FTDI Chip

Future Technology Devices International Ltd.

FTChipID Programmer's Guide

Table of Contents

1 Welcome to the FTChipID Programmer's Guide

This document describes the functions available in the FTChipID DLL which can be used to return FTDIChip-ID information for FT232R and FT245R devices.

Specifically, the FTID_GetDeviceChipID 7 function can be used to extract the unique FTDIChip-ID from an FT232R or FT245R device. This number is not changeable by the end user and can be used to tie application software to a specific piece of hardware containing an FT232R or FT245R device.

Please note that the latest version of FTDI's D2XX drivers must be installed to use the FTChipID DLL. The latest D2XX driver can be downloaded form the D2XX Drivers page of the FTDI web site.

The current version of the FTChipID DLL and several code examples are available for free download from the FTDIChip-ID page of the FTDI web site.

2 FTChipID Functions

2.1 FTID_GetNumDevices

Returns the number of available FT232R and FT245R devices connected to a system.

FTID_STATUS **FTID_GetNumDevices** (*lpdwNumDevices*)

Parameters

lpdwNumDevices	Pointer to a variable of type DWORD which receives the actual number of available FT232R and FT245R devices connected to a system

Return Value

FTID_SUCCESS if successful, otherwise the return value is one of the following FTID error codes:

FTID_IO_ERROR

Remarks

This function can be used to provide the maximum index for using with FTID_GetDeviceSerialNumber 4, FTID_GetDeviceDescription 5, FTID_GetDeviceLocationID 6 and FTID_GetDeviceChipID 7.

Example

```
FTID_STATUS Status = FTID_SUCCESS;
DWORD NumDevices = 0;

Status = FTID_GetNumDevices(&NumDevices);
```

2.2 FTID_GetDeviceSerialNumber

Returns the serial number of an available FT232R or FT245R device.

FTID_STATUS **FTID_GetDeviceSerialNumber** (DWORD *dwDeviceIndex*, LPSTR *lpSerialBuffer*, DWORD *dwSerialBufferLength*)

Parameters

dwDeviceIndex	Index of the FT232R or FT245R device.
lpSerialBuffer	Pointer to buffer that receives the serial number of the FT232R or FT245R device. The string will be NULL terminated.
dwSerialBufferLength	Length of the buffer created for the device serial number.

Return Value

FTID_SUCCESS if successful, otherwise the return value is one of the following FTID error codes:

FTID_DEVICE_NOT_FOUND
FTID_INVALID_DEVICE_NAME_INDEX
FTID_PASSED_NULL_POINTER
FTID_BUFFER_SIZE_TOO_SMALL
FTID_IO_ERROR

Remarks

The FTID_GetNumDevices 3 function can be used to obtain the number of available FT232R and FT245R devices connected to a system. The device index is 0 based.

Example

```
FTID_STATUS Status = FTID_SUCCESS;
char SerialNumber[256];

Status = FTID_GetDeviceSerialNumber(0, SerialNumber, 256);
```

2.3 FTID_GetDeviceDescription

Returns the description of an available FT232R or FT245R device.

FTID_STATUS **FTID_GetDeviceDescription** (DWORD *dwDeviceIndex*, LPSTR *lpDescriptionBuffer*, DWORD *dwDescriptionBufferLength*)

Parameters

dwDeviceIndex	Index of the FT232R or FT245R device.
lpDescriptionBuffer	Pointer to buffer that receives the description of the FT232R or FT245R device. The string will be NULL terminated.
dwDescriptionBufferLength	Length of the buffer created for the device description.

Return Value

FTID_SUCCESS if successful, otherwise the return value is one of the following FTID error codes:

FTID_DEVICE_NOT_FOUND
FTID_INVALID_DEVICE_NAME_INDEX
FTID_PASSED_NULL_POINTER
FTID_BUFFER_SIZE_TOO_SMALL
FTID_IO_ERROR

Remarks

The FTID_GetNumDevices 3 function can be used to obtain the number of available FT232R and FT245R devices connected to a system. The device index is 0 based.

Example

```
FTID_STATUS Status = FTID_SUCCESS;
char Description[256];

Status = FTID_GetDeviceDescription(0, Description, 256);
```

2.4 FTID_GetDeviceLocationID

Returns the location ID of an available FT232R or FT245R device.

FTID_STATUS **FTID_GetDeviceLocationID** (DWORD *dwDeviceIndex*, LPDWORD *lpdwLocationIDBuffer*)

Parameters

dwDeviceIndex	Index of the FT232R or FT245R device.
lpdwLocationIDBuffer	Pointer to buffer that receives the location ID for the FT232R or FT245R device.

