Manual USB Terminal Element

WERMA Signaltechnik GmbH + Co. KG

Manual for USB Terminal Element KombiSIGN 71 and KOMPAKT 71 with USB Interface

Version 1.2.0 – 2011





Contents

Contents **3**
1. Introduction **4**
2. Technical information **5**
2.1 Assembly 5
2.1.1 Terminal Element with USB interface (640.840.00) 5
2.1.2 Kompakt 71 with USB interface (697.430.53) 5
2.2 System requirements 5
2.3 Safety instructions 6
2.4 Technical data 6
3. Automatic Driver Installation **7**
4. Manual driver installation **7**
5. Operation with COM Port **10**
5.1 Settings 10
5.2 Commands 10
5.2.1 Version 10
5.2.2 Name 10
5.2.3 Write 11
5.2.4 Read 12
5.3 HyperTerminal 12
6. Operation with Dynamic Link Library (DLL) **15**
6.1 Introduction 15
6.2 Related files and executables 15
Runtime files: 15
6.3 DLL Functions 15
6.3.1 Function Overview 15
6.3.2 Basic Functions 16
6.3.2.1 SetUsbDevice 16
6.3.2.2 SetDeviceStatus 16
6.3.2.3 GetDeviceStatus 17
6.3.2.4 GetDeviceSerial 17
6.3.2.5 GetDeviceLocation 18
6.3.2.6 GetFirmwareVersion 18
6.3.2.7 GetLibVersion 19
6.3.3 Further Functions 19
6.3.3.1 SetIOStatus 19
6.3.3.2 SetSingleIO 20
6.3.3.3 GetIOStatus 20
6.3.3.4 GetSingleIO 21
7. USB Terminal Element Demo **22**

1. Introduction

Many applications require faults or the operating status to be displayed by an optical signal. As a PLC or machine control system is not always available, PCs are frequently used for system control and monitoring.

The ideal solution here is to use the terminal element with USB interface, which is available for both signal tower series Kombi*SIGN* 71 and Kompakt 71. Developed in-house at WERMA, this innovative product allows direct PC actuation for fast and convenient commissioning. As well as actuation via a DLL (Dynamic Link Library), the element can also be simply commissioned by means of VCP (Virtual COM Port) actuation. This ensures for simple integration into any customer-specific software.

Up to five signal towers with a maximum of five signalling elements each can be connected per PC. An optimized electronic circuit allows each of the elements to be triggered either as a permanent or blinking light. This means that up to ten different operating statuses can be signalled using only a single signalling tower. The terminal element with USB interface requires neither its own separate power supply nor any additional hardware.

WERMA Signaltechnik GmbH + Co.KG
D-78604 Rietheim-Weilheim
Fon: +49 (0)7424 / 9557-0
Fax: +49 (0)7424 / 9557-44
info@werma.com
www.werma.com

As of: 09/2011

310.640.08

2. Technical information

2.1 Assembly

2.1.1 Terminal Element with USB interface (640.840.00)

Description	Quantity
USB terminal element	1
USB cable 1.8 m	1
Software CD	1
Instruction leaflet	1

2.1.2 Kompakt 71 with USB interface (697.430.53)

Description	Quantity
Pre-assembled tower (red/yellow/green) with USB terminal element	1
USB cable 1.8 m	1
Software CD	1
Instruction leaflet	1

2.2 System requirements

- 500 MHz Pentium II processor or higher
- 256 MB RAM (512 MB is recommended)
- 150 MB free hard disk space
- CD/DVD drive
- USB port
- Windows 2000 service pack 4, Windows XP, Windows Vista or Windows 7
- Windows Server 2000 service Pack 4, Windows Server 2003, Windows Server 2008, Windows Server 2008 R2
- Windows Installer 3.1

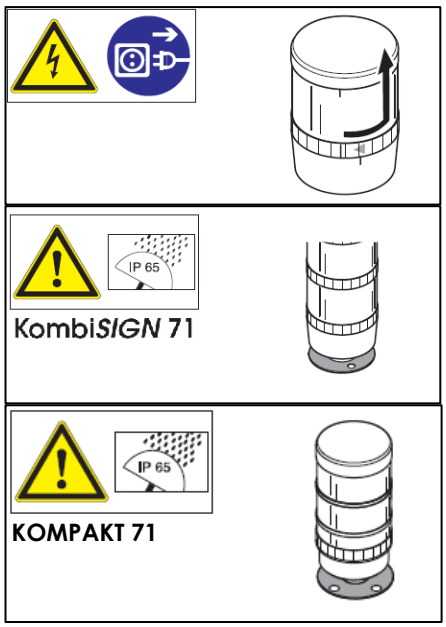
2.3 Safety instructions

Caution!

- Please also refer to the instruction leaflet for your WERMA signal tower.
- The USB terminal element is not suitable for safety critical applications.
- For use with 24 V WERMA signal elements with max. 90 mA.



