Release Notes

CCURPWM (WC-PWM-1012 Output)

Driver	ccurpwm (WC-PWM-1012)	v 23.0_1
OS	RedHawk	
Vendor	Concurrent Real-Time, Inc.	
Hardware	12-Channel PWM Output Card (CP-PWM-1012)	
Date	April 13, 2018	

Table of Contents

1. INTRODUCTION 1

2. REQUIREMENTS 1

3. DOCUMENTATION 1

4. INSTALLATION AND REMOVAL 1

4.1. Hardware Installation 1

4.2. Software Installation 2

4.3. Software Removal 3

5. AUTO-LOADING THE DRIVER 4

6. TESTING AND USAGE 4

7. RE-BUILDING THE DRIVER, LIBRARY AND TESTS 4

8. SOFTWARE SUPPORT 5

8.1. Device Configuration 5

8.2. Library Interface 6

8.3. Firmware Updates 6

8.4. Debugging 6

9. NOTES AND ERRATA 7

APPENDIX A: MODES OF OPERATION 8

APPENDIX B: EXTERNAL CONNECTIONS AND PIN-OUTS 10

APPENDIX C: THE BOARD 12

1. Introduction

This document assists the user in installing the CP-PWM-1012-Linux ***ccurpwm*** driver and related software on the RedHawk OS for use with the CP-PWM-1012 board. The directions in this document supersede all others – they are specific to installing the software on Concurrent Real-Time'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 ***ccurpwm*** and ***ccurpwm_lib*** man pages.

Brief Description of the CP-PWM-1012 Board
The CP-PWM-1012 is an FPGA-based Pulse Width Modulation (PWM) card from Concurrent. The CP-PWM-1012 autonomously generates TTL pulse width modulated signals with high accuracy. With a timing resolution of 50 ns and the ability to program sine frequencies, PWM frequencies, dead-band, and duty cycle in real time makes this card ideal for use in Hardware-In-Loop (HIL) systems. The CP-PWM-1012 comes in PCIe form factor. Multiple CP-PWM-1012 cards can be placed in one system. A Molex LFH-60 connector is mounted on each card for connection to external devices.

This driver is not supported with 32-bit applications running on RedHawk 7.0 and 7.2 64-bