The Embedded I/O Company

TEWS TECHNOLOGIES

TCP866-SW-82

Linux Device Driver

8 Channel Serial PMC

Version 1.0.x

User Manual

Issue 1.0

December 2003

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

TEWS TECHNOLOGIES LLC
1 E. Liberty Street, Sixth Floor
Reno, Nevada 89504 / USA
Phone: +1 (775) 686 6077
Fax: +1 (775) 686 6024
e-mail: usasales@tews.com
www.tews.com

TCP866-SW-82

8 Channel Serial PMC

Linux Device Driver

Issue	Description	Date
1.0	First Issue	February 10, 2000

Table of Contents

1 INTRODUCTION 4
2 INSTALLATION 5
2.1 Build and install the device driver 5
2.2 Uninstall the device driver 5
2.3 Install device driver into the running kernel 6
2.4 Remove device driver from the running kernel 6
2.5 Change Major Device Number 7
2.6 FIFO Configuration 8
3 DEVICE DRIVER PROGRAMMING 9
3.1 Simple Programming example 9
4 DIAGNOSTIC 10

1 Introduction

The TCP866-SW-82 Linux device driver is a full-duplex serial driver which allows the operation of a TCP866 serial PMC on Linux operating systems with Intel and Intel-compatible x86 CPU.

The TCP866-SW-82 device driver based on the standard Linux serial device driver and supports all standard terminal functions (TERMIOS).

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 flow control (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. This will greatly improve flow control reliability.
- Direct support of different physical interfaces (RS-232, RS-422, RS-485 half-duplex and full-duplex).
- Designed as Linux kernel module with dynamically loading.
- Supports shared IRQ's.
- Automatic configuration on startup, i.e. driver scans the entire PCI bus and initializes all found TCP866 modules.
- Creates a TTY device ttyTCP and dialout device cuaTCP with dynamically allocated or fixed major device numbers.

2 Installation

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

The directory A:\TCP866-SW-82 contains the following files:

TCP866-SW-82.pdf	This manual in PDF format
TCP866-SW-82.tar.gz	GZIP compressed archive with driver source code

The GZIP compressed archive TCP866-SW-82.tar.gz contains the following files and directories:

tcp866.c	Driver source code
tcp866.h	Driver header file
load866	Script for driver loading
unload866	Script for driver unloading
makefile	Device driver make file
tcp866-SW-82.pdf	This Manual in PDF format
example/test866.c	Simple driver test program (only example code)
example/setspeed.c	Baudrate setup utility (only example code)

In order to perform an installation, extract all files of the archive TCP866-SW-82.tar.gz to the desired target directory.

2.1 Build and install the device driver

- Login as *root*
- Change to the target directory
- To create and install the driver in the module directory */lib/modules/<version>/misc* enter:

  # make install

Some Linux distributions and kernel versions requires modification of the Makefile. If you got errors during compilation or installation of the driver please verify the macros VER and INCLUDEDIR in the Makefile.

2.2 Uninstall the device driver

- Login as *root*
- Change to the target directory
- To remove the driver from the module directory */lib/modules/<version>/misc* enter:

  # make uninstall

2.3 Install device driver into the running kernel

To insert the driver into the running kernel login as root and execute the shell script load866.

 # sh load866

This shell script removes a previously installed TCP866 driver, installs the new one into the kernel and creates nodes for all 8 channels of a TCP866.

Created tty device nodes are:

```
/dev/ttyTCP0, /dev/ttyTCP1, ..., /dev/ttyTCP7
```

Created cua device nodes are:

```
/dev/cuaTCP0, /dev/cuaTCP1, ..., /dev/cuaTCP7
```

The unmodified driver use dynamic allocation of major device numbers. To get the current used major number the script extracts the major number for the TTY and CUA driver from /proc/devices to create the correct device nodes.

2.4 Remove device driver from the running kernel

To remove the driver from the running kernel login as root and execute the shell script *unload866*.

 # sh unload866

2.5 Change Major Device Number

The released TPCM866 driver use dynamic allocation of major device numbers. If this isn't suitable for the application it's possible to define a major number separately for the *TTY* and *CUA* driver.

To change the major number edit the file tcp866.c, change the following symbols to appropriate values and enter *make install* to create a new driver.

TCP866_TTY_MAJOR Defines the value for the terminal device. Valid numbers are in range between 0 and 255. A value of 0 means dynamic number allocation.

TCP866_CUA_MAJOR Defines the value for the dialout device. Valid numbers are in range between 0 and 255. A value of 0 means dynamic number allocation.

Example:

```
#define TCP866_TTY_MAJOR        122
#define TCP866_CUA_MAJOR        123
```

Be sure that the desired major number isn't used by other drivers. Please check */proc/devices* to see which numbers are free.

Keep in mind that's necessary to create new device nodes if the major number for the TCP866 driver has changed and the load866 script isn't used.

2.6 FIFO Configuration

After installation of the TCP866 Device Driver the trigger level for transmit and receive FIFO are set to their default values.

Default values are:

Receive FIFO	Transmit FIFO
56	8

The configuration of the FIFO trigger level is used for all TCP866 devices in common.

To change the trigger levels edit the file tcp866.c, change the following symbols to appropriate values and enter *make install* to create a new driver.

TCP866_RX_TRG_DEF Defines the trigger level for the receiver FIFO.
Valid trigger levels are:
UART_FCR_R_TRIGGER_8
UART_FCR_R_TRIGGER_16
UART_FCR_R_TRIGGER_56
UART_FCR_R_TRIGGER_60

TCP866_TX_TRG_DEF Defines the trigger level for the transmitter FIFO.
Valid trigger levels are:
UART_FCR_T_TRIGGER_8
UART_FCR_T_TRIGGER_16
UART_FCR_T_TRIGGER_32
UART_FCR_T_TRIGGER_56

Please refer to the User Manual of the ST16C654 controller to get more information how to customize suitable FIFO trigger level.

