



Getting Started Guide
Serial Dual-Port Memory Interface with netX

Hilscher Gesellschaft für Systemautomation mbH

www.hilscher.com

DOC120210GS04EN | Revision 4 | English | 2015-04 | Released | Public

Table of Contents

1	Introduction.....	3
1.1	About this Document.....	3
1.2	List of Revisions	3
1.3	Overview	4
1.4	Requirements.....	5
1.5	Supported Hardware and Limitations.....	5
1.6	Terms, Abbreviations and Definitions	6
1.7	References to Documents.....	6
1.8	Legal Notes	7
1.8.1	Copyright.....	7
1.8.2	Important Notes.....	7
1.8.3	Exclusion of Liability	8
1.8.4	Export.....	8
2	Hardware Connection	9
2.1	Pin Connection of netX 10	9
2.2	Pin Connection of netX 50	9
2.3	Pin Connection of netX 51/52	10
2.4	Pin Connection of netX 100/500	10
3	Enable the serial DPM Interface on the netX.....	11
3.1	netX 10 and netX 51/52	11
3.2	netX 50.....	13
3.3	netX 100/500.....	13
4	Host Software Implementation.....	14
5	Serial DPM Protocol Description	15
5.1	netX Chip Detection	15
5.2	netX10 serial DPM Protocol	16
5.3	netX 50 serial DPM Protocol.....	17
5.4	netX 51/52 serial DPM Protocol.....	18
5.5	netX 100/500 serial DPM Protocol.....	20
6	Appendix	21
6.1	List of Tables	21
6.2	List of Figures.....	21
6.3	Contacts	22

1 Introduction

1.1 About this Document

This manual guides through the process of interfacing any SPI capable host CPU to a netX based hardware via the serial dual-port memory (DPM) interface.

1.2 List of Revisions

Rev	Date	Name	Chapter	Revision
1	2012-02-06	SS	All	Created
2	2012-07-19	SS	All	Support for netX51/52 added
3	2014-07-24	HH	5.4	netX 51/52: Initialization of serial dual-port memory communication by two read commands from host.
4	2015-04-15	SS, HH	3.1 5.2	netX 51: Tag list configuration fixed (Figure 4). netX 10: Initialization of serial dual-port memory communication by two read commands from host.

Table 1: List of Revisions

1.3 Overview

This manual describes the serial DPM interface of the netX with the aim to support and lead you during the process of interfacing any CPU with a SPI unit, running under your own operating system. The concept of the serial DPM interface is illustrated in the following figure.

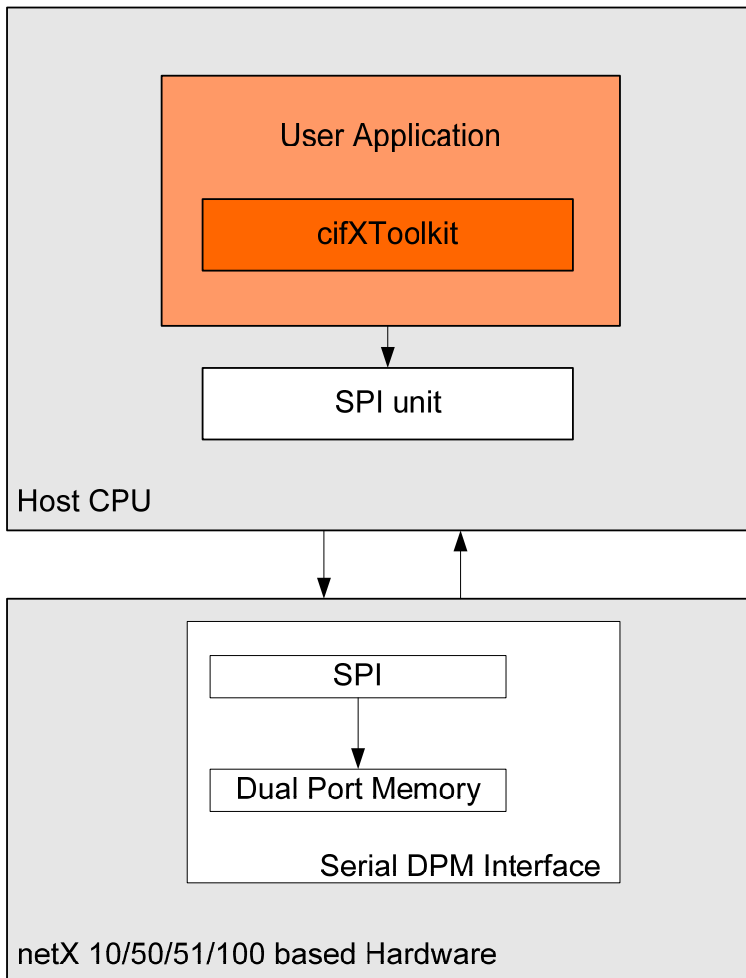


Figure 1: Serial DPM - Architecture

1.4 Requirements

- netX with serial DPM support (see section *Supported Hardware* on page 5)
- Host CPU with SPI unit capable to use default SPI Master mode 3 (clock idle state high with sampling on the trailing clock edge), the SPI Clock rate must not exceed
 - 50 MHz for netX 10,
 - 50 MHz for netX 51/52,
 - 16 MHz for netX 100/500 and
 - 10 MHz for netX 50
- cifXToolkit V1.1.0.0 or higher
- netX Tag List Editor for patching the generic netX 10/51/52 2nd Stage Bootloader.
- Knowledge of the serial DPM protocol (section *Serial DPM Protocol Description* on page 15).
- Guideline to port the cifX Toolkit to a custom CPU/OS. See manual [4].