Return Value

FTID_SUCCESS if successful, otherwise the return value is one of the following FTID error codes:

FTID_DEVICE_NOT_FOUND
FTID_INVALID_DEVICE_NAME_INDEX
FTID_IO_ERROR

Remarks

The FTID_GetNumDevices 3 function can be used to obtain the number of available FT232R and FT245R devices connected to a system. The device index is 0 based. **Please note that Linux does not support location IDs**.

Example

```
FTID_STATUS Status = FTID_SUCCESS;
DWORD LocID = 0;

Status = FTID_GetDeviceLocationID(0, &LocID);
```

2.5 FTID_GetDeviceChipID

Returns the FTDIChip-ID of an available FT232R or FT245R device.

FTID_STATUS **FTID_GetDeviceChipID** (DWORD *dwDeviceIndex*, LPDWORD *lpdwChipIDBuffer*)

Parameters

dwDeviceIndex	Index of the FT232R or FT245R device.
lpdwChipIDBuffer	Pointer to buffer that receives the FTDIChip-ID for the FT232R or FT245R device.

Return Value

FTID_SUCCESS if successful, otherwise the return value is one of the following FTID error codes:

FTID_DEVICE_NOT_FOUND
FTID_INVALID_DEVICE_NAME_INDEX
FTID_IO_ERROR

Remarks

The FTID_GetNumDevices 3 function can be used to obtain the number of available FT232R and FT245R devices connected to a system. The device index is 0 based.

Example

```
FTID_STATUS Status = FTID_SUCCESS;
DWORD ChipID = 0;

Status = FTID_GetDeviceChipID(0, &ChipID);
```

2.6 FTID_GetChipIDFromHandle

Returns the FTDIChip-ID of an FT232R or FT245R device using its handle.

FTID_STATUS **FTID_GetChipIDfromHandle** (FT_HANDLE *Handle*, LPDWORD *lpdwChipIDBuffer*)

Parameters

Handle	Valid handle of the FT232R or FT245R device.
lpdwChipIDBuffer	Pointer to buffer that receives the FTDIChip-ID for the FT232R or FT245R device.

Return Value

FTID_SUCCESS if successful, otherwise the return value is one of the following FTID error codes:

FTID_INVALID_HANDLE
FTID_DEVICE_NOT_FOUND
FTID_PASSED_NULL_POINTER
FTID_INVALID_RHANDLE
FTID_IO_ERROR

Remarks

The ftHandle parameter is a valid FT232R or FT245R handle returned from the D2XX functions FT_Open or FT_OpenEx. If the handle is for a different device type or is not a valid handle, the function will return FTID_INVALID_RHANDLE. The device handle must be closed using the D2XX FT_Close function when communication with the device is complete.

Example

```
FT_HANDLE Handle;
FT_STATUS ftStatus;
FTID_STATUS Status = FTID_SUCCESS;
DWORD ChipID = 0;

ftStatus = FT_Open(0, &Handle);
if(ftStatus != FT_OK) {
// FT_Open failed
return;
}

Status = FTID_GetDeviceChipID(Handle, &ChipID);
if(ftStatus != FT_OK) {
// Failed to get ChipID
return;
}

FT_Close(Handle);
```

2.7 FTID_GetDllVersion

Returns the FTChipID DLL version number.

FTID_STATUS **FTID_GetDLLVersion** (LPSTR *lpVersionBuffer*, DWORD *VersionBufferSize*)

Parameters

lpVersionBuffer	Pointer to buffer that receives the version number string of the FTChipID DLL.
VersionBufferSize	Length of the buffer created for the DLL version number.

Return Value

FTID_SUCCESS if successful, otherwise the return value is one of the following FTID error codes:

FTID_BUFFER_SIZE_TOO_SMALL
FTID_PASSED_NULL_POINTER

Example

```
FTID_STATUS Status = FTID_SUCCESS;
char Version[100];

Status = FTID_GetDLLVersion(Version, 100);
```

2.8 FTID_GetErrorCodeString

Returns an error code explanation in English.

FTID_STATUS **FTID_GetErrorCodeString** LPSTR *lpLanguage*, FTID_STATUS *ErrorCode*, LPSTR *lpErrorBuffer*, DWORD *ErrorBufferLength*

Parameters

lpLanguage	Language to return the error code explanation in.
ErrorCode	FTID_STATUS code to return the string for.
lpErrorBuffer	Buffer to receive the error code string.
ErrorBufferLength	Length of the buffer created for the DLL version number.

