

Document	:	FlexCopFrontEnd
Author	:	M. Majoor
Subject	:	Front end DLL for FLEXCOP based cards

Revision/Date	Author	Description
20050416	M.Majoor	Initial draft

FlexCopFrontEnd.dll is a dynamic link library which can be used by application programmers to control the front end device (tuner) of the FLEXCOP based cards.

Since the full source code is included with the dll, programmers should be able to retrieve all relevant information from that. For those which just want to use the dll, without browsing through source code, this document gives the basic information on how to use it.

For most of the functions involved you should refer to the LINUX DVB API Version 3 manual which can be found at <http://www.linuxtv.org/>. Most functions are based on this API.

Basic information:

- All function/procedures in the dll should be accessed using the **stdcall** calling convention.
- When accessing the dll functions by name, be aware that these names are case sensitive!
- Result codes and IOCTL identifiers(*) are those as defined by the Linux DVB API.
- All pointer types require the caller to have allocated the memory for returning the result.

(*) IOCTL identifiers are actually variable by means of settings in the FlexCop.ini file.

Where type declarations are concerned the following applies:

INTEGER: 32 bit

PCHAR: Pointer to zero terminated string (character based)

OPEN (index 1, name 'Open')		
function Open(DeviceName, Flags)		
IN	PCHAR DeviceName	Name of specific device. Although the name itself is not used, the trailing number in the name denotes the card number to use. This number is zero based, so to address the first card the number '0' is used. A typical name (as Linux uses) is 'frontend0'.
	INTEGER Flags	Not used
OUT	INTEGER	File descriptor (handle) or error code

CLOSE (index 2, name 'Close')		
function Close(FileDescriptor)		
IN	INTEGER FileDescriptor	File descriptor (handle) as acquired by a previous OPEN call.
OUT	INTEGER	Result code

IOCTL (index 3, name 'IoCtl')			
function IoCtl(FileDescriptor, Request, Parameter)			
IN	INTEGER	FileDescriptor	File descriptor (handle) as acquired by a previous OPEN call.
	INTEGER	Request	IOCTL identifier The actual link between an IOCTL identifier and the function to call is set in the FlexCop.ini file (more on this later).
	POINTER	Parameter	Parameter(s) associated with the <i>Request</i> . In most cases a pointer to the parameter, but some functions pass the value instead. Refer to the LINUX DVB API manual for the parameters associated with the different requests.
OUT	INTEGER		Result code

The following IOCTL calls have been implemented (but are not necessarily 'supported'):

```
FE_GET_INFO
FE_DISEQC_RESET_OVERLOAD
FE_DISEQC_SEND_MASTER_CMD
FE_DISEQC_RECV_SLAVE_REPLY
FE_DISEQC_SEND_BURST
FE_SET_TONE
FE_SET_VOLTAGE
FE_ENABLE_HIGH_LNB_VOLTAGE
FE_READ_STATUS
FE_READ_BER
FE_READ_SIGNAL_STRENGTH
FE_READ_SNR
FE_READ_UNCORRECTED_BLOCKS
FE_SET_FRONTEND
FE_GET_FRONTEND
FE_GET_EVENT
```

There are some additional functions not covered by the Linux API itself. These functions are:

```
FE_DISEQC_SEND_BURST2
FE_SET_TONE2
FE_SET_VOLTAGE2
FE_ENABLE_HIGH_LNB_VOLTAGE2
```

The only difference with their associated functions (eg. the function without the trailing '2'), is that these functions allow the parameter they require to be passed as pointer instead as by value.

Example: The function `FE_DISEQC_SEND_BURST` has as its parameter the direct value of the type of burst to set. The function `FE_DISEQC_SEND_BURST2` uses the parameter as being a pointer to that value.

In the FlexCop.ini file the actual IOCTL identifiers the dll responds to are set. This means that an application can use its own typical identifiers to call specific functions. When for instance an application wants to call the `FE_GET_INFO` function, then it should use the identifier as being set in the FlexCop.ini file. When the .ini file is set as follows:

```
[IOctl]
FE_GET_INFO=10
```

then the application should use '10' as IOCTL identifier (Request).

Thus:

```
Ioctl(FileDescriptor, 10, ParameterPointer)
```

will call the `FE_GET_INFO` function.

The dll uses the FlexCop.ini file for some of it's settings. These are the actual IOCTL identifiers the dll uses and what kind of hardware to control. Other settings control the way a log-file is generated (if any).

Here below is a typical FlexCop.ini file. The IOCTL identifiers here are those as used by Linux:

```
[Ioctl]
; This is the list of Ioctl identifiers used. These identifiers
; should be identical to the identifiers used by the application
; calling the DLL.
; Note: The 'xxxxx2' definitions are identical to the original
;       definitions except that they accept a pointer to the value
;       instead of the value itself.
FE_GET_INFO=                $80A86F3D
FE_DISEQC_RESET_OVERLOAD=   $00006F3E
FE_DISEQC_SEND_MASTER_CMD=  $40076F3F
FE_DISEQC_RECV_SLAVE_REPLY= $800C6F40
FE_DISEQC_SEND_BURST=       $00006F41
FE_DISEQC_SEND_BURST2=      $00006F42
FE_SET_TONE=                $00006F43
FE_SET_TONE2=               $00006F43
FE_SET_VOLTAGE=             $00006F43
FE_SET_VOLTAGE2=            $00006F43
FE_ENABLE_HIGH_LNB_VOLTAGE= $00006F44
FE_ENABLE_HIGH_LNB_VOLTAGE2= $00006F44
FE_READ_STATUS=             $80046F45
FE_READ_BER=                $80046F46
FE_READ_SIGNAL_STRENGTH=    $80026F47
FE_READ_SNR=                $80026F48
FE_READ_UNCORRECTED_BLOCKS= $80046F49
FE_SET_FRONTEND=            $40246F4C
FE_GET_FRONTEND=            $80246F4D
FE_GET_EVENT=               $80286F4E

[FrontEnd]
; LogLevel
; The log file 'FlexCopFrontEnd.log' is used for logging data
; were the type of messages send to this log file is set by this level
; 0: No log (debug) messages to log file
; 1: -
; 2: Errors
; 3: Main procedures/functions
; 4: Sub procedures/functions
; 5: Intermediate messages
; Note: Only valid if the DLL supports it.
LogLevel=0

; LogClear
; 'Yes' clears the log (only when logging active)
; 'No' appends messages to the log
LogClear=Yes
```