

TDRV024-SW-82

Linux Device Driver

Digital Inputs and Digital Outputs

Version 1.0.x

User Manual

Issue 1.0.0

July 2026

TDRV024-SW-82

Linux Device Driver

Digital Inputs and Digital Outputs

Supported Modules:

TQMC601

TQMC602

TQMC603

TQMC604

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.

©2026 by TEWS Technologies GmbH

Issue	Description	Date
1.0.0	First Issue	July 8, 2026

Table of Contents

1	INTRODUCTION.....	4
2	INSTALLATION.....	5
	2.1 Build and install the Device Driver	5
	2.2 Uninstall the Device Driver	6
	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
3	API DOCUMENTATION	8
	3.1 General Functions.....	8
	3.1.1 tdrv024Open	8
	3.1.2 tdrv024Close	10
	3.1.3 tdrv024GetModuleInfo.....	12
	3.2 Device Input Functions	15
	3.2.1 tdrv024DigitalInputRead.....	15
	3.2.2 tdrv024DigitalInputTransitionWait	17
	3.2.3 tdrv024DebouncerEnable	19
	3.2.4 tdrv024DebouncerDisable	21
	3.3 Device Output Functions	23
	3.3.1 tdrv024DigitalOutputGetGroupMask.....	23
	3.3.2 tdrv024DigitalOutputConfigure.....	25
	3.3.3 tdrv024DigitalOutputWrite	28
	3.3.4 tdrv024DigitalOutputWriteMask	30
	3.3.5 tdrv024DigitalOutputLineSet	32
	3.3.6 tdrv024DigitalOutputLineClear	34
	3.3.7 tdrv024DigitalOutputStateRead	36
	3.3.8 tdrv024DigitalOutputFaultRead.....	39
	3.3.9 tdrv024WatchdogEnable.....	41
	3.3.10 tdrv024WatchdogDisable.....	43
	3.3.11 tdrv024WatchdogReset	45
	3.3.12 tdrv024WatchdogConfigure	48
	3.4 Global Functions.....	50
	3.4.1 tdrv024GlobalEventWait	50
4	DIAGNOSTIC.....	52

1 Introduction

The TDRV024-SW-82 Linux device driver allows the operation of the supported digital input and digital output modules on Linux operating systems.

The TDRV024-SW-82 device driver software includes the following features:

- read digital inputs
- set digital outputs
- configure debouncer for digital inputs
- configure and handle watchdog for digital outputs
- use interrupts for waiting on digital input events
- use interrupts to detect fault events

The TDRV024-SW-82 device driver supports the modules listed below:

TQMC601	32 Digital Inputs, Isolated	(QMC)
TQMC602	16 Digital Inputs, 16 Digital Outputs (0.3 A), Isolated	(QMC)
TQMC603	20 Digital Outputs (5-36 V, 0.3 A), Isolated	(QMC)
TQMC604	32 Digital Outputs (5-36 V, 0.15 A), Isolated	(QMC)

In this document all supported modules and devices will be called TDRV024. Specials for certain devices will be advised.

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

Corresponding Modules Hardware User Manual
--

2 Installation

Following files are located on the distribution media:

Directory path 'TDRV024-SW-82':

TDRV024-SW-82-1.0.0.pdf	This manual in PDF format
TDRV024-SW-82-SRC.tar.gz	GZIP compressed archive with driver source code
Release.txt	Release information
ChangeLog.txt	Release history

The GZIP compressed archive TDRV024-SW-82-SRC.tar.gz contains the following files and directories:

Directory path './tdrv024/':

tdrv024.c	Driver source code
tdrv024def.h	Driver include file
tdrv024.h	Driver include file for application program
Makefile	Device driver make file
makenode	Script for device node creation
COPYING	Copy of the GNU Public License (GPL)
api/tdrv024api.h	API include file
api/tdrv024api.c	API source file
include/config.h	Include of system dependent config.h
include/tpxxxhwdep.c	Low level hardware access functions source file
include/tpxxxhwdep.h	Access functions header file
include/tpmodule.c	Driver independent library
include/tpmodule.h	Driver independent library header file
example/tdrv024exa.c	Example application
example/Makefile	Example application makefile

In order to perform an installation, extract all files of the archive TDRV024-SW-82-SRC.tar.gz to the desired target directory. The command 'tar -xzf TDRV024-SW-82-SRC.tar.gz' will extract the files into the local directory.

- Login as *root* and change to the target directory
- Copy tdrv024.h and api/tdrv024api.h to */usr/include*

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
- To update the device driver's module dependencies, enter:

 # depmod -a

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 load the device driver into the running kernel, login as root and execute the following commands:

modprobe tdrv024drv

- After the first build or if you are using dynamic major device allocation it is necessary to create new device nodes on the file system. Please execute the script file *makenode* to do this. If your kernel has enabled a device file system (devfs or sysfs with udev) then you have to skip running the *makenode* script. Instead of creating device nodes from the script the driver itself takes creating and destroying of device nodes in its responsibility.

sh makenode

On success the device driver will create a minor device for each TDRV024 module found. The first module of the first TDRV024 module can be accessed with device node */dev/tdrv024_0*, the second module with device node */dev/tdrv024_1*, the third TDRV024 module with device node */dev/tdrv024_2* and so on.

The assignment of device nodes to physical TDRV024 modules depends on the search order of the PCI bus driver.

2.4 Remove Device Driver from the running Kernel

- To remove the device driver from the running kernel login as root and execute the following command:

modprobe -r tdrv024drv

If your kernel has enabled devfs or sysfs (udev), all */dev/tdrv024_x* nodes will be automatically removed from your file system after this.

