The Embedded I/O Company

TEWS TECHNOLOGIES

TDRV006-SW-65

Windows Device Driver

64 Channel Digital I/O

Version 2.0.x

User Manual

Issue 2.0.0

September 2011

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

TDRV006-SW-65

Windows Device Driver

64 Channel Digital I/O

Supported Modules:
TPMC681

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.

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

©2006-2011 by TEWS TECHNOLOGIES GmbH

Issue	Description	Date
1.0.0	First Issue	March 31, 2006
1.0.1	New Address TEWS LLC	September 20, 2006
1.0.2	Files moved to subdirectory	June 25, 2008
2.0.0	Driver supports Windows7, Description of API Functions added, Description of I/O Functions removed, General Revision	September 16, 2011

Table of Contents

1 Introduction

The TDRV006-SW-65 Windows device driver is a kernel mode driver which allows the operation of supported hardware modules on an Intel or Intel-compatible Windows operating system. Supported Windows versions are:

- Windows 2000
- Windows XP
- Windows XP Embedded
- Windows 7 (32bit and 64bit)

The standard file and device (I/O) functions (CreateFile, CloseHandle, and DeviceIoControl) provide the basic interface for opening and closing a resource handle and for performing device I/O control operations.

The TDRV006-SW-65 device driver supports the following features:

- read actual input value
- write new output value
- configure I/O lines for input or output
- wait for input events

The TDRV006-SW-65 device driver supports the modules listed below:

TPMC681	64 I/O Lines	PMC

In this document all supported modules and devices will be called TDRV006. Specials for certain devices will be advised.

To get more information about the features and use of TDRV006 devices it is recommended to read the manuals listed below.

TPMC681 User manual
TPMC681 Engineering Manual

2 Installation

Following files are located in directory TDRV006-SW-65 on the distribution media:

i386\	Directory containing driver files for 32bit Windows versions
amd64\	Directory containing driver files for 64bit Windows versions
installer_32bit.exe	Installation tool for 32bit systems (Windows XP or later)
installer_64bit.exe	Installation tool for 64bit systems (Windows XP or later)
tdrv006.inf	Windows installation script
tdrv00h	Header file with IOCTL codes and structure definitions
example\tdrv006exa.c	Example application
api\tdrv006api.c	Application Programming Interface source
api\tdrv006api.h	Application Programming Interface header
TDRV006-SW-65-2.0.0.pdf	This document
Release.txt	Information about the Device Driver Release
ChangeLog.txt	Release history

2.1 Software Installation

2.1.1 Windows 2000

This section describes how to install the TDRV006 Device Driver on a Windows 2000 operating system.

After installing the TDRV006 card(s) and boot-up your system, Windows 2000 setup will show a "***New hardware found***" dialog box.

1. The "***Upgrade Device Driver Wizard***" dialog box will appear on your screen.
   Click "***Next***" button to continue.
2. In the following dialog box, choose "***Search for a suitable driver for my device***".
   Click "***Next***" button to continue.
3. Insert the TDRV006 driver media; select "***Disk Drive***" in the dialog box.
   Click "***Next***" button to continue.
4. Now the driver wizard should find a suitable device driver on the media.
   Click "***Next***" button to continue.
5. Complete the upgrade device driver and click "***Finish***" to take all the changes effect.
6. Now copy all needed files (tdrv006.h and API files) to the desired target directories.

After successful installation the TDRV006 device driver will start immediately and creates devices (TDRV006_1, TDRV006_2 ...) for all recognized TDRV006 modules.

2.1.2 Windows 7 / XP

This section describes how to install the TDRV006-SW-65 Device Driver on a Windows 7 (32bit or 64bit) or Windows XP (32-bit) operating system.

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

Copy needed files (tdrv006.h and API files) to desired target directory.

After successful installation a device is created for each module found (TDRV006_1, TDRV006_2 ...).

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. For Windows 2000 / XP, open the "***Control Panel***" from "***My Computer***" and click the "***System***" icon and choose the "***Hardware***" tab, and then click the "***Device Manager***" button.
    b. For Windows 7, open the "***Control Panel***" from "***My Computer***" and then click the "***Device Manager***" entry.
2. Click the "**+**" in front of "***Embedded I/O***".
   The driver "***TEWS TECHNOLOGIES – TDRV006 (Digital I/O) (TPMC681)***" should appear for each installed device.

