



TPMC860-SW-72
LynxOS Device Driver
4-Channel Serial RS232 PMC

Version 1.0

Reference Manual
Issue 1.0

February 15, 2002

TEWS TECHNOLOGIES GmbH
Am Bahnhof 7
D-25469 Halstenbek
Germany
Tel.: +49 (0)4101 4058-0
Fax.: +49 (0)4101 4058-19
<http://www.tews.com>
e-mail: info@tews.com

TPMC860-SW-72
TPMC860 - 4 Channel
Serial RS232 Interface
Module
LynxOS Device Driver

This document contains information, which is proprietary to TEWS TECHNOLOGIES GmbH. Any reproduction without written permission is forbidden.

TEWS TECHNOLOGIES GmbH has made any effort to ensure that this manual is accurate and complete. However TEWS TECHNOLOGIES GmbH reserves the right to change the product described in this document at any time without notice.

This product has been designed to operate with PCI Mezzanine Card (PMC) compatible carriers. Connection to incompatible hardware is likely to cause serious damage.

TEWS TECHNOLOGIES GmbH is not liable for any damage arising out of the application or use of the device described herein.

©2002 by TEWS TECHNOLOGIES GmbH

Issue	Description	Date
1.0	First Issue	February 15, 2002

Table of Contents

1	INTRODUCTION	4
2	INSTALLATION.....	5
2.1	Device Driver Installation.....	5
2.1.1	Static Installation.....	5
2.1.1.1	Build the driver object.....	5
2.1.1.2	Create Device Information Declaration	5
2.1.1.3	Modify the Device and Driver Configuration File	6
2.1.1.4	Rebuild the Kernel.....	6
2.1.2	Dynamic Installation.....	7
2.1.3	Device Information Definition File.....	8
2.1.4	Configuration File: CONFIG.TBL.....	10
3	TPMC860 DEVICE DRIVER PROGRAMMING.....	11

1 Introduction

The TPMC860-SW-72 LynxOS device driver is a full-duplex tty device driver which allows the operation of a TPMC860 serial PMC on a PowerPC platform with DRM based PCI interface.

The TPMC860-SW-72 device driver based on the standard LynxOS terminal manager. Due to this way of implementation the driver interface and function is absolute compatible to the standard LynxOS terminal driver.

All standard utility programs for configuration and maintaining terminal interfaces could be used in the same manner. There is only one exception, the TPMC860-SW-72 can't be used as console driver.

Additional supported features:

- ✓ Extended baudrates up to 460,8 kBaud.
- ✓ Each channel has a 64 Byte transmit and receive FIFO
- ✓ Programmable trigger level for transmit and receive FIFO.
- ✓ Hardware (RTS/CTS) and software handshake (Xon/Xoff) direct controlled by the serial controller. The advantage of this feature is that the transmission of characters will immediately stop as soon as a complete character is transmitted and not when the transmit FIFO is empty for handshake under software control.

2 Installation

The software is delivered on a PC formatted 3½" HD diskette.

Following files are located on the diskette:

tpmc860.c	Driver source code
tpmc860_info.c	Device information definition
tpmc860_info.h	Device information definition header
tpmc860.cfg	Driver configuration file include
tpmc860.import	Linker import file
Makefile	Device driver make file
Makefile.dlidd	Make file for dynamic driver installation
tpmc860-sw-72.pdf	This Manual in PDF format

2.1 Device Driver Installation

The two methods of driver installation are as follows:

- Static Installation
- Dynamic Installation

2.1.1 Static Installation

With this method, the driver object code is linked with the kernel routines and is installed during system start-up.

In order to perform a static installation, copy the following files to the target directories:

1. Create a new directory in the system drivers directory path.
For example: `/sys/drivers.pp_drm/tpmc860`
2. Copy the following files to this directory: `tpmc860.c`, `Makefile`
3. Copy `tpmc860_info.c` to `/sys/devices/`
4. Copy `tpmc860_info.h` to `/sys/dheaders/`
5. Copy `tpmc860.cfg` to `/sys/cfg.ppc/`

2.1.1.1 Build the driver object

1. Change to the directory `/sys/drivers.pp_drm/tpmc860`
2. To update the library `/sys/lib/libdrivers.a` enter:

```
make install
```

2.1.1.2 Create Device Information Declaration

1. Change to the directory `/sys/devices/`
2. Add the following dependencies to the `Makefile`

```
DEVICE_FILES_prep = ...tpmc860_info.x
```

And at the end of the Makefile

```
...  
tpmc860_info.o:$(DHEADERS)/tpmc860_info.h
```

3. To update the library `/sys/lib/libdevices.a` enter:

```
make install
```

2.1.1.3 Modify the Device and Driver Configuration File

In order to insert the driver object code into the kernel image, an appropriate entry in file `CONFIG.TBL` must be created.

