Driver Manual

cifX Device Driver

IntervalZero RTX®

V8.1 / 2009 / 2011

Hilscher Gesellschaft für Systemautomation mbH

www.hilscher.com

DOC090702DRV04EN | Revision 4 | English | 2011-10 | Released | Public

Table of Contents

1 Introduction

1.1 About this Document

This manual describes the Hilscher cifX Device Driver for IntervalZero RTX® and its architecture. The driver offers access to the Hilscher netX based hardware with the same functional API as the cifX device driver for Windows® and offers transparent access to the different devices. IntervalZero's RTX® software is a **R**eal **T**ime e**X**tension for the Windows Vista, Windows 2000, Windows XP and Windows Server 2003 environment.

The Hilscher RTX® driver is represented as a dynamic loadable library, which supports different netX based hardware designs. First design is a cifX PCI/PCIe bus based PC card. This card is not using any FLASH memory and the driver is responsible to download the necessary firmware and configuration files during the startup phase of the hardware, to get the boards in operational mode. The second type of hardware, which is supported by the driver, is the NXSB-PCA and NX-PCA-PCI boards. These boards enable the connection of a NXSB 100 / NXHX board to the PCI bus. In addition, to the PCI based hardware, the driver also supports devices, connected via the ISA bus.

1.2 List of Revisions

Rev	Date	Name	Chapter	Revisions
1	2009-07-23	SS		created
2	2009-09-08	SS	5	Chapter 'Frequently Asked Questions' added
3	2010-06-09	SS	2.1 2.1, 2.1.1 2.1, 4 3.2 2.1, 2.1.1, 5 2.1	Support for DMA mode Support for loadable modules Note about use of registry file added Return types of additional functions adapted to stdint data types Slot number support NXPCA-PCI Timings parameter support
4	2011-10-21	SS/RM	1.5 2 2.1, 2.1.x	Support for netJACK 100 Guideline for device configuration revised Support for RTX 2011 - Device configuration is obtained directly from windows registry

Table 1: List of Revisions

1.3 Overview

The cifX Device Driver for IntervalZero RTX® is available as a dynamic library built around the cifX Toolkit. Any application which needs to access a cifX device can use the device specific functions provided by this driver library. The dynamic library is implemented as a RTDLL, which is the analog of an explicitly loaded Win32 DLL. User processes gain access to the cifX driver functions by using the *LoadLibrary()* and *GetProcAddress()* calls. The concept of the cifX device driver is illustrated in the following figure.

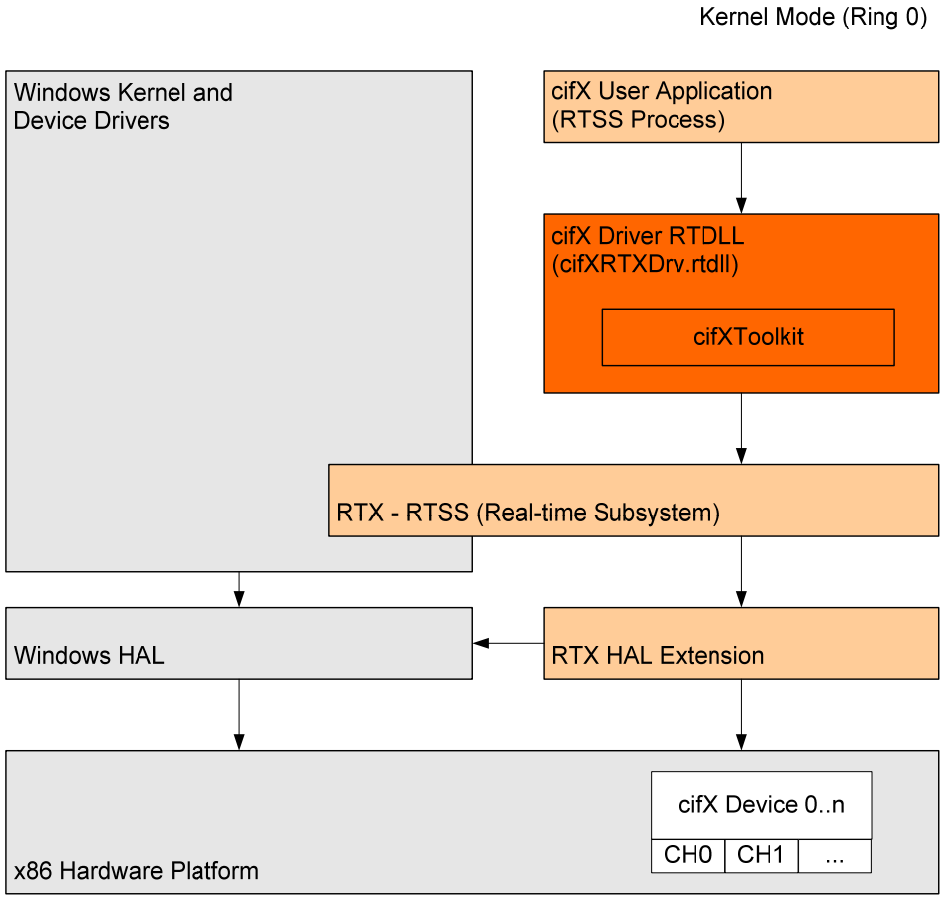

Figure 1: cifX Device Driver Architecture

1.4 Requirement

- IntervalZero RTX® version 8.1, 2009 or 2011
- cifX Device Driver for Windows® 7/Vista/XP/2000
- Microsoft Visual 2005 or higher for driver test /cifX TCP Server application

1.5 Supported Hardware

- cifX board (PCI / PCIe)
- netPLC
- NXSB-PCA / NXSB100 / NXHX board
- NX-PCA-PCI / NXHX
- CIFX 104 (ISA)
- netJACK 100

1.6 Features

- Based on the cifX Toolkit source (V1.1.0.0)
- Interrupt support for cifX boards
- DMA data transfer for I/O data
- Support for loadable modules
- Interrupt notification for applications

1.7 Limitations

- Only one RTX® process can use the driver at the same time
- Only little-endian hosts supported
- No 64 bit support (RTX Runtime only supports 32-bit operating systems)
- No DMA support for NXSB-PCA, NX-PCA-PCI and CIFX104

1.8 CD Contents

Folder		Content
Documentation		Driver documentation
Driver		Driver runtimes
	RTX 8.1	Driver runtimes for RTX 8.1
	RTX 2009	Driver runtimes for RTX 2009
	RTX 2011	Driver runtimes for RTX 2011
Examples		Examples
	Example Configuration File	Example card configuration registry file
	Test Application	Source code for driver test application
	cifXTCPServer	Source code for cifX TCP Server application

Table 2: CD Contents

1.9 Terms and Abbreviations

Term	Description
cifX	**C**ommunication **I**nter**f**ace based on netX
comX	**Co**mmunication **M**odule based on netX
PCI	**P**eripheral **C**omponent **I**nterconnect
API	**A**pplication **P**rogramming **I**nterface
DPM	**D**ual-**P**ort-**M**emory Physical interface to all communication board (DPM is also used for PROFIBUS-**DP M**aster).

