Release Notes
CCURDSCC (WC-AD3224-DS)

Driver	ccurdscc (WC-AD3224-DS)	Rev 6.3
OS	RedHawk	Rev 6.3
Vendor	Concurrent Computer Corporation	
Hardware	PCIe 32-Channel Delta Sigma Converter Card (CP-AD3224-DS)	
Date	July 29th, 2013	

concurrent
REAL-TIME

This page intentionally left blank

Table of Contents

This page intentionally left blank

1. Introduction

This document assists the user in installing the CCUR-PCIe-DSCC Linux ***ccurdscc*** driver and related software on the RedHawk OS for use with the CCUR-PCIe-DSCC board. The directions in this document supercede all others – they are specific to installing the software on Concurrent Computer Corporation's RedHawk systems. Other information provided as part of this release, when it may contradict Concurrent's directions, should be ignored and Concurrent's directions should prevail.

For additional information on this driver and usage refer to the ***ccurdscc*** man page.

The CCUR-PCIe-DSCC is a 32-channel analog to digital 24-bit delta sigma converter card with a PCI express interface. It is implemented using four Cirrus Logic CS5368 8-channel converters. The PCI interface utilizes a PLX Technology PEX-8311AA PCI-express-to-local bus bridge. There is a Lattice ECP2M FPGA for control of board functions including registers and storage. Each converter has an independently selectable clock source generated by a low jitter PLL. The external clocking interface consists of LVDS signaling connected via RJ-25 (6-pin) style cabling.

Features and Characteristics of the DSCC are:

- 32-channel 24-bit Delta Sigma A to D Conversion.
- Fully Differential +/-5V Input Interface.
- Industry Standard SCSI 68-pin Connector for Inputs.
- RJ-12 (6-pin phone style) Connectors for External Clocks.
- PCI Express x1 Revision 1.0a.
- Supports MSI Interrupts.
- Independent Clocking for Four Channel Groups.
- Low Jitter Phase Lock Loop (PLL) Clock Generators.
- Supports Multi-board Clocking & Synchronization.
- Directly Addressable Conversion Data Registers.
- 64K Word Conversion Data FIFO with DMA.
- Low Noise Analog Power Generation.
- Positive and Negative Calibration Voltage.
- Gain and Offset Calibration Values Accessible.
- Differential Input Impedance >200K ohm.
- Input Over-Voltage Protection +/-30V @ <150 milliamp.
- Input Over-Current Protection 150 milliamp Fused.
- Sampling Rate (Fs) 2Khz to 216Khz.

The board and driver provide support for MSI interrupts.

2. Requirements

- CCUR-DSCC PCIe board physically installed in the system.
- Selected versions of RedHawk Revision 6.3.x only on either i386, i686 or x86_64 platforms. Actual supported versions depend on the driver being installed.

3. Documentation

- PCIe 32-Channel Delta Sigma Converter Card (DSCC) Software Interface by Concurrent Computer Corporation.
- PCIe 32-Channel Delta Sigma Converter Card (DSCC) Design Specification (No. 0610099) by Concurrent Computer Corporation.

4. Installation and Removal

4.1. Hardware Installation

The CCUR-DSCC card is a x1 PCI Express product and is compatible with any PCI Express slot. The board must be installed in the system before attempting to use the driver.

Caution: ***when installing the card insure the computer is powered off and the machine's power cord is disconnected. Please observe electrostatic discharge precautions such as the use of a grounding strap.***

The ***ccurdscc*** driver is designed to support IRQ sharing. If this device's IRQ is being shared by another device then this driver's performance could be compromised. Hence, as far as possible, move this board into a PCI slot who's IRQ is not being shared with other devices.

An ***'lspci -v'*** or the ***'lsirq'*** command can be used to determine the IRQs of various devices in the system.

```
# lspci –v
```

05:04.0 System peripheral: Concurrent Computer Corporation Device 9277 (rev 01)
Subsystem: PLX Technology, Inc. Device 9056
Flags: bus master, 66MHz, medium devsel, latency 96, IRQ 98
Memory at c4b01000 (32-bit, non-prefetchable) [size=512]
Memory at c4b00000 (32-bit, non-prefetchable) [size=2K]
Capabilities: <access denied>

```
# lsirq
```

98 05:04.0 Concurrent Computer Corporation Unknown device (rev 01)

The default driver configuration uses MSI interrupts. If the kernel supports MSI interrupts, then sharing of interrupts will not occur, in which case the board placement will not be an issue.

