



**Application Note**

## **POWERLINK Controlled Node Programming Example**

**Configuration and Communication**

**openPOWERLINK Managing Node to Hilscher Controlled Node**

**Hilscher Gesellschaft für Systemautomation mbH**

**[www.hilscher.com](http://www.hilscher.com)**

DOC101105AN02EN | Revision 2 | English | 2013-07 | Released | Public

## Table of Contents

<b>1</b>	<b>Introduction.....</b>	<b>3</b>
1.1	About this Document.....	3
1.2	List of Revisions .....	3
<b>2</b>	<b>Features of the Programming Example .....</b>	<b>4</b>
<b>3</b>	<b>Set up the POWERLINK Network.....</b>	<b>5</b>
3.1	Set up the Hardware .....	5
3.2	Configuring the Hilscher POWERLINK Controlled Node .....	6
3.3	Set up the openPOWERLINK .....	9
3.4	Testing.....	10
3.4.1	General Test.....	10
3.4.2	Test by Using the Demo Object.....	10
3.4.3	Testing the Process Data .....	12
<b>4</b>	<b>Appendix .....</b>	<b>13</b>
4.1	List of Tables .....	13
4.2	List of Figures.....	13
4.3	Contacts .....	14

# 1 Introduction

## 1.1 About this Document

This manual describes how to run the basic sample application for Hilscher POWERLINK Controlled Node firmware in combination with openPOWERLINK as Ethernet POWERLINK Managing Node.

The components used are:

- 1 PC with 1 standard Ethernet Card and 1 PC card cifX
- Hilscher POWERLINK Controlled Node firmware version 2.1.22.0
- openPOWERLINK version 1.7
- openCONFIGURATOR version 1.2.0
- Microsoft Visual Studio 2005

## 1.2 List of Revisions

Rev	Date	Name	Chapter	Revision
1	2010-11-04	EO	all	Created
2	2013-07-17	HH	all	Revised

Table 1: List of Revisions

## 2 Features of the Programming Example

The POWERLINK Controlled Node basic sample application is a straight forward implementation which shows how to use cifX Device Driver API and how to communicate using 'packets'.

The application program configures a controlled node with the following features:

- 4 byte PDO input
- 4 byte PDO output
- 1 SDO simple variable object – non indexed access (0x6000)

The application program will simply mirror the received data. This means, that the application will copy the received data and send it back to the managing node.

Furthermore, the application program registers to be able to receive 'write indications' of the demo object 0x6000. Each time the object is written from the managing node, this will be indicated within an output in the console program.

## 3 Set up the POWERLINK Network

### 3.1 Set up the Hardware

Use following environment:

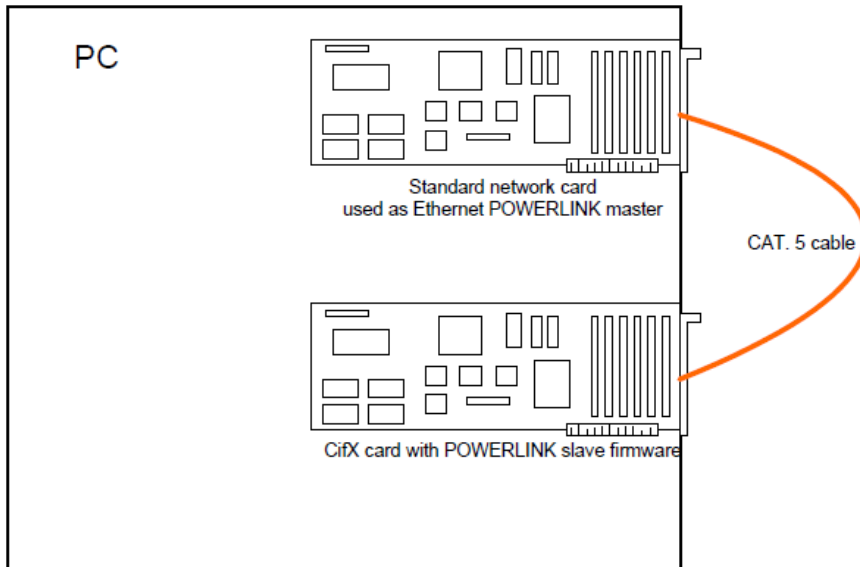


Figure 1: Hardware Setup

## 3.2 Configuring the Hilscher POWERLINK Controlled Node

- Load POWERLINK Controlled Node firmware file via the cifX Driver Setup Utility into PC card cifX.