Note: Theoretical up to 125 MHz should be possible (netX 10 / netX 51 / netX 52). Due to existing SPI Controllers max. 50 MHz are verified by tests

1.5 Supported Hardware and Limitations

The serial DPM is available for the netX hardware listed below:

- netX 10 (supported by hardware)
- netX 50 (software emulation with limited performance)
- netX 100/500 (handled by XC microcode, limited performance)
- netX 51 and netX 52, with its high performance SPS interface

Please consider the following limitations:

- netX 50

Serial DPM is handled via software by the ARM CPU itself. This results in a much lower performance of the serial DPM and limits the maximum SPI clock to 10 MHz. Furthermore, firmware which causes high CPU load will also influence the performance of the serial DPM interface.
- netX 100/500

Serial DPM handling on the netX100/500 is also done in software which limits the maximum SPI clock to 16 MHz. But CPU load, generated by the firmware, has no affect to the SPI because serial DPM handling is done by a separate XC unit.

 - SPI Modes 0 and 2 are not supported
 - Access to netX memory must be DWORD aligned
- The protocol for accessing the serial DPM differs slightly depending on the used netX chip.

Overview netX chip serial DPM features

Property	netX 10	netX 50	netX 51/52	netX 100/500
Implementation	Hardware	Software	Hardware	XC microcode
SPI activation	via 2nd stage bootloader	via firmware	via 2nd stage bootloader	via firmware
Supported SPI modes	0-3 (default 3) Adjustable via 2nd stage bootloader	0-3 (default 3) Adjustable via firmware	0-3 (default 3) Adjustable via 2nd stage bootloader	1, 3 (default 3) Adjustable via firmware
Maximum SPI clock	125 MHz (until 50MHz confirmed with test runs)	10 MHz (may be lower, depends on CPU load caused by firmware)	125 MHz (until 50MHz confirmed with test runs)	16 MHz
Address alignment	Byte	Byte	Byte	DWord

Table 2: Overview netX Chip serial DPM Feature

1.6 Terms, Abbreviations and Definitions

Term	Description
ARM	Advanced RISC Machines
CPU	Central Processing Unit
DPM	Dual Port Memory
LSB	Least Significant Bit
MSB	Most Significant Bit
SPI	Serial Peripheral Interface

Table 3: Terms, Abbreviations and Definitions

All variables, parameters, and data used in this manual have the LSB/MSB (“Intel”) data format. This corresponds to the convention of the Microsoft C Compiler.

All IP addresses in this document have host byte order.

1.7 References to Documents

This document refers to the following documents:

- [1] Hilscher Gesellschaft für Systemautomation mbH: Technical Reference Guide, netX 10, Revision 0.9, English, 2012
- [2] Hilscher Gesellschaft für Systemautomation mbH: Program Reference Guide netX 50, Revision 10, English, 2008
- [3] Hilscher Gesellschaft für Systemautomation mbH: SPI Slave as DPM Interface – netX 100/500, Revision 2, english
- [4] Hilscher Gesellschaft für Systemautomation mbH: cifX / netX Toolkit Manual, Revision 6, English, 2012

Table 4: References to Documents

1.8 Legal Notes

1.8.1 Copyright

© 2012-2015 Hilscher Gesellschaft für Systemautomation mbH

All rights reserved.

The images, photographs and texts in the accompanying material (user manual, accompanying texts, documentation, etc.) are protected by German and international copyright law as well as international trade and protection provisions. You are not authorized to duplicate these in whole or in part using technical or mechanical methods (printing, photocopying or other methods), to manipulate or transfer using electronic systems without prior written consent. You are not permitted to make changes to copyright notices, markings, trademarks or ownership declarations. The included diagrams do not take the patent situation into account. The company names and product descriptions included in this document may be trademarks or brands of the respective owners and may be trademarked or patented. Any form of further use requires the explicit consent of the respective rights owner.

1.8.2 Important Notes

The user 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 user 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 user 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 user 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.

1.8.3 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.

1.8.4 Export

The delivered product (including the technical data) is subject to export or import laws as well as the associated regulations of different countries, 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.

2 Hardware Connection

The netX 500/100/50/10 offers a SPI Slave interface which will be used for serial access to the DPM of the hardware. The general connection of the netX serial DPM to any SPI capable host CPU is illustrated below. Please note that the netX50 needs an additional chip select signal connected to a specific GPIO input (required GPIO input depends on the firmware).