(D) Anschluss **ausschließlich** durch ausgebildete Elektro-Fachkräfte.
(GB) Electrical connection is to be made by trained electrical specialists **only**.
(F) Le branchement doit **uniquement** être effectué par des professionnels.
(I) Il collegamento deve essere eseguito **solo** da elettricisti specializzati.
(E) La conexión **sólo** debe ser realizada por electricistas debidamente formados.
(PT) A ligação deve ser feita **exclusivamente** por profissional eléctrico especializado.
(CZ) Připojení smí **provádět** pouze kvalifikovaný personál.
(PL) Podłączenie **wyłącznie** przez specjalistów-elektryków.
(FI) Liittäminen kuuluu **ainoastaan** koulutettujen sähköalan ammattilaisten tehtäviin.
(RUS) Подключение проводится только специалистом-электриком.
(TR) Bağlantı **sadece** eğitimli elektrik teknisyenleri tarafından yapılmalıdır.
(CN) 布线需由专业电工执行



2.4 Technical data

Operating voltage	5 V DC (USB)
Output voltage	24 V DC (+/- 10 %)
Current consumption	Typ. 500 mA
Output current	Max. 90 mA
Min. operating temperature	- 25 °C
Typ. operating temperature	+ 25 °C
Max. operating temperature	+ 60 °C

The signal tower elements can be triggered directly from a PC or Laptop via a standard USB interface. The USB port must deliver min. 500 mA, if not a USB-Hub with external power supply must be used.

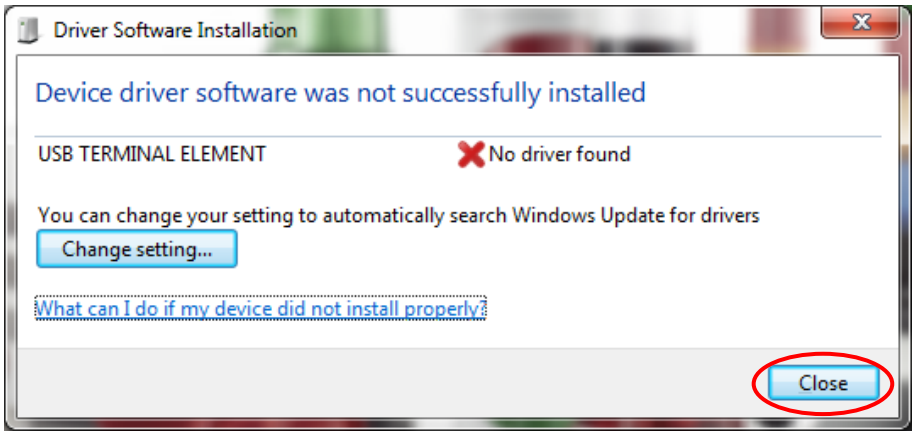
3. Automatic Driver Installation

Insert the USB terminal element CD and install the file Setup_Win_2k_XP_Vista_7. This setup installs the drivers for Windows 2000, Windows XP, Windows Vista and Windows 7.

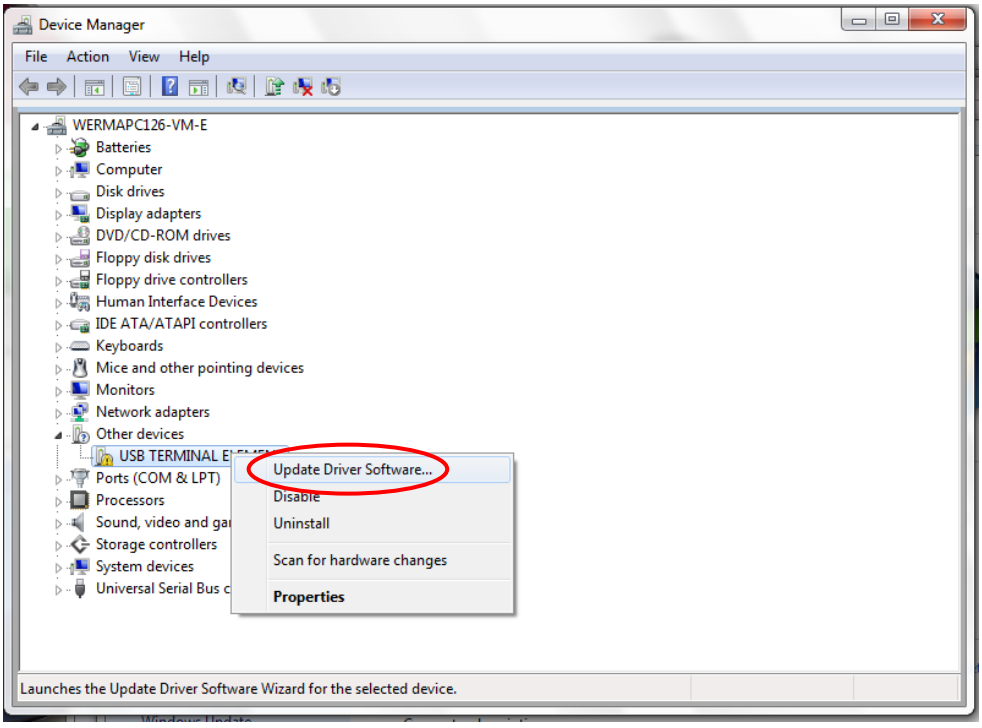
Note: You must have administrator rights to install the drivers.

