Napatech
NT Adapters
with Napatech Driver Software
Software Installation Guide for Linux

Document Number	DN-0128
Revision	2
Abstract	This document explains how to install and build the Napatech Linux Driver and how to upload new images to the flash memory in the FPGA of the Napatech NT adapters with Napatech Driver Software.

Modification history

This document has been updated as follows:

Rev.	Date	Comment
1	2011-01-28	Initial version.
2	2011-03-18	Changes have been made in Section 2.2. A number of minor changes have been made.

Contents

Preface

Style conventions

Bold typeface is used for names of, for instance, keys (example: press **Return**). Note that bold typeface is also used in other contexts.

Italics are used for names of directories and paths (example: */opt/napatech/bin*).

`Monospaced typeface` is used for code (example: `sh install.sh debug`).

Abbreviations

This table explains the abbreviations used in this document.

Abbreviation	Explanation
API	Application Programming Interface
BIOS	Basic Input/Output System
CPLD	Complex Programmable Logic Device
DN	Document Number
Fpga, FPGA	Field-Programmable Gate Array
IP	Internet Protocol
NEBS	Network Equipment-Building System
nt, NT	NapaTech
Os, OS	Operating System
PCB	Printed Circuit Board
PCI	Peripheral Component Interconnect
Ref.	REFerence
Rev.	REVision
STD	STanDard
SW	SoftWare
TUN	TUNnel

References

This table shows the documents that are referenced by this document.

Ref.	Document Title
1	Napatech, Network Adapters with Napatech Driver Software, Programmer's Guide, DN-0059
2	Napatech, NT20X, Hardware Installation Guide, DN-0105

Ref.	Document Title
3	Napatech, Network Adapters with Napatech Driver Software, Tools Guide, DN-0159
4	Napatech, NT4E-4, Hardware Installation Guide, DN-0166
5	Napatech, NT4E-4T, Hardware Installation Guide, DN-0167
6	Napatech, NT20E, Hardware Installation Guide, DN-0169
7	Napatech, NTPORT4E-4, Hardware Installation Guide, DN-0202
8	Napatech, NTPORT4E-4T, Hardware Installation Guide, DN-0203
9	Napatech, NT4E-4-STD, Hardware Installation Guide, DN-0248
10	Napatech, NT4E-4T-STD, Hardware Installation Guide, DN-0249
11	Napatech, NTBPE, Hardware Installation Guide, DN-0271
12	Napatech, NT20E2, Hardware Installation Guide, DN-0295
13	Napatech, NT4E-4-NEBS, Hardware Installation Guide, DN-0488
14	Napatech, NT20E-NEBS, Hardware Installation Guide, DN-0490

1 Introduction

In this document

This document contains information about SW installation related to the Napatech NT adapters, NT20X, NT20E, NT20E-NEBS, NT20E2, NT4E, NT4E-4-NEBS, NTPORT4E, NT4E-STD and NTBPE, with Napatech Driver Software.

Caution: Refer to these documents for information on how to install the adapter in a server:

- NT20X: Ref. 2 on page 5
- NT20E: Ref. 6 on page 6
- NT20E-NEBS: Ref. 14 on page 6
- NT20E2: Ref. 12 on page 6
- NT4E-4: Ref. 4 on page 6
- NT4E-4-NEBS: Ref. 13 on page 6
- NT4E-4T: Ref. 5 on page 6
- NTPORT4E-4: Ref. 7 on page 6
- NTPORT4E-4T: Ref. 8 on page 6
- NT4E-4-STD: Ref. 9 on page 6
- NT4E-4T-STD: Ref. 10 on page 6
- NTBPE: Ref. 11 on page 6

Terminology

In this document the term NT4E does not include NT4E-4-NEBS and NT4E-STD adapters, and the term NT20E does not include NT20E-NEBS and NT20E2 adapters.

Reading instructions

The document contains these chapters:

- "1 Introduction" on page 7. This chapter introduces the document.
- "2 Installing the Napatech Linux Driver" on page 9. This chapter describes how to install the Linux driver.
- "3 Flash Memory Images on the NT Network Adapters" on page 15. This chapter describes the maintenance of the FPGA of the NT network adapters.
- "4 Flash Memory Images on the NTBPE Bypass Adapters" on page 35. This chapter describes the maintenance of the firmware of the NTBPE bypass adapters.

2 Installing the Napatech Linux Driver

In this chapter

This chapter explains how to install the Napatech Linux Driver.

The chapter contains these sections:

2.1 Preparing the Server

To prepare the server

You receive the Napatech Linux Driver as source code from the Napatech Support Center. To prepare the server for installation of the driver:

Step	Action
1	Install Linux on the server.
2	Copy the product or driver package from the Napatech Support Center, and extract the *software* directory including all subdirectories to the server.
3	If you want to prepare for 32-bit application support on a 64-bit OS, add the line: `#define NT_COMPAT_32BIT` in the **include/ostype.h** file. **Warning: This will break the backwards compatibility.**

2.2 Installing and Compiling the Linux Driver and the Driver Tools

To install and compile the Linux driver and the driver tools

To install and compile the Napatech Linux Driver and all the driver tools:

Note: Root privileges are required for compiling and installing the Linux device driver.

Step	Action
1	Change directory to *scripts*.

Step	Action
2	To compile the driver, write `sh install.sh parameters` where *parameters* are optional (see "Install parameters" on page 10), and press **Return**.

The compiled driver tools are placed in */opt/napatech/bin*.

Install parameters

This table explains the optional install parameters. One or more parameters separated by spaces can be supplied with one **install** command.

Command	Description
`sh install.sh debug`	Enables debug information.
`sh install.sh help`	Shows a message explaining the install parameters. If other parameters are supplied together with `help`, these other parameters will have no effect.

