NPS-400 EZdk Release Notes

SW Rev 18.0400.00

NOTE:
THIS HARDWARE, SOFTWARE OR TEST SUITE PRODUCT ("PRODUCT(S)") AND ITS RELATED DOCUMENTATION ARE PROVIDED BY MELLANOX TECHNOLOGIES "AS-IS" WITH ALL FAULTS OF ANY KIND AND SOLELY FOR THE PURPOSE OF AIDING THE CUSTOMER IN TESTING APPLICATIONS THAT USE THE PRODUCTS IN DESIGNATED SOLUTIONS. THE CUSTOMER'S MANUFACTURING TEST ENVIRONMENT HAS NOT MET THE STANDARDS SET BY MELLANOX TECHNOLOGIES TO FULLY QUALIFY THE PRODUCT(S) AND/OR THE SYSTEM USING IT. THEREFORE, MELLANOX TECHNOLOGIES CANNOT AND DOES NOT GUARANTEE OR WARRANT THAT THE PRODUCTS WILL OPERATE WITH THE HIGHEST QUALITY. ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT ARE DISCLAIMED. IN NO EVENT SHALL MELLANOX BE LIABLE TO CUSTOMER OR ANY THIRD PARTIES FOR ANY DIRECT, INDIRECT, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES OF ANY KIND (INCLUDING, BUT NOT LIMITED TO, PAYMENT FOR PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY FROM THE USE OF THE PRODUCT(S) AND RELATED DOCUMENTATION EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Mellanox Technologies
350 Oakmead Parkway Suite 100
Sunnyvale, CA 94085
U.S.A.
www.mellanox.com
Tel: (408) 970-3400
Fax: (408) 970-3403

Table of Contents

1 Introduction

The EZdk software development kit is a comprehensive set of design and runtime tools for developing both data-plane and control-plane applications for NPS-400 devices.

This release provides post-silicon support for the NPS-400 network processor. It fully supports real chip bring-up and application development.

This document provides additional information related to the release not contained in the product's manuals. It is recommended that you read this document before using the product. These release notes outline the modifications employed since EZdk release 18.0300.00.

2 This Release

This release of the EZdk software development kit for NPS-400 devices is aimed at allowing customers to develop NPS data-plane and control plane applications. This release includes the following main components:

- EZldk – Linux® development kit enabling customers to develop and build the Linux kernel, file system and GNU tool-chain for the NPS device.
- EZdp – Data-plane service library providing the APIs used to develop data-plane applications on the NPS device.
- EZcp – Control-plane service library providing the APIs used to develop control-plane applications for the NPS device.
- Sample applications demonstrating how to develop applications for the NPS-400 devices using EZdk.

This release represents work in progress, and is subject to change in future releases.

3 System Requirements

EZdk-nps is compliant with RedHat6.x distribution or any derivative such as CentOS6.x. Any other environment, such other versions of the distribution or different distributions are not guaranteed to work.

NOTE: The GCC toolchain is provided with EZdk as precompiled binaries. Since it is dynamically linked to standard libraries of distribution and built upon RedHat6.x using the toolchain, it may not work on other environments due to missing libraries.

4 Documentation

The following documentation can be found in the doc folder of the EZdk installation:

- *EZdk Installation Guide* – provides information on EZdk installation procedure and contents.
- *NPS-400 Developer's Guide* – describes each of the EZdk control plane and data plane libraries and their respective APIs. Describes the various Control Plane Library (EZcp) components and how they can be ported to various platforms and used to develop control-plane applications for NPS-based products.
- *NPS-400 Demo Application* – sample data plane and control plane application designed to enable developers to quickly begin writing an application for an NPS-400 system.