Table 3: Terms and Abbreviations

1.10 References to Documents

This manual is based on the following documents and specifications:

[1] Hilscher Gesellschaft für Systemautomation mbH: Driver Manual cifX Device Driver - Windows 2000/XP/Vista/7/CE V1.1.x.x. Revision 20, english, 2011

[2] Hilscher Gesellschaft für Systemautomation mbH: Operating Instruction Manual CIFX Device Driver - Installation and Operation V1.0.x.x. Revision 6, english 2010

[3] Hilscher Gesellschaft für Systemautomation mbH: Dual-Port Memory Interface Manual - netX based products. Revision 11, english, 2011

[4] IntervalZero Inc.: Working with Hardware Resource Limitations. Document Number: RTX-810-004

[5] IntervalZero Inc.: RTX Documentation - Using RTX Runtime –> Managing Devices -> Converting a Windows Device to an RTX Device

Table 4: References to Documents

1.11 Legal Notes

1.11.1 Copyright

1.11.2 Important Notes

The manual, accompanying texts and the documentation were created for the use of the products by qualified experts, however, errors cannot be ruled out. For this reason, no guarantee can be made and neither juristic responsibility for erroneous information nor any liability can be assumed. Descriptions, accompanying texts and documentation included in the manual do not present a guarantee nor any information about proper use as stipulated in the contract or a warranted feature. It cannot be ruled out that the manual, the accompanying texts and the documentation do not correspond exactly to the described features, standards or other data of the delivered product. No warranty or guarantee regarding the correctness or accuracy of the information is assumed.

We reserve the right to change our products and their specification as well as related manuals, accompanying texts and documentation at all times and without advance notice, without obligation to report the change. Changes will be included in future manuals and do not constitute any obligations. There is no entitlement to revisions of delivered documents. The manual delivered with the product applies.

Hilscher Gesellschaft für Systemautomation mbH is not liable under any circumstances for direct, indirect, incidental or follow-on damage or loss of earnings resulting from the use of the information contained in this publication.

1.11.3 Exclusion of Liability

The software was produced and tested with utmost care by Hilscher Gesellschaft für Systemautomation mbH and is made available as is. No warranty can be assumed for the performance and flawlessness of the software for all usage conditions and cases and for the results produced when utilized by the user. Liability for any damages that may result from the use of the hardware or software or related documents, is limited to cases of intent or grossly negligent violation of significant contractual obligations. Indemnity claims for the violation of significant contractual obligations are limited to damages that are foreseeable and typical for this type of contract.

It is strictly prohibited to use the software in the following areas:

- for military purposes or in weapon systems;
- for the design, construction, maintenance or operation of nuclear facilities;
- in air traffic control systems, air traffic or air traffic communication systems;
- in life support systems;
- in systems in which failures in the software could lead to personal injury or injuries leading to death.

We inform you that the software was not developed for use in dangerous environments requiring fail-proof control mechanisms. Use of the software in such an environment occurs at your own risk. No liability is assumed for damages or losses due to unauthorized use.

1.11.4 Export

The delivered product (including the technical data) is subject to export or import laws as well as the associated regulations of different counters, in particular those of Germany and the USA. The software may not be exported to countries where this is prohibited by the United States Export Administration Act and its additional provisions. You are obligated to comply with the regulations at your personal responsibility. We wish to inform you that you may require permission from state authorities to export, re-export or import the product.

1.11.5 Registered Trademarks

Please notice: RTX® is a registered trademark of IntervalZero.

Windows® Vista, Windows® 2000, Windows® XP and Windows® Server 2003 are registered trademarks of Microsoft Corporation.

All other mentioned trademarks are property of their respective legal owners.

2 Installation

To use a cifX device in the RTX® environment, it must be explicitly removed from Windows® device manager and prepared for RTX® support. Several steps must be performed to install the Hilscher RTX® driver:

Step 1: Install the cifX Device Driver for Windows® 2000 / Windows® XP. Consult reference [2] for further installation instructions.

Step 2: Configure your cifX device (see section *Device Configuration* on page 12) with the *cifX Setup Tool*, included with the cifX device driver for Windows®.

Note: An online connection to a cifX device is not available if the cifX device is already converted to a RTX® device. Thus, configuration of the cifX device should be done in advance.

Step 3: Add RTX® support for a cifX device.

- Start the *RTX Properties* dialogue from Windows® start menu
- Open the *'Hardware'* tab and enter the Plug and Play configuration via the *'Settings'* button.
- Select the cifX device which should be controlled with the RTX® software, choose *'Add RTX INF support'* and select *'Apply'*.

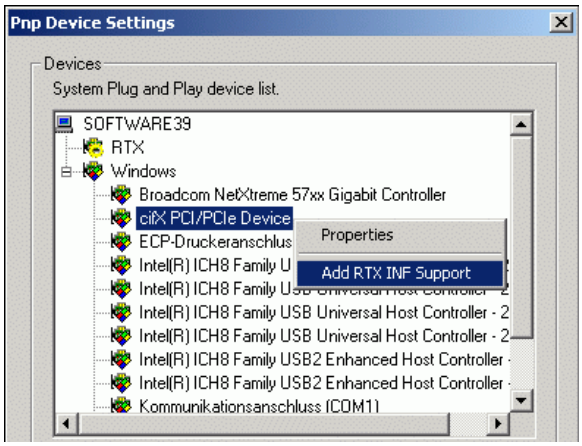

Figure 2: Add RTX® Support for a cifX Device

Step 4: Converting a Windows® Device to a RTX® Device (Windows® XP)

- Open the Windows® device manager
- Select the cifX device which should be converted and choose *'Update driver'*
- Select *'Let me pick from a list of device drivers on my computer'*
- To device drivers are listed: The standard cifX driver for Windows® and the cifX driver with RTX support. Select the cifX driver with RTX support to convert the selected cifX device to a RTX® device.

