Feasa tools DLL Library

for 32 & 64 bit

reference guide

Table of Contents

1. Introduction

Feasa tools DLL is a general purpose library developed by Feasa Enterprises Ltd. This library is intended to be a complement for the FeasaCom DLL, containing functions to parse the values obtained from the reading of the different Feasa devices and to provide results from the measurements formatted into standard data types.

2. Previous considerations

To be able to use this DLL, you should ensure that the used programming language is able to work with DLL libraries.

Remember that DLLs should be generally copied to the windows/System32 folder. In some platforms or Windows versions could be necessary to register the library before use it (you can learn how to do it in the next section). Some compilers also accept or need the DLL to be placed in the same path of the EXE file.

You can notice that there are two versions of the DLL a 32bit (feasa_tools.dll) and a 64bit version (feasa_tools64.dll); so you can get error messages or have communication problems if you are trying to use a wrong version of the DLL or your compiler is not targeting the right platform for the EXE file. Almost all compilers allow to select the target, so you should set it to x86, win32, or equivalent for the 32bit DLL and Wow64, x64 or equivalent for the 64bit version.

Note that all programming examples included in the CD are targeted to 32bit platforms, so they use the 32bit version of the DLL (feasacom.dll).

The functions of this DLL follow the STDCALL calling convention. This should not be an issue since any modern programming language allow to select the calling convention when importing a DLL function; others have specific importing functions depending on the calling convention.

You should be careful when declaring the data types of the parameters passed on to the functions, since they have to perfectly match with the ones described in the original headers. There are equivalences for all the variable types in all most used programming language. If you use a wrong type of variable, unexpected results can be obtained, even cause the computer to lock up.

Depending on the programming language used, Integer variables does not match in size with the Integer described in the DLL (32 bit). Please, check your language's reference manual in order to be sure that the proper data types are being used.

This DLL has been written for the Windows OS and tested on Win7, 8, 8.1 and 10 versions, so Feasa cannot ensure that this DLL will work on different conditions than those described.

Important: some functions header have changed. Please, ensure that all parameters declaration match with the one defined in your code.

Important: the DLL name has changed from “feasala_tools.dll” to “feasa_tools.dll”

3. Installation instructions

Before use the DLL you should previously copy it to the windows/System32 folder, which can be generally found inside the windows folder.

Some compilers and programming languages allow to use the DLL if you previously copy it to the same folder of the EXE file or programming project where you are calling it from, so in this situations there is no need to copy the DLL to the system folder.

Arrived at this point, if your program is not able to find the DLL or you can't communicate with it (shouldn't happen), you may need to register it on windows (depending on the OS version and Language used), so if you experience problems (...”DLL could not be loaded”..), you can try to register the library, executing this command or equivalent:

```
regsvr32 “[system32_path]/feasa_tools.dll”
```

If you are planning to redistribute your application or you are installing it on other computers, you should repeat this installation/set-up process in all the target computers.

4. Functions

4.1. Parse 16bit integer

This is a generic parsing function used to convert a part from a passed string into integer. The received string is split into "Parameters", which are a set of sub-strings separated by the character *space*, and the specified parameter is extracted and converted to Integer-16 bit.

```
__int16 Feasa_Parse_Int16(const char * StringToParse, char Parameter);
```

Parameters:

- StringToParse: string containing the set of parameters to be extracted
- Parameter: the number of parameter to be extracted from the StringToParse starting by 1

Example:

```
mydata = Feasa_Parse_Int16("001 252 002 00243", 2);
```

The result from the execution of this command will be the number 252, which will be stored in the variable "mydata".

4.2 Parse 32bit integer

This is a generic parsing function used to convert a part from a passed string into integer. The received string is split into "Parameters", which are a set of sub-strings separated by the character *space*, and the specified parameter is extracted and converted to Integer-32 bit.

```
int Feasa_Parse_Int32(const char * StringToParse, char Parameter);
```

Parameters:

- StringToParse: string containing the set of parameters to be extracted
- Parameter: the number of parameter to be extracted from the StringToParse starting by 1

This function returns the extracted parameter in a 16 bit integer format. 0 is returned in case of error.

