bridgeCo™

THESYCON
Thesycon® Systemsoftware & Consulting GmbH

BridgeCo 1394 Download Tools

Reference Manual

Version:	2.30.0
Date:	13 July 2006

Authors: Alexander Landgraf e-mail: Alexander.Landgraf@thesycon.de
Rene Moeller e-mail: Rene.Moeller@thesycon.de

BridgeCo AG
Ringstrasse 40
CH-8600 Duebendorf
Switzerland
+41 1 802 3333
http://www.bridgeco.net

Thesycon Systemsoftware & Consulting GmbH
Werner-von-Siemens-Str. 2
D-98693 Ilmenau
Germany
+49 3677 8462 0
http://www.thesycon.de

Contents

1 Overview

This document describes the features, the programming interface and the usage of the BridgeCo IEEE1394 download tools.

The download tools consist of the following components:

- A device driver named bco_dnld.sys. This driver will be installed automatically during setup.
- A DLL named bcodl.dll. This DLL provides a C style function-based application programming interface (API) that supports firmware download operations. The API is defined in the header file bcodl.h. Refer to chapter 2 for a detailed description of the programming interface.
- A console-mode utility named bcodl.exe. This utility supports all operations provided by the download DLL. Refer to chapter 3 for a description of the bcodl utility.

  To demonstrate the usage of the download API, the source code of the bcodl utility is provided as an application example.

2 Download DLL

This section describes the API exported by the BridgeCo 1394 download DLL (bcodl.dll).

2.1 Functions

This section describes the functions exported by the download DLL. These functions are declared in bcodl.h.

BCODL_GetVersion

The function returns the current version number of the software interface (API) exported by the DLL.

Definition

```
unsigned long
BCODL_GetVersion();
```

Return Value

The return value indicates the current API version number supported by the DLL. The high-order 16 bits contain the major version number, the low-order 16 bits contain the minor version number.

Comments

If changes are made to the programming interface that are compatible with previous versions the minor version number (low order byte) will be incremented. If changes are made that cause an incompatibility with previous versions of the interface the major version number (high order byte) will be incremented. Applications should check the return value of this function against the **`BCODL_VERSION`** constant to make sure the DLL supports the expected API version. The major version number must match the expected number exactly. The minor version number needs to be greater than or equal to the expected number.

BCODL_OpenBcdFile

The function opens the specified BCD file.

Definition

```
BCOFWDL_STATUS
BCODL_OpenBcdFile(
     const char* BcdFileName
     );
```

Parameter

`BcdFileName`
Specifies the name of the BCD file to open.

Return Value

If the function succeeds the return value is **`BCOFWDL_STATUS_SUCCESS`**. If the function fails one of the **`BCOFWDL_STATUS_xxx`** status codes is returned.

Comments

There can be only one open BCD file at a given point in time. In order to open another BCD file **BCODL_CloseBcdFile** has to be called before.

If a download is in progress the function call will fail.

See Also

BCODL_CloseBcdFile (page 15)

BCODL_GetBcdFileFlags

The function returns the flags field from the header of the currently open BCD file.

Definition

```
unsigned long
BCODL_GetBcdFileFlags();
```

Return Value

The return value contains the flags of the BCD file or zero if no file is open.

Comments

The BCD file has to be opened before by a call to **BCODL_OpenBcdFile**.

See Also

BCODL_OpenBcdFile (page 7)

BCODL_GetBcdFileSwId

The function returns the software ID field from the header of the currently open BCD file.

Definition

```
unsigned long
BCODL_GetBcdFileSwId();
```

Return Value

The return value contains the software ID of the BCD file or zero if no file is open.

Comments

The BCD file has to be opened before by a call to **BCODL_OpenBcdFile**.

See Also

BCODL_OpenBcdFile (page 7)

BCODL_GetBcdFileSwVersion

The function returns the software version field from the header of the currently open BCD file.

Definition

```
unsigned long
BCODL_GetBcdFileSwVersion();
```

Return Value

The return value contains the software version of the BCD file or zero if no file is open.