Additional Documents

These documents contain supplementary information and are not supplied in the EZdk installation:

- *EZdk Release Notes* (this document) – provides additional information related to the release that is not contained in the product's manuals.
- *NPS-400 Architectural Specifications* – overview of the architecture, feature set and functionality of the NPS-400 network processor.
- *NPS-400 Programming Manual* – overview of the software programming model and concepts for the NPS-400 network processor.
- *NPS-400 L23Qos Application* – describes a simple Layer 2 switching and Layer 3 routing sample application provided on the NPS-400 network processor.
- *NPS-400 IPsec Gateway Application* – describes a sample IPsec gateway application provided on the NPS-400 network processor.

5 Changes and New Features

Following are the main new features and functionality introduced in this release.

5.1 General Changes and New Features

Supports PCI offload to enhance control plane search structures and TCAM updates/writes to the NPS-400 device. The acceleration is achieved by using PCI out-bound DMA operations instead of using the configuration bridge and offloading the operations to the data plane. External hashes and external tables are supported by PCI offload in version 18.0400. See “PCI Offload” in the *NPS Developer’s Guide* for more information.

5.2 CPE Changes and New Features

Table 1: CPE Changes and New Features

No.	Description
EZcp	
1.	Added direct table support for 4 byte result in IMEM.
2.	Allocate all hash pools in CP library automatically. Disregard the hash memory manager parameters for signature and result index pools.
3.	Use NPS memory pools for static hashes.
4.	Added hash size output parameter for the CP’s EZapiStruct_MemUsage function. This allows users to configure PMU timers according to the hash main table size.
5.	PCI offload feature added, allow offloading of the PCI to CFGB interface, for search and other operations.
6.	Loopback interfaces allow maximum of 4 loopback interfaces per side, always separate different COS to different queues.
7.	Management interface, API removed, enabled internally by U-boot.
8.	PMU timers allow changing timers period at run time.
9.	System info API removed, enabled by default.
10.	Protocol decoder allows changing protocol decoder configuration at run time.
11.	ICU classification allows disabling classification. In which case ICU outputs the default COS and CLASS and hash of 0.
12.	Handle mutual configuration of reduced & muxed modes.
13.	Added MBIST status check by CP. Halt in case of MBIST error.
14.	Auto negotiation – added AN and TX training services.
15.	IF engine, Ethernet MAC and Interlaken MAC APIs – removed redundant fields.
EZdev	
16.	EZdev_ControlChannelCmd – Added option to get the DMA buffer address.
17.	Allow access to the DMA buffer, used for PCI offload. EZdev_ReadValDMAChannel - Read a 32-bit value from a channel device DMA buffer. EZdev_WriteValDMAChannel - Write a 32-bit value to a channel device DMA buffer. EZdev_ReadBufDMAChannel - Read a burst of bytes from a channel device DMA buffer.

No.	Description
	EZdev_WriteBufDMAChannel - Write a burst of bytes to a channel device DMA buffer.
UIO	
18.	Allocate DMA buffer of 16KB, used for PCI offload feature.
EZspy	
19.	Performance tool added tool for debugging NPS performance based on spy monitors.
20.	Code style changes.

5.3 DPE Changes and New Features

Table 2: DPE Changes and New Features

No.	Description
EZdp	
21.	Added DP low level function ezdp_prm_get_hash_slots to retrieve the hash main table size.
22.	Changed the DP ezdp_scan_hash_slot function to return an EINVAL error in case the given slot number exceeds the hash main table size.
EZframe	
23.	ezframe_load_buf – Added the EZFRAME_ASSUME_NOT_FIRST_BUFFER optimization flag, indicates that the buffer to load is not the first buffer.
24.	ezframe_store_buf – Added the EZFRAME_ASSUME_NOT_FIRST_BUFFER optimization flag, indicates that the buffer to store is not the first buffer. Added the EZFRAME_BUF_DATA_SIZE_NOT_CHANGED optimization flag, indicates that buffer data size to store has not changed.
25.	ezframe_change_queue – Added the EZFRAME_WAIT_FOR_SEQ_NUMBER flag, indicates a request to wait until frame is updated with sequence number of target PMU queue. When sequence number is disabled on target queue and this flag is set, application will get stuck. Added the EZFRAME_ALLOW_JOBS_REORDER flag, indicates that job reordering in PQ is allowed.
IPSECLib	
26.	New IPSECLib library was added to EZdk SDK. Performance optimization up to 20K tunnels.

