



Total Performance. Total Control.™

Netronome Flow Driver

Version 1.3

Migration Guide

**Product code
070-00001-006**

Netronome Flow Driver: Migration Guide

Copyright © 2006-2008 Netronome Systems, Inc.

COPYRIGHT

No part of this publication or documentation accompanying this Product may be reproduced in any form or by any means or used to make any derivative work by any means including but not limited to by translation, transformation or adaptation without permission from Netronome Systems, Inc., as stipulated by the United States Copyright Act of 1976. Contents are subject to change without prior notice.

WARRANTY

Netronome warrants that any media on which this documentation is provided will be free from defects in materials and workmanship under normal use for a period of ninety (90) days from the date of shipment. If a defect in any such media should occur during this 90-day period, the media may be returned to Netronome for a replacement.

NETRONOME DOES NOT WARRANT THAT THE DOCUMENTATION SHALL BE ERROR-FREE. THIS LIMITED WARRANTY SHALL NOT APPLY IF THE DOCUMENTATION OR MEDIA HAS BEEN (I) ALTERED OR MODIFIED; (II) SUBJECTED TO NEGLIGENCE, COMPUTER OR ELECTRICAL MALFUNCTION; OR (III) USED, ADJUSTED, OR INSTALLED OTHER THAN IN ACCORDANCE WITH INSTRUCTIONS FURNISHED BY NETRONOME OR IN AN ENVIRONMENT OTHER THAN THAT INTENDED OR RECOMMENDED BY NETRONOME.

EXCEPT FOR WARRANTIES SPECIFICALLY STATED IN THIS SECTION, NETRONOME HEREBY DISCLAIMS ALL EXPRESS OR IMPLIED WARRANTIES OF ANY KIND, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE.

Some jurisdictions do not allow the exclusion of implied warranties, so the above exclusion may not apply to some users of this documentation. This limited warranty gives users of this documentation specific legal rights, and users of this documentation may also have other rights which vary from jurisdiction to jurisdiction.

LIABILITY

Regardless of the form of any claim or action, Netronome's total liability to any user of this documentation for all occurrences combined, for claims, costs, damages or liability based on any cause whatsoever and arising from or in connection with this documentation shall not exceed the purchase price (without interest) paid by such user.

IN NO EVENT SHALL NETRONOME OR ANYONE ELSE WHO HAS BEEN INVOLVED IN THE CREATION, PRODUCTION, OR DELIVERY OF THE DOCUMENTATION BE LIABLE FOR ANY LOSS OF DATA, LOSS OF PROFITS OR LOSS OF USE OF THE DOCUMENTATION OR FOR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL, EXEMPLARY, PUNITIVE, MULTIPLE OR OTHER DAMAGES, ARISING FROM OR IN CONNECTION WITH THE DOCUMENTATION EVEN IF NETRONOME HAS BEEN MADE AWARE OF THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT SHALL NETRONOME OR ANYONE ELSE WHO HAS BEEN INVOLVED IN THE CREATION, PRODUCTION, OR DELIVERY OF THE DOCUMENTATION BE LIABLE TO ANYONE FOR ANY CLAIMS, COSTS, DAMAGES OR LIABILITIES CAUSED BY IMPROPER USE OF THE DOCUMENTATION OR USE WHERE ANY PARTY HAS SUBSTITUTED PROCEDURES NOT SPECIFIED BY NETRONOME.

Revision History

Date	Revision	Description
25 March 2008	006	Support for NFD version 1.3.
8 January 2008	005	Support for NFD version 1.2.3.
20 September 2007	004	Support for NFD version 1.2.1.
1 July 2007	003	Support for NFD version 1.2.
1 March 2007	002	New document layout.
15 September 2006	001	Initial Release.