Comments

The BCD file has to be opened before by a call to **BCODL_OpenBcdFile**.

See Also

BCODL_OpenBcdFile (page 7)

BCODL_GetBcdFileCompHwId

The function returns the compatible hardware ID field from the header of the currently open BCD file.

Definition

```
unsigned long
BCODL_GetBcdFileCompHwId();
```

Return Value

The return value contains the compatible hardware ID of the BCD file or zero if no file is open.

Comments

The BCD file has to be opened before by a call to **BCODL_OpenBcdFile**.

See Also

BCODL_OpenBcdFile (page 7)

BCODL_GetBcdFileCreationDate

The function returns the creation date field from the header of the currently open BCD file as zero terminated string.

Definition

```
const char*
BCODL_GetBcdFileCreationDate();
```

Return Value

The return value contains the creation date of the BCD file in the format "YYYYMMDD". An empty string is returned if no file is open.

Comments

The BCD file has to be opened before by a call to **BCODL_OpenBcdFile**.

See Also

BCODL_OpenBcdFile (page 7)

BCODL_GetBcdFileCreationTime

The function returns the creation time field from the header of the currently open BCD file as zero terminated string.

Definition

```
const char*
BCODL_GetBcdFileCreationTime();
```

Return Value

The return value contains the creation time of the BCD file in the format "hhmmss". An empty string is returned if no file is open.

Comments

The BCD file has to be opened before by a call to **BCODL_OpenBcdFile**.

See Also

BCODL_OpenBcdFile (page 7)

BCODL_GetBcdFileInfoText

The function returns the info text field from the header of the currently open BCD file as zero terminated string.

Definition

```
const char*
BCODL_GetBcdFileInfoText();
```

Return Value

The return value contains the info text of the BCD file. An empty string is returned if no file is open.

Comments

The BCD file has to be opened before by a call to **BCODL_OpenBcdFile**.

See Also

BCODL_OpenBcdFile (page 7)

BCODL_CloseBcdFile

The function closes the currently open BCD file.

Definition

```
BCOFWDL_STATUS
BCODL_CloseBcdFile();
```

Return Value

If the function succeeds the return value is **`BCOFWDL_STATUS_SUCCESS`**. If the function fails one of the **`BCOFWDL_STATUS_xxx`** status codes is returned.

Comments

If no BCD file is open the function does nothing.

If a download is in progress the function call will fail.

See Also

BCODL_OpenBcdFile (page 7)

BCODL_EnumerateDevices

The function enumerates the connected devices.

Definition

```
BCOFWDL_STATUS
BCODL_EnumerateDevices();
```

Return Value

If the function succeeds the return value is **BCOFWDL_STATUS_SUCCESS**. If the function fails one of the **BCOFWDL_STATUS_xxx** status codes is returned.

Comments

Any enumeration information retrieved by a previous call to **BCODL_EnumerateDevices** will be discarded. So don't rely on device indices determined before.

The number of enumerated devices can be retrieved subsequently by calling **BCODL_GetNumberOfDevices**.

The function call will fail if a device is currently open.

It will also fail if a download is in progress.

See Also

BCODL_GetNumberOfDevices (page 18)
BCODL_OpenDevice (page 19)
BCODL_CloseDevice (page 35)

BCODL_ClearEnumeration

The function discards any enumeration information retrieved by a previous call to **BCODL_EnumerateDevices**.

Definition

```
BCOFWDL_STATUS
BCODL_ClearEnumeration();
```

Return Value

If the function succeeds the return value is **BCOFWDL_STATUS_SUCCESS**. If the function fails one of the **BCOFWDL_STATUS_xxx** status codes is returned.

Comments

The function call will fail if a device is currently open.

It will also fail if a download is in progress.

See Also

BCODL_EnumerateDevices (page 16)
BCODL_CloseDevice (page 35)

BCODL_GetNumberOfDevices

The function provides the number of connected devices enumerated by a previous call to **BCODL_EnumerateDevices**.