4. Manual driver installation

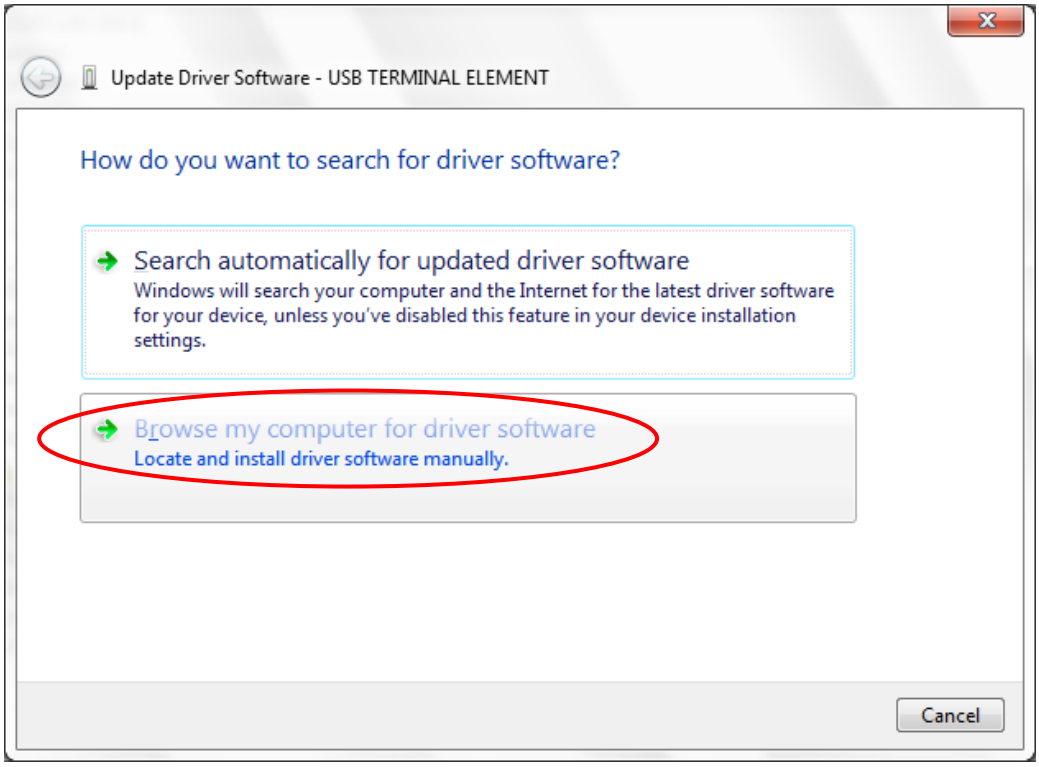
Note: Manual driver installation is only required if the USB terminal element is not recognised. Windows will open the following window.



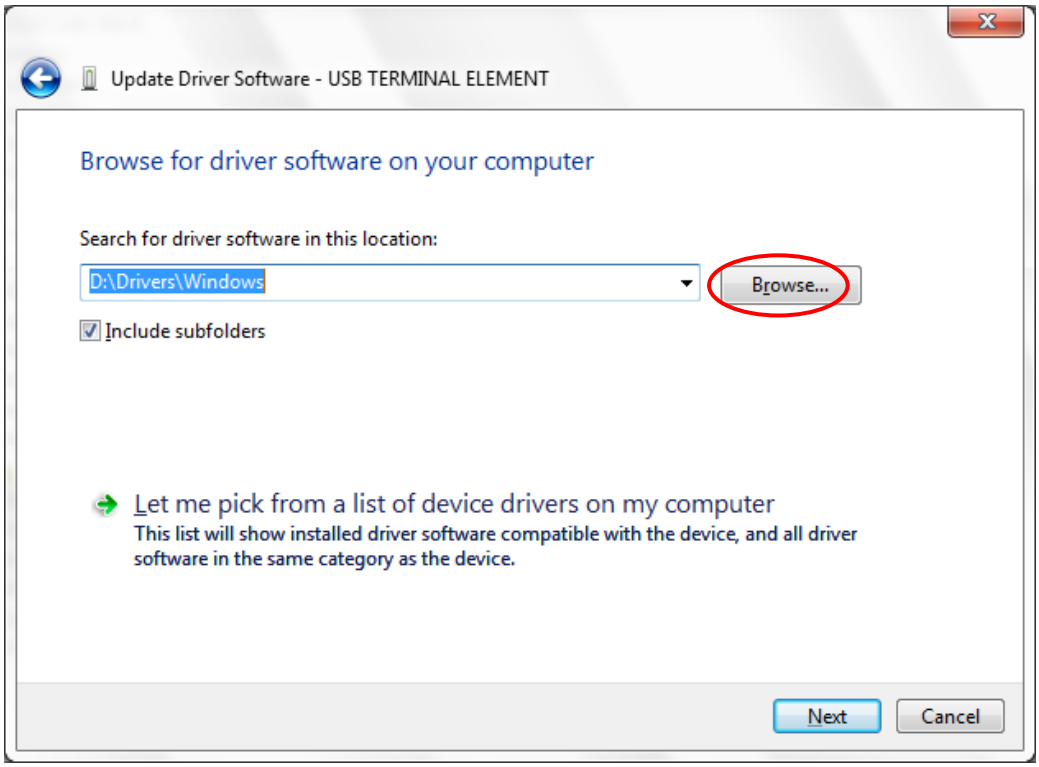
Close this window and open the Device Manager (Computer/Properties/Device Manager).