`sh install.sh KernelDir=path`
where *path* is the kernel path in question. Example:
`KernelDir=/usr/src/linux-2.6`
	Compiles the driver with a specific kernel path.

`sh install.sh MakeOpts="option"`
where *option* is the make option in question. Example:
`MakeOpts="-j3"`
	Adds a make option.

`sh install.sh MaxAdapters=number`
where *number* is the maximum number of adapters. Example:
`MaxAdapters=2`
	Specifies the maximum number of adapters. The default value is 4.

`sh install.sh MaxPacketFeeds=number`
where *number* is the maximum number of packet feeds. Example:
`MaxPacketFeeds=16`
	Specifies the maximum number of packet feeds. The default value is 32.

`sh install.sh MaxSegments=number`
where *number* is the maximum number of segments. Example:
`MaxSegments=8`
	Specifies the maximum number of segments. The default value is 16.
	Note: For TX segments the maximum number of segments can never be higher than 16 regardless of the value of `MaxSegments`.

Command	Description
`sh install.sh NtOnlyLib`	Applies if you are using both NT and X adapters in one system, and you want NT/X-specific independent libraries. Makes a common interface library that only supports NT adapters. This library (**libntonlycommoninterface.so**) must be used together with the **libxycommoninterface.so** library to make sure that XYCI calls are not going through the **libntonlycommoninterface.so** library (see Ref. 1 on page 5).

2.3 Loading the Driver and Configuring the Interfaces

To load the driver and configure the interfaces

When you have compiled the Napatech Linux Driver, you can load it. To load the driver and configure the interfaces:

Note: Root privileges are required for operating the Linux driver.

Step	Action
1	Change directory to */opt/napatech/bin*.
2	To load the driver, write `sh load_driver.sh` and press **Return**.
3	If you want the adapters to run in OS mode (see "2.3.1 OsMode" on page 12), write `/opt/napatech/bin/OsMode` and press **Return**.
4	If the adapters are running in OS mode: a) To configure interface 0, write `ifconfig ntxs0 yyy.yyy.yyy.yyy up` where *yyy.yyy.yyy.yyy* is an IP address matching your network, and press **Return**. b) To configure the other interfaces, repeat Step 4 a) for each interface.

Note: A number of configuration parameters can be used in connection with the **load_driver.sh** command. They are described in Ref. 1 on page 5.

2.3.1 OsMode

The OsMode tool

The **OsMode** tool enables OS mode (standard interface mode) for all adapters. This mode forces the adapters to mimic a standard Ethernet interface enabling the use of Berkeley sockets and OS bring-up/bring-down operations. The **OsMode** tool destroys all existing packet feeds in the system, and creates a standard network interface per port called ntxs<*port number*>. The tool relies on the TUN/TAP device to be loaded.

Syntax

The syntax for **OsMode** is as shown below.

```
OsMode [-help]
```

Command example

This is an example of how to execute **OsMode**.

```
/opt/napatech/bin/OsMode
```

Output example

This is an example of an output from the **OsMode** tool.

```
OsMode (v. 4.22.A - 2010-12-10-12-08-09)
================================================================
Created: ntxs0
Created: ntxs1
```

2.4 Completing 32-bit Support on a 64-bit Platform (Optional)

To complete 32-bit support on a 64-bit platform

If you have prepared for it (see Step 3 in "To prepare the server" on page 9) you can complete the 32-bit application support on a 64-bit platform:

Step	Action
1	Compile a 32-bit version of **libntcommoninterface.so**.
2	Create the directory */opt/napatech/lib32*.
3	Copy the 32-bit version of **libntcommoninterface.so** to */opt/napatech/lib32*.
4	In the **/etc/ld.co.conf** file, add the line: `/opt/napatech/lib32`
5	Run **ldconfig**.
6	Run a 32-bit application on the 64-bit platform to check that it works.

Once 32-bit applications are supported on a 64-bit platform, applications must be compiled with **CFLAGS=-DNT_COMPAT_32BIT**.

To determine if an application is 32-bit or 64-bit

To determine if an application is a 32-bit or a 64-bit application:

Step	Action
1	Run **objdump -d <application> | grep “file format”**.

3 Flash Memory Images on the NT Network Adapters

In this chapter

This chapter describes how to upload a new image to the flash memory in the FPGA of an NT network adapter, get the new image up and running and get information concerning the images in flash memory and the image that the FPGA is currently running.

Note: This chapter does not apply to NTBPE adapters.

The chapter contains these sections:

3.1 Overview

FPGA images

The main headlines for the FPGA images are:

- An NT adapter has two FPGA images in the flash memory.
- At manufacturing these two images are identical.
- It is possible for the customer to update the FPGA images in the flash memory (how this is done is described in "3.4.2 Writing a new FPGA Image to the Flash Memory" on page 28).
- It is not possible to update the FPGA image that the FPGA has been booted on.
- One image is defined to be the primary image, and the other image is defined to be the secondary image.
- It is possible for the customer to configure which of the two images is the primary image.
- The two image locations are numbered 0 and 1. Each location can contain either the primary or the secondary image.
- By default the FPGA is booting on the primary image.
- If the image to be loaded is corrupted, the adapter will automatically load the other image.
- The **FpgaImageStatus** tool (see "3.3.4 FpgaImageStatus" on page 21) can read the FPGA boot status from the driver (such as the states of the images in the flash memory; and which image is loaded).

FPGA boot

Booting of the FPGA is always done by the AVR for NT20E2 adapters, and by the CPLD for other network adapters. At power up a flash setting controls

which image is booted. When the FPGA and the driver are up and running, reboots of the FPGA can be controlled via the driver API.