3 Device Driver Programming

The TCP866-SW-82 driver loosely based on the standard Linux terminal driver. Due to this way of implementation the driver interface and functionality is compatible to the standard Linux terminal driver.

Please refer to the TERMIOS man page and driver programming related man pages for more information about serial driver programming.

3.1 Simple Programming example

This example program opens the first serial channel of a TCP866 for read/write. After the device is open it writes a “Hello World” string to the device and receives up 80 bytes from the serial channel.

```
main()
{
   int fd;
   int count;
   char buffer[81];

   /* open the desired TCP866 device */
   fd = open( “/dev/ttyTCP0”, O_RDWR | O_NOCTTY);

   if (fd < 0) exit(-1);

   /* write data to TCP866 device */
   count = write(fd, “Hello World\n”, 12);

   printf(“%d bytes written\n”, count);

   /* read up to 80 bytes from the device */
   count = read(fd, buffer, 80)

   if (count < 0) {
      printf(“read error\n”);
   }
   else {
      buffer[count] = 0;
      printf(“%d bytes read <%s>\n”, count, buffer);
   }

   close(fd);
}
```

The source files *test866.c* and *setspeed.c* contains additional programming examples.

4 Diagnostic

If the TCP866 does not work properly it is helpful to get some status information from the driver respective kernel.

The Linux */proc* file system provides information about kernel, resources, driver, devices and so on. The following screen dumps displays information of a correct running TCP866 driver (see also the proc man pages).

```
# cat /proc/tty/driver/tcp866
TPCM866 serial driver:1.0.0 revision:2003-12-06
0: uart:ST16C654 port:D880 irq:10 tx:0 rx:0
1: uart:ST16C654 port:D888 irq:10 tx:0 rx:0
2: uart:ST16C654 port:D890 irq:10 tx:0 rx:0 DSR|CD|RI
3: uart:ST16C654 port:D898 irq:10 tx:0 rx:0 DSR|CD|RI
4: uart:ST16C654 port:D8A0 irq:10 tx:0 rx:0 DSR|CD|RI
5: uart:ST16C654 port:D8A8 irq:10 tx:0 rx:0 DSR|CD|RI
6: uart:ST16C654 port:D8B0 irq:10 tx:0 rx:0 DSR|CD|RI
7: uart:ST16C654 port:D8B8 irq:10 tx:0 rx:0 DSR|CD|RI


# cat /proc/tty/drivers
tcp866               /dev/cuaTCP   253   0-255 serial:callout
tcp866               /dev/ttyTCP   254   0-255 serial
serial               /dev/cua        5  64-127 serial:callout
serial               /dev/ttyS       4  64-127 serial
pty_slave            /dev/pts      143   0-255 pty:slave
pty_master           /dev/ptm      135   0-255 pty:master
pty_slave            /dev/ttyp       3   0-255 pty:slave
pty_master           /dev/pty        2   0-255 pty:master
/dev/vc/0            /dev/vc/0       4       0 system:vtmaster
/dev/ptmx            /dev/ptmx       5       2 system
/dev/console         /dev/console    5       1 system:console
/dev/tty             /dev/tty        5       0 system:/dev/tty
unknown              /dev/vc/%d      4    1-63 console


# cat /proc/interrupts
           CPU0
  0:      60612          XT-PIC  timer
  1:        893          XT-PIC  keyboard
  2:          0          XT-PIC  cascade
  5:          0          XT-PIC  usb-uhci
  8:          1          XT-PIC  rtc
 10:         30          XT-PIC  sym53c8xx, TCP866
 11:        277          XT-PIC  eth0, eth1
 12:       1679          XT-PIC  PS/2 Mouse
 14:       9917          XT-PIC  ide0
 15:         10          XT-PIC  ide1
NMI:          0
ERR:          0
```

```
# cat /proc/ioports
0000-001f : dma1
0020-003f : pic1
0040-005f : timer
0060-006f : keyboard
0070-007f : rtc
0080-008f : dma page reg
00a0-00bf : pic2
00c0-00df : dma2
00f0-00ff : fpu
0170-0177 : ide1
01f0-01f7 : ide0
02e8-02ef : serial(auto)
02f8-02ff : serial(auto)
0376-0376 : ide1
0378-037a : parport0
03c0-03df : vga+
03f6-03f6 : ide0
0400-043f : Intel Corp. 82371AB/EB/MB PIIX4 ACPI
0440-045f : Intel Corp. 82371AB/EB/MB PIIX4 ACPI
0cf8-0cff : PCI conf1
c000-cfff : PCI Bus #01
d000-dfff : PCI Bus #02
  d880-d8ff : PCI device 1498:2362 (TEWS Datentechnik GmBH)
    d880-d8c9 : TCP866 (PCI)
  dc00-dc7f : PCI device 1498:2362 (TEWS Datentechnik GmBH)
e800-e8ff : LSI Logic / Symbios Logic (formerly NCR) 53c875
  e800-e87f : sym53c8xx
ee80-eebf : Intel Corp. 82559ER (#2)
  ee80-eebf : e100
ef00-ef3f : Intel Corp. 82559ER
  ef00-ef3f : e100
ef80-ef9f : Intel Corp. 82371AB/EB/MB PIIX4 USB
  ef80-ef9f : usb-uhci
ffa0-ffaf : Intel Corp. 82371AB/EB/MB PIIX4 IDE
  ffa0-ffa7 : ide0
  ffa8-ffaf : ide1
```