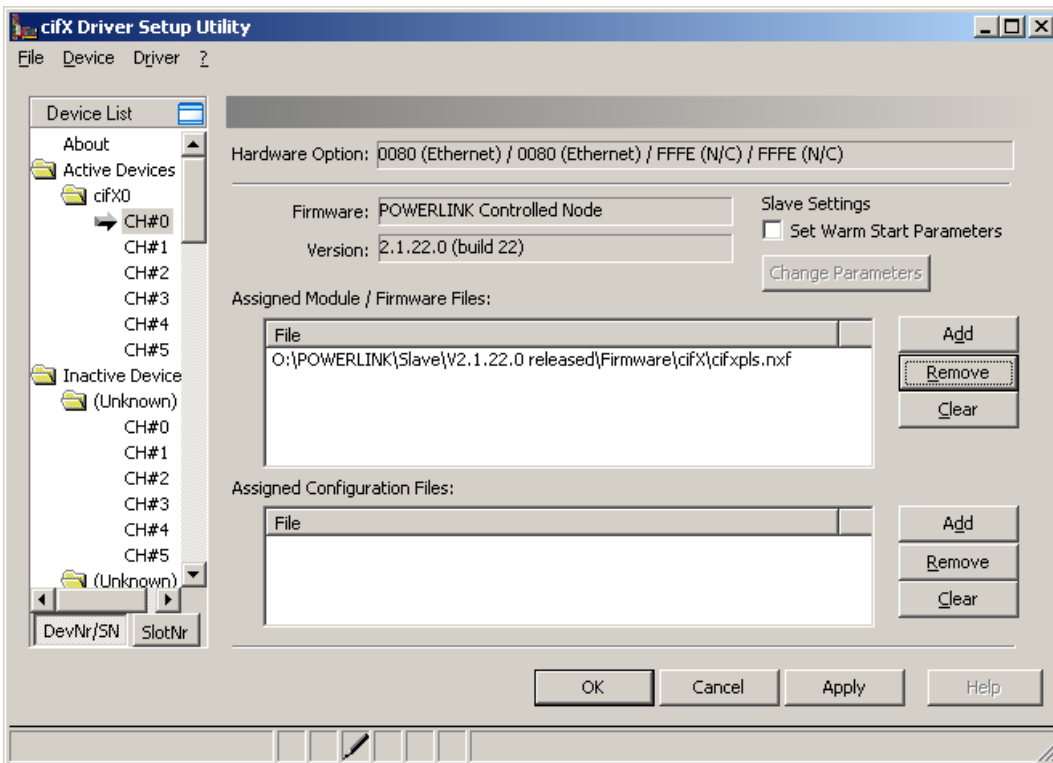


Figure 2: cifX Setup to load the POWERLINK Controlled Node Firmware

- Open the POWERLINK basic sample application (CifX\_ApDemo.sln) using Microsoft Visual Studio 2005 or higher.
- Open the file CifX\_ApDemo.cpp and adjust the board name at the beginning of the main function. Set the board instance for the PC card cifX to that one which you have loaded the firmware to.

