The Embedded I/O Company

TEWS TECHNOLOGIES

TPMC866-SW-65

Windows Device Driver

8 Channel Serial Interface

Version 2.0.x

User Manual

Version 2.0.1

September 2024

TEWS Technologies GmbH
Eggerstedter Weg 14, 25421 Pinneberg
Phone: +49 (0) 4101 4058 0
info@tews.com www.tews.com

TPMC866-SW-65

Windows Device Driver

8 Channel Serial Interface

Supported Modules:
TPMC866-10
TPMC866-11
TPMC866-12

Issue	Description	Date
1.0	First Issue	June 5, 2003
1.1	TPMC866-12 Support added	September 8, 2003
1.1.1	New file list	February 18, 2005
1.1.2	File list change, more global description of installation	July 27, 2006
1.1.3	New Address TEWS LLC	September 27, 2006
1.1.4	Driver files located in subdirectory, file list changed	September 18, 2008
2.0.0	Windows 7 / 64-bit Support added, Property Pages added, General Revision	September 03, 2013
2.0.1	Support for Windows 10 added. Support for earlier Windows versions removed New Address TEWS Technologies GmbH	September 24, 2024

Table of Contents

1 Introduction

The TPMC866-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 TPMC866 device driver is fully compatible to the standard Windows serial device driver (*serial.sys*).

The TPMC866-SW-65 device driver supports the modules listed below:

TPMC866-10	8 Channel Serial Interface (RS232)	PMC
TPMC866-11	8 Channel Serial Interface (RS422)	PMC
TPMC866-12	8 Channel Serial Interface (RS422, RS485 Full- / Halfduplex))	PMC

2 Installation

Following files are located in directory TPMC866-SW-65 on the distribution media:

tpmc866bus\	Directory containing bus driver files
tpmc866port\	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
example\tpmc866exa.c	Example application source
TPMC866-SW-65-2.0.1.pdf	This document
Release.txt	Information about the Device Driver Release
ChangeLog.txt	Release history

2.1 Software Installation

2.1.1 Windows 10/11

This section describes how to install the TPMC866-SW-65 Device Driver on a Windows 10 (32bit or 64bit) and Windows 11 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 "***TPMC866 Serial Port Enumerator (TPMC866)***" should appear, where <*Board Type*> displays the name of the mounted board and <*n*> the number of supported serial channels.
3. Click the "**+**" in front of "***Ports (COM & LPT)***".
   The serial port devices ***"TPMC866 Serial Port Device (COM<x>)***" should appear, where <*x*> refers to the assigned COM channel number.

2.3 Device and Software De-Installation

To prevent a reservation of COM-ports that will not be needed any more it is necessary to remove devices correctly. Therefor you have to decide, if the port should be removed temporarily – the board will be remounted later and the COM names should be used again – or if it should be removed permanently and the COM names can be assigned to other devices.

2.3.1 Temporary Remove and Reinstallation of Devices

Removing devices temporary is done quite simple, by a shutdown of the system and remove of the hardware. Windows will keep the configuration and naming of the devices, but the devices will not be shown.

For a reinstallation it is necessary that the hardware is mounted to the same slot again. The system will automatically start using all old configurations and names.

If the board is mounted in another slot, Windows will recognize new devices and will assign new COM names.

2.3.2 Permanent Remove of Devices

For a permanent and clean remove of serial devices it is necessary that the hardware, which shall be removed, is present in the system. The system must be started and the devices must be de-installed using the Windows Device Manager. First remove the port devices and afterwards the enumerator device. This will make sure, that the system will allow a reuse of the COM port names. After that the system should be stopped and the hardware can be removed.

If a new serial board is mounted to the system afterwards, windows will assign COM port names beginning with the first unused number, including the COM names of the de-installed ports.

2.3.3 De-installation of the Software

If the software has been installed with the installer application as described in 2.1.1, the driver should be de-installed using “Software” in the “Control Panel”. To avoid a permanent reservation of COM names, the affected devices should be removed first as described in 2.3.2.

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.

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)

3.2 Advanced Port Settings

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

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

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 TPMC866 devices are using a 64 byte (TPMC866-12: 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 TPMC866. The receive trigger level can be set to 1, 4, 8 and 14 in the property page, which sets up receive trigger levels of 8, 16, 56 and 60. The transmit trigger level can only be set to 1, 4, 8 and 14 in the property page, which sets up receive trigger levels of 8, 16, 32 and 56.

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 *Devicetpmc866_0,* of the second port *Device*\ *tpmc866_1* and so on.**

The DOS device name for TPMC866 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 TPMC866 devices can be retrieved from the 'Advanced Port Settings'.

4.1 TPMC866 Files and I/O Functions

The following section does not contain a full description of the Win32 functions for interaction with the TPMC8660 device driver. This chapter decribes how to open and close ports. The devices can be generally accessed as the serial onboard COM ports COM0 and COM1.

4.1.1 Opening a TPMC866 Device

Before you can perform any I/O, the TPMC866 device must be opened by invoking the CreateFile function. CreateFile returns a handle that can be used to access the TPMC866 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 TPMC866 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 TPMC866.
For the TPMC866 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 TPMC866 devices.

dwCreationDistribution

Specifies the action to take on existing files, and which action to take when files do not exist. TPMC866 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 TPMC866 devices.

hTemplateFile

This value must be NULL for TPMC866 devices.

Return Value

If the function succeeds, the return value is an open handle to the specified TPMC866 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,           // TPMC866 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 TPMC866 Device

The CloseHandle function closes an open TPMC866 handle.

BOOL CloseHandle
(
HANDLE *hDevice;*
)

Parameters

hDevice

Identifies an open TPMC866 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

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 eight COM ports of a TPMC866 are assigned to COM7 up to COM14, instead of COM3 up to COM10 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.