Definition

```
BCOFWDL_STATUS
BCODL_GetNumberOfDevices();
```

Return Value

The return value contains the number of connected devices. Zero is returned if no devices are connected and turned on, if **BCODL_EnumerateDevices** hasn't been called before or the enumeration information isn't available anymore. The latter is true after a download has been started by calling **BCODL_StartDownload** or reset has been executed by calling **BCODL_ResetDevice**.

See Also

BCODL_EnumerateDevices (page 16)
BCODL_StartDownload (page 37)
BCODL_ResetDevice (page 45)

BCODL_OpenDevice

The function opens one of the devices previously enumerated by **BCODL_EnumerateDevices**.

Definition

```
BCOFWDL_STATUS
BCODL_OpenDevice(
     unsigned long DeviceIndex
     );
```

Parameter

DeviceIndex

Specifies a zero-based index that identifies the device to open. The valid index range is 0 to NumberOfDevices-1. NumberOfDevices is returned by **BCODL_GetNumberOfDevices**.

Return Value

If the function succeeds the return value is **BCOFWDL_STATUS_SUCCESS**. If the function fails one of the **BCOFWDL_STATUS_xxx** status codes is returned.

Comments

There can be only one device opened at a given point in time. In order to open another device **BCODL_CloseDevice** has to be called before.

If a download is in progress the function call will fail.

See Also

BCODL_EnumerateDevices (page 16)
BCODL_GetNumberOfDevices (page 18)
BCODL_CloseDevice (page 35)

BCODL_GetDevProtocolVersion

The function provides the protocol version of the currently open device.

Definition

```
unsigned long
BCODL_GetDevProtocolVersion();
```

Return Value

The return value contains the protocol version of the device or zero if no device is open.

Comments

The device has to be opened before by a call to **BCODL_OpenDevice**.

See Also

BCODL_OpenDevice (page 19)

BCODL_GetDevBootloaderVersion

The function provides the bootloader version of the currently open device.

Definition

```
unsigned long
BCODL_GetDevBootloaderVersion();
```

Return Value

The return value contains the bootloader version of the device or zero if no device is open.

Comments

The device has to be opened before by a call to **BCODL_OpenDevice**.

See Also

BCODL_OpenDevice (page 19)

BCODL_GetDevGuid

The function provides a single byte of the GUID of the currently open device.

Definition

```
long
BCODL_GetDevGuid(
    unsigned long ByteIndex
    );
```

Parameter

`ByteIndex`
Specifies the zero-based index of the byte that should be returned.

Return Value

The return value contains the specified byte of the device GUID, zero if no device is open or -1 for an invalid index.

Comments

Currently the GUID consists of 8 bytes and should be queried in the following index order: 3-2-1-0-7-6-5-4.

The device has to be opened before by a call to **BCODL_OpenDevice**.

See Also

BCODL_OpenDevice (page 19)
BCODL_SetGuid (page 43)

BCODL_GetDevHwModelId

The function provides the hardware model ID of the currently open device.

Definition

```
unsigned long
BCODL_GetDevHwModelId();
```

Return Value

The return value contains the hardware model ID of the device or zero if no device is open.

Comments

The device has to be opened before by a call to **BCODL_OpenDevice**.

See Also

BCODL_OpenDevice (page 19)

BCODL_GetDevHwVersion

The function provides the hardware version of the currently open device.

Definition

```
unsigned long
BCODL_GetDevHwVersion();
```

Return Value

The return value contains the hardware version of the device or zero if no device is open.

Comments

The device has to be opened before by a call to **BCODL_OpenDevice**.

See Also

BCODL_OpenDevice (page 19)

BCODL_GetDevSwDate

The function provides the software date of the currently open device as zero terminated string.

Definition

```
const char*
BCODL_GetDevSwDate();
```

Return Value

The return value contains the software date of the device in the format "YYYYMMDD". An empty string is returned if no device is open.

Comments

The device has to be opened before by a call to **BCODL_OpenDevice**.

See Also

BCODL_OpenDevice (page 19)

BCODL_GetDevSwTime

The function provides the software time of the currently open device as zero terminated string.

