriptors to be scoped to an entire device, a specific configuration, or a function.

Window Version Specific Properties

In some cases a device may want to be enumerated differently depending on the version of Windows on the system to which it is attached. MS OS 2.0 descriptors allow the device to return multiple descriptor sets where each set applies to a specific range of Windows version.

Microsoft OS descriptor version 2.0 architecture

The Microsoft OS 2.0 descriptor set must be returned as a single set of descriptors, similar to how the USB configuration descriptor and BOS descriptor sets are returned.

Microsoft OS 2.0 descriptor set organization

The Microsoft OS 2.0 descriptor set is organized as nested groups of descriptor subsets that can describe properties specific to a particular scope of the subset they are contain in. Scopes defined for the MS OS 2.0 descriptor set are device, configuration, or function. The hierarchy of descriptor organization within a descriptor set is as follows:

wopDDFB.tmpMicrosoft_Visio_2003-2010_Drawing1.vsd

Figure - Descriptor Subset Hierarchy

At its highest level the descriptor set contains a Microsoft OS 2.0 descriptor set header and zero or more Microsoft OS 2.0 feature descriptors that apply to the whole device regardless of its configuration, and zero or more configuration subsets that apply to specific USB configurations. There must be at least one feature descriptor or configuration subset descriptor defined in the top level Microsoft OS 2.0 descriptor set.

wopEEC5.tmpMicrosoft_Visio_2003-2010_Drawing2.vsd

Figure - Microsoft OS 2.0 descriptor set organization

Configuration subset organization

Each configuration subset consists of zero or more MS OS 2.0 feature descriptors that apply to a specific USB device configuration and zero or more function subsets that apply to specific functions within a device. A function is defined as a group of one or more interfaces. There must be at least one MS OS 2.0 feature descriptor or function subset defined in each configuration subset.

wopEEE5.tmpMicrosoft_Visio_2003-2010_Drawing3.vsd

Figure - Configuration Subset Organization

Configuration subsets are only used for composite devices that load USB Generic Parent Driver (Usbccgp.sys) as the parent driver for the entire device. If a device defines **OriginalConfigurationValue** and **AltlConfigurationValue** registry values for selecting non-default configurations, the configuration subset allows the device to define different properties for each function within a particular non-default configuration. For the default configuration the device can specify a top-level function subset.

Function subset organization

Each function subset consists of one or more MS OS 2.0 feature descriptors that apply to a specific USB function within a configuration. A function is defined as a group of one or more USB interfaces. Function subsets are only used for composite devices, or single-function devices that use Usbccgp.sys as their client driver.

wopEF15.tmpMicrosoft_Visio_Drawing1.vsdx

Figure - Function Subset Organization

Example Layouts

Non-composite device with single feature descriptor

The simplest MS OS 2.0 descriptor set has MS OS 2.0 descriptor and a MS OS 2.0 feature descriptor. This feature descriptor is applied to the whole device for all configurations.

wopEF36.tmpMicrosoft_Visio_2003-2010_Drawing4.vsd

Figure - Non-Composite MS OS 2.0 Descriptor Set Layout

Composite device with device and function level feature descriptors

In a scenario where some properties must be applied at the device level and others at the function level .This example applies properties to the device level for all configurations and to configuration 1, composite functions 0 and 3. Function numbers are determined by the first (lowest) **bInterfaceNumber** of the interface(s) assigned to the function.

wopEF56.tmpMicrosoft_Visio_2003-2010_Drawing5.vsd

Figure - Composite Device MS OS 2.0 Descriptor Set Layout

Multiple configurations

In some cases a device that supports multiple configurations may want to apply different properties depending on the selected configuration. This example layout shows MS OS 2.0 feature descriptors applied to a multi-configuration device with a non-composite configuration 1 and composite Configuration 2.

wopEF76.tmpMicrosoft_Visio_Drawing2.vsdx

Figure - Multiple Configuration MS OS 2.0 Descriptor Layout

Platform capability BOS descriptor

A new device capability type and descriptor must be defined for the USB BOS descriptor to return platform-specific device capabilities.