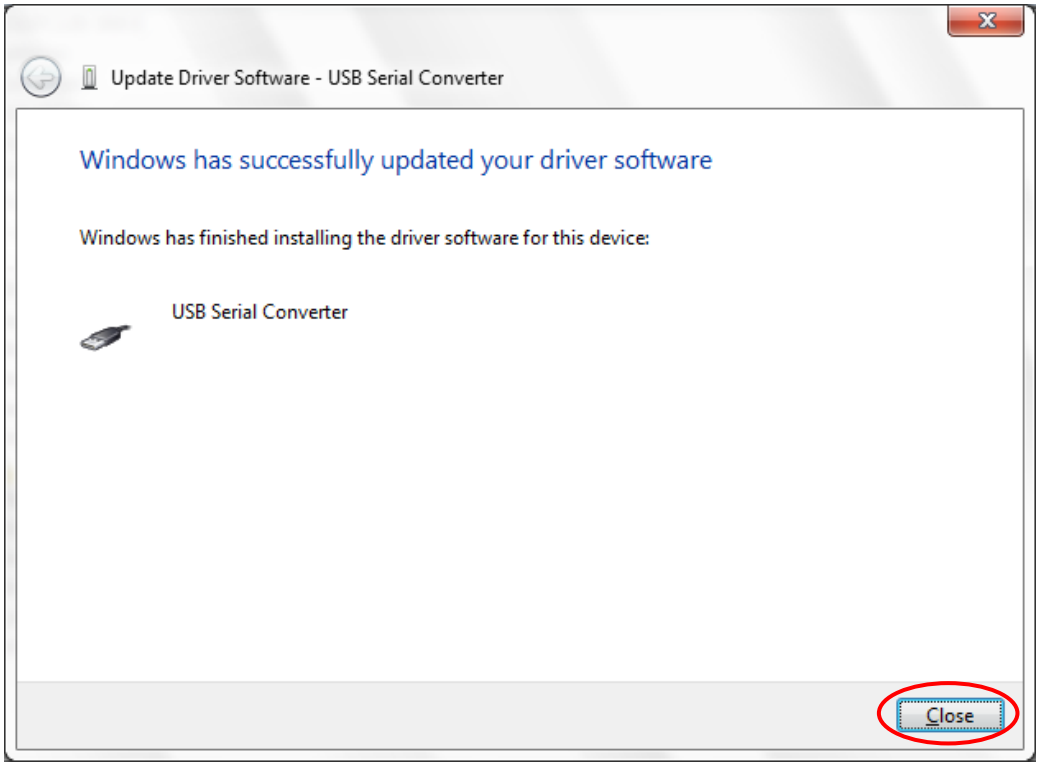
Click with the right mouse button on USB TERMINAL ELEMENNT and then click on "Update Driver Software".



Select "Brows my computer for driver software" and insert the CD of the USB terminal element into your CD-ROM drive.



Specify the location of the driver by clicking "Browse". Select the "Driver" folder in the CD-ROM drive and confirm by clicking "Next".



The driver for the USB terminal element has been successfully installed. Click "Close" to finish the update.

Note: The USB terminal element is now ready for operation.

5. Operation with COM Port

5.1 Settings

For the COM port you need the following settings:

Baud rate	9600 bits per second
Data bits	8
Stop bit	1
Parity	None
Flow control	No

5.2 Commands

Note: Every command has to end with a return or line feed.

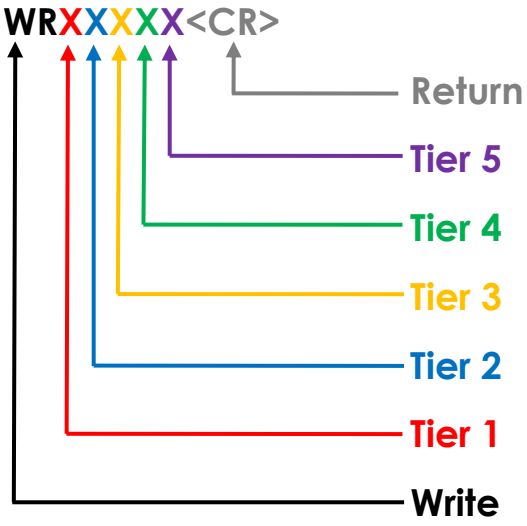
5.2.1 Version

Type	Command	Description
Version	V	Information about the version.  Example: V<CR> → Version:1.2.0

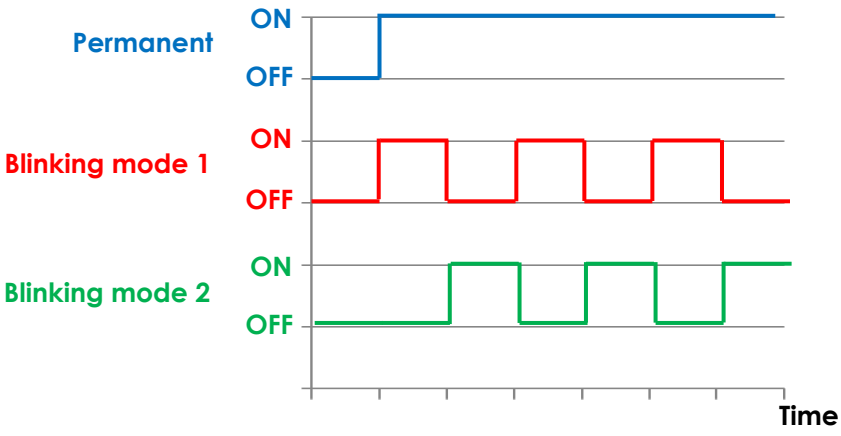
5.2.2 Name

Type	Command	Description
Name	N	Information about the name.  Example: N<CR> → USB TERMINAL ELEMENT

