MicroLAB Systems

Ultimate DSP Development Solutions

TI Third Party Network Member

DSP

TEXAS INSTRUMENTS

DIGITAL SIGNAL PROCESSING

TORNADO

Software Development Kit (TSDK)

for TORNADO DSP Boards for PC

for Windows 9x/NT/2000/XP
and Microsoft Visual C/C++ IDE

User's Guide

covers:
TSDK v.1.3

MicroLAB Systems Ltd
phone/fax: +7-(495)-900-6208 *E-mail: info@mlabsys.com* *WWW: http://www.mlabsys.com*

WARRANTY OBLIGATION

MicroLAB Systems, Ltd (MLS) warrants, all products for the lifetime of the product, that its products sold hereunder will at the time of shipment be free from defects in material and workmanship and will conform to MLS's applicable specifications or, if appropriate, to Customer's specifications previously accepted by MLS in writing. If products sold hereunder are not as warranted, MLS shall, at its option either refund the purchase price, or repair or replace the product, provided proof of purchase and written notice of nonconformance are received by MLS within one (1) year of date of purchase and provided said nonconforming products are, with MLS's written authorization, returned in protected shipping containers. MLS will pay for transporting the repaired or exchanged product to Customer. This warranty shall not apply to any products MLS determined to have been, by Customer otherwise, subjected to mishandling, misuse, neglect, improper testing, repair alteration, damage, assembly or processing that alters physical or electrical properties.

IMPORTANT NOTICE

This product has been designed for installation in a laboratory and scientific equipment in accordance with the specifications provided herein for normal temperature and environmental conditions. MLS assumes no liability if this product is used and/or installed in a different environment unless otherwise confirmed by a written certificate from MLS. Should this product be damaged due to a misuse in a different installation and/or environments, then this will result in a warranty termination with the corresponding service charges applied.

MLS's products are not designed for health and life-critical installations. Usage of MLS's products in such equipment and installation is strictly prohibited without written permission from MLS.

MLS has provided the best possible efforts to minimize the RF interference in the product line, however, the products may radiate RF energy, which may interfere and create noise for other electronic devices. The customer must provide his own efforts to further minimize this interference.

ITEMS OF LICENSE AGREEMENT

MLS reserves the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to MLS's terms and conditions of sale supplied at the time of order acknowledgment.

MLS warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with MLS's standard warranty. Testing and other quality control techniques are used to the extent MLS deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

MLS assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using MLS products and services (or their components) unless otherwise is confirmed by a written certificate from the MLS. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

Neither the hardware and/or software part of this product can be reversed engineered, retraced and/or modified for reproduction of either the electronic schematics, and/or of the construction, and/or of the functionality algorithm using either technique. Reproduction and/or broadcasting of information in MLS data books and/or data sheets is permissible only if reproduction is without alteration and is accompanied by a mandatory referencing to the MLS and all associated warranties, conditions, limitations, and notices. Reproduction of this document with alteration is an unfair and deceptive business practice. MLS is not responsible or liable for such altered documentation.

Purchasing this product denotes that the customer agrees with items of the license agreement as well as with the items of all applicable laws.

DOC: MLS-SPUG-090E revision 1-3
March 2006

About this Document

This user's guide contains description for *TORNADO* Software Development Kit (*TSDK*) from MicroLAB Systems Ltd.

This document does not include detail description neither for *TORNADO* DSP systems, nor for the corresponding software and hardware applications. To get the corresponding information please refer to the following original documentation:

1. ***TORNADO-3x. User's Guide.*** MicroLAB Systems, 2000.
2. ***TORNADO-54x. User's Guide.*** MicroLAB Systems, 2000.
3. ***TORNADO-6x. User's Guide.*** MicroLAB Systems, 2000.
4. ***TORNADO-P6x. User's Guide.*** MicroLAB Systems, 2004.
5. ***MIRAGE-510DX/UECMX User's Guide.*** MicroLAB Systems Ltd, 1999.

Warranty

The warranty period for all hardware and software products manufactured by MicroLAB Systems Ltd is a full lifetime warranty unless other is specified. MicroLAB Systems Ltd guarantees free of charge repair or replacement for the manufacturer caused damaged products during warranty period. Software updates will be sent free of charge to the customer during warranty period.

Product registration procedure

MicroLAB Systems strongly recommends that you register each of your purchased hardware/software product in order to get free product updates and free technical support within the warranty period.

The registration procedure is as easy as the following:

- Open the PRODUCT REGISTRATION FORM, which is contained in the REGISTER.TXT text file available either on the MicroLAB Systems 'Technical & Programming Guide' CD-ROM or from the MicroLAB Systems FTP-site. If you are unable to locate the REGISTER.TXT file, then call/email to MicroLAB Systems.
- Fill in the applicable fields of the PRODUCT REGISTRATION FORM. It is important that you will to specify your name, post address, phone/fax, email address, purchased product name and serial number, and the product reception date.
- Return the PRODUCT REGISTRATION FORM to MicroLAB Systems either via email, fax or regular mail.

Note, that all product purchased from MicroLAB Systems shall be registered within 90 days after the date of the shipment.

If you need assistance, documentation or information...

Should you need technical assistance for purchased MicroLAB Systems Ltd products, or if you want to order additional documentation, or if you want to get latest information about MicroLAB Systems Ltd products, please email, call, fax or post to MicroLAB Systems Ltd customer support service:

address: 83 Dubninskaya str, #612, 127591, Moscow, RUSSIA
MicroLAB Systems Ltd

phone/fax: +7-(095)-900-6208

information request: info@mlabsys.com

technical support: support@mlabsys.com

product registration: register@mlabsys.com

WWW: http://www.mlabsys.com

FTP: ftp://ftp.mlabsys.com

Trademarks

Contents

Chapter 1. Introduction

This chapter contains general description for *TORNADO* Software Development Kit (*TSDK*).

1.1 General Information

TSDK is a host PC software programming environment for for Windows 95/98/NT/2000/XP host PC platforms, which has been designed to create host PC applications for *TORNADO* PC plug-in DSP boards (for PCI and ISA-bus) from MicroLAB Systems Ltd.

TSDK provides universal software programming interface for all *TORNADO* PC plug-in DSP boards, and is recommended as the basic software development tool for development of user host applications for *TORNADO* DSP boards running under Windows 95/98/NT/2000/XP.

software components

TSDK comprises of the following software components:

- *TORNADO Configuration Utility*
- *TORNADO COFF Loader*
- *TORNADO Control Center*
- *TORNADO Host API Function Library*
- *UECMX Control Center* for universal emulation control daughter-card module for *TORNADO* DSP systems and *MIRAGE-510DX* emulator
- Windows 95/98/NT/2000/XP system drivers
- programming examples and demos.

supported TORNADO DSP boards for PC

The following *TORNADO* DSP boards from MicroLAB Systems Ltd are supported by *TSDK*:

- *TORNADO-3x* DSP boards for ISA-bus PC
- *TORNADO-54x* DSP boards for ISA-bus PC
- *TORNADO-6x* DSP boards for ISA-bus PC
- *TORNADO-P3x* DSP boards for PCI-bus.
- *TORNADO-P6x* DSP boards for PCI-bus.

user applications for Windows 95/98, Windows NT and Windows 2000/XP

TSDK allows to develop universal user applications for all *TORNADO* DSP systems, which will run under both Windows 95/98, Windows NT and Windows 2000/XP platforms. Specific features for Windows 95/98, Windows NT and Windows 2000/XP are automatically handled by *TORNADO Host API Function Library* and system drivers.

NOTE

TSDK has been designed to develop user applications for Windows 95/98/NT/2000/XP using Microsoft Visual C/C++ version.6.0 or later.

Although other C/C++ compilers for Windows 95/98/NT/2000/XP might be also compatible with *TSDK* , this was not tested and is not guaranteed.

1.2 Software distribution terms

TSDK is distributed free of charge and comes standard with all *TORNADO* DSP systems and other products from MicroLAB Systems Ltd.. Users are encouraged to register via supplied MicroLAB Systems registration form either via email or fax. Technical support is provided via email, phone or fax within 48 hours after reception of technical request.

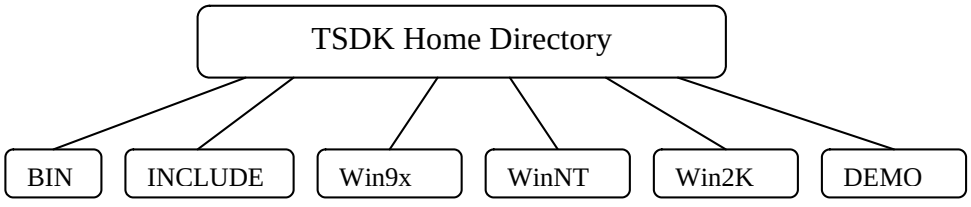
Chapter 2. Getting Started

TSDK has been designed by software engineers for software engineers, and all possible efforts have been done in order to make installation procedure transparent and predictable for the user. *TSDK* installation procedure is partially manual, which guarantees no uncontrolled update of system registry files. However, on the other hand, this requires that the user is familiar with Windows system structure on HDD of his PC.

This chapter contains description for *TSDK* source distribution kit and for *TSDK* installation procedure.

2.1 *TSDK* Source Distribution Kit

TSDK software comes on a distribution CD in a source unpacked form. Directory structure of *TSDK* source software distribution kit is shown at the figure below.



TSDK Home Directory directory is the home directory of *TSDK* source software distribution kit. This directory can be renamed by the user during installation process.

BIN\ subdirectory includes all binary executable components of *TSDK* software.

INCLUDE\ subdirectory includes files, which are required for compilation of user *TSDK* compatible application for Windows.

Win9x\, *WinNT*\ and *Win2K*\ subdirectories includes *TSDK* drivers for the corresponding host Windows platform.

DEMO\ subdirectory includes *TSDK* compatible demo applications for all supported *TORNADO* DSP boards for PCI and ISA-bus.

2.2 *TSDK* Installation

TSDK installation procedure is simple and straightforward. In order to keep complete control over installation procedure, there is intentionally no automatic *SETUP.EXE* executable as for all PC application software.

NOTE

Prior you will install *TORNADO* DSP board into your PC, copy *TSDK* source software distribution kit (as shown in section 2.1 earlier in this chapter) from a distribution CD to the HDD of your PC. Use standard *Windows Explorer* accessories utility or any other application to perform the copy operation.

Feel free to rename target *TSDK* installation directory on the HDD of your PC as you want. For simplicity, further after in this manual, we'll be assuming that the *C:\TSDK* subdirectory is used to keep the contents of *TSDK* source software distribution kit on the HDD of your PC.

After you have copied *TSDK* source software distribution kit from a distribution CD to the HDD of your PC, switch off the power of your PC, install your *TORNADO* DSP board(s) and reboot your PC. Proceed with the described below *TSDK* installation procedure, which corresponds to the Windows platform of your PC.

Specific TSDK installation for Windows 95/98

The following is a *TSDK* specific installation procedure for Windows 9x plug-and-play (PnP) platform:

- Boot your PC.
- ***In case you have installed at least one TORNADO DSP board for PCI-bus into your PC***, then make sure that the board is detected by the system and New Hardware Wizard dialog will appear. When the driver will be requested, browse to the *C:\TSDK\Win9x* subdirectory and select the .INF file. Proceed with the dialog and complete driver installation. Make sure that the driver has been installed successfully. You can check installation status by just going to the *My Computer -> Properties -> Device Manager* dialog and ensure that your installed *TORNADO* DSP board has been registered.
- ***In case you have installed only TORNADO DSP board(s) for ISA-bus into your PC***, then the driver installation procedure is manual since there is no PnP detection of installed ISA boards under Windows 9x platform. Do the following:
    - Go to the *C:\TSDK\Win9x* subdirectory and locate the *THDRV.VXD* file. You will need to check 'SHOW HIDDEN SYSTEM FILES" option in the *FOLDER OPTIONS->VIEW* dialog of *Windows Explorer* utility in order to perform this operation.
    - Copy *THDRV.VXD* file to the *C:\WINDOWS\SYSTEM* directory.
- Go to the *C:\TSDK\BIN* subdirectory and copy *TCC.EXE*, *TCOFF. EXE* and *THAPI.DLL* files to the *C:\WINDOWS\SYSTEM* directory.
- Proceed to the *"Common TSDK installation procedure for all Windows platforms"* subsection below in this section.

You do not need to restart Windows.

Specific TSDK installation for Windows 2000 and Windows XP

The following is a *TSDK* specific installation procedure for Windows 2000/XP plug-and-play (PnP) platform:

- Boot your PC.

- ***In case you have installed at least one TORNADO DSP board for PCI-bus into your PC***, then make sure that the board is detected by the system and New Hardware Wizard dialog will appear. When the driver will be requested, browse to the *C:\TSDK\Win2K* subdirectory and select the .INF file. Proceed with the dialog and complete driver installation. Make sure that the driver has been installed successfully. You can check installation status by just going to the *My Computer -> Properties -> Hardware ->Device Manager* dialog and ensure that your installed *TORNADO* DSP board has been registered.
- ***In case you have installed only TORNADO DSP board(s) for ISA-bus into your PC***, then the driver installation procedure is manual since there is no PnP detection of installed ISA boards under Windows 2000/XP platform. Do the following:
    - Go to the *C:\TSDK\Win2K* subdirectory and locate the *THDRV.SYS* file. You will need to check 'SHOW HIDDEN SYSTEM FILES" option in the *FOLDER OPTIONS->VIEW* dialog of *Windows Explorer* utility in order to perform this operation.
    - Copy *THDRV.SYS* file to the *C:\WINNT\SYSTEM32\DRIVERS* directory for Windows 2000 and into *C:\WINDOWS\SYSTEM32\DRIVERS* directory for Windows XP.
- Go to the *C:\TSDK\Win2K* subdirectory and locate *UECMX.SYS* file. You will need to check 'SHOW HIDDEN SYSTEM FILES" option in the *FOLDER OPTIONS->VIEW* dialog of *Windows Explorer* utility in order to perform this operation.
- Copy *UECMX.SYS* file to the *C:\WINNT\SYSTEM32\DRIVERS* directory for Windows 2000 and into *C:\WINDOWS\SYSTEM32\DRIVERS* directory for Windows XP.
- Go to the *C:\TSDK\BIN* subdirectory and copy *TCC.EXE*, *TCOFF. EXE* and *THAPI.DLL* files to the *C:\WINNT\SYSTEM32* directory for Windows 2000 and into the *C:\WINDOWS\SYSTEM32* directory for Windows XP.
- Proceed to the *"Common TSDK installation procedure for all Windows platforms"* subsection below in this section.

You do not need to restart Windows.

Specific TSDK installation for Windows NT Service Pack 6

The following is a *TSDK* specific installation procedure for Windows NT non plug-and-play platform, which does not automatically detects neither PCI no ISA-bus boards:

- Boot your PC.
- Make sure that you have Windows NT with Service Pack 6 or later installed. If not, then update your Windows NT.
- Go to the *C:\TSDK\WinNT* subdirectory and locate the *THDRV.SYS* and *UECMX.SYS* files. You will need to check 'SHOW HIDDEN SYSTEM FILES" option in the FOLDER OPTIONS->VIEW dialog of *Windows Explorer* utility in order to perform this operation.
- Copy *THDRV.SYS* and *UECMX.SYS* files to the *C:\WINNT\SYSTEM32\DRIVERS* directory.
- Go to the *C:\TSDK\BIN* subdirectory and copy *TCC.EXE*, *TCOFF. EXE* and *THAPI.DLL* files to the *C:\WINNT\SYSTEM32* directory.
- Proceed to the *"Common TSDK installation procedure for all Windows platforms"* subsection below in this section.

You do not need to restart Windows.

Common TSDK installation procedure for all Windows platforms

The following subsection describes final actions, which you will need to do in order to complete *TSDK* installation for all Windows platforms.

Note, that you first have to complete one of the specific TSDK installation procedure, which corresponds to your Windows platform, which are described earlier in this section.

Note that the below described actions are required for software engineers in order to design TSDK compatible applications. In case you are not a software design engineer and need just to run already compiled TSDK applications (for example you are a system integrator, etc), they you may not need to proceed with this subsection.

The final *TSDK* installation actions include are recommended for software design engineer in order to simplify usage of *TSDK* during software design process:

- Create shortcuts on the Windows desktop (or elsewhere inside Windows Start Menu) for *C:\TSDK\TSFG.EXE* and *C:\TSDK\UECMXCCW.EXE* utilities. Along with *TCC.EXE* and *TCOFF.EXE* utilities, these are the most common used utilities.
- Make sure that you have a simple and quick access to the Windows MS-DOS Prompt in order to run *TCC.EXE* and *TCOFF.EXE* command line utilities, which you should have already copied to the Windows system directory for easy calling.
- Open your Microsoft Visual C/C++ IDE. Go to the *Tools -> Options* configuration dialog and include the *C:\TSDK\INCLUDE* directory into the list of INCLUDE directories for INCLUDE files and for the LIBRARY files.

This finishes *TSDK* installation.

You must proceed to the “Configuring TSDK” section below in this chapter in case you have at least one *TORNADO* DSP board for ISA-bus board installed in your PC.

In case you have only *TORNADO* DSP board for PCI-bus board(s) installed in your PC, then you are completely done and ready to run precomplied *TSDK* demo & test applications and to design your own *TSDK* compatible applications.