Definition

```
const char*
BCODL_GetDevSwTime();
```

Return Value

The return value contains the software time of the device in the format "hhmmss". An empty string is returned if no device is open.

Comments

The device has to be opened before by a call to **BCODL_OpenDevice**.

See Also

BCODL_OpenDevice (page 19)

BCODL_GetDevSwId

The function provides the software ID of the currently open device.

Definition

```
unsigned long
BCODL_GetDevSwId();
```

Return Value

The return value contains the software ID of the device or zero if no device is open.

Comments

The device has to be opened before by a call to **BCODL_OpenDevice**.

See Also

BCODL_OpenDevice (page 19)

BCODL_GetDevSwVersion

The function provides the software version of the currently open device.

Definition

```
unsigned long
BCODL_GetDevSwVersion();
```

Return Value

The return value contains the software version of the device or zero if no device is open.

Comments

The device has to be opened before by a call to **BCODL_OpenDevice**.

See Also

BCODL_OpenDevice (page 19)

BCODL_GetDevBaseAddress

The function provides the base address of the currently open device.

Definition

```
unsigned long
BCODL_GetDevBaseAddress();
```

Return Value

The return value contains the base address of the device or zero if no device is open.

Comments

The device has to be opened before by a call to **BCODL_OpenDevice**.

See Also

BCODL_OpenDevice (page 19)

BCODL_GetDevMaxImageLength

The function provides the maximum image length of the currently open device.

Definition

```
unsigned long
BCODL_GetDevMaxImageLength();
```

Return Value

The return value contains the maximum image length of the device or zero if no device is open.

Comments

The device has to be opened before by a call to **BCODL_OpenDevice**.

See Also

BCODL_OpenDevice (page 19)

BCODL_GetDevBootloaderDate

The function provides the bootloader date of the currently open device as zero terminated string.

Definition

```
const char*
BCODL_GetDevBootloaderDate();
```

Return Value

The return value contains the bootloader date of the device in the format "YYYYMMDD". An empty string is returned if no device is open.

Comments

The device has to be opened before by a call to **BCODL_OpenDevice**.

See Also

BCODL_OpenDevice (page 19)

BCODL_GetDevBootloaderTime

The function provides the bootloader time of the currently open device as zero terminated string.

Definition

```
const char*
BCODL_GetDevBootloaderTime();
```

Return Value

The return value contains the bootloader time of the device in the format "hhmmss". An empty string is returned if no device is open.

Comments

The device has to be opened before by a call to **BCODL_OpenDevice**.

See Also

BCODL_OpenDevice (page 19)

BCODL_GetDevVendorDesc

The function provides a vendor description of the currently open device as zero terminated string.

Definition

```
const char*
BCODL_GetDevVendorDesc();
```

Return Value

The return value contains a vendor description of the device. An empty string is returned if no device is open or if no description is available.

Comments

The device has to be opened before by a call to **BCODL_OpenDevice**.

See Also

BCODL_OpenDevice (page 19)

BCODL_GetDevModelDesc

The function provides a model description of the first unit of the currently open device as zero terminated string.

Definition

```
const char*
BCODL_GetDevModelDesc();
```

Return Value

The return value contains a model description of the first unit of the device. An empty string is returned if no device is open or if no description is available.

Comments

The device has to be opened before by a call to **BCODL_OpenDevice**.

See Also

BCODL_OpenDevice (page 19)

BCODL_CloseDevice

The function closes the currently open device.

Definition

```
BCOFWDL_STATUS
BCODL_CloseDevice();
```

Return Value

If the function succeeds the return value is **BCOFWDL_STATUS_SUCCESS**. If the function fails one of the **BCOFWDL_STATUS_xxx** status codes is returned.

Comments

If no device is open the function does nothing.

If a download is in progress the function call will fail.

See Also

BCODL_OpenDevice (page 19)

BCODL_SetTimeout

The function sets the timeout interval for the specified operation.