5.2.3 Write

Type	Command	Description
Write	WRXXXXX	Sets the output, X could be: 0 → Off 1 → On 2 → Blinking mode 1 3 → Blinking mode 2 X → Don't care, no change on that tier  Example: WR120X3<CR> → OK (Tier 1: On; Tier 2: Blinking mode 1; Tier 3: Off; Tier 4: Don't care; Tier 5: Blinking mode 2)

Note: Blinking mode1 and Blinking mode 2 differ by blinking alternately (see diagram).



5.2.4 Read

Type	Command	Description
Read	RD	Information of current status. **RD**<CR> Return **Information of status** Example: RD<CR> → 02131 (Tier 1: Off; Tier 2: Blinking mode 1; Tier 3: On; Tier 4: Blinking mode 2; Tier 5: On)

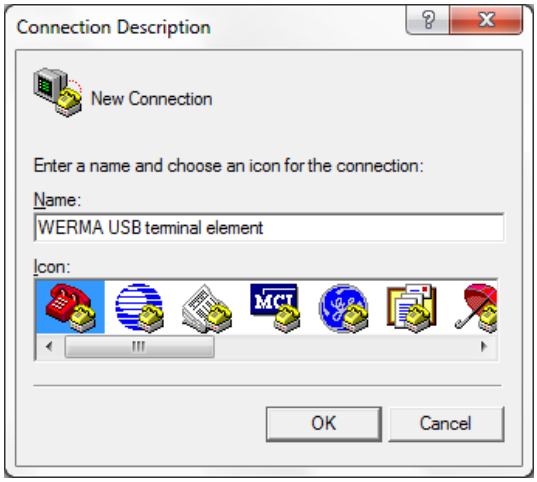
5.3 HyperTerminal

Windows HyperTerminal or a comparable Linux application shows how the USB terminal element can be triggered. Using the same settings, the USB terminal element can be triggered from other applications.

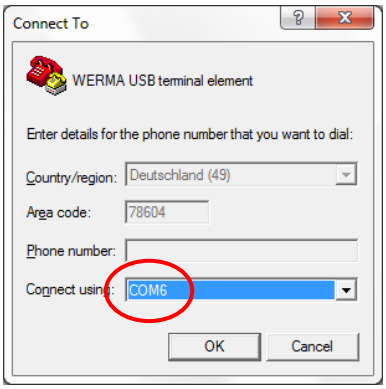
Note: As of Windows Vista, HyperTerminal is no longer included. But you can download this free from the Internet or use another terminal program.

Example with HyperTerminal:

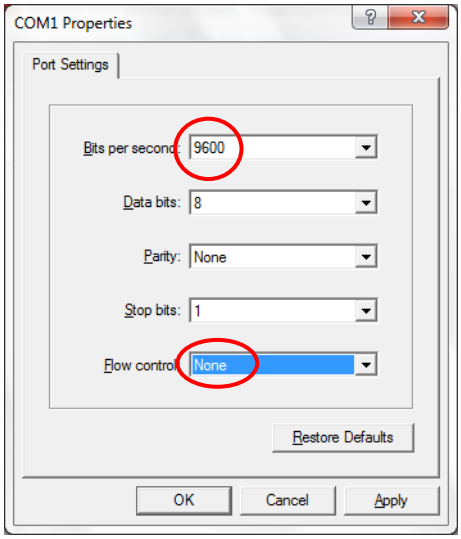
1. Connect the USB terminal element to a USB port on your computer.
2. Start the HyperTerminal: Start/Programs/Accessories/Communication/HyperTerminal
3. Enter a name in the first window (e.g. WERMA USB terminal element)



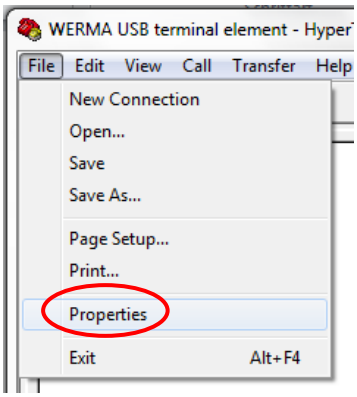
4. Now select the corresponding port (e.g. COM6)