5.4 LDK Changes and New Features

Table 3: LDK Changes and New Features

No.	Description
27.	Upgraded Linux version from 4.2 to 4.8.
28.	Configure the active dbg_lan, by using U-boot environment variable "dbg_lan_side". For example : setenv dbg_lan_side east

No.	Description
29.	In order to read kernel log after core 0 was halted for some reason, use netconsole on NPS side and rsyslog on the remote system. To configure the NPS write the following in the uboot command line: setenv krn_args netconsole=@<nps_ip>/eth0,10514@<reciever_ip> saveenv To configure the receiver side: 1. Make sure that the rsyslog service is installed and running. Installed : [root@nps06 ~]# rpm -qa | grep rsyslog rsyslog-5.8.10-8.el6.x86_64 Running : [root@nps06 ~]# ps -A | grep rsyslog 16822 ? 00:00:00 rsyslogd 2. In the rsyslog configuration file (/etc/rsyslog.conf) add the following lines in order to capture UDP packets: # Provides UDP syslog reception $ModLoad imudp $UDPServerRun 10514 *.* /var/log/remote_nps.log The kernel log will be saved to /var/log/remote_nps.log

5.5 Sample Application Changes and New Features

Table 4: Sample Application Changes and New Features

No.	Description
IPsec Gateway Application	
30.	Application was updated to work with the new IPSECLib library

Important Note: Existing NPS Application and Target projects may need to be re-generated to conform to the above changes.

6 Bug Fixes

Following are the known issues fixed in this release.

6.1 CPE Bug Fixes

Table 5: CPE Fixed Bugs

No.	Description
EZcp	
1.	Fixed false indication of memory error on table delete of result size bigger than 16 bytes.
2.	Fixed support for multiple iterators on cached hash structures.
3.	When performing an operation to a cached hash structure, i.e. add entry, the operation was performed for the cache even when it failed on the table in chip memory, thus generating an inconsistency between the cache and the table in the chip memory. This was fixed by performing the operation on the cache only when the operation on the table written to the chip memory has succeeded.
4.	Fixed a possible array overflow in table delete with result size larger than 32 bytes.
EZspy	
5.	Print pipe function: (1) PMU section: added initialization of counters and (2) TND section: fixed print typos.
6.	Spy monitors -- fixed calculation errors.
7.	Data flow counters function -- fixed segmentation fault (close file when it is not open).
8.	BMU checker – fixed return status; passed was returned even when an error occurred.

6.2 DPE Bug Fixes

Table 6: DPE Fixed Bugs

No.	Description
EZdp	
9.	The application load frame data to CMEM buffer (256 byte) alignment to buffer pointer even though RX channel configured with a header offset which was different than zero. That means there is no "free" space in buffer for the client's data changes. It has been fixed and data loads to buffer according to header offset configuration.
PCI Offload	
10.	Fixed PCI offload bug that caused the DP application to get stuck when using application ID 0.

7 Known Limitations

7.1 CPE Known Limitations

Table 7: CPE Known Limitations

No.	Description
EZcp	
1.	For some combinations of keys, the spread of the hash function used by the hash search structure is not ideal, which can lead to signature pages and result pages exhaustion. Implement a workaround by setting the EZapiStruct_HashMemMngParams uiSigPagePercent and uiResPagePercent values to 100.

7.2 DPE Known Limitations

Table 8: DPE Known Limitations

No.	Description
EZdp	
2.	ezdp_load_data_from_sum_addr(_async) and ezdp_store_data_to_sum_addr(_async) APIs: 256-byte values in cannot be immediate in these APIs and the user should use non-immediate values.
EZframe	
3.	The ezframe_send_to_if and ezframe_send_to_tm APIs fail to update the header offset to be optimized. EZFRAME_DONT_OPTIMIZE_HEADER_OFFSET is not functional.