Definition

```
BCOFWDL_STATUS
BCODL_SetTimeout(
     BCODL_TIMEOUT TimeoutParameter,
     unsigned long TimeoutInterval
     );
```

Parameters

`TimeoutParameter`
Specifies the operation for which the timeout interval is to be set. See **BCOFWDL_TIMEOUT** for more information.

`TimeoutInterval`
Specifies the timeout interval, in milliseconds, for the specified operation.

Return Value

If the function succeeds the return value is **`BCOFWDL_STATUS_SUCCESS`**. If the function fails one of the **`BCOFWDL_STATUS_xxx`** status codes is returned.

Comments

A default value is used for every timeout parameter not set by this function (see **`BCOFWDL_DEFAULT_xxx`**). Use this function only if required.

The timeout value set by this function applies to all subsequent API calls until a new value is set by this function or the DLL is reloaded.

If a download is in progress the function call will fail.

See Also

BCOFWDL_TIMEOUT (page 49)

BCODL_StartDownload

The function starts the download of the currently open BCD file to the currently open device.

Definition

```
BCOFWDL_STATUS
BCODL_StartDownload(
    BCOFWDL_IMAGE_TYPE ImageType,
    unsigned long Flags
    );
```

Parameters

ImageType
Specifies the type of image data contained in the BCD file that should be downloaded. See **BCOFWDL_IMAGE_TYPE** for further information.

Flags
Specifies flags that control the download. The value has to be zero or any combination (bit-wise OR) of the following values:

BCOFWDL_DISABLE_HW_MODEL_ID_CHECK
If this flag is specified no check is performed whether the hardware model ID provided with the device's info registers is equal to the compatibility hardware ID provided with the header of the BCD file to download.

Note: This flag is only intended for developers. It should never be possible for end users to set this flag.

BCOFWDL_DISABLE_NODE_VENDOR_ID_CHECK
If this flag is specified no check is performed whether the node vendor ID provided with the device's info registers is equal to the node vendor ID provided with the header of the BCD file to download.

Note: This flag is only intended for developers. It should never be possible for end users to set this flag.

Return Value

If the function succeeds the return value is **BCOFWDL_STATUS_SUCCESS**. If the function fails one of the **BCOFWDL_STATUS_xxx** status codes is returned.

Comments

The device has to be opened before by a call to **BCODL_OpenDevice**. Furthermore the BCD file has to be opened before by a call to **BCODL_OpenBcdFile**.

The function initializes and starts the download procedure and returns immediately. The download process is handled internally by a worker thread and can take several minutes.

An application should monitor the progress by calling **BCODL_GetDownloadStatus**, **BCODL_GetTotalBytesToDownload** and **BCODL_GetBytesDownloaded**.

Note: The download requires a device reset which causes a bus-reset too. Every bus-reset invalidates the list of enumerated devices. To open another device and/or start the download again **BCODL_EnumerateDevices** has to be called again. However, the currently open device still has to be closed after the download has been finished.

If a download is in progress the function call will fail.

See Also

BCODL_EnumerateDevices (page 16)
BCODL_OpenDevice (page 19)
BCODL_OpenBcdFile (page 7)
BCODL_GetDownloadStatus (page 39)
BCODL_GetTotalBytesToDownload (page 40)
BCODL_GetBytesDownloaded (page 41)
BCODL_SetTimeout (page 36)

BCODL_GetDownloadStatus

The function retrieves the status of the current download.

Definition

```
BCOFWDL_STATUS
BCODL_GetDownloadStatus();
```

Return Value

The return value contains the status of the download. If **BCOFWDL_STATUS_SUCCESS** is returned the download has been finished successfully.
BCOFWDL_STATUS_DOWNLOAD_IN_PROGRESS is returned if the download is still in progress. If the download fails, another **BCOFWDL_STATUS_xxx** status codes will be returned.

Comments

The download has to be started before by a call to **BCODL_StartDownload**.

See Also

BCODL_StartDownload (page 37)
BCODL_GetTotalBytesToDownload (page 40)
BCODL_GetBytesDownloaded (page 41)

BCODL_GetTotalBytesToDownload

The function retrieves the total amount of bytes to download.