Table of Contents

1. Introduction	5
1.1. Scope	5
1.2. Related Documents	5
1.3. Terminology	5
2. Migration from NFD v1.2.x to NFD v1.3	6
2.1. General Notes	6
2.2. Host API changes	6
3. Migration from NFD v1.2 to NFD v1.2.3	7
3.1. General Notes	7
3.1.1. NFE firmware upgrade	7
3.1.2. Host API changes	7
4. Migration from NFD v1.1 to NFD v1.2	8
4.1. General Notes	8
4.1.1. NFE firmware upgrade	8
4.1.2. Message Signaled Interrupts	8
4.1.3. Driver Behavior	9
4.1.4. User Message Type	9
4.1.5. Directory Structure Changes	9
4.2. Porting Microcode Applications	10
4.3. Porting Host Platform Code	10
4.4. Porting XScale Code	10
5. Migration from NFD v1.0 to NFD v1.1	11
5.1. General Notes	11
5.2. Porting Microcode Applications	11
5.2.1. Build Switches	11
5.2.2. Dynamic RAM Usage	12
5.2.3. Static RAM Rings	12
5.2.4. Buffer offsets	13
5.2.5. Include Files	14
5.3. Porting Host Platform Code	14
5.3.1. Configuration changes	14
5.3.2. Error code changes	14
5.3.3. Application Programming Interface changes	15
5.4. Porting XScale Code	15
5.4.1. Configuration changes	15
5.4.2. Application Programming Interface changes	15
6. Technical Support	16

1. Introduction

1.1 Scope

This document is intended for developers who require to migrate applications from an older Netronome Flow Driver (NFD) revision to NFD v1.3. Special consideration is given to Application Programming Interface and functional changes.

Chapter 2 provides information to migrate applications from NFD v1.2.x to NFD v1.3, Chapter 3 provides information to migrate applications from NFD v1.2 to NFD v1.2.3, Chapter 4 provides information to migrate applications from NFD v1.1 to NFD v1.2, and Chapter 5 provides information to migrate applications from NFD v1.0 to NFD v1.1.

1.2 Related Documents

Descriptive Name	Filename
Netronome Flow Driver Programmer's Reference Manual	NFD_PRM.pdf
Netronome Flow Driver User's Guide	NFD_UG.pdf

1.3 Terminology

The following terms and abbreviations are used in this document:

Acronym	Description
API	Application Programming Interface
DMA	Direct Memory Access
DRAM	Dynamic RAM
ME	Microengine
MSI	Message Signaled Interrupts
NFD	Netronome Flow Driver
NFE	Netronome Flow Engine
PCIe	PCI Express
SRAM	Static RAM
UP	Uniprocessor
XS	XScale processor

2. Migration from NFD v1.2.x to NFD v1.3

2.1 General Notes

The packaging of the Netronome Flow Driver software has changed as of v1.3. All the software is now contained within the source tarball. The Netronome Flow Engine tarball no longer exists and its content can now be found in the `nfe_bsp` subdirectory in the source tarball. The Netronome Flow Driver source tarball no longer includes any firmware or firmware upgrade mechanism and as such the `nfd-1.2.3-upgrade` script that was packaged with NFD-1.2.3 is deprecated and no longer available. NFE firmware as well as the firmware upgrade mechanism is now packaged separately.

2.2 Host API changes

The `recv_callback_t` callback used in the Host Platform Kernelmode Messaging API has changed from

```
typedef int (*recv_callback_t)(ns_msg_data_h, void*)
```

to

```
typedef int (*recv_callback_t)(ns_msg_data_h, unsigned int nfe_dest_id, void*)
```

adding the NFE destination ID to the callback. All modules that make use of the Host Platform Kernelmode Messaging API to receive messages will have to be updated.

3. Migration from NFD v1.2 to NFD v1.2.3

3.1 General Notes

Some of the new features in Netronome Flow Driver v1.2.3 that may require special action are listed here for convenience. Please refer to the *Netronome Flow Driver Programmer's Reference Manual* and the *Netronome Flow Driver User's Guide* for more detail on each of the features.

3.1.1 NFE firmware upgrade

The NFE platform firmware should be upgraded such that it can be used with NFD-1.2.3. After installing the host NFD software, the NFE platform's firmware should be upgraded as described in the *Netronome Flow Driver User's Guide*. The upgrade can most simply be done by executing the following application:

```
/opt/netronome/nfd/bin/nfd-1.2.x-upgrade
```

The above command automatically determines how to upgrade the NFE devices present in the system. Please refer to the *Netronome Flow Driver User's Guide* for more detail regarding firmware upgrading.



Warning

The upgrade process should not be interrupted as this may render the NFE platform unusable. Once the upgrade is complete, a message will be printed indicating that the host platform should be completely shutdown to allow the firmware changes to take effect.

3.1.2 Host API changes

The `nfe_device_state()` API function has been added to the Netronome Flow Engine platform information API (`libnfe`) to obtain device state information such as the NPU temperature. Refer to the *Netronome Flow Driver Programmer's Reference Manual* for further details.