Example:

```
mydata = Feasa_Parse_Int32("001 252 002 00243", 4);
```

The result from the execution of this command will be the number 243, which will be stored in the variable "mydata".

4.3 Parse float

This is a generic parsing function used to convert a part from a passed string into float. The received string is split into "Parameters", which are a set of sub-strings separated by the character *space*, and

the specified parameter is extracted and converted to 4-byte Floating point.

```
float Feasa_Parse_Float(const char * StringToParse, char Parameter);
```

Parameters:

- StringToParse: string containing the set of parameters to be extracted
- Parameter: the number of parameter to be extracted from the StringToParse starting by 1

This function returns the extracted parameter in a 16 bit integer format. 0 is returned in case of error.

Example:

```
mydata = Feasa_Parse_Float("0.3244 0.3892", 1);
```

The result from the execution of this command will be the number 0.3244, which will be stored in the variable "mydata".

4.4 Parse RGBI

This function is used to retrieve the numerical values of RGB and Intensity from the data string obtained from the LED Analyser as a result of executing the commandfrom the data string obtained from the LED Analyser as a result of executing the command GETRGBI.

```
int Feasa_Parse_RGBI(const char * AnalyserResponse, unsigned char * Red,
    unsigned char * Green, unsigned char * Blue, int * Intensity);
```

Parameters:

- AnalyserResponse: string which was obtained from the response of the Feasa LED Analyser.
- Red: numerical value of Red
- Green: numerical value of Green
- Blue: numerical value of Blue
- Intensity: numerical value of Intensity

This function returns a value of 1 if the parsing was successful and 0 otherwise.

4.5 Parse RGBI-ALL

This function is used to retrieve the numerical values of all RGB and Intensity values from the data string obtained from the LED Analyser as a result of executing the command GETRGBIALL.

```
int Feasa_Parse_RGBI_All(const char * AnalyserResponse, unsigned char *
RedValues, unsigned char * GreenValues, unsigned char * BlueValues, int *
                                IntensityValues);
```

Parameters:

- AnalyserResponse: string which was obtained from the response of the Feasa LED Analyser

- RedValues: array of Red values. Value in position 0 belongs to fiber/channel 1, while value in position n-1 belongs to fiber/channel n
- GreenValues: array of Green values. Value in position 0 belongs to fiber/channel 1, while value in position n-1 belongs to fiber/channel n
- BlueValues: array of Blue values. Value in position 0 belongs to fiber/channel 1, while value in position n-1 belongs to fiber/channel n
- IntensityValues: array of Intensity values. Value in position 0 belongs to fiber/channel 1, while value in position n-1 belongs to fiber/channel n

This function returns an integer value indicating the number of fibers/lines of data parsed.

4.6 Parse HSI

This function is used to retrieve the numerical values of Hue, Saturation and Intensity from the data string obtained from the LED Analyser as a result of executing the command GETHSI.

```
int Feasa_Parse_HSI(const char * AnalyserResponse, float * Hue, int *
                    Saturation, int * Intensity);
```

Parameters:

- AnalyserResponse: string which was obtained from the response of the Feasa LED Analyser.
- Hue: numerical value of Hue
- Saturation: numerical value of Saturation
- Intensity: numerical value of Intensity

This function returns a value of 1 if the parsing was successful and 0 otherwise.

4.7 Parse HSI-ALL

This function is used to retrieve the numerical values of all Hue, Saturation and Intensity from the data string obtained from the LED Analyser as a result of executing the command GETHSIALL.

```
int Feasa_Parse_HSI_All(const char * AnalyserResponse, float * HueValues,
             int * SaturationValues, int * IntensityValues);
```

Parameters:

- AnalyserResponse: string which was obtained from the response of the Feasa LED Analyser
- Hue: array of Hue values. Value in position 0 belongs to fiber/channel 1, while value in position n-1 belongs to fiber/channel n
- Saturation: array of Saturation values. Value in position 0 belongs to fiber/channel 1, while value in position n-1 belongs to fiber/channel n
- Intensity: array of Intensity values. Value in position 0 belongs to fiber/channel 1, while

value in position n-1 belongs to fiber/channel n

This function returns an integer value indicating the number of fibers/lines of data parsed.