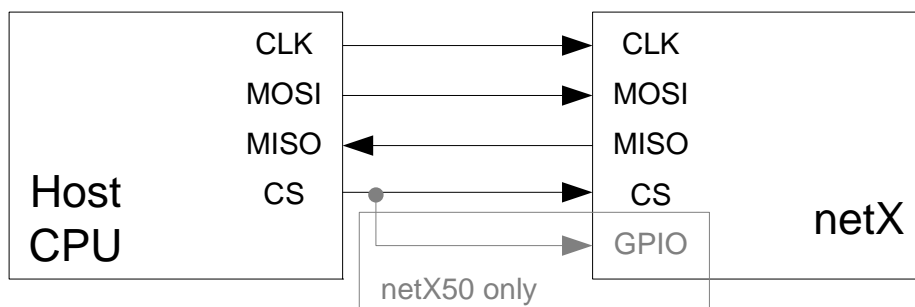


Figure 2: Serial DPM - Hardware Connection

2.1 Pin Connection of netX 10

Please connect the SPI signals of your host CPU to the netX10 hardware according to the pinning table given below.

Function	Pin	netX10 Signal Name
CLK	E14	DPM_D11 / DPM_SPI_CLK
MOSI	G14	DPM_D09/ DPM_SPI_MOSI
MISO	H14	DPM_D08 / DPM_SPI_MISO
CS	F14	DPM_D10 / DPM_SPI_CS _n

Table 5: Pin Connection of the netX 10

2.2 Pin Connection of netX 50

Since the hardware connection of the serial DPM is determined by the firmware running on the netX50, a hardware pinning cannot be given here.

The SPI pins can be mapped by the firmware anywhere on the MMIO matrix of the netX 50 (see reference [2], section 2.3: IO Configuration).

Note: Please note that in difference to netX 10/100/500, the netX 50 needs an additional chip select signal connected to a specific GPIO input. This GPIO is also defined by the firmware. Please consult the hardware/firmware manual or contact Hilscher support to find out the hardware connection of the serial DPM.

2.3 Pin Connection of netX 51/52

Please connect the SPI signals of your host CPU to the netX51/52 hardware according to the pinning table given below.

Function	netX 51 Pin / Signal Name	netX 52 Pin / Signal Name
CLK	G16 / DPM_SPI_CLK	D18 / DPM_SPI_CLK
MOSI	H15 / DPM_SPI_MOSI	G16 / DPM_SPI_MOSI
MISO	J16 / DPM_SPI_MISO	G18 / DPM_SPI_MISO
CS	H16 / DPM_SPI_CSn	F17 / DPM_SPI_CSn

Table 6: Pin Connection of the netX 51/52

2.4 Pin Connection of netX 100/500

The serial DPM handling on netX100/500 is done by a XC unit. The firmware running on the netX determines which XC unit shall be used.

Please connect the SPI signals of your host CPU to the netX 100/500 hardware with respect to the used XC unit according to the pinning table given below.

Function	netX 100/500 Pin / Signal Name			
	XC0	XC1	XC2	XC3
CLK	T21 / XM0_ECLK	V21 / XM1_ECLK	T20 / XM2_ECLK	V20 / XM3_ECLK
MOSI	N21 / XM0_RX	P21 / XM1_RX	R21 / XM2_RX	U21 / XM3_RX
MISO	N20 / XM0_TX	P20 / XM1_TX	R20 / XM2_TX	U20 / XM3_TX
CS	N19 / XM0_IO0	P19 / XM1_IO0	R19 / XM2_IO0	U19 / XM3_IO0

Table 7: Pin Connection of the netX 100/500

3 Enable the serial DPM Interface on the netX

The serial DPM interface is disabled by default. Depending on the netX chip the required action to enable the serial DPM differs. The necessary actions for each netX chip are described in the sections below.

3.1 netX 10 and netX 51/52

Activating the serial DPM on netX 10 and netX 51/52 is done by the 2nd stage bootloader.

With release of the generic 2nd stage bootloader V1.4.8.0 (V1.4.9.0 for netX51/52 based devices) enabling of serial or parallel DPM is done according to the DIRQ/SIRQ pins. Detailed information about the automatic detection via DIRQ/SIRQ pins is given in the table below.

DIRQ	SIRQ	Mode
0	X	Serial DPM (SPI mode 3)
1	0	16 Bit parallel DPM
1	1	8 Bit parallel DPM

Table 8: Serial DPM autodetection via DIRQ/SIRQ pin

Function	netX 10 Pin / Signal Name	netX 51 Pin / Signal Name	netX 52 Pin / Signal Name
DIRQ	G12 / DPM_DIRQ	C17 / DPM_DIRQn	D16 / DPM_DIRQn
SIRQ	A12 / DPM_SIRQ	B11 / DPM_SIRQn	C08 / DPM_SIRQn

Table 9: DIRQ/SIRQ pin Connection of the netX 10/51/52

Note: Serial DPM enabled via DIRQ/SIRQ will always work in SPI mode 3. To use a different mode, please consider manual activation of the serial DPM via Tag List Editor.