4. Migration from NFD v1.1 to NFD v1.2

4.1 General Notes

Some of the new features in Netronome Flow Driver v1.2 that may require special action are listed here for convenience. Please refer to the *Netronome Flow Driver Programmer's Reference Manual* and the *Netronome Flow Driver User's Guide* for more detail on each of the features.

4.1.1 NFE firmware upgrade

The NFE platform firmware should be upgraded such that it can be used with NFD-1.2. After installing the host NFD software as described in the *Netronome Flow Driver User's Guide*, the NFE platform's firmware can be upgraded by doing the following:

For NFEi8000 devices, do:

```
/opt/netronome/nfd/bin/nfeops.py -f /opt/netronome/nfd/bin/nfd12-upgrade-i8000.ini -s
```

For NFEi4000 devices, do:

```
/opt/netronome/nfd/bin/nfeops.py -f /opt/netronome/nfd/bin/nfd12-upgrade-i4000.ini -s
```



Warning

The upgrade process should not be interrupted as this may render the NFE platform unusable. Once the upgrade is complete, a message will be printed indicating that the host system should be completely shutdown to allow the firmware changes to take effect.

4.1.2 Message Signaled Interrupts

The Message Signaled Interrupts (MSI) mechanism is used in NFD v1.2 for improved performance and latency. Legacy interrupts are not supported and the NFE message driver will not compile on systems where MSI support has not been enabled.

The `nfemsg` driver requires that MSI support is enabled both at compile-time and runtime. Therefore, the user must ensure that the `CONFIG_PCI_MSI` configuration option is set for the kernel that the `nfemsg` driver will be compiled for.

For uniprocessor (UP) systems, the "Local APIC support on uniprocessors" under the "Processor type and features" section of the kernel configuration must be enabled before the "MSI" option becomes available under the "Bus" section. The user must therefore ensure that the `CONFIG_X86_UP_APIC` and

`CONFIG_X86_LOCAL_APIC` configuration options are set on UP systems to make the `CONFIG_PCI_MSI` option available.

4.1.3 Driver Behavior

Multiple NFE platforms (up to four may be installed in the host) are supported by the host platform messaging driver (`nfemsg`).

The host platform messaging driver initializes the buffer pools for a particular NFE platform upon the first invocation of `ns_msg_init()` for that particular NFE platform. The host platform messaging driver destroys the buffer pools for a particular NFE platform upon the last corresponding invocation of `ns_msg_shutdown()` for that particular NFE platform.

It is therefore no longer required that any NFE platforms have microcode loaded before the host platform messaging driver is loaded.

4.1.4 User Message Type

A new message type has been added to the messaging API. The new message type does not have an associated payload buffer and can therefore carry 15 DWords of user-defined data instead of the current 10 DWords.

The default behavior of the host messaging API is to only make use of the current data message type. This ensures backwards compatibility with existing applications. The `NS_MSG_ALLOC_USER_TYPE` flag can be set to enable the new user message type to be used on the host. The XScale and microengine applications must be written to support the user message type before enabling it on the host.

4.1.5 Directory Structure Changes

The directory structures changed in both the NFD source package and the NFD installation directory.

For the host platform API's, the header, library and binary files are located respectively in the `include`, `lib` and `bin` directories in the installation directory. Headers for host kernel module API development are located in the `/src/include` directories.

For NFE platform API's, the header, library and binary files are located respectively in the `include`, `lib` and `bin` directories in the `armv5teb-hardhat-linux/opt/netronome/nfd` directory. Headers for kernel module API development are located in the `armv5teb-hardhat-linux/opt/netronome/nfd/src/include` directory. These directories are relative to the installation directory.

The default installation directory is `/opt/netronome/nfd`.

The installation process of the NFD is described in Section 4 of the *Netronome Flow Driver User's Guide*.

4.2 Porting Microcode Applications

NFD v1.1 microcode applications do not require any change unless the new message type is enabled as described in Section 4.1.4.

4.3 Porting Host Platform Code

NFD v1.1 host applications do not require any change unless the new message type is enabled as described in Section 4.1.4.

4.4 Porting XScale Code

With NFD v1.1, the XScale messaging API made exclusive use of the only message type structure (`ns_msg_data_t`) available. With the additional "user" message type declared in NFD v1.2, all API functions should now use the common structure `ns_msg_t` which must be casted, depending on the use, to one of the message type structures `ns_msg_data_t` or `ns_msg_user_t`. Refer to the example at `ns_msg_rcv` in Section 7.4.8 of the *Netronome Flow Driver Programmer's Reference Manual*.