3 API Documentation

3.1 General Functions

3.1.1 tdrv006Open

NAME

tdrv006Open – Opens a Device

SYNOPSIS

```
TDRV006_HANDLE tdrv006Open
(
    char                        *DeviceName
);
```

DESCRIPTION

Before I/O can be performed to a device, a file descriptor must be opened by a call to this function.

PARAMETERS

DeviceName

This parameter points to a null-terminated string that specifies the name of the device.

EXAMPLE

```
#include “tdrv006api.h”

TDRV006_HANDLE     hdl;

/*
** open file descriptor to device
*/
hdl = tdrv006Open(“\\\\.\\TDRV006_1”);
if (hdl == NULL)
{
     /* handle open error */
}
```

RETURNS

A device handle, or NULL if the function fails. To get extended error information, call ***GetLastError***.

ERROR CODES

All error codes are standard error codes set by the I/O system.

3.1.2 tdrv006Close

NAME

tdrv006Close – Closes a Device

SYNOPSIS

```
TDRV006_STATUS tdrv006Close
(
    TDRV006_HANDLE          hdl
);
```

DESCRIPTION

This function closes previously opened devices.

PARAMETERS

hdl

This value specifies the device handle to the hardware module retrieved by a call to the corresponding open-function.

EXAMPLE

```
#include “tdrv006api.h”

TDRV006_HANDLE hdl;
TDRV006_STATUS result;

/*
** close file descriptor to device
*/
result = tdrv006Close(hdl);
if (result != TDRV006_OK)
{
    /* handle close error */
}
```

RETURNS

On success TDRV006_OK, or an appropriate error code.

ERROR CODES

All error codes are standard error codes set by the I/O system.

3.2 Device Access Functions

3.2.1 tdrv006Read

NAME

tdrv006Read – read state of the I/O lines

SYNOPSIS

```
TDRV006_STATUS tdrv006Read
(
        TDRV006_HANDLE                  hdl,
        unsigned int                    *input_31_0,
        unsigned int                    *input_63_32
);
```

DESCRIPTION

This function reads the value of the input registers on the specified device. These registers show the current state of the I/O lines independent on their configured direction.

PARAMETERS

hdl

This argument specifies the device handle to the hardware module retrieved by a call to the corresponding open-function.

input_31_0

This argument points to a 32-bit value where the state of the I/O lines 31 down to 0 will be returned. The MSB shows the state of I/O line 31 and the LSB shows the state of I/O line 0.

input_63_32

This argument points to a 32-bit value where the state of the I/O lines 63 down to 32 will be returned. The MSB shows the state of I/O line 63 and the LSB shows the state of I/O line 32.

EXAMPLE

```
#include “tdrv006api.h”

TDRV006_HANDLE   hdl;
TDRV006_STATUS   result;
unsigned int     inVal_31_0;
unsigned int     inVal_63_32;

/* show the current state of the I/O lines */
result = tdrv006Read (  hdl,
                        &inVal_31_0,
                        &inVal_63_32);
if (result != TDRV006_OK)
{
     /* handle error */
}

printf(“I/O lines (63…0) = %08X%08Xh\n”, inVal_63_32, inVal_31_0);
```

RETURNS

On success, TDRV006_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV006_ERR_INVALID_HANDLE	The specified TDRV006_HANDLE is invalid.

3.2.2 tdrv006Write

NAME

tdrv006Write – write value to output registers (set output lines)

SYNOPSIS

```
TDRV006_STATUS tdrv006Write
(
        TDRV006_HANDLE                  hdl,
        unsigned int                    output_31_0,
        unsigned int                    output_63_32
);
```

DESCRIPTION

This function sets the value of the output registers on the specified device. I/O lines configured for output will be set. Input lines will not be affected.

PARAMETERS

hdl

This argument specifies the device handle to the hardware module retrieved by a call to the corresponding open-function.

output_31_0

This 32-bit value specifies the new value of the output register. The state of the MSB specifies the value for I/O line 31 and the LSB specifies the value for I/O line 0.

output_63_32

This 32-bit value specifies the new value of the output register. The state of the MSB specifies the value for I/O line 63 and the LSB specifies the value for I/O line 32.