Furthermore serial DPM access can be enabled explicitly. This can be done by patching the 2nd stage bootloader with the netX Tag List Editor (please note that patching the 2nd stage bootloader will override the autodetection mechanism stated above).

Changing the 2nd Stage Loader DPM handling to SPI:

- Start the Tag List Editor and load the generic 2nd stage bootloader for netX 10
- Change the netX 10/51/52 HIF/DPM configuration as illustrated in the figure below

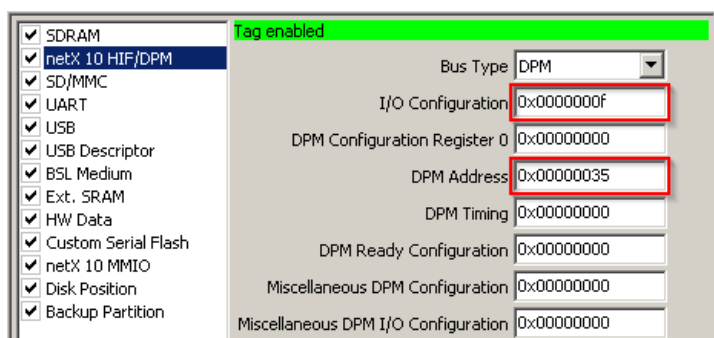


Figure 3: Patching the 2nd Stage Bootloader to enable serial DPM for netX 10

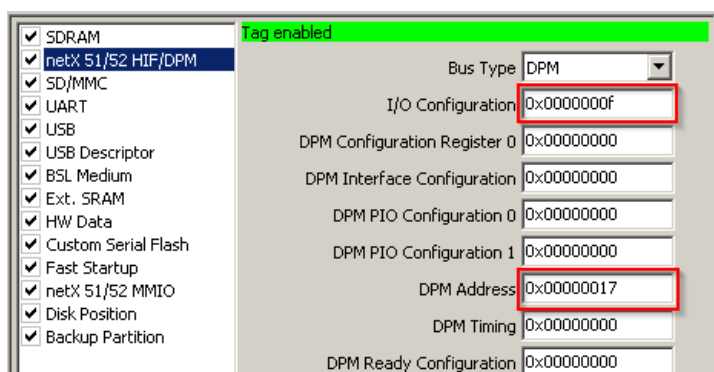


Figure 4: Patching the 2nd Stage Bootloader to enable serial DPM for netX 51/52

Note: To get started, a ready to use 2nd stage bootloader, with activated serial DPM access, is included in the examples.

If you want to use a different SPI mode than the default (mode 3), please consider the table below:

SPI mode	I/O Configuration (see Tag List Editor)
0	0x00000003
1	0x00000007
2	0x0000000B
3 (default)	0x0000000F

Table 10: SPI Mode Configuration (netX 10/51/52)

- Save the changes and flash the 2nd stage bootloader to your netX10/51/52 device

Any netX10/51/52 compatible fieldbus firmware will now work with serial DPM access.

3.2 netX 50

In contrast to the netX10 hardware implementation, where the SPI controller is able to directly access the netX DPM without ARM CPU interaction, on a netX 50 Chip, the ARM CPU is necessary to evaluate the SPI data and to handle the data transfer between the netX DPM and the SPI controller.

Note: Enabling serial DPM on netX50 requires customizing your fieldbus firmware. Please contact Hilscher support to equip your fieldbus firmware with serial DPM mode.

Please note that due to the necessary ARM CPU handling, the running Firmware has performance effects to the SPI transfer! Firmware causing a high CPU load, may limit the maximum SPI clock rate to 5 MHz or even lower!

Note: We recommend to use the pin compatible netX 51 chips instead of the netX 50 when using the serial DPM interface.
The netX 51 serial DPM interface is hardware driven, allowing much higher SPI clock rates, independent of the firmware CPU load and without changing the standard firmware to enable serial DPM functions.

3.3 netX 100/500

The serial DPM on netX100/500 is done by a separate XC inside the netX. A dedicated microcode is necessary to enable the serial DPM feature. This microcode is loaded by the firmware running on the ARM CPU.

Note: Please refer to manual [3] (Section 2: The Interface) and contact Hilscher support to equip your fieldbus firmware with the serial DPM microcode.

4 Host Software Implementation

A Host application will use the CIFX API, offered by all HILSCHER drivers. Basis of the Hilscher drivers is the cifX Toolkit offering the basic access functions to the Hilscher defined DPM.

In general the cifX-Toolkit is independent of any operating system and can be used with or without an operating system and it is scalable.

A short instruction on how to port the cifX Toolkit to your own embedded system is given in reference [4] (section 2: How to port the cifX Toolkit).

