The Embedded I/O Company

TEWS TECHNOLOGIES

TIP865-SW-72

LynxOS Device Driver

4 Channel Serial I/O

User Manual

Issue 1.0 Version 1.0.0

February 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

TIP865-SW-72

4 Channel Serial I/O

LynxOS Device Driver

Issue	Description	Date
1.0	First Issue	February 14, 2003

Table of Content

1 Introduction

The TIP865-SW-72 LynxOS device driver is a full-duplex tty device driver which allows the operation of a TIP865 serial IPAC on a PowerPC platform.

The TIP865-SW-72 device driver is 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 TIP865-SW-72 can not be used as console driver.

Additional supported features:

Direct support of different physical interfaces (RS-232, RS-422, RS-485 and TTL).

2 Installation

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

Following files are located on the diskette:

tip865.c	Driver source code
tip865def.h	Definitions and data structures for the driver
tip865_info.c	Device information definition
tip865_info.h	Device information definition header
tip865.cfg	Driver configuration file include
tip865.import	Linker import file
Makefile	Device driver make file
Makefile.dldd	Make file for dynamic driver installation
TIP865-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 (only native LynxOS systems)

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 their target directories:

1. Create a new directory in the system driver directory path */sys/drivers.xxx*, where *xxx* represents the BSP that supports the target hardware. For example: */sys/drivers.pp_drm/tip865*
2. Copy the following files to this directory: *tip865.c, tip865def.h, Makefile*
3. Copy *tip865_info.c* to */sys/devices.xxx/* or */sys/devices* if */sys/devices.xxx* does not exist (xxx represents the BSP).
4. Copy *tip865_info.h* to */sys/dheaders/*
5. Copy *tip865.cfg* to */sys/cfg.ppc/*

2.1.1.1 Build the driver object

1. Change to the directory */sys/drivers.xxx/tip865,* where *xxx* represents the BSP that supports the target hardware.
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.xxx* or */sys/devices* if */sys/devices.xxx* does not exist (xxx represents the BSP).
2. Add the following dependencies to the *Makefile*

   DEVICE_FILES_all = ... tip865_info.x
3. And at the end of the Makefile

   tip865_info.o:$(DHEADERS)/tip865_info.h
4. 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/* respective */sys/bsp.xxx*, where *xxx* represents the BSP that supports the target hardware.
2. Create an entry at the end of the file CONFIG.TBL

   ```
   I:tip865.cfg
   ```

2.1.1.4 Rebuild the Kernel

1. Change to the directory */sys/lynx.os/* (*/sys/bsp.xxx*)
2. Enter the following command to rebuild the kernel:

   ```
   make install
   ```
3. Reboot the newly created operating system by the following command (not necessary for KDIs):

   ```
   reboot –aN
   ```
4. The **N** flag instructs *init* to run *mknod* and create all the nodes mentioned in the new *nodetab*.
5. After reboot you should find the following new devices (depends on the device configuration): */dev/t865a, …]*

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.

The following steps describe how to do a dynamic installation:

1. Create a new directory in the system driver directory path */sys/drivers.xxx*, where *xxx* represents the BSP that supports the target hardware. For example: */sys/drivers.pp_drm/tip865*

1. Copy the following files to this directory:

   Tip865.c
   Tip865def.h
   Tip865_info.c
   Tip865_info.h
   Tip865.import
   Makefile.dldd

2. Change to the directory */sys/drivers.xxx/tip865*

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

   ```
   make –f Makefile.dldd
   ```

4. Create a device definition file for one major device

   ```
   gcc –DDLDD –o tip865_info tip865_info.c
   ./tip865_info > t865a_info
   ```

5. To install the driver enter:

   ```
   drinstall –c tip865.obj
   ```

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

6. To install the major device enter:

   ```
   devinstall –c –d <driver-ID> t865a_info
   ```

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

7. To create nodes for the devices enter:

   ```
   mknod /dev/t865a c <major_no> 0...
   ```

If all steps are successful completed the TIP865 is ready to use.

