The Embedded I/O Company

TEWS TECHNOLOGIES

TPMC861-SW-65

Windows Device Driver

4 Channel Serial Interface (RS485/RS422)

Version 2.0.x

User Manual

Version 2.0.2

September 2021

powerBridge Computer

Ehlbeek 15a
30938 Burgwedel
fon 05139-9980-0
fax 05139-9980-49

www.powerbridge.de
info@powerbridge.de

TEWS TECHNOLOGIES GmbH
Am Bahnhof 7 25469 Halstenbek, Germany
9 (0) 4101 4058 0 Fax: +49 (0) 4101 4058 19
il: info@tews.com www.tews.com

TPMC861-SW-65

Windows Device Driver

4 Channel Serial Interface (RS485/RS422)

Supported Modules:
TPMC861

Issue	Description	Date
1.0.0	First Issue	September 8, 2003
1.0.1	File list modified	February 18, 2005
1.0.2	Device names changed	June 13, 2005
1.0.3	File list change, Description Return value of CloseHandle() corrected, more global description of installation	July 26, 2006
1.0.4	New Address of TEWS LLC	August 15, 2007
2.0.0	Windows 7 / 64-bit Support added, Property Pages added, General Revision	April 11, 2013
2.0.1	Installation chapter modified	March 6, 2018
2.0.2	Installation chapter modified	September 24, 2021

Table of Contents

1 Introduction

The TPMC861-SW-65 Windows device driver is a kernel mode driver which allows the operation of the supported hardware modules on an Intel or Intel-compatible Windows operating system.

The standard file input and output (I/O) functions (CreateFile, CloseHandle, ReadFile, ReadFileEx, WriteFile, WriteFileEx and DeviceIoControl) provide the basic interface for opening and closing a communications resource handle and for performing read and write operations.

The TPMC861 device driver is fully compatible to the standard Windows serial device driver (*serial.sys*).

The TPMC861-SW-65 device driver supports the modules listed below:

TPMC861	4 Channel Serial Interface	PMC

2 Installation

Following files are located in directory TPMC861-SW-65 on the distribution media:

tpmc861bus\	Directory containing bus driver files
tpmc861port\	Directory containing serial port driver files
installer_32bit.exe	Installation tool for 32bit systems
installer_64bit.exe	Installation tool for 64bit systems
dpinst.xml	Installation XML file
TPMC861-SW-65-2.0.2.pdf	This document
Release.txt	Information about the Device Driver Release
ChangeLog.txt	Release history

2.1 Software Installation

2.1.1 Windows 10

This section describes how to install the TPMC861-SW-65 Device Driver on a Windows 10 (32bit or 64bit) operating system.

Depending on the operating system type, execute the installer binary for either 32bit or 64bit systems. This will install all required driver files using an installation wizard.

After successful installation a device is created for each channel found.

2.2 Confirming Driver Installation

To confirm that the driver has been properly loaded, perform the following steps:

1. Open the Windows Device Manager:
   a. Open the "***Control Panel***" from "***My Computer***" and then click the "***Device Manager***" entry.
2. Click the "**+**" in front of "***Embedded I/O***".
   The enumerator device
   " ***TPMC861 Serial Port Enumerator***" should appear.
3. Click the "**+**" in front of "***Ports (COM & LPT)***".
   The serial port devices
   "***TEWS TECHNOLOGIES TPMC861 Serial Port Device (TPMC861) (COMxx)***" should appear.

3 Default Configuration

The default configuration of the port can be modified by using the property page of the port device.

The property page can be opened from the device manager. A right-click to the port device will open a menu where 'Properties' can be selected. The property page will open. The tab 'Port Settings' will show the default settings of the port.

All settings will only be used on device startup. Therefore it is necessary to restart the device after modifying any of the settings below. (Restart the device using the device manager, or simply restart the system)

3.1 Basic Port Settings

Using this page the basic settings of the port can be changed. Basic settings are:

- 'Bits per second'	baud rate	
- 'Data bits'	number of data	(5, 6, 7, 8)
- 'Parity'	parity mode	(None, Even, Odd, Space, Mark)
- 'Stop bits'	number of stopbits	(1, 1.5, 2)
- 'Flow control'	flow control mode	(None) (Xon/Xoff, Hardware are unsupported)

3.2 Advanced Port Settings

The advanced port settings can be opened by pressing the 'Advanced' Button at the Basic Port Settings page.

This site allows modification of the buffer trigger levels for 'Receive Buffer' and 'Transmit Buffer'. Increasing a value means, that system load is decreased, but the risk of an overrun for receive, or a gap in transmission stream is increased.

The TPMC861 devices are using a 128 byte internal FIFO, but the property page supports a 16 character FIFOs of legacy serial UARTs. Therefore the trigger levels are not compatible to that of the TPMC861. The FIFO values shown in the property sheet are multiplied with a factor of 8. E.g. if the receive trigger level is configured to 4 the channel is using a trigger level of 32.

Disabling the FIFOs is not recommended, because this will increase the possibility of data loss and will also increase system load.

The site also allows advising COM-Port numbers. This may be useful for applications that only allow the use of some special port numbers.