Be sure that the driver isn't opened by any application program. If opened you will get the response "*tdrv024drv: Device or resource busy*" and the driver will still remain in the system until you close all opened files and execute *modprobe -r* again.

2.5 Change Major Device Number

This paragraph is only for Linux kernels without dynamic device management. The TDRV024 driver uses dynamic allocation of major device numbers per default. If this isn't suitable for the application it's possible to define a major number for the driver.

To change the major number edit the file `tdrv024def.h`, change the following symbol to appropriate value and enter `make install` to create a new driver.

TDRV024_MAJOR	Valid numbers are in range between 0 and 255. A value of 0 means dynamic number allocation.
---------------	---

Example:

```
#define TDRV024_MAJOR 122
```

Be sure that the desired major number is not used by other drivers. Please check `/proc/devices` to see which numbers are free.

Keep in mind that it is necessary to create new device nodes if the major number for the TDRV024 driver has changed and the `makenode` script is not used.

3 API Documentation

3.1 General Functions

3.1.1 tdrv024Open

NAME

tdrv024Open – opens a device

SYNOPSIS

```
TDRV024_HANDLE tdrv024Open
(
    char      *DeviceName
)
```

DESCRIPTION

Before I/O-operations can be performed to a device, a device handle 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. The first TDRV024 device is named “/dev/tdrv024_0” the second device is named “/dev/tdrv024_1” and so on.

EXAMPLE

```
#include "tdrv024api.h"

TDRV024_HANDLE  hdl;

/*
** open the specified device
*/
hdl = tdrv024open("/dev/tdrv024_0");
if (hdl == NULL)
{
    /* handle open error */
}
```

RETURNS

A device handle, or NULL if the function fails. An error code will be stored in *errno*.

ERROR CODES

The error codes are stored in *errno*.

The error code is a standard error code set by the I/O system.

3.1.2 tdrv024Close

NAME

tdrv024Close – closes a device

SYNOPSIS

```
TDRV024_STATUS tdrv024Close
(
    TDRV024_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 "tdrv024api.h"

TDRV024_HANDLE    hdl;
TDRV024_STATUS    result;

/*
** close the device
*/
result = tdrv024Close(hdl);
if (result != TDRV024_OK)
{
    /* handle close error */
}
```

RETURNS

On success, TDRV024_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV024_ERR_INVALID_HANDLE	The specified device handle is invalid

3.1.3 tdrv024GetModuleInfo

NAME

tdrv024GetModuleInfo – get information of the hardware module

SYNOPSIS

```
TDRV024_STATUS tdrv024GetModuleInfo
(
    TDRV024_HANDLE          hdl,
    TDRV024_PCIINFO_BUF     *pPciInfoBuf,
    TDRV024_BOARDINFO_BUF   *pBoardInfoBuf
)
```

DESCRIPTION

This function returns information about the hardware module, including module type and PCI header as well as the PCI localization.

PARAMETERS

hdl

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

pPciInfoBuf

This argument is a pointer to the structure TDRV024_PCIINFO_BUF that receives information of the module PCI header.

```
typedef struct
{
    unsigned short    vendorId;
    unsigned short    deviceId;
    unsigned short    subSystemId;
    unsigned short    subSystemVendorId;
    int               pciBusNo;
    int               pciDevNo;
    int               pciFuncNo;
} TDRV024_PCIINFO_BUF;
```

vendorId

PCI module vendor ID.

deviceId

PCI module device ID

subSystemId

PCI module sub system ID

subSystemVendorId

PCI module sub system vendor ID

pciBusNo

Number of the PCI bus, where the module resides.

pciDevNo

PCI device number

pciFuncNo

PCI function number

pBoardInfoBuf

This argument is a pointer to the structure `TDRV024_BOARDINFO_BUF` that receives information about the module hardware.

typedef struct

```
{
    unsigned int    moduleType;
    unsigned int    firmwareVersion;
    unsigned int    digitalOutputLines;
    unsigned int    digitalInputLines;
} TDRV024_BOARDINFO_BUF;
```

moduleType

This argument is a pointer to an *unsigned int* (32bit) data buffer, where the module type is returned. Possible values are:

Value	Description
TDRV024_MODTYPE_TQMC601	Current module is a TQMC601
TDRV024_MODTYPE_TQMC602	Current module is a TQMC602
TDRV024_MODTYPE_TQMC603	Current module is a TQMC603
TDRV024_MODTYPE_TQMC604	Current module is a TQMC604

firmwareVersion

This value returns the version of the firmware used on the device.

digitalOutputLines

This value returns the number of digital output lines available on the specified device

digitalInputLines

This value returns the number of digital input lines available on the specified device

EXAMPLE

```
#include "tdrv024api.h"

TDRV024_HANDLE      hdl;
TDRV024_STATUS      result;
TDRV024_PCIINFO_BUF pciInfoBuf;
TDRV024_BOARDINFO_BUF boardInfoBuf;

/*
** get module information
*/
result = tdrv024GetModuleInfo( hdl, &pciInfoBuf, &boardInfoBuf);
if (result != TDRV024_OK)
{
    /* handle error */
}

printf ("Boardtype:      %d", boardInfoBuf.moduleType);
printf ("Input-Lines:    %d", boardInfoBuf.digitalInputLines);
printf ("Output-Lines: %d", boardInfoBuf.digitalOutputLines);
```

RETURN VALUE

On success, TDRV024_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV024_ERR_INVALID_HANDLE	The specified device handle is invalid
TDRV024_ERR_INVALID	Specified pointer is invalid.