Return Value

FTID_SUCCESS if successful, otherwise the return value is one of the following FTID error codes:

FTID_BUFFER_SIZE_TOO_SMALL
FTID_PASSED_NULL_POINTER

Example

```
FTID_STATUS Status = FTID_SUCCESS;
char ErrorMessage[256];

dStatus = FTID_BUFFER_SIZE_TOO_SMALL;
Status = FTID_GetErrorCodeString("EN", dStatus, ErrorMessage, 256);
```

3 Appendix

3.1 Type Definitions

For Visual C++ applications, these values are pre-declared in the header file (FTChipID.h 12). For other languages, these definitions will have to be converted to use equivalent types and may have to be defined in an include file or within the body of the code.

DWORD	Unsigned long (4 bytes)
LPDWORD	Long pointer to a DWORD value
BOOL	Boolean value (4 bytes)
LPSTR	Long pointer to a NULL terminated string

FTID_STATUS (DWORD)

FTID_SUCCESS = 0
FTID_INVALID_HANDLE = 1
FTID_DEVICE_NOT_FOUND = 2
FTID_DEVICE_NOT_OPENED = 3
FTID_IO_ERROR = 4
FTID_INSUFFICIENT_RESOURCES = 5

FTID_BUFFER_SIZE_TOO_SMALL = 20
FTID_PASSED_NULL_POINTER = 21
FTID_INVALID_LANGUAGE_CODE = 22
FTID_INVALID_RHANDLE = 23
FTID_INVALID_STATUS_CODE = 0xFFFFFFFF

3.2 FTChipID.H

```
#ifndef __FTCHIPID_H_
#define __FTCHIPID_H_

// The following ifdef block is the standard way of creating macros which make exporting
// from a DLL simpler. All files within this DLL are compiled with the FTCHIPID_EXPORTS
// symbol defined on the command line. this symbol should not be defined on any project
// that uses this DLL. This way any other project whose source files include this file see
// FTCHIPID_API functions as being imported from a DLL, wheras this DLL sees symbols
// defined with this macro as being exported.
#ifdef FTCHIPID_EXPORTS
#define FTCHIPID_API __declspec(dllexport)
#else
#define FTCHIPID_API __declspec(dllimport)
#endif

typedef unsigned long FTID_STATUS;

// this can be moved to the API header
#define FTID_SUCCESS                                    0
#define FTID_INVALID_HANDLE                             1 //
FT_INVALID_HANDLE
#define FTID_DEVICE_NOT_FOUND                   2 // FT_DEVICE_NOT_FOUND
#define FTID_DEVICE_NOT_OPENED                  3 // FT_DEVICE_NOT_OPENED
#define FTID_IO_ERROR                                   4 // FT_IO_ERROR
#define FTID_INSUFFICIENT_RESOURCES             5 //
FT_INSUFFICIENT_RESOURCES
#define FTID_INVALID_PARAMETER                  6 // FT_INVALID_PARAMETER

#define FTID_BUFFER_SIZE_TOO_SMALL              20
#define FTID_PASSED_NULL_POINTER                21
#define FTID_INVALID_LANGUAGE_CODE              22
#define FTID_INVALID_RHANDLE                    23
#define FTID_INVALID_STATUS_CODE                0xFFFFFFFF

#ifdef __cplusplus
extern "C" {
#endif

// Device Related
FTCHIPID_API
FTID_STATUS WINAPI FTID_GetNumDevices(unsigned long * Devices);

FTCHIPID_API
FTID_STATUS WINAPI FTID_GetDeviceSerialNumber(unsigned long DeviceIndex, char *
SerialBuffer, unsigned long SerialBufferLength);

FTCHIPID_API
FTID_STATUS WINAPI FTID_GetDeviceDescription(unsigned long DeviceIndex, char *
DescriptionBuffer, unsigned long DescriptionBufferLength);

FTCHIPID_API
FTID_STATUS WINAPI FTID_GetDeviceLocationID(unsigned long DeviceIndex, unsigned long *
LocationIDBuffer);

FTCHIPID_API
FTID_STATUS WINAPI FTID_GetDeviceChipID(unsigned long DeviceIndex, unsigned long *
ChipIDBuffer);

FTCHIPID_API
FTID_STATUS WINAPI FTID_GetChipIDFromHandle(FT_HANDLE Handle, unsigned long *
ChipIDBuffer);

// General
FTCHIPID_API
FTID_STATUS WINAPI FTID_GetDllVersion(char * VersionBuffer, unsigned long
VersionBufferSize);

FTCHIPID_API
FTID_STATUS WINAPI FTID_GetErrorCodeString(char * Language, FTID_STATUS ErrorCode,
char * ErrorBuffer, unsigned long ErrorBufferLength);

#ifdef __cplusplus
}
#endif


#endif
```

Index