1. Change to the directory `/sys/lynx.os/`
2. Create an entry in the file `CONFIG.TBL`
Insert the entry after the console driver section

```
# End of console devices  
I:tpmc860.cfg
```

2.1.1.4 Rebuild the Kernel

1. Change to the directory `/sys/lynx.os/`
2. To rebuild the kernel enter the following command:

```
make install
```

3. Reboot the newly-created operating system by the following command:

```
reboot -aN
```

The **N** flag instructs *init* to run *mknod* and create all the nodes mentioned in the new *nodetab*.

4. After reboot you should find the following new devices (depends on the device configuration): `/dev/tp860a1 ... /dev/tp860a4`

2.1.2 Dynamic Installation

This method allows you to install the driver after the operating system is booted. The driver object code is attached to the end of the kernel image and the operating system dynamically adds this driver to its internal structures. The driver can also be removed dynamically.

Unlike the description of the dynamic installation in the manual "Writing Device Drivers for LynxOS", the driver source must be placed in a directory under `/sys/drivers.pp_drm/`

The following steps describe how to do a dynamic installation:

1. Create a new directory in the system drivers directory path.

For example: `/sys/drivers.pp_drm/tpmc860`

2. Copy the following files to this directory:

- tpmc860.c
- tpmc860_info.c
- tpmc860_info.h
- tpmc860.import
- Makefile.dldd

3. Change to the directory `/sys/drivers.pp_drm/tpmc860`

4. To make the dynamic link-able driver enter :

```
make -f Makefile.dldd
```

5. Create a device definition file for one major device

```
gcc -DDLDD -o tpmc860_info tpmc860_info.c  
./tpmc860_info > tp860A
```

6. To install the driver enter:

```
drinstall -c tpmc860.obj
```

If successful *drinstall* returns a unique *<driver-ID>*

7. To install the major device enter:

```
devinstall -c -d <driver-ID> tp860A
```

The *<driver-ID>* is returned by the *drinstall* command

8. To create nodes for the devices enter:

```
mknod /dev/tp860a1 c <major_no> 0  
mknod /dev/tp860a2 c <major_no> 1  
...
```

If all steps are successful completed the TPMC860 is ready to use.

To uninstall the TPMC860 device enter the following commands:

```
devinstall -u -c <device-ID>  
drinstall -u <driver-ID>
```

2.1.3 Device Information Definition File

The device information definition contains information necessary to install the TPMC860 major device.

The implementation of the device information definition is done through a C structure which is defined in the header file *tpmc860_info.h*.

This structure contains following parameter:

PCIBusNumber	Contains the PCI bus number at which the TPMC860 is connected. Valid bus numbers are in range from 0 to 255.
PCIDeviceNumber	Contains the device number (slot) at which the TPMC860 is connected. Valid device numbers are in range from 0 to 31. NOTE. If both <i>PCIBusNumber</i> and <i>PCIDeviceNumber</i> are -1 then the driver will autoscan for the TPMC860 device. The first device found in the scan order will be allocated by the driver for this major device. Already allocated devices can't be allocated twice. This is important to know if you have more that one TMPC860 major device.
FIFO_Settings[]	Contains the trigger level for the receive and transmit FIFO's. Valid settings for the receiver FIFO are: 1, 8, 16, 56, 60. Valid settings for the transmitter FIFO are: 1, 8, 16, 32, 56. NOTE. A trigger level of 1 will disable the receive or transmit FIFO.
sg	This structure contains initial tty parameter like baudrate special characters and so on. Refer also to the tty man pages.

A device information definition is unique for every TPMC860 major device. The file *tpmc860_info.c* on the distribution disk contains two device information declarations, **tp860A** for the first major device and **tp860B** for the second major device.

If the driver should support more than two major devices it is necessary to copy and paste an existing declaration and rename it with unique name for example **tp860C**, **tp860D** and so on.

NOTE. It is also necessary to modify the device and driver configuration file respectively the configuration include file *tpmc860.cfg*.