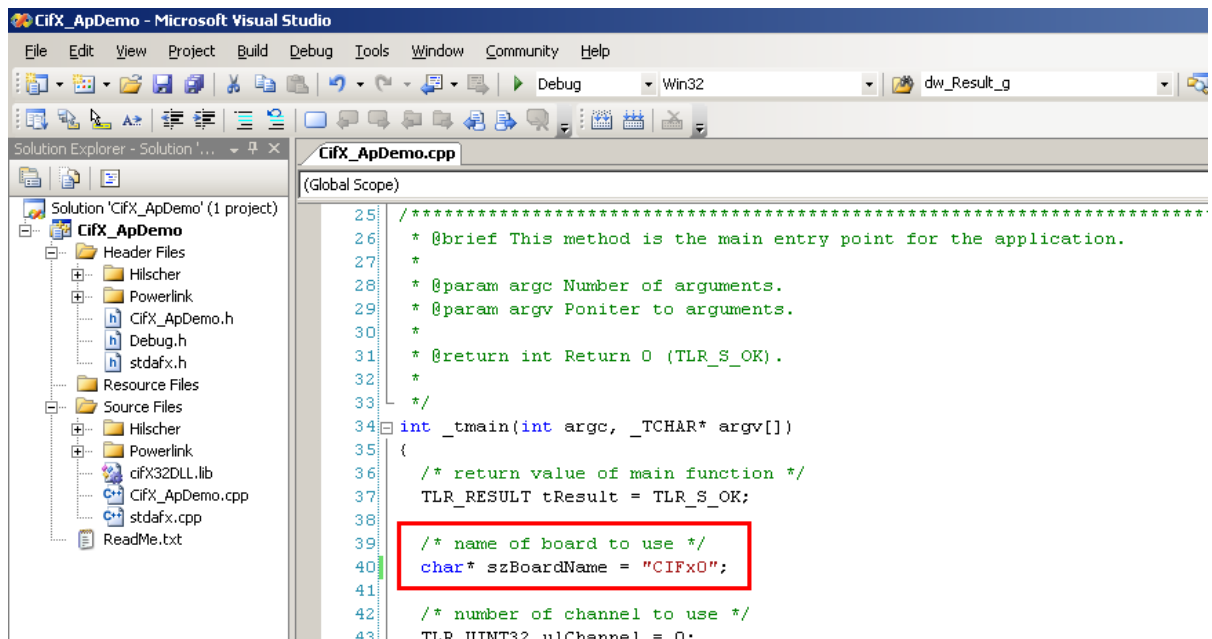


Figure 3: Adjust the cifX Board Instance

- Build and start the Basic Sample Application.

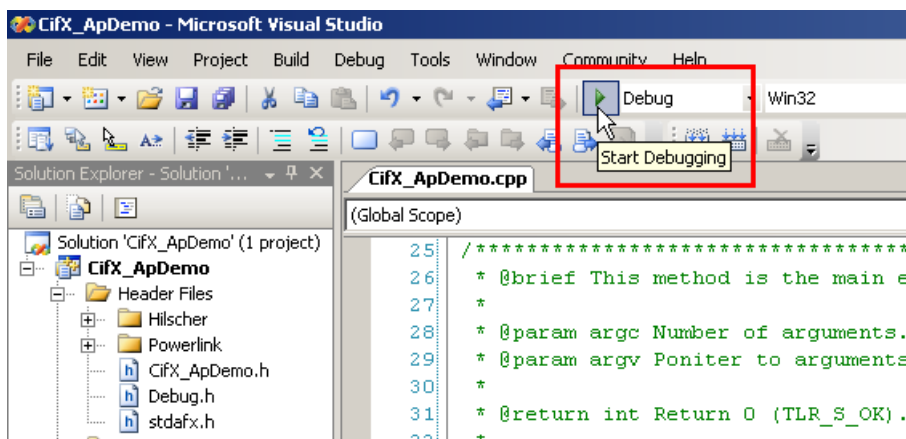


Figure 4: Starting the Example

- You can monitor the system startup by checking the console output.

```

d:\Workspace_POWERLINK\PowerLink-ControlledNode_latest\Examples\CifX\CifX_ApDe...
*****
*
*   Basic sample application for Ethernet POWERLINK.
*
*   Copyright (c) Hilscher GmbH. All Rights Reserved.
*
*****

Opening driver...
Opening channel 0 on board CIfx0...
Processing system restart...
Sending configuration request...
Processing channel init...

```

Figure 5: Startup

- After the startup, the application runs into an endless loop. Within this loop, packets (acyclic data) as well as process data (cyclic) are handled. Every time a packet is handled, the application prints a "-". When the handling of process data fails, e.g. if there is no managing node communication, an "F" will be printed. When process data exchange was successful, an "X" will be printed.

```

d:\Workspace_POWERLINK\PowerLink-ControlledNode_latest\Examples\CifX\CifX_ApDe...
Setting bus state on...
Entering endless loop...

>> Press any key to leave. <<

FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

```

Figure 6: Entering Cyclic I/O Exchange

- As soon as the managing node writes the Demo Object (0x6000), the new value will also be shown in the console output.



### 3.3 Set up the openPOWERLINK

SYSTEC electronic provides an open source industrial Ethernet solution called openPOWERLINK. We use openPOWERLINK as Managing Node.

openPOWERLINK needs a network configuration, this can be created using openCONFIGURATOR. openCONFIGURATOR is also an open source tool for configuring and maintaining POWERLINK networks. Because an appropriate configuration (and also openCONFIGURATOR project) is delivered with the sample application, this document does not describe how to create such one. For information concerning this, please refer chapter *How To* within the help of openCONFIGURATOR.