After installing the card, reboot the system and verify the hardware has been recognized by the operating system by executing the following command:

```
# lspci –d 1542:9277
```

For each CCUR-DSCC PCIe board installed, a line similar to one of the following will be printed, depending on the revision of the system's ***/usr/share/hwdata/pci.ids*** file:

05:04.0 System peripheral: Concurrent Computer Corporation Device 9277 (rev 01)

If a line similar to the above is not displayed by the **lspci** command, the board has not been properly installed in the system. Make sure that the device has been correctly installed prior to attempting to use the software. One similar line should be found for each installed card.

4.2. Software Installation

Concurrent Computer Corporation™ port of the ***ccurdscc*** software is distributed in RPM format on a CD-ROM. Source for the API library, example test programs, and kernel loadable driver are included, as is documentation in PDF format.

The software is installed in the **/usr/local/CCUR/drivers/ccurdscc** directory. This directory will be referred to as the "top-level" directory by this document.

Warning: Before installing the software, the kernel build environment **must** be set up and match the current OS kernel you are using. If you are running one of the preconfigured kernels supplied by Concurrent and have not previously done so, run the following commands while logged in as the root user before installing the driver software:

```
# cd /lib/modules/`uname -r`/build
# ./ccur-config -c -n
```

If you have built and are running a customized kernel configuration the kernel build environment should already have been set up when that custom kernel was built.

To install the ***ccurdscc*** package, load the CD-ROM installation media and issue the following commands as the **root** user. The system should auto-mount the CD-ROM to a mount point in the **/media** directory based on the CD-ROM's volume label – in this case **/media/ccurdscc_driver**. Then enter the following commands from a shell window:

```
== as root ==
# cd /media/ccurdscc_driver
# rpm -ivh ccurdscc_RedHawk_driver_6.3.*.rpm
# cd /
# eject
```

On successful installation the source tree for the ***ccurdscc*** package, including the loadable kernel module, API libraries, and test programs is extracted into the **/usr/local/CCUR/drivers/ccurdscc** directory by the rpm installation process, which will then compile and install the various software components.

The loadable kernel module is installed in the **/lib/modules/`uname –r`/misc** directory.

4.3. Software Removal

The ***ccurdscc*** driver is a dynamically loadable driver that can be unloaded, uninstalled and removed. Once removed, the only way to recover the driver is to re-install the ***rpm*** from the installation CDROM:

If any changes have been made to the driver package installed in ***/usr/local/CCUR/drivers/ccurdscc*** directory, they need to be backed up prior to invoking the removal; otherwise, all changes will be lost.

```
=== as root ===
# rpm -e ccurdscc     (driver unloaded, uninstalled, and deleted)
```

If, for any reason, the user wishes to un-load and uninstall the driver and not remove it, they can perform the following:

```
=== as root ===
# /sbin/service ccurdscc unload
      --- or ---
# /sbin/service ccurdscc stop
      --- or ---
# cd /usr/local/CCUR/drivers/ccurdscc
# make unload          (unload the driver from the kernel)
```

To uninstall the ***ccurdscc*** driver, do the following after it has been unloaded:

```
=== as root ===
# cd /usr/local/CCUR/drivers/ccurdscc
# make uninstall        (uninstall the driver and library)
```

In this way, the user can simply issue the ***'make install'*** and ***'make load'*** in the ***/usr/local/CCUR/drivers/ccurdscc*** directory at a later date to re-install and re-load the driver.

5. Auto-loading the Driver

The ***ccurdscc*** driver is a dynamically loadable driver. Once you install the package or perform the ***'make install'***, appropriate installation files are placed in the /etc/rc.d/rc*.d directories so that the driver is automatically loaded and unloaded when Linux is booted and shutdown. If, for any reason, you do not wish to automatically load and unload the driver when Linux is booted or shutdown, you will need to manually issue the following command to enable/disable the automatic loading of the driver:

```
=== as root ===
# /sbin/chkconfig --add ccurdscc      (enable auto-loading of the driver)
# /sbin/chkconfig --del ccurdscc      (disable auto-loading of the driver)
```

6. Testing and Usage

Build and run the driver test programs, if you have not already done so:

```
# cd /usr/local/CCUR/drivers/ccurdscc
# make test                          (build the test programs)
```

