1. Device installation

Some motherboards have certain PCI Express slots connected directly to the processor and others connected to the chipset; for optimal performance, install the ExaNIC in a slot directly connected to the processor.

Install SFP+ modules into the ExaNIC as required. There are no limitations on the type or manufacturer of SFP+ modules that can be used, we supply and recommend Finisar modules.

2. Software installation

There are a number of different ways of installing the ExaNIC software: using the pre-built packages provided by Exablaze, building your own packages, or installing from source.

Pre-built packages

Pre-built packages are available for some Linux distributions. The easiest way to access these is to add the Exablaze repository to your yum or apt sources, or you can manually download the desired packages directly from the repository.

The yum repository containing .rpm packages for Redhat/CentOS is located at:

http://exablaze.com/downloads/yum/

The apt repository containing .deb packages for Debian/Ubuntu is located at:

http://exablaze.com/downloads/apt/

In both cases there is a README.txt file which explains how to add the repository, what distributions are currently supported and what packages are available.

Build your own packages

You can also build your own packages using the packaging provided by Exablaze.

To build your own .rpm packages, download the latest source rpm from http://exablaze.com/downloads/yum/redhat/el6/SRPMS/ and run `rpmbuild --rebuild exanic-*.src.rpm` . Packages will be output to `~/rpmbuild/RPMS`.

To build your own .deb packages, download and unpack the source from the support website as per the first step below, then run `dpkg-buildpackage` from within the source directory. Packages will be output to the parent directory.

Install from source

If binary packages are not available for your distribution, the ExaNIC software is also easy to build and install from source. To start, download the ExaNIC source package from the support website and unpack it.

```
$ tar xzf exanic-1.5.0.tar.gz
$ cd exanic-1.5.0
```

Running the following commands should build and install the driver, library and utilities:

```
$ make
$ sudo make install
```

By default, the library will be installed to `/usr/local/lib/libexanic.a`, include files to `/usr/local/include/exanic` and utilities to `/usr/local/bin/exanic-*`. (You can add a `PREFIX=` argument to the `make install` command to use a prefix other than `/usr/local`.) Additionally the driver (`exanic.ko`) will be installed into `/lib/modules/`uname -r`/extra`.

If the driver fails to build, ensure that you have the kernel headers package for your kernel installed (this is typically named `kernel-devel` [RedHat/CentOS] or `linux-headers` [Debian/Ubuntu]).

3. After installation

Assuming installation completed successfully, load the driver and verify that the `exanic-config` utility works:

```
$ sudo modprobe exanic
$ sudo exanic-config exanic0
```

If either of these commands fail, run `dmesg` and check for errors.

The `exanic-config` output will show the Linux interfaces corresponding to each port of the ExaNIC. You can verify that packets are being received by bringing up an interface and running `tcpdump`. For example:

```
$ sudo ifconfig eth7 up
$ sudo tcpdump -n -i eth7
```

If desired you can also build and run the benchmarking utilities (see section 11).

4. Configuration

Since the ExaNIC appears to Linux as a normal network card, most configuration can be performed through the normal Linux mechanisms. For example, in Redhat-based distributions, configuration is generally performed using files in the

`/etc/sysconfig/network-scripts` directory. Alternatively, the interface can be configured temporarily by issuing a command such as `ifconfig eth7 192.168.1.100`.

If the interface is being used through the low-level userspace API (libexanic) only, it is sufficient to bring the interface up without assigning an IP address. This can be done through either `ifconfig` (e.g. `ifconfig eth7 up`) or `exanic-config` (e.g. `exanic-config exanic0:0 up`).

5. The exanic-config utility

The exanic-config utility can be used to inspect diagnostic information about the card and SFP modules. For example:

```
$ exanic-config
Device exanic0:
  Hardware type: ExaNIC X4
  Board ID: 0x02
 Temperature: 53.2 C   VCCint: 1.00 V   VCCaux: 1.81 V
 Fan speed: 6987 RPM
 Function: network interface
 Firmware date: Thu Nov 28 20:54:56 2013
 Bridging: off
 Port 0:
   Interface: eth7
   Port speed: 10000 Mbps
    Port status: enabled, SFP present, signal detected, link active
    Mirroring: off
    Promiscuous mode: off
    Bypass-only mode: off
 MAC address: 64:3f:5f:01:13:18
 RX packets: 6  ignored: 0  error: 0
 TX packets: 6
 ...

$ exanic-config exanic0:0 sfp status
Device exanic0 port 0 SFP status:
 Vendor: FINISAR CORP.    PN: FTLX8571D3BCL     rev: A
                          SN: ALF0QE7          date: 111006
 Wavelength: 850 nm
 Nominal bit rate: 10300 Mbps
 Rx power: -2.5 dBm (0.56 mW)
 Tx power: -1.9 dBm (0.65 mW)
 Temperature: 33.8 C
```

The `exanic-config` utility will either accept the Linux interface name (e.g. `eth7`) or the device name of the ExaNIC device (e.g. `exanic0` for the first ExaNIC in the system by PCI ID, and `exanic0:0` for the first port of that card).