Tools and scripts for FPGA maintenance

These tools and scripts are useful for FPGA maintenance:

- **UpdateImage**
- **unload_driver**
- **ChangePrimaryImage** (only used indirectly)
- **FpgaImageStatus**
- **DriverLog**

The tools/scripts are located in the */opt/napatech/bin* directory. They are described in detail in "3.3 Tools and Scripts" on page 17.

3.2 Uploading, Activating and Verifying FPGA Images

To upload, activate and verify a new image compatible with the current driver

This is the general procedure for uploading, activating and verifying a new image that is compatible with the current driver (see the relevant driver release note for information about FPGA compatibility).

Step	Action
1	Load the driver by using the **load_driver** script.
2	Upload the new image to the flash memory by using the **UpdateImage** tool (see "3.3.1 UpdateImage" on page 18).
3	Unload the driver and change the primary image by using the **unload_driver** script (see "3.3.2 unload_driver" on page 19). The server is rebooted.
4	Load the driver by using the **load_driver** script.
5	Get status information about the images in the flash memory and the active FPGA image by using the **FpgaImageStatus** tool (see "3.3.4 FpgaImageStatus" on page 21).

Command example

This is an example of a command sequence for uploading, activating and verifying a new image that is compatible with the current driver.

```
/opt/napatech/bin/load_driver.sh

/opt/napatech/bin/UpdateImage -img
  /images/200-0008-05_0_070524_1547.img -ini nn -opt Program

/opt/napatech/bin/unload_driver.sh ChangePrimaryImage all
```

(Wait for reboot.)

```
/opt/napatech/bin/load_driver.sh

/opt/napatech/bin/FpgaImageStatus
```

To upload, activate and verify a new image not compatible with the current driver

This is the general procedure for uploading, activating and verifying a new image that is not compatible with the current driver (see the relevant driver release note for information about FPGA compatibility).

Step	Action
1	Load the current driver by using the **load_driver** script.
2	Upload the new image to the flash memory by using the **UpdateImage** tool (see "3.3.1 UpdateImage" on page 18).
3	Unload the current driver and change the primary image by using the **unload_driver** script (see "3.3.2 unload_driver" on page 19). The server is rebooted.
4	Install the new driver using the **install.sh** command.
5	Load the driver by using the **load_driver** script.
6	Get status information about the images in the flash memory and the active FPGA image by using the **FpgaImageStatus** tool (see "3.3.4 FpgaImageStatus" on page 21).

Command example

This is an example of a command sequence for uploading, activating and verifying a new image that is compatible with the current driver.

```
/opt/napatech/bin/load_driver.sh
/opt/napatech/bin/UpdateImage -img
  /images/200-0008-05_0_070524_1547.img -ini nn -opt Program
/opt/napatech/bin/unload_driver.sh ChangePrimaryImage all
```

(Wait for reboot.)

```
/nt_driver_linux_x.y.z/scripts/sh install.sh
/opt/napatech/bin/load_driver.sh
/opt/napatech/bin/FpgaImageStatus
```

3.3 Tools and Scripts

In this section

This section contains these subsections:

3.3.1 UpdateImage

The UpdateImage tool

The **UpdateImage** tool uploads a new FPGA image and, for NT20E2, possibly new firmware images to the flash memory of an adapter. The driver ensures that the image is supported by the adapter and performs checksum calculations to ensure that the image file is correct. When the image has been uploaded, the driver compares the contents of the flash with the image file to verify that they are identical.

Syntax

The syntax for **UpdateImage** is as shown below.

```
UpdateImage -img <filename> [-ini <initials>] -opt <option> [-help]
            [-adapter <adapter number>]
```

Mandatory parameters

This table explains the mandatory parameters.

Parameter	Description
`-img <filename>`	Is the **.img** file or, for NT20E2, the **.ntimg** file to upload.
`-opt <option>`	Determines whether to upload or verify the image. `<option>` can take these values: • `Program` Uploads the image. • `Verify` Compares the specified image file with the primary image, unless a new image has been uploaded and the system has not been rebooted yet. In this case the specified image file is compared to the secondary image, which has just been updated.

Optional parameters

This table explains the optional parameters.

Parameter	Description
`-help`	Displays a help message describing the tool including syntax and parameters.

Parameter	Description
`-adapter <adapter number>`	Specifies the adapter that the tool is to operate on. `<adapter number>` can take the values `0`, `1`, `2`, ... specifying adapter number 0, 1, 2, ... **Note:** If no adapter number is specified, and more than one adapter is present, the tool queries for an adapter number.
`-ini <initials>`	Is the initials of the individual who uploads the image.

Command example

This is an example of how to execute **UpdateImage**.

```
/opt/napatech/bin/UpdateImage -img
  /images/200-0008-05_0_070524_1547.img -ini nn -opt Program
```

3.3.2 unload_driver

The unload_driver script

The **unload_driver** script unloads the driver and optionally changes the primary image. If the **unload_driver** script is executed without any parameters, it simply unloads the driver. With the **ChangePrimaryImage** parameter, the script also forces the CPLD/AVR to load the secondary image from the flash memory to the FPGA using the **ChangePrimaryImage** tool (see "3.3.3 ChangePrimaryImage" on page 20), and reboots the server. This is useful when a new image has been uploaded to the adapter.

Syntax

The syntax is as shown below.

```
unload_driver.sh
```

or

```
unload_driver.sh ChangePrimaryImage [<adapter number>]
```

Parameters

This table explains the parameters.

Parameter	Description
`ChangePrimaryImage`	Changes a primary image.

Parameter	Description
`<adapter number>`	Specifies the adapter in question. `<adapter number>` can take the values `0`, `1`, `2`, ... or `all` specifying that the primary image is to be changed on adapter number 0, 1, 2, ... or on all adapters. **Note:** If no adapter number is specified, and more than one adapter is present, the tool queries for an adapter number.