7.3 LDK Known Limitations

Table 9: LDK Known Limitations

No.	Description
4.	The following items are not supported/operational on the NPS-400 real chip in this release: • Ethernet: 10/40GBASE-KR, Synchronous Ethernet. • GCI devices. • Operation of the NPS management LAN at 100 Mbps is not supported (only operational at 1 Gbps).

7.4 Sample Application Known Limitations

Table 10: Sample Application Known Limitations

No.	Description
L23QoS Sample Application	
5.	L23 launcher scripts are not sending EZware to background. In run_ezware() add & at the end of the command line.

8 Changes and New Features History

8.1 EZdk Update Release 18.0300.00

8.1.1 New Features Enabled in Update Release 18.0300.00

General Changes and New Features

1. EZdk product number was changed for this release. In the previous release it was 2.1a Open. This release it is 18.0300.00, where 18 is the EZdk OpenNPU product number (i.e. fixed number) and the other digits represent the software release number that changes with each release.

CPE Changes and New Features

EZcp

2. EZapiChannel_PCIInfo was removed.
3. Number of logical IDs available to the user was reduced from 256 to 255. Logical ID 255 is used by the DP application (EZframe).
4. Removed PCI advance NIC capabilities support.
5. Removed error queue support.
6. Allow PMU sequence numbering in PMU queue basis; removed per RX channel sequence numbering.
7. Allow loopback interfaces rate of up to 100Gbps.
8. Force buffer allocation on IMEM when RX allocation profile mode is IMEM only.
9. System Info is enabled by default.
10. Optimized the load partition command by enhancing the reset of search related NPS memory.
11. Optimized search multiple NPS operations by supporting channel interleaving in memory operations.
12. Increased the number of supported search structures from 128 to 256.
13. Removed the possibility for a recycle race condition on DP updates of dynamic hash. This includes not allowing sharing of a hash designated index pool for more than one structure. Also using the same pool for both signature and entry page is no longer allowed.
14. Added support for direct table with an 8 byte result in NPS internal memory.

EZcp2

15. New Config to Python (EZc2p) library in the /cpe/c2p/ folder allows users to export their system control plane configuration into a Python file.

EZspy

16. Monitor simple for monitoring single defined field at memory or register.

DPE Changes and New Features

EZdp

17. Aging pool APIs were removed.
18. PCI Express interface APIs were removed.

LDK Changes and New Features

19. The 5 pct auxiliary registers of the core can be accessed through an ioctl call through the dpl module. The options for the ioctl calls of the read and write to those registers are macros defined in the header dpl.h located under ldk/sources/linux-nps/arch/arc/include/uapi/asm/.
20. Once a dp process starts to run on a cpu, no other dp processes can run on the same cpu until the previous process has died. In case another process tries to perform ezdp_init_local on a cpu

that already runs a dp process, the opening of the dpl module will fail and EBUSY will be stored in errno.

21. The NPS now supports only one physical function and one lane for the PCI. Therefore, the configuration of the PCI was changed. The U-Boot environment variable pci_config was changed to pci_gen. The number of physical functions and lanes for the PCI cannot be changed, and the value that pci_gen should get is a number between 1 and 3 to indicate the PCI generation currently used by the NPS.
    For example, changing the PCI generation to Gen 3 will be done like this (from the u-boot environment):
    nps# setenv pci_gen 3
    nps# saveenv
22. The PCI and PCIe capabilities have been reduced to the minimum required by the NPS.
    The capabilities that are necessary:
    - Power Management version 3 (PCI capability with ID of 0x1)
    - MSI (PCI capability with ID 0x5)
    - Express Endpoint (PCI capability with ID 0x10)
    - Advanced Error Reporting (PCIe capability with ID 0x1)
    - Vendor Specific Information (PCIe capability with ID 0xb)
    - Secondary PCIe Extended (PCIe capability with ID 0x19)