To uninstall the TIP865 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 TIP865 major device.

The implementation of the device information definition is done through a C structure, which is defined in the header file *tip865_info.h*.

This structure contains the following parameter:

IpIoVirtualAddress This parameter contains the kernel virtual address of the IP I/O space (device registers). This address depends on the configuration of the IP carrier board. In case of a VMEbus carrier this space usually appears in the VMEbus short I/O space A16/D16.

IpIdVirtualAddress This parameter contains the kernel virtual address of the IP ID space (ID-PROM). This address depends on the configuration of the IP carrier board. In case of a VMEbus carrier this space usually appears in the VMEbus short I/O space A16/D16.

vector12 Contains the vector at which the TIP865 generates interrupts for the first SCC (channel 1 and 2). If the TIP865 is plugged on a VMEbus carrier any free vector from 64 to 255 can be used. The driver setup the vector register of the first SCC with this vector during driver initialization.

vector34 Contains the vector at which the TIP865 generates interrupts for the second SCC (channel 3 and 4). If the TIP865 is plugged on a VMEbus carrier any free vector from 64 to 255 can be used. The driver setup the vector register of the first SCC with this vector during driver initialization.

sg[] This structure contains initial tty parameter like baudrate special characters and so on. Refer also to the **tty** man pages. The index specifies the channel. (Index 0 for channel 1, index 1 for channel 2 etc.)

If the TIP865 is plugged on a VMEbus carrier be sure that the appropriate VMEbus driver *uvme* or *vme* is started (CONFIG.TBL). See also *Chapter 5 – Accessing Hardware* in the "Writing Device Drivers for LynxOS" manual and the man pages *uvmedrvr* and *vmedrvr* for information about the VMEbus configuration and mapping.

Be sure that the used VME address windows (A16/D32) are enabled. If the *uvmedrvr* driver is responsible for the VMEbus access please check the file *../dheaders/uvmeinfo.h* (pci slave A16 D32).

A device information definition is unique for every TIP865 major device. The file *tip865_info.c* on the distribution disk contains a device information declaration.

If the driver should support more major devices it is necessary to copy and paste an existing declaration and rename it with unique name for example **t865b_info**, **t865c_info** and so on.

It is also necessary to modify the device and driver configuration file, respectively the configuration include file *tip865.cfg*.

The following device declaration information expected that the IP spaces appear at virtual address 0xCFFF8000 for IP I/O and at virtual address 0xCFFF8080 for IP ID space. The interrupt vector 64 is the first usable vector on the VMEbus.

```
T865_INFO t865a_info =
{
     0xCfff8000                  /*  IP I/O Space         */
     0xCfff8080                  /*  IP ID Space          */
     64                          /*  Interrupt Vector 1/2 */
     66                          /*  Interrupt Vector 3/4 */
     {
          {    /* Channel 1 - Setup */
               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        */
          },

...
```

```
...

        {   /* Channel 2 - Setup */
            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        */
        },
        {   /* Channel 3 - Setup */
            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        */
        },

...
```

```
...

        {    /* Channel 4 - Setup */
             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        */
        }
    }
};
```

***ECHO* should be disabled for RS485 channels. Therefore simply remove the *ECHO* definition in the associated sg structure. *ECHO* enabled may cause problems receiving data, because the RS485 interface only supports half-duplex mode.**

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 the file and produces a new set of driver and device configuration tables and a corresponding *nodetab*.

To install the TIP865 driver and devices into the LynxOS system, the configuration include file *tip865.cfg* must be included in the *CONFIG.TBL*.

The file *tip865.*cfg on the distribution disk contains the driver entry (C:tip865:\....) and one enabled major device entry (D:TIP865 1-4:t865a_info::) with four minor device entry (N: t865a0:0 / N: t865a1:1 / N: t865a2:2 / N: t865a3:3).

If the driver should support more than one major device (TIP865) the following entries for major and minor devices must be enabled by removing the comment character (#). By copy and paste an existing major and four minor entries and renaming the new entries, it is possible to add any number of additional TIP865 devices.