Note that the host API will only send user messages if enabled by the `NS_MSG_ALLOC_USER_TYPE` flag.

5. Migration from NFD v1.0 to NFD v1.1

5.1 General Notes

During initialization, the NFD v1.1 host platform kernel module reads and interprets a new NFE platform device ID presented by the platform via the PCI region. The ID contains state information which the kernel module uses to intelligently determine if it should continue with the initialization process or not.

The possible states are described in the following table:

Table 5.1. Different possible states of the NFE Platform

Status	Description
Correct	Correct device ID, after which the platform is initialized.
Incompatible	Device ID that indicates an incompatible microengine <code>pcie_driver</code> building block. The message "Device # is using an incompatible microengine PCIe driver" is displayed.
Not ready	Device ID that indicates that the platform is not ready. This implies that the microengine <code>pcie_driver</code> building block has not been loaded nor started.
Invalid	Invalid device ID that indicates an unknown platform or microengine failure.

5.2 Porting Microcode Applications

Changes made to the PCIe microengine drivers include:

- Changed build switches.
- Reduced number of symbols need to be patched when making use of import variables.
- Renamed identifiers, so as to prevent possible name clashes.

5.2.1 Build Switches

The compile time build switches have changed. Review all the switches in Table 6.1 of the *Netronome Flow Driver Programmer's Reference Manual*.

5.2.2 Dynamic RAM Usage

5.2.2.1 Single DRAM Buffer

The PCIe Messaging uses a single DRAM base address in NFD v1.1. This significantly reduces the number of symbols that need to be patched when import variables are used. All other DRAM blocks are specified as offsets from the single base address `PCIE_MSG_DRAM_BASE`. Refer to the example application `pcie_msg_echo`.

5.2.2.2 Renamed DRAM Buffers

The following definitions have been renamed:

Table 5.2. Renamed DRAM Buffers

NFD v1.0	NFD v1.1
<code>PCIE_FL_DRAM_BASE_ADDR</code>	<code>PCIE_FREELIST_DRAM_OFFSET</code> (offset from <code>PCIE_MSG_DRAM_BASE</code>)
<code>PCIE_FL_DRAM_SIZE_BYTES</code>	<code>PCIE_FREELIST_DRAM_SIZE</code>
<code>PCIE_LOOPBACK_FIFO_BASE_ADDR</code>	<code>PCIE_LOOPBACK_DRAM_OFFSET</code> (offset from <code>PCIE_MSG_DRAM_BASE</code>)
<code>PCIE_LOOPBACK_FIFO_SIZE_BYTES</code>	<code>PCIE_LOOPBACK_DRAM_SIZE</code>
<code><FIFO NAME>_BASE_ADDR</code>	<code><FIFO NAME>_DRAM_OFFSET</code> (offset from <code>PCIE_MSG_DRAM_BASE</code>)
<code><FIFO NAME>_FIFO_SIZE</code>	<code><FIFO NAME>_DRAM_SIZE</code>

5.2.2.3 Renamed Free List

The following definitions have been renamed:

Table 5.3. Renamed Free List

NFD v1.0	NFD v1.1
<code>PCIE_READ_FL_ID</code>	<code>PCIE_READ_FREE_LIST</code>
<code>PCIE_WRITE_FL_ID</code>	<code>PCIE_WRITE_FREE_LIST</code>

5.2.3 Static RAM Rings

5.2.3.1 Renamed Rings

The SRAM rings used to send messages and DMA descriptors to the driver microengines have been renamed as follows:

Table 5.4. Renamed Rings

NFD v1.0	NFD v1.1
MSG_WRITE_RING	PCIE_MSG_PUT_RING
DMA_READ_RING	PCIE_DMA_READ_RING
DMA_WRITE_RING	PCIE_DMA_WRITE_RING

5.2.3.2 Message Ring

The following definitions have been removed:

- PCIEMSG_PUT_HW_RING_TYPE
- PCIEMSG_PUT_HW_RING_ADDR

Use PCIE_MSG_PUT_RING to send messages from the NFE platform. It must be an SRAM ring.