6. Setting port speed

Normally the ExaNIC operates in 10 Gigabit Ethernet (only) mode. To set a port to 1 Gigabit speed, use `ethtool` as follows:

```
$ ethtool -s eth7 speed 1000
```

Similarly, to change a port back to 10 Gigabit Ethernet mode, use:

```
$ ethtool -s eth7 speed 10000
```

1 Gigabit Ethernet support requires firmware dated 20140206 or later.

7. Port mirroring and bridging

To configure card-specific functionality such as port mirroring and bridging functionality, you can use either `ethtool` or `exanic-config`. With ethtool, use `ethtool --show-priv-flags and ethtool --set-priv-flags`. For example:

```
$ sudo ethtool --show-priv-flags eth7
Private flags for eth7:
bypass_only: off
mirror_rx: off
mirror_tx: off
bridging: off

$ sudo ethtool --set-priv-flags eth7 bridging on
```

The same settings can also be configured through `exanic-config`:

```
$ sudo exanic-config exanic0 bridging on
exanic0: bridging on (ports 0 and 1)

$ sudo exanic-config exanic0:0 mirror-rx on
exanic0:0: RX mirroring on
```

Enabling receive port mirroring (`mirror_rx`) on a port causes the received data to be copied to port 3 (the last port; in this document the ports are numbered from 0). Enabling transmit port mirroring (`mirror_tx`) on a port causes the transmitted data to be copied to port 3. Enabling bridging (on either ports 0 or 1) causes any data received on port 0 that is not destined for the host to be forwarded to port 1, and vice versa. Port mirroring and bridging settings are stored in non-volatile memory and are persistent across reboot and power off.

8. Bypass-only mode

If the interface is being used through the userspace API only and kernel CPU load on high traffic connections is a concern, the `bypass-only` setting in `exanic-config` (or `bypass_only` in `ethtool`) can be used to detach the kernel driver from the interface. This

means that the kernel driver will not receive any packets. However, tools such as `tcpdump` will not function correctly while this is enabled. Additionally, this setting is not currently persistent across reboots so, if required, should be added to post-up scripts for the interface.

9. Setting permissions

If the ExaNIC is to be used in kernel bypass mode, it may be useful to set permissions on the ExaNIC device files (`/dev/exanic0`, etc, and the `/dev/exasock` device file used by ExaNIC Sockets) so that they can be accessed by certain non-root applications. Permissions can be set with `chown/chgrp/chmod` as normal, however this will not persist across reboots. A persistent configuration can be achieved by appropriate configuration of the `udev` daemon. For example, to force the device files to be accessible by a group called `exanic`, create a `/etc/udev/rules.d/exanic.rules` file as follows:

```
KERNEL=="exanic*", GROUP="exanic", MODE="0660"
KERNEL=="exasock", GROUP="exanic", MODE="0660"
```

In the current software release the ExaNIC device files also expose the device control registers so only trusted users should be given access to the ExaNIC in this way.

10. Clock synchronisation

To use the ExaNIC for timestamping, it is necessary to run `exanic-clock-sync`, a daemon that manages the ExaNIC's clock synchronization. Each argument to `exanic-clock-sync` specifies a `<device>:<synchronization-source>` pair, with the synchronization source being one of `host` (host clock), `pps` (pulse-per-second input) or `exanicN` (another ExaNIC device). For example:

```
$ exanic-clock-sync exanic0:host exanic1:exanic0
```

If desired this can be placed in system startup scripts. A sample `init.d` script is provided in `configs/exanic-clock-sync`.

By default `pps` accepts a differential input such as RS-422. For a TTL or RS-232 single-ended input, specify `pps-single-ended` instead of `pps`.

11. Running benchmarking utilities

A number of benchmarking utilities, both for the ExaNIC and for other cards, are located in the perf-test directory provided with the distribution. To build the benchmarks for ExaNIC:

```
$ cd perf-test
$ make exanic
```

The simplest benchmark to run is `exanic_loopback`, which sends packets out one port and receives them on another. Connect a cable from port 0 to port 1 of the device, bring the

respective interfaces up, and run, for example (to obtain 100 samples at 64 byte frame size):

```
$ ./exanic_loopback exanic0 0 1 64 100
```

12. Updating firmware

The firmware version currently running on the ExaNIC card is shown under 'Firmware date' in `exanic-config`. If there is a newer version available on the website, you can update your firmware as follows. First, uncompress the downloaded firmware image:

```
$ gunzip exanic_x4_20131128.fw.gz
```

Now run `exanic-fwupdate`:

```
$ exanic-fwupdate exanic_x4_20131128.fw
```

Note that there are different firmware files for ExaNIC X2 and ExaNIC X4 cards. The utility will prevent you from installing firmware incompatible with your hardware.

13. Uninstallation

If you installed from packages then you should use the appropriate yum or apt command (`sudo yum remove 'exanic*'` or `sudo apt-get remove 'exanic*'`).

If you installed from source the following command should remove everything that was installed by `make install`:

```
$ sudo make uninstall
```

Also remove any files that were manually created while following these instructions (`/etc/udev/rules.d/exanic.rules` and any scripts that run `exanic-clock-sync`).