

# **DYNAMIC ENGINEERING**

150 DuBois St. Suite C Santa Cruz CA 95060

831-457-8891 Fax 831-457-4793

<http://www.dyneng.com>

[sales@dyneng.com](mailto:sales@dyneng.com)

Est. 1988

## **Software User's Guide (Linux)**

# **PCI-NECL2-RTN10**

One-Channel Full Duplex NECL Interface

## PCI-NECL2-RTN10

Dynamic Engineering  
150 DuBois St Suite C  
Santa Cruz, CA 95060  
831-457-8891  
831-457-4793 FAX

©2016-2018 by Dynamic Engineering.  
Other trademarks and registered trademarks are owned by their  
respective manufactures.  
Revised 02/28/18

This document contains information of proprietary interest to Dynamic Engineering. It has been supplied in confidence and the recipient, by accepting this material, agrees that the subject matter will not be copied or reproduced, in whole or in part, nor its contents revealed in any manner or to any person except to meet the purpose for which it was delivered.

Dynamic Engineering has made every effort to ensure that this manual is accurate and complete. Still, the company reserves the right to make improvements or changes in the product described in this document at any time and without notice. Furthermore, Dynamic Engineering assumes no liability arising out of the application or use of the device described herein.

The electronic equipment described herein generates, uses, and can radiate radio frequency energy. Operation of this equipment in a residential area is likely to cause radio interference, in which case the user, at his own expense, will be required to take whatever measures may be required to correct the interference.

Dynamic Engineering's products are not authorized for use as critical components in life support devices or systems without the express written approval of the president of Dynamic Engineering.

Connection of incompatible hardware is likely to cause serious damage.



Product Description .....	4
Software Description .....	4
Installation.....	5
Application Programming model.....	5
Sample application.....	5
Invocation parameters .....	5
Warranty and Repair .....	7
Service Policy .....	8
Out of Warranty Repairs .....	8
For Service Contact: .....	8

## Product Description

The PCI-NECL2-RTN10 board is a one channel, full duplex interface card implementing byte wide receive and transmit port. Further, a 12 bit GPIO port is implemented and supported by the driver.

For a detailed description of the hardware including register definitions, see HW User Manual, PCI-NECL2-RTN10.

## Software Description

The driver supports full duplex operation of the I/O port. Each I/O card has 1 NECL I/O port and 1 GPIO port. Note: RTN10 derives from STE3. The driver retains STE3 naming.

A default configuration is applied when the either port is opened for the first time. These default settings are defined in the driver header file, de\_NeclSte3.h. The default I/O port config settings is named de\_default\_io\_config, the default GPIO port settings is named de\_default\_gpio\_config. These default config parameters can be customized for a particular application, and the driver recompiled. This may eliminate the need for invoking the config ioctl.

Applicable I/O configuration parameters include blocking timeout, SDRAM enable, rx and tx sdram allocations and internal loopback enable (for test purposes). Blocking timeout provides a mechanism to timeout on blocking read operations. Enabling SDRAM in the Rx/Tx path provides more data buffering in the I/O path in addition to the internal FIFOs. SDRAM can be allocated in 1 MB chunks for the Rx/Tx path. The card is populated with 32 MB of SDRAM.

Default I/O configuration is as follows: Blocking timeout on reads = 5 sec. (if opened as blocking), SDRAM enabled, 16 MB SDRAM allocated for Rx, 16 MB SDRAM allocated for Tx, and internal loopback disabled.

GPIO configuration parameters consists of GPIO direction (input or output), and an initial output assignment (if any GPIO are enabled for output). The default configuration is all GPIO set as input

The version of this driver is v1.0.0. The driver has been validated on an i7 Ubuntu server running 3.8.0-44 kernel (64 bit) SMP, and a Red Hat server running RHEL 6.5 (64 bit) SMP kernel version 2.6.32-41.el6.x86\_64



## Installation

- 1) Copy de\_NeclSte3.c and de\_NeclSte3.h to your module build directory. Invoke the system “make.” A makefile for this module has been included in the release tar-ball.
- 2) Copy the resulting de\_NeclSte3.ko module to the target platform/directory.
- 3) Copy the startup script bnm to the target.
- 4) Invoke the script (./bnm), it will create the devices required by the driver and performs an insmod of the module. You may invoke this script from the systems rc.local file as well.

## Application Programming model

After a port is opened, it may be configured for the desired mode of operation via the DE\_CONFIG\_PT ioctl. Both blocking and non-blocking modes of operation are supported. This behavior is set via the standard file flags upon open.