3.2 Device Input Functions

Device input functions are supported if digital input lines are available at the device. If this function is called for a device without digital input lines the function will complete with an appropriate error code.

If an input function returns a value for a not present digital input line, the value will be filled with zeros.

3.2.1 tdrv024DigitalInputRead

NAME

tdrv024DigitalInputRead – read state of digital input lines

SYNOPSIS

```
TDRV024_STATUS tdrv024DigitalInputRead
(
    TDRV024_HANDLE      hdl,
    unsigned int         *pInputValue
)
```

DESCRIPTION

This function reads the state of the digital input lines.

PARAMETERS

hdl

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

pInputValue

This argument points to a 32 bit buffer where the input value will be returned. Bit 0 corresponds to the digital input value of digital input line 0, bit 1 corresponds to digital input value of digital input line 1 and so on.

EXAMPLE

```
#include "tdrv024api.h"

TDRV024_HANDLE    hdl;
TDRV024_STATUS    result;
unsigned int       in_value;

/*
** read current state of the input lines
*/
result = tdrv024DigitalInputRead(hdl, &in_value);
if (result != TDRV024_OK)
{
    /* handle error */
}
else
{
    printf("value: 0x%08X\n", in_value);
}
```

RETURN VALUE

On success, TDRV024_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV024_ERR_INVALID_HANDLE	The specified device handle is invalid.
TDRV024_ERR_NOT_SUPPORTED	The specified device does not support digital input.

3.2.2 tdrv024DigitalInputTransitionWait

NAME

tdrv024DigitalInputTransitionWait – wait for one of the specified digital input transitions

SYNOPSIS

```
TDRV024_STATUS tdrv024DigitalInputTransitionWait
(
    TDRV024_HANDLE    hdl,
    unsigned int       highTransitionMask,
    unsigned int       lowTransitionMask,
    int                msTimeout
)
```

DESCRIPTION

This function waits until one of the specified transitions has occurred. Multiple digital input lines and transitions can be selected, the first occurrence of any of the specified events will terminate the function.

PARAMETERS

hdl

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

highTransitionMask

This parameter specifies the digital input lines where a low to high transition shall trigger an event. Bit 0 corresponds to digital input line 0, bit 1 corresponds to digital input line 1 and so on.

lowTransitionMask

This parameter specifies the digital input lines where a high to low transition shall trigger an event. Bit 0 corresponds to digital input line 0, bit 1 corresponds to digital input line 1 and so on.

msTimeout

This argument specifies the time (in ms) the function is willing to wait for the event. The timeout granularity depends on the system. If the specified time has passed and the event has not occurred, the function will return with an appropriate error code. The function will never timeout if a value of -1 is specified.

EXAMPLE

```
#include "tdrv024api.h"

TDRV024_HANDLE    hdl;
TDRV024_STATUS    result;

/*
** wait for high transition on input line 1 or 2, or
**      for low transition on input line 1
** timeout after 10 seconds
*/
result = tdrv024DigitalInputTransitionWait (hdl,
                                           (1 << 2) | (1 << 1),
                                           (1 << 1),
                                           10000);
if (result != TDRV024_OK)
{
    /* handle error */
}
else
{
    /* event has occurred */
}
```

RETURN VALUE

On success, TDRV024_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV024_ERR_INVALID_HANDLE	The specified device handle is invalid.
TDRV024_ERR_TIMEDOUT	Waiting for event has exceeded the specified time.
TDRV024_ERR_NOT_SUPPORTED	The specified device does not support digital input.
TDRV024_ERR_BUSY	The number of active waiting jobs is reached.

3.2.3 tdrv024DebouncerEnable

NAME

tdrv024DebouncerEnable – enable debouncer function for digital input

SYNOPSIS

```
TDRV024_STATUS tdrv024DebouncerEnable
(
    TDRV024_HANDLE    hdl,
    int                nsDebounceTime
)
```

DESCRIPTION

This function enables the debouncer function for all digital input lines and configures the debounce time.

PARAMETERS

hdl

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

nsDebounceTime

This value specifies the debounce time for digital input debouncing. The time is specified in nanoseconds, but the time will be scaled into steps depending on the used hardware (e.g. 8 ns for TQMC601).

The specified time is the time a signal must be stable before the state of the digital input line will appear in the digital input register.

EXAMPLE

```
#include "tdrv024api.h"

TDRV024_HANDLE    hdl;
TDRV024_STATUS    result;

/*
** enable digital input debouncer
** debounce time shall be 10 us
*/
result = tdrv024DebouncerEnable (hdl, 10000);
if (result != TDRV024_OK)
{
    /* handle error */
}
else
{
    /* successful */
}
```

RETURN VALUE

On success, TDRV024_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV024_ERR_INVALID_HANDLE	The specified device handle is invalid.
TDRV024_ERR_INVAL	Specified debounce time parameter is invalid.
TDRV024_ERR_NOT_SUPPORTED	The specified device does not support digital input.

3.2.4 tdrv024DebouncerDisable

NAME

tdrv024DebouncerDisable – disable debouncer function for digital input

SYNOPSIS

```
TDRV024_STATUS tdrv024DebouncerDisable
(
    TDRV024_HANDLE    hdl
)
```

DESCRIPTION

This function disables the debouncer function for all digital input lines.