Exceptional conditions for support of TORNADO DSP boards for PCI-bus under Windows 2000 and Windows XP

Starting from release 1.3 of *TSDK* software, there is a special automatic start-up test procedure for Windows 2000/XP, which is used to verify compliance of host PCI interface configuration of all installed *TORNADO* DSP boards for PCI-bus with the *TSDK* software.

This procedure is built into the *TSDK* low level system driver for Windows 2000/XP and starts automatically on the system boot. In case this procedure succeeds, then TSDK will start-up successfully, otherwise *TSDK* is aborted with the corresponding message is displayed during start of the first *TSDK* compatible application.

The following is a list of exceptional condition, which result in detection on either an abort or warning condition for *TSDK*:

- In case you have *TORNADO-P64xx* DSP boards for PCI-bus installed and any of these boards has the dual-port RAM area of host PCI-bus interface allocated into the system UMB (upper memory block, i.e. <1MB) area, then *TSDK* aborts.
- In case you have *TORNADO-P3x/P6xxxx* DSP boards for PCI-bus installed and any of these boards has the dual-port RAM area of host PCI-bus interface not allocated, then *TSDK* aborts.

- In case you any of *TORNADO* DSP boards for PCI-bus boards has incorrect allocation of I/O and reserved areas, then *TSDK* aborts.
- In case you have *TORNADO-P3x/P6xxxx* DSP boards for PCI-bus installed and any of these boards has the dual-port RAM area of host PCI-bus interface allocated into the system UMB (upper memory block, i.e. <1MB) area, then *TSDK* will start successfully, but the *TSDK* warning message will be displayed during start of first *TSDK* compatible application.

NOTE

It is recommended to configure the DPRAM area of host PCI-bus interface of all installed *TORNADO* DSP boards for PCI-bus to the 32-bit PCI memory space. This excludes confusions of system memory space allocation and guarantees correct operation of installed hardware.

2.3 Configuring *TSDK*

This section applies for those users, which have at least one *TORNADO* DSP board for ISA-bus installed in your PC. The user, who have only *TORNADO* DSP board for PCI-bus installed in your PC may skip this section.

Registering TORNADO boards for ISA-bus with TCFG.EXE utility

After *TSDK* installation procedure will complete, it is required to properly configure *TSDK* in order it could detect all *TORNADO* DSP board for ISA-bus, which are installed into your PC. This is performed by means of *TORNADO Configuration Utility* (*TCFG.EXE*), which is available in the *TC:\TSDK\BIN* subdirectory.

NOTE

It is required to register configurations for installed *TORNADO* DSP boards for ISA-bus only, since ISA-bus boards do not have the PnP feature.

All installed *TORNADO* DSP boards for PCI-bus are automatically detected and registered within *TSDK*.

In order to register configurations for ISA-bus *TORNADO* DSP systems, which are installed into your PC, run *TCFG.EXE* and follow intuitive and simple interface of this software utility. You will need to “add’ the board at specific address with specific interrupt request line and *UECMX* emulator module allocated at specific address. All these parameters are entered via the TCFG “Add ISA board” dialog.

NOTE

In case the installed *TORNADO* DSP boards for ISA-bus are not registered within *TSDK*, then it is not possible to access these *TORNADO* DSP boards from the *TSDK* compatible applications.

Configuring the UECMX emulator module with UECMXCCW.EXE utility

In case you have the *UECMX* emulator module installed onto your *TORNADO* DSP board for ISA-bus, then you may want to get access to control the functions of this module via windows base application rather than via the command-line *TCC.EXE* utility.

UECMX emulator module allows many advanced features and is directly controlled using *UECMX Control Center for Windows* (*UECMXCCW.EXE*) software utility, which is available in the *TC:\TSDK\BIN* subdirectory.

In order to allow *UECMXCCW.EXE* software utility to detect all installed *UECMX* modules, you must first advise I/O base addresses for all installed *UECMX* module. This is done by checking the corresponding I/O base address in the *UECMXCCW.EXE* utility dialog. After the *UECMXCCW.EXE* utility detects all *UECMX* modules, you will gain complete control over the functions of these *UECMX* modules using simple intuitive interface of the *UECMXCCW.EXE* utility.

2.4 *TSDK* Demo Applications

TSDK software comes with a complete set of demo & test samples for all types of supported *TORNADO* board. You will find the demo samples in the *C:\TSDK\DEMO* directory.

NOTE

Make sure to run all *TSDK* demo applications for every of your installed *TORNADO* board in order to ensure that each board is visible within the *TSDK* environment and functions properly.

Not only the *TSDK* demo & test applications can be used to test the installed *TORNADO* DSP boards, they are also a good start point for designing your own *TSDK* compatible application. *TSDK* demo & test applications come in source code and cover all aspects of *TORNADO* board functionality.

2.5 *TSDK* Software Update Procedure

We recommend you to keep your *TSDK* software up to date in order to ensure that you have the latest *TSDK* utilities, latest *TSDK* API functions support and no known software bugs.

TSDK update procedure is very simple:

- Download the latest release of *TSDK* software from MicroLAB Systems FTP site (ftp://ftp.mlabsys.com/sft/tsdk).
- Unzip the downloaded *TSDK* archive into any temporary subdirectory.
- Remove contents of your current *C:\TSDK* directory. Instead of removing the old contents of *C:\TSDK* directory, you may want to rename the directory and keep the previous release for your archive.
- Copy the contents of downloaded archive into the *C:\TSDK* directory.
- ***For Windows 9x users:***
    - ☐ Go to the *C:\TSDK\Win9x* subdirectory and locate the *THDRV.VXD* file. You will need to check 'SHOW HIDDEN SYSTEM FILES" option in the *FOLDER OPTIONS->VIEW* dialog of *Windows Explorer* utility in order to perform this operation.
    - ☐ Copy *THDRV.VXD* file to the *C:\WINDOWS\SYSTEM* directory.
    - ☐ Go to the *C:\TSDK\BIN* subdirectory and copy *TCC.EXE*, *TCOFF. EXE* and *THAPI.DLL* files to the *C:\WINDOWS\SYSTEM* directory.
- ***For Windows 2000 and Windows XP users:***
    - ☐ Go to the *C:\TSDK\Win2K* subdirectory and locate the *THDRV.SYS* file. You will need to check 'SHOW HIDDEN SYSTEM FILES" option in the *FOLDER OPTIONS->VIEW* dialog of *Windows Explorer* utility in order to perform this operation.
    - ☐ Copy *THDRV.SYS* file to the *C:\WINNT\SYSTEM32\DRIVERS* directory for Windows 2000 and into *C:\WINDOWS\SYSTEM32\DRIVERS* directory for Windows XP.
    - ☐ Go to the *C:\TSDK\Win2K* subdirectory and locate the *UECMX.SYS* file. You will need to check 'SHOW HIDDEN SYSTEM FILES" option in the *FOLDER OPTIONS->VIEW* dialog of *Windows Explorer* utility in order to perform this operation.
    - ☐ Copy *UECMX.SYS* file to the *C:\WINNT\SYSTEM32\DRIVERS* directory for Windows 2000 and into *C:\WINDOWS\SYSTEM32\DRIVERS* directory for Windows XP.
    - ☐ Go to the *C:\TSDK\BIN* subdirectory and copy *TCC.EXE*, *TCOFF. EXE* and *THAPI.DLL* files to the *C:\WINNT\SYSTEM32* directory for Windows 2000 and into the *C:\WINDOWS\SYSTEM32* directory for Windows XP.
- ***For Windows NT users:***
    - ☐ Go to the *C:\TSDK\WinNT* subdirectory and locate the *THDRV.SYS* and *UECMX.SYS* files. You will need to check 'SHOW HIDDEN SYSTEM FILES" option in the FOLDER OPTIONS->VIEW dialog of *Windows Explorer* utility in order to perform this operation.
    - ☐ Copy *THDRV.SYS* and *UECMX.SYS* files to the *C:\WINNT\SYSTEM32\DRIVERS* directory.
    - ☐ Go to the *C:\TSDK\BIN* subdirectory and copy *TCC.EXE*, *TCOFF. EXE* and *THAPI.DLL* files to the *C:\WINNT\SYSTEM32* directory.
- If there are new *TSDK* utilities included with the new *TSDK* release (check with the enclosed *TSDK* release text file), then create shortcuts for these utilities. You may need to make specific installation for these new utilities as it may be described in the enclosed *TSDK* release text file.
- Run *TCFG.EXE* and *TCC.EXE* applications in order to ensure that update has completed successfully.

2.6 Uninstall Procedure

Since there is no automatic *TSDK* installation procedure, then there is no automatic *TSDK* uninstall procedure. However, in case you do not need to use *TSDK* anymore, then just uninstall all *TORNADO* boards, and if you wish, then just remove the *C:\TSDK* form the HDD of your PC. *TSDK* will be not activated in case no *TORNADO* DSP boards are installed.

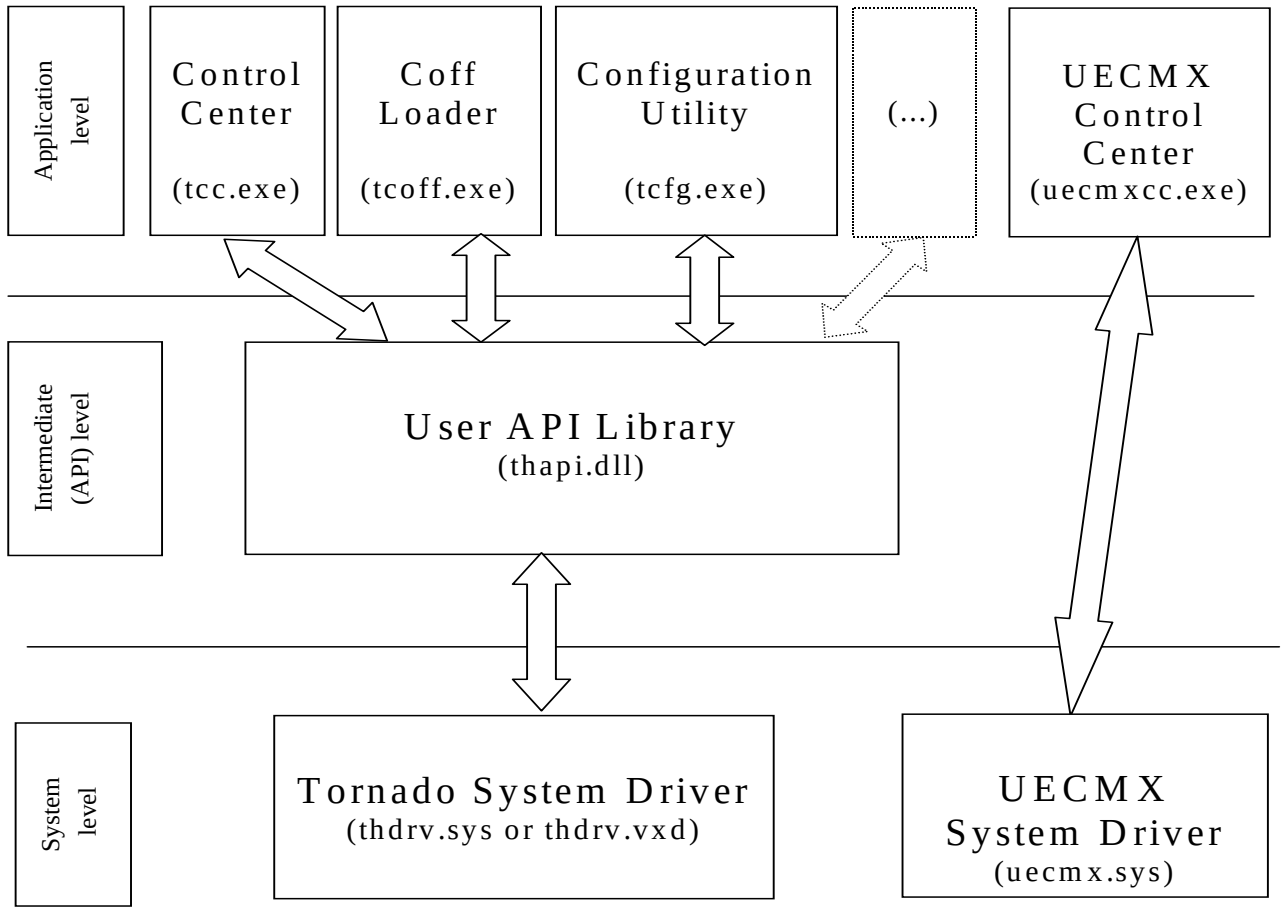
Chapter 3. *TSDK* Software Structure

This chapter contains description for *TSDK* software structure.

3.1 *TSDK* Software Components

TSDK is the multi-level software system for Windows. It comprises the following components:

- low level system drivers for Windows (*THDRV.SYS* and *THDRV.VXD*)
- intermediate level, which is actually the *API Function Library (THAPI.DLL)*
- application level, which comprises the enclosed *TSDK* software utilities (*TCC.EXE*, *TCOFF.EXE*, *TCFG.EXE*, *UECMXCCW.EXE*) and user designed *TSDK* compatible applications.



system level

TSDK system level includes Windows drivers for *TORNADO* DSP systems and *UECMX* emulator DCM. These drivers perform automatic hardware detection and communicate with installed hardware on requests from *TSDK Host API Function Library*.

Host API Function Library

TSDK Host API Function Library is actually an intermediate level of *TSDK* and shall be used by all applications at the upper level for *TORNADO* and *UECMX* control and data transmission between host PC and DSP environment.

TSDK Host API Function Library has been designed as dynamic link library (DLL) with uniform board programming interface and software portability in the mind, i.e. every library function allows to operate with every *TORNADO* DSP system available in case this function is applicable for particular *TORNADO* DSP system.

TSDK Host API Function Library supports multi-thread applications and features embedded locking mechanism in order exclude collisions when multiple applications need to access the same installed *TORNADO* and *UECMX* hardware. If different threads select different *TORNADO* and/or *UECMX* hardware, then the *TSDK Host API Function Library* will keep this correspondence and all further hardware access calls will be assumed to provide access to the pre-selected boards.

TSDK software utilities

TSDK software utilities have been designed to configure *TSDK* software environment, and get access to installed *TORNADO* and/or *UECMX* hardware via command line and multi-window user interface.

TSDK software utilities comprise of the following utilities:

- *TORNADO Configuration Utility (TCFG.EXE)*
- *TORNADO COFF Loader (TCOFF.EXE)*
- *TORNADO Control Center (TCC.EXE)*
- *UECMX Control Center (UECMXCCN.EXE* and *UECMXCCW.EXE)*.

TORNADO Configuration Utility

TORNADO Configuration Utility (*TCFG.EXE*), shall be used in order to view and enter updates for hardware configuration of installed *TORNADO* DSP board(s). *TORNADO Configuration Utility* features simple and intuitive windows interface.

TORNADO Configuration Utility allows to view current hardware configuration for all installed *TORNADO* DSP board(s) and allows to modify configurations for installed *TORNADO* DSP board(s) for ISA-bus, which do not have the PnP feature. All installed *TORNADO* DSP boards for PCI-bus are automatically detected and registered within *TSDK* and do not allow to modify their hardware configuration via *TORNADO Configuration Utility*.

CAUTION

Once installed *TORNADO* DSP board for ISA-bus is not registered within *TSDK*, then it is not possible to access this *TORNADO* DSP board from the *TSDK*-based host applications.

CAUTION

Each installed *TORNADO* DSP board for ISA-bus and PCI-bus is recognized by *TSDK* using unique board index, which is automatically assigned by *TSDK* and is displayed by *TORNADO Configuration Utility*.

TORNADO Control Center command line utility

TORNADO Control Center (*TCC.EXE*) is a command line software utility for Windows and allows delivers easy and powerful user control for installed *TORNADO* hardware.

TORNADO Control Center must be invoked from command line prompt with up to ten command line options:

TCC.EXE [-b#] [- option1] [- option2] [- option3]...

where # defines board index for particular *TORNADO* DSP board.

TORNADO Control Center is upward compatible with the corresponding former DOS command-line 'control center' utility for each *TORNADO* product line (*TORNADO-3x/54x/6x/P6x/etc*). A list of valid command line options is available for either from the user's guide for *TORNADO* DSP system, or on-line when *TORNADO Control Center* is invoked without any command line option.

TORNADO COFF Loader command line utility

TORNADO COFF Loader (*TCOFF.EXE*) is a command line software utility for Windows and allows to load DSP application in TI COFF file format into the DSP environment of installed *TORNADO* DSP system via host PC interface of *TORNADO* DSP board without utilizing the JTAG emulator.

TORNADO COFF Loader must be invoked from command line prompt with up to five command line options specified simultaneously:

TCOFF.EXE [-b#] [- option1] [- option2] [- option3]...

where # defines board index for particular *TORNADO* DSP board.

TORNADO COFF Loader is upward compatible with former DOS command-line 'COFF loader' utility for every *TORNADO* product line (*TORNADO-3x/54x/6x/P6x/etc*). A list of valid command line options is available for either from the user's guide for *TORNADO* DSP board, or on-line when *TORNADO COFF Loader* is invoked without any command line option.

UECMX Control Center

UECMX Control Center software utilities for Windows shall be used in order to control the *UECMX* emulator DCM for *TORNADO* DSP board and *MIRAGE-510DX* JTAG/MPSD emulators.

UECMX Control Center utilities include command line utility (*UECMXCCN.EXE*) and window application (*UECMXCCW.EXE*) with simple and intuitive window interface.