Definition

```
unsigned long
BCODL_GetTotalBytesToDownload();
```

Return Value

The return value contains the total amount of bytes to download.

Comments

The download has to be started before by a call to **BCODL_StartDownload**.

See Also

BCODL_StartDownload (page 37)
BCODL_GetDownloadStatus (page 39)
BCODL_GetBytesDownloaded (page 41)

BCODL_GetBytesDownloaded

The function retrieves the amount of bytes already downloaded.

Definition

```
unsigned long
BCODL_GetBytesDownloaded();
```

Return Value

The return value contains the amount of bytes already downloaded.

Comments

The download has to be started before by a call to **BCODL_StartDownload**.

See Also

BCODL_StartDownload (page 37)
BCODL_GetDownloadStatus (page 39)
BCODL_GetTotalBytesToDownload (page 40)

BCODL_ResetDevice

The function resets the currently open device.

Definition

```
BCOFWDL_STATUS
BCODL_ResetDevice(
     BCOFWDL_RESET_TYPE Type,
     unsigned long Flags
     );
```

Parameters

`Type`
Indicates the type of the reset. See **BCOFWDL_RESET_TYPE** for further information.

`Flags`
This parameter is reserved for further use and should be set to 0.

Return Value

If the function succeeds the return value is **`BCOFWDL_STATUS_SUCCESS`**. If the function fails one of the **`BCOFWDL_STATUS_xxx`** status codes is returned.

Comments

The device has to be opened before by a call to **BCODL_OpenDevice**.

Note, any bus-reset invalidates the list of enumerated devices. To open another device after the function returned **BCODL_EnumerateDevices** has to be called again. However, the currently open device still has to be closed.

If a download is in progress the function call will fail.

See Also

BCODL_OpenDevice (page 19)
BCODL_CloseDevice (page 35)
BCODL_EnumerateDevices (page 16)
BCODL_SetTimeout (page 36)

BCODL_SetGuid

The function sets the GUID of the currently open device.

Definition

```
BCOFWDL_STATUS
BCODL_SetGuid(
     long Byte0,
     long Byte1,
     long Byte2,
     long Byte3,
     long Byte4,
     long Byte5,
     long Byte6,
     long Byte7
     );
```

Parameters

`Byte0`
The first byte of the device GUID.

`Byte1`
The second byte of the device GUID.

`Byte2`
The third byte of the device GUID.

`Byte3`
The 4th byte of the device GUID.

`Byte4`
The 5th byte of the device GUID.

`Byte5`
The 6th byte of the device GUID.

`Byte6`
The 7th byte of the device GUID.

`Byte7`
The 8th byte of the device GUID.

Return Value

If the function succeeds the return value is **`BCOFWDL_STATUS_SUCCESS`**. If the function fails one of the **`BCOFWDL_STATUS_xxx`** status codes is returned.

Comments

The device has to be opened before by a call to **BCODL_OpenDevice**.

If a download is in progress the function call will fail.

See Also

BCODL_OpenDevice (page 19)
BCODL_GetDevGuid (page 22)
BCODL_SetTimeout (page 36)

BCODL_ResetDevice

The function initializes the persistent parameters of the currently open device.

Definition

```
BCOFWDL_STATUS
BCODL_ResetDevice();
```

Return Value

If the function succeeds the return value is **`BCOFWDL_STATUS_SUCCESS`**. If the function fails one of the **`BCOFWDL_STATUS_xxx`** status codes is returned.

Comments

The device has to be opened before by a call to **BCODL_OpenDevice**.

If a download is in progress the function call will also fail.

See Also

BCODL_OpenDevice (page 19)
BCODL_SetTimeout (page 36)

BCODL_EnableLogging

The function enables or disables logging of internal processing.

Definition

```
BCOFWDL_STATUS
BCODL_EnableLogging(
     const char* LogFile
     );
```

Parameter

`LogFile`

Zero-terminated ASCII string that specifies the log file that will receive messages produced by the DLL. If the file already exists the messages will be appended to the existing file. The current log file, if any, will be closed before the specified file is opened.