Command example

This is an example of how to execute **unload_driver**.

```
/opt/napatech/bin/unload_driver.sh ChangePrimaryImage all
```

Changing the primary image

When the **unload_driver** script is executed with the **ChangePrimaryImage** parameter, it goes through these stages to change the primary image:

Stage	Description
1	The script calls the **ChangePrimaryImage** tool (see "3.3.3 ChangePrimaryImage" on page 20). This informs the driver to change the image when the driver is unloaded.
2	The script unloads the driver. The CPLD/AVR will begin to load the new image from the flash memory to the FPGA 500 ms after the driver has been unloaded. After this point the adapter will not be visible on the PCI until the server is booted.
3	The script reboots the server. This stage is necessary for the image change to take effect. **Note:** Do not power down the server during this step. A warm boot is required to activate the new image.

3.3.3 ChangePrimaryImage

Restriction

The **ChangePrimaryImage** tool is only used indirectly through the **unload_driver** script (see "3.3.2 unload_driver" on page 19).

The ChangePrimaryImage tool

The **ChangePrimaryImage** tool changes the primary image by forcing the CPLD/AVR to load the secondary image from the flash memory to the FPGA. This is useful when a new image has been uploaded to the adapter.

Note: The driver must be unloaded and the server rebooted afterwards for the image change to take effect. Do not power down the server during reboot. A warm boot is required to activate the new image.

Syntax

The syntax is as shown below.

```
ChangePrimaryImage [-adapter <adapter number>] [-all]
```

Parameters

This table explains the parameters.

Parameter	Description
`-adapter <adapter number>`	Specifies the adapter that the tool is to operate on. `<adapter number>` can take the values `0`, `1`, `2`, ... specifying adapter number 0, 1, 2, ... **Note:** If no adapter number is specified, and more than one adapter is present, the tool queries for an adapter number.
`-all`	Specifies that the primary image is to be changed on all adapters. **Note:** The `-all` parameter overwrites the `-adapter <adapter number>` parameter.

Command example

This is an example of how to execute **ChangePrimaryImage**.

```
ChangePrimaryImage -adapter 0
```

3.3.4 FpgaImageStatus

The FpgaImageStatus tool

The **FpgaImageStatus** tool displays the FPGA image status, that is information about the images uploaded to the flash memory and about which image the FPGA is currently using. It can be used to verify that the new image is running as expected.

Syntax

The syntax for **FpgaImageStatus** is as shown below.

```
FpgaImageStatus [-help] [-adapter <adapter number>]
```

Optional parameters

This table explains the optional parameters.

Parameter	Description
`-help`	Displays a help message describing the tool including syntax and parameters.
`-adapter <adapter number>`	Specifies the adapter that the tool is to operate on. `<adapter number>` can take the values `0`, `1`, `2`, ... specifying adapter number 0, 1, 2, ... **Note:** If no adapter number is specified, and more than one adapter is present, the tool queries for an adapter number.

Command example

This is an example of how to execute **FpgaImageStatus**.

```
/opt/napatech/bin/FpgaImageStatus
```

Output example

This is an example of an output from the **FpgaImageStatus** tool, after a new image has been uploaded and activated by the **unload_driver** script (see "3.3.2 unload_driver" on page 19).

```
FpgaImageStatus (v. 4.21.B - 2010-09-09-12-46-25)
=========================================================

NIC image boot status
---------------------------------------------------------
Image 0 (Primary) loaded, no special events Primary image = 0

Fpga image status in flash
---------------------------------------------------------
Image Number     : 0
Image type       : 1
version          : 41.05
Created by       : DEFAULT
Build by         : DEFAULT
Source filename  : 200-9220-41-05-1023-100812-0822.hex
Source date      : Thu Aug 12 10:55:45 2010
Note             : IMAGE BUILD: 200-9220-41-05-1023-100812-0822
Downloaded by    : bune
Filename         : 200-9220-41-05-1023-100812-0822_0.img
Download date    : Fri Aug 20 18:27:59 2010
```

```
Image state     : Image is OK

Image Number    : 1
Image type      : 1
version         : 41.05
Created by      : DEFAULT
Build by        : DEFAULT
Source filename : 200-9220-41-05-1023-100812-0822.hex
Source date     : Thu Aug 12 10:55:45 2010
Note            : IMAGE BUILD: 200-9220-41-05-1023-100812-0822
Downloaded by   : bune
Filename        : 200-9220-41-05-1023-100812-0822_1.img
Download date   : Fri Aug 20 18:30:30 2010

Image state     : Image is OK

Main board AVR image status
-----------------------------------------------------------
Image Number    : 2
Image type      : AVR Front Image
version         : 06i
Source filename : ntmainb1e2-v0.6i.hex
Downloaded by   : ke
Download date   : Wed Sep  1 23:43:12 2010

Front board AVR image status
-----------------------------------------------------------
Image Number    : 3
Image type      : AVR Main Image
version         : 06i
Source filename : ntfront20b1-v0.6i.hex
Downloaded by   : ke
Download date   : Wed Sep  1 23:43:12 2010
```

3.3.5 DriverLog

The DriverLog tool

The **DriverLog** tool retrieves information from the driver log.

Syntax

The syntax for **DriverLog** is as shown below.

```
DriverLog -mask <64bit mask> [-help] [-adapter <adapter number>]
```

```
[-wait <0|1>] [-clear <0|1>]
```

Mandatory parameters

This table explains the mandatory parameters.

Parameter	Description
`-mask<64bit mask>`	Is a bitmask specifying which information is dumped from the driver log. The bits in `<64bit mask>` have the following meaning: • Bit 0 Specifies information about errors. • Bit 1 Specifies information about warnings. • Bit 2 Specifies general information, such as release information. • Bit 3 Specifies debug information. • Bit 4 Specifies information about the image updating process. • Bit 5 Specifies diagnostics information. • Bit 6 Specifies AVR log information. Example: `0x3` specifies that information about errors and warnings is dumped.