EXAMPLE

```
#include “tdrv006api.h”

TDRV006_HANDLE   hdl;
TDRV006_STATUS   result;

// Set output lines 8…16, reset all the other
result = tdrv006Write ( hdl,
                        0x0001FF00,
                        0x00000000);
if (result != TDRV006_OK)
{
     /* handle error */
}
```

RETURNS

On success, TDRV006_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV006_ERR_INVALID_HANDLE	The specified TDRV006_HANDLE is invalid.

3.2.3 tdrv006WriteMasked

NAME

tdrv006WriteMasked – write a masked value to output registers (set output lines)

SYNOPSIS

```
TDRV006_STATUS tdrv006WriteMasked
(
        TDRV006_HANDLE                  hdl,
        unsigned int                    output_31_0,
        unsigned int                    output_63_32,
        unsigned int                    mask_31_0,
        unsigned int                    mask_63_32
);
```

DESCRIPTION

This function sets parts of the value of the output registers selected by a mask on the specified device. I/O lines configured for output will be set. Input lines will not be affected.

PARAMETERS

hdl

This argument specifies the device handle to the hardware module retrieved by a call to the corresponding open-function.

output_31_0

This 32-bit value specifies the new value of the output register. The state of the MSB specifies the value for I/O line 31 and the LSB specifies the value for I/O line 0.

output_63_32

This 32-bit value specifies the new value of the output register. The state of the MSB specifies the value for I/O line 63 and the LSB specifies the value for I/O line 32.

mask_31_0

This 32-bit value specifies the parts that shall be affected with this write. A ‘1’ for the corresponding bit means that the specified output value will be used, a ‘0’ means that it will be ignored. The MSB specifies the mask for I/O line 31 and the LSB specifies the mask for I/O line 0.

mask_63_32

This 32-bit value specifies the parts that shall be affected with this write. A ‘1’ for the corresponding bit means that the specified output value will be used, a ‘0’ means that it will be ignored. The MSB specifies the mask for I/O line 63 and the LSB specifies the mask for I/O line 32.

EXAMPLE

```
#include “tdrv006api.h”

TDRV006_HANDLE   hdl;
TDRV006_STATUS   result;

// Clear output lines 0…3, do not affect the other lines
result = tdrv006WriteMasked( hdl,
                             0x00000000,
                             0x00000000,
                             0x0000000F,
                             0x00000000);
if (result != TDRV006_OK)
{
     /* handle error */
}
```

RETURNS

On success, TDRV006_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	**Description**
TDRV006_ERR_INVALID_HANDLE	The specified TDRV006_HANDLE is invalid.

3.2.4 tdrv006OutputSet

NAME

tdrv006OutputSet – set the specified output lines

SYNOPSIS

```
TDRV006_STATUS tdrv006OutputSet
(
        TDRV006_HANDLE              hdl,
        unsigned int                mask_31_0,
        unsigned int                mask_63_32
);
```

DESCRIPTION

This function sets the specified bits in the output registers on the selected device. Specified I/O lines configured for output will be set. Input lines will not be affected.

PARAMETERS

hdl

This argument specifies the device handle to the hardware module retrieved by a call to the corresponding open-function.

mask_31_0

This 32-bit mask specifies the output lines that shall be set. A ‘1’ means set the output line, a ‘0’ means keep its state. The MSB specifies the mask for I/O line 31 and the LSB specifies the mask for I/O line 0.

mask_63_32

This 32-bit mask specifies the output lines that shall be set. A ‘1’ means set the output line, a ‘0’ means keep its state. The MSB specifies the mask for I/O line 63 and the LSB specifies the mask for I/O line 32.

EXAMPLE

```
#include “tdrv006api.h”

TDRV006_HANDLE   hdl;
TDRV006_STATUS   result;

// Set output lines 0…6
result = tdrv006OutputSet (  hdl,
                             0x0000007F,
                             0x00000000);
if (result != TDRV006_OK)
{
     /* handle error */
}
```

RETURNS

On success, TDRV006_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV006_ERR_INVALID_HANDLE	The specified TDRV006_HANDLE is invalid.

3.2.5 tdrv006OutputClear

NAME

tdrv006OutputClear – clear the specified output lines

SYNOPSIS

```
TDRV006_STATUS tdrv006OutputClear
(
      TDRV006_HANDLE                hdl,
      unsigned int                  mask_31_0,
      unsigned int                  mask_63_32
);
```

DESCRIPTION

This function clears the specified bits in the output registers on the selected device. Specified I/O lines configured for output will be cleared. Input lines will not be affected.