You can download the programs from following sources:

- <http://sourceforge.net/projects/openpowerlink/>
- <http://sourceforge.net/projects/openconf/>

openPOWERLINK requires a WinPcap installation:

- <http://www.winpcap.org/>

openCONFIGURATOR needs additionally Tcl85:

- <http://www.activestate.com/activetcl>

For our demonstration we use one of the examples from the openPOWERLINK distribution. Open following project:

<openPOWERLINKInstallDir>\Examples\X86\Windows\VC8\demo\_cfm\_pcap\demo\_cfm\_pcap.sln

This project contains a file called demo\_main.c. This file must be removed against the file which comes together with the Hilscher cifX basic sample application. The new file has some special features:

- Continuously sending process data to the controlled node (incrementing counter)
- Reading and writing demo object via key pressing (acyclic operation)

Furthermore you have to copy the configuration, generated by openCONFIGURATOR, to the project folder.

Therefore copy the content of:

<ExampleInstallDir>\openCONFIGURATOR\Project\cdc\_ap\

To:

<openPOWERLINKInstallDir>\Examples\X86\Windows\VC8\demo\_cfm\_pcap\

Overwrite already existing items.

## 3.4 Testing

### 3.4.1 General Test

- Start the POWERLINK basic sample application as described in chapter *Features of the Programming Example* on page 4.
- Run the openPOWERLINK example
- Select the interface where your controlled node is connected to:

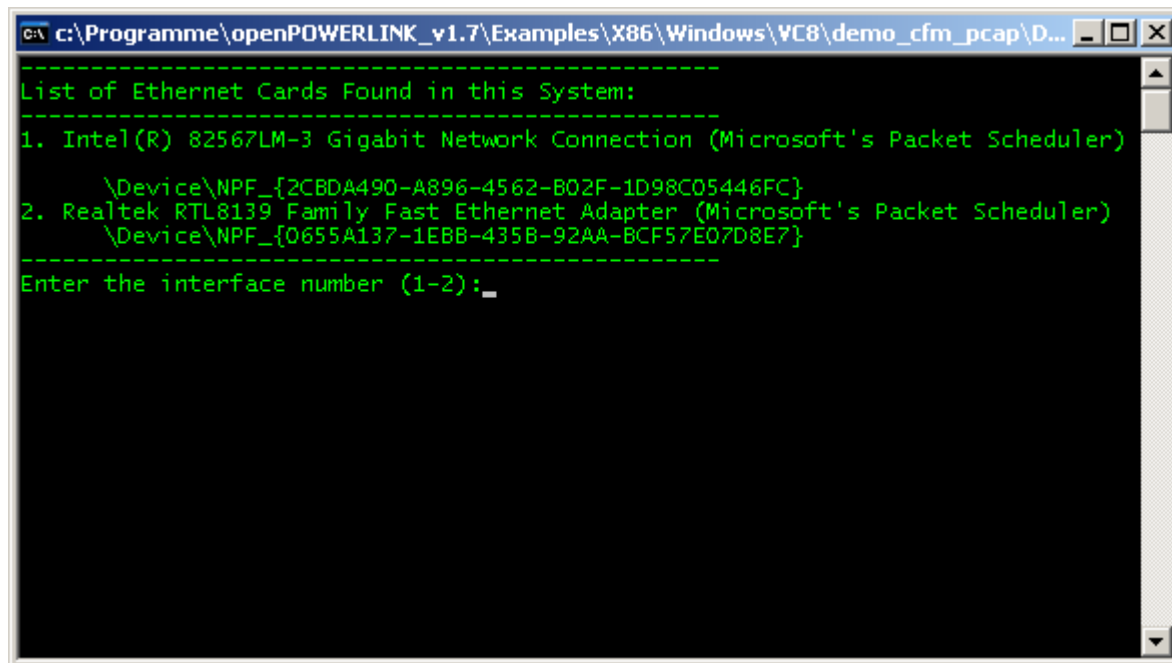


Figure 7: Interface Selection

- After a successful start-up of both applications, the POWERLINK controlled node will start with cyclic data exchange. This will be indicated with an "X" within the console.