To allow the Toolkit to handle netX based devices which are connected via a SPI interface, an optional custom hardware interface was introduced with release of version 1.1.0.0. This custom hardware interface extends the basic access functions by Read/Write function calls, replacing the default *memcpy()* and pointer access commands used in the Toolkit to access the physical DPM of the netX.

Please refer to the Toolkit Manual [4] (section 4.10: Custom Hardware Access Interface) to get detailed information about the custom hardware interface.

The functional principle of the custom SPI hardware interface is illustrated on the basis of the *xChannelGetMBXState()* call (see figure below):

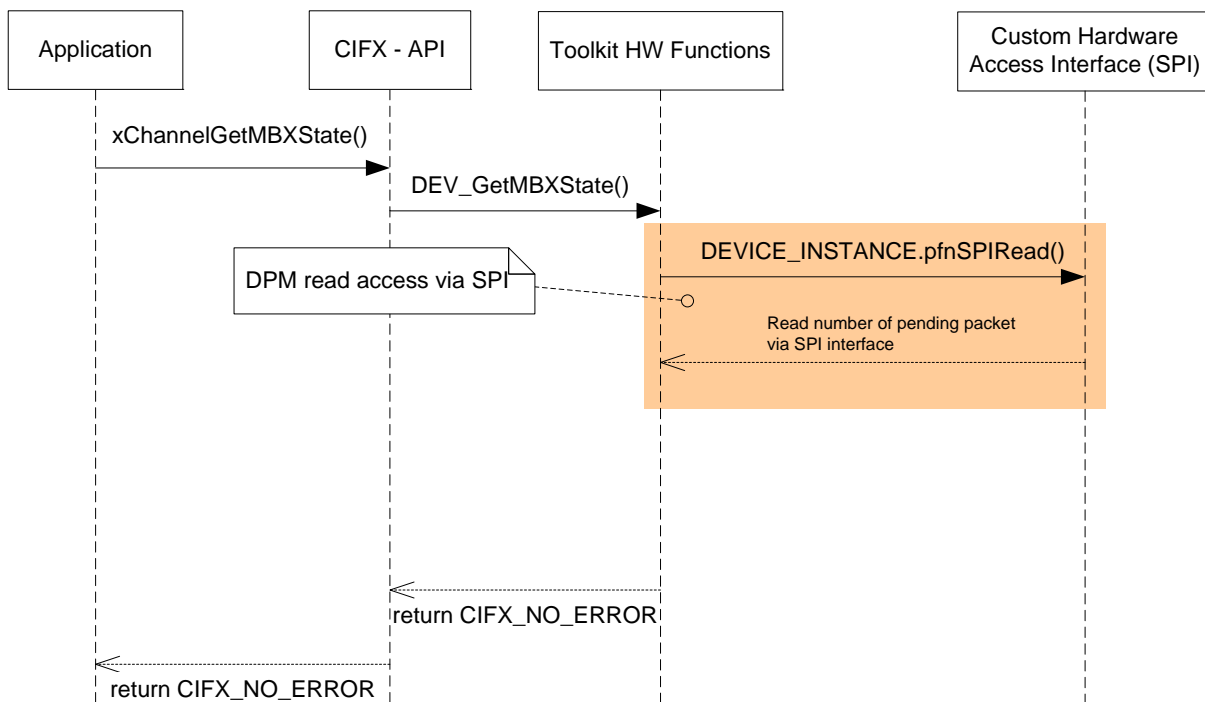


Figure 5: cifXToolkit - SPI Access Interface

The *Read/Write* function calls, to access the SPI unit, must be implemented by the user, since SPI handling depends highly on the used host CPU and host controller. The protocol for accessing the serial DPM differs slightly depending on the used netX chip.

For an easy start refer to the host examples offered with this manual.

5 Serial DPM Protocol Description

The general structure of the serial DPM protocol consists of a header and a subsequent data section. The header is always generated by the host CPU SPI Master interfaces via MOSI. It contains address information, control information and length of transfer information.

While address and length must be a multiple of 4 on netX 100/500, this is not necessary on netX 10 and netX 50. The control information determines if the current transfer reads data from or writes data to the netX.

5.1 netX Chip Detection

As stated above, the protocol differs for each netX chip type. The protocol type can be detected by evaluating the data returned by the netX while transferring the header section of each data transfer. The data returned by the netX chips are listed below:

- netX 10: Returns always 0x00 during transfer of the header section
- netX 50: Returns always 0xFF during transfer of the header section
- netX 51: During transfer of the header section first byte returned is serial DPM status (Status is fine for Bit[0:4] == 0x11)
- netX 100/500: During transfer of the header section first byte returned is 0x64

Serial DPM Header section for read access

	DPM Address Offset: 255		Control information and length: Read 4 bytes
MOSI	0x00	0xFF	0x84
MISO netX10	0x00	0x00	0x00
MISO netX50	0xFF	0xFF	0xFF
MISO netX51	Bit[0:4]: 0x11
MISO netX100	0x64

Figure 6: Detect serial DPM Protocol

Note: To get started, it is highly recommended to use the examples enclosed with this manual.

