

## mcHID API Reference

### Functions

[Connect](#)  
[Disconnect](#)  
[Read](#)  
[Write](#)  
[ReadEx](#)  
[WriteEx](#)  
[GetVendorName](#)  
[GetProductName](#)  
[GetSerialNumber](#)  
[GetVendorID](#)  
[GetProductID](#)  
[GetVersion](#)  
[GetInputReportLength](#)  
[GetOutputReportLength](#)  
[GetHandle](#)  
[GetItem](#)  
[GetItemCount](#)  
[SetReadNotify](#)  
[IsReadNotifyEnabled](#)  
[IsAvailable](#)

### Events

[WM\\_HID\\_EVENT](#)  
[NotifyPlugged](#)  
[NotifyUnplugged](#)  
[NotifyChanged](#)  
[NotifyRead](#)

### Constants and Types

[Messages](#)

## Contact Information

Mecanique  
85 Marine Parade  
Saltburn by the Sea  
TS12 1BZ  
United Kingdom

Tel    *from the UK*                      01287 622382  
         *outside the UK*                +44 1287 622382

FAX    *from the UK*                      08700 520279  
         *outside the UK*                +44 8700 520279

email                                      [enquiries@mecanique.co.uk](mailto:enquiries@mecanique.co.uk)

## Connect

BOOL **Connect**(HWND *pHostWin*)

### Overview

Connect to the DLL.

### Parameters

- *pHostWin* - The window handle of the client application. If the value is NULL, the client application will be unable to receive [notification messages](#) from the DLL.

### Return Value

If the function succeeds the return value is non zero. If the function fails, the return value is zero.

### See Also

[Disconnect](#)

## Disconnect

BOOL **Disconnect**

### Overview

Disconnect from the DLL. There is no need to explicitly call this function during a session, as a connection is automatically terminated when the client application shuts down. However, you can call this function if you wish to terminate the connection before the client application is closed.

### Parameters

None

### Return Value

If the function succeeds the return value is non zero. If the function fails, the return value is zero.

### See Also

[Connect](#)

---

```
BOOL Read(UINT pHandle, VOID *pData)
```

## Overview

Read a report from a HID device. The number of bytes read is determined by the input report length. The report ID is contained in the first byte of the *pData* buffer.

## Parameters

- *pHandle* - A handle to a HID device.
- *pData* - Buffer for storing the bytes read.

## Return Value

If the function succeeds the return value is true. If the function fails, the return value is false.

## Example

```
Public Sub OnRead(ByVal pHandle As Long )  
  
    ' read all the data into the BufferIn array ...  
    If hidRead(pHandle, BufferIn(0)) Then  
  
        ' first byte is the report ID, e.g. BufferIn(0)  
        ' the other bytes are the data from the microcontrolller...  
  
    End If  
End Sub
```

## See Also

Write  
ReadEx  
WriteEx

---

```
BOOL Write(UINT pHandle, VOID *pData)
```

## Overview

Write a report to a HID device. The number of bytes written is determined by the output report length. The report ID should be contained in the first byte of the *pData* buffer.

## Parameters

- *pHandle* - A handle to a HID device.
- *pData* - Buffer containing the bytes to be written.

## Return Value

If the function succeeds the return value is true. If the function fails, the return value is false.

## Example

```
Public Sub WriteSomeData()  
    DeviceHandle As Long  
  
    BufferOut(0) = 0 ' first by is always the report ID  
    BufferOut(1) = 10 ' first data item, etc etc  
  
    ' write all the data in BufferOut...  
    DeviceHandle = hidGetHandle(VendorID, ProductID)  
    hidWriteEx VendorID, ProductID, BufferOut(0)  
End Sub
```

## See Also

Read  
ReadEx  
WriteEx

## ReadEx

```
BOOL ReadEx(UINT pVendorID, UINT pProductID, VOID *pData)
```

### Overview

Read a report from a HID device. The number of bytes read is determined by the input report length. The report ID is contained in the first byte of the pData buffer.

### Parameters

- *pVendorID* - A vendor (manufacturer) ID.
- *pProductID* - A product ID.
- *pData* - Buffer for storing the bytes read.

### Return Value

If the function succeeds the return value is true. If the function fails, the return value is false.

### See Also

[Read](#)  
[Write](#)  
[WriteEx](#)

## WriteEx

```
BOOL WriteEx(UINT pVendorID, UINT pProductID, VOID *pData)
```

### Overview

Write a report to a HID device. The number of bytes written is determined by the output report length. The report ID should be contained in the first byte of the *pData* buffer.

### Parameters

- *pVendorID* - A vendor (manufacturer) ID.
- *pProductID* - A product ID.
- *pData* - Buffer containing the bytes to be written.

### Return Value

If the function succeeds the return value is true. If the function fails, the return value is false.