Platform device capability type

Table . Platform capability BOS descriptor

Capability Code	Value	Description
PLATFORM	05H	Defines a device capability specific to a particular platform/operating system.

Platform device capability descriptor

The platform device capability descriptor contains a 128-bit UUID value that is defined and published by the platform/operating system vendor, and is used to identify a unique platform-specific device capability. This descriptor may also contain one or more bytes of data associated with the capability.

Table . Platform device capability descriptor

Offset	Field	Size	Description
0	**bLength**	1	The length, in bytes, of this descriptor.
1	**bDescriptorType**	1	DEVICE CAPABILITY Descriptor Type
2	**bDevCapabilityType**	1	Capability type: PLATFORM
3	**bReserved**	1	This field is reserved and shall be set to zero.
4	**PlatformCapabilityUUID**	16	A 128-bit number that uniquely identifies a platform-specific capability of the device. Refer to IETF RFC 4122 for details on generation of a UUID. See Microsoft OS 2.0 descriptor platform capability descriptor.
20	**CapabilityData**	Variable	A variable-length field containing data associated with the platform specific capability. This field may be zero bytes in length. See Microsoft OS 2.0 descriptor capability data.

Microsoft OS 2.0 descriptor platform capability descriptor

Microsoft defines a BOS platform capability descriptor and UUID that is used to indicate that the device supports the Microsoft OS 2.0 descriptor.

Table - Microsoft OS 2.0 descriptor platform capability UUID

Name	Platform Capability ID	Description
Microsoft OS 2.0 descriptor platform capability ID	D8DD60DF-4589-4CC7-9CD2-659 D9E648A9F	Indicates the device supports the Microsoft OS 2.0 descriptor

Microsoft OS 2.0 descriptor capability data

The MS OS 2.0 platform capability descriptor consists of a header followed by an array or one or more descriptor set information structures. Each descriptor set information structure contains information about a unique Microsoft OS 2.0 descriptor set. A device may want to return different versions of the Microsoft OS 2.0 descriptor set depending on the version of Windows running on the system to which it is connected.

wopEFB6.tmpMicrosoft_Visio_2003-2010_Drawing6.vsd

Figure - Microsoft OS 2.0 platform capability descriptor

This table shows the header section of the Microsoft OS 2.0 platform capability descriptor header.

Table . Microsoft OS 2.0 platform capability descriptor header

Offset	Field	Size	Value
0	**bLength**	1	Variable
1	**bDescriptorType**	1	16
2	**bDevCapabilityType**	1	5
3	**bReserved**	1	0
4	**MS_OS_20_Platform_Capability_ID**	16	D8DD60DF-4589-4CC7-9CD2-659D9E648A9F

This table shows the descriptor set information structure that follows the header section or another descriptor set. Each descriptor set information structure indicates the Windows version to which it applies.

Table . Descriptor set information structure

Offset	Field	Size	Description
0	**dwWindowsVersion**	4	Windows version
4	**wMSOSDescriptorSetTotalLength**	2	The length, in bytes of the MS OS 2.0 descriptor set.
6	**bMS_VendorCode**	1	Vendor defined code to use to retrieve this version of the MS OS 2.0 descriptor and also to set alternate enumeration behavior on the device.
7	**bAltEnumCode**	1	A non-zero value to send to the device to indicate that the device may return non-default USB descriptors for enumeration. If the device does not support alternate enumeration, this value shall be 0.

The Windows version indicates the minimum version of Windows that the descriptor set can be applied to. The DWORD value corresponds to the published NTDDI version constants in SDKDDKVER.H. The Windows USB driver stack uses the RtlIsServicePackVersionInstalled function to allow specifying service pack version numbers. The minimum Windows version allowed for the MS OS 2.0 descriptor set is Windows 8.1 Preview. Within the array of descriptor set information structures each **dwWindowsVersion** value must be unique.