PARAMETERS

hdl

This argument specifies the device handle to the hardware module retrieved by a call to the corresponding open-function.

mask_31_0

This 32-bit mask specifies the output lines that shall be cleared. A ‘1’ means clear the output line, a ‘0’ means keep its state. The MSB specifies the mask for I/O line 31 and the LSB specifies the mask for I/O line 0.

mask_63_32

This 32-bit mask specifies the output lines that shall be cleared. A ‘1’ means clear the output line, a ‘0’ means keep its state. The MSB specifies the mask for I/O line 63 and the LSB specifies the mask for I/O line 32.

EXAMPLE

```
#include “tdrv006api.h”

TDRV006_HANDLE   hdl;
TDRV006_STATUS   result;

// Clear output lines 32…37
result = tdrv006OutputClear( hdl,
                             0x00000000,
                             0x0000003F);
if (result != TDRV006_OK)
{
     /* handle error */
}
```

RETURNS

On success, TDRV006_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV006_ERR_INVALID_HANDLE	The specified TDRV006_HANDLE is invalid.

3.2.6 tdrv006SetOutputLine

NAME

tdrv006SetOutputLine – sets a single output line

SYNOPSIS

```
TDRV006_STATUS tdrv006SetOutputLine
(
        TDRV006_HANDLE                  hdl,
        unsigned int                    lineNo
);
```

DESCRIPTION

This function sets a specified output line on a selected device. If the specified line is not configured for output the function have no effect.

PARAMETERS

hdl

This argument specifies the device handle to the hardware module retrieved by a call to the corresponding open-function.

lineNo

This value specifies the output line that shall be set. Allowed values are between 0 and 63.

EXAMPLE

```
#include “tdrv006api.h”

TDRV006_HANDLE   hdl;
TDRV006_STATUS   result;

// Set output line 54
result = tdrv006SetOutputLine (  hdl,
                                 54);
if (result != TDRV006_OK)
{
     /* handle error */
}
```

RETURNS

On success, TDRV006_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV006_ERR_INVALID_HANDLE	The specified TDRV006_HANDLE is invalid.
TDRV006_ERR_INVAL	A specified parameter has an invalid value.

3.2.7 tdrv006ClearOutputLine

NAME

tdrv006ClearOutputLine – clears a single output line

SYNOPSIS

```
TDRV006_STATUS tdrv006ClearOutputLine
(
        TDRV006_HANDLE                  hdl,
        unsigned int                    lineNo
);
```

DESCRIPTION

This function clears a specified output line on a selected device. If the specified line is not configured for output the function has no effect.

PARAMETERS

hdl

This argument specifies the device handle to the hardware module retrieved by a call to the corresponding open-function.

lineNo

This value specifies the output line that shall be cleared. Allowed values are between 0 and 63.

EXAMPLE

```
#include “tdrv006api.h”

TDRV006_HANDLE  hdl;
TDRV006_STATUS  result;

// Clear output line 54
result = tdrv006ClearOutputLine (hdl,
                                 54);
if (result != TDRV006_OK)
{
    /* handle error */
}
```

RETURNS

On success, TDRV006_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV006_ERR_INVALID_HANDLE	The specified TDRV006_HANDLE is invalid.
TDRV006_ERR_INVAL	A specified parameter has an invalid value.

3.2.8 tdrv006DirectionConfigure

NAME

tdrv006DirectionConfigure – configure the direction of selected I/O lines

SYNOPSIS

```
TDRV006_STATUS tdrv006DirectionConfigure
(
        TDRV006_HANDLE                  hdl,
        unsigned int                    config_31_0,
        unsigned int                    config_63_32,
        unsigned int                    mask_31_0,
        unsigned int                    mask_63_32
);
```

DESCRIPTION

This function configures the direction of selected I/O lines. If a line is configured for output, the last value of the output register will appear at the output immediately.