### See Also

[Read](#)  
[Write](#)  
[ReadEx](#)



## GetVendorName

```
void GetVendorName(UINT pHandle, LPSTR pText, UINT pLen)
```

### Overview

Return the vendor (manufacturer) name.

### Parameters

- *pHandle* - A handle to a HID device.
- *pText* - Pointer to a buffer into which the function is to copy the vendor name.
- *pLen* - Specifies the length, in characters, of the buffer pointed to by the *pText* parameter.

### Return Value

None

### Example

```
Dim DeviceHandle As Long
Dim VendorName As String * 255
Dim ProductName As String * 255
Dim SerialNumber As String * 255

' get a device handle...
DeviceHandle = hidGetHandle(VendorID, ProductID)

' now get the vendor and product name from the handle...
hidGetVendorName DeviceHandle, VendorName, 255
hidGetProductName DeviceHandle, ProductName, 255
hidGetSerialNumber DeviceHandle, SerialNumber, 255
```

### See Also

[GetProductName](#)

[GetSerialNumber](#)

[GetVendorID](#)

[GetProductID](#)

[GetVersion](#)

## GetProductName

```
void GetProductName(UINT pHandle, LPSTR pText, UINT pLen)
```

### Overview

Return the product name.

### Parameters

- *pHandle* - A handle to a HID device.
- *pText* - Pointer to a buffer into which the function is to copy the product name.
- *pLen* - Specifies the length, in characters, of the buffer pointed to by the *pText* parameter.

### Return Value

None

### Example

```
Dim DeviceHandle As Long
Dim VendorName As String * 255
Dim ProductName As String * 255
Dim SerialNumber As String * 255

' get a device handle...
DeviceHandle = hidGetHandle(VendorID, ProductID)

' now get the vendor and product name from the handle...
hidGetVendorName DeviceHandle, VendorName, 255
hidGetProductName DeviceHandle, ProductName, 255
hidGetSerialNumber DeviceHandle, SerialNumber, 255
```

### See Also

[GetVendorName](#)  
[GetSerialNumber](#)  
[GetVendorID](#)  
[GetProductID](#)  
[GetVersion](#)

## GetSerialNumber

```
void GetSerialNumber(UINT pHandle, LPSTR pText, UINT pLen)
```

### Overview

Return the product serial number.

### Parameters

- *pHandle* - A handle to a HID device.
- *pText* - Pointer to a buffer into which the function is to copy the product serial number.
- *pLen* - Specifies the length, in characters, of the buffer pointed to by the *pText* parameter.

### Return Value

None

```
Dim DeviceHandle As Long
Dim VendorName As String * 255
Dim ProductName As String * 255
Dim SerialNumber As String * 255

' get a device handle...
DeviceHandle = hidGetHandle(VendorID, ProductID)

' now get the vendor and product name from the handle...
hidGetVendorName DeviceHandle, VendorName, 255
hidGetProductName DeviceHandle, ProductName, 255
hidGetSerialNumber DeviceHandle, SerialNumber, 255
```

### See Also

[GetVendorName](#)  
[GetProductName](#)  
[GetVendorID](#)  
[GetProductID](#)  
[GetVersion](#)

## GetVendorID

UINT GetVendorID(UINT *pHandle*)

### Overview

Get the vendor (manufacturer) ID.

### Parameters

- *pHandle* - A handle to a HID device.

### Return Value

The vendor (manufacturer) ID.

### See Also

[GetVendorName](#)  
[GetProductName](#)  
[GetSerialNumber](#)  
[GetProductID](#)  
[GetVersion](#)

## GetProductID

UINT GetProductID(UINT *pHandle*)

### Overview

Get the product ID.

### Parameters

- *pHandle* - A handle to a HID device.

### Return Value

The product ID.

### See Also

[GetVendorName](#)

[GetProductName](#)

[GetSerialNumber](#)

[GetVendorID](#)

[GetVersion](#)

## GetVersion

UINT GetVersion(UINT *pHandle*)

### Overview

Get the version number.

### Parameters

- *pHandle* - A handle to a HID device.

### Return Value

The version number.

### See Also

[GetVendorName](#)

[GetProductName](#)

[GetSerialNumber](#)

[GetVendorID](#)

[GetProductID](#)

## GetInputReportLength

UINT GetInputReportLength(UINT *pHandle*)

### Overview

Get the input buffer (report) length.

### Parameters

- *pHandle* - A handle to a HID device.

### Return Value

The input report length.

### See Also

[GetOutputReportLength](#)

## GetOutputReportLength

UINT GetOutputReportLength(UINT *pHandle*)

### Overview

Get the output buffer (report) length.

### Parameters

- *pHandle* - A handle to a HID device.

### Return Value

The output report length.