Optional parameters

This table explains the optional parameters.

Parameter	Description
`-help`	Displays a help message describing the tool including syntax and parameters.

Parameter	Description
`-adapter <adapter number>`	Specifies the adapter that the tool is to operate on. `<adapter number>` can take the values `0`, `1`, `2`, ... specifying adapter number 0, 1, 2, ... **Note:** If no adapter number is specified, and more than one adapter is present, the tool queries for an adapter number.
`-wait 1`	Specifies that adapter must wait for the driver log to be updated with new entries.
`-clear 1`	Specifies the driver log must be cleared after having been read.

Command example

This is an example of how to execute **DriverLog**.

```
/opt/napatech/bin/DriverLog -mask=0x8
```

Output example

This is an example of an output from the **DriverLog** tool.

```
DriverLog (v. 4.19.A - 2009-12-17-11-58-47)
================================================================
DriverLog: LogLevel mask: 0x8
------------------+---------+-----------------------------------
------------------+---------+-----------------------------------
Timestamp         | LogType | Log entry
------------------+---------+-----------------------------------
------------------+---------+-----------------------------------
1649D043.00051FBA | #DBG    | NTKI_StopPacketFeed: 0x003D0000
1649D043.00051FC9 | #DBG    | NTKI_DestroyPacketFeed: 0x003D0000
------------------+---------+-----------------------------------
------------------+---------+-----------------------------------
```

3.4 Algorithms for FPGA Maintenance

In this section

This section describes some algorithms for maintenance of the FPGA.

The section contains these subsections:

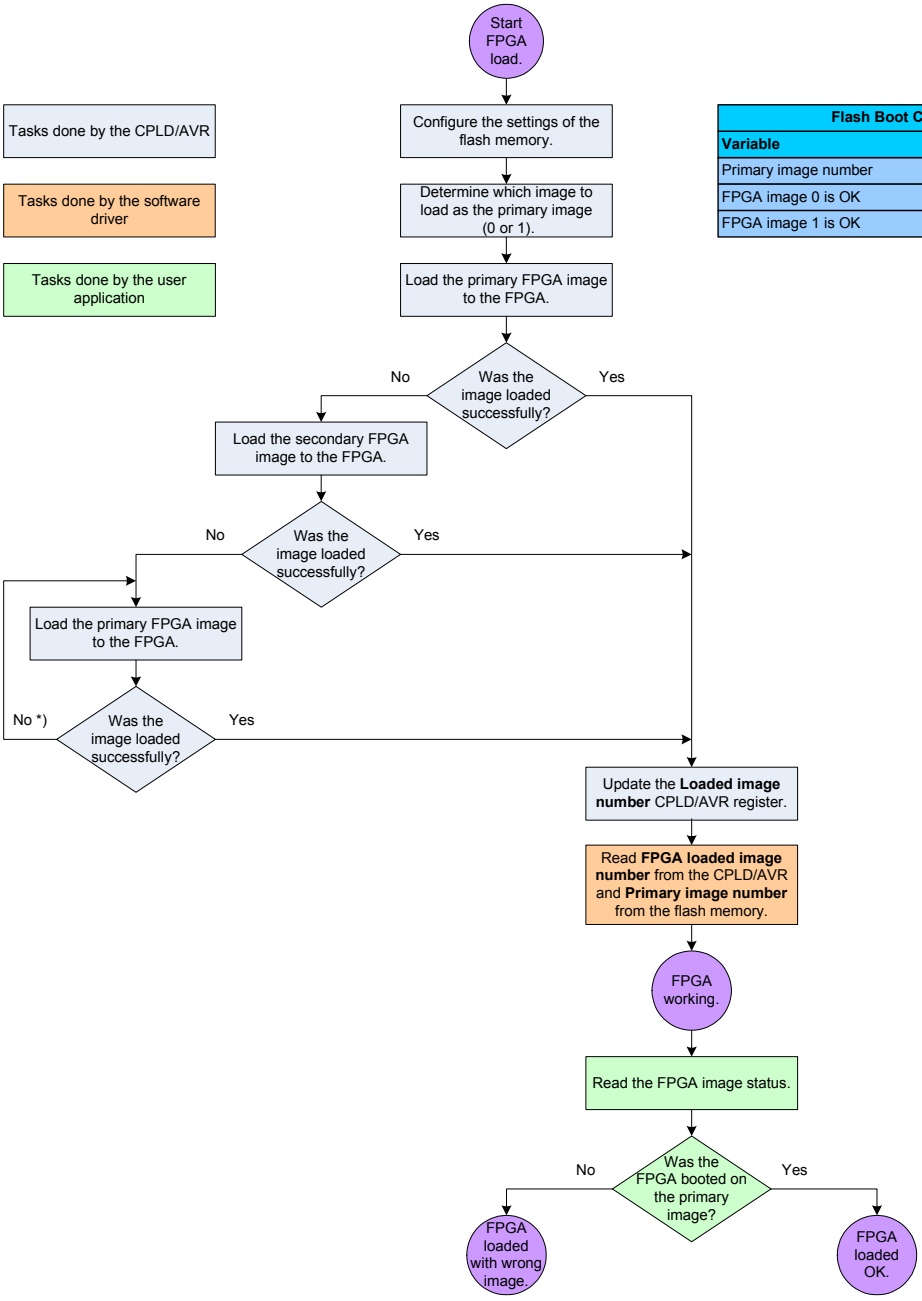
- "3.4.3 Changing the Primary Boot Image" on page 29

3.4.1 Cold Boot of the FPGA

The cold boot algorithm

When the server with the NT adapter is cold-booted, the CPLD/AVR loads the FPGA with the primary FPGA image as specified in the flash configuration. When the driver and the user application are started, the driver verifies that the correct image is loaded. The user application can also perform a verification.