PARAMETERS

hdl

This argument specifies the device handle to the hardware module retrieved by a call to the corresponding open-function.

config_31_0

This 32-bit value specifies which I/O line shall be output or input. A specified ‘1’ set the I/O line to output, a ‘0’ to input. The MSB specifies the direction for I/O line 31 and the LSB specifies the direction for I/O line 0.

config_63_32

This 32-bit value specifies which I/O line shall be output or input. A specified ‘1’ set the I/O line to output, a ‘0’ to input. The MSB specifies the direction for I/O line 63 and the LSB specifies the direction for I/O line 32.

mask_31_0

This 32-bit value specifies the I/O that shall be configured. A ‘1’ for the corresponding bit means that the specified I/O line will be configured, a ‘0’ means that it will keep the configuration. The MSB specifies the mask for I/O line 31 and the LSB specifies the mask for I/O line 0.

mask_63_32

This 32-bit value specifies the I/O that shall be configured. A ‘1’ for the corresponding bit means that the specified I/O line will be configured, a ‘0’ means that it will keep the configuration. The MSB specifies the mask for I/O line 63 and the LSB specifies the mask for I/O line 32.

EXAMPLE

```
#include “tdrv006api.h”

TDRV006_HANDLE  hdl;
TDRV006_STATUS  result;

// Configure lines 0…15 for input, 16…31 for output and
// keep the configuraion of the other lines
result = tdrv006DirectionConfigure (  hdl,
                                      0xFFFF0000,
                                      0x00000000,
                                      0xFFFFFFFF,
                                      0x00000000);
if (result != TDRV006_OK)
{
     /* handle error */
}
```

RETURNS

On success, TDRV006_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV006_ERR_INVALID_HANDLE	The specified TDRV006_HANDLE is invalid.

3.2.9 tdrv006DirectionRead

NAME

tdrv006DirectionRead – read the current configuration of the I/O line direction

SYNOPSIS

```
TDRV006_STATUS tdrv006DirectionRead
(
    TDRV006_HANDLE                  hdl,
    unsigned int                    *config_31_0,
    unsigned int                    *config_63_32
);
```

DESCRIPTION

This function reads the current configuration of I/O line direction.

PARAMETERS

hdl

This argument specifies the device handle to the hardware module retrieved by a call to the corresponding open-function.

config_31_0

This argument points to a 32-bit value where the configuration of the I/O lines 31 down to 0 will be returned. A bit set to ‘1’ shows the line is configured for output, a ‘0’ means input. The MSB shows the configuration of I/O line 31 and the LSB shows the configuration of I/O line 0.

config_63_32

This argument points to a 32-bit value where the configuration of the I/O lines 63 down to 32 will be returned. A bit set to ‘1’ shows the line is configured for output, a ‘0’ means input. The MSB shows the configuration of I/O line 63 and the LSB shows the configuration of I/O line 32.

EXAMPLE

```
#include “tdrv006api.h”

TDRV006_HANDLE   hdl;
TDRV006_STATUS   result;
unsigned int     confVal_31_0;
unsigned int     confVal_63_32;

/* show the current state of the I/O lines */
result = tdrv006DirectionRead (   hdl,
                                  &confVal_31_0,
                                  &confVal_63_32);
if (result != TDRV006_OK)
{
     /* handle error */
}

printf(“I/O line configuration (63…0) = %08X%08Xh\n”,
         confVal_63_32, confVal_31_0);
```

RETURNS

On success, TDRV006_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV006_ERR_INVALID_HANDLE	The specified TDRV006_HANDLE is invalid.

3.2.10 tdrv006WaitForHighTransition

NAME

tdrv006WaitForHighTransition – wait for a low to high transition on a specified input line

SYNOPSIS

```
TDRV006_STATUS tdrv006WaitForHighTransition
(
        TDRV006_HANDLE                  hdl,
        unsigned int                    lineNo,
        int                             timeout
);
```

DESCRIPTION

This function waits for a low to high transition on a specified input line. A transition on an I/O line configured for output will not trigger the event.

PARAMETERS

hdl

This argument specifies the device handle to the hardware module retrieved by a call to the corresponding open-function.

lineNo

This value specifies the input line that shall be observed for the specified transition. Allowed values are between 0 and 63.

timeout

This value specifies the time the function shall wait for the event. If the specified time passes, the function shall return announcing an appropriate error. The time shall be specified in milliseconds, but the timeout granularity is 1 second. A timeout value of '-1' specifies that the function will never timeout.