Several tests have been provided in the ***/usr/local/CCUR/drivers/ccurdscc/test*** directory and can be run to test the driver and board.

```
=== as root ===
# cd /usr/local/CCUR/drivers/ccurdscc
# make test                          (build the test programs)
# ./test/ccurdscc_disp               (display channel data)
# ./test/ccurdscc_get_sps            (determine sample rate for the channels)
# ./test/ccurdscc_rdreg              (display board resisters)
# ./test/ccurdscc_reg                (Display board resisters)
# ./test/ccurdscc_regedit            (Interactive board register editor test)
# ./test/ccurdscc_tst                (Interactive test to test driver and board)
# ./test/ccurdscc_wreg               (edit board resisters)

# ./test/lib/ccurdscc_calibrate      (library: get/set board calibration)
# ./test/lib/ccurdscc_compute_pll_clock  (library: compute pll clock)
# ./test/lib/ccurdscc_disp           (library: display channel data)
# ./test/lib/ccurdscc_fifo           (library: perform FIFO reads)
# ./test/lib/ccurdscc_tst_lib        (library: Interactive test to test driver and board)
```

7. Re-building the Driver, Library and Tests

If for any reason the user needs to manually rebuild and load an *installed* ***rpm*** package, they can go to the installed directory and perform the necessary build.

Warning: Before installing the software, the kernel build environment **must** be set up and match the current OS kernel you are using. If you are running one of the preconfigured kernels supplied by Concurrent and have not previously done so, run the following commands while logged in as the root user before installing the driver software:

```
# cd /lib/modules/`uname -r`/build
# ./ccur-config -c -n
```

If you have built and are running a customized kernel configuration the kernel build environment should already have been set up when that custom kernel was built.

To build the driver and tests:

```
=== as root ===
# cd /usr/local/CCUR/drivers/ccurdscc
# make clobber      (perform cleanup)
# make              (make package and build the driver, library and tests)
```

*(Note: if you only wish to build the driver, you can enter the '**make driver**' command instead)*

After the driver is built, you will need to install the driver. This install process should only be necessary if the driver is re-built with changes.

```
=== as root ===
# cd /usr/local/CCUR/drivers/ccurdscc
# make install      (install the driver software, library and man page)
```

Once the driver and the board are installed, you will need to ***load*** the driver into the running kernel prior to any access to the CCURDSCC board.

```
=== as root ===
# cd /usr/local/CCUR/drivers/ccurdscc
# make load         (load the driver)
```

8. Software Support

This driver package includes extensive software support and test programs to assist the user in communicating with the board. Refer to the *CONCURRENT PCIe 32-Channel Delta Sigma Converter Card (DSCC) Software Interface* document for more information on the product.

8.1. Device Configuration

After the driver is successfully loaded, the device to card association file ***ccurdscc_devs*** will be created in the ***/usr/local/CCUR/drivers/ccurdscc/driver*** directory, if it did not exist. Additionally, there is a symbolic link to this file in the ***/usr/lib/config*** directory as well. If the user wishes to keep the default one-to-one device to card association, no further action is required. If the device to card association needs to be changed, this file can be edited by the user to associate a particular device number with a card number that was found by the driver. The commented portion on the top of the ***ccurdscc_devs*** file is automatically generated every time the user issues the ***'make load'*** or ***'/sbin/service ccurdscc load'*** command with the current detected cards, information. Any device to card association edited and placed in this file by the user is retained and used during the next ***'make load'*** or ***'/sbin/service ccurdscc load'*** process.

If the user deletes the ***ccurdscc_devs*** file and performs a ***'make load'*** or if the user does not associate any device number with card number, the driver will provide a one to one association of

device number and card number. For more information on available commands, view the commented section of the ***ccurdscc_devs*** configuration file.

Warning: If you edit the ***ccurdscc_devs*** file to associate a device to a card, you will need to re-issue the ***'make load'*** or ***'/sbin/service ccurdscc load'*** command to generate the necessary device to card association. This device to card association will be retained until the user changes or deletes the association. **If any invalid association is detected, the loading of the driver will fail**.

8.2. Library Interface

There is an extensive software library that is provided with this package. For more information on the library interface, please refer to the *PCIe 32-Channel Delta Sigma Converter Card (DSCC) Software Interface by Concurrent Computer Corporation* document.

8.3. Firmware Updates

This board is capable of being re-programmed in the field as new firmware updates are made available by *Concurrent Computer Corporation™.* The procedure for re-programming the firmware will be supplied to the user at the time when a firmware update is necessary.

8.4. Debugging

This driver has some debugging capability and should only be enabled while trying to trouble-shoot a problem. Once resolved, debugging should be disabled otherwise it could adversely affect the performance and behavior of the driver.