Command line *UECMX Control Center* (*UECMXCCN.EXE*) must be invoked from command line prompt with up to five command line options:

UECMXCCN.EXE [-pXXX] [- option1] [- option2] [- option3]...

where *XXX* defines the I/O base address board for particular *UECMX* DCM.

Command line *UECMX Control Center* (*UECMXCCN.EXE*) is upward compatible with former DOS command-line 'control center' utility for *UECMX/MIRAGE-510DX*. A list of valid command line options is available for either from the user's guide for *UECMX/MIRAGE-510DX*, or on-line when *UECMX Control Center* is invoked without any command line option.

Chapter 4. API Function Library

This chapter contains description of all functions from the API Function Library, which shall be used in user applications in order to interface to *TSDK*.

4.1 Types definition

The following types are defined and used by *TSDK* and user API library. It is strongly recommended for user to use them too.

```
typedef     unsigned long     ULONG,*PULONG;
typedef     unsigned short    USHORT,*PUSHORT;
typedef     unsigned char     UCHAR,*PUCHAR;
typedef     char              CHAR,*PCHAR;
typedef     void              VOID,*PVOID;
typedef     void              *HANDLE;
typedef     unsigned char     BOOLEAN;
typedef     ULONG             TH_BOARD_TD,*PTH_BOARD_TD;
```

Board information structure declared as following:

```
typedef     struct{

      ULONG       hw_brd_id;  // hardware device id
      ULONG       io_addr;    // ISA config io space base addr
      ULONG       int_line;   // configured interrupt line

}               TH_BRD_INFO_TD,*PTH_BRD_INFO_TD;
```

Interrupt registers set structure declared as following:

```
typedef     struct{

      UCHAR       type;       // the type of board
      ULONG       int_mask;   // mask of interrupt sources
      union
        {
            struct
            {
                  ULONG pcic_intcsr;      // interrpt control/status
                                          // register
                  ULONG pcic_inmbox;      // input mailbox register
                  UCHAR hif_is;           // HIF interrupt status
                                          // register
            }pci;

            struct
            {
                  UCHAR ctrl;
```

```
            }isa;
        }u;
} TH_INTREGS_TD,*PTH_INTREGS_TD;
```

It is assumed that all API functions have `TH_` prefix characters and structure type definitions have `_TD` postfix characters.

4.2 General functions

General functions are mainly purposed to perform auxiliary action to initialize boards to work properly, to get necessary information about boards and so on. These action aren't directly connected with access functions. They simply tune TORNADO Library and System TORNADO Driver to work properly with a user selected board and force TORNADO boards to get ready for actual processing. General information functions are included in this subsection also.

General functions are listed below:

Function	Description
TH_area_info	Gets a host interface area address and its descriptor
TH_board_close	Closes the selected board
TH_board_id	Gets the board hardware identificator
TH_board_info	Retrieves information about all detected boards
TH_board_lock	Locks the selected board
TH_board_name	Gets the selected board stringable name
TH_board_open	Opens the selected board
TH_board_select	Selects a board
TH_board_total	Gets the total number of detected boards
TH_board_type	Checks the type of a board (ISA or PCI)
TH_board_unlock	Unlocks the selected board
TH_connect_interrupt	Connects to interrupt line
TH_interrupt_line	Gets interrupt line of the selected board
TH_interrupt_regs	Gets set of interrupt registers

TH_interrupt_regs_clear	Clears interrupt registers buffer
TH_mapmem_view	Maps shareble memory for PCI mastering
TH_unmapmem_view	Unmaps shareable memory

4.3 Error control functions

Error control functions are required in order to perform run-time error control. Error control functions are listed below:

Function	Description
TH_err_message	Gets error code corresponding string
TH_flag_err	Returns the first occured error code
TH_last_err	Returns last error code

CAUTION

Each function of *TSDK* API function library returns the exit error code, which is the negative value. However, some functions also return the requested data, which is a positive value. The user application can recognize between the returned error code or requested data by means of analyzing the sign of returned value.

Note that in case of success, each function of *TSDK* API function library returns either *TH_OK* code (which is defined as zero) or requested data (which is a positive value), otherwise negative nonzero error code is returned.

The following is an example of an effective usage of these functions inside the console application.

```
…
#include "th.h"
...
int err;
ULONG addr;  CHAR name[20];
TH_BOARD_DESCR_TD selected_index;
...
err = TH_area_info(TH_PCI_HIF_AREA,&addr,name);
if(err){
    printf(TH_err_message(err));
    if(TH_INTERNAL_ERR == err){
    // some special action
    }
}
```

```
else
    printf("  HIF/HPI area     : 0x%08lx (%s)\n",addr & ~0x3L,name);
...
// Let's think that the index corresponds to PCI board
selected_index = 0;


TH_board_select(selected_index);    //supposing no errors
TH_board_open();                    // no errors
TH_clr_dpram_err();                 // failed
TH_set_smp_mode_spa();              // no errors
...
// check for an error here
if(err = TH_last_err())
 {    // will return TH_OK because last called function has executed
      // successfully
   printf(TH_err_message(err));
   return;
 }
if(err = TH_flag_err())
 {  // will indicate that the error has occured somewhere
   printf(TH_err_message(err));
   return;
 }
...
```

4.4 SMP access functions

SMP access functions are listed below:

Function	Description
TH_set_smp_dpram_area	Sets SMP to DPRAM
TH_set_smp_dpsem_area	Sets SMP to DPSEM
TH_set_smp_mode_dpa	Sets dual page SMP access mode
TH_set_smp_mode_spa	Sets single page SMP access mode
TH_set_smp_off	Sets off SMP segment
TH_set_smp_page	Writes data to SMP page register
TH_set_smp_ptr	Gets virtual pointer to specified SMP area
TH_set_smp_segm	Sets SMP segment to some UMB address
TH_smp_area_data	Checks which of SMP areas is selected

TH_smp_mode_data	Checks SMP mode
TH_smp_page_data	Ges data from SMP page register
TH_smp_segm	Gets SMP segment currently set
TH_smp_segm_table	Lists possible SMP segments to be used

4.5 DPRAM/DPSEM access functions

DPRAM/DPSEM functions are listed below:

Function	Description
TH_clr_dpram_err	Clears DPRAM error, if any
TH_clr_dpram_err_ie	Clears DPRAM interrupt enable error, if any
TH_dpram_data	Reads data from DPRAM
TH_dpram_err_data	Checks whether DPRAM error is set
TH_dpram_err_ie_data	Checks whether DPRAM interrupt enable error is set
TH_dpram_ie_data	Checks if DPRAM interrupt enable is set
TH_dpram_irq_data	Gets DPRAM data
TH_dpram_len	Gets DPRAM length
TH_dpsem_data	Reads data from DPSEM
TH_set_dpram	Writes data to DPRAM
TH_set_dpram_err_ie	Enables interrupt on DPRAM error
TH_set_dpram_ie_off	Disables DPRAM interrupt
TH_set_dpram_ie_on	Enables DPRAM interrupt
TH_set_dpsem	Writes data to DPSEM

4.6 Shared Bus access functions

Shared Bus access functions are listed below:

Function	Description
TH_clr_sb_err	Clears SB error, if any
TH_sb_ack_data	Returns value for SB_ACK (ISA-bus boards only)
TH_sb_ccl_data	Returns value for SB_CCL bits of CONTROL REGISTER (ISA-bus only)
TH_sb_err_data	Checks SB error (ISA-bus only)
TH_sb_err_ie_data	Returns value for interrupt enable on SB ERROR (ISA-bus only)
TH_sb_glock_data	Returns value for SB_GLOCK bit of CONROL REGISTER (ISA-bus only)
TH_sb_lock_data	Returns value for SB_LOCK bit of CONROL REGISTER(ISA-bus only)
TH_set_sb_ccl_byte	Sets 8-bit SB cycle (ISA-bus only)
TH_set_sb_ccl_dword	Sets 32-bit SB cycle (ISA-bus only)
TH_set_sb_ccl_word	Sets 16-bit SB cycle (ISA-bus only)
TH_set_sb_err_ie_off	Disable interrupt enable on SB ERROR (ISA-bus only)
TH_set_sb_err_ie_on	Disable interrupt enable on SB ERROR(ISA-bus only)
TH_set_sb_glock_off	Disable SB_GLOCK feature (ISA-bus only)
TH_set_sb_glock_on	Enable SB_GLOCK feature(ISA-bus only)
TH_set_sb_lock_off	Disable SB_LOCK feature (ISA-bus only)
TH_set_sb_lock_on	Enable SB_LOCK feature (ISA-bus only)

4.7 HPI access functions

HPI access functions are listed below:

Function	Description
TH_clr_hpi_err	Clears HPI error, if any
TH_clr_hpi_err_ie	Clears HPI interrupt enable error, if any
TH_clr_hpi_hint_ie	Disable interrupt on HINT
TH_clr_hpic_hint	Clear HINT
TH_clr_hpic_xhpia	Clear extended HPIA register
TH_hpi_disable_data	Return current value for the HPI_DISABLE
TH_hpi_err_data	Checks for HPI error
TH_hpi_err_ie_data	Checks for HPI interrupt enable error
TH_hpi_hint_data	Returns current HINT value
TH_hpi_hint_ie_data	Returns current value for interrupt enable on HINT
TH_hpi_ie_frg_data	Returns current value for HPI_IE_FRG flag register
TH_hpi_init	Initializes HPI to work properly
TH_hpia_reset	Clear HPIA
TH_hpic_dspint_data	Returns value for DSPINT bit of HPIC
TH_hpic_fetch_data	Returns value for FETCH bit of HPIC
TH_hpic_hint_data	Returns value for HINT bit of HPIC
TH_hpic_hrdy_data	Returns value for HRDY bit of HPIC
TH_hpic_hwob_data	Returns value for BOB bit of HPIC (T6x/TP6x only)
TH_set_hpi_disable	Disable HPI (TORNADO-5402 only)
TH_set_hpi_enable	Enable HPI (TORNADO-5402 only)
TH_set_hpi_err_ie	Enable interrupt on HPI access timeout
TH_set_hpi_hint_ie	Enable interrupt on HINT

TH_set_hpic_bob	Set BOB bit of HPIC (T54x only)
TH_set_hpic_dspint	Set DSPINT bit of HPIC
TH_set_hpic_fetch	Set FETCH bit of HPIC
TH_set_hpic_hwob	Set HWOB bit of HPIC (T6x/TP6x only)
TH_set_hpic_xhpia	Set extended HPIA register

4.8 ECC access functions

UECM/ECC functions are listed below:

Function	Description
TH_ecc_clr_err	Clears ECC error, if any
TH_ecc_err_data	
TH_ecc_irq_data	
TH_ecc_off	
TH_ecc_reset	
TH_ecc_stat_rg	Returns ECC STAT register contents
TH_emu_default	
TH_emu_io_baddr	Returns EMU base address
TH_emu_path_data	
TH_set_emu_io_baddr	Sets EMU base address
TH_set_emu_mdsp	
TH_set_emu_reset	
TH_set_emu_sel_ecc	
TH_set_emu_sel_rg	

TH_set_emu_sel_xemu	
TH_set_emu_xdsp	
TH_uecm_io_baddr_table	Gets table of UECM base addresses

4.9 Registers access functions

Registers access functions are listed below:

Function	Description
TH_ctrl_rg	Gets CONTROL register contents
TH_emu_sel_rg	Gets EMU SELECTOR register contents
TH_fdata_rg	Gets FDATA register contents
TH_frg_data	Gets FRG register contents
TH_fsel_rg	Gets FSEL register contents
TH_hm_rq0_rg	Gets HM_RQ0 register contents
TH_hm_rq1_rg	Gets HM_RQ1 register contents
TH_hpia_rg	Gets HPIA register contents
TH_hpic_rg	Gets HPIC register contents
TH_hpid_ainc_rg	Gets HPID_AINC register contents
TH_hpid_rg	Gets HPID register contents
TH_im_rg	Gets HIF_IM register contents
TH_is_rg	Gets HIF_IS register contents
TH_pcic_intcsr_rg	Gets PCI contoller INTCSR register contents
TH_pcic_fifo_rg	Gets PCI contoller FIFO register contents
TH_pcic_imbx_rg	Gets PCI contoller IMBX register contents

TH_pcic_mbef_rg	Gets PCI contoller MBEF register contents
TH_pcic_mcsr_rg	Gets PCI contoller MCSR register contents
TH_pcic_mrar_rg	Gets PCI contoller MRAR register contents
TH_pcic_mrtc_rg	Gets PCI contoller MRTC register contents
TH_pcic_mwar_rg	Gets PCI contoller MWAR register contents
TH_pcic_mwtc_rg	Gets PCI contoller MWTC register contents
TH_pcic_ombx_rg	Gets PCI contoller OMBX register contents
TH_set_ctrl_rg	Sets CONTROL register with data specified
TH_set_fdata_rg	Sets FDATA register with data specified
TH_set_frg_verif	Sets FRG register with data specified
TH_set_fsel_rg	Sets FSEL register with data specified
TH_set_fsel_rg_verif	Sets FSEL register with data specified and checks
TH_set_hm_rq0_rg	Sets HM_RQ0 register with data specified
TH_set_hm_rq1_rg	Sets HM_RQ1 register with data specified
TH_set_hpia_rg	Sets HPIA register with data specified
TH_set_hpic_rg	Sets HPIC register with data specified
TH_set_hpid_ainc_rg	Sets HPID_AINC register with data specified
TH_set_hpid_rg	Sets HPIA register with data specified
TH_set_im_rg	Sets HIF_IM register with data specified
TH_set_is_rg	Sets HIF_IS register with data specified
TH_set_pcic_intcsr_rg	Sets PCI contoller INTCSR register with specified data
TH_set_pcic_fifo_rg	Sets PCI contoller FIFO register with specified data
TH_set_pcic_mcsr_rg	Sets PCI contoller IMBX register with specified data

TH_set_pcic_ombx_rg	Sets PCI contoller MBEF register with specified data
TH_set_smp_rg	Sets SMP register with specified data
TH_smp_rg	Gets SMP register contents

4.10 Other functions

All functions not included in the previous sections are listed below:

Function	Description
TH_clr_mh_rq	Clears Master-to-Host request.
TH_dram_len	Gets DRAM area length
TH_dsp_amen_data	Returns current value for PCI-bus mastering enable from the DSP environment (PCI-bus boards with PCI-bus mastering only).
TH_dsp_hpi_disable_data	Returns current value for HPI disable bit (TORNADO-5402 only).
TH_dsp_mlock_data	Checks if DSP_MLOCK mode is set
TH_dsp_mreset_data	Checks if DSP_MRESET mode is set
TH_dsp_pd_data	Checks if DSP_PD mode is set
TH_dsp_stat_frg_data	Gets DSP STAT FRG data
TH_init	Initializes CONTROL and SMP registers
TH_mh_rq_data	Returns current state for Master-to-Host request
TH_mh_rq_ie_data	Returns current state for interrupt enable on Master-to-Host request
TH_mreset_data	Checks if DSP is in reset state
TH_mreset_mode_data	Returns current state for mode selector for DSP reset (PCI-based boards only)
TH_nvram_read	Reads NVRAM address

TH_nvram_write	Writes data specified to NVRAM address
TH_read_byte	Reads byte from specified host interface area
TH_read_dword	Reads 32-bit word from specified host interface area
TH_read_word	Reads 16-bit word from specified host interface area
TH_set_hm_rq	Sets Host-to-Master request
TH_set_dsp_amen_off	Disables PCI-bus mastering from the DSP environment (PCI-bus boards with PCI-bus mastering only)
TH_set_dsp_amen_on	Enable PCI-bus mastering from the DSP environment (PCI-bus boards with PCI-bus mastering only)
TH_set_mgo	Runs DSP
TH_set_mh_rq_ie_off	Disables interrupt request on Master-to-Host request.
TH_set_mh_rq_ie_on;	Enables interrupt request on Master-to-Host request.
TH_set_mreset	Resets DSP
TH_set_mreset_mode_host	Set DSP reset to be controlled by host PC (PCI-bus boards with PCI-bus mastering only)
TH_set_mreset_mode_sa	Set DSP reset to be controlled by on-board reset controller (PCI-bus boards with PCI-bus mastering only)
TH_set_xhpia	Set HPIA register
TH_sram_len	Gets SRAM area length
TH_sys_stat_frg_data	Gets SYS_STAT_FRG register bits combination
TH_write_byte	Writes byte from specified host interface area
TH_write_dword	Writes 32-bit word to specified host interface area
TH_write_word	Writes 16-bit word to specified host interface area
TH_xhpia_reset	Resets XHPIA

4.11 Functions Reference

This section contains a complete reference for *TSDK* API function library. Each function is supplied with detail description of performed operation, function parameters and example.

CAUTION

Each function of *TSDK* API function library returns the exit error code, which is the negative value. However, some functions also return the requested data, which is a positive value. The user application can recognize between the returned error code or requested data by means of analyzing the sign of returned value.

TH_area_info

int TH_area_info(UCHAR area,PULONG phaddr, PCHAR s)

Parameters

area

Area selector.

phaddr

Points to the area corresponding base address.

s

After return contains stringable remark where specified area is allocated (“<1Mb”, “IO space” and so on)

Return Value

Error code.

Description

Gets area physical base address and its stringable description about where it is allocated. The function is valid only for PCI *TORNADO* boards.