### See Also

[GetInputReportLength](#)



## GetHandle

```
UINT GetHandle(UINT pVendorID, UINT pProductID)
```

### Overview

Get a HID device handle from a given vendor and product ID.

### Parameters

- *pVendorID* - A vendor (manufacturer) ID.
- *pProductID* - A product ID.

### Return Value

If the function succeeds the return value is a non zero HID device handle. If the function fails, the return value is zero.

### Example

```
' get the device handle of the device we are interested in,  
' then set its read notify flag to true - this ensures you get  
' a read notification message when there is some data to read...  
DeviceHandle = hidGetHandle(VendorID, ProductID)  
hidSetReadNotify DeviceHandle, True
```

---

UINT GetItem(UINT *pIndex* )

## Overview

Get a HID device handle from a given index value. The number of available HID devices can be obtained with a call to GetItemCount .

## Parameters

- *pIndex* - HID item index.

## Return Value

None

## Example

```
' display all connected HID devices...
Public Sub DisplayDevices()
    Dim Index As Integer
    Dim DeviceHandle As Long
    Dim VendorName As String * 255
    Dim ProductName As String * 255

    ListBox.Clear

    ' do we have items to display...
    If hidGetItemCount() = 0 Then
        ListBox.AddItem ("No HID devices are connected")
    Else
        For Index = 0 To hidGetItemCount() - 1

            ' get a device handle...
            DeviceHandle = hidGetItem(Index)

            ' now get the vendor and product name from the handle...
            hidGetVendorName DeviceHandle, VendorName, 255
            hidGetProductName DeviceHandle, ProductName, 255

            ' display to the screen...
            ListBox.AddItem (VendorName)
            ListBox.List(Index) = ListBox.List(Index) + " - " + ProductName
        Next Index
    End If
End Sub
```

See Also

GetItemCount

## GetItemCount

UINT GetItemCount ( )

### Overview

Gets the total number of available HID devices.

### Parameters

None

### Return Value

Number of available HID devices.

### See Also

[GetItem](#)

## SetReadNotify

```
VOID SetReadNotify(UINT pHandle, BOOL pValue)
```

### Overview

When set to true, a NOTIFY\_READ event is triggered each time the DLL receives data from a given HID device. The NOTIFY\_READ event is disabled by default.

Set read notification to true during a NOTIFY\_CHANGED event. For example,

```
DevHandle = GetHandle(VendorID, ProductID);  
SetReadNotify(DevHandle, true);
```

### Parameters

- *pHandle* - A handle to a HID device.
- *pValue* - NOTIFY\_READ event flag.

### Return Value

None.

### Example

```
' get the device handle of the device we are interested in,  
' then set its read notify flag to true - this ensures you get  
' a read notification message when there is some data to read...  
DeviceHandle = hidGetHandle(VendorID, ProductID)  
hidSetReadNotify DeviceHandle, True
```

### See Also

[IsReadNotifyEnabled](#)

## IsReadNotifyEnabled

BOOL IsReadNotifyEnabled(UINT *pHandle*)

### Overview

Determines if the NOTIFY\_READ event flag is set. The NOTIFY\_READ event is disabled by default.

### Parameters

- *pHandle* - A handle to a HID device.

### Return Value

Returns true if read notification is enabled, false if it is disabled.

### See Also

[SetReadNotify](#)

## IsAvailable

BOOL IsAvailable(UINT *pVendorID*, UINT *pProductID*)

### Overview

Determines if a HID device is available.

### Parameters

- *pVendorID* - A vendor (manufacturer) ID.
- *pProductID* - A product ID.

### Return Value

Returns true if a HID device is available, false if unavailable.

### See Also

[ReadEx](#)

[WriteEx](#)

## Notification Messages

### WM\_HID\_EVENT

The WM\_HID\_EVENT message is triggered when a HID event occurs. The client application will only receive notification messages if a valid window handle has been passed to [Connect](#). The wParam parameter indicates the type of message sent and can be one of the following:

#### NotifyPlugged

Indicates that a HID device has been attached. The lParam parameter is a handle to HID device.

#### NotifyUnplugged

Indicates that HID device has been removed. The lParam parameter is a handle to HID device.

#### NotifyChanged

Indicates that a HID device has been attached or removed. This event is fired after either NotifyPlugged or NotifyUnplugged.

#### NotifyRead

Indicates that a HID device has sent a report. The lParam parameter is a handle to HID device.

See Also

[Notification Constants](#)

## Notification Constants

WM_HID_EVENT	WM_APP + 200
--------------	--------------

### Notification IDs

NOTIFY_PLUGGED	0x0001
NOTIFY_UNPLUGGED	0x0002
NOTIFY_CHANGED	0x0003
NOTIFY_READ	0x0004