To disable logging and close the currently open log file set the parameter to NULL. If no log file is open and NULL is specified the function does nothing.

Return Value

If the function succeeds the return value is **`BCOFWDL_STATUS_SUCCESS`**. If the function fails one of the **`BCOFWDL_STATUS_xxx`** status codes is returned.

Comments

The log file produced by the DLL may be helpful in debugging and problem analysis. The messages provide information on the actions performed by the DLL and about any problems that occurred.

If a download is in progress the function call will fail.

2.2 Structures and Types

This section describes data structures and data types exported by the download DLL. These structures and types are declared in bcodldefs.h.

BCOFWDL_IMAGE_TYPE

Defines the type of firmware to download.

Definition

```
typedef enum _BCOFWDL_IMAGE_TYPE{
     ImageTypeApplicationAndCnE,
     ImageTypeApplicationOnly,
     ImageTypeCnEOnly,
     ImageTypeBootloader,
     ImageType32bitEnum
} BCOFWDL_IMAGE_TYPE;
```

Entries

`ImageTypeApplicationAndCnE`
The application and the CnE data will be downloaded.

`ImageTypeApplicationOnly`
Only the application data will be downloaded.

`ImageTypeCnEOnly`
Only the CnE data will be downloaded.

`ImageTypeBootloader`
The bootloader will be downloaded.

`ImageType32bitEnum`
reserved, do not use

BCOFWDL_RESET_TYPE

Defines the type of device reset that specifies what the device has to do after reset is done.

Definition

```
typedef enum _BCOFWDL_RESET_TYPE{
     ResetTypeStartApplication,
     ResetTypeHaltInBootloader,
     ResetTypeStartDebugger,
     ResetType32bitEnum
} BCOFWDL_RESET_TYPE;
```

Entries

`ResetTypeStartApplication`
start the application

`ResetTypeHaltInBootloader`
stay in bootloader mode

`ResetTypeStartDebugger`
start the debugger

`ResetType32bitEnum`
reserved, do not use

BCOFWDL_TIMEOUT

Defines the timeout controlled operations.

Definition

```
typedef enum _BCOFWDL_TIMEOUT{
     TimeoutReset,
     TimeoutDownloadStart,
     TimeoutDownloadBlock,
     TimeoutDownloadEnd,
     TimeoutProgramGuid,
     TimeoutInitPersParams,
     TimeoutForce32bitEnum
} BCOFWDL_TIMEOUT;
```

Entries

`TimeoutReset`
reset device

`TimeoutDownloadStart`
start download operation which includes erasing of FLASH memory

`TimeoutDownloadBlock`
download block operation which includes programming of FLASH memory

`TimeoutDownloadEnd`
download end operation which includes calculating of a checksum

`TimeoutProgramGuid`
program GUID operation

`TimeoutInitPersParams`
initialize persistent parameters operation

`TimeoutForce32bitEnum`
reserved, do not use

3 Download Utility

This section describes the usage of the bcodl console-mode download utility. The utility supports various commands and options. Some commands require additional arguments to be specified. A detailed description of every command and the arguments and options that apply is given below.

3.1 Online Help

```
bcodl
bcodl -h
bcodl -?
```

If the bcodl utility is executed without any arguments or with the `'-h'` or `'-?'` option given, a short help text will be displayed.

3.2 Return Codes

The bcodl utility returns zero for success. A positive return code represents a status code of the bcodl DLL API. See bcodldefs.h for a list of these codes. Errors occured in the bcodl utility itself are represented by negative return codes. These are described in the table below.

Table 1: bcodl return codes

Code	Error specifier	Short description
0	SUCCESS	Success, no error occured.
-1	ERR_ERROR	An unspecified error occured.
-2	ERR_INVALID_ARG	An invalid argument was passed.
-3	ERR_WRONG_API_VERSION	The DLL does not provide the expected API version.
-4	ERR_APP_ALREADY_RUNNING	Another instance of the utility is already running.