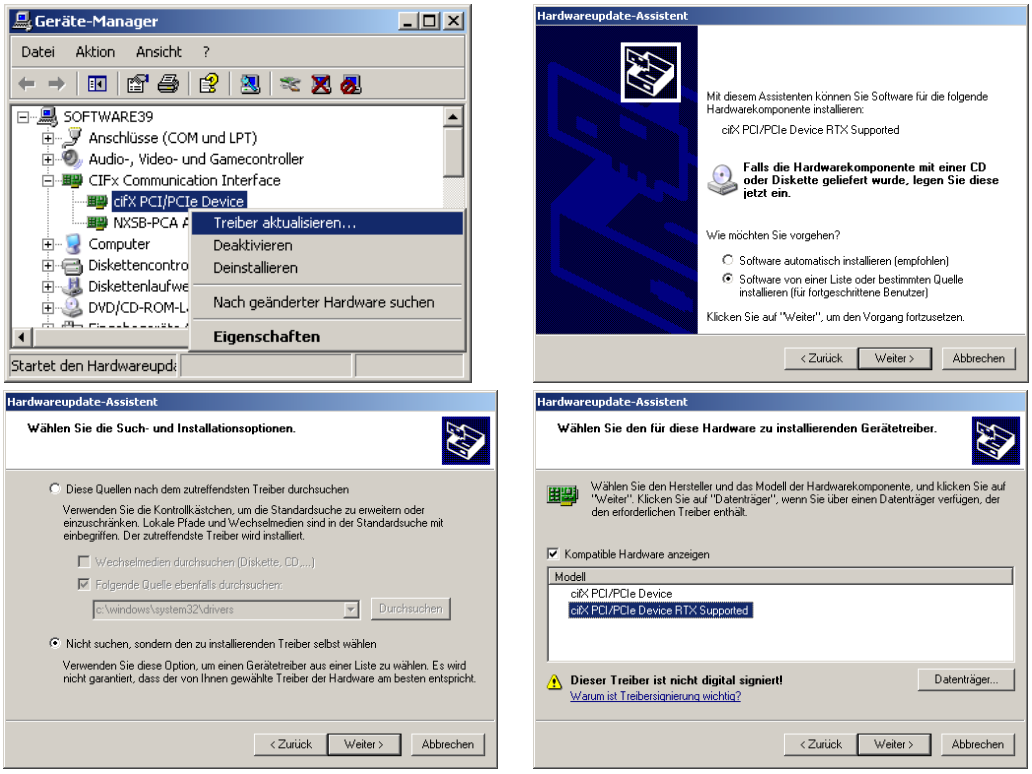

Figure 3: Convert the Windows® Device to a RTX® Device

Note: A detailed description of the conversion process for other Windows® systems is given in reference [5].

Step 5: Copy the cifX RTX® driver

- Copy the Hilscher RTX® driver DLL **cifXRTXDrv.rtdll** for your RTX® version from the installation CD to the local hard disk.

Step 6: Register the driver DLL

- The driver DLL **cifXRTXDrv.rtdll** must be registered within the RTX® environment. Therefore the RTX® program *RTSSrun* can be used. The RTX® environment will generate a local copy of the driver DLL and makes the driver DLL available for RTSS applications.

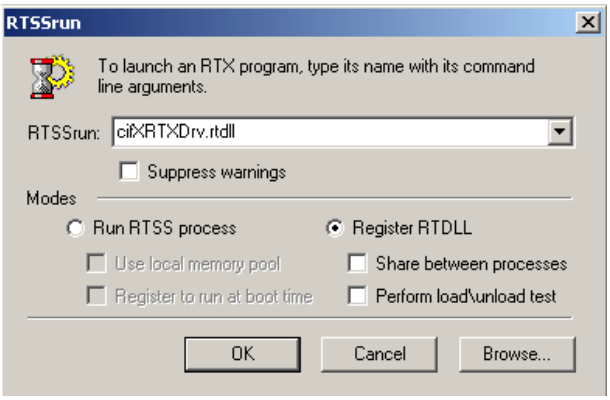

Figure 4: RTSSrun Program Example

Note: You have to repeat this step if you getting an updated driver DLL

Note: As the cifX driver DLL cannot be loaded by more than one RTX® process simultaneously, make sure that the option *'Share between processes'* is unchecked.

Annotations to interrupt support

The cifX device can be used in polling or interrupt mode. If interrupt mode should be used, IRQ resource limitations must be considered. As RTX® does not support sharing IRQ lines with Windows® devices, the interrupt line which RTX® uses must be available for exclusive use. Finding an exclusive IRQ often requires physically moving hardware around in the system or disabling other Windows® devices. If you have difficulty configuring your hardware for use with RTX® due to IRQ resource limitations, please consult reference [4] for further information.

2.1 Device Configuration

Each device has several parameters which can be configured, e.g. *Alias Name* or *Interrupt Support*. Some of the parameters are also important to get the device working, e.g. *Bootloader* and *firmware* and *fieldbus configuration files*.

Note: Perform the initial device configuration before assigning your cifX device to the RTX® environment, as the cifX device is no longer accessible via a Windows® process.

The device configuration will be done with the *cifX Driver Setup Utility*, included with the cifX device driver for Windows® and the complete device configuration information is stored in the Windows® registry.

Also the cifX device driver for RTX® needs these configuration and depending on the RTX® runtime version, access to the Windows® registry and therewith to the device configuration is available or not.

For RTX® versions without registry access (RTX® version prior RTX® 2011) the registry data must be exported into a file to be accessible by the cifX device driver for RTX®.

If the cifX device is already converted to a RTX® device, the online connection to the cifX device is no longer available. Because of this the initial device configuration must be done in advance.

2.1.1 Firmware and Configuration File Storage

cifX PCI cards are not using any flash memory to store a firmware or configuration on the card. Every time the card is powered-up the firmware and configuration must be downloaded to the hardware.

Note: Firmware and configurations are not stored on the hardware and must be downloaded each time the card is powered-up.

It is the task of the driver to initialize the card and therefore the driver has to know which files must be loaded to the hardware.

To allow device specific configuration, every file that needs to be downloaded must reside in the folder tree which is created by the cifX device driver for Windows® (or manually generates). The location of this folder tree is passed to the cifX RTX® driver via a configuration file (see section *Device Configuration* on page 12).

- **Use the *Slotnumber* (hardware rotary switch)**

  The *Slotnumber* serves to distinguish cifX cards from each other clearly, especially if more cifX cards are installed in one PC. The *Slotnumber* must be set at the cifX card using the Rotary Switch *Slotnumber*. While *Slotnumber* 0 means, that the cifX card is identified via its device and serial number, values from 1 to 9 corresponds to the *Slotnumber* 1 to 9. The firmware and configuration file must reside in the subdirectory *Slot_<1..9>*.

- **Use the device and serial number (default)**

  If the cifX device is not equipped with a rotary switch or the *Slotnumber* should not be used, the device is identified by its device and serial number. The firmware and configuration file must reside in the subdirectory /*<Device Number>_<Serial Number>*/.