Parameter *area* can take the following values:

area	**description**
TH_PCI_PCIC_AREA	AMCC registers area
TH_PCI_HIF_AREA	Host Interface area
TH_PCI_ECC_AREA	On-board emulator interface area
TH_PCI_DPM_AREA	DPRAM area

Example

```
…
#include “th.h”
...
int err;
ULONG addr;  CHAR name[20];
...
err = TH_area_info(TH_PCI_HIF_AREA,&addr,name);
if(!err) printf("HIF/HPI area: 0x%08lx (%s)\n",addr & ~0x3L,name);
```

. . .

TH_board_close

int TH_board_close()

Return Value

Error code.

Description

Closes selected board. All the board access calls will fail after this one is called.

Example

```
...
#include "th.h"
…
err = TH_board_select(some_index);
...
err = TH_board_open();
...
TH_board_close();
...
```

See Also

TH_board_close, TH_board_id

TH_board_id

int TH_board_id()

Return Value

Returns either the hardware board id (for example TP67_DEV_ID) or error code.

Description

The routine gets harware id of the currently selected board.

Example

```
...
#include "th.h"
…
int err,i;
…
if(!TH_board_total()) return;
err = TH_board_select(0);
if(!err)
      printf("Board id 0x%04lx\n",TH_board_id());
...
```

See Also

TH_board_total, TH_board_name, TH_board_type

TH_board_info

int TH_board_info(PTH_BOARD_INFO_TD board_info, PULONG num_boards)

Parameters

board_info

Specifies a pointer to array of structures of TH_BOARD_INFO_TD type to return to the list of detected *TORNADO* boards. The array of TH_MAX_PLUGGED_BOARDS size must be alloceted by a user.

num_boards

Pointer in which the number of detected *TORNADO* board is returned.

Return Value

Error code.

Description

This routine gets information about *TORNADO* boards which are currently plugged into your computer collected by driver. It is possible for Isa boards to be configured by special utility so that they will be not visible to driver and tornado software kit in whole.

Example

```
...
#include "th.h"
…
int i,err;
ULONG num_brd;
TH_BOARD_INFO_TD  board_info[ TH_MAX_PLUGGED_BOARDS ];
…
err = TH_board_info(board_info, num_brd);
if(!err){
for(i = 0;i<num_brd;i++) printf("HW board id
0x%lx\n",board_info[i].hw_board_id);
}
      ...
```

See Also

TH_board_total, TH_board_name, TH_board_type

TH_board_lock

int TH_board_lock()

Return Value

Error code.

Description

Locks a selected board. All the board access calls from other threads will fail after this one is called untill **TH_board_unlock** is called.

Example

```
...
#include "th.h"
…
TH_board_select(1);
...
TH_board_lock();
TH_set_smp_segm(some_index); // setting some_index must be above
...
TH_smp_off();
TH_board_unlock();
...
```

See Also

TH_board_unlock

TH_board_name

int TH_board_name(TH_BOARD_TD board_descr,PCHAR board_name)

Parameters

board_descr

Specifies an index of a board to get its name.

board_name

Pointer in which *TORNADO* board name is returned. String must be allocated by user

Return Value

Error code.

Description

This routine gets stringable *TORNADO* board name. It isn't influenced by **TH_board_select** call.

Example

```
...
#include "th.h"
…
int err,i;
CHAR board_name[20];
…
for(i = 0;i<TH_board_total();i++){
        err = TH_board_name(0,board_name);
        if(!err) printf("board name %s\n", board_name);
}
```

See Also

TH_board_total, TH_board_id, TH_board_type

TH_board_open

int TH_board_open()

Return Value

Error code.

Description

Opens selected board. All board access calls will fail until this one is called

Example

```
...
#include "th.h"
…
err = TH_board_select(1);
if(err) return;
err = TH_board_open();
if(err) return;
...
```

See Also

TH_board_close, TH_board_id

TH_board_select

int TH_board_select(TH_BOARD_DESCR_TD board_id)

Parameters

board_id

Specifies an index of a board to select.

Return Value

Error code.

Description

This routine tells library what a board will be meant in followed acccess calls. Boards numeration begins from zero. Maximum index can be TH_MAX_PLUGGED_BOARDS – 1.

Example

```
...
#include "th.h"
…
ULONG num_brd;
…
num_brd = TH_board_total();
if(num_brd) {
    TH_board_select(0);
    TH_board_open();
}
```

See Also

TH_board_total, TH_board_info

TH_board_type

int TH_board_type(TH_BOARD_TD board_descr)

Parameters

board_descr

Specifies an index of a board to get its type.

Return Value

The type of board.

Description

This routine allows getting *TORNADO* board type without any additional calls.

Return value can take the following values:

area	**description**
TH_PCI_BOARD_TYPE	PCI board type
TH_ISA_BOARD_TYPE	ISA board type
TH_UNDEFINED_BOARD_TYPE	In case of an error

Example

```
…
#include "th.h"

TH_BOARD_TD board_descr;
...
for(board_descr = 0; board_descr <TH_board_total();board_descr ++)
    if(TH_PCI_BOARD_TYPE = TH_board_type(board_descr)){
       // do something with it
    }
...
```

See Also

TH_board_total, TH_board_name, TH_board_id

TH_board_total

int TH_board_total()

Return Value

The number of currently plugged boards or zero.

Description

This routine gets the number of *TORNADO* boards which are currently plugged into your computer. It is possible for Isa boards to be configured by special utility so that they will be not visible to driver and *TORNADO* Software Kit in whole.

Example

```
…
#include "th.h"

TH_BOARD_TD board_descr;
...
for(board_descr = 0; board_descr <TH_board_total();board_descr ++)
    if(TH_PCI_BOARD_TYPE = TH_board_type(board_descr)){
       // do something with it
    }
...
```

See Also

TH_board_info, TH_board_name, TH_board_type

TH_board_unlock

int TH_board_unlock()

Return Value

Error code.

Description

Unlocks selected locked board. Subsequent **TH_board_lock** call. These two functions **TH_board_lock** and **TH_board_unlock** should enclose piqued sections in multithreaded applications.

Example

```
...
#include "th.h"
…
TH_board_select(1);
...
TH_board_lock();
TH_set_smp_segm(1);
...
TH_smp_off();
TH_board_unlock();
...
```

See Also

TH_board_lock

TH_clr_dpram_err

int TH_clr_dpram_err ()

Return Value

Error code.

Description

Clears DPRAM error status. Actually the function zeroizes *TH_PCI_HIF_CLR_DPRAM_ERR* register. The function is usable only for PCI boards, it will fail for other ones.

Example

```
...
#include "th.h"
…
if(TH_dpram_err_data())
      TH_clr_dpram_err();
…
```

See Also

TH_dpram_err_data

TH_clr_dpram_err_ie

int TH_clr_dpram_err_ie ()

Return Value

Error code.

Description

Clears *DPRAM_ERR_IE* bit in *TH_PCI_HIF_IM_RG* register. The function is usable only for PCI boards, it will fail for other ones.

Example

```
...
#include "th.h"
…
if(TH_dpram_err_ie_data())
      TH_clr_dpram_err_ie();
…
```

See Also

TH_set_dpram_err_ie, TH_dpram_err_ie_data

TH_clr_hpi_err

int TH_clr_hpi_err ()

Return Value

Error code.

Description

The function clears *TH_PCI_HIF_CLR_HPI_ERR* register. It can't be used for *TORNADO-3x* DSP systems.

Example

```
...
#include "th.h"
…
if(TH_hpi_err_data())
      TH_clr_hpi_err();
…
```

See Also

TH_hpi_err_data

TH_clr_hpi_err_ie

int TH_clr_hpi_err_ie ()

Return Value

Error code.

Description

Clears *HPI_ERR_IE* bit in *TH_PCI_HIF_IM_RG* register. It can't be used for *TORNADO-3X* DSP systems.

Example

```
...
#include "th.h"
…
if(TH_hpi_err_ie_data())
      TH_clr_hpi_err_ie();
…
```

See Also

TH_set_hpi_err_ie, TH_hpi_err_ie_data

TH_clr_hpi_hint_ie

int TH_clr_hpi_hint_ie()

Return Value

Error code.

Description

Clears *HPI_HINT_IE* bit in *TH_PCI_HIF_IM_RG* register. It can't be used for *TORNADO-3x* DSP systems.

Example

```
...
#include "th.h"
…
if(TH_hpi_hint_ie_data())
      TH_clr_hpi_hint_ie();
…
```

See Also

TH_set_hpi_hint_ie, TH_hpi_hint_ie_data

TH_clr_hpic_hint

int TH_clr_hpic_hint ()

Return Value

Error code.

Description

Clears HINT bit in HPIC register. It isn't valid for boards of *TORNADO*-3X series.

Example

```
...
#include "th.h"
…
if(TH_hpic_hint_data())
      TH_clr_hpic_hint();
…
```

See Also

TH_hpic_hint_data

TH_clr_hpic_xhpia

int TH_clr_hpic_xhpia()

Return Value

Error code.

Description

Clears XHPIA bit of HPIC MSB and LSB registers. The function has the meaning only for *TORNADO*-54X boards series.

Example

```
...
#include "th.h"
…
TH_clr_hpic_xhpia();
…
```

See Also

TH_set_hpic_xhpia

TH_clr_mh_rq

int TH_clr_mh_rq ()

Return Value

Error code.

Description

Clears CLR_MH_RQ register.

Example

```
...
#include "th.h"
…
if(TH_mh_rq_data())
      TH_clr_mh_rq();
…
```

See Also

TH_mh_rq_data

TH_clr_sb_err

int TH_clr_sb_err()

Return Value

Error code.

Description

Clears CLR_SB_ERR_FRG register. It isn't valid for boards of *TORNADO*-P6X series.

Example

```
...
#include "th.h"
…
if(TH_sb_err_data())
     TH_clr_sb_err()
…
```

See Also

TH_sb_err_data

TH_connect_interrupt

int TH_connect_interrupt(ULONG mask,HANDLE * event)

Parameters

mask

Reserved for the future.

event

Kernel object. It will be in signalled state if hardware interrupt is catched. It should be used as a parameter in one of Win32 API wait functions.

Return Value

Error code.

Description

Allows user to be informed when interrupt occures in the subsequent call of WaitForSingleObject or WaitForMultipleObject functions.

Example

```
...
#include "th.h"
…
ULONG mask = 0;
HANDLE int_event;
…
TH_connect_interrupt(mask, &int_event);
…
if(WAIT_OBJECT_0 == WaitForSingleObject(int_event,100)){
      // do something here
}
…
```

See Also

TH_disconnect_interrupt, TH_interrupt_line

TH_ctrl_rg

UCHAR TH_ctrl_rg()

Return Value

The value of contol register if success, otherwise 0xff.

Description

Reads control register (HIF_CNTR_RG for PCI boards) .

Example

```
...
#include "th.h"

UCHAR old_ctrl_rg;…
…
old_ctrl_rg = TH_ctrl_rg();
…
TH_set_ctrl_rg(old_ctrl_rg);
…
```

See Also

TH_set_ctrl_rg

TH_dpram_data

ULONG TH_dpram_data(ULONG addr))

Parameters

addr

DPRAM address.

Return Value

Read value if success othewise 0xffffffff.

Description

Reads 32-bit from DPRAM area. The function is used for occasional access to DPRAM. Only for *TORNADO*-P6X boards series.

Example

```
…
#include “th.h”

ULONG addr,eaddr;value;
...
eaddr = TH_dpram_len() - sizeof(ULONG);
for(addr = 0;addr < eaddr;addr+= sizeof(ILONG)){
    value = TH_dpram_data(addr);
    TH_set_dpram(addr,++value);
}
...
```

See Also

TH_set_dpram

TH_dpram_err_data

BOOLEAN TH_dpram_err_data()

Return Value

TRUE value if DPRAM error status is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether DPRAM error occured or not. Only for *TORNADO*-P6X boards series.

Example

```
...
#include "th.h"
…
if(TH_dpram_err_data())
      TH_clr_dpram_err();
…
```

See Also

TH_clr_dpram_err

TH_dpram_err_ie_data

BOOLEAN TH_dpram_err_ie_data()

Return Value

TRUE value if DPRAM_ERR_IE bit in HIF_IM_RG register is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether DPRAM_ERR_IE bit in HIF_IM_RG register is set or not. Only for *TORNADO*-P6X boards series.

Example

```
...
#include "th.h"
…
if(TH_dpram_err_ie_data())
      TH_clr_dpram_err_ie();
…
```

See Also

TH_set_dpram_err_ie, TH_clr_dpram_err_ie

TH_dpram_ie_data

BOOLEAN TH_dpram_ie_data()

Return Value

TRUE value if DPRAM_IE bit in HIF_IM_RG register is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether DPRAM_IE bit in HIF_IM_RG register is set or not. Only for *TORNADO*-P6X boards series.

Example

```
...
#include "th.h"
…
if(!TH_dpram_ie_data())
      TH_set_dpram_ie_on();
…
```

See Also

TH_set_dpram_ie_on, TH_set_dpram_ie_off

TH_dpram_irq_data

BOOLEAN TH_dpram_irq_data()

Return Value

TRUE value if DPRAM_IRQ bit in HIF_IS_RG register is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether DPRAM_IRQ bit in HIF_IS_RG register is set or not. Only for *TORNADO*-P6X boards series.

Example

```
...
#include "th.h"
…
if(TH_dpram_irq_data()){
    ...
    // do something
    ...
}
…
```

TH_dpram_len

ULONG TH_dpram_len()

Return Value

DPRAM length if success. In case of an error 0xffffffff value is returned.

Description

Returns current board DPRAM legth. Only for *TORNADO*-P6X boards series.

Example

```
…
#include "th.h"

ULONG addr,eaddr;value;
...
eaddr = TH_dpram_len() - sizeof(ULONG);
for(addr = 0;addr < eaddr;addr+= sizeof(ILONG)){
    value = TH_dpram_data(addr);
    TH_set_dpram(addr,++value);
}
...
```

See Also

TH_dram_len, TH_sram_len

TH_dpsem_data

ULONG TH_dpsem_data(ULONG addr)

Parameters

addr

DPSEM address to read.

Return Value

DPSEM read value if success. In case of an error 0xffffffff value is returned.

Description

It is used for occasional access to DPSEM. Only for *TORNADO*-P6X boards series.

Example

```
…
#include "th.h"

ULONG sem;value[TP6X_DPM_DPSEM_LEN];
...
for(sem = 0;sem < TP6X_DPM_DPSEM_LEN;sem++){
    value[sem] = TH_dpsem_data(sem);
}
...
```

See Also

TH_set_dpsem

TH_dram_len

ULONG TH_dram_len()

Return Value

DRAM length if success. In case of an error 0xffffffff value is returned.

Description

Returns current board DRAM legth. Only for *TORNADO*-P6X boards series.

Example

```
…
#include "th.h"

ULONG dram_len;
...
dram_len = TH_dram_len();
...
```

See Also

TH_sram_len

TH_dsp_amen_data

BOOLEAN TH_dsp_amen_data()

Return Value

TRUE value if DSP_AMEN bit in HIF_CNTR_RG register is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether DSP_AMEN bit in HIF_CNTR_RG register is set or not. Only for *TORNADO*-P6X boards series.

Example

```
...
#include "th.h"
…
if(!TH_dsp_amen_data())
      TH_set_dsp_amen_on();
…
```

See Also

TH_set_dsp_amen_on, TH_set_dsp_amen_off

TH_dsp_hpi_disable_data

BOOLEAN TH_dsp_hpi_disable_data()

Return Value

TRUE value if DSP_HPI_DISABLE bit in DSP_STAT_FRG register is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether DSP_HPI_DISABLE bit in DSP_STAT_FRG register is set or not. Only for *TORNADO*-54X boards series.

Example

```
...
#include "th.h"
…
if(TH_dsp_hpi_disable_data()){
    // do something
}
…
```

TH_dsp_mlock_data

BOOLEAN TH_dsp_mlock_data()

Return Value

TRUE value if DSP_MLOCK bit in DSP_STAT_FRG register is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether DSP_MLOCK bit in DSP_STAT_FRG register is set or not. Only for *TORNADO*-54X/6X boards series.

Example

```
...
#include "th.h"
…
if(TH_dsp_mlock_data()){
    // do something
}
…
```

TH_dsp_mreset_data

BOOLEAN TH_dsp_mreset_data()

Return Value

TRUE value if DSP_MRESET bit is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether DSP_MRESET bit (in DSP_STAT_FRG register for ISA boards and HIF_CNTR_RG register for PCI boards) is set or not. Only for *TORNADO*-54X/6X/P6X boards series.

Example

```
...
#include "th.h"
…
if(TH_dsp_mreset_data()){
    // do something
}
…
```

TH_dsp_pd_data

BOOLEAN TH_dsp_pd_data()

Return Value

TRUE value if DSP_PD bit is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether DSP_MRESET bit (in DSP_STAT_FRG register for ISA boards and HIF_CNTR_RG register for PCI boards) is set or not. Only for *TORNADO*-6X/P6X boards series.

Example

```
...
#include "th.h"
…
if(TH_dsp_pd_data()){
    // do something
}
…
```

TH_dsp_stat_frg_data

UCHAR TH_dsp_stat_frg_data()

Return Value

Byte value. In case of an error 0xff value is returned.

Description

Returns corresponding combination of bits from DSP_STAT_FRG register that are ORed in result value (for T6X boards DSP_MLOCK, DSP_MRESET, DSP_PD, for T54X - DSP_MLOCK, DSP_HPI_DISABLE, DSP_MRESET bits are used). Only for *TORNADO*-54X/6X boards series.