4.8 Parse xy

This function is used to retrieve the numerical values of chromaticity from the data string obtained from the LED Analyser as a result of executing the command GETxy.

```
int Feasa_Parse_xy(const char * AnalyserResponse, float * x, float * y);
```

Parameters:

- AnalyserResponse: string which was obtained from the response of the Feasa LED Analyser.
- x: numerical value of x coordinate
- y: numerical value of y coordinate

This function returns a value of 1 if the parsing was successful and 0 otherwise.

4.9 Parse xy-ALL

This function is used to retrieve the numerical values of chromaticity from the data string obtained from the LED Analyser as a result of executing the command GETxyALL.

```
int Feasa_Parse_xy_All(const char * AnalyserResponse, float * xValues,
                       float * yValues);
```

Parameters:

- AnalyserResponse: string which was obtained from the response of the Feasa LED Analyser
- x: array of x values. Value in position 0 belongs to fiber/channel 1, while value in position n-1 belongs to fiber/channel n
- y: array of y values. Value in position 0 belongs to fiber/channel 1, while value in position n-1 belongs to fiber/channel n

This function returns an integer value indicating the number of fibers/lines of data parsed.

4.10 Parse uv

This function is used to retrieve the numerical values of u' and v' from the data string obtained from the LED Analyser as a result of executing the command GETuv.

```
int Feasa_Parse_uv(const char * AnalyserResponse, float * u, float * v);
```

Parameters:

- AnalyserResponse: string which was obtained from the response of the Feasa LED Analyser.

- u: here is returned the u' value which is extracted from the LED Analyser response. 4-byte Float type pointer.
- v: here is returned the v' value which is extracted from the LED Analyser response. 4-byte Float type pointer.

This function returns a value of 1 if the parsing was successful and 0 otherwise.

4.11 Parse uv-ALL

This function is used to retrieve the numerical values of u' and v' from the data string obtained from the LED Analyser as a result of executing the command GETuvALL.

```
int Feasa_Parse_uv_All(const char * AnalyserResponse, float * uValues,
                       float * vValues);
```

Parameters:

- AnalyserResponse: string which was obtained from the response of the Feasa LED Analyser
- u: array of u' values. Value in position 0 belongs to fiber/channel 1, while value in position n-1 belongs to fiber/channel n
- v: array of v' values. Value in position 0 belongs to fiber/channel 1, while value in position n-1 belongs to fiber/channel n

This function returns an integer value indicating the number of fibers/lines of data parsed.

4.12 Parse CCT

This function is used to retrieve the numerical values of Correlated Color Temperature and Delta u'v' distance from the data string obtained from the LED Analyser as a result of executing the command GETCCT.

```
int Feasa_Parse_CCT(const char * AnalyserResponse, int * CCT, float *
                    delta);
```

Parameters:

- AnalyserResponse: string which was obtained from the response of the Feasa LED Analyser
- CCT: numerical value of CCT
- delta: numerical value of delta u'v' distance

This function returns a value of 1 if the parsing was successful and 0 otherwise.

4.13 Parse CCT-ALL

This function is used to retrieve the numerical values of Correlated Color Temperature and Delta

u'v' distance from the data string obtained from the LED Analyser as a result of executing the command GETCCTALL.

```
int Feasa_Parse_CCT_All(const char * AnalyserResponse, int * CCTValues,
                        float * deltaValues);
```

Parameters:

- AnalyserResponse: string which was obtained from the response of the Feasa LED Analyser
- CCT: array of CCT values. Value in position 0 belongs to fiber/channel 1, while value in position n-1 belongs to fiber/channel n
- delta: array of delta u'v' values. Value in position 0 belongs to fiber/channel 1, while value in position n-1 belongs to fiber/channel n

This function returns an integer value indicating the number of fibers/lines of data parsed.

4.14 Parse Wavelength

This function is used to retrieve the numerical value of Wavelength from the data string obtained from the LED Analyser as a result of executing the command GETWAVELENGTH.

```
int Feasa_Parse_Wavelength(const char * AnalyserResponse, float *
                           Wavelength);
```

Parameters:

- AnalyserResponse: string which was obtained from the response of the Feasa LED Analyser
- Wavelength: numerical value of Wavelength

This function returns a value of 1 if the parsing was successful and 0 otherwise.