The following table describes the different subdirectory levels created by the cifX device driver for Windows®:

Subdirectory	Description
<InstallDir>	Installation directory of the cifX device driver for Windows®. **Note:** This directory must contain the second stage PCI bootloader (e.g. NETX100-BSL.bin)
▪ <Device Nr.>_<Serial Nr.> ▪ Slot_<1..9>	Device and serial number of the device or slot number if the device provides a rotary switch. If the slot number is 0 the device and serial number is always used to identify the device. **Note:** This directory must contain the rcX base firmware, if loadable modules are used.
Channel<#>	Channel specific files (loadable modules, monolithic firmware files, fieldbus database files) Note**:** Currently only channel 0 is supported

Table 5: Firmware and Configuration File Storage

Sample file structure for a cifX device with device number 1250100 and serial number 20217:

```
+ <InstallDir>
|
|-- NETX100-BSL.BIN (second stage PCI bootloader)
|
|--+ 1250100_20217
   |
   |--+ Channel0
   |  |
   |  |--cifXdps.nxf (monolithic firmware)
   |  |--config.nxd (fieldbus database)
   |
   |--+ Channel1
   |--+ Channel2
   |--+ Channel3
   |--+ Channel4
   |--+ Channel5
   |
```

Sample file structure for a cifX device identified by Slot number 2 and loadable module support:

```
+ <InstallDir>
|
|-- NETX100-BSL.BIN (second stage PCI bootloader)
|
|--+ Slot_2
   |
   |--+ Channel0
   |  |
   |  |--nx100dpm.nxo (loadable module)
   |  |--config.nxd (fieldbus database)
   |
   |--+ Channel1
   |--+ Channel2
   |--+ Channel3
   |--+ Channel4
   |--+ Channel5
   |
   |-- cifXrcX.nxf (rcX base firmware)
```

2.1.2 Guideline to Create a Device Configuration

A cifX device needs a configuration defining the firmware and fieldbus configuration files for each cifX device. This chapter describes how to create a card configuration and how to make it available to the cifX device driver for RTX®.

Note: Creating a device configuration is done under the Windows® operating system.

- Configure your cifX device with the *cifX Driver Setup Utility*.
  A configuration instruction can be found in the *CIFX Device Driver Installation and Operation* manual (cifX Device Driver_usermanual_en.pdf).
  Start the utility from the Windows® *Control Panel* and insert the necessary information.
  Press the *Apply* button and confirm the request to restart the device.
  This will store the device configuration in the Windows® registry and create the required directory tree for the device configuration and firmware file storage (see section *Firmware and Configuration File Storage* on page 12).

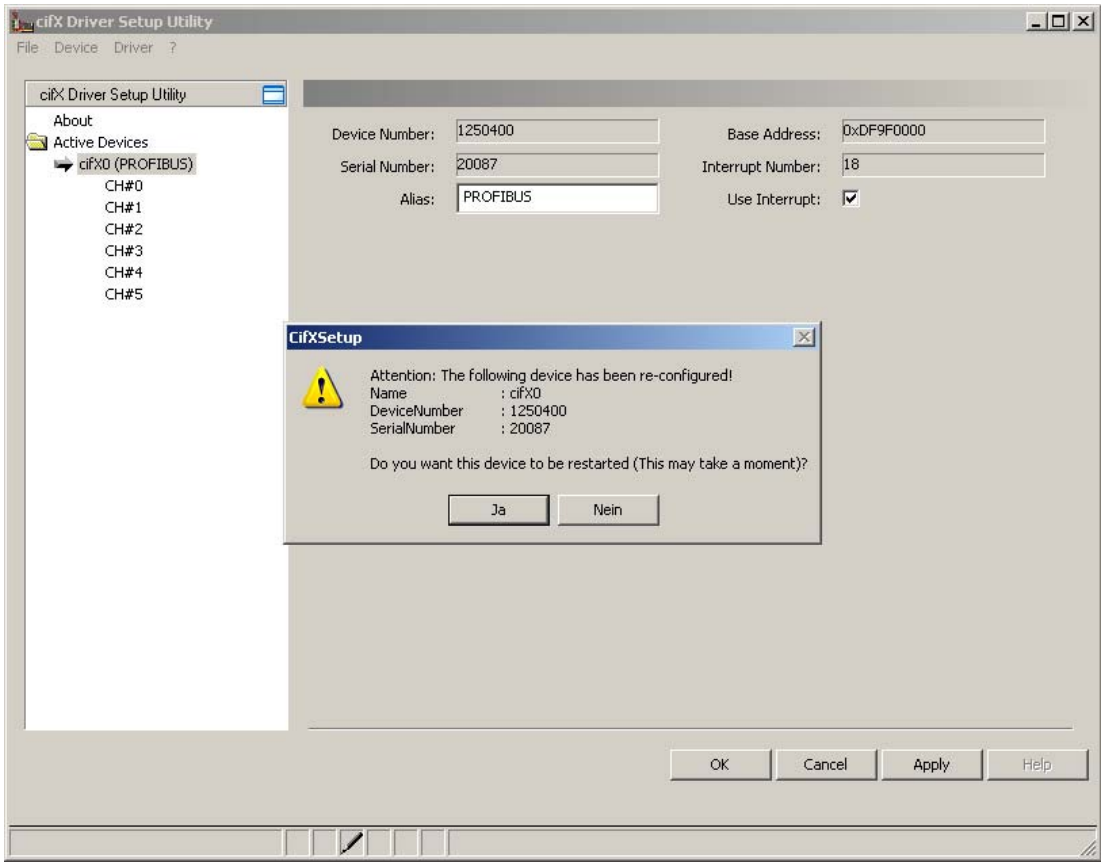

Figure 5: cifX Driver Setup Utility

2.1.3 Providing the Configuration for RTX2011

RTX 2011 offers direct access to the Windows® registry via a real-time application. This new feature is used by cifX RTX® 2011 driver to access the device configuration and no further initialization steps are necessary.

2.1.4 Providing the Device Configuration for RTX 8.1 / 2009

The Windows® registry is not accessible via the RTX application, thus the device configuration must be exported to a *'.reg'* file. For this purpose the export function of the Windows® registry editor must be used.

The location and the filename of this registry file must be supplied to the cifX RTX® driver via the driver parameters (see section *Structure RTX_CIFXDRV_PARAMETERS_T* on page 18).

Exporting the device configuration from the Windows® registry:

- Open the Windows® *Start* menu and select '*Run*'
- Execute the Windows® Registry Editor by entering the command 'regedit' and confirm via the OK button.
- In the Windows® Registry Editor, navigate to the registry key:
  HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CIFxDrv
- In the file menu select *Export…*.
  Make sure that the file type is *Registration Files (.reg)* or *Win9x/NT4 Registration files (.reg)*.
  Enter an arbitrary filename and select *Save*. Remember the file name and the file location!
- In your RTX application you have to pass the path to the exported registry file to the driver initialization routine *cifXInitDriver()* via the driver parameter *szRegFile* (see section *Structure RTX_CIFXDRV_PARAMETERS_T* on page 18).