Example

```
…
#include "th.h"

UCHAR dsp_stat;
...
dsp_stat = TH_dsp_stat_frg_data();
...
```

TH_ecc_clr_err

int TH_ecc_clr_err ()

Return Value

Error code.

Description

Clears ECC_AUX_CLR_ECC_ERR register. The function is valid only for *TORNADO*-P6X boards series.

Example

```
...
#include "th.h"
…
if(TH_ecc_err_data())
    TH_ecc_clr_err();
…
```

See Also

TH_ecc_err_data, TH_ecc_reset_clr_err

TH_ecc_err_data

BOOLEAN TH_ecc_err_data()

Return Value

TRUE value if ECC_ERR bit is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks if ECC_ERR bit is set in ECC_AUX_STAT_RG register. The function is valid only for *TORNADO*-P6X boards series.

Example

```
...
#include "th.h"
…
if(TH_ecc_err_data())
    TH_ecc_clr_err();
…
```

See Also

TH_ecc_clr_err

TH_ecc_irq_data

BOOLEAN TH_ecc_irq_data()

Return Value

TRUE value if ECC_IRQ bit is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks if ECC_IRQ bit is set in ECC_AUX_STAT_RG register. The function is valid only for *TORNADO*-P6X boards series.

Example

```
...
#include "th.h"
…
if(TH_ecc_err_data()){
    // do something;
}
…
```

TH_ecc_off

int TH_ecc_off()

Return Value

Error code.

Description

The same as **TH_set_emu_xdsp** function. Only for ISA boards.

Example

```
…
#include "th.h"
...
TH_ecc_off();
...
```

See Also

TH_set_emu_xdsp

TH_ecc_reset

int TH_ecc_reset()

Return Value

Error code.

Description

Clears ECC_AUX_ECC_RESET register. The function is valid only for *TORNADO*-P6X boards series.

Example

```
…
#include “th.h”
...
TH_ecc_reset();
...
```

TH_ecc_stat_rg

UCHAR TH_ecc_stat_rg()

Return Value

Byte value. In case of an error 0xff value is returned.

Description

Returns corresponding combination of bits from ECC_AUX_STAT_RG register that are ORed in result value (ECC_ERR and ECC_IRQ bits are used). Only for *TORNADO*-P6X boards series.

Example

```
…
#include "th.h"

UCHAR ecc_stat;
...
ecc_stat = TH_ecc_stat_rg();
...
```

TH_emu_default

int TH_emu_default()

Return Value

Error code.

Description

Set EMU IO base address to 0x240 value. Only for ISA boards

Example

```
…
#include “th.h”
...
TH_emu_default();
...
```

See Also

TH_emu_io_baddr()

TH_emu_io_baddr

ULONG TH_emu_io_baddr()

Return Value

Currently set EMU IO address if success. In case of an error 0xffffffff value is returned.

Description

Returns currently set EMU IO address. Only for ISA boards.

Example

```
…
#include "th.h"
...
if(0x240 == TH_emu_io_baddr()){
    // do something, or do nothing
}
```

See Also

TH_emu_default

TH_emu_path_data

BOOLEAN TH_emu_path_data()

Return Value

TRUE value if XPATH selected, if MPATH - FALSE. In case of an error 0xff value is returned.

Description

Checks if XPATH or MPATH is selected. The function is valid only for ISA boards.

Example

```
…
#include "th.h"
...
if(TH_emu_path_baddr()){
    // do something, or do nothing
}
```

TH_emu_sel_rg

UCHAR TH_emu_sel_rg()

Return Value

The value of ECC_AUX_EMU_SEL_RG if success, otherwise 0xff.

Description

Gets the contents of ECC_AUX_EMU_SEL_RG register. Only for *TORNADO*-P6X boards series.

Example

```
…
#include "th.h"

UCHAR emu_sel_rg;
...
emu_sel_rg = TH_emu_sel_rg();
...
```

See Also

TH_set_emu_sel_rg

TH_err_message

char *** ***TH_err_message(int err_code)

Parameters

err_code

An error code which is an output from one of the kit function for which stringable name is necessary.

Return Value

Error message. It is allocated inside library.

Description

This function shows an error message corresponding to the specified error code.

Example

```
...
#include "th.h"
…
err = TH_board_info(board_info, num_brd);
if(err) printf(TH_err_message(err));
...
```

See Also

TH_flag_err, TH_last_err

TH_fdata_rg

UCHAR TH_fdata_rg()

Return Value

The value of FDATA register if success, otherwise 0xff.

Description

Reads FDATA register. Valid only ISA boards.

Example

```
…
#include "th.h"

UCHAR fdata_rg;
...
fdata_rg = TH_fdata_rg();
...
```

See Also

TH_set_fdata_rg

TH_flag_err

int TH_flag_err()

Return Value

Flag error code.

Description

Returns the fist error code of a call sequence. Immediatly next call of this routine clears flag error.

Example

```
...
#include "th.h"
…
TH_board_info(board_info, num_brd);
...
TH_board_select();
TH_board_open();
if(err = TH_flag_err()) printf(TH_err_message(err));
err = TH_flag_err(); // will return 0
...
```

See Also

TH_last_err, TH_err_message

TH_frg_data

UCHAR TH_frg_data(UCHAR frg)

Parameters

frg

FRG register to read.

Return Value

The value of some FRG register if success, otherwise 0xff.

Description

Reads the FRG register specified. Valid only ISA boards.

Example

```
…
#include "th.h"

UCHAR dev_id;
...
dev_id = TH_frg_data(T3X_DEV_ID1_FRG);
...
```

See Also

TH_set_frg_verif

TH_fsel_rg

UCHAR TH_fsel_rg()

Return Value

The value of FSEL register if success, otherwise 0xff.

Description

Reads FSEL register. Valid only ISA boards.

Example

```
…
#include "th.h"

UCHAR fsel_rg;
...
fsel_rg = TH_fsel_rg();
...
```

See Also

TH_set_fsel_rg

TH_hm_rq0_rg, TH_hm_rq1_rg

UCHAR TH_hm_rq0_rg(), UCHAR TH_hm_rq1_rg()

Return Value

The value of HIF_HM_RQ0/ HIF_HM_RQ1 register if success, otherwise 0xff.

Description

Reads HIF_HM_RQ0/ HIF_HM_RQ1 register. Valid only PCI boards.

Example

```
…
#include "th.h"

UCHAR hm0_rq;
...
hm0_rq = TH_hm_rq0_rg();
...
```

See Also

TH_set_hm_rq0_rg, TH_set_hm_rq1_rg

TH_hpi_disable_data

BOOLEAN TH_hpi_disable_data()

Return Value

TRUE value if HPI_DISABLE bit in HPI_IE_FRG register is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether HPI_DISABLE bit in HPI_IE_FRG register is set or not. Only for *TORNADO*-5402 boards series.

Example

```
...
#include "th.h"
…
if(TH_hpi_disable_data())
    TH_set_hpi_enable();
…
```

See Also

TH_set_hpi_disable, TH_set_hpi_enable

TH_hpi_err_data

BOOLEAN TH_hpi_err_data()

Return Value

TRUE value if HPI_ERR bit is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether HPI_ERR bit in SET_HM_RQ_FRG register for ISA boards and HIF_IS_RG for PCI boards is set or not. The function isn't valid for *TORNADO*-3X boards series.

Example

```
...
#include "th.h"
…
if(TH_hpi_err_data())
    TH_clr_hpi_err();
…
```

See Also

TH_clr_hpi_err

TH_hpi_err_ie_data

BOOLEAN TH_hpi_err_ie_data()

Return Value

TRUE value if HPI_ERR_IE bit is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether HPI_ERR_IE bit in HPI_IE_FRG register for ISA boards and HIF_IM_RG for PCI boards is set or not. The function isn't valid for *TORNADO*-3X boards series.

Example

```
...
#include "th.h"
…
if(TH_hpi_err_ie_data())
    TH_clr_hpi_err_ie();
…
```

See Also

TH_set_hpi_err_ie, TH_clr_hpi_err_ie

TH_hpi_hint_data

BOOLEAN TH_hpi_hint_data()

Return Value

TRUE value if HPI_HINT bit is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether HPI_HINT bit in SET_HM_RQ_FRG register for ISA boards and HIF_IS_RG for PCI boards is set or not. The function isn't valid for *TORNADO*-3X boards series.

Example

```
...
#include "th.h"
…
if(TH_hpi_hint_data()){
    // do something
}
…
```

TH_hpi_hint_ie_data

BOOLEAN TH_hpi_hint_ie_data()

Return Value

TRUE value if HPI_HINT_IE bit is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether HPI_HINT_IE bit in HPI_IE_FRG register for ISA boards and HIF_IM_RG for PCI boards is set or not. The function isn't valid for *TORNADO*-3X boards series.

Example

```
...
#include "th.h"
…
if(TH_hpi_hint_ie_data())
    TH_clr_hpi_hint_ie();
…
```

See Also

TH_set_hpi_hint_ie, TH_clr_hpi_hint_ie

TH_hpi_ie_frg_data

UCHAR TH_hpi_ie_frg_data()

Return Value

Byte value. In case of an error 0xff value is returned.

Description

Returns corresponding combination of bits from DSP_STAT_FRG register that are ORed in result value (for T6X boards HPI_HINT_IE, HPI_ERR_IE, for T54X - HPI_ERR_IE, HPI_HINT_IE, HPI_DISABLE bits are used). Only for *TORNADO*-54X/6X boards series.

Example

```
…
#include "th.h"

UCHAR ie_frg;
...
ie_frg = TH_hpi_ie_frg_data();
...
```

TH_hpi_init

int TH_hpi_init()

Return Value

Error code.

Description

Initializes HPIC and HPIA registers to begin to work with HPI properly. For T54X boards performs reset of HPIA/XHPIA. The function isn't valid for *TORNADO*-3X boards series.

Example

```
…
#include "th.h"
...
TH_hpi_init();
...
```

See Also

TH_hpia_reset, TH_xhpia_reset

TH_hpia_reset

int TH_hpia_reset()

Return Value

Error code.

Description

Zeroizes HPIA_RG registers. Only for *TORNADO*-54X/6X boards series.

Example

```
…
#include "th.h"
...
TH_hpia_reset();
...
```

See Also

TH_hpia_rg, TH_set_hpia_rg

TH_hpia_rg

ULONG TH_hpia_rg()

Return Value

The value of HPIA_RG if success, otherwise 0xffffffff.

Description

Gets the contents of HPIA_RG register. The function isn't valid for *TORNADO*-3X boards series.

Example

```
…
#include "th.h"
...
```

See Also

TH_set_hpia_rg

TH_hpic_dspint_data

BOOLEAN TH_hpic_dspint_data()

Return Value

TRUE value if DSP_INT bit in HPIC_RG register is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether DSP_INT bit in HPIC_RG register is set or not. Only for *TORNADO*-6X/P6X boards series.

Example

```
...
#include "th.h"
…
if(!TH_hpic_dspint_data())
    TH_set_hpic_dspint();
…
```

See Also

TH_set_hpic_dspint

TH_hpic_fetch_data

BOOLEAN TH_hpic_fetch_data()

Return Value

TRUE value if HPIC_FETCH bit in HPIC_RG register is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether HPIC_FETCH bit in HPIC_RG register is set or not. Only for *TORNADO*-6X/P6X boards series.

Example

```
...
#include "th.h"
…
if(!TH_hpic_fetch_data())
    TH_set_hpic_fetch();
…
```

See Also

TH_set_hpic_fetch

TH_hpic_hint_data

BOOLEAN TH_hpic_hint_data()

Return Value

TRUE value if HPIC_HINT bit in HPIC_RG register is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether HPIC_HINT bit in HPIC_RG register is set or not. Only for *TORNADO*-54X/6X/P6X boards series.

Example

```
...
#include "th.h"
…
if(TH_hpic_hint_data())
    TH_clr_hpic_hint();
…
```

See Also

TH_clr_hpic_hint

TH_hpic_hrdy_data

BOOLEAN TH_hpic_hrdy_data()

Return Value

TRUE value if HPIC_HRDY bit in HPIC_RG register is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether HPIC_HRDY bit in HPIC_RG register is set or not. Only for *TORNADO*-6X/P6X boards series.

Example

```
...
#include "th.h"
…
if(TH_hpic_hrdy_data()){
    // do something
}
…
```

TH_hpic_hwob_data

BOOLEAN TH_hpic_hwob_data()

Return Value

TRUE value if HPIC_HWOB bit in HPIC_RG register is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether HPIC_HWOB bit in HPIC_RG register is set or not. Only for *TORNADO*-6X/P6X boards series.

Example

```
...
#include "th.h"
…
if(!TH_hpic_hwob_data())
    TH_set_hpic_hwob();
…
```

See Also

TH_set_hpic_hwob

TH_hpic_rg

ULONG TH_hpic_rg()

Return Value

The value of HPIC_RG if success, otherwise 0xffffffff.

Description

Gets the contents of HPIC_RG register. The function isn't valid for *TORNADO*-3X boards series.

Example

```
...
#include "th.h"

ULONG rg;
…
rg = TH_hpic_rg();
…
```

See Also

TH_set_hpic_rg

TH_hpid_ainc_rg

ULONG TH_hpid_ainc_rg()

Return Value

The value of HPID_AINC_RG if success, otherwise 0xffffffff.

Description

Gets the contents of HPID_AINC_RG register. The function isn't valid for *TORNADO*-3X boards series.

Example

```
...
#include "th.h"

ULONG rg;
…
rg = TH_hpid_ainc_rg();
…
```

See Also

TH_set_hpid_ainc_rg

TH_hpid_rg

ULONG TH_hpid_rg()

Return Value

The value of HPID_RG if success, otherwise 0xffffffff.

Description

Gets the contents of HPIA_RG register. The function isn't valid for *TORNADO*-3X boards series.

Example

```
...
#include "th.h"

ULONG rg;
…
rg = TH_hpid_rg();
…
```

See Also

TH_set_hpid_rg

TH_im_rg

UCHAR TH_im_rg()

Return Value

The value of HIF_IM_RG if success, otherwise 0xff.

Description

Gets the contents of HIF_IM_RG register. The function is valid only for *TORNADO*-P6X boards series.

Example

```
...
#include "th.h"

UCHAR rg;
…
rg = TH_im_rg();
…
```

See Also

TH_set_im_rg

TH_init

int TH_init()

Return Value

Error code.

Description

Clears control and smp registers. Only for ISA boards.

Example

```
…
#include “th.h”
...
TH_init();
...
```

TH_interrupt_line

UCHAR TH_interrupt_line()

Return Value

Used interrupt line of the currently selected board if success, otherwise 0xff.

Description

If no error occured while calling the function and no interrupt line is set to the device specified zero value is returned. In case of PCI boards the function returns non zero value always.

Example

```
…
#include "th.h"

UCHAR int_line;
...
int_line = TH_interrupt_line();
if(int_line){
    // the board has interrupt line
}
```

See Also

TH_interrupt_regs_clear, TH_interrupt_regs

TH_interrupt_regs

BOOLEAN TH_interrupt_regs(PTH_INTREGS_TD regs)

Parameters

regs

A pointer to the structure to return to the set of registers.

Return Value

TRUE –if there was an interrupt occured and the set of registers successfully returned, FALSE – otherwise.

Description

The function returns interrupt registers set corresponding to the certain *TORNADO* board which are buffered by driver interrupt service routine any time the buffer isn't empty. The buffer is organized as FIFO one.

When interrupts are assigned the source (or sources) of it is detected and ORed in respective bit of *int_mask* field of *regs* structure.

Effective use of this function expects a programmer to use WaitForSingleObject or WaitForMultipleObject functions with it. Note that interrupt event object returned by **TH_connect_interrupt** is in active state every time the buffer contains at least one structure.

The interrupt sources for PCI boards can take the following values:

source	**description**
TH_TARGET_ABORT_INT	Target abort
TH_MASTER_ABORT_INT	Master abort interrupt
TH_RD_COMPLETE_INT	DSP-to-PCI master reading complete
TH_WR_COMPLETE_INT	DSP-to-PCI master writing complete
TH_OMBX_EMPTY_INT	Output mailbox goes empty
TH_IMBX_FULL_INT	Input mailbox becomes full

The interrupt sources for *TORNADO*-6X boards can take the following values:

source	**description**

The interrupt sources for *TORNADO*-3X boards can take the following values:

source	**description**

The interrupt sources for *TORNADO*-54X boards can take the following values:

source	**description**

Example

```
…
#include <wtypes.h>
#include <winbase.h>
#include "th.h"

int               err;
TH_INTREGS_TD     regs;
HANDLE            int_handle;
UCHAR       mb_num;
…
err = TH_connect_interrupt(&int_handle);
if(err) goto exit_err;
…
// flush interrupts buffer
TH_interrupt_regs_clear();
// enable incoming MB interrupts for mb0
TH_set_pcic_intcsr_rg(0x1000);
…
// IT IS SUPPOSED TO BE A TORNADO-P6X BOARD
TH_set_pcic_ombx_rg(0,0x12345678);
if(err = TH_flag_err()) goto exit_err;
switch(WaitForSingleObject(int_handle,100)){ // 0.1 sec will be
enough
// see Win32 API reference for WaitForSingleObject  function
case WAIT_TIMEOUT: // timed out
      goto exit_err;
case WAIT_OBJECT_0:// an interrupt occured -> process it
      if(TH_interrupt_regs(&regs)){
            // check interrupt source
            if(regs.int_mask & TH_IMBX_FULL_INT){
                  // check mailbox number, must be 0
```

```
                        mb_num=(regs.u.pci.pcic_intcsr>>10)&0x3;
                        if(mb_num != 0)
                              goto exit_err;
                        if(regs.u.pci.pcic_imbx != ~0x12345678){
                        // unexpected data returned
                              goto exit_err;
                        }
                  }else{
                        // unexpected source
                        goto exit_err;
                  }
            }
            else{
                  // The buffer is empty
                  goto exit_err;
            }
            break;
      }
      …
      exit_err: // do something with it
      …
```

See Also

TH_connect_interrupt, TH_interrupt_regs_clear

TH_interrupt_regs_clear

int TH_interrupt_regs_clear()

Return Value

Error code.