5.2 netX10 serial DPM Protocol

Since the serial DPM mode allows direct access to the DPM memory area without any software interaction on netX side (e.g. moving data to and from Send and Receive FIFOs), an access protocol had to be defined which is described below.

The MSB is always transferred first in the serial data. Each transfer starts with a transfer-header and is followed by at least one data byte to be exchanged between host CPU and netX. Transfer header length is 3-byte or 4-byte depending on programmed external address range: less and equal than 64K: 3-Byte, 128K: 4-Byte.

Important: The host has to do **two** read commands to initialize the serial dual-port memory communication (SPI). Any address can be used.

The chip select signal (DPM_SPI_CS_n) is used by the DPM interface as asynchronous reset of receive and transmit logic, hence DPM_SPI_CS_n must remain active throughout the complete transfer and must become inactive between sequential transfers.

Serial transfers are mapped inside netX DPM interface to standard 8-bit parallel DPM accesses. Hence each transfer must provide the address to be accessed and the access type (read or write) as transfer-header information. Even using small external address range, this information takes at least 3 bytes. To avoid a huge transfer overhead by transmitting the header before each single data byte, the header is followed by a data byte stream. With each data byte, the related address is automatically incremented by one.

Using external address range (DPM size) of 128K requires 4-byte instead of 3-byte header. Header size is derived from the value programmed in external address range configuration register. The default after reset is the smallest supported external address range. Hence after netX 10 reset always 3-byte header must be used. If external address range is configured to 128K, all headers after configuration access to external address range configuration register must be 4-byte headers.

The following table shows the transfer header structure depending on the programmed external address range:

Address range	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4
64K or less	Addr. [MSB..8]	Addr. [7..0]	nW/R, length	Data 0	(Data 1)
128K	Addr. [MSB..16]	Addr. [15..8]	Addr. [7..0]	nW/R, length	Data 0

Figure 7: SPI Frame Format (netX 10)

The header is always generated by host CPU SPI interface. First 2 bytes (3 for external address range of 128K) contains address information (MSB first). These bytes are always followed by a control byte. Control byte MSB defines data transfer direction. If this bit is set, the current transfer reads data from netX, otherwise data is written to the netX. The lower 7 bits of the control byte contain transfer length information (number of bytes to be read).

Read accesses through the serial DPM interface, internally always result in read ahead accesses, as otherwise it would not be possible to deliver serial read data on SPI_MISO without data polling or some kind of ready handshake mechanism. To avoid problems with access sensitive address areas (e.g. FIFOs), the number of bytes to be read can be specified in the header, allowing to stop the read-ahead before reaching an access sensitive area. If the transfer length is set to 0, internal consecutive read accesses are done until the transfer is ended by the host by setting SPI_CS_n to high level, while there will be always one internal read access beyond the last byte that was actually transferred. The length parameter is ignored for write accesses, as the number of bytes to

be written simply arises from the number of bytes transferred by the host before ending the transfer. Please refer to reference [1] (Section 2.9.3 Serial (SPI) DPM interface mode) for detailed information about the serial DPM protocol of the netX 10.

5.3 netX 50 serial DPM Protocol

Because of the necessary ARM CPU handling, the running Firmware has performance effects to the SPI transfer. To minimize these effects, the SPI protocol is extended by some special handling described in the following protocol description.

The header is always generated by the host CPU SPI interface. The first 2 bytes contains address information (MSB first), followed by a control byte. Control byte MSB defines data transfer direction (nW/R). If this bit is set, the current transfer reads data from netX, if it is not set, data is written to netX. The proceeding 7 lowest bits of the control byte contain transfer length information (length, bytes to be transferred). The length has to be set to a value between 1 and 127. Data transfers which are done until the end of SPI transfer (chip-select inactive) without setting the transfer length information are not supported.

Write Access

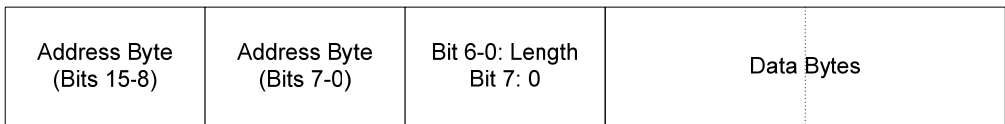


Figure 8: SPI Frame Format for Write Request (netX 50)

The header is immediately followed by the data bytes written to the netX DPM. As the length field is 7 bit in width, a single write access must not exceed a maximum of 127 data bytes. The chip select signal is active during the whole SPI frame transfer.