To enable debugging, the ***Makefile*** file in ***/usr/local/CCUR/drivers/ccurdscc/driver*** should be edited to un-comment the statement (*remove the preceding '#'*):

```
# EXTRA_CFLAGS += -DCCURDSCC_DEBUG
```

Next, compile and install the driver

```
# cd /usr/local/CCUR/drivers/ccurdscc/driver
# make
# make install
```

Next, edit the ***ccurdscc_config*** file in ***/usr/local/CCUR/drivers/ccurdscc/driver*** to un-comment the statement (remove the preceding '**#**'):

```
# ccurdscc_debug_mask=0x00002040
```

Additionally, the value of the debug mask can be changed to suite the problem investigated. Once the file has been edited, the user can load the driver by issuing the following:

```
# cd /usr/local/CCUR/drivers/ccurdscc/driver
# make load
```

The user can also change the debug flags after the driver is loaded by passing the above debug statement directly to the driver as follows:

```
# echo "ccurdscc_debug_mask=0x00082047" > /proc/driver/ccurdscc
```

Following are the supported flags for the debug mask as shown in the ***ccurdscc_config*** file.

```
##############################################################################
#                                                                            #
#         D_ENTER         0x00000001  /* enter routine */                    #
#         D_EXIT          0x00000002  /* exit routine */                     #
#                                                                            #
#         D_L1            0x00000004  /* level 1 */                          #
#         D_L2            0x00000008  /* level 2 */                          #
#         D_L3            0x00000010  /* level 3 */                          #
#         D_L4            0x00000020  /* level 4 */                          #
#                                                                            #
#         D_ERR           0x00000040  /* level error */                      #
#         D_WAIT          0x00000080  /* level wait */                       #
#                                                                            #
#         D_INT0          0x00000100  /* interrupt level 0 */                #
#         D_INT1          0x00000200  /* interrupt level 1 */                #
#         D_INT2          0x00000400  /* interrupt level 2 */                #
#         D_INT3          0x00000800  /* interrupt level 3 */                #
#         D_INTW          0x00001000  /* interrupt wakeup level */           #
#         D_INTE          0x00002000  /* interrupt error */                  #
#                                                                            #
#         D_RTIME         0x00010000  /* display read times */               #
#         D_WTIME         0x00020000  /* display write times */              #
#         D_REGS          0x00040000  /* dump registers */                   #
#         D_IOCTL         0x00080000  /* ioctl call */                       #
#                                                                            #
#         D_DATA          0x00100000  /* data level */                       #
#         D_DMA           0x00200000  /* DMA level */                        #
#         D_DBUFF         0x00800000  /* DMA buffer allocation */            #
#                                                                            #
#         D_NEVER         0x00000000  /* never print this debug message */   #
#         D_ALWAYS        0xffffffff  /* always print this debug message */  #
#         D_TEMP          D_ALWAYS    /* Only use for temporary debug code */ #
##############################################################################
```

Another variable ***ccurdscc_debug_ctrl*** is also supplied in the ***ccurdscc_config*** that the driver developer can use to control the behavior of the driver. The user can also change the debug flags after the driver is loaded by passing the above debug statement directly to the driver as follows:

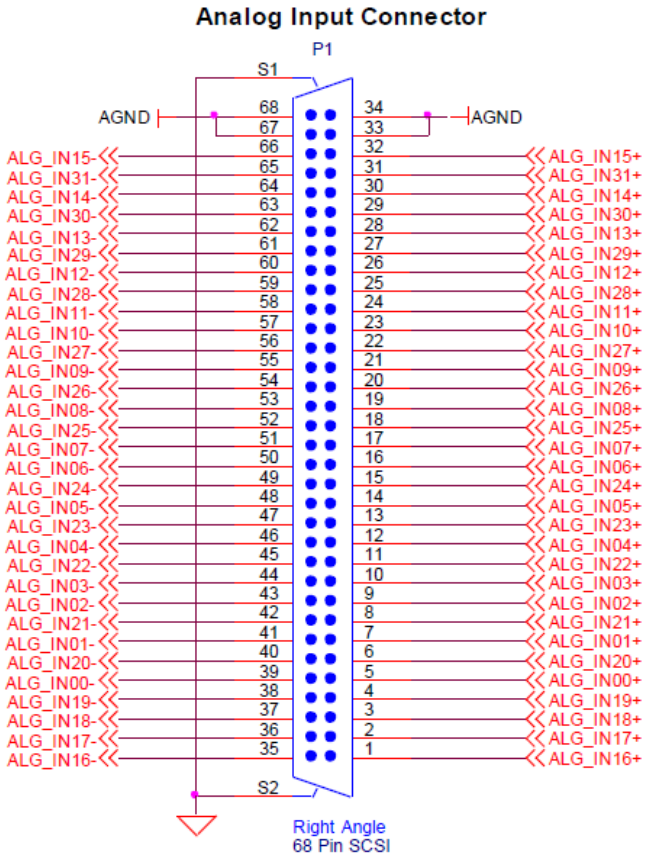
```
# echo "ccurdscc_debug_ctrl=0x00001234" > /proc/driver/ccurdscc
```