### 3.4.2 Test by Using the Demo Object

As described above, there are some special features which allow you reading and writing of the demo object via openPOWERLINK managing node. These features can be accessed via a single key press. Following table shows which keys and features are available:

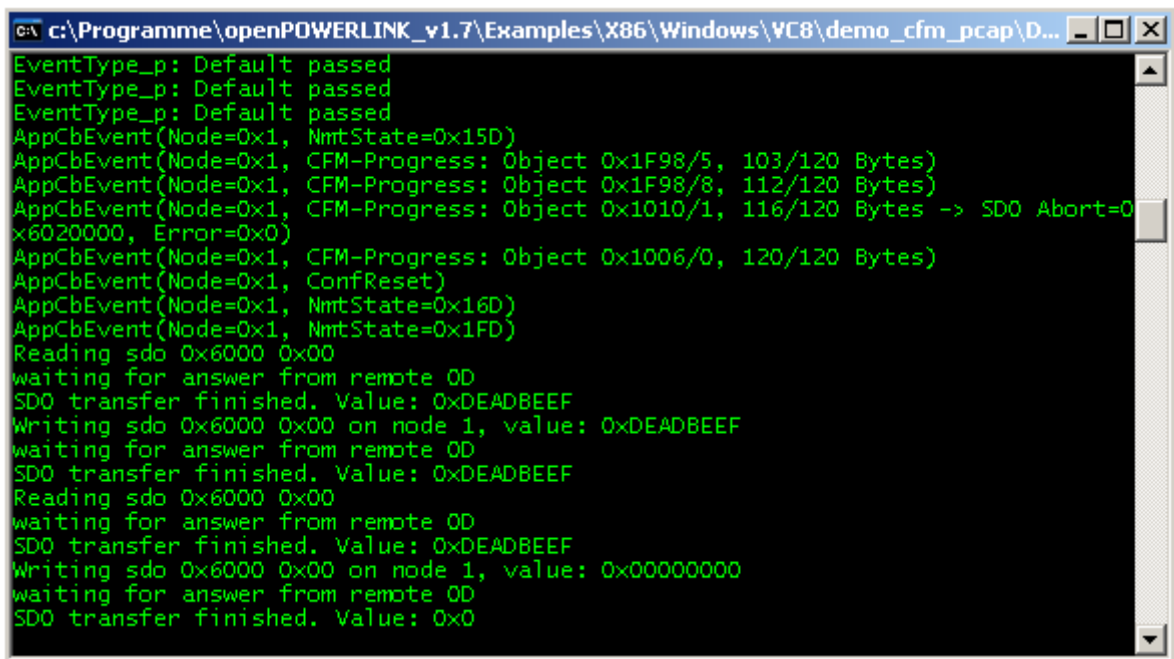
Key	Feature
1	Read Object 0x6000 Subobject 0x00 from Node 1
2	Not used.
3	Write Object 0x6000 sub object 0x00 of Node 1 – New value: 0xDEADBEEF
4	Write Object 0x6000 sub object 0x00 of Node 1 – New value: 0x00000000

Figure 8: Key Usage

All actions will be indicted within the console.

The screens below show the following test sequence:

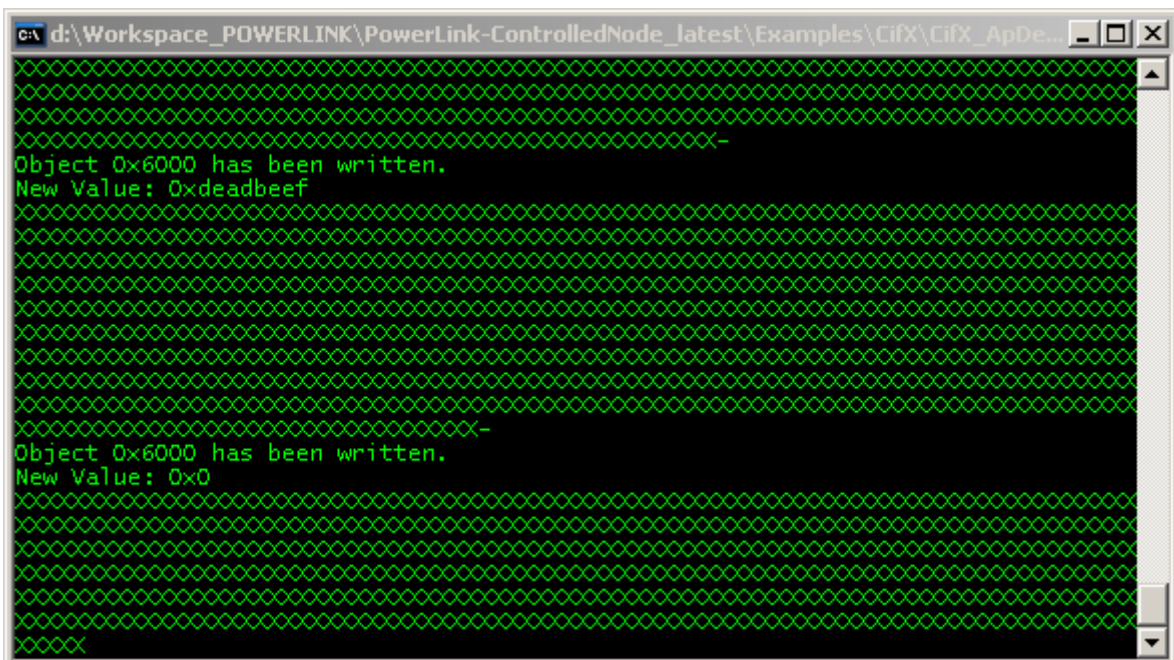
- Read object 0x6000 sub object 0x00 from Node 1
- Write object 0x6000 sub object 0x00 of Node 1 – New value: 0xDEADBEEF
- Read object 0x6000 sub object 0x00 from Node 1
- Write object 0x6000 sub object 0x00 of Node 1 – New value: 0x00000000



```

c:\Programme\openPOWERLINK_v1.7\Examples\X86\Windows\VC8\demo_cfm_pcap\D...
EventType_p: Default passed
EventType_p: Default passed
EventType_p: Default passed
AppCbEvent(Node=0x1, NmtState=0x15D)
AppCbEvent(Node=0x1, CFM-Progress: Object 0x1F98/5, 103/120 Bytes)
AppCbEvent(Node=0x1, CFM-Progress: Object 0x1F98/8, 112/120 Bytes)
AppCbEvent(Node=0x1, CFM-Progress: Object 0x1010/1, 116/120 Bytes -> SDO Abort=0
x6020000, Error=0x0)
AppCbEvent(Node=0x1, CFM-Progress: Object 0x1006/0, 120/120 Bytes)
AppCbEvent(Node=0x1, ConfReset)
AppCbEvent(Node=0x1, NmtState=0x16D)
AppCbEvent(Node=0x1, NmtState=0x1FD)
Reading sdo 0x6000 0x00
waiting for answer from remote OD
SDO transfer finished. Value: 0xDEADBEEF
Writing sdo 0x6000 0x00 on node 1, value: 0xDEADBEEF
waiting for answer from remote OD
SDO transfer finished. Value: 0xDEADBEEF
Reading sdo 0x6000 0x00
waiting for answer from remote OD
SDO transfer finished. Value: 0xDEADBEEF
Writing sdo 0x6000 0x00 on node 1, value: 0x00000000
waiting for answer from remote OD
SDO transfer finished. Value: 0x0
  
```

Figure 9: Test Sequence – Managing Node Output



```

d:\Workspace_POWERLINK\PowerLink-ControlledNode_latest\Examples\CifX\CifX_ApDe...
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx-
Object 0x6000 has been written.
New Value: 0xdeadbeef
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx-
Object 0x6000 has been written.
New Value: 0x0
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx-
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
  
```

Figure 10: Test Sequence – Controlled Node Output

### 3.4.3 Testing the Process Data

- At first check that the controlled node prompts an “X” within the output of the console program.
- Afterwards set a breakpoint into the cyclic data exchange method of the managing node:  
tEplKernel PUBLIC AppCbSync(void)