    The capabilities that were removed:
    - MSI-X (PCI capability)
    - Vital Product Data (PCI capability)
    - Virtual Channel (PCIe capability)
    - Alternative Routing-ID Interpretation (PCIe capability)
    - Single Root I/O Virtualization (PCIe capability)
    - Address Translation Service (PCIe capability)
    - Latency Tolerance Reporting (PCIe capability)
23. The logs of the kernel describing the cause of an exception have been improved.
    When an invalid CMEM access (e.g. beyond private or shared CMEM sizes) is performed, the following information will be printed by the kernel: virtual address that was accessed, sizes of the private and shared CMEM and data-cache sections and more information regarding the limits of this access.
    When an invalid FMT access is performed (e.g. writing to a read-only memory space), the following information will be printed by the kernel: virtual address of the access, number of the FMT slot and the type of the mapped memory space that was accessed (e.g. x4 cluster code).
    Note: In the two cases above, a SEGFAULT will be sent to the process that performed the access.
    When an invalid memory space access is performed (e.g. write to a disabled memory space), the following information will be printed by the kernel: number of the memory space that was accessed, type of the memory space that was accessed (e.g. x1 cluster data) and offset of the access. This is in addition to the printings regarding the error code and transaction code that were added for this case in the 2.1a release.
    Note: In the case above, the core will finish printing another general information regarding ARC generic and auxiliary registers and then will halt. The information will be available, if not through the application, through the dmesg command from the shell.
24. The format of defining krn_possible_cpus and krn_present_cpus was changed in order to enable the selection of any group of cpus.
    Example of the new format:
    Defining 0-7,16-23,32-39,.....,3064-3071,3072-4095 will be done in the following way :
    0-3071%16=0-7,3072-4095.

25. Added EZmodpost tool to assist in identifying EMEM mapped functions/variables used in fast path code that could reduce performance. Located in /tools/EZmodpost and /ldk/toolchain/bin/EZmodpost.
26. The watchdog timer driver has been added to the NPS.
    The timer is reset every 3 seconds by default by a background process running on cpu 0.
    To change the frequency of the reset, one needs to change the configuration file located in ldk/sources/buildroot/package/busybox/Config.in and make busybox.
    The timeout of the watchdog device may be changed by an ioctl call to the /dev/watchdog device. The timeout ranges from 1 to 5 seconds.
    The watchdog timer operates on a counter that decreases every cycle.
    When the watchdog counter reaches 0 the first time, nothing happens, but when it reaches 0 the second time, an interrupt is sent from the watchdog device.
    The interrupt from the watchdog device reaches the GIM block, which sends it outside the NPS through the INT_OUT line.

Sample Application Changes and New Features

IPsec Gateway Application

27. Refactoring of sample code.
28. AES_128_GCM_128 optimized to reach 180 G
29. HMAC_SHA1_96 AES_CBC_128 optimized to reach 145 G
30. HMAC_SHA256_128 AES_CBC_256 optimized to reach 160 G

8.1.2 Bug Fixes in 18.0300.00

CPE Bug Fixes

EZcp

31. When initialization of shared CMEM fails, printed error might be wrong.
32. In the CP library memory partition, a user defined internal memory space with index number 63 was not initialized.
33. When deleting all entries from a cached hash structure, the entries were not deleted from the NPS memory as well.

EZspy

34. Bug fix at print pipe: Wrongfully reported busy threads at cluster due to missing init of counters. "CLUSTER_1: busy threads due to non-posted (read) command (out of 16) = 0x3 (3)."

LDK Bug Fixes

35. Problematic sequence:
    - run demo_dp -run_cpus 16-4095
    - killall -15 demo_dp
    - run demo_dp -run_cpus 16-4095
    *** oom killer is invoked ***
    Solution:
    1. Add to kernel parameters, through bootargs environment variable - “slub_max_order=0” , This will overrun the kernel default value 3.
    2. Shrink all slabs after each killall: For i in ‘ls /sys/kernel/slab/*/shrink’; do echo 1 > $i ; done
36. Irrelevant values from configuration file were exported to Python and could cause errors in the control plane application.