Retrieving the Microsoft OS 2.0 descriptor set

After Windows has retrieved the MS OS vendor code from the Microsoft OS descriptor 2.0 platform specific capability descriptor, it retrieves the Microsoft OS descriptor 2.0 set by using a vendor-specific control request. The retrieval mechanism is similar to how it retrieves existing MS OS feature descriptors. For details please see the "How to retrieve an OS Feature Descriptor" section of the "Microsoft OS Descriptors Overview" document available here: http://msdn.microsoft.com/en-us/library/windows/hardware/gg463179.aspx.

Table . Format of the control request to retrieve MS OS 2.0 vendor-specific descriptor

bmRequestType	bRequest	wValue	wIndex	wLength	Data
1100 0000B	**bMS_VendorCode**	0x00	0x07	Length	Returned MS OS 2.0 Descriptor Set

bmRequestType

- Data Transfer Direction – Device to Host
- Type – Vendor
- Recipient - Device

bRequest

The **bMS_VendorCode** value returned in the descriptor set information structure.

wValue

Set to 0x00.

wIndex

0x7 for MS_OS_20_DESCRIPTOR_INDEX.

wLength

Length of the MS OS 2.0 descriptor set, as returned in the descriptor set information structure. Limited to 0xFFFF bytes.

MS OS 2.0 set alternate enumeration command

This command is sent by Windows if the Microsoft OS 2.0 descriptor set has a non-zero **bAltEnumCode** value (see Table 5. Descriptor set information structure). After receiving that command, the device may subsequently return alternate USB descriptors when Windows requests the information. The device shall revert to returning its default USB descriptors when the device is reset, and other default behavior.

Table 7. Format of the control request for MS OS 2.0 set alternate enumeration

bmRequestType	bRequest	wValue	wIndex	wLength	Data

0100 0000B	**bMS_VendorCode**	High Byte – bAltEnum Code Low Byte - 0	0x08	0	None.

bmRequestType

- Data Transfer Direction – Host to Device
- Type – Vendor
- Recipient - Device

bRequest

The bMS_VendorCode value returned in the descriptor set information structure.

wValue

- High Byte - Set to the non-zero bAltEnumCode value specified in the descriptor set information structure.
- Low Byte – Must be zero.

wIndex

0x8 for MS_OS_20_SET_ALT_ENUMERATION

wLength

Zero.

Microsoft OS 2.0 descriptor wIndex values

The Microsoft OS 2.0 descriptor set is retrieved in a similar manner as used for a version 1.0 Microsoft OS feature descriptor. As such, a new Microsoft OS feature descriptor **wIndex** value has been defined for the Microsoft OS 2.0 descriptor to avoid any potential conflicts with existing MS OS feature descriptor indexes.

These are the defined Microsoft OS 2.0 descriptor **wIndex** values.

Table . Microsoft OS 2.0 descriptor wIndex values

wIndex	Value
MS_OS_20_DESCRIPTOR_INDEX	0x07
MS_OS_20_SET_ALT_ENUMERATION	0x08

Microsoft OS 2.0 descriptor types

These values are used in the **wDescriptorType** field of the feature descriptors and headers.

Table . Microsoft OS 2.0 descriptor wDescriptorType values

wDescriptorType	Value
MS_OS_20_SET_HEADER_DESCRIPTOR	0x00
MS_OS_20_SUBSET_HEADER_CONFIGURATION	0x01
MS_OS_20_SUBSET_HEADER_FUNCTION	0x02
MS_OS_20_FEATURE_COMPATBLE_ID	0x03

MS_OS_20_FEATURE_REG_PROPERTY	0x04
MS_OS_20_FEATURE_MIN_RESUME_TIME	0x05
MS_OS_20_FEATURE_MODEL_ID	0x06
MS_OS_20_FEATURE_CCGP_DEVICE	0x07

Microsoft OS 2.0 descriptor set header

The MS OS 2.0 descriptor set header, subset headers, and feature descriptors are formatted as outlined in the following tables.