Read Access

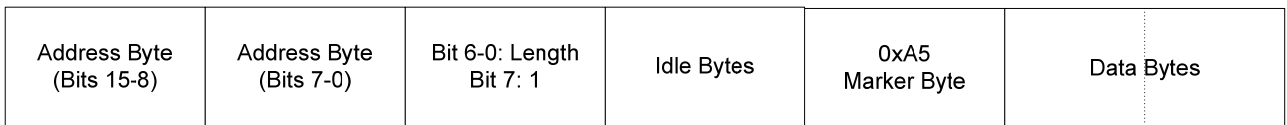


Figure 9: SPI Frame Format for Read Request (netX 50)

In contrast to the netX10, the SPI protocol implementation of the netX50 differs in the way of transferring the data bytes for read accesses. A read access requires the SPI Master to pause a period of time after transferring the header. So the slave device is able to prepare the data to read from netX DPM. The SPI master generates cyclic idle bytes immediately after the header as long as the SPI slave device does not signal its ready state with a special marker byte (0xA5). The SPI master in turn polls the first data byte by transferring the same marker byte to the slave. As the length field is 7 bit in width, a single read access must not exceed a maximum of 127 data bytes. The chip select signal is active during the whole SPI frame transfer.

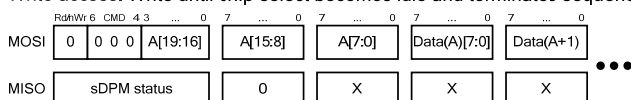
5.4 netX 51/52 serial DPM Protocol

Basically all serial DPM transfers of SDP2 consist of a header followed by a byte-orientated address-incrementing data stream. All bytes (header and data) are serial transmitted MSB first (i.e. standard SPI). Chip-select signal must remain active all time during an access sequence (inactive chip-select will terminate a transfer). The transfer-header consists always of a direction-bit (read/not-write bit), three command (cmd) bits and a 20 bit address.

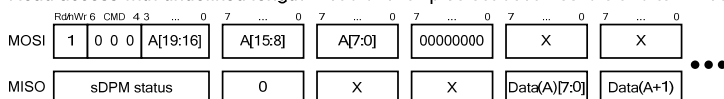
Important: The host has to do **two** read commands to initialize the serial dual-port memory communication (SPI). Any address can be used.

To enable straight data stream mode all command bits must be zero. Other settings for command are reserved. The transfer length for read and write access are given in the table below. All headers of a read-sequence are extended by an additional length-byte before the first data byte. This is necessary to avoid problems with read-sensitive addresses (e.g. FIFOs) as read accesses must be performed as read-ahead internally. When a read sequence is not terminated after <length> bytes, invalid read data will be returned. No read access will be done netX inside then.

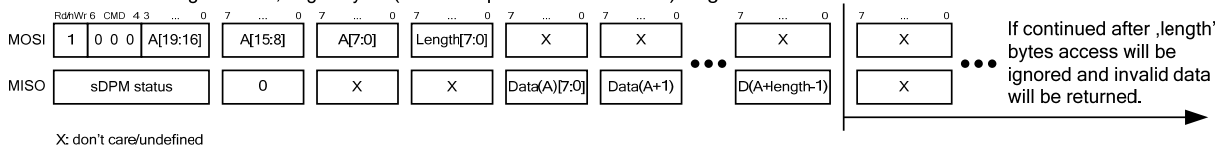
Write access: Write until chip-select becomes idle and terminates sequence:



Read access with undefined length: Read until chip-select becomes idle and terminates sequence, send 0 for length:



Read access with length: Read 'length' bytes (or until chip-select becomes idle) - e.g. FIFO read:



X: don't care/undefined

Figure 10: SPI Frame Format for Read/Write Request (netX 51/52)

During the first byte of the header the current serial DPM status will always be clocked out on MISO. This provides information about the current DPM error states and could be used when access timing is unpredictable (similar to netX 10 error status checking via dpm_status register). See table below.

Bits	Name	Description
7:5	-	reserved (for DPM-SPI read data will be unpredictable)
4	SEL_DPM_SERIAL	DPM_MODE configuration input state. 0: DPM is in parallel mode (DPM_MODE configuration input is low). 1: DPM is in serial mode (DPM_MODE configuration input is high).
3	RDY_TO_ERR	DPM_RDY Timeout Error Status Flag. This error could occur if host device tries to access permanently busy netX address area (e.g. netX xPEC program RAM while xPEC is running). To avoid host device stalling DPM_RDY sig. is released to ready state after 2048 system clock cycles (i.e. 20.48us) at least. 1: Last access went to netX busy address and was broken to avoid host device stalling. 0: Access was finished successfully by DPM_RDY assertion to ready state.
2	WR_ERR	DPM Write Error Status Flag. Write errors occur if ready signal (DPM_RDY) is not respected by host device and external DPM write access terminated before data could be stored. In some cases certain netX address areas could be busy for not predictable time. If DPM_RDY is not used, check for write error after write access to these areas. In case of write error this bit is set immediately after the appropriate write access. Repeat the write access until no error occurs. 1: The external DPM write access was too fast to store write data. Repeat the write access. 0: Write access terminated without error.
1	RD_ERR	DPM Read Error Status Flag. Read errors occur if ready signal (DPM_RDY) is not respected by host device and external DPM read access terminated before read data could be asserted on the external DPM data bus (view also t_rds in dpm_timing_cfg register). In case of read error this bit is set immediately after the appropriate read access. Repeat the read access until no error occurs. 1: The external DPM read access was too fast. Repeat the read access. 0: Read data OK.
0	UNLOCKED	DPM is locked during netX power up and boot phase. DPM access to other addresses than these DPM control address area cannot be done before this bit is set to 1. Poll for 1 after power up or reset.