4 Device Driver Programming

The Microsoft® Win32® application programming interface (API) also includes a set of functions that provide special communication services like reading and setting communication parameter, transmitting immediate characters, setting timeouts and so on.

All of these standard Win32 communication functions are described in detail in the Windows Platform SDK Documentation (Windows base services / Communication).

For details refer to the Win32 Programmers Reference of your used programming tools (C++, Visual Basic etc.)

The Windows name of the first port is *Devicetpmc861_0,* of the second port *Device*\ *tpmc861_1* and so on.**

The DOS device name for TPMC861 devices is COM1, COM2, COM3 and so on. If there are other serial devices in the system the prefix starts with a higher number (see Windows name).

The mapping between Windows device names and DOS device names for TPMC861 devices can be retrieved from the 'Advanced Port Settings'.

4.1 TPMC861 Files and I/O Functions

The following section does not contain a full description of the Win32 functions for interaction with the TPMC861 device driver. This chapter describes how to open and close ports. The devices can be generally accessed like the serial onboard COM ports COM1 and COM2.

4.1.1 Opening a TPMC861 Device

Before you can perform any I/O, the TPMC861 device must be opened by invoking the CreateFile function. CreateFile returns a handle that can be used to access the TPMC861 device.

```
HANDLE CreateFile
(
      LPCTSTR   lpFileName,
      DWORD     dwDesiredAccess,
      DWORD     dwShareMode,
      LPSECURITY_ATTRIBUTES lpSecurityAttributes,
      DWORD     dwCreationDistribution,
      DWORD     dwFlagsAndAttributes,
      HANDLE    hTemplateFile
)
```

PARAMETERS

lpFileName

This parameter points to a null-terminated string, which specifies the name of the TPMC861 to open. The *lpFileName* string should be of the form \\.\COM*x* to open the device *x.*

dwDesiredAccess

This parameter specifies the type of access to the TPMC861.
For the TPMC861 this parameter must be set to read-write access (GENERIC_READ | GENERIC_WRITE)

dwShareMode

Set of bit flags that specify how the object can be shared. Set to 0.

lpSecurityAttributes

This argument is a pointer to a security structure. Set to NULL for TPMC861 devices.

dwCreationDistribution

Specifies the action to take on existing files, and which action to take when files do not exist. TPMC861 devices must be always opened OPEN_EXISTING.

dwFlagsAndAttributes

Specifies the file attributes and flags for the file. This value must be set to 0 for TPMC861 devices.

hTemplateFile

This value must be NULL for TPMC861 devices.

RETURN VALUE

If the function succeeds, the return value is an open handle to the specified TPMC861 device. If the function fails, the return value is INVALID_HANDLE_VALUE. To get extended error information, call ***GetLastError***.

EXAMPLE

```
HANDLE    hDevice;

hDevice = CreateFile(
     "\\\\.\\COM5",
     GENERIC_READ | GENERIC_WRITE,
     0,
     NULL,                    // no security attrs
     OPEN_EXISTING,           // TPMC861 device always open existing
     0,                       // no overlapped I/O
     NULL);
if (hDevice == INVALID_HANDLE_VALUE)
{
     ErrorHandler("Could not open device"); // process error
}
```

SEE ALSO

CloseHandle(), Win32 documentation CreateFile()

4.1.2 Closing a TPMC861 Device

The CloseHandle function closes an open TPMC861 handle.

BOOL CloseHandle(
HANDLE *hDevice;*
)

PARAMETERS

hDevice

Identifies an open TPMC861 handle.

RETURN VALUE

If the function succeeds, the return value is nonzero.

If the function fails, the return value is zero. To get extended error information, call ***GetLastError*.**

EXAMPLE

```
HANDLE  hDevice;

if(!CloseHandle(hDevice))
{
     ErrorHandler("Could not close device"); // process error
}
```

SEE ALSO

CreateFile (), Win32 documentation CloseHandle ()

5 Known Problems

5.1 Order of Serial Ports (Windows and later)

The order of the Serial Ports shown in the Devices Manager may not match channel numbering on the board. Also the assignment of COM Port numbers may not match to the local channel numbers, and also not match to the order shown in the device manager.

Fixing COM Port assignment can be done as described in chapter 3.2 Advanced Port Settings. The local channel number is shown as 'Path' by the device properties.

Stopping and restarting devices by the Device Manager or system restarts will not affect the port assignment.

5.2 COM Port Assignment on Higher Port Numbers

If the COM Port assignment does not start with first unused COM Port or the assignment shows gaps in the COM Port assignment, e.g. the four COM ports of a TPMC861 are assigned to COM7 up to COM10, instead of COM3 up to COM6 as expected, this may be caused by problems when uninstalling devices and drivers. This assignment can be corrected in two steps.

1. Check and remove hidden and no more needed COM devices, if any are found. Therefore it may be necessary to enable hidden devices shown in the device manager. This can be enabled by setting the following Environment Variables:

   ```
   Devmgr_show_details=1
   Devmgr_show_nonpresent_devices=1
   ```

2. Use the COM port assignment as described in the 3.2 Advanced Port Settings to assign the correct COM Port name.

5.3 Settings in HyperTerminal

The driver does not support changing settings with HyperTerminal. Other terminal applications will work fine.