PARAMETERS

hdl

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

EXAMPLE

```
#include "tdrv024api.h"

TDRV024_HANDLE    hdl;
TDRV024_STATUS    result;

/*
** disable digital input debouncer
*/
result = tdrv024DebouncerDisable (hdl);
if (result != TDRV024_OK)
{
    /* handle error */
}
else
{
    /* successful */
}
```

RETURN VALUE

On success, TDRV024_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV024_ERR_INVALID_HANDLE	The specified device handle is invalid.
TDRV024_ERR_NOT_SUPPORTED	The specified device does not support digital input.

3.3 Device Output Functions

Device output functions are supported if digital output lines are available at the device. If this function is called for a device without digital output lines the function will complete with an appropriate error code.

If an output function sets a value for a not present digital output line, the corresponding bit will be ignored.

Depending on the hardware the output configuration needs to be set in groups. In this case the configuration value must be set for all lines of the group to avoid unexpected configuration of digital output lines. The function *tdrv024DigitalOutputGetGroupMask()* can be used to determine a mask for a special digital output line, which shows all digital output lines of the corresponding group.

Please always keep in mind that all digital output lines of a group need to be set to the same output configuration. For more details please refer to the User Manual of the corresponding hardware.

3.3.1 tdrv024DigitalOutputGetGroupMask

NAME

tdrv024DigitalOutputGetGroupMask – get a mask of configuration dependent digital output lines

SYNOPSIS

```
TDRV024_STATUS tdrv024DigitalOutputGetGroupMask
(
    TDRV024_HANDLE    hdl,
    unsigned int       outputLine,
    unsigned int       *pDependentMask
)
```

DESCRIPTION

This function returns a mask showing configuration dependent digital output lines of a single digital output line. The returned mask depends on the hardware implementation of the supported modules.

PARAMETERS

hdl

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

outputLine

This value specifies the digital output line.

pDependentMask

This argument is a pointer to a 32 bit buffer where the dependent mask of digital output lines will be returned.

EXAMPLE

```
#include "tdrv024api.h"

TDRV024_HANDLE    hdl;
TDRV024_STATUS    result;
unsigned int       dependMask;

/*
** Get the dependent digital output lines of digital output line 5
*/
result = tdrv024DigitalOutputGetGroupMask (hdl, 5, &dependMask);
if (result != TDRV024_OK)
{
    /* handle error */
}
else
{
    /* function succeeded */
}
```

RETURN VALUE

On success, TDRV024_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV024_ERR_INVALID_HANDLE	The specified device handle is invalid.
TDRV024_ERR_INVALID	Invalid argument specified, e.g. the number of the specified output line is invalid.
TDRV024_ERR_NOT_SUPPORTED	The specified device does not support digital output.

3.3.2 tdrv024DigitalOutputConfigure

NAME

tdrv024DigitalOutputConfigure – configure digital output lines

SYNOPSIS

```
TDRV024_STATUS tdrv024DigitalOutputConfigure
(
    TDRV024_HANDLE      hdl,
    unsigned int         outputLineMask,
    unsigned int         outputMode
)
```

DESCRIPTION

This function configures the functional output mode of digital output lines.

PARAMETERS

hdl

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

outputLineMask

This value specifies a mask of digital output lines which shall be configured to the specified functional output mode.

Depending on the hardware the functional output mode must be configured in groups. The groups can be requested with the function *tdrv024DigitalOutputGetGroupMask()*.

outputMode

This argument specifies the functional output mode of the digital output lines. The following functional output modes are defined:

Functional Output Mode	Description
TDRV024_OUTPUT_MODE_HIGHSIDE	The digital output lines will be configured as a high-side output switch.
TDRV024_OUTPUT_MODE_LOWSIDE	The digital output lines will be configured as a low-side output switch.
TDRV024_OUTPUT_MODE_PUSH_PULL	The digital output lines will be configured as a push/pull output switch. <i>This mode is only available if the output lines can be configured individually.</i>
TDRV024_OUTPUT_MODE_HI_Z	The digital output lines will be configured in Hi-Z mode (outputs deactivated).

EXAMPLE

```
#include "tdrv024api.h"

TDRV024_HANDLE    hdl;
TDRV024_STATUS    result;

/*
** Configure digital output lines 0-7 to high-side switching interface
*/
result = tdrv024DigitalOutputConfigure (hdl,
                                         0x000000FF,
                                         TDRV024_OUTPUT_MODE_HIGHSIDE);
if (result != TDRV024_OK)
{
    /* handle error */
}
else
{
    /* function succeeded */
}
```

RETURN VALUE

On success, TDRV024_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV024_ERR_INVALID_HANDLE	The specified device handle is invalid.
TDRV024_ERR_INVALID	Invalid argument specified, e.g. an invalid outputMode.
TDRV024_ERR_INVALID_MASK	The specified outputLineMask is invalid, e.g. only a part of output lines of an output configuration group is selected.
TDRV024_ERR_NOT_SUPPORTED	The specified device does not support digital output.

3.3.3 tdrv024DigitalOutputWrite

NAME

tdrv024DigitalOutputWrite – write value to set the digital output lines

SYNOPSIS

```
TDRV024_STATUS tdrv024DigitalOutputWrite
(
    TDRV024_HANDLE      hdl,
    unsigned int         outputValue
)
```

DESCRIPTION

This function writes the specified value to set all digital output lines at once.

PARAMETERS

hdl

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

outputValue

This argument specifies the new output value. The value of each bit sets the state of a digital output line. Bit 0 corresponds to digital output line 0, bit 1 corresponds to digital output line 1 and so on.