5.2.3.3 Parameters

The parameters required to specify an SRAM ring have changed. Refer to the *Netronome Flow Driver Programmer's Reference Manual* for more information.

5.2.4 Buffer offsets

In NFD v1.0:

- The target host platform offset (NFE platform→host platform transfer) was specified by PCIE_IA_BUFFER_OFFSET, which defaulted to 256 if not defined.
- The target NFE platform offset was always 384.
- The target offset is incremented by 0.7 if the source is not 8-Byte aligned.

In NFD v1.1:

- If PCIE_IXP_BUFFER_OFFSET is defined, this will be used as the target offset in the NFE platform destination buffer, else the offset in the host platform source buffer will be used.
- If PCIE_IA_BUFFER_OFFSET is defined, this will be used as the target offset in the host platform destination buffer, else the offset in the NFE platform source buffer will be used.
- The target offset is incremented by 0.7 if the source is not 8-Byte aligned.

Refer to the section on DMA Transfer Alignment in the *Netronome Flow Driver Programmer's Reference Manual* for more information.

5.2.5 Include Files

The PCIe ME driver does not include the file `d1_system.h` directly. The programmer must include the file `d1_system.h` from the application specific `pcie_msg_user.h`. In NFD v1.0 all the message fifo definitions were specified in `pcie_msg_user.h`. All memory definitions can now be specified in `d1_system.h`, or in an application specific memory header file that is included by `d1_system.h`.

5.3 Porting Host Platform Code

5.3.1 Configuration changes

Configuration of a host platform destination is governed by the `ns_msg_config_t` structure that is passed to `ns_msg_init()`. The following changes have been made to this structure:

- The `K2U_fifoslots` and `U2K_fifoslots` fields no longer exist as these parameters are no longer adjustable by applications. All destination fifo sizes are fixed. The number of slots is defined by `DEST_FIFO_SLOTS` in `ns_msg_defs.h`.
- The `max_concurrent_datahandles` field no longer exists as this parameter is no longer adjustable by applications. Message handles are now drawn from a global pool configured by the host platform Linux Kernel driver (`nfei8000.ko`).
- The `version` field has been added to indicate the library version. It should not directly be set by the application - `ns_msg_config_init()` should be used instead.
- The `flags` field has been added to associate configuration flags with a particular destination upon initialization. Refer to the *Netronome Flow Driver Programmer's Reference Manual* for detailed information regarding the available configuration flags.

5.3.2 Error code changes

The following error codes were added (Refer to the *Netronome Flow Driver Programmer's Reference Manual* for further details.):

- `NS_MSG_VERSION_MISMATCH`
- `NS_MSG_OUT_OF_DESTS`
- `NS_MSG_MMAP_FAILED`
- `NS_MSG_OUT_OF_BUFFERS`

5.3.3 Application Programming Interface changes

The following changes were made to the host platform messaging API:

- `ns_msg_set_data_length()` no longer checks whether the given length is less than or equal to 1664 (2048-384) since the `pcie_driver` building block will no longer necessarily transfer payloads to an offset of 384 inside an *Intel packet buffer*. Refer to Section 5.2.4.
- `ns_msg_get_userdata_ptr()` now returns `unsigned int*` to reflect that the NFE platform treats userdata as 32 bit words.
- `ns_msg_get_dest_id()` was added to obtain the destination ID of a destination that was automatically assigned.

5.4 Porting XScale Code

5.4.1 Configuration changes

`ns_msg_config_init()`, the `ns_msg_config_t`, `fifo_t` structures and the `ring_t` enum have been removed from the XScale messaging API. All error codes related to configuration have also been removed. Instead all configuration is done by the XScale `pcie_driver` kernel module. Refer to the *Neutronome Flow Driver Programmer's Reference Manual* for further details.

5.4.2 Application Programming Interface changes

The following changes were made to the XScale messaging API:

- `MAX_SUPPORTED_IXP_FIFOS` has been renamed to `MAX_IXP_FIFOS`.
- `MAX_SUPPORTED_IA_FIFOS` has been renamed to `MAX_IA_FIFOS`.
- `ns_msg_init()` no longer takes a pointer to `ns_msg_config_t`. Refer to Section 5.4.1.
- `ns_msg_shutdown()` now returns a `ns_msg_return_t` where it previously returned nothing.

6. Technical Support

To obtain additional information, or to provide feedback, please email <support@netronome.com> or contact the nearest **Netronome Systems** technical support representative.