Description

Clears interrupt registers buffer.

Example

```
…
#include "th.h"

int                err;
TH_INTREGS_TD      regs;
HANDLE             int_handle;
UCHAR       mb_num;
…
err = TH_connect_interrupt(&int_handle);
if(err) goto exit_err;
…
// flush interrupts buffer
TH_interrupt_regs_clear();
// enable incoming MB interrupts for mb0
TH_set_pcic_intcsr_rg(0x1000);
…
TH_set_pcic_ombx_rg(0,0x12345678);
if(err = TH_flag_err()) goto exit_err;
switch(WaitForSingleObject(int_handle,100)){ // 0.1 sec will be
enough
case WAIT_TIMEOUT: // timed out
      goto exit_err;
case WAIT_OBJECT_0:// an interrupt occured -> process it
      if(TH_interrupt_regs(&regs)){
            // check interrupt source
            if(regs.int_mask & TH_IMBX_FULL_INT){
                  // check mailbox number, must be 0
                  mb_num=(regs.u.pci.pcic_intcsr>>10)&0x3;
                  if(mb_num != 0)
                        goto exit_err;
                  if(regs.u.pci.pcic_imbx != ~0x12345678){
                  // unexpected data returned
                        goto exit_err;
                  }
            }else{
                  // unexpected source
```

```
                        goto exit_err;
                }
        }
        else{
                // The buffer is empty
                goto exit_err;
        }
        break;
}
…
exit_err: // do something with it
…
```

See Also

TH_interrupt_regs

TH_is_rg

UCHAR TH_is_rg()

Return Value

The value of HIF_IS_RG if success, otherwise 0xff.

Description

Gets the contents of HIF_IS_RG register. The function is valid only for *TORNADO*-P6X boards series.

Example

```
...
#include "th.h"

UCHAR rg;
…
rg = TH_is_rg();
…
```

See Also

TH_set_is_rg

TH_last_err

int TH_last_err()

Return Value

Last error code.

Description

Returns an error code of last function call.

Example

```
...
#include "th.h"
…
TH_board_info(board_info, num_brd);
if(err = TH_last_err()) printf(TH_err_message(err));
...
```

See Also

TH_flag_err, TH_err_message

TH_mapmem_view

int TH_mapmem_view(PULONG vtaddr,PULONG phaddr,PULONG size)

Parameters

vtaddr

Virtual address to use in Win32 aplication.

phaddr

Phisical address to pass to dsp resident program.

size

Maximal size of the allocated area.

Return Value

Error code.

Description

The function is used to get phisical and virtual addresses pointing to the same host memory space. *Phaddr* is supposed to be used from dsp resident program to initialize properly and start pci master transfers. *Vtaddr* points to the same phisical memory but is mapped to virtual address space of Win32 application communicating with dsp running program.

Example

```
...
#include "th.h"
…
ULONG buf_ptr, BUF, size;
int err;
struct MCD_DD {
ULONG   saddr,
              len;
}*MCD;
...
TH_set_smp_ptr(0x200,TP6X_SMP_DPRAM_AREA,(void**)&MCD,0);
TH_mapmem_view(&BUF,&buf_ptr,&size);
if(err = TH_flag_err()) some_processing_error_function(err);
// pass the address to dsp program
MCD->saddr = buf_ptr;
MCD->len     = size;
...
```

See Also

TH_unmapmem_view

TH_mh_rq_data

BOOLEAN TH_mh_rq_data()

Return Value

TRUE value if MH_RQ bit is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether MH_RQ bit in SET_HM_RQ_FRG register for ISA boards and HIF_IS_RG register for PCI boards is set or not.

Example

```
...
#include "th.h"
…
if(TH_mh_rq_data())
    TH_clr_mh_rq();
…
```

See Also

TH_clr_mh_rq

TH_mh_rq_ie_data

BOOLEAN TH_mh_rq_ie_data()

Return Value

TRUE value if MH_RQ_IE bit is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether MH_RQ_IE bit in CNTR_RG register is set or not.

Example

```
...
#include "th.h"
…
if(!TH_mh_rq_ie_data())
    TH_set_mh_rq_ie_on();
…
```

See Also

TH_set_mh_rq_ie_on, TH_set_mh_rq_ie_off

TH_mreset_data

BOOLEAN TH_mreset_data()

Return Value

TRUE value if MRESET bit is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether DSP program is in reset state or not.

Example

```
...
#include "th.h"
…
if(TH_mreset_data())
    TH_set_mgo();
…
```

See Also

TH_set_mreset, TH_set_mgo

TH_mreset_mode_data

BOOLEAN TH_mreset_mode_data()

Return Value

TRUE value if MRESET_MODE bit is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether MRESET_MODE bit in CNTR_RG register is set or not. Only for PCI boards.

Example

```
...
#include "th.h"
…
if(TH_mreset_mode_data())
    TH_set_mreset_mode_host();
…
```

See Also

TH_set_mreset_mode_host, TH_set_mreset_mode_sa

TH_nvram_read

UCHAR TH_nvram_read (ULONG addr)

Parameters

addr

NVRAM address to read.

Return Value

NVRAM value if success otherwise 0xff.

Description

The function is puposed for random access to NVRAM. Valid only for PCI boards.

Example

```
...
#include "th.h"

UCHAR sram_len;
…
TH_nvram_read(TP6X_NVRAM_SRAM_LEN_ID_ADDR, &sram_len);
…
```

See Also

TH_nvram_write

TH_nvram_write

int TH_nvram_write(ULONG addr,UCHAR value)

Parameters

addr

NVRAM address to write to.

value

The value to write.

Return Value

Error code.

Description

The function is puposed for random access to NVRAM. Valid only for PCI boards.

Example

```
...
#include "th.h"

…
TH_nvram_read(TP6X_NVRAM_SRAM_LEN_ID_ADDR,0);
…
```

See Also

TH_nvram_read

TH_pcic_intcsr_rg

ULONG TH_pcic_intcsr_rg()

Return Value

The value of INTCSR if success, otherwise 0xffffffff.

Description

Gets the contents of AMCC PCI contoller INTCSR register. The function is valid only for *TORNADO*-P6X boards series.

Example

```
...
#include "th.h"

ULONG rg;
…
rg = TH_pcic_intcsr_rg();
…
```

See Also

TH_set_pcic_intcsr_rg

TH_pcic_fifo_rg

ULONG TH_pcic_fifo _rg()

Return Value

The value of FIFO if success, otherwise 0xffffffff.

Description

Gets the contents of AMCC PCI contoller FIFO register. The function is valid only for *TORNADO*-P6X boards series.

Example

```
...
#include "th.h"

ULONG rg;
…
rg = TH_pcic_fifo_rg();
…
```

See Also

TH_set_pcic_fifo_rg

TH_pcic_imbx_rg

ULONG TH_pcic_imbx_rg(ULONG n)

Parameters

n

Mailbox number.

Return Value

The value of the selected input mailbox if success, otherwise 0xffffffff.

Description

Gets the contents of the selected AMCC PCI contoller input mailbox register. The function is valid only for *TORNADO*-P6X boards series.

The *n* parameter can be from 0 to 3.

Example

```
...
#include "th.h"

ULONG rg;
…
rg = TH_pcic_imbx_rg(0);
…
```

See Also

TH_set_pcic_imbx_rg

TH_pcic_mbef_rg

ULONG TH_pcic_mbef_rg()

Return Value

The value of MBEF if success, otherwise 0xffffffff.

Description

Gets the contents of AMCC PCI contoller MBEF register. The function is valid only for *TORNADO*-P6X boards series.

Example

```
...
#include "th.h"

ULONG rg;
…
rg = TH_pcic_mbef_rg();
…
```

See Also

TH_set_pcic_mbef_rg

TH_pcic_mcsr_rg

ULONG TH_pcic_mcsr_rg()

Return Value

The value of MCSR if success, otherwise 0xffffffff.

Description

Gets the contents of AMCC PCI contoller MCSR register. The function is valid only for *TORNADO*-P6X boards series.

Example

```
...
#include "th.h"

ULONG rg;
…
rg = TH_pcic_mcsr_rg();
…
```

See Also

TH_set_pcic_mcsr_rg

TH_pcic_mrar_rg

ULONG TH_pcic_mrar_rg()

Return Value

The value of MRAR if success, otherwise 0xffffffff.

Description

Gets the contents of AMCC PCI contoller MRAR register. The function is valid only for *TORNADO*-P6X boards series.

Example

```
...
#include "th.h"

ULONG rg;
…
rg = TH_pcic_mrar_rg();
…
```

See Also

TH_set_pcic_mrar_rg

TH_pcic_mrtc_rg

ULONG TH_pcic_mrtc_rg()

Return Value

The value of MRTC if success, otherwise 0xffffffff.

Description

Gets the contents of AMCC PCI contoller MRTC register. The function is valid only for *TORNADO*-P6X boards series.

Example

```
...
#include "th.h"

ULONG rg;
…
rg = TH_pcic_mrtc_rg();
…
```

See Also

TH_set_pcic_mrtc_rg

TH_pcic_mwar_rg

ULONG TH_pcic_mwar_rg()

Return Value

The value of MWAR if success, otherwise 0xffffffff.

Description

Gets the contents of AMCC PCI contoller MWAR register. The function is valid only for *TORNADO*-P6X boards series.

Example

```
...
#include "th.h"

ULONG rg;
…
rg = TH_pcic_mwar_rg();
…
```

See Also

TH_set_pcic_mwar_rg

TH_pcic_mwtc_rg

ULONG TH_pcic_mwtc_rg()

Return Value

The value of MWTC if success, otherwise 0xffffffff.

Description

Gets the contents of AMCC PCI contoller MWTC register. The function is valid only for *TORNADO*-P6X boards series.

Example

```
...
#include "th.h"

ULONG rg;
…
rg = TH_pcic_mwtc_rg();
…
```

See Also

TH_set_pcic_mwtc_rg

TH_pcic_ombx_rg

ULONG TH_pcic_ombx_rg(ULONG n)

Parameters

n

Mailbox number.

Return Value

The value of output mailbox if success, otherwise 0xffffffff.

Description

Gets the contents of AMCC PCI contoller output mailbox register. The function is valid only for *TORNADO*-P6X boards series.

Mailbox number can be from 0 to 3.

Example

```
...
#include "th.h"

ULONG rg;
…
rg = TH_pcic_ombx_rg();
…
```

See Also

TH_set_pcic_ombx_rg

TH_read_byte

int TH_read_byte(UCHAR area,ULONG addr,PUCHAR value)

Parameters

area

Area selector.

addr

Address offset within selected area.

value

Pointer to return result of reading.

Return Value

Error code.

Description

Reads byte from an address within selected area. The function is purposed for random access to any valid area in the context of the selected board.

Parameter *area* can take the following values:

Area	**description**
TH_PCI_PCIC_AREA	AMCC registers area
TH_PCI_HIF_AREA	Host Interface area
TH_PCI_ECC_AREA	On-board emulator interface area
TH_PCI_DPM_AREA	DPRAM area
TH_ISA_IO_AREA	ISA operation registers
TH_ISA_EMU_IO_AREA	On-board emulator interface area
TH_ISA_SMP_AREA	SMP area

Example

```
...
#include "th.h"
```

```
…
UCHAR value;
TH_read_byte(TH_ISA_IO_AREA, TH_ISA_SMP_RG, &value);
…
```

See Also

TH_write_byte, TH_read_word, TH_write_word, TH_read_dword, TH_write_dword

TH_read_dword

int TH_read_dword(UCHAR area,ULONG addr,PULONG value)

Parameters

area

Area selector.

addr

Address offset within selected area.

value

Pointer to return result of reading.

Return Value

Error code.

Description

Reads 32-bit word from an address within selected area. It doesn't matter for function whether selected area supports 32-bit acces or not. The function is purposed for random access to any valid area in the context of the selected board.

Parameter *area* can take the following values:

area	**description**
TH_PCI_PCIC_AREA	AMCC registers area
TH_PCI_HIF_AREA	Host Interface area
TH_PCI_ECC_AREA	On-board emulator interface area
TH_PCI_DPM_AREA	DPRAM area
TH_ISA_IO_AREA	ISA operation registers
TH_ISA_EMU_IO_AREA	On-board emulator interface area
TH_ISA_SMP_AREA	SMP area

Example

```
...
#include "th.h"
```

```
…
ULONG value;
TH_read_dword(TH_PCI_PCIC_AREA, TH_AMCC_FIFO_RG, &value);
…
```

See Also

TH_read_byte, TH_write_byte, TH_read_word, TH_write_word, TH_write_dword

TH_read_word

int TH_word_byte(UCHAR area,ULONG addr,PUSHORT value)

Parameters

area

Area selector.

addr

Address offset within selected area.

value

Pointer to return result of reading.

Return Value

Error code.

Description

Reads 16-bit word from an address within selected area. The function is purposed for random access to any valid area in the context of the selected board.

Parameter *area* can take the following values:

area	**description**
TH_PCI_PCIC_AREA	AMCC registers area
TH_PCI_HIF_AREA	Host Interface area
TH_PCI_ECC_AREA	On-board emulator interface area
TH_PCI_DPM_AREA	DPRAM area
TH_ISA_IO_AREA	ISA operation registers
TH_ISA_EMU_IO_AREA	On-board emulator interface area
TH_ISA_SMP_AREA	SMP area

Example

```
...
#include "th.h"
…
```

```
USHORT value;
TH_read_word(TH_ISA_IO_AREA, TH_ISA_SMP_RG, &value);
…
```

See Also

TH_write_byte, TH_read_byte, TH_write_word, TH_read_dword, TH_write_dword

TH_sb_ack_data

BOOLEAN TH_sb_ack_data()

Return Value

TRUE value if *SB_ACK* bit in *SYSTEM_STATUS_FRG* flag register is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether SB_ACK bit in *SYSTEM_STATUS_FRG* flag register is set or not. Only for ISA boards series.

Example

```
…
#include "th.h"
...
if(TH_sb_ack_data()){
    // do something
}
```

TH_sb_ccl_data

UCHAR TH_sb_ccl_data()

Return Value

Byte value. In case of an error 0xff value is returned.

Description

Gets CCL mode of the currently selected board (BYTE, WORD or DWORD). Corresponding bits of CNTR_RG are ORed in the result value. Only for *TORNADO*-3X/6X boards series.

Example

```
…
#include "th.h"

UCHAR ccl;
...
ccl = TH_sb_ccl_data();
...
```

See Also

TH_set_sb_ccl_byte, TH_set_sb_ccl_word, TH_set_sb_ccl_dword

TH_sb_err_data

BOOLEAN TH_sb_err_data()

Return Value

TRUE value if SB_ERR bit in SET_HM_RQ_FRG register is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether SB_ERR bit in SET_HM_RQ_FRG register is set or not. Only for ISA boards series.

Example

```
...
#include "th.h"
…
if(TH_sb_err_data())
    TH_clr_sb_err();
…
```

See Also

TH_clr_sb_err

TH_sb_err_ie_data

BOOLEAN TH_sb_err_ie_data()

Return Value

TRUE value if SB_ERR_IE bit in CNTR_RG register is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether SB_ERR_IE bit in CNTR_RG register is set or not. Only for ISA boards series.

Example

```
...
#include "th.h"
…
if(TH_sb_err_ie_data())
    TH_set_sb_err_off();
…
```

See Also

TH_set_sb_err_ie_on, TH_set_sb_err_ie_off

TH_sb_glock_data

BOOLEAN TH_sb_glock_data()

Return Value

TRUE value if SB_GLOCK bit in CNTR_RG register is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether SB_GLOCK bit in CNTR_RG register is set or not. Only for ISA boards series.

Example

```
...
#include "th.h"
…
if(TH_sb_glock_data())
    TH_set_sb_glock_off();
…
```

See Also

TH_set_sb_glock_on, TH_set_sb_glock_off

TH_sb_lock_data

BOOLEAN TH_sb_lock_data()

Return Value

TRUE value if SB_LOCK bit in CNTR_RG register is set, FALSE otherwise. In case of an error 0xff value is returned.

Description

Checks whether SB_LOCK bit in CNTR_RG register is set or not. Only for ISA boards series.

Example

```
...
#include "th.h"
…
if(TH_sb_lock_data())
    TH_set_sb_lock_off();
…
```

See Also

TH_set_sb_lock_on, TH_set_sb_lock_off

TH_set_ctrl_rg

int TH_set_ctrl_rg(UCHAR value)

Parameters

value

The value to set.

Return Value

Error code.

Description

Sets CNTR_RG register to the value.

Example

```
…
#include “th.h”
...
TH_set_ctrl_rg(0);
...
```

See Also

TH_ctrl_rg

TH_set_dpram

int TH_set_dpram(ULONG addr,ULONG value)

Parameters

addr

DPRAM address to write to.

value

The value to write.