4.15 Parse Wavelength-ALL

This function is used to retrieve the numerical value of Wavelength from the data string obtained from the LED Analyser as a result of executing the command GETWAVELENGTHALL.

```
int Feasa_Parse_Wavelength_All(const char * AnalyserResponse, float *
                               WavelengthValues);
```

Parameters:

- AnalyserResponse: string which was obtained from the response of the Feasa LED Analyser
- Wavelength: array of Wavelength values. Value in position 0 belongs to fiber/channel 1, while value in position n-1 belongs to fiber/channel n

This function returns an integer value indicating the number of fibers/lines of data parsed.

4.16 Parse Wavelength & Intensity (WI)

This function is used to retrieve the numerical values of Wavelength and Intensity from the data string obtained from the LED Analyser as a result of executing the command GETWI.

```
int Feasa_Parse_WI(const char * AnalyserResponse, float * Wavelength, int
                                * Intensity);
```

Parameters:

- AnalyserResponse: string which was obtained from the response of the Feasa LED Analyser
- Wavelentgh: numerical value of Wavelentgh
- Intensity: numerical value of Intensity

This function returns a value of 1 if the parsing was successful and 0 otherwise.

4.17 Parse WI-ALL

This function is used to retrieve the numerical values of Wavelength and Intensity from the data string obtained from the LED Analyser as a result of executing the command GETWIALL.

```
int Feasa_Parse_WI_All(const char * AnalyserResponse, float *
          WavelengthValues, int * IntensityValues);
```

Parameters:

- AnalyserResponse: string which was obtained from the response of the Feasa LED Analyser
- Wavelentgh: array of Wavelength values. Value in position 0 belongs to fiber/channel 1, while value in position n-1 belongs to fiber/channel n
- Intensity: array of Intensity values. Value in position 0 belongs to fiber/channel 1, while value in position n-1 belongs to fiber/channel n

This function returns an integer value indicating the number of fibers/lines of data parsed.

4.18 Parse Intensity

This function is used to retrieve the numerical values of Intensity from the data string obtained from the LED Analyser as a result of executing the command GETINTENSITY.

```
int Feasa_Parse_Intensity(const char * AnalyserResponse, int *
                            Intensity);
```

Parameters:

- AnalyserResponse: string which was obtained from the response of the Feasa LED Analyser
- Intensity: numerical value of Intensity

This function returns a value of 1 if the parsing was successful and 0 otherwise.

4.19 Parse Intensity-All

This function is used to retrieve the numerical values of Intensity from the data string obtained from the LED Analyser as a result of executing the command GETINTENSITY ALL.

```
int Feasa_Parse_Intensity_All(const char * AnalyserResponse, int *
                        IntensityValues);
```

Parameters:

- AnalyserResponse: string which was obtained from the response of the Feasa LED Analyser
- Intensity: array of Intensity values. Value in position 0 belongs to fiber/channel 1, while value in position n-1 belongs to fiber/channel n

This function returns an integer value indicating the number of fibers/lines of data parsed.

4.20 Parse Spectrum

This function is used to retrieve the numerical values of Wavelength and Intensity from the data string obtained from the LED Spectrometer as a result of executing the command GETSPD.

```
int Feasa_Parse_Spectrum(const char * AnalyserResponse, float *
          WavelengthValues, double * IntensityValues);
```

Parameters:

- AnalyserResponse: string which was obtained from the response of the Feasa LED Analyser
- Wavelength: array of Wavelength values. Value in position 0 belongs to fiber/channel 1, while value in position n-1 belongs to fiber/channel n
- Intensity: array of Intensity values. Value in position 0 belongs to fiber/channel 1, while value in position n-1 belongs to fiber/channel n

This function returns an integer value indicating the number of lines of wavelength points/data parsed.

5. Examples

You can find several programming examples demonstrating how to use the Feasa Communications DLL on the CD. These examples are written on different languages like C#, Visual Basic dot NET, Pascal, Python, LabVIEW or CVI and they can be found in separate folders, depending on the level of difficulty of each exercise, from 0_ to x_, where 0_ is the simplest example.

Note: not all the examples and programming languages are available on all the CD's, so please check your device's CD to find out more.

Function Index