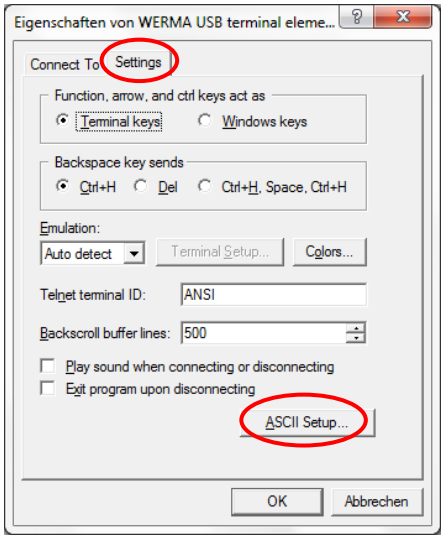


5. Set the bits per second in the following window to 9600 and flow control to none.

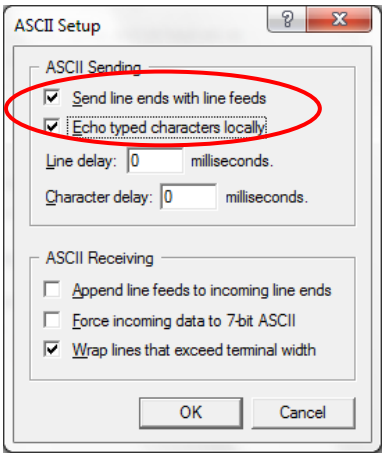


6. Open the Properties File/Properties and the register Settings and press the button "ASCII Setup".





7. Activate the line feeds and local echo Echo typed characters locally and confirm the window with OK.



You can save these settings on your PC in order to retrieve them as required. Now you can pass the commands to the USB terminal element.

Note the following points:

- Make sure it is spelt correctly / correct case.
- In the event of false entries press the Return button and start again.
- It is not possible to make corrections during the entry.

6. Operation with Dynamic Link Library (DLL)

6.1 Introduction

The USB terminal element provides connectivity for a signal tower to the PC via USB.
The user controls the USB module by using a specified dynamic link library (DLL) file found on the CD provided with the USB terminal element. The DLL contains several functions to set and retrieve the light module status.
The control over the light module is generally carried out by a customer application. The control over the module from the user application is achieved via a specific DLL which shares all the commonly needed functions. The DLL library is a standard windows functions library which can be easily used with the most common application development tools such as Visual Basic, Visual C++ etc.

6.2 Related files and executables

Runtime files:
The DLL Library should reside in a file located in the same folder with user project .EXE file. This file communicates with the USB module.

slma.dll Contains functions which provide control over the USB module

6.3 DLL Functions

6.3.1 Function Overview

Basic Functions

SetUsbDevice	**Returns** the **number of devices** currently found in the system
SetDeviceStatus[2]	**Sets** the status (ON, OFF, Blinking mode 1, Blinking mode 2) for **all Outputs**
GetDeviceStatus[2]	**Gets** the current Output status (ON, OFF, Blinking mode 1, Blinking mode 2) for **all Outputs**
GetDeviceSerial	**Gets** the specified device **serial code**
GetDeviceLocation[1]	**Gets** the **Location ID** of the USB port, where the device is connected
GetFirmwareVersion	**Gets** the specified device **firmware version**
GetLibVersion	**Gets** the currently used **library version**

[1] Please note that Linux, Mac OS X and Windows CE do not support location IDs.
[2] Please note that these two functions cover the further functions (see next page).

Further Functions

SetIOStatus	**Sets** the status (ON or OFF) for **all Outputs**
SetSingleIO	**Sets** the status (ON, OFF, Blinking mode 1, Blinking mode 2) of a **single Output**
GetIOStatus	**Gets** the current Output status (ON or OFF) for **all Outputs**
GetSingleIO	**Gets** the status (ON, OFF, Blinking mode 1, Blinking mode 2) of a specified **single Output**

6.3.2 Basic Functions

6.3.2.1 SetUsbDevice

int SetUsbDevice()

SetUsbDevice retrieves the devices number currently found in the system.

Parameters

None

Return Value

The amount of devices currently found in the system. When no devices are found, zero will be returned.

6.3.2.2 SetDeviceStatus

int SetDeviceStatus ([in] int DevIdx, [in] char Pos1, [in] char Pos2, [in] char Pos3, [in] char Pos4, [in] char Pos5)

SetDeviceStatus sets the status for all outputs.

Parameters

DevIdx
Index to the desire device currently available in the system. The first index is 1.

Pos1, Pos2, Pos3, Pos4, Pos5
Sets port 1, 2, 3, 4 and 5 to OFF, ON, Blinking mode 1 or Blinking mode 2, depending on value.

Macro	Value As Char	Value in ASCII	Description
OFF	0	0x30	Port off
ON	1	0x31	Port on
Blinking mode 1	2	0x32	Blinking mode 1
Blinking mode 2	3	0x33	Blinking mode 2
Don't Care	X	0x58	No change on that port

Return Value

1 if successful, otherwise the return value is -1.

6.3.2.3 GetDeviceStatus

int GetDeviceStatus ([in] int DevIdx, [out] char * Pos1, [out] char * Pos2, [out] char * Pos3, [out] char * Pos4, [out] char * Pos5)

GetDeviceStatus gets the status for all outputs.

Parameters

DevIdx
Index to the desire device currently available in the system. The first index is 1.

Pos1, Pos2, Pos3, Pos4, Pos5
Pointer to a variable that will contain the status of the output.

Macro	Value As Char	Value in ASCII	Description
OFF	0	0x30	Port off
ON	1	0x31	Port on
Blinking mode 1	2	0x32	Blinking mode 1
Blinking mode 2	3	0x33	Blinking mode 2

Return Value

1 if successful, otherwise the return value is -1.