This figure shows the FPGA cold boot algorithm.



Flash Boot Configuration Variables		
Variable	**Used by**	**Updated by**
Primary image number	CPLD/AVR & SW	SW
FPGA image 0 is OK	SW	SW
FPGA image 1 is OK	SW	SW

Note: *) This loop does not occur under normal circumstances. It can only happen in case of a serious HW failure, for instance if the flash memory fails, if it contains 2 invalid images or no images at all.

3.4.2 Writing a new FPGA Image to the Flash Memory

Writing a new image to the flash memory

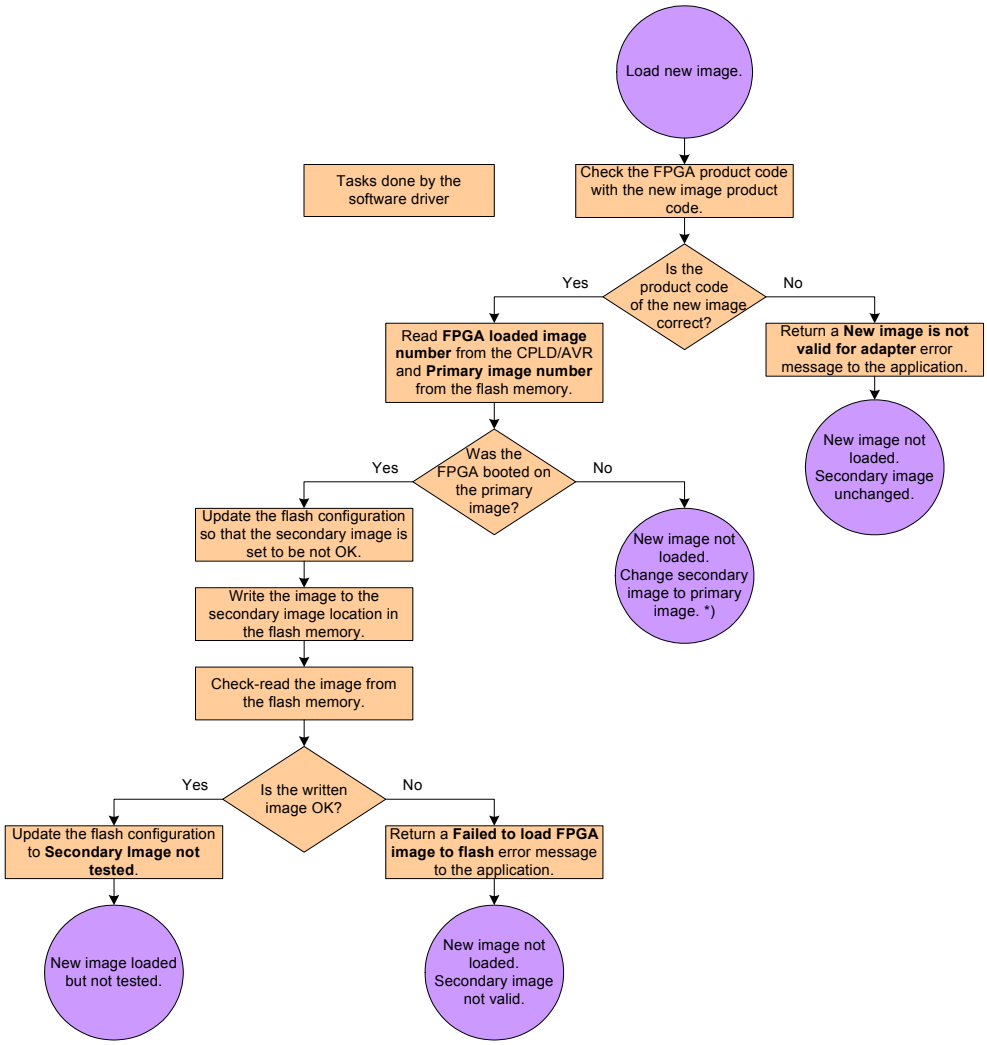
For the driver to write a new FPGA image to the flash memory, the adapter must be booted on the primary image (so that it is asserted that a working primary image is present in the flash memory). The new FPGA image is always written to the secondary FPGA image location in the flash memory.

When the user requests the driver to load a new image, these stages are involved:

Stage	Description
1	The driver verifies that the image to be loaded is valid for the adapter.
2	The driver verifies that the FPGA has been loaded on the primary image.
3	The driver writes the new image to the location of the secondary image in the flash memory.
4	The driver verifies that the image has been written correctly to the flash memory.

The algorithm for writing a new image to the flash memory

This figure shows the algorithm for writing a new FPGA image to the flash memory.



Note: *) See "3.4.3 Changing the Primary Boot Image" on page 29.