Depending on the configured functional output mode the following state will be set at the corresponding output lines:

Bit	High-side Switch	Low-side Switch	Push/Pull Mode	Hi-Z Mode (deactivated)
Unset (0)	OFF	OFF	Ground	OFF
Set (1)	Vcc	Ground	Vcc	OFF

EXAMPLE

```
#include "tdrv024api.h"

TDRV024_HANDLE    hdl;
TDRV024_STATUS    result;

/*
** Set digital output lines 0-5 to "1", all other lines to "0"
*/
result = tdrv024DigitalOutputWrite (hdl, 0x0000003F);
if (result != TDRV024_OK)
{
    /* handle error */
}
else
{
    /* function succeeded */
}
```

RETURN VALUE

On success, TDRV024_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV024_ERR_INVALID_HANDLE	The specified device handle is invalid.
TDRV024_ERR_IO	The output register is locked by a watchdog failure. Execute the function <i>tdrv024WatchdogReset()</i> to reset the watchdog error.
TDRV024_ERR_NOT_SUPPORTED	The specified device does not support digital output.

3.3.4 tdrv024DigitalOutputWriteMask

NAME

tdrv024DigitalOutputWriteMask – write masked value to set specific digital output lines

SYNOPSIS

```
TDRV024_STATUS tdrv024DigitalOutputWriteMask
(
    TDRV024_HANDLE          hdl,
    unsigned int             outputValue,
    unsigned int             outputMask
)
```

DESCRIPTION

This function writes a masked value to set specific digital output lines.

PARAMETERS

hdl

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

outputValue

This argument specifies the new output value. The value of each bit sets the state of a digital output line. Bit 0 corresponds to digital output line 0, bit 1 corresponds to digital output line 1 and so on.

Depending on the configured functional output mode the following state will be set at the corresponding output lines:

Bit	High-side Switch	Low-side Switch	Push/Pull Mode	Hi-Z Mode (deactivated)
Unset (0)	OFF	OFF	Ground	OFF
Set (1)	Vcc	Ground	Vcc	OFF

outputMask

Only those bits in *outputValue* for which the corresponding bit position in this mask is set to 1 will be changed in the digital output value.

EXAMPLE

```
#include "tdrv024api.h"

TDRV024_HANDLE    hdl;
TDRV024_STATUS    result;

/*
** Set digital output lines 0-5 to "1",
** and digital output lines 6 and 7 to "0".
** All other lines shall keep their current state.
*/
result = tdrv024DigitalOutputWriteMask (hdl, 0x0000003F, 0x000000FF);
if (result != TDRV024_OK)
{
    /* handle error */
}
else
{
    /* function succeeded */
}
```

RETURN VALUE

On success, TDRV024_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV024_ERR_INVALID_HANDLE	The specified device handle is invalid.
TDRV024_ERR_IO	The output register is locked by a watchdog failure. Execute the function <i>tdrv024WatchdogReset()</i> to reset the watchdog error.
TDRV024_ERR_NOT_SUPPORTED	The specified device does not support digital output.

3.3.5 tdrv024DigitalOutputLineSet

NAME

tdrv024DigitalOutputLineSet – set a single digital output line

SYNOPSIS

```
TDRV024_STATUS tdrv024DigitalOutputLineSet
(
    TDRV024_HANDLE    hdl,
    int                outputLine
)
```

DESCRIPTION

This function sets a single digital output line. All other output lines are not affected. Depending on the functional output mode the output state will be set to:

High-side Switch	Low-side Switch	Push/Pull Mode	Hi-Z Mode (deactivated)
Vcc	Ground	Vcc	OFF

PARAMETERS

hdl

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

outputLine

This argument specifies the digital output line that shall be set to active state. A 0 specifies digital output line 0, 1 specifies digital output line 1, and so on.

EXAMPLE

```
#include "tdrv024api.h"

TDRV024_HANDLE    hdl;
TDRV024_STATUS    result;

/*
** Set digital output line 7
*/
result = tdrv024DigitalOutputLineSet (hdl, 7);
if (result != TDRV024_OK)
{
    /* handle error */
}
else
{
    /* function succeeded */
}
```

RETURN VALUE

On success, TDRV024_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV024_ERR_INVALID_HANDLE	The specified device handle is invalid.
TDRV024_ERR_INVAL	Invalid argument specified, e.g. the number of the specified output line is invalid.
TDRV024_ERR_IO	The output register is locked by a watchdog failure. Execute the function <i>tdrv024WatchdogReset()</i> to reset the watchdog error.
TDRV024_ERR_NOT_SUPPORTED	The specified device does not support digital output.

3.3.6 tdrv024DigitalOutputLineClear

NAME

tdrv024DigitalOutputLineClear – clear a single digital output line

SYNOPSIS

```
TDRV024_STATUS tdrv024DigitalOutputLineClear
(
    TDRV024_HANDLE      hdl,
    int                  outputLine
)
```

DESCRIPTION

This function unsets a single digital output line. All other output lines are not affected. Depending on the functional output mode the output state will be set to:

High-side Switch	Low-side Switch	Push/Pull Mode	Hi-Z Mode (deactivated)
OFF	OFF	Ground	OFF

PARAMETERS

hdl

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

outputLine

This argument specifies the digital output line that shall be cleared. A 0 specifies digital output line 0, 1 specifies digital output line 1, and so on.