6.3.2.4 GetDeviceSerial

int GetDeviceSerial([in] int DevIdx, [out] int * TypeCode, [out] int * BatchCode, [out] int * SerialCode)

GetDeviceSerial gets the specified device serial code. This code is unique and can be use to distinguish between various devices currently installed in the system.

Parameters

DevIdx
Index to the desire device currently available in the system. The first index is 1.

TypeCode
Pointer to a variable that will contain the device type code.

BatchCode
Pointer to a variable that will contain the device batch code.

SerialCode
Pointer to a variable that will contain the device serial code.

Return Value

1 if successful, otherwise the return value is -1.

6.3.2.5 GetDeviceLocation

int GetDeviceLocation ([in] int DevIdx, [out] int * Loc)

GetDeviceLoation gets the Location ID of the USB port, where the device is connected. The Location ID is unique and can be used to distinguish between various devices currently installed in the system.*

Parameters

DevIdx
: Index to the desire device currently available in the system. The first index is 1.

Loc
: Pointer to a variable that will contain the Location ID of the USB-Port.

Return Value

1 if successful, otherwise the return value is -1.

* Please note that Linux, Mac OS X and Windows CE do not support location IDs

6.3.2.6 GetFirmwareVersion

int GetFirmwareVersion ([in] int DevIdx, [out] int * Major, [out] int * Minor, [out] int * Build)

GetFirmwareVersion gets the specified device firmware version.

Parameters

DevIdx
: Index to the desire device currently available in the system. The first index is 1.

Major
: Pointer to a variable that will contain the device major firmware version value.

Minor
: Pointer to a variable that will contain the device minor firmware version value.

Build
: Pointer to a variable that will contain the device build firmware version value.

Return Value

1 if successful, otherwise the return value is -1.

6.3.2.7 GetLibVersion

int GetLibVersion ([out] int * Major, [out] int * Minor)

GetLibVersion gets the currently used library version.

Parameters

Major
Pointer to a variable that will contain the library major version value.

Minor
Pointer to a variable that will contain the library minor version value.

Return Value

1 if successful, otherwise the return value is -1.

6.3.3 Further Functions

Note: These functions are covered by the functions: SetDeviceStatus and GetDeviceStatus.

6.3.3.1 SetIOStatus

int SetIOStatus([in] int DevIdx, [in] int IOMask)

SetIOStatus sets the status for all Outputs.

Parameters

DevIdx
Index to the desire device currently available in the system. The first index is 1.

IOMask
Specifies the lights status mask of all the Outputs. The value 1 at a given position in the mask indicates that the Output is set to ON status. The value 0 indicates the OFF status. This parameter can be a combination of the following Output port indexes. The value range is between 0-31.

Macro	Value	Description
OFF	0	No port enable
O1	1	Output #1 enable
O2	2	Output #2 enable
O3	4	Output #3 enable
O4	8	Output #4 enable
O5	16	Output #5 enable

Return Value

1 if successful, otherwise the return value is -1.

6.3.3.2 SetSingleIO

int SetSingleIO([in] int DevIdx, [in] int IOIdx, [in] int Status)

SetSinlgeIO sets the status of a single Output.

Parameters

DevIdx
Index to the desire device currently available in the system. The first index is 1.

IOIdx
Sets the status of a single Output. This value range is between 1 and 5.

Status
Specifies the desire status for the given Output.
0 – OUTPUT OFF
1 – OUTPUT ON
2 – Blinking mode 1
3 – Blinking mode 2

Return Value

1 if successful, otherwise the return value is -1.

6.3.3.3 GetIOStatus

int GetIOStatus([in] int DevIdx, [out] int * OutMask)

GetIOStatus gets the current Output status for all Outputs.

Parameters

DevIdx
Index to the desire device currently available in the system. The first index is 1.

OutMask
Pointer to a variable that will contain the current Output status mask. This parameter can be the sum of the following Output port indexes.

Macro	Value	Description
OFF	0	No port enable
O1	1	IO #1 enable, also if Blinking mode1 or Blinking mode 2 is set
O2	2	IO #2 enable, also if Blinking mode1 or Blinking mode 2 is set
O3	4	IO #3 enable , also if Blinking mode1 or Blinking mode 2 is set
O4	8	IO #4 enable , also if Blinking mode1 or Blinking mode 2 is set
O5	16	IO #5 enable , also if Blinking mode1 or Blinking mode 2 is set

Return Value

1 if successful, otherwise the return value is -1.

6.3.3.4 GetSingleIO

int GetSingleIO ([in] int DevIdx, [in] int IOIdx, [out] int *Status)

GetSingleIO gets the status of a specified single Output.

Parameters

DevIdx
: Index to the desire device currently available in the system. The first index is 1.

IOIdx
: Specifies the index of current desire Output.
This value range between 1 to 5.