3.4.3 Changing the Primary Boot Image

Changing the primary boot image

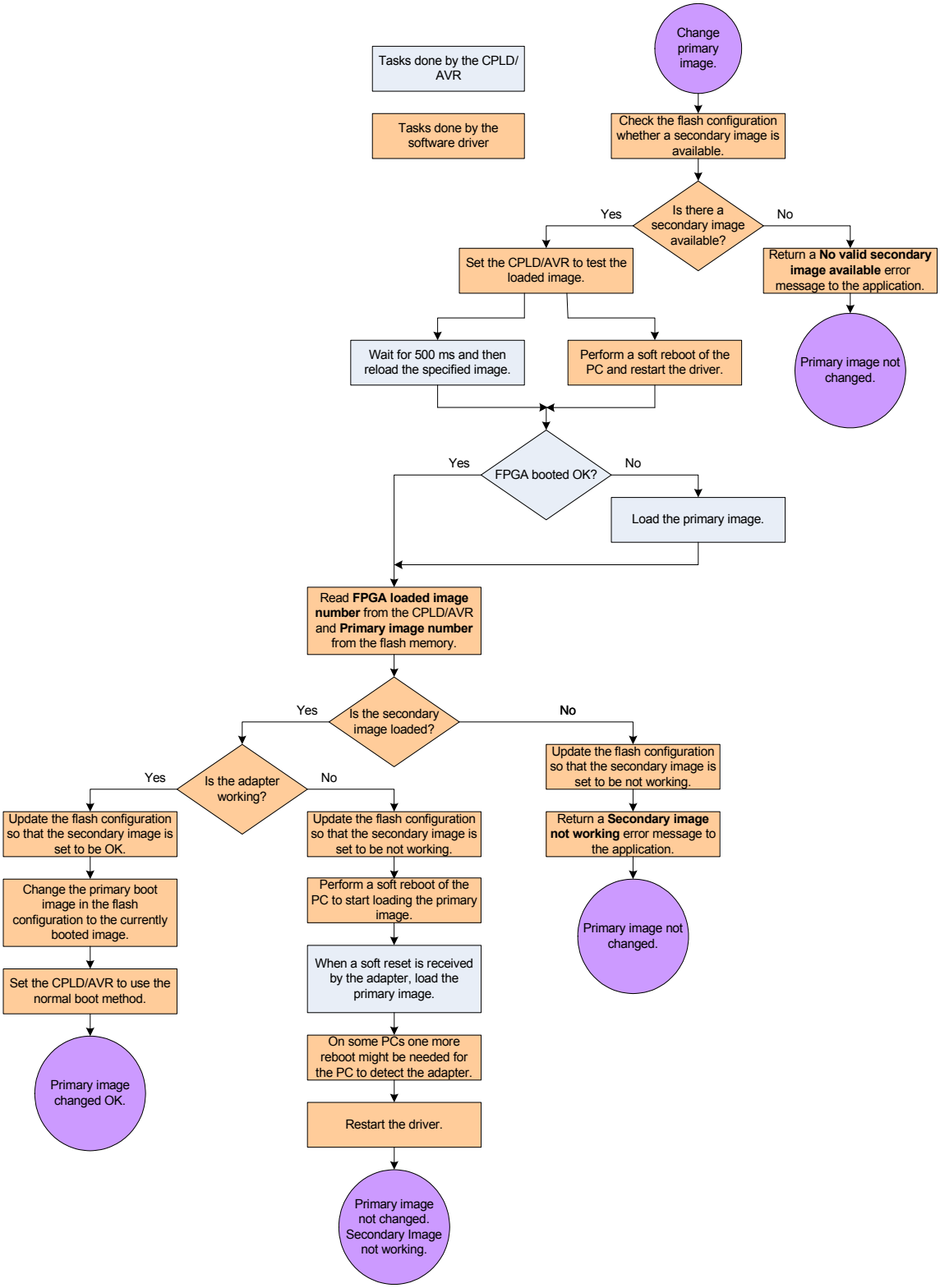
When the user requests the driver to change the primary boot image, these stages are involved:

Stage	Description
1	The driver verifies that a secondary image is present in the flash memory.

Stage	Description
2	The driver requests the CPLD/AVR to test-load the secondary image.
3	The CPLD/AVR loads the FPGA with the secondary FPGA image.
4	The user must perform a warm reboot of the server (for the BIOS to detect the adapter).
5	While the server is rebooting, the CPLD/AVR checks if the FPGA loading was successful, and if not, the CPLD/AVR loads the primary FPGA image. This stage is completed before the BIOS starts to search for PCI adapters.
6	The driver is reloaded and checks if the FPGA has been booted on the secondary image.
7	If the primary image has been loaded, the driver marks the secondary image in the flash memory as not working.
8	If the secondary image has been loaded, the driver checks if basic access to the adapter is possible.
9	If the secondary image is working, the driver updates the flash configuration, so that the secondary image becomes the primary image (and the primary image becomes the secondary image).
10	If the secondary image is not working: • The driver marks the secondary image in the flash as not working. • The user must perform a warm boot. • The CPLD/AVR reloads the FPGA with the primary image when it detects the reset signal. • If the BIOS performs the search for PCI adapters before the primary FPGA image is completely loaded, the server must undergo a warm reboot one more time for the adapter to be detected by the BIOS. • The driver is loaded.

The algorithm for changing the primary boot image

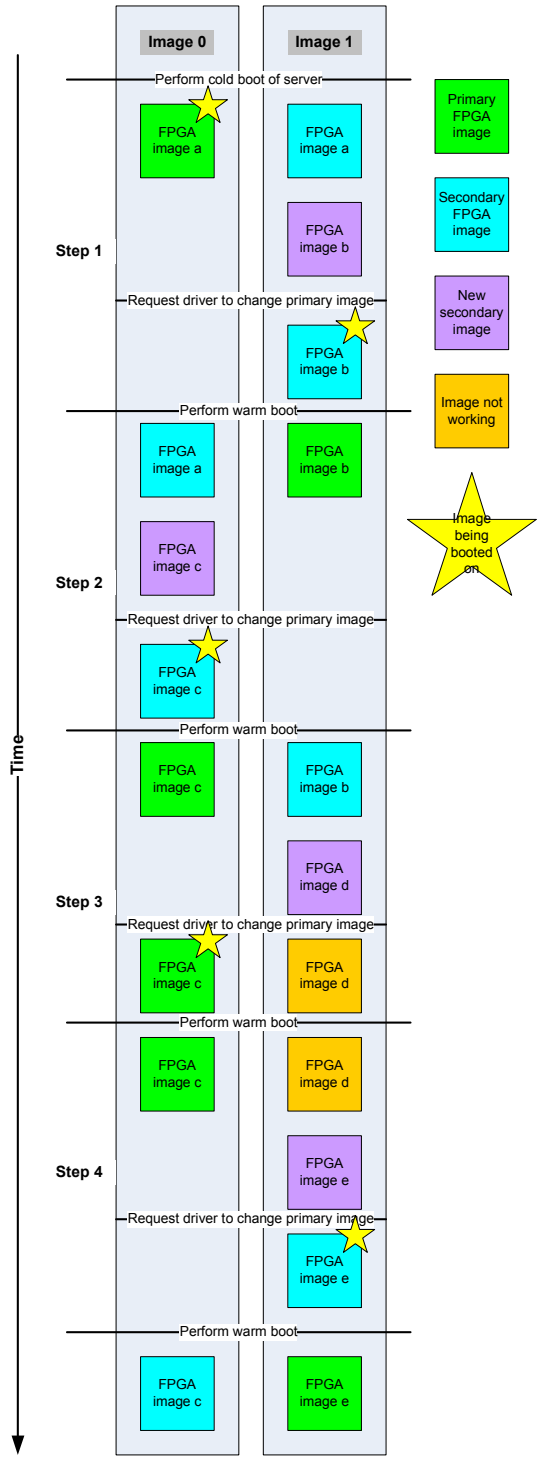
This figure shows the algorithm for changing the primary boot image.