The following device declaration information uses the auto find method to detect the TPMC860 module on PCI bus.

The Rx trigger level is set to 56 Byte and the Tx trigger level is set to 4 for all minor devices.

```

TP860_INFO tp860A = {
    -1,                /* auto find the TPMC860 on any PCI bus */
    -1,
    { /* Rx, Tx trigger level */
        { 56, 8 }, /* channel 0 ( minor device 0 ) */
        { 56, 8 }, /* channel 1 ( minor device 1 ) */
        { 56, 8 }, /* channel 2 ( minor device 2 ) */
        { 56, 8 }, /* channel 3 ( minor device 3 ) */
    },
    {
        B9600, B9600, /* input and output speed */
        'H' - '@', /* erase char */
        -1, /* 2nd erase char */
        'U' - '@', /* kill char */
        ECHO | CRMOD, /* mode */
        'C' - '@', /* interrupt character */
        '\\\' - '@', /* quit char */
        'Q' - '@', /* start char */
        'S' - '@', /* stop char */
        'D' - '@', /* EOF */
        -1, /* brk */
        (LCRTBS | LCRTERA | LCRTKIL | LCTLECH), /* local mode word */
        'Z' - '@', /* process stop */
        'Y' - '@', /* delayed stop */
        'R' - '@', /* reprint line */
        'O' - '@', /* flush output */
        'W' - '@', /* word erase */
        'V' - '@' /* literal next char */
    }
};

```

2.1.4 Configuration File: CONFIG.TBL

The device and driver configuration file *CONFIG.TBL* contains entries for device drivers and its major and minor device declarations. Each time the system is rebuild, the *config* utility reads this file and produces a new set of driver and device configuration tables and a corresponding *nodetab*.

To install the TPMC860 driver and devices into the LynxOS system, the configuration include file *tpmc860.cfg* must be included in the *CONFIG.TBL* (see also 2.1.1.3). The file *tpmc860.cfg* on the distribution disk contains the driver entry (C:tpmc860:\....) and one enabled major device entry (D:TPMC860 1-4:tp860A::) with 4 minor device entries (N: tp860a1:0 ... N:tp860a4:0).

If the driver should support more than one major device the following entries for major and minor devices must be enabled by removing the comment character (#). By copy and paste an existing major and minor entry and renaming the new entries, it is possible to add any number of additional TPMC860 device.

NOTE. The name of the device information declaration (info-block-name) must match to an existing C structure in the file *tpmc860_info.c*.

This example shows a driver entry with one major device and 4 minor devices:

```
#Format:
#C:driver-name:open:close:read:write:select:control:install:uninstall
#D:device-name:info-block-name:raw-partner-name
#N:node-name:minor-dev

C:tpmc860:\
  :tp860open:tp860close:tp860read:tp860write:\
  :tp860select:tp860ioctl:tp860install:tp860uninstall
D:TPMC860 1-8:tp860A::
N:tp860a1:0
N:tp860a2:1
N:tp860a3:2
N:tp860a4:3
```

The configuration above creates the following nodes in the */dev* directory.

```
/dev/tp860a1
/dev/tp860a2
/dev/tp860a3
/dev/tp860a4
```

3 TPMC860 Device Driver Programming

The TPMC860-SW-72 device driver based on the standard LynxOS terminal manager. Due to this way of implementation the driver interface and function is absolute compatible to the standard LynxOS terminal driver.

The TPMC860-SW-72 device driver supports two additional ioctl functions to enable and disable the hardware handshake (RTS/CTS).

TIOSHWH (290)* Enable the hardware handshake with RTS/CTS. This functions is only accepted for TPMC860-10 modules, all other TPMC860 variants returns **SYSERR**.

TIOCHWH (291)* Disable the hardware handshake.

* Function code

No arguments are necessary for these commands. The parameter *arg* in the device control function call can be set to 0.

Example (without checking error situations!):

```
#define TIOSHWH 290
#define TIOCHWH 291

main()
{
    int fd;
    int result;

    /* open the desired TPMC860 device */
    fd = open( "/dev/tp860a1", O_RDWR );

    /* enable hardware handshake */
    result = ioctl( fd, TIOSHWH, 0 );

    /* ... do I/O ... */

    /* disable hardware handshake */
    result = ioctl( fd, TIOCHWH, 0 );

    close( fd );
}
```