Please see de\_NeclSte3.h for details of the parameters for this and other supported ioctls.

## Sample application

Three sample applications (de\_loApp.c, de\_loAppUni.c de\_loctlApp.c) are provided to demonstrate configuration, ioctl invocation, and I/O in the supported modes.

- 1) Compile the sample application for your platform, the output executable for these examples are dyn\_io and dyn\_ioctl.
  - a. Nominal compilation gcc

```
gcc -Wall -o dyn_io de_loApp.c
gcc -Wall -o dyn_uni de_loAppUni.c
gcc -Wall -o dyn_ioctl de_loctlApp.c
```

The apps should compile without warnings, it is assumed de\_NeclSte3.h is resident in the same directory as the applications for these examples.

## Invocation parameters

**I/O application (ping-pong) invocation is as follows:**

```
./dyn_io master(1=master) port(0-N) frame_len(32 bit words)
```



The first parameter specifies mode of operation (master or slave). The second parameter specifies the port. The third parameter, frame length is specified in 32 bit words. Optionally an iteration count may be specified as the last parameter after frame length.

The app assumes two I/O cards are installed in the system and cabled together properly in order to execute without error. This app validates proper I/O port operation and GPIO port operation. Even number ports are I/O ports, odd are GPIO ports.

For I/O port test execution, one port is selected as the master. The remaining command line parameters must be identical. The “slave” app must be started first for synchronization purposes. First the master posts a write, followed by a read. Upon receipt of data at the slave it is validated, and the same data is transmitted back to the master and validated upon receipt. This sequence is repeated N times unless an error is encountered. The following is an example of I/O test invocation from two different terminal windows:

```
./dyn_io 0 0 2048 /* Slave app, port 0, 2048 LWs */  
/* Start master within 5 seconds of invoking slave app in another  
terminal window */  
./dyn_io 1 2 2048 /* Master app, port 2, 2048 LWs */
```

#### **I/O application (uni-directional) invocation is as follows:**

Uni-directional application is the same as ping pong except the master only transmits, and the slave only receives. Data validation is performed at the slave port. Uni-directional app invocation is as follows:

```
./dyn_uni 0 0 2048 /* Slave app, port 0, 2048 LWs */  
/* Start master within 5 seconds of invoking slave app in another terminal  
window */  
./dyn_uni 1 2 2048 /* Master app, port 2, 2048 LWs */
```

## GPIO validation:

GPIO port operation is validated by selecting ports 1 and 3 utilizing the ping pong app. The slave app is invoked first, it sets all GPIO as output and writes a test pattern to the GPIO port. Upon invoking the master app, it will configure all GPIO as input and verifies correct test pattern is read from the port. Invocation is as follows:

```
./dyn_io 0 1 /* Slave app, port 1 */  
/* Invoke slave within 5 seconds from another terminal window */  
./dyn_io 1 3 /* Slave app, port 3 */
```

## Ioctl application invocation is as follows:

```
./dyn_ioctl
```

A menu will be displayed:  
Enter p(II program)||r(eg ops)|e(xit)

The ioctl application demonstrates pll programming, register R/W/RMW operations.

## Support Contract

Dynamic Drivers are provided AS-IS and sometimes our clients need a little help. Please refer to the support contract page on our website for options about getting help with your driver use and SW development.

<http://www.dyneng.com/TechnicalSupportFromDE.pdf>

## Warranty and Repair

Please refer to the warranty page on our website for the current warranty offered and options.

<http://www.dyneng.com/warranty.html>



## Service Policy

Before returning a product for repair, verify as well as possible that the suspected unit is at fault. Then call the Customer Service Department for a RETURN MATERIAL AUTHORIZATION (RMA) number. Carefully package the unit, in the original shipping carton if this is available, and ship prepaid and insured with the RMA number clearly written on the outside of the package. Include a return address and the telephone number of a technical contact. For out-of-warranty repairs, a purchase order for repair charges must accompany the return. Dynamic Engineering will not be responsible for damages due to improper packaging of returned items. For service on Dynamic Engineering Products not purchased directly from Dynamic Engineering contact your reseller. Products returned to Dynamic Engineering for repair by other than the original customer will be treated as out-of-warranty.

### Out of Warranty Repairs

Software support contracts are available to update, add features, change for different revisions of OS etc. Please contact Dynamic Engineering for these options.

### For Service Contact:

#### Customer Service Department

Dynamic Engineering  
150 DuBois St. Suite C  
Santa Cruz, CA 95060  
831-457-8891  
831-457-4793 fax  
InterNet Address [support@dyneng.com](mailto:support@dyneng.com)