EXAMPLE

```
#include “tdrv006api.h”

TDRV006_HANDLE   hdl;
TDRV006_STATUS   result;

// Wait at least 20 seconds for a low to high transition on input line 10
result = tdrv006WaitForHighTransition (    hdl,
                                           10,
                                           20000);
if (result != TDRV006_OK)
{
     /* handle error */
}
```

RETURNS

On success, TDRV006_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV006_ERR_INVALID_HANDLE	The specified TDRV006_HANDLE is invalid.
TDRV006_ERR_INVAL	A specified parameter has an invalid value.
TDRV006_ERR_TIMEOUT	The specified timeout has passed
TDRV006_ERR_ABORTED	The waiting has been aborted by a power down of the system

3.2.11 tdrv006WaitForLowTransition

NAME

tdrv006WaitForLowTransition – wait for a high to low transition on a specified input line

SYNOPSIS

```
TDRV006_STATUS tdrv006WaitForLowTransition
(
        TDRV006_HANDLE                  hdl,
        unsigned int                    lineNo,
        int                             timeout
);
```

DESCRIPTION

This function waits for a high to low transition on a specified input line. A transition on an I/O line configured for output will not trigger the event.

PARAMETERS

hdl

This argument specifies the device handle to the hardware module retrieved by a call to the corresponding open-function.

lineNo

This value specifies the input line that shall be observed for the specified transition. Allowed values are between 0 and 63.

timeout

This value specifies the time the function shall wait for the event. If the specified time passes, the function shall return announcing an appropriate error. The time shall be specified in milliseconds, but the timeout granularity is 1 second. A timeout value of '-1' specifies that the function will never timeout.

EXAMPLE

```
#include “tdrv006api.h”

TDRV006_HANDLE   hdl;
TDRV006_STATUS   result;

// Wait at least 5 seconds for a high to low transition on input line 11
result = tdrv006WaitForLowTransition( hdl,
                                      11,
                                      5000);
if (result != TDRV006_OK)
{
     /* handle error */
}
```

RETURNS

On success, TDRV006_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV006_ERR_INVALID_HANDLE	The specified TDRV006_HANDLE is invalid.
TDRV006_ERR_INVAL	A specified parameter has an invalid value.
TDRV006_ERR_TIMEOUT	The specified timeout has passed
TDRV006_ERR_ABORTED	The waiting has been aborted by a power down of the system

3.2.12 tdrv006WaitForTransition

NAME

tdrv006WaitForTransition – wait for a transition on a specified input line

SYNOPSIS

```
TDRV006_STATUS tdrv006WaitForTransition
(
        TDRV006_HANDLE                  hdl,
        unsigned int                    lineNo,
        int                             timeout
);
```

DESCRIPTION

This function waits for a transition on a specified input line. A transition on an I/O line configured for output will not trigger the event.

PARAMETERS

hdl

This argument specifies the device handle to the hardware module retrieved by a call to the corresponding open-function.

lineNo

This value specifies the input line that shall be observed for the transition. Allowed values are between 0 and 63.

timeout

This value specifies the time the function shall wait for the event. If the specified time passes, the function shall return announcing an appropriate error. The time shall be specified in milliseconds, but the timeout granularity is 1 second. A timeout value of ‘-1’ specifies that the function will never timeout.

EXAMPLE

```
#include “tdrv006api.h”

TDRV006_HANDLE  hdl;
TDRV006_STATUS  result;

// Wait endless until for a transition on input line 14
result = tdrv006WaitForTransition (   hdl,
                                      14,
                                      -1);
if (result != TDRV006_OK)
{
     /* handle error */
}
```

RETURNS

On success, TDRV006_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV006_ERR_INVALID_HANDLE	The specified TDRV006_HANDLE is invalid.
TDRV006_ERR_INVAL	A specified parameter has an invalid value.
TDRV006_ERR_TIMEOUT	The specified timeout has passed
TDRV006_ERR_ABORTED	The waiting has been aborted by a power down of the system

4 Hibernate Mode

The driver supports the “Hibernate”-mode of Windows. The driver will remember the device configuration and the current output value and it will restore the configuration immediately after leaving the hibernation mode.

If there are pending requests, (e.g. waiting for an input transition) and the system wants to enter hibernation mode, the request will be aborted with an appropriate error code. The application may now decide how to handle this situation.