EXAMPLE

```
#include "tdrv024api.h"

TDRV024_HANDLE    hdl;
TDRV024_STATUS    result;

/*
** Clear digital output line 7
*/
result = tdrv024DigitalOutputLineClear (hdl, 7);
if (result != TDRV024_OK)
{
    /* handle error */
}
else
{
    /* function succeeded */
}
```

RETURN VALUE

On success, TDRV024_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV024_ERR_INVALID_HANDLE	The specified device handle is invalid.
TDRV024_ERR_INVAL	Invalid argument specified, e.g. the number of the specified output line is invalid.
TDRV024_ERR_IO	The output register is locked by a watchdog failure. Execute the function <i>tdrv024WatchdogReset()</i> to reset the watchdog error.
TDRV024_ERR_NOT_SUPPORTED	The specified device does not support digital output.

3.3.7 tdrv024DigitalOutputStateRead

NAME

tdrv024DigitalOutputStateRead – read state of digital output value

SYNOPSIS

```
TDRV024_STATUS tdrv024DigitalOutputStateRead
(
    TDRV024_HANDLE          hdl,
    unsigned int             *pOutputValue,
    unsigned int             *pStatusFlags
)
```

DESCRIPTION

This function reads the status of digital output.

PARAMETERS

hdl

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

pOutputValue

This argument points to a 32 bit buffer where the output value will be returned, that has been set latest. This is not necessarily the current state at the digital output lines, for example if an error event has occurred. Bit 0 corresponds to the output value of digital output line 0, bit 1 corresponds to output value of digital output line 1 and so on.

pStatusFlags

This argument points to a 32 bit buffer where the error and state flag will be returned. The list below shows the defined state flags:

Error and State Flag	Description
TDRV024_STATEFLAG_WDACTIVE	A watchdog event has occurred and the watchdog state is active. The returned value by pOutputValue does reflect the last state before the watchdog got active.
TDRV024_STATEFLAG_FAULT	At least one digital output line (or group) signals an output fault. A detailed description of output fault can be found in the hardware User Manual. Faults can be acknowledge by reading the fault state value (<i>tdrv024DigitalOutputFaultRead()</i>). The returned value reflects the last digital output value this may differ from the state of digital output lines. Faulty digital output lines may turn off. Refer to the hardware User Manual to get more information how FAULT detection and handling works.

EXAMPLE

```
#include "tdrv024api.h"

TDRV024_HANDLE    hdl;
TDRV024_STATUS    result;
unsigned int       out_value;
unsigned int       out_state;

/*
** read current state of the output lines
*/
result = tdrv024DigitalOutputStateRead(hdl,
                                       &out_value,
                                       &out_state);
if (result != TDRV024_OK)
{
    /* handle error */
}
else
{
    printf("value:          0x%08X\n", out_value);
    if (out_state & TDRV024_STATEFLAG_WDACTIVE)
        printf("          Watchdog state is active\n");
}
}
```

RETURN VALUE

On success, TDRV024_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV024_ERR_INVALID_HANDLE	The specified device handle is invalid.
TDRV024_ERR_NOT_SUPPORTED	The specified device does not support digital output.

3.3.8 tdrv024DigitalOutputFaultRead

NAME

tdrv024DigitalOutputFaultRead – read and acknowledge of digital output fault state

SYNOPSIS

```
TDRV024_STATUS tdrv024DigitalOutputFaultRead
(
    TDRV024_HANDLE    hdl,
    unsigned int       *pOutputFault,
    unsigned int       ackMask
)
```

DESCRIPTION

This function reads the fault status of digital output lines. A mask allows to acknowledge the fault state of specific digital output lines.

PARAMETERS

hdl

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

pOutputFault

This argument points to a 32 bit buffer which returns a mask of digital output lines, where a fault state has been detected or may have been detected.

Bit 0 corresponds to the fault state value of digital output line 0, bit 1 corresponds to fault state value of digital output line 1 and so on.

Depending on the hardware, fault states may not be announced per digital output line, but in groups. In this case a fault event on any digital output line of a group will announce a fault state. In this case the fault state for all digital output lines of the group will be set and each of the fault bits must be acknowledged. Please refer to the hardware user Manual for more information about fault grouping.

ackMask

This argument specifies which output fault states shall be reset. The bit assignment corresponds to values returned by pOutputFault.

If a fault is still detected at the digital output line, the fault state for the corresponding digital output line will be set again immediately.

EXAMPLE

```
#include "tdrv024api.h"

TDRV024_HANDLE    hdl;
TDRV024_STATUS    result;
unsigned int       output_fault;

/*
** read current fault state of the output lines
** - clear pending fault states of digital output line 0 up to 7
*/
result = tdrv024DigitalOutputFaultRead (hdl, &output_fault, 0x000000FF);
if (result != TDRV024_OK)
{
    /* handle error */
}
else
{
    printf("fault state:          0x%08X\n", output_fault);
}
```

RETURN VALUE

On success, TDRV024_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV024_ERR_INVALID_HANDLE	The specified device handle is invalid.
TDRV024_ERR_NOT_SUPPORTED	The specified device does not support digital output.

3.3.9 tdrv024WatchdogEnable

NAME

tdrv024WatchdogEnable – enable digital output watchdog functionality

SYNOPSIS

```
TDRV024_STATUS tdrv024WatchdogEnable
(
    TDRV024_HANDLE      hdl
)
```

DESCRIPTION

This function enables the digital output watchdog functionality. The watchdog function is activated after the next write operation to the device.

Please remember that if the watchdog is enabled and no access to the output register occurs within the configured watchdog time, all outputs are set to the specified safe output state. To unlock the output register and leave the safe output state the function *tdrv024WatchdogReset()* must be executed.