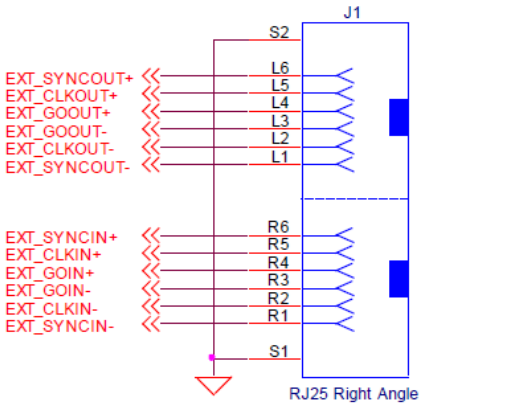
In order to make use of this variable, the driver must be coded to interrogate the bits in the ***ccurdscc_debug_ctrl*** variable and alter its behavior accordingly.

9. Notes and Errata

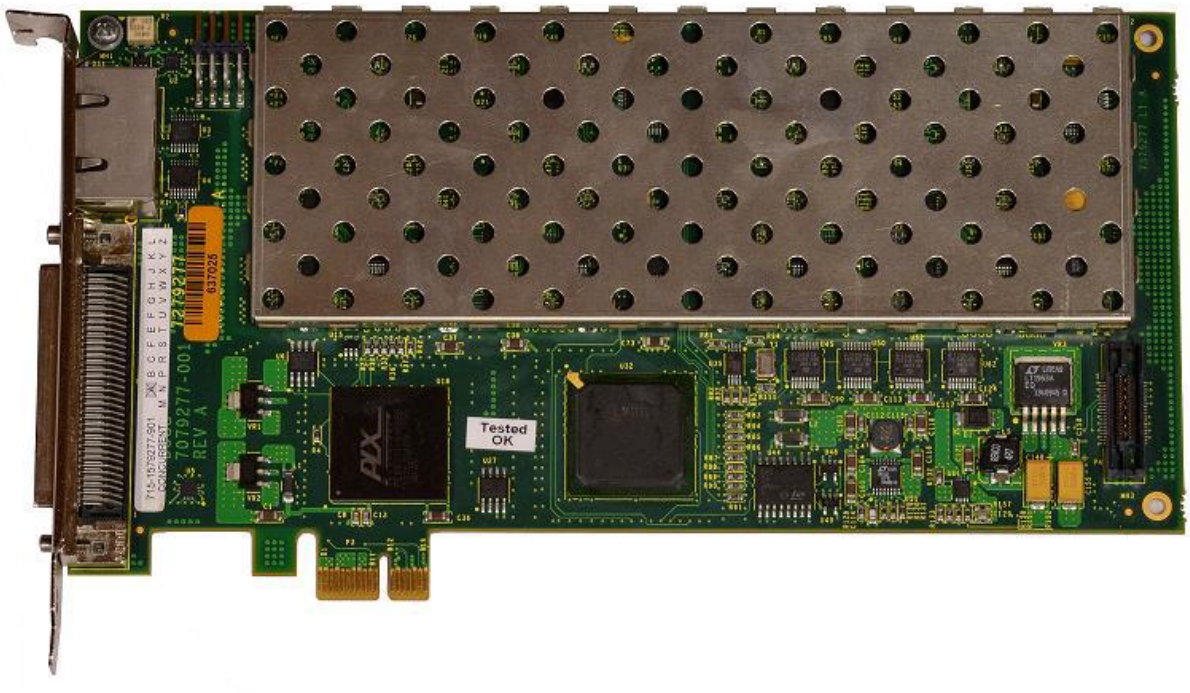
- The board can be ordered with optional maximum voltage range. The maximum voltage range for the default board is +/- 5 Volts.
- Only differential mode is supported by this board.
- When synchronizing multiple cards, only one synchronization clock can be selected even though the board supports multiple clocks.
- Driver and board supports MSI interrupts.

Appendix A: External Connections and Pin-outs





Appendix B: The Board



CCUR-PCIe-DSCC Card

This page intentionally left blank