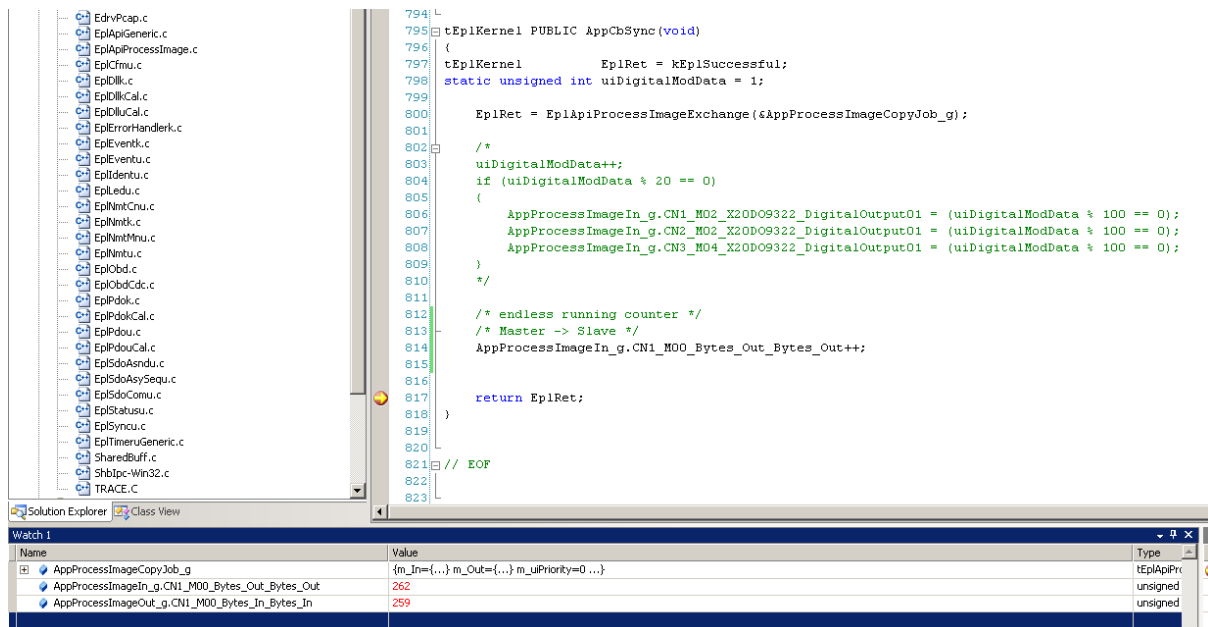


Figure 11: Test of Cyclic Data Using Microsoft Visual Studio 2005 Debugger

- Compare the values of the input and output image – the slave has to mirror the data.

## 4 Appendix

### 4.1 List of Tables

Table 1: List of Revisions .....	3
----------------------------------	---

### 4.2 List of Figures

Figure 1: Hardware Setup .....	5
Figure 2: cifX Setup to load the POWERLINK Controlled Node Firmware .....	6
Figure 3: Adjust the cifX Board Instance .....	7
Figure 4: Starting the Example .....	7
Figure 5: Startup .....	8
Figure 6: Entering Cyclic I/O Exchange .....	8
Figure 7: Interface Selection .....	10
Figure 8: Key Usage .....	10
Figure 9: Test Sequence – Managing Node Output .....	11
Figure 10: Test Sequence – Controlled Node Output .....	11
Figure 11: Test of Cyclic Data Using Microsoft Visual Studio 2005 Debugger .....	12

## 4.3 Contacts

### 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](mailto:info@hilscher.com)

#### Support

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

### Subsidiaries

#### China

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

#### Support

Phone: +86 (0) 21-6355-5161  
E-Mail: [cn.support@hilscher.com](mailto: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](mailto:info@hilscher.fr)

#### Support

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

#### India

Hilscher India Pvt. Ltd.  
New Delhi - 110 065  
Phone: +91 11 26915430  
E-Mail: [info@hilscher.in](mailto:info@hilscher.in)

#### Italy

Hilscher Italia S.r.l.  
20090 Vimodrone (MI)  
Phone: +39 02 25007068  
E-Mail: [info@hilscher.it](mailto:info@hilscher.it)

#### Support

Phone: +39 02 25007068  
E-Mail: [it.support@hilscher.com](mailto:it.support@hilscher.com)

#### Japan

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

#### Support

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

#### Korea

Hilscher Korea Inc.  
Seongnam, Gyeonggi, 463-400  
Phone: +82 (0) 31-789-3715  
E-Mail: [info@hilscher.kr](mailto:info@hilscher.kr)

#### Switzerland

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

#### Support

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

#### USA

Hilscher North America, Inc.  
Lisle, IL 60532  
Phone: +1 630-505-5301  
E-Mail: [info@hilscher.us](mailto:info@hilscher.us)

#### Support

Phone: +1 630-505-5301  
E-Mail: [us.support@hilscher.com](mailto:us.support@hilscher.com)