Table . Microsoft OS 2.0 descriptor set header

Offset	Field	Size	Description
0	**wLength**	2	The length, in bytes, of this header. Shall be set to 10.
2	**wDescriptorType**	2	MSOS20_SET_HEADER_DESCRIPTOR
4	**dwWindowsVersion**	4	Windows version.
8	**wTotalLength**	2	The size of entire MS OS 2.0 descriptor set. The value shall match the value in the descriptor set information structure.

Microsoft OS 2.0 configuration subset header

Table . Configuration subset header

Offset	Field	Size	Description
0	**wLength**	2	The length, in bytes, of this subset header. Shall be set to 8.
2	**wDescriptorType**	2	MS_OS_20 _SUBSET_HEADER_CONFIGURATION
4	**bConfigurationValue**	1	The configuration value for the USB configuration to which this subset applies.
5	**bReserved**	1	Shall be set to 0.
6	**wTotalLength**	2	The size of entire configuration subset including this header.

Microsoft OS 2.0 function subset header

Table . Function subset header

Offset	Field	Size	Description

0	**wLength**	2	The length, in bytes, of this subset header. Shall be set to 8.
2	**wDescriptorType**	2	MS_OS_20 _SUBSET_HEADER_FUNCTION
4	**bFirstInterface**	1	The interface number for the first interface of the function to which this subset applies.
5	**bReserved**	1	Shall be set to 0.
6	**wSubsetLength**	2	The size of entire function subset including this header.

Microsoft OS 2.0 feature descriptors

Compatible ID

The Microsoft OS 2.0 compatible ID descriptor is used to define a compatible device ID. Its usage is identical to the Microsoft OS extended configuration descriptor defined in MS OS descriptors specification version 1.0.

The compatible ID can be applied to the entire device or a specific function within a composite device.

Table . Microsoft OS 2.0 compatible ID descriptor

Offset	Field	Size	Description
0	**wLength**	2	The length, bytes, of the compatible ID descriptor including value descriptors. Shall be set to 20.
2	**wDescriptorType**	2	MS_OS_FEATURE_COMPATIBLE_ID
4	**CompatibleID**	8	Compatible ID String
12	**SubCompatibleID**	8	Sub-compatible ID String

Microsoft OS 2.0 registry property descriptor

The Microsoft OS 2.0 registry property descriptor is used to add per-device or per-function registry values that is read by the Windows USB driver stack or the device's function driver. Usage is similar to Microsoft OS extended property descriptor defined MS OS descriptors specification version 1.0.

Windows retrieves MS OS 2.0 registry property descriptor values during device enumeration as part of the overall descriptor set. However, only the values that are retrieved during the first device enumeration are written to the registry and used subsequently. The behavior is to maintain registry values that might be changed by the user.

The registry property descriptor can be applied to the entire device or a specific function within a composite device.

Table . Microsoft OS 2.0 registry property descriptor

Offset	Field	Size	Description
0	**wLength**	2	The length, in bytes, of this descriptor.
2	**wDescriptorType**	2	MS_OS_20 _FEATURE_REG_PROPERTY

4	**wPropertyDataType**	2	The type of registry property. See Table 15.
6	**wPropertyNameLength**	2	The length of the property name.
8	**PropertyName**	Variable	The name of registry property.
8+M	**wPropertyDataLength**	2	The length of property data.
10+M	**PropertyData**	Variable	Property data

Table . wPropertyDataType values for the Microsoft OS 2.0 registry property descriptor