PARAMETERS

hdl

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

EXAMPLE

```
#include "tdrv024api.h"

TDRV024_HANDLE  hdl;
TDRV024_STATUS  result;

...
```

```
...

/*
** enable digital output watchdog
*/
result = tdrv024WatchdogEnable (hdl);
if (result != TDRV024_OK)
{
    /* handle error */
}
else
{
    /* successful */
}
```

RETURN VALUE

On success, TDRV024_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV024_ERR_INVALID_HANDLE	The specified device handle is invalid.
TDRV024_ERR_NOT_SUPPORTED	The specified device does not support digital output.

3.3.10 tdrv024WatchdogDisable

NAME

tdrv024WatchdogDisable – disable digital output watchdog functionality

SYNOPSIS

```
TDRV024_STATUS tdrv024WatchdogDisable
(
    TDRV024_HANDLE      hdl
)
```

DESCRIPTION

This function disables the digital output watchdog functionality.

PARAMETERS

hdl

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

EXAMPLE

```
#include "tdrv024api.h"

TDRV024_HANDLE  hdl;
TDRV024_STATUS  result;

/*
** disable digital output watchdog
*/
result = tdrv024WatchdogDisable (hdl);
if (result != TDRV024_OK)
{
    /* handle error */
}
else
{
    /* successful */
}
```

RETURN VALUE

On success, TDRV024_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV024_ERR_INVALID_HANDLE	The specified device handle is invalid.
TDRV024_ERR_NOT_SUPPORTED	The specified device does not support digital output.

3.3.11 tdrv024WatchdogReset

NAME

tdrv024WatchdogReset – resets the digital output watchdog state

SYNOPSIS

```
TDRV024_STATUS tdrv024WatchdogReset
(
    TDRV024_HANDLE      hdl,
    unsigned int         mode,
    unsigned int         outputValue
)
```

DESCRIPTION

This function resets the digital output watchdog state and unlocks the protection of digital output write-operations initiated by the watchdog event occurrence.

PARAMETERS

hdl

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

mode

A set flag specifies how the output lines shall behave when leaving the watchdog state. One of the following flags must be used.

Mode	Description
TDRV024_WDMODE_STAYSAFE	After leaving the watchdog state the digital output lines will still be in safe state until a write access to the digital output value is executed. The digital output watchdog is active again, but timer is started with the next update of the output value.
TDRV024_WDMODE_PREVOUTPUT	After leaving the watchdog state the digital output lines will return to the value, which has been active before the watchdog event has occurred. The digital output watchdog is active again and the timer is started immediately.
TDRV024_WDMODE_NEWOUTPUT	After leaving the watchdog state the digital output lines will be set to the specified outputValue. The digital output watchdog is active again and the timer is started immediately.

outputValue

This value is only used if the flag TDRV024_WDMODE_NEWOUTPUT is selected, otherwise the value will not be used. The argument specifies the new output value which will be set when the watchdog state is reset. The value of each bit sets the state of a digital output line. Bit 0 corresponds to digital output line 0, bit 1 corresponds to digital output line 1 and so on.

Depending on the configured functional output mode the following state will be set at the corresponding output lines:

Bit	High-side Switch	Low-side Switch	Push/Pull Mode	Hi-Z Mode (deactivated)
Unset (0)	OFF	OFF	Ground	OFF
Set (1)	Vcc	Ground	Vcc	OFF

EXAMPLE

```
#include "tdrv024api.h"

TDRV024_HANDLE    hdl;
TDRV024_STATUS    result;

/*
** Reset digital output watchdog, keep safe state
*/
result = tdrv024WatchdogReset (hdl, TDRV024_WDMODE_STAYSAFE, 0);
if (result != TDRV024_OK)
{
    /* handle error */
}
else
{
    /* successful */
}
```

RETURN VALUE

On success, TDRV024_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV024_ERR_INVALID_HANDLE	The specified device handle is invalid.
TDRV024_ERR_NOT_SUPPORTED	The specified device does not support digital output.

3.3.12 tdrv024WatchdogConfigure

NAME

tdrv024WatchdogConfigure – configure the behavior of digital output watchdog functionality

SYNOPSIS

```
TDRV024_STATUS tdrv024WatchdogConfigure
(
    TDRV024_HANDLE          hdl,
    unsigned int             safeOutputValue,
    int                      msWatchdogTimeout
)
```

DESCRIPTION

This function configures how the digital output watchdog functionality behaves.

PARAMETERS

hdl

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

safeOutputValue

This argument specifies a safe digital output value, that will be used to set the output line in case of a watchdog event occurrence. Bit 0 corresponds to the output value of digital output line 0, bit 1 corresponds to output value of digital output line 1 and so on. By default all value will be set to 0, this is the reset value.

msWatchdogTimeout

This value specifies the maximum time which is allowed between accesses to the digital output, before the watchdog event occurs and sets the digital output lines into the specified safe state. The time is specified in milliseconds, but the time will be scaled into steps depending on the used hardware (e.g. 1 ms for TQMC604).

EXAMPLE

```
#include "tdrv024api.h"

TDRV024_HANDLE    hdl;
TDRV024_STATUS    result;

/*
** Configure digital output watchdog
**     safe state all output are 0
**     watchdog time 300 ms
**/
result = tdrv024WatchdogConfigure (hdl, 0x00000000, 300);
if (result != TDRV024_OK)
{
    /* handle error */
}
else
{
    /* successful */
}
```

RETURN VALUE

On success, TDRV024_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV024_ERR_INVALID_HANDLE	The specified device handle is invalid.
TDRV024_ERR_INVAL	Specified watchdog timeout parameter is invalid.
TDRV024_ERR_NOT_SUPPORTED	The specified device does not support digital output.

