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
Est. 1988

Software User's Guide (Linux)

PCIe104Diff

One-Channel LVDS Interface

PCIe104Diff

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

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

The PCIe104Diff-OS1 board is a one channel, half-duplex LVDS I/O card.

For a detailed description of the hardware including register definitions, see HW User Manual, PCIe104Diff OS1.

Software Description

The driver supports half-duplex operation of the I/O port. Each I/O card has one I/O port.

A default configuration is applied when the port is opened for the first time. The default settings are defined in the driver header file, de_DiffOs1.h. The default I/O port config setting is named de_default_io_config. The 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 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. Though the Os1 version is populated with SDRAM, it has not been validated with SDRAM enabled. The driver currently always bypasses SDRAM in the rx/tx paths.

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

The version of this driver is v1.0.1. The driver has been validated on an i7 Ubuntu server running 3.8.0-44 kernel (64 bit) SMP.

Note, a PLL read/write ioctl is provided. However, the Os1 uses the on board oscillator rather than the PLL.

Installation

1) Copy de_DiffOs1.c and de_DiffOs1.h to your module build directory. Invoke the system ñmake.ò A makefile for this module has been included in the release tar-ball.
2) Copy the resulting de_DiffOs1.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_DiffOs1.h for details of the parameters for this and other supported ioctls.

Sample application

Three sample applications (de_IoApp.c, de_IoctlApp.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 ï o dyn_io de_IoApp.c
gcc -Wall ï o dyn_ioctl de_IoctlApp.c
The apps should compile without warnings, it is assumed de_DiffOs1.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.

For I/O port test execution, one port is selected as the master. The remaining command line parameters must be identical. The master port must be attached to cable end labeled Tx and the slave port attached to the Rx end. The ñslaveò app must be started first for synchronization purposes. First the master posts a writes. Upon receipt of data at the slave it is validated. 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 */
```

```
./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 r(eg ops)|e(xit)

The Ioctl application demonstrates 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