Return Value

Error code.

Description

Writes 32-bit word to DPRAM. The function is used to random access to DPRAM. The function is valid only for PCI boards.

Example

```
…
#include "th.h"

ULONG addr,eaddr;value;
...
eaddr = TH_dpram_len() - sizeof(ULONG);
for(addr = 0;addr < eaddr;addr+= sizeof(ILONG)){
    value = TH_dpram_data(addr);
    TH_set_dpram(addr,++value);
}
...
```

See Also

TH_dpram_data

TH_set_dpram_err_ie

int TH_set_dpram_err_ie()

Return Value

Error code.

Description

Clears DPRAM_ERR_IE bit in HIF_IM_RG register. The function is valid only for PCI boards.

Example

```
...
#include "th.h"
…
if(!TH_dpram_err_ie_data())
    TH_set_dpram_err_ie();
…
```

See Also

TH_dpram_err_ie_data, TH_clr_dpram_err_ie

TH_set_dpram_ie_off

int TH_set_dpram_ie_off()

Return Value

Error code.

Description

Clears DPRAM_IE bit in HIF_IM_RG register. The function is valid only for PCI boards.

Example

```
...
#include "th.h"
…
if(TH_dpram_ie_data())
      TH_set_dpram_ie_off();
…
```

See Also

TH_dpram_ie_data, TH_set_dpram_ie_on

TH_set_dpram_ie_on

int TH_set_dpram_ie_on()

Return Value

Error code.

Description

Sets DPRAM_ERR_IE bit in HIF_IM_RG register. The function is valid only for PCI boards.

Example

```
...
#include "th.h"
…
if(!TH_dpram_ie_data())
      TH_set_dpram_ie_on();
…
```

See Also

TH_dpram_ie_data, TH_set_dpram_ie_off

TH_set_dpsem

int TH_set_dpsem(ULONG addr,ULONG value)

Parameters

add

DPSEM address to write to.

value

The value to write.

Return Value

Error code.

Description

Writes 32-bit word to DPSEM and is used to random access to DPSEM area. The function is valid only for PCI boards.

Example

```
…
#include "th.h"

ULONG sem;value[TP6X_DPM_DPSEM_LEN];
...
for(sem = 0;sem < TP6X_DPM_DPSEM_LEN;sem++){
    value[sem] = TH_dpsem_data(sem);
}
...
for(sem = 0;sem < TP6X_DPM_DPSEM_LEN;sem++){
    TH_set_dpsem(sem, value[sem]);
}
...
```

See Also

TH_dpsem_data

TH_set_dsp_amen_off

Int TH_set_dsp_amen_off ()

Return Value

Error code.

Description

Resets DSP_AMEN bit in HIF_CNTR_RG register. The function is valid only for PCI boards.

Example

```
...
#include "th.h"
…
if(TH_dsp_amen_data())
      TH_set_dsp_amen_off();
…
```

See Also

TH_dsp_amen_data, TH_set_dsp_amen_on

TH_set_dsp_amen_on

Int TH_set_dsp_amen_on()

Return Value

Error code.

Description

Sets DSP_AMEN bit in HIF_CNTR_RG register. The function is valid only for PCI boards.

Example

```
...
#include "th.h"
…
if(!TH_dsp_amen_data())
      TH_set_dsp_amen_on();
…
```

See Also

TH_dsp_amen_data, TH_set_dsp_amen_off

TH_set_emu_io_baddr

int TH_set_emu_io_baddr(UCHAR index)

Parameters

index

The index of corresponding EMU IO base address to set.

Return Value

Error code.

Description

Sets specified EMU IO base address. Base addresses are in definite correspondence with indices (0-0,1-0,2-0,3-0,4-0x240,5-0x280,6-0x320,7-0x340). The function is valid only for ISA boards.

Example

```
…
#include "th.h"

int err;
...
err = TH_set_emu_io_baddr(5);
if(err) {
    // do something
}
...
```

See Also

TH_emu_io_baddr

TH_set_emu_mdsp

int TH_set_emu_mdsp()

Return Value

Error code.

Description

Sets UECM to on-board DSP. The function is valid only for ISA boards.

Example

```
…
#include “th.h”

int err;
...
err = TH_set_emu_mdsp();
if(err) {
    // do something
}
...
```

See Also

TH_set_emu_xdsp

TH_set_emu_reset

int TH_set_emu_reset()

Return Value

Error code.

Description

Sets UECM/ECC reset. The function is valid only for ISA boards.

Example

```
…
#include "th.h"

int err;
...
err = TH_set_emu_reset();
if(err) {
    // do something
}
...
```

TH_set_emu_sel_ecc

Int TH_set_emu_sel_ecc()

Return Value

Error code.

Description

Selects ECC for the currently selected board. The function is valid only for PCI boards.

Example

```
…
#include “th.h”
...
TH_set_emu_sel_ecc();
...
```

See Also

TH_set_emu_sel_xemu

TH_set_emu_sel_rg

int TH_set_emu_sel_rg(UCHAR value)

Parameters

value

The value to set.

Return Value

Error code.

Description

Sets EMU_SEL_RG register to the value specified. The function is valid only for *TORNADO*-P6X series.

Example

```
…
#include "th.h"
...
TH_set_emu_sel_rg(0);
...
```

See Also

TH_set_emu_sel_xemu, TH_set_emu_sel_ecc

TH_set_emu_sel_xemu

int TH_set_emu_sel_xemu()

Return Value

Error code.

Description

Selects XEMU for the currently selected board. The function is valid only for PCI boards.

Example

```
…
#include “th.h”

int err;
...
err = TH_set_emu_sel_xemu();
if(err) {
    // do something
}
...
```

See Also

TH_set_emu_sel_ecc

TH_set_emu_xdsp

int TH_set_emu_xdsp ()

Return Value

Error code.

Description

Sets UECM/(T3X, T548, T6X) to external DSP or sets ECC/(T31M, T542L, T6XMX) off.. The function is valid only for ISA boards.

Example

```
…
#include "th.h"

int err;
...
err = TH_set_emu_xdsp();
if(err) {
    // do something
}
...
```

See Also

TH_set_emu_mdsp

TH_set_fdata_rg

int TH_set_fdata_rg(UCHAR value)

Parameters

value

The value to set.

Return Value

Error code.

Description

Sets FDATA_RG register to the value specified. The function is valid only for ISA boards.

Example

```
…
#include "th.h"
...
TH_set_fdata_rg(0);
...
```

See Also

TH_fdata_rg

TH_set_frg_verif

int TH_set_frg_verif(UCHAR frg,UCHAR value)

Parameters

frg

FRG register to write to.

value

The value to write.

Return Value

Error code.

Description

Sets FRG register to the value and checks result of writing. The function is valid only for ISA boards.

Example

```
…
#include "th.h"

int err;
...
err = TH_set_frg_verif(T54X_CLR_SB_ERR_FRG,0);
if(err){
    // process the error
}
...
```

See Also

TH_frg_data

TH_set_fsel_rg

int TH_set_fsel_rg(UCHAR value)

Parameters

value

The value to set.

Return Value

Error code.

Description

Sets FSEL_RG register to the value specified. The function is valid only for ISA boards.

Example

```
…
#include "th.h"
...
TH_set_fsel_rg(T54X_CLR_SB_ERR_FRG);
...
```

See Also

TH_set_fsel_rg_verif, TH_fsel_rg

TH_set_fsel_rg_verif

int TH_set_fsel_rg_verif(UCHAR frg)

Parameters

frg

The value to set.

Return Value

Error code.

Description

Sets FSEL_RG register to the value specified and checks if writing succeed or not. The function is valid only for ISA boards.

Example

```
…
#include "th.h"

int err;
...
err = TH_set_fsel_rg_verif(T54X_CLR_SB_ERR_FRG);
if(err){
    // process the error
}
...
```

See Also

TH_set_fsel_rg

TH_set_hm_rq

int ***TH_set_hm_rq ()***

Return Value

Error code.

Description

Sets host to master request (only for ISA boards).

Example

```
…
#include "th.h"
...
TH_set_hm_rq();
...
```

See Also

TH_hm_rq0_rg, TH_set_hm_rq0_rg, TH_hm_rq1_rg, TH_set_hm_rq1_rg

TH_set_hm_rq0_rg, TH_set_hm_rq1_rg

TH_set_hm_rq0_rg(UCHAR value), TH_set_hm_rq1_rg(UCHAR value)

Parameters

value

The value to set.

Return Value

Error code.

Description

Sets HM_RQ0_RG/ HM_RQ1_RG register to the value specified. Actually generate active HM_RQ0/HM_RQ1 PCI-to-DSP interrupt. The function is valid only for PCI boards.

Example

```
…
#include "th.h"
...
TH_set_hm_rq0_rg(0);
...
```

See Also

TH_hm_rq0_rg, TH_hm_rq1_rg

TH_set_hpi_disable

int TH_set_hpi_disable ()

Return Value

Error code.

Description

Disables HPI port. Only for *TORNADO*-5402 boards.

Example

```
…
#include "th.h"
...
TH_set_hpi_disable();
...
```

See Also

TH_set_hpi_enable

TH_set_hpi_enable

int TH_set_hpi_enable()

Return Value

Error code.

Description

Enables HPI port. Only for *TORNADO*-5402 boards.

Example

```
…
#include "th.h"
...
TH_set_hpi_enable();
...
```

See Also

TH_set_hpi_disable

TH_set_hpi_err_ie

int TH_set_hpi_err_ie()

Return Value

Error code.

Description

Sets HPI_ERR_IE bit in HIF_IM_RG register for PCI boards and in HPI_IE_FRG register for ISA boards. The function isn't valid for *TORNADO*-3X boards.

Example

```
...
#include "th.h"
…
if(!TH_hpi_err_ie_data())
    TH_set_hpi_err_ie();
…
```

See Also

TH_hpi_err_ie_data, TH_clr_hpi_err_ie

TH_set_hpi_hint_ie

int TH_set_hpi_hint_ie()

Return Value

Error code.

Description

Sets HPI_HINT_IE bit in HIF_IM_RG register for PCI boards and in HPI_IE_FRG register for ISA boards. The function isn't valid for *TORNADO*-3X boards.

Example

```
...
#include "th.h"
…
if(!TH_hpi_hint_ie_data ())
    TH_set_hpi_hint_ie();
…
```

See Also

TH_hpi_hint_ie_data, TH_clr_hpi_hint_ie

TH_set_hpia_rg

int TH_set_hpia_rg(ULONG value)

Parameters

value

The value to set.

Return Value

Error code.

Description

Sets HPIA_RG register to the address specified. The function isn't valid for *TORNADO*-3X boards.

Example

```
...
#include "th.h"
…
TH_set_hpia_rg(0);
…
```

See Also

TH_hpia_rg

TH_set_hpic_bob

int TH_set_hpic_bob ()

Return Value

Error code.

Description

Sets HPIC_BOB bit in HPIC LSB and MSB registers. The function is valid only for *TORNADO*-54X boards.

Example

```
...
#include "th.h"
…
TH_set_hpic_bob();
…
```

TH_set_hpic_dspint

int TH_set_hpic_dspint()

Return Value

Error code.

Description

Interrupts DSP via HPI. The function isn't valid for *TORNADO*-3X boards.

Example

```
…
#include "th.h"
...
TH_set_hpic_dsp_int();
...
```

See Also

TH_hpic_dspint_data

TH_set_hpic_fetch

int TH_set_hpic_fetch ()

Return Value

Error code.

Description

Sets FETCH bit in HPIC register. The function is valid only for *TORNADO*-6X/P6X boards.

Example

```
...
#include "th.h"
…
TH_set_hpic_fetch();
…
```

See Also

TH_hpic_fetch_data

TH_set_hpic_hwob

int TH_set_hpic_hwob()

Return Value

Error code.

Description

Sets HWOB bit in HPIC register. The function is valid only for *TORNADO*-6X/P6X boards.

Example

```
...
#include "th.h"
…
TH_set_hpic_hwob();
…
```

See Also

TH_hpic_hwob_data

TH_set_hpic_rg

int TH_set_hpic_rg(ULONG value)

Parameters

value

The value to set.

Return Value

Error code.

Description

Sets HPIC_RG register to the value specified. The function isn't valid for *TORNADO*-3X boards.

Example

```
…
#include "th.h"

ULONG old_rg;
...
old_rg = TH_hpic_rg();
...
TH_set_hpic_rg(old_rg);
...
```

See Also

TH_hpic_rg

TH_set_hpic_xhpia

int TH_set_hpic_xhpia()

Return Value

Error code.

Description

Sets XHPIA bit of HPIC MSB and LSB registers. The function has the meaning only for *TORNADO*-54X boards series.

Example

```
...
#include "th.h"
…
TH_set_hpic_xhpia();
…
```

See Also

TH_clr_hpic_xhpia

TH_set_hpid_ainc_rg

int TH_set_hpid_ainc_rg(ULONG value)

Parameters

value

The value to set.

Return Value

Error code.

Description

Sets HPID_AINC_RG register to the value specified. The function isn't valid for *TORNADO*-3X boards.

Example

```
…
#include "th.h"
...
TH_set_hpid_ainc_rg(0);
...
```

See Also

TH_hpid_ainc_rg

TH_set_hpid_rg

int TH_set_hpid_rg(ULONG value)

Parameters

value

The value to set.

Return Value

Error code.

Description

Sets HPID_RG register to the value specified. The function isn't valid for *TORNADO*-3X boards.

Example

```
…
#include "th.h"
...
TH_set_hpid_rg(0xffffffff);
...
```

See Also

TH_hpid_rg ,TH_hpia_rg, TH_hpid_ainc_rg

TH_set_im_rg

int TH_set_im_rg(UCHAR value)

Parameters

value

The value to set.

Return Value

Error code.

Description

Sets HIF_IM_RG register to the value specified. The function is valid only for *TORNADO*-P6X boards.

Example

```
…
#include "th.h"

UCHAR old_rg;
...
old_rg = TH_im_rg();
...
TH_set_im_rg(old_rg);
...
```

See Also

TH_im_rg

TH_set_mgo

int TH_set_mgo ()

Return Value

Error code.

Description

Runs dsp program.

Example

```
...
#include "th.h"
…
TH_set_mgo();
…
```

See Also

TH_set_mreset, TH_mreset_data

TH_set_mh_rq_ie_off

int TH_set_mh_rq_ie_off()

Return Value

Error code.

Description

Disables MH_RQ interrupts to come. The function is valid only for ISA boards.

Example

```
...
#include "th.h"
…
if(TH_mh_rq_ie_data())
    TH_set_mh_rq_ie_off();
…
```

See Also

TH_mh_rq_ie_data, TH_set_mh_rq_ie_on

TH_set_mh_rq_ie_on

int TH_set_mh_rq_ie_on()

Return Value

Error code.

Description

Enables MH_RQ interrupts to come. The function is valid only for ISA boards.

Example

```
...
#include "th.h"
…
if(!TH_mh_rq_ie_data())
    TH_set_mh_rq_ie_on();
…
```

See Also

TH_mh_rq_ie_data, TH_set_mh_rq_ie_off

TH_set_mreset

int TH_set_mreset ()

Return Value

Error code.

Description

Drives DSP to reset state.

Example

```
...
#include "th.h"
…
TH_set_mreset();
…
```

See Also

TH_mreset_data, TH_set_mgo

TH_set_mreset_mode_host

int TH_set_mreset_mode_host()

Return Value

Error code.

Description

Resets MRESET_MODE bit in HIF_CNTR_RG register. The function is valid only for PCI boards.

Example

```
…
#include “th.h”
...
TH_set_mreset_mode_host();
...
```

See Also

TH_set_mreset_mode_sa

TH_set_mreset_mode_sa

int TH_set_mreset_mode_sa()

Return Value

Error code.

Description

Sets MRESET_MODE bit in HIF_CNTR_RG register. The function is valid only for PCI boards.

Example

```
…
#include "th.h"
...
TH_set_mreset_mode_sa();
...
```

See Also

TH_set_mreset_mode_host

TH_set_pcic_intcsr_rg

int TH_set_pcic_intcsr_rg(ULONG value)

Parameters

value

The value to write.

Return Value

Error code.

Description

Sets AMCC PCI contoller INTCSR register to the value specified. The function is valid only for *TORNADO*-P6X boards series.

Example

```
…
#include “th.h”

ULONG old_rg;
...
old_rg = TH_pcic_intcsr_rg();
...
TH_set_pcic_intcsr_rg(0);
...
TH_set_pcic_intcsr_rg(old_rg);
...
```

See Also

TH_pcic_intcsr_rg

TH_set_pcic_fifo_rg

int TH_set_pcic_fifo_rg(ULONG value)

Parameters

value

The value to write.

Return Value

Error code.

Description

Sets AMCC PCI contoller FIFO register to the value specified. The function is valid only for *TORNADO*-P6X boards series.

Example

```
…
#include "th.h"
...
TH_set_pcic_fifo_rg(0xff00ff00);
...
```

See Also

TH_pcic_fifo_rg

TH_set_pcic_mcsr_rg

int TH_set_pcic_mcsr_rg(ULONG value)

Parameters

value

The value to write.

Return Value

Error code.

Description

Sets AMCC PCI contoller MCSR register to the value specified. The function is valid only for *TORNADO*-P6X boards series.