3.4 Global Functions

Global functions support global handling of the device. If functions are called with parameters referencing on not available functionality the function will complete with an appropriate error code.

3.4.1 tdrv024GlobalEventWait

NAME

tdrv024GlobalEventWait – wait for specified event

SYNOPSIS

```
TDRV024_STATUS tdrv024GlobalEventWait
(
    TDRV024_HANDLE    hdl,
    unsigned int       event,
    int                msTimeout
)
```

DESCRIPTION

This function waits until the specified event occurs. The first occurrence of the specified event will terminate the function.

PARAMETERS

hdl

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

event

This parameter specifies the event that must occur to terminate this function.

The list below shows the implemented event to wait for:

Event	Description
TDRV024_EVENT_OUTPUT_FAULT	The event of an output fault occurrence terminates the function. The fault must be handled by the application. For a detailed description of the fault reasons and how to handle the fault, please refer to the corresponding hardware User Manual.

msTimeout

This argument specifies the time (in ms) the function is willing to wait for the event. The timeout granularity depends on the system. If the specified time has passed and the event has not occurred, the function will return with an appropriate error code. The function will never timeout if a value of -1 is specified.

EXAMPLE

```
#include "tdrv024api.h"

TDRV024_HANDLE    hdl;
TDRV024_STATUS    result;

/*
** wait for detection of an output fault
** do not timeout
*/
result = tdrv024GlobalEventWait (hdl,
                                TDRV024_EVENT_OUTPUT_FAULT,
                                -1);
if (result != TDRV024_OK)
{
    /* handle error */
}
else
{
    /* event has occurred */
}
```

RETURN VALUE

On success, TDRV024_OK is returned. In the case of an error, the appropriate error code is returned by the function.

ERROR CODES

Error Code	Description
TDRV024_ERR_INVALID_HANDLE	The specified device handle is invalid.
TDRV024_ERR_TIMEDOUT	Waiting for event has exceeded the specified time.
TDRV024_ERR_NOT_SUPPORTED	The specified device does not support the specified event.
TDRV024_ERR_BUSY	The number of active waiting jobs is reached.

4 Diagnostic

If the TDRV024 does not work properly it is helpful to get some status information from the driver respective kernel. To get debug output from the driver enable the following symbols in 'tdrv024.c' by replacing "#undef" with "#define":

```
#define DEBUG_TDRV024
```

The Linux /proc file system provides information about kernel, resources, driver, devices, and so on. The following screen dumps display information of a correctly running TDRV024 driver (see also the proc man pages).

The example below shows the output with an installed TQMC601 and an installed TQMC604.

```
# tail -f /var/log/syslog /* or journalctl -f when calling modprobe */
linux kernel: TEWS Technologies - TDRV024 - Digital I/O - version 1.0.x (<release
date>)
linux kernel: TDRV024: Probe new device (vendor=0x1498, device=0xC25C)
linux kernel: TDRV024: Probe new device (vendor=0x1498, device=0xC259)
...
```

```
# lspci -v
```

```
...
03:00.0 Signal processing controller: TEWS Technologies GmbH Device c25c
Subsystem: TEWS Technologies GmbH Device c00a
Flags: fast devsel, IRQ 17
Memory at a1100000 (32-bit, non-prefetchable) [size=512]
Capabilities: [40] Power Management version 3
Capabilities: [48] MSI: Enable- Count=1/1 Maskable- 64bit+
Capabilities: [60] Express Endpoint, IntMsgNum 0
Capabilities: [100] Device Serial Number 00-00-00-00-00-00-00-00
Kernel driver in use: TEWS Technologies - TDRV024 - Digital IO -
Kernel modules: tdrv024drv

04:00.0 Signal processing controller: TEWS Technologies GmbH Device c259
Subsystem: TEWS Technologies GmbH Device c00a
Flags: fast devsel, IRQ 18
Memory at a1000000 (32-bit, non-prefetchable) [size=512]
Capabilities: [40] Power Management version 3
Capabilities: [48] MSI: Enable- Count=1/1 Maskable- 64bit+
Capabilities: [60] Express Endpoint, IntMsgNum 0
Capabilities: [100] Device Serial Number 00-00-00-00-00-00-00-00
Kernel driver in use: TEWS Technologies - TDRV024 - Digital IO -
Kernel modules: tdrv024drv
...
```

```
# cat /proc/devices
```

```
...
```

```
237 tdrv024drv
```

```
...
```

```
# cat /proc/interrupts
```

	CPU0	CPU1	CPU2	CPU3			
...							
17:	0	0	0	0	IO-APIC	17-fasteoi	TDRV024
18:	0	0	0	0	IO-APIC	18-fasteoi	TDRV024
...							

```
# cat /proc/iomem
```

```
...
```

```
8f800000-dfffffff : PCI Bus 0000:00
8f800000-8f9fffff : PCI Bus 0000:05
8fa00000-8fbfffff : PCI Bus 0000:09
90000000-9fffffff : 0000:00:02.0
a0000000-a0ffffff : 0000:00:02.0
a1000000-a11fffff : PCI Bus 0000:01
a1000000-a11fffff : PCI Bus 0000:02
a1000000-a10fffff : PCI Bus 0000:04
a1000000-a10001ff : 0000:04:00.0
a1000000-a10001ff : TDRV024
a1100000-a11fffff : PCI Bus 0000:03
a1100000-a11001ff : 0000:03:00.0
a1100000-a11001ff : TDRV024
```

```
...
```