3.3 Commands

3.3.1 bcdinfo

```
bcodl bcdinfo <bcd-filename>
```

If the bcodl utility is executed with the `bcdinfo` command it prints some information about the bcd file specified by the `bcd-filename` argument. This argument is required and therefore the bcodl utility will return -2 if no file name was supplied at the command prompt. No further arguments are required and no additional options are useful for this command.

3.3.2 list

```
bcodl list
```

If the bcodl utility is executed with the `list` command it shows a list of the connected devices. This list provides onyl a few information about a device. For full information about a device refer to section 3.3.3. No arguments are required and no additional options are useful for this command.

3.3.3 info

```
bcodl info
```

If the bcodl utility is executed with the `info` command it shows detailed information about the device specified by the device number which where supplied with the `'-d'` option. If nothing was specified device 0 will be taken by default. No arguments are required and no additional options are useful for this command.

3.3.4 load

```
bcodl load <bcd-filename>
```

If the bcodl utility is executed with the `load` command it downloads the bcd file specified by the `bcd-filename` argument. This argument is required and therefore the bcodl utility will return -2 if no file name was supplied. By default, the application and the CnE image from the specified bcd file will be downloaded. You can change this behavior by supplying additional options. Useful options for this command are `'-mA'`, `'-mB'`, `'-mC'`, `'-mI'`, `'-tS'`, `'-tB'`, `'-tE'`, `'-tR'`. Please refer to section 3.4 for a description of these options.

3.3.5 setguid

```
bcodl setguid <b0,b1,b2,b3,b4,b5,b6,b7>
```

If the bcodl utility is executed with the `setguid` command it programs the device with the specified guid, which is supplied by a comma-separated list. This argument is required and therefore the bcodl utility will return -2 if no guid is given. The device can be specified by the `'-d'` option. If not specified device 0 will be taken by default. You can use the `'-tG'` option to change the timeout. Please refer to section 3.4 for further information about the options.

3.3.6 getguid

```
bcodl getguid
```

If the bcodl utility is executed with the `getguid` command it reads the guid from the device and prints it on the screen. The device can be specified by the `'-d'` option. If not specified device 0 will be taken by default. There are no further useful options for this command.

3.3.7 initpers

```
bcodl initpers
```

If the bcodl utility is executed with the `initpers` command it initializes the persistent parameters with bootloader built-in values (only supported by download protocol version 1 and). The device can be specified by the `'-d'` option. If not specified device 0 will be taken by default.Further useful option is the `'-tI'` option. Please refer to section 3.4 for a description of this options.

3.3.8 reset

```
bcodl reset
```

If the bcodl utility is executed with the `reset` command it resets the device. The device can be specified by the `'-d'` option. If not specified device 0 will be taken by default.Further useful options are the `'-r'` and the `'-tR'` option. Please refer to section 3.4 for a description of these options.

3.4 Options

Table 2: bcodl options

Option	Description	Used with command
-dN	Selects a device, N is the zero-based device index. Device 0 is selected by default.	any but bcdinfo and list
-mA	Download application image if present in the bcd file.	load
-mB	Download L2 bootloader if present in the bcd file.	load
-mC	Download CnE image if present in the bcd file.	load
-mI	Download application and CnE image if present in the bcd file. This is the default behavior if -m is not given.	load
-f	Force download, do not check compatibility of the bcd file.	load
-r0	Reset mode: start application. This is the default behavior if -r is not given.	reset
-r1	Reset mode: halt in boot loader.	reset
-tRN	Set the timeout for reset, N specifies the time in ms.	reset, load
-tSN	Set the timeout for download start, N specifies the time in ms	load
-tBN	Set the timeout for download block, N specifies the time in ms	load
-tEN	Set the timeout for download end, N specifies the time in ms.	load
-tGN	Set the timeout for programming the GUID, N specifies the time in ms.	setguid
-tIN	Set the timeout for initializing persistent parameters, N specifies the time in ms.	initpers
-v	Enabled verbose mode.	any
-llogfile	Enable logging of all DLL actions to the file specified by logfile.	any

Index