Table 11: Serial DPM status (netX 10/51/52)

5.5 netX 100/500 serial DPM Protocol

The header is always generated by the host CPU SPI Master interface via MOSI. The first 2 bytes contains address information (MSB first), followed by a control byte. The address must be a multiple of 4. Control byte MSB defines data transfer direction (nW/R). If this bit is set, the current transfer reads data from netX, if it is not set, data is written to netX. Proceeding 7 LSBs of the control byte contain transfer length information (length, bytes to be transferred). The length information must be a multiple between 4 and between 4 and 124. The SPI Slave interface maps the 16-bit address information into a 32-bit internal netX address using 3 configurable windows.

During the first address information byte the netX will send its ID character on MISO. The value of the ID for netX100/500 is 0x64 (MSB first). During the second address information byte the netX will send a confirmation byte about the previous request on MISO (MSB first). The byte has following meaning:

- Bit 0: the translated internal address was invalid
- Bit 1: chip-select de-assert error
- Bit 2: the received length was smaller 4 or greater 124 or not a multiple of 4
- Bit 3: write request detected with received length smaller header length
- Bit 4: a new request (read or write) was received before previous write request finished

Write Access

Address Byte (Bits 15-8)	Address Byte (Bits 7-0)	Bit 6-0: Length Bit 7: 0	Data Bytes
-----------------------------	----------------------------	-----------------------------	------------

Figure 11: SPI Frame Format for Write Request (netX 100/500)

The header is immediately followed by the data bytes written to the netX internal memory (on MOSI). As the length field is 7 bit in width, a single write access must not exceed a maximum of 124 data bytes. The chip select signal is active during the whole SPI frame transfer.

Read Access

Address Byte (Bits 15-8)	Address Byte (Bits 7-0)	Bit 6-0: Length Bit 7: 1	Idle Bytes	0xA5 Marker Byte	Data Bytes
-----------------------------	----------------------------	-----------------------------	------------	---------------------	------------

Figure 12: SPI Frame Format for Read Request (netX 100/500)

Each access requires the SPI Master to pause a period of time after transferring the header. So the slave device is able to prepare the data to read from netX internal memory. The SPI master generates cyclic idle bytes immediately after the header as long as the SPI slave device does not signal its ready state with a special marker byte (0xA5). The SPI master in turn polls the first data byte (on MISO) by transferring a dummy byte to the SPI slave (on MOSI). As the length field is 7 bit in width and the interface only works dword-granular, a single read access must not exceed a maximum of 124 data bytes. The chip select signal is active during the whole SPI frame transfer. Please refer to reference [3] (Section 3.2 Access Request Frame Format) for detailed information about the serial DPM protocol of the netX 100/500.

6 Appendix

6.1 List of Tables

Table 1: List of Revisions	3
Table 2: Overview netX Chip serial DPM Feature	6
Table 3: Terms, Abbreviations and Definitions	6
Table 4: References to Documents	6
Table 5: Pin Connection of the netX 10	9
Table 6: Pin Connection of the netX 51/52	10
Table 7: Pin Connection of the netX 100/500	10
Table 8: Serial DPM autodetection via DIRQ/SIRQ pin	11
Table 9: DIRQ/SIRQ pin Connection of the netX 10/51/52	11
Table 10: SPI Mode Configuration (netX 10/51/52)	12
Table 11: Serial DPM status (netX 10/51/52)	19

6.2 List of Figures

Figure 1: Serial DPM - Architecture	4
Figure 2: Serial DPM - Hardware Connection	9
Figure 3: Patching the 2nd Stage Bootloader to enable serial DPM for netX 10	12
Figure 4: Patching the 2nd Stage Bootloader to enable serial DPM for netX 51/52	12
Figure 5: cifXToolkit - SPI Access Interface	14
Figure 6: Detect serial DPM Protocol	15
Figure 7: SPI Frame Format (netX 10)	16
Figure 8: SPI Frame Format for Write Request (netX 50)	17
Figure 9: SPI Frame Format for Read Request (netX 50)	17
Figure 10: SPI Frame Format for Read/Write Request (netX 51/52)	18
Figure 11: SPI Frame Format for Write Request (netX 100/500)	20
Figure 12: SPI Frame Format for Read Request (netX 100/500)	20

6.3 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

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.
Pune, Delhi, Mumbai
Phone: +91 8888 750 777
E-Mail: info@hilscher.in

Italy

Hilscher Italia S.r.l.
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.
Seongnam, Gyeonggi, 463-400
Phone: +82 (0) 31-789-3715
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