3.5 History Example of Changing FPGA Images

FPGA image load history example

This figure shows an example of uploading a number of FPGA images to an NT adapter.



This table explains the 4 steps in the figure.

Step	Description
1	a) The primary and the secondary images are the same. This is, for instance, the case when the adapter has just been manufactured. b) The adapter is booted on the primary image (FPGA image a). c) The secondary image is updated with FPGA image b. d) The driver is requested to change the primary image. e) The FPGA is loaded with the secondary image, and the loading is successful. f) The server undergoes a warm boot, and the driver is started. g) FPGA image b is working so the driver changes FPGA image b to be the primary image.
2	a) The secondary image is updated with FPGA image c. b) The driver is requested to change the primary image. c) The FPGA is loaded with the secondary image, and the loading is successful. d) The server undergoes a warm boot, and the driver is started. e) FPGA image c is working so the driver changes FPGA image c to be the primary image.
3	a) The secondary image is updated with FPGA image d. b) The driver is requested to change the primary image. c) The FPGA is loaded with the secondary image, but the loading is not successful. d) The FPGA is loaded with the primary image. e) The server undergoes a warm boot, and the driver is started. f) FPGA image d is marked as not working.
4	a) The secondary image is updated with FPGA image e. b) The driver is requested to change the primary image. c) The FPGA is loaded with the secondary image, and the loading is successful. d) The server undergoes a warm boot, and the driver is started. e) FPGA image e is working so the driver changes FPGA image e to be the primary image.

4 Flash Memory Images on the NTBPE Bypass Adapters

In this chapter

This chapter describes how to update the firmware image of an NTBPE bypass adapter.

The chapter contains these sections:

4.1 Overview

Firmware images

The main headlines for the firmware images are:

- An NTBPE adapter has one firmware image in the flash memory.
- It is possible for the customer to update the firmware image in the flash memory.

Tool for firmware maintenance

The **UpdateImage** tool is used for firmware maintenance. It is located in the */opt/napatech/bin* directory. The tool is described in detail in "4.2.2 UpdateImage" on page 36.

4.2 Uploading Firmware Images

In this section

This section contains these subsections:

4.2.1 Procedures

To upload a new image

This is the general procedure for uploading a new image.

Step	Action
1	Push the **FW upgrade** switch located on the adapter PCB away from the front plate.
2	Load the driver by using the **load_driver** script.
3	Upload the new image to the flash memory by using the **UpdateImage** tool (see "4.2.2 UpdateImage" on page 36).

Step	Action
4	Push the **FW upgrade** switch towards the front plate.
5	Run the **AdapterInfo** tool (see Ref. 3 on page 6) to get status information about the running firmware version.

Command example

This is an example of a command sequence for uploading a new image.

```
/opt/napatech/bin/load_driver.sh

/opt/napatech/bin/UpdateImage -img
  /images/202-9001-01-01-090623-1621.bin -opt Program

/opt/napatech/bin/AdapterInfo
```

4.2.2 UpdateImage

The UpdateImage tool

The **UpdateImage** tool uploads a new firmware image to the flash memory of an adapter. When the image has been uploaded, the driver compares the contents of the flash with the image file to verify that they are identical.

Syntax

The syntax for **UpdateImage** is as shown below.

```
UpdateImage -img <filename> [-ini <initials>] -opt <option> [-help]
            [-adapter <adapter number>]
```

Mandatory parameters

This table explains the mandatory parameters.

Parameter	Description
`-img <filename>`	Is the **.bin** file to upload.
`-opt <option>`	Determines whether to upload or verify the image. `<option>` can take these values: • `Program` Uploads the image. • `Verify` Compares the specified image file with the image uploaded to the flash memory.

Optional parameters

This table explains the optional parameters.

Parameter	Description
`-help`	Displays a help message describing the tool including syntax and parameters.
`-adapter <adapter number>`	Specifies the adapter that the tool is to operate on. `<adapter number>` can take these values: • `0` Specifies adapter number 0. • `1` Specifies adapter number 1. • `2` Specifies adapter number 2. • `3` Specifies adapter number 3. **Note:** If no adapter number is specified, and more than one adapter is present, the tool queries for an adapter number.
`-ini <initials>`	Not used.

Command example

This is an example of how to execute **UpdateImage**.

```
/opt/napatech/bin/UpdateImage -img
  /images/202-9001-01-01-090623-1621.bin -opt Program
```

Index

Numerics

A

B

C

D

F

T

U

V

W