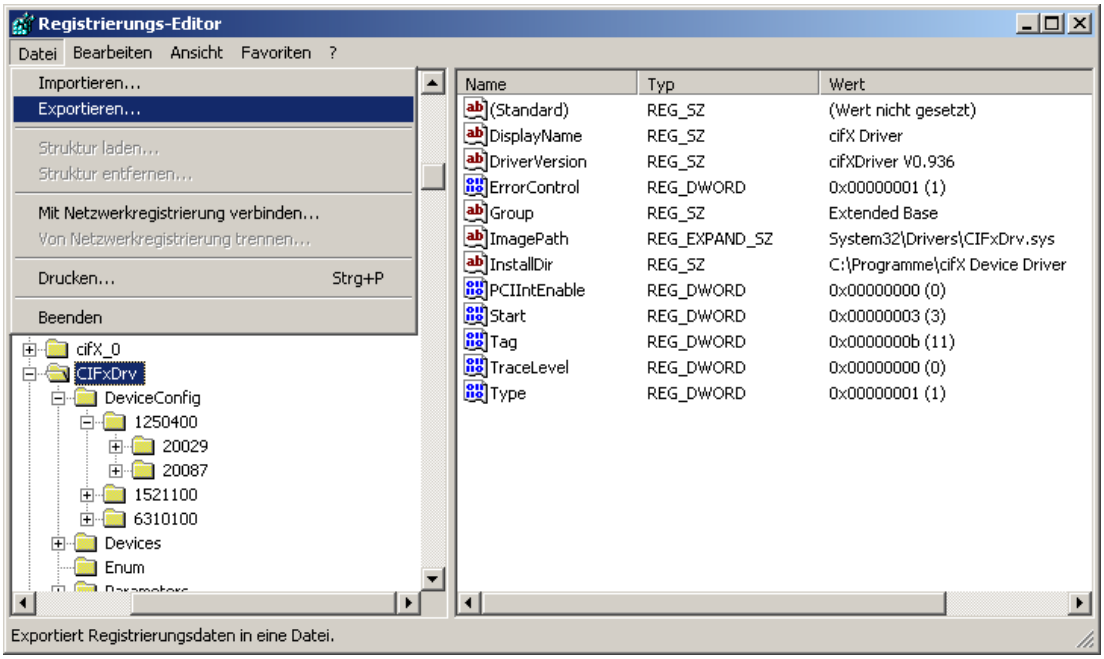

Figure 6: Export the Device Configuration with the Windows® Registry Editor

A sample configuration file is also located on the installation CD and can also be manually modify The relevant configuration keys are listed and described in the chapter

2.1.5 Description of the Device Configuration File

A sample configuration file is located on the installation CD. To modify the configuration file manually, the relevant configuration keys are listed below.

Key Value	Description
[HKLM\SYSTEM\CurrentControlSet\Services\CIFxDrv]	
InstallDir	Path to installation directory, created by the cifX device driver for Windows®
▪ **[HKLM\SYSTEM\CurrentControlSet\Services\CIFxDrv\DeviceConfig\<Device Nr.>\<Serial Nr.>]** ▪ **[HKLM\SYSTEM\CurrentControlSet\Services\CIFxDrv\DeviceConfig\Slot\<Slot Nr.>]**	
Alias	Statically assigned alias name
InterruptEnable	Device interrupt handling enable/disable 0 = disable (**default**) 1 = enabled
DMAEnable	Direct memory access enable/disable 0 = disable (**default**) 1 = enable
OSFile	To use loadable modules, a rcX base firmware is required. This entry specifies the path to the rcX base firmware file relative to: ▪ <InstallationDir>\<Device Number>_<Serial Number> ▪ <InstallationDir>\Slot_<Slot Nr.>
▪ **[HKLM\SYSTEM\CurrentControlSet\Services\CIFxDrv\DeviceConfig\<Device Nr.>\<Serial Nr.>\Channel<0..6>]** ▪ **[HKLM\SYSTEM\CurrentControlSet\Services\CIFxDrv\DeviceConfig\Slot\<Slot Nr.>\Channel<0..6>]** **Note:** Currently the firmware only supports channel 0	
ModuleCount	Number of configured loadable modules / monolithic firmware files
Module<0..ModuleCount>	Each loadable module or monolithic firmware file gets an own entry with an index as suffix. This specifies the path to the module file relative to: ▪ <InstallationDir>\<Dev. Number>_<Serial Number>\Channel<0..6> ▪ <InstallationDir>\Slot_<Slot Nr>\Channel<0..6>
ConfigCount	Number of configured fieldbus databases (configuration files)
Config<0..ModuleCount>	Each database gets an own entry with an index as suffix. This specifies the path to the database file relative to: ▪ <InstallationDir>\<Dev. Number>_<Serial Number>\Channel<0..6> ▪ <InstallationDir>\Slot_<Slot Nr>\Channel<0..6>
WarmstartFile	Full file name to warmstart parameter file residing in directory: ▪ <InstallationDir>\<Dev. Number>_<Serial Number>\Channel<0..6> ▪ <InstallationDir>\Slot_<Slot Nr>\Channel<0..6> **Note:** Warmstart file support depends on the used firmware and is maybe not supported
[HKLM\SYSTEM\CurrentControlSet\Services\CIFxDrv\Parameters\NX-PCA-PCI]	
DPM_8_Bit	NXPCA-PCI Timings parameter for 8 bit mode.
DPM_16_Bit	NXPCA-PCI Timings parameter for 16 bit mode.
DPM_32_Bit	NXPCA-PCI Timings parameter for 32 bit mode.

Table 6: Device Configuration File Keys

Sample configuration registry file:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CIFxDrv]
"InstallDir"="C:\\Programme\\CifX Device Driver"

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CIFxDrv\DeviceConfig]

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CIFxDrv\DeviceConfig\1250100]

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CIFxDrv\DeviceConfig\1250100\20217]
"Alias"=""
"InterruptEnable"=dword:00000001
"DMAEnable"=dword:00000001
"OSFile"="cifXrcX.nxf"

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CIFxDrv\DeviceConfig\1250100\20217\
Channel0]
"ModuleCount"=dword:00000001
"Module0"="cifXdps.nxf"
"ConfigCount"=dword:00000001
"Config0"="config.nxd"
"WarmstartFile"="warmstart.dat"

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CIFxDrv\DeviceConfig\1250100\20217\
Channel1]
"ModuleCount"=dword:00000000
"ConfigCount"=dword:00000000

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CIFxDrv\DeviceConfig\1250100\20217\
Channel2]
"ModuleCount"=dword:00000000
"ConfigCount"=dword:00000000

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CIFxDrv\DeviceConfig\1250100\20217\
Channel3]
"ModuleCount"=dword:00000000
"ConfigCount"=dword:00000000

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CIFxDrv\DeviceConfig\1250100\20217\
Channel4]
"ModuleCount"=dword:00000000
"ConfigCount"=dword:00000000