Status
: Specifies a pointer to variable that will contain the Output status. The value set in the variable can be as following.
0 – Output is OFF
1 – Output is ON
2 – Output is Blinking mode 1
3 – Output is Blinking mode 2

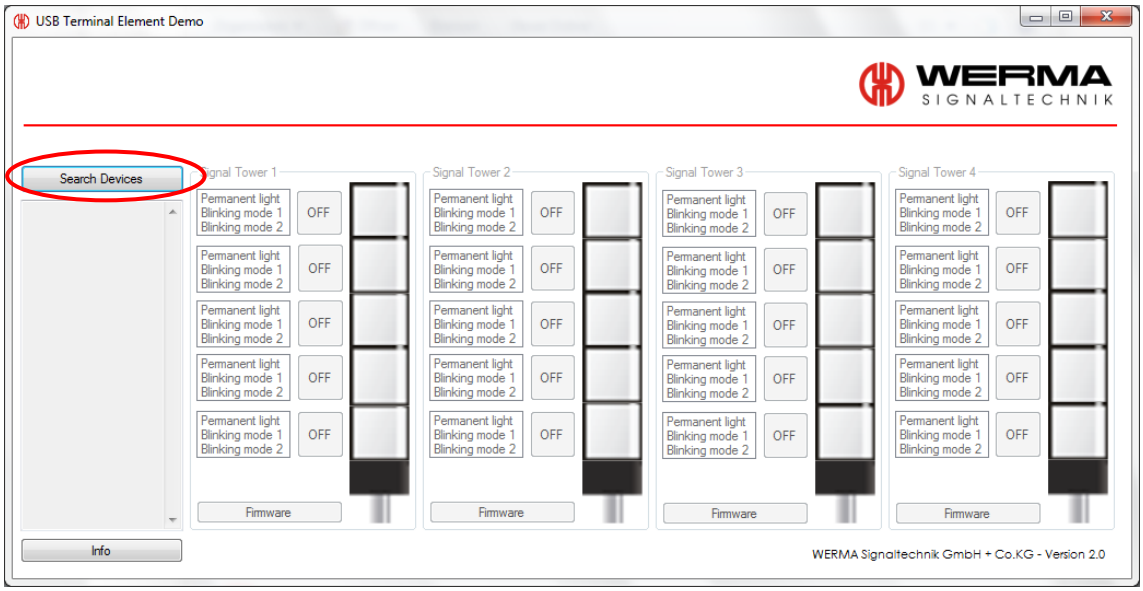
Return Value

1 if successful, otherwise the return value is 0.

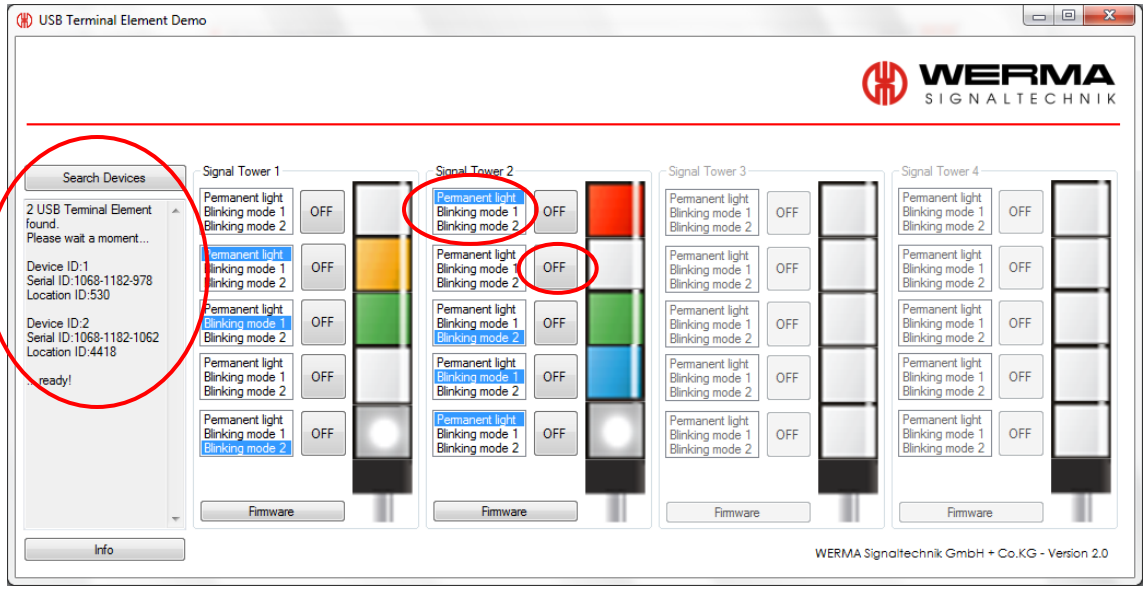
7. USB Terminal Element Demo

Start the Demo.exe, which is stored in the folder “Demo” on the CD. The following window opens.

Note: Make sure that the Demo.exe and slma.dll are stored in the same folder. Otherwise you will get an error message that the DLL was not found.



Connect the USB terminal element to a USB port on your PC. Click on “Search Devices” to find all connected USB terminal elements (max. four elements in the demo).



On the left side you can see all the USB terminal elements, which are connected to the PC. By clicking on “Permanent light”, “Blinking mode 1” or “Blinking mode 2” you get different light effects for each tier and each signal tower. Clicking on “OFF” deactivates the light.



WERMA Signaltechnik GmbH + Co.KG
D-78604 Rietheim-Weilheim
Fon: +49 (0)7424 / 9557-0
Fax: +49 (0)7424 / 9557-44
info@werma.com
www.werma.com

310.640.008.0911