Example

```
…
#include "th.h"

ULONG old_rg;
...
old_rg = TH_pcic_mcsr_rg();
...
TH_set_pcic_mcsr_rg(old_rg);
...
```

See Also

TH_pcic_mcsr_rg

TH_set_pcic_ombx_rg

int TH_set_pcic_ombx_rg(ULONG n,ULONG value)

Parameters

n

Output mailbox number.

value

The value to write.

Return Value

Error code.

Description

Sets AMCC PCI contoller OMBX register to the value specified. The function is valid only for *TORNADO*-P6X boards series.

Example

```
…
#include "th.h"
...
TH_set_pcic_ombx_rg(0,0xdfdfdfdf);
...
```

See Also

TH_pcic_ombx_rg

TH_set_sb_ccl_byte

int TH_set_sb_ccl_byte()

Return Value

Error code.

Description

Sets 8-bit format of SB data cycle. The function is valid only for *TORNADO*-3X/6X boards series.

Example

```
…
#include "th.h"
...
TH_set_sb_ccl_byte();
...
```

See Also

TH_set_sb_ccl_word, TH_set_sb_ccl_dword

TH_set_sb_ccl_dword

int TH_set_sb_ccl_dword()

Return Value

Error code.

Description

Sets 32-bit format of SB data cycle. The function is valid only for *TORNADO*-3X/6X boards series.

Example

```
…
#include “th.h”
...
TH_set_sb_ccl_dword();
...
```

See Also

TH_set_sb_ccl_word, TH_set_sb_ccl_byte

TH_set_sb_ccl_word

int TH_set_sb_ccl_word()

Return Value

Error code.

Description

Sets 16-bit format of SB data cycle. The function is valid only for *TORNADO*-3X/6X boards series.

Example

```
…
#include “th.h”
...
TH_set_sb_ccl_word();
...
```

See Also

TH_set_sb_ccl_dword, TH_set_sb_ccl_byte

TH_set_sb_err_ie_off

int TH_set_sb_err_ie_off()

Return Value

Error code.

Description

Disables interrupts on SB error active flag. The function is valid only for ISA boards.

Example

```
...
#include "th.h"
…
if(TH_sb_err_ie_data())
    TH_set_sb_err_off();
…
```

See Also

TH_sb_err_ie_data, TH_set_sb_err_ie_on

TH_set_sb_err_ie_on

int TH_set_sb_err_ie_on()

Return Value

Error code.

Description

Enables interrupts on SB error active flag. The function is valid only for ISA boards.

Example

```
...
#include "th.h"
…
if(!TH_sb_err_ie_data())
    TH_set_sb_err_on();
…
```

See Also

TH_sb_err_ie_data, TH_set_sb_err_ie_off

TH_set_sb_glock_off

int TH_set_sb_glock_off()

Return Value

Error code.

Description

Clears SB_GLOCK bit of CONTROL REGISTER and unlocks SB. The function is valid only for ISA boards.

Example

```
...
#include "th.h"
…
if(TH_sb_glock_data())
    TH_set_sb_glock_off();
…
```

See Also

TH_sb_glock_data, TH_set_sb_glock_on

TH_set_sb_glock_on

int TH_set_sb_glock_on()

Return Value

Error code.

Description

Sets SB_GLOCK bit of CONTROL REGISTER for immediate active SB locking.. The function is valid only for ISA boards.

Example

```
...
#include "th.h"
…
if(!TH_sb_glock_data())
    TH_set_sb_glock_on();
…
```

See Also

TH_sb_glock_data, TH_set_sb_glock_off

TH_set_sb_lock_off

int TH_set_sb_lock_off()

Return Value

Error code.

Description

Clears SB_LOCK bit of CONTROL REGISTER and unlocks SB. The function is valid only for ISA boards.

Example

```
...
#include "th.h"
…
if(TH_sb_lock_data())
    TH_set_sb_lock_off();
…
```

See Also

TH_sb_lock_data, TH_set_sb_lock_on

TH_set_sb_lock_on

int TH_set_sb_lock_on()

Return Value

Error code.

Description

Sets SB_GLOCK bit of CONTROL REGISTER and issue active SB locking during nearest SB access from host ISA-bus memory interface. The function is valid only for ISA boards.

Example

```
...
#include "th.h"
…
if(!TH_sb_lock_data())
    TH_set_sb_lock_on();
…
```

See Also

TH_sb_lock_data, TH_set_sb_lock_off

TH_set_smp_dpram_area

int TH_set_smp_dpram_area(UCHAR smp)

Parameters

smp

SMP register 0 or 1.

Return Value

Error code.

Description

Selects DPRAM area. The function is valid only for PCI boards

Example

```
…
#include “th.h”
...
TH_set_smp_dpram_area(0);
...
```

See Also

TH_set_smp_dpsem_area

TH_set_smp_dpsem_area

int TH_set_smp_dpsem_area(UCHAR smp)

Parameters

smp

SMP register 0 or 1.

Return Value

Error code.

Description

Selects DPSEM area. The function is valid only for PCI boards.

Example

```
…
#include "th.h"
...
TH_set_smp_dpsem_area(0);
...
```

See Also

TH_set_smp_dpram_area

TH_set_smp_mode_dpa

int TH_set_smp_mode_dpa()

Return Value

Error code.

Description

Sets dual-page DMPRAM/DPSEM access mode. The function is valid only for PCI boards.

Example

```
…
#include "th.h"
...
TH_set_smp_mode_dpa();
...
```

See Also

TH_set_smp_mode_spa

TH_set_smp_mode_spa

int TH_set_smp_mode_spa()

Return Value

Error code.

Description

Sets single-page DMPRAM/DPSEM access mode. The function is valid only for PCI boards.

Example

```
…
#include "th.h"
...
TH_set_smp_mode_spa();
...
```

See Also

TH_set_smp_mode_dpa

TH_set_smp_off

int TH_set_smp_off()

Return Value

Error code.

Description

Sets off SMP segment. Accessing SMP after calling this functions will cause memory spoiling The function has the meaning only for ISA boards.

Example

```
…
#include "th.h"
...
TH_set_smp_segm(5);
...
TH_set_smp_off();
...
```

See Also

TH_set_smp_segm

TH_set_smp_page

int TH_set_smp_page(UCHAR smp,UCHAR value)

Parameters

smp

The number of SMP register (0 or 1).

value

The value to write.

Return Value

Error code.

Description

Writes the value specified to the selected SMP PAGE register. The function is valid only for PCI boards.

Example

```
…
#include "th.h"
...
TH_set_smp_page(0,0x01);
...
```

See Also

TH_smp_page_data

TH_set_smp_ptr

int TH_set_smp_ptr(ULONG address,UCHAR area,PVOID* p,UCHAR smp = 0)

Parameters

address

An DSP address within a specified area range.

area

An area to access.

p

Virtlual address corresponding specified DSP address to access it from Win32 applications.

smp

SMP register.

Return Value

Error code.

Description

The function maps user defined DSP area to virtual address spaces of Win32 application. First **TH_set_smp_segm** function must be called for ISA boards. SMP number (0 or 1) has the meaning only for PCI boards.

Example

```
...
#include "th.h"
…
int err,i;
PULONG vtaddr;
err = TH_set_smp_segm(5); // 0xd8000
if(err) return; // pocess the error
…
err = TH_set_spm_ptr(0x0000, T6X_SMP_SRAM_AREA,(void**)&vtaddr);
if(!err) for(i = 0; i<100; i++) vtaddr[i] = i*10; // actual writing
…
```

See Also

TH_set_smp_off, TH_set_smp_segm

TH_set_smp_rg

int TH_set_smp_rg(USHORT value)

Parameters

value

The value to set.

Return Value

Error code.

Description

Sets SMP_RG register to the value specified.

Example

```
…
#include "th.h"

UCHAR old_rg;
...
old_rg = TH_smp_rg();
...
TH_set_smp_rg(old_rg);
...
```

See Also

TH_smp_rg

TH_set_smp_segm

int TH_set_smp_segm(UCHAR index)

Parameters

index

The index of corresponding SMP segment address to set. Zero means to set off SMP segment.

Return Value

Error code.

Description

Sets specified SMP segment address which is in definite correspondence with indices (0-0x00000, 1-0xb8000, 2-0xc0000, 3-0xc8000, 4-0xd0000, 5-0xd8000, 6-0xe0000, 7-0xe8000). It possible to get described correspondence by using TH_smp_segm_table function. The functions discussed are valid only for ISA boards.

Example

```
…
#include "th.h"
...
TH_set_smp_segm(5); // corresponds to 0xd8000
...
```

See Also

TH_set_smp_off, TH_smp_segm_table

TH_set_xhpia

int TH_set_xhpia(USHORT value)

Parameters

value

The value to set.

Return Value

Error code.

Description

Writes to XHPIA register the value specified. The function has the meaning only for *TORNADO*-54X boards series.

Example

```
…
#include "th.h"
...
TH_set_xhpia(0x1234);
...
```

See Also

TH_xhpia_reset

TH_smp_area_data

BOOLEAN TH_smp_area_data(UCHAR smp)

Parameters

smp

The number of SMP register (0 or 1).

Return Value

TRUE – if DPRAM area is selected, FALSE – DPSEM. In case of an error – 0xff value is returned.

Description

Allows user to check which of SMP DPRAM/DPSEM area is selected. The function is valid only for PCI boards.

Example

```
…
#include "th.h"
...
if(TH_smp_area_data(0)){
    // do something or do nothing
}
```

See Also

TH_set_smp_dpram_area, TH_set_smp_dpsem_area

TH_smp_mode_data

BOOLEAN TH_smp_mode_data()

Return Value

TRUE – if SPA mode is selected, FALSE – DPA. In case of an error – 0xff value is returned.

Description

Allows user to check which of SMP access mode is selected – single page or dual page (SPA/DPA). The function is valid only for PCI boards.

Example

```
…
#include "th.h"
...
if(TH_smp_mode_data()){
    // do something or do nothing
}
```

See Also

TH_set_smp_mode_spa, TH_set_smp_mode_dpa

TH_smp_page_data

UCHAR TH_smp_page_data(UCHAR smp)

Parameters

smp

The number of SMP page register to read.

Return Value

The value of selected SMP page register if success. In case of an error – 0xff value is returned.

Description

Reads the selected SMP page register. The function is valid only for PCI boards.

Example

```
…
#include "th.h"

UCHAR smp_page_rg1;
...
smp_page_rg1 = TH_smp_page_data(1);
...
```

See Also

TH_set_smp_page

TH_smp_rg

USHORT TH_smp_rg()

Return Value

The value of the SMP register if success. In case of an error – 0xffff value is returned.

Description

Reads SMP register.

Example

```
…
#include “th.h”

USHORT smp_rg;
...
smp_rg = TH_smp_rg();
...
```

See Also

TH_set_smp_rg

TH_smp_segm

ULONG TH_smp_segm()

Return Value

The value of SMP segment address if success. In case of an error – 0xffffffff value is returned.

Description

Returns physical address which is set for the currently selected board. The function is valid only for ISA boards.

Example

```
…
#include "th.h"

ULONG smp_segm;
...
smp_segm = TH_smp_segm();
if(smp_segm && smp_segm != (ULONG)-1){
    printf("We are using 0x%06x SMP segment address");
}
...
```

See Also

TH_set_smp_segm

TH_smp_segm_table

ULONG TH_smp_segm_table(UCHAR index)

Parameters

index

SMP segment table index to get from a corresponding address.

Return Value

Address value if success, otherwise 0xffffffff value.

Description

Returns possible SMP adrresses to be set by its index (0-0x00000, 1-0xb8000, 2-0xc0000, 3-0xc8000, 4-0xd0000, 5-0xd8000, 6-0xe0000, 7-0xe8000).

Example

```
…
#include "th.h"

int index;
...
// if we want 0xe0000 address to be used
for(index = 0;index < TH_ISA_MAX_SMP_BADDRS;index++)
    if(0xe0000 == TH_smp_segm_table(index))
       TH_set_smp_segm(index);
...
```

See Also

TH_set_smp_segm, TH_sm_segm

TH_sram_len

ULONG TH_sram_len()

Return Value

SRAM length if success. In case of an error 0xffffffff value is returned.

Description

Returns current board SRAM legth. Only for *TORNADO*-P6X boards series.

Example

```
…
#include "th.h"

ULONG sram_len;
...
sram_len = TH_sram_len();
...
```

TH_sys_stat_frg_data

UCHAR TH_sys_stat_frg_data()

Return Value

Byte value. In case of an error 0xff value is returned.

Description

Returns corresponding combination of bits from SET_HM_RQ_FRG register that are ORed in result value (SB_ERR, SB_ACK, MH_RQ for T3X boards and additional HPI_HINT and HPI_ERR bits for T54X/T6X are used). Only for ISA boards series.

Example

```
…
#include "th.h"

UCHAR sys_stat;
...
sys_stat = TH_sys_stat_frg_data();
...
```

See Also

TH_frg_data, TH_set_frg_verif

TH_uecm_io_baddr_table

ULONG TH_uecm_io_baddr_table(UCHAR index)

index

UECM IO base adresses table index to get from a corresponding address.

Return Value

Address value if success, otherwise 0xffffffff value.

Description

Returns possible UECM IO base adresses to be set by its index (0- 0x240, 1-0x280, 2-0x320, 3-0x340).

Example

```
…
#include "th.h"

ULONG baddr;
...
baddr = TH_uecm_io_baddr_table(0);
...
```

See Also

TH_emu_io_baddr

TH_unmapmem_view

int TH_unmapmem_view(ULONG vtaddr)

Parameters

vtaddr

Points to a virtual address to be used in Win32 aplication which is obtained from **TH_mapmem_view** function.

Return Value

Error code.

Description

Disconnects Win32 application from common memory space.

Example

```
...
#include "th.h"
…
ULONG buf_ptr, BUF, size;
...
TH_mapmem_view(&BUF,&buf_ptr,&size);
if(err = TH_flag_err()) some_processing_error_function(err);
// some processing
...
TH_unmapmem_view(&BUF);
...
```

See Also

TH_mapmem_view

TH_write_byte

int TH_write_byte(UCHAR area,ULONG addr,UCHAR value)

Parameters

area

Area selector.

addr

Address offset within selected area.

value

Value to write.

Return Value

Error code.

Description

Writes byte to an address within selected area.

Parameter *area* can take the following values:

Area	description
TH_PCI_PCIC_AREA	AMCC registers area
TH_PCI_HIF_AREA	Host Interface area
TH_PCI_ECC_AREA	On-board emulator interface area
TH_PCI_DPM_AREA	DPRAM area
TH_ISA_IO_AREA	ISA operation registers
TH_ISA_EMU_IO_AREA	On-board emulator interface area
TH_ISA_SMP_AREA	SMP area

Example

```
...
#include "th.h"
…
```

```
UCHAR value_old;
TH_read_byte(TH_ISA_IO_AREA, TH_ISA_SMP_RG, &value_old);
TH_write_byte(TH_ISA_IO_AREA, TH_ISA_SMP_RG, 0);
…
```

See Also

TH_read_byte, TH_read_word, TH_write_word, TH_read_dword, TH_write_dword

TH_write_dword

int TH_write_dword(UCHAR area,ULONG addr,ULONG value)

Parameters

area

Area selector.

addr

Address offset within selected area.

value

Value to write.

Return Value

Error code.

Description

Writes 32-bit word to an address within selected area. It doesn't matter for function whether selected area supports 32-bit acces or not.

Parameter *area* can take the following values:

Area	**description**
TH_PCI_PCIC_AREA	AMCC registers area
TH_PCI_HIF_AREA	Host Interface area
TH_PCI_ECC_AREA	On-board emulator interface area
TH_PCI_DPM_AREA	DPRAM area
TH_ISA_IO_AREA	ISA operation registers
TH_ISA_EMU_IO_AREA	On-board emulator interface area
TH_ISA_SMP_AREA	SMP area

Example

```
...
#include "th.h"
```

```
…
TH_write_dword(TH_PCI_PCIC_AREA, TH_AMCC_FIFO_RG, 0x12);
…
```

See Also

TH_read_byte, TH_write_byte, TH_read_word, TH_write_word, TH_read_dword

TH_write_word

int TH_write_word(UCHAR area,ULONG addr,USHORT value)

Parameters

area

Area selector.

addr

Address offset within selected area.

value

Value to write.

Return Value

Error code.

Description

Writes 16-bit word to an address within selected area.

Parameter *area* can take the following values:

Area	**description**
TH_PCI_PCIC_AREA	AMCC registers area
TH_PCI_HIF_AREA	Host Interface area
TH_PCI_ECC_AREA	On-board emulator interface area
TH_PCI_DPM_AREA	DPRAM area
TH_ISA_IO_AREA	ISA operation registers
TH_ISA_EMU_IO_AREA	On-board emulator interface area
TH_ISA_SMP_AREA	SMP area

Example

. . .

```
#include "th.h"
…
UCHAR value_old;
TH_read_word(TH_ISA_IO_AREA, TH_ISA_SMP_RG, &value_old);
TH_write_word(TH_ISA_IO_AREA, TH_ISA_SMP_RG, 0);
…
```

See Also

TH_read_byte, TH_write_byte, TH_read_word, TH_read_dword, TH_write_dword

TH_xhpia_reset

int TH_xhpia_reset()

Return Value

Error code.

Description

Clears XHPIA register. The function has the meaning only for *TORNADO*-54X boards series.

Example

```
…
#include “th.h”

int err;
char * err_str;
...
err = TH_xhpia_reset();
if(err){
    err_str = TH_err_mesage(err);
    // do something with it
}
...
```

See Also

TH_set_xhpia