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CIFxDrv\DeviceConfig\1250100\20217\
Channel5]
"ModuleCount"=dword:00000000
"ConfigCount"=dword:00000000
```

3 Driver Specific Information

The cifX driver for RTX® needs some special initialization functions and structures.

3.1 Additional Structures

Some of the RTX® specific functions need parameters provided through structures. The structures and the meaning of the internal data are described in the following chapter.

3.1.1 Structure RTX_CIFXDRV_PARAMETERS_T

This structure is used to initialize the cifX driver.

Element	Data Type	Description
fScanPCI	int	Driver Initialization options: 0 = FALSE Driver does not scan for PCI cards. Cards can be added manually by *ulUserDevCount* and *ptUserDevList* parameters. 1 = TRUE Driver scans for all available PCI cards and adds them to the application.
ulTraceLevel	unsigned long	Set the trace level of the driver: 1 = TRACE_LEVEL_DEBUG 2 = TRACE_LEVEL_INFO 4 = TRACE_LEVEL_WARNING 8 = TRACE_LEVEL_ERROR
ulPollInterval	unsigned long	Polling interval in milliseconds [ms] for non-interrupt driven cards (used for Change of State (COS) signaling, see reference [3]) 0 = 500ms default
szRegFile	const char*	**RTX 2011:** not used **RTX 8.1/2009:** Set the path to the Windows® registry file. Can be NULL to use the default of *'c:\CIFxDrv.reg'*. Creation of this registry file is explained in section *Device Configuration* on page 12. NULL = 'c:\CIFxDrv.reg' default
ulUserDevCount	unsigned long	Number of user cards entries in the *ptUserDevList* to add to the driver (e.g. if a card is connected via DPM). 0 = none
ptUserDevList	RTX_CIFXDRV_DEVICEENTRY_T **	Array of user added cards. Number of entries is defined by *ulUserDevCount*. See section *Structure RTX_CIFXDRV_DEVICEENTRY_T* on page 19.

Table 7: Structure RTX_CIFXDRV_PARAMETERS_T Definition

3.1.2 Structure RTX_CIFXDRV_DEVICEENTRY_T

This structure describes a cifX device which should be added to the driver. This structure can be acquired through *cifXFindDevice()* or filled by the user if a custom card should be added.

Element	Data Type	Description
ulPhysicalAddress	unsigned long	Physical address of the DPM (this value is used to detect the PCI card linked to the DPM)
bIrqNumber	unsigned char	Interrupt number
pvDPMAddress	void*	Virtual pointer to card DPM
ulDPMSize	unsigned long	Size of the DPM in bytes
fPCICard	int	0 = FALSE Device is connected via DPM. 1 = TRUE Device is connected to PCI bus
tBusInfo	RTX_CIFXDRV_DEVICEBUSINFO_T	Bus information, see section *Structure RTX_CIFXDRV_DEVICEBUSINFO_T* on page 19

Table 8: Structure RTX_CIFXDRV_DEVICEENTRY_T Definition

3.1.3 Structure RTX_CIFXDRV_DEVICEBUSINFO_T

Bus information structure used to store bus specific information for cifX devices connected via PCI bus.

Element	Data Type	Description
ulBusNumber	int	Bus number
ulSlotNumber	int	Slot number

Table 9 : Structure RTX_CIFXDRV_DEVICEBUSINFO_T Definition

3.2 Additional functions

This chapter describes functions which are only available for the RTX® version of the driver. These functions are used to initialize the cifX device driver.

3.2.1 cifXInitDriver()

This function must be called before accessing any driver function. The cifX driver initialization includes discovering all available cifX PCI devices and downloading the firmware and configuration files.

Note: *cifXInitDriver()* calls *cifXFindDevice()* internally to search for available cifX PCI cards (*fScanPCI* = TRUE). If the cards should be defined manually, *cifXFindDevice()* can be used to add cards manually (*fScanPCI* = FALSE).

Note: For none PCI cards, the application has to create a RTX_CIFXDRV_DEVICEENTRY_T with the corresponding device information. Afterwards, the structure must be placed into the RTX_CIFXDRV_PARAMETERS_T structure (see parameters *ulUserDevCount, ptUserDevList, fScanPCI = FALSE)* before *cifXInitDriver()* is called.

Function call:

```
int32_t cifXInitDriver (RTX_CIFXDRV_PARAMETERS_T*  ptDriverParams);
```

Arguments:

Argument	Data Type	Description
ptDevEntry	RTX_CIFXDRV_PARAMETERS_T*	Driver parameters, see section *Structure RTX_CIFXDRV_PARAMETERS_T* on page 18

Return Values:

Return Values	
CIFX_NO_ERROR	Driver initialization successful
CIFX_DRV_INIT_ERROR	Driver initialization failed (no cifX device available)

Example:

```
RTX_CIFXDRV_VTABLE            tVTable          = {0};
RTX_CIFXDRV_PARAMETERS_T      tDriverParams    = {0};
HANDLE                        hDll             = NULL;
char*                         szRegFile         = "C:\\CIFxDrv.reg";

/* Set driver parameters */
tDriverParams.fScanPCI       = TRUE;
tDriverParams.ulUserDevCount = 0;
tDriverParams.szRegFile      = szRegFile;

/* Load cifXRTXDrv.rtdll */
hDll = LoadLibrary("cifxRTXDrv.dll");

/* Assign function pointer */
tVTable.cifXInitDriver = (RTXDRV_INIT) GetProcAddress( hDll, "cifXInitDriver");

/* Scan for all available cifX PCI devices and initialize the cifX device driver */
tVTable.cifXInitDriver (&tDriverParams);

FreeLibrary(hDll);
```

3.2.2 cifXDeinitDriver()

Un-initialize the driver and remove all devices from the control of the cifX driver library. After calling this function the application must not access any cifX driver API function any more.

Function call:

```
int32_t cifXDeinitDriver ( void );
```

Arguments:

Argument	Data Type	Description
none		

Return Values:

Return Values	
CIFX_NO_ERROR	Driver deinitialization successful
CIFX_DEV_HW_PORT_IS_USED	Device is in use, so driver deinitialization is denied
CIFX_DRV_NOT_INITIALIZED	Driver was not initialized

3.2.3 cifXFindDevice()

This function scans for a cifX PCI device in the system and builds a RTX_CIFXDRV_DEVICEENTRY_T structure for each discovered device.

The function is used internally by *cifXInitDriver()* if PCI cards should be detected automatically (RTX_CIFXDRV_PARAMETERS_T, *fScanPCI* parameter = TRUE).

cifXFindDevice() can also be used by an application to manually define a cifX card. Therefore the function can be called with a fixed device number (*iNum*). If the given device is available, the function returns the RTX_CIFXDRV_DEVICEENTRY_T structure for the device.

Afterwards the application has to insert this structure into the RTX_CIFXDRV_PARAMETERS_T structure (see section Structure RTX_CIFXDRV_PARAMETERS_T on page 18), setting the *fScanPCI* flag to FALSE and calling *cifXInitDriver()*.

Function call:

```
BOOL cifXFindDevice ( RTX_CIFXDRV_DEVICEENTRY_T*    ptDevEntry,
                      int                           iNum);
```

Arguments:

Argument	Data Type	Description
ptDevEntry	RTX_CIFXDRV_DEVICEENTRY_T*	Pointer to a RTX_CIFXDRV_DEVICEENTRY_T structure, to place returned values in
iNum	int	Number of the device in the system 0 = first device

Return Values:

Return Values	
TRUE	A device with number *iNum* was found
FALSE	A device with number *iNum* could not be found

Example:

```
RTX_CIFXDRV_VTABLE           tVTable          = {0};
RTX_CIFXDRV_PARAMETERS_T     tDriverParams    = {0};
RTX_CIFXDRV_DEVICEENTRY_T    tDevEntry        = {0};
HANDLE                       hDll             = NULL;

/* Load cifXRTXDrv.rtdll */
hDll = LoadLibrary("cifxRTXDrv.dll");

/* Assign function pointer */
tVTable.cifXInitDriver = (RTXDRV_INIT) GetProcAddress( hDll, "cifXInitDriver");
tVTable.cifXFindDevice = (RTXDRV_FIND_DEVICE)
                         GetProcAddress( hDll, "cifXFindDevice");

/* Find the first cifX PCI device */
if (tVTable.cifXFindDevice (&tDevEntry, 0))
{
  tDriverParams.fScanPCI       = FALSE;
  tDriverParams.ulUserDevCount = 1;
  tDriverParams.ptUserDevList  = &tDevEntry;

  /* initialize the cifX device driver */
  tVTable.cifXInitDriver (&tDriverParams);
}

FreeLibrary(hDll);
```

3.2.4 cifXGetDriverVersion()

This function requests the version of the cifX driver for RTX®.

Function call:

```
int32_t cifXGetDriverVersion ( char* szVersion, uint32_t ulSize )
```

Arguments:

Argument	Data Type	Description
szVersion	char*	String buffer to return the version of the cifX driver
ulSize	uint32_t	Maximum buffer size for the version string

Return Values:

Return Values	
CIFX_NO_ERROR	Driver version successfully copied to string buffer
CIFX_INVALID_BUFFERSIZE	Supplied buffer to small to hold the driver version string

Example:

```
RTX_CIFXDRV_VTABLE   tVTable              = {0};
char                 szDrvVersion[20]     = "";
HANDLE               hDll                 = NULL;

/* Load cifXRTXDrv.rtdll */
hDll = LoadLibrary("cifxRTXDrv.dll");

/* Assign function pointer */
tVTable.cifXGetDriverVersion = (RTXDRV_GET_VERSION)
                               GetProcAddress( hDll, "cifXGetDriverVersion");

/* Get driver version */
tVTable.cifXGetDriverVersion( szDrvVersion,
                              sizeof(szDrvVersion)/sizeof(*szDrvVersion) );

/* Print driver version to screen */
RtPrintf("%s", szDrvVersion);

FreeLibrary(hDll);
```

3.3 Driver Startup Procedure

The driver startup procedure can be controlled by the user. The following two use cases are available:

- Automatically add all found cifX PCI devices and optionally add user specific devices
- Skip cifX PCI device scan and add user specific device manually

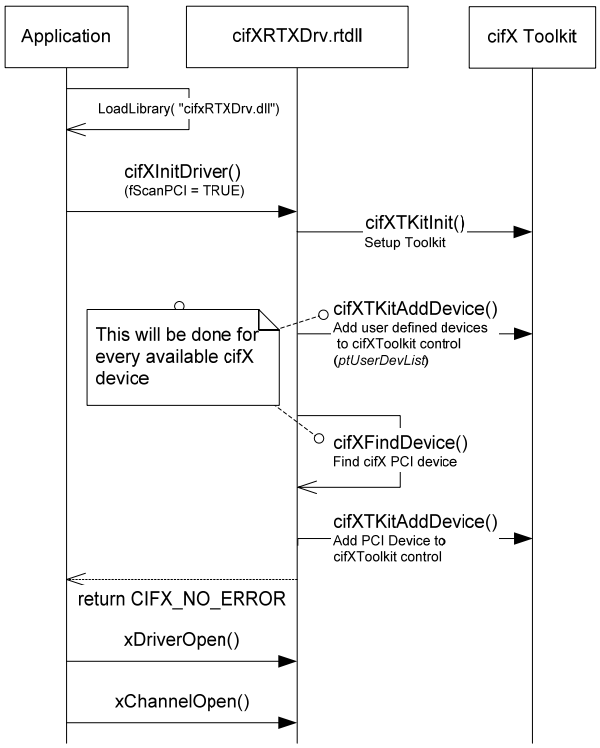

Figure 7: Initialization of the cifX Driver with fScanPCI = TRUE

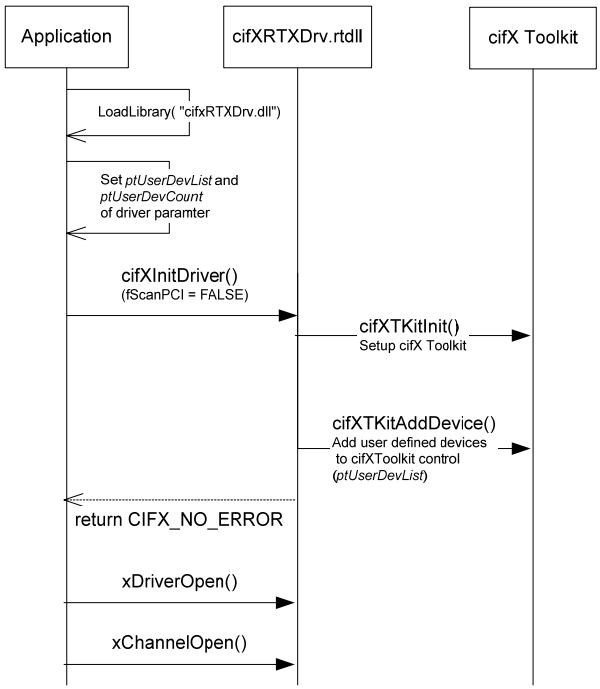

Figure 8: Initialization of the cifX Driver with fScanPCI = FALSE

4 Programming with the Driver

The Application Programming Interface (API) of the Hilscher RTX® driver is based on the already known cifX device driver Interface (reference [1]). Therefore the '*cifX Device Driver*' manual can be used. This manual describes the driver functions (cifX API), error codes and shows some program examples. The installation CD also includes a *'Test Application'* directory with a RTX® specific example.

The following C application demonstrates the minimum functions which must be called to enable an application to work with a cifX device