The name of the device information declaration (info-block-name) must match to an existing C structure in the file *tip865_info.c*.

This example shows a driver entry with one major device:

```
#    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:tip865:\
     :t865open:t865close:t865read:::::t865install:t865uninstall
D:TIP865 1-4:t865a_info::
N:t865a0:0
N:t865a1:1
N:t865a2:2
N:t865a3:3
```

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

```
/dev/t865a0
/dev/t865a1
/dev/t865a2
/dev/t865a3
```

3 TIP865 Device Driver Programming

The TIP865-SW-72 device driver is 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.

4 Debugging and Diagnostic

This driver was successfully tested on a Motorola MVME2306-900 board, and developed on a Windows Cross Environment for LynxOS V4.0.0.

If the driver will not work properly please enable debug outputs by removing the comments around the symbol *DEBUG.*

The debug output should appear on the console. If not please check the symbol *KKPF_PORT* in *uparam.h*. This symbol should be configured to a valid COM port (e.g. *SKDB_COM1*).

The debug output displays the device information data for the current major device, a memory dump of the IP ID space (contents of the ID-PROM) and a memory dump of the IP I/O space (registers) like this.

```
TIP865 Device Driver Install
IP I/O Virtual Address   = CFFF8000
IP ID  Virtual Address   = CFFF8080
Interrupt vectors        = 64/66

IP ID  space (ID-PROM)...
CFFF8080 : FF 49 FF 50 FF 41 FF 43 FF B3 FF 10 FF 10 FF 00
CFFF8090 : FF 00 FF 00 FF 10 FF C4 FF 04 FF 04 FF 04 FF 04
CFFF80A0 : FF 00 FF 00 FF 00 FF 00 FF 00 FF 00 FF 00 FF 00
CFFF80B0 : FF 00 FF 00 FF 00 FF 00 FF 00 FF 00 FF 00 FF 00

IP I/O space (Registers)...
CFFF8000 : FF 6C FF 00 FF 6C FF 00 FF EC FF 00 FF 6C FF 00
CFFF8010 : FF 6C FF 00 FF 6C FF 00 FF EC FF 00 FF 6C FF 00
CFFF8020 : FF 6C FF 00 FF 6C FF 00 FF EC FF 00 FF 6C FF 00
CFFF8030 : FF 6C FF 00 FF 6C FF 00 FF EC FF 00 FF 6C FF 00
CFFF8040 : FF 6C FF 00 FF 6C FF 00 FF EC FF 00 FF 6C FF 00
CFFF8050 : FF 6C FF 00 FF 6C FF 00 FF EC FF 00 FF 6C FF 00
CFFF8060 : FF 6C FF 00 FF 6C FF 00 FF EC FF 00 FF 6C FF 00
CFFF8070 : FF 6C FF 00 FF 6C FF 00 FF EC FF 00 FF 6C FF 00
CFFF8080 : FF 49 FF 50 FF 41 FF 43 FF B3 FF 10 FF 10 FF 00
CFFF8090 : FF 00 FF 00 FF 10 FF C4 FF 04 FF 04 FF 04 FF 04
CFFF80A0 : FF 00 FF 00 FF 00 FF 00 FF 00 FF 00 FF 00 FF 00
CFFF80B0 : FF 00 FF 00 FF 00 FF 00 FF 00 FF 00 FF 00 FF 00
CFFF80C0 : FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
CFFF80D0 : FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
CFFF80E0 : FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
CFFF80F0 : FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

...
```

```
...

Channel 1: Adr: CFFF8005h (RS485)
Set Baudrate: 9600 Baud
Channel 2: Adr: CFFF8001h (RS485)
Set Baudrate: 9600 Baud
Channel 3: Adr: CFFF800Dh (RS485)
Set Baudrate: 9600 Baud
Channel 4: Adr: CFFF8009h (RS485)
Set Baudrate: 9600 Baud
```

The debug output above is only an example. Debug output on other systems may be different for addresses and data in some locations.