Value	Description
0	RESERVED
1	A NULL-terminated Unicode String (REG_SZ)
2	A NULL-terminated Unicode String that includes environment variables (REG_EXPAND_SZ)
3	Free-form binary (REG_BINARY)
4	A little-endian 32-bit integer (REG_DWORD_LITTLE_ENDIAN)
5	A big-endian 32-bit integer (REG_DWORD_BIG_ENDIAN)
6	A NULL-terminated Unicode string that contains a symbolic link (REG_LINK)
7	Multiple NULL-terminated Unicode strings (REG_MULTI_SZ)
8 and higher	RESERVED

Microsoft OS 2.0 minimum USB resume time descriptor

The Microsoft OS 2.0 minimum USB resume time descriptor is used to indicate to the Windows USB driver stack the minimum time needed to recover after resuming from suspend, and how long the device requires resume signaling to be asserted. This descriptor is used for a device operating at high, full, or low-speed. It is not used for a device operating at SuperSpeed or higher.

This descriptor allows devices to recover faster than the default 10 millisecond specified in the USB 2.0 specification. It can also allow the host to assert resume signaling for less than the 20 milliseconds required in the USB 2.0 specification, in cases where the timing of resume signaling is controlled by software.

The USB resume time descriptor is applied to the entire device.

Table . Microsoft OS 2.0 minimum USB recovery time descriptor

Offset	Field	Size	Description
0	**wLength**	2	The length, in bytes, of this descriptor. Shall be set to 6.
2	**wDescriptorType**	2	MS_OS_20 _FEATURE_MIN_RESUME_TIME
4	**bResumeRecoveryTime**	1	The number of milliseconds the device requires to recover from port resume. (Valid values are 0 to 10)
5	**bResumeSignalingTime**	1	The number of milliseconds the device requires resume signaling to be asserted. (Valid values 1 to 20)

Microsoft OS 2.0 model ID descriptor

The Microsoft OS 2.0 model ID descriptor is used to uniquely identify the physical device.

The model ID descriptor is applied to the entire device.

Table . Microsoft OS 2.0 model ID descriptor

Offset	Field	Size	Description
0	**wLength**	2	The length, in bytes, of this descriptor. Shall be set to 20.
2	**wDescriptorType**	2	MS_OS_20_FEATURE_MODEL_ID
4	**ModelID**	16	This is a 128-bit number that uniquely identifies a physical device. Refer to IETF RFC 4122 for details on generation of a UUID.

Microsoft OS 2.0 CCGP device descriptor

The Microsoft OS 2.0 CCGP device descriptor is used to indicate that the device should be treated as a composite device by Windows regardless of the number of interfaces, configuration, or class, subclass, and protocol codes, the device reports.

The CCGP device descriptor must be applied to the entire device.

Table . Microsoft OS 2.0 CCGP device descriptor

Offset	Field	Size	Description
0	**wLength**	2	The length, in bytes, of this descriptor. Shall be set to 4.
2	**wDescriptorType**	2	MS_OS_20_CCGP_DEVICE

Example: Microsoft OS 2.0 descriptor sets for a registry value

This example demonstrates how Microsoft 2.0 descriptor sets can be used to provide a registry value. The device defines a single DWORD registry value of “SelectiveSuspendEnabled” that applies to all versions of Windows.

```
UCHAR Example1_MSOS20PlatformCapabilityDescriptor[0x1B] =
{
    //
    // Microsoft OS 2.0 Platform Capability Descriptor Header
    //
    0x1C,                    // bLength - 28 bytes
    0x10,                    // bDescriptorType - 16
    0x05,                    // bDevCapability – 5 for Platform Capability
    0x00,                    // bReserved - 0
    0xDF, 0x60, 0xDD, 0xD8,  // MS_OS_20_Platform_Capability_ID -
    0x89, 0x45, 0xC7, 0x4C,  // {D8DD60DF-4589-4CC7-9CD2-659D9E648A9F}
    0x9C, 0xD2, 0x65, 0x9D,  //
    0x9E, 0x64, 0x8A, 0x9F,  //

    //
    // Descriptor Information Set for Windows 8.1 or later
    //
    0x00, 0x00, 0x03, 0x06,  // dwWindowsVersion – 0x06030000 for Windows Blue
    0x48, 0x00,              // wLength – size of MS OS 2.0 descriptor set
    0x01,                    // bMS_VendorCode
    0x00,                    // bAltEnumCmd – 0 Does not support alternate enum
}

UCHAR Example1_MSOS20DescriptorSet[0x48] =
{
    //
    // Microsoft OS 2.0 Descriptor Set Header
    //
    0x0A, 0x00,             // wLength - 10 bytes
    0x00, 0x00,             // MSOS20_SET_HEADER_DESCRIPTOR
    0x00, 0x00, 0x03, 0x06, // dwWindowsVersion – 0x06030000 for Windows Blue
    0x48, 0x00,             // wTotalLength – 72 bytes

    //
    // Microsoft OS 2.0 Registry Value Feature Descriptor
    //
    0x3E, 0x00,             // wLength - 62 bytes
    0x04, 0x00,             // wDescriptorType – 4 for Registry Property
    0x04, 0x00,             // wPropertyDataType - 4 for REG_DWORD
    0x30, 0x00              // wPropertyNameLength – 48 bytes
    0x53, 0x00, 0x65, 0x00, // Property Name - “SelectiveSuspendEnabled”
    0x6C, 0x00, 0x65, 0x00,
    0x63, 0x00, 0x74, 0x00,
    0x69, 0x00, 0x76, 0x00,
    0x65, 0x00, 0x53, 0x00,
    0x75, 0x00, 0x73, 0x00,
    0x70, 0x00, 0x65, 0x00,
    0x6E, 0x00, 0x64, 0x00,
    0x45, 0x00, 0x6E, 0x00,
    0x61, 0x00, 0x62, 0x00,
    0x6C, 0x00, 0x65, 0x00,
    0x64, 0x00, 0x00, 0x00,
    0x04, 0x00,             // wPropertyDataLength – 4 bytes
    0x01, 0x00, 0x00, 0x00  // PropertyData - 0x00000001
}
```

Example: Microsoft OS 2.0 descriptor sets for a registry value based on specific Windows version

This example demonstrates how Microsoft 2.0 descriptor sets can be used to provide a single DWORD registry value of “SelectiveSuspendEnabled” that applies to Windows versions. For versions Windows 8.1 Build (NTDDI version 0x060300000), and a future version of Windows (NTDDI version 0x060?00000).

```
UCHAR Example2_MSOS20PlatformCapabilityDescriptor[0x24] =
{
    //
    // Microsoft OS 2.0 Platform Descriptor Header
    //
    0x24,                    // bLength - 36 bytes
    0x10,                    // bDescriptorType - 16
    0x05,                    // bDevCapability – 5 for Platform Capability
    0x00,                    // bReserved - 0
    0xDF, 0x60, 0xDD, 0xD8,  // MS_OS_20_Platform_Capability_ID -
    0x89, 0x45, 0xC7, 0x4C,  // {D8DD60DF-4589-4CC7-9CD2-659D9E648A9F}
    0x9C, 0xD2, 0x65, 0x9D,  //
    0x9E, 0x64, 0x8A, 0x9F,  //

    //
    // Descriptor Information Set for Windows 8.1 or later
    //
    0x00, 0x00, 0x03, 0x06,  // dwWindowsVersion – 0x06030000 for Windows 8.1 Build
    0x48, 0x00,              // wLength – size of MS OS 2.0 descriptor set
    0x01,                    // bMS_VendorCode_
    0x00,                    // bAltEnumCmd – 0 Does not support alternate enum

    //
    // Descriptor Information Set for future version of Windows or later
    //
    0x00, 0x00, 0x0?, 0x06,  // dwWindowsVersion – 0x060?0000 for future Windows version
    0x48, 0x00,              // wLength – size of MS OS 2.0 descriptor set
    0x02,                    // bMS_VendorCode
    0x10                     //  bAltEnumCmd – non-zero, supports alternate enum
 }

UCHAR Example2_MSOS20DescriptorSetForWindows Windows81OrLater[0x48] =
{
    //
    // Microsoft OS 2.0 Descriptor Set Header
    //
    0x0A, 0x00,             // wLength - 12 bytes
    0x00, 0x00,             // MSOS20_SET_HEADER_DESCRIPTOR
    0x00, 0x00, 0x03, 0x06, // dwWindowsVersion – 0x06030000 for Windows 8.1 Build
    0x4A, 0x00,             // wTotalLength – 72 bytes

    //
    // Microsoft OS 2.0 Registry Value Feature Descriptor
    //
    0x3E, 0x00,             // wLength - 62 bytes
    0x04, 0x00,             // wDescriptorType – 5 for Registry Property
    0x04, 0x00,             // wPropertyDataType - 4 for REG_DWORD
    0x30, 0x00,             // wPropertyNameLength – 48 bytes
    0x53, 0x00, 0x65, 0x00, // Property Name - “SelectiveSuspendEnabled”
    0x6C, 0x00, 0x65, 0x00,
    0x63, 0x00, 0x74, 0x00,
    0x69, 0x00, 0x76, 0x00,
    0x65, 0x00, 0x53, 0x00,
    0x75, 0x00, 0x73, 0x00,
    0x70, 0x00, 0x65, 0x00,
    0x6E, 0x00, 0x64, 0x00,
    0x45, 0x00, 0x6E, 0x00,
    0x61, 0x00, 0x62, 0x00,
    0x6C, 0x00, 0x65, 0x00,
    0x64, 0x00, 0x00, 0x00,
    0x04, 0x00,             // wPropertyDataLength – 4 bytes
    0x00, 0x00, 0x00, 0x00  // PropertyData - 0x00000000
}


// Applies to a specific future Windows version and later.

UCHAR Example2_MSOS20DescriptorSetForFutureWindows[0x4A] =
{
    //
    // Microsoft OS 2.0 Descriptor Set Header
    //
    0x0A, 0x00,             // wLength - 12 bytes
    0x00, 0x00,             // MSOS20_SET_HEADER_DESCRIPTOR
    0x00, 0x00, 0x0?, 0x06, // dwWindowsVersion – 0x060?0000 for future Windows version
    0x4A, 0x00,             // wTotalLength – 72 bytes

    //
    // Microsoft OS 2.0 Registry Value Feature Descriptor
    //
    0x3E, 0x00,             // bLength - 62 bytes
    0x04, 0x00,             // wDescriptorType – 5 for Registry Property
    0x04, 0x00,             // wPropertyDataType - 4 for REG_DWORD
    0x30, 0x00,             // wPropertyNameLength – 48 bytes
    0x53, 0x00, 0x65, 0x00, // Property Name - “SelectiveSuspendEnabled”
    0x6C, 0x00, 0x65, 0x00,
    0x63, 0x00, 0x74, 0x00,
    0x69, 0x00, 0x76, 0x00,
    0x65, 0x00, 0x53, 0x00,
    0x75, 0x00, 0x73, 0x00,
    0x70, 0x00, 0x65, 0x00,
    0x6E, 0x00, 0x64, 0x00,
    0x45, 0x00, 0x6E, 0x00,
    0x61, 0x00, 0x62, 0x00,
    0x6C, 0x00, 0x65, 0x00,
    0x64, 0x00, 0x00, 0x00,
    0x04, 0x00,              // wPropertyDataLength – 4 bytes
    0x01, 0x00, 0x00, 0x00  // PropertyData - 0x00000001
}
```

Resources

Microsoft OS 1.0 Descriptors

http://msdn.microsoft.com/en-us/library/windows/hardware/gg463179.aspx

USB Specification

http://www.usb.org/developers/docs

USB Generic Parent Driver (Usbccgp.sys)

http://msdn.microsoft.com/en-us/library/windows/hardware/ff539234