```
#include <windows.h>
#include <rtapi.h>
#include "cifXUser.h"
#include "cifXErrors.h"
#include "cifxRTXDrv.h"

/*****************************************************************************/
/*! The main function
*   \return 0 on success                                                     */
/*****************************************************************************/
int main(int argc, char* argv[])
{
  HANDLE                        hDll           = NULL;
  CIFXHANDLE                    hDriver        = NULL;
  RTX_CIFXDRV_VTABLE            tVTable        = {0};
  RTX_CIFXDRV_PARAMETERS_T      tDriverParams  = {0};
  long                          lRet           = CIFX_NO_ERROR;
  char*                         szRegFile      = "C:\\CIFxDrv.reg";

  /* Set driver parameters */
  tDriverParams.fScanPCI       = TRUE;
  tDriverParams.ulUserDevCount = 0;
  tDriverParams.szRegFile      = szRegFile;
  tDriverParams.ulTraceLevel   = TRACE_LEVEL_ERROR   | TRACE_LEVEL_WARNING |
                                 TRACE_LEVEL_INFO    | TRACE_LEVEL_DEBUG;
  /* Load cifXRTXDrv.rtdll */
  hDll = LoadLibrary("cifxRTXDrv.dll");

  /* Assign function pointer */
  tVTable.cifXInitDriver  =(RTXDRV_INIT)   GetProcAddress(hDll, "cifXInitDriver");
  tVTable.cifXDeinitDriver=(RTXDRV_DEINIT) GetProcAddress(hDll, "cifXDeinitDriver");
  tVTable.xDriverOpen     =(X_DRIVER_OPEN) GetProcAddress(hDll, "xDriverOpen");
  tVTable.xDriverClose    =(X_DRIVER_CLOSE)GetProcAddress(hDll, "xDriverClose");
  /* TODO: assign further api function */

  /* Initialize and open the cifX device driver */
  if( (CIFX_NO_ERROR == (lRet = tVTable.cifXInitDriver (&tDriverParams))) &&
      (CIFX_NO_ERROR == (lRet = tVTable.xDriverOpen(&hDriver))) )
  {
    /* Work with the cifX API */

    /* Close the cifX driver */
    tVTable.xDriverClose(hDriver);
  } else
  {
    RtPrintf("Error opening driver. lRet=0x%08X\r\n", lRet);
  }

  /* Deinit the cifX device driver */
  tVTable.cifXDeinitDriver ();

  FreeLibrary(hDll);

  return 0;
}
```

5 Frequently Asked Questions

Q: The driver presents the following warning at startup: '*OS_Memalloc: Not enough free memory in RTSS memory pool! Request from windows memory pool*'. What does that mean?

A: To fulfill RTSS applications memory requests, RTX deterministically allocates memory from a special memory pool (Local memory pool). RTSS applications that allocate memory from that pool don't have to initiate a Service Request Interrupt to request memory from Windows (This request would be non-deterministic). If the memory needs of the RTSS application exceeds the initial size of the local memory pool, RTX initiate a non-deterministic Service Request Interrupt to request memory from Windows. This action is signaled to the user via the drivers warning message. This scenario can be avoided, by increasing the initial local memory pool size to the memory needs of the driver and the RTSS application (see memory settings in the RTX properties application). If the creation of the local memory pool is not done at startup, but with the first call to a local memory pool allocation function the drivers warning message is also signaled. The creation of the local memory pool at startup is initiated, a) by executing the RTSS application with the parameter */local* or b) by modifying the memory settings in the RTX properties application (see RTX Manual).

Q: Is it possible to develop a cifX application with Visual Studio 6.0?

A: If RTX 8.1 is used, application development with Visual Studio 6.0 is supported. But with release of RTX 2009 the development of RTSS application with Visual Studio 6.0 is no longer supported (see http://intervalzero.com/assets/OSCompatMatrixRTX.pdf). With release of RTX 2011 the development of RTSS application with Visual Studio 2003 is also no longer supported

Q: Is it possible to configure a cifX card, independently of its serial and device number?

A: The cifX Device Driver for RTX from version 1.0.1.0 on identifies cifX cards alternatively via its slot number. Please note that your cifX device must provide a rotary switch to use slot number support.

Q: The compiled test application crashes the system. What's wrong?

A: If you don't use Visual Studio 2005 to compile the test application, you probably forgot to adjust the linker path of the Visual Studio specific RTX libraries.

6 Appendix

6.1 List of Tables

6.2 List of Figures

6.3 Contact

Headquarters

Germany
Hilscher Gesellschaft für
Systemautomation mbH
Rheinstrasse 15
65795 Hattersheim
Phone: +49 (0) 6190 9907-0
Fax: +49 (0) 6190 9907-50
E-Mail: info@hilscher.com

Support
Phone: +49 (0) 6190 9907-99
E-Mail: de.support@hilscher.com

Subsidiaries

China
Hilscher Systemautomation (Shanghai) Co. Ltd.
200010 Shanghai
Phone: +86 (0) 21-6355-5161
E-Mail: info@hilscher.cn

Support
Phone: +86 (0) 21-6355-5161
E-Mail: cn.support@hilscher.com

France
Hilscher France S.a.r.l.
69500 Bron
Phone: +33 (0) 4 72 37 98 40
E-Mail: info@hilscher.fr

Support
Phone: +33 (0) 4 72 37 98 40
E-Mail: fr.support@hilscher.com

India
Hilscher India Pvt. Ltd.
New Delhi - 110 025
Phone: +91 11 40515640
E-Mail: info@hilscher.in

Italy
Hilscher Italia srl
20090 Vimodrone (MI)
Phone: +39 02 25007068
E-Mail: info@hilscher.it

Support
Phone: +39 02 25007068
E-Mail: it.support@hilscher.com

Japan
Hilscher Japan KK
Tokyo, 160-0022
Phone: +81 (0) 3-5362-0521
E-Mail: info@hilscher.jp

Support
Phone: +81 (0) 3-5362-0521
E-Mail: jp.support@hilscher.com

Korea
Hilscher Korea Inc.
Suwon, 443-734
Phone: +82 (0) 31-695-5515
E-Mail: info@hilscher.kr

Switzerland
Hilscher Swiss GmbH
4500 Solothurn
Phone: +41 (0) 32 623 6633
E-Mail: info@hilscher.ch

Support
Phone: +49 (0) 6190 9907-99
E-Mail: ch.support@hilscher.com

USA
Hilscher North America, Inc.
Lisle, IL 60532
Phone: +1 630-505-5301
E-Mail: info@hilscher.us

Support
Phone: +1 630-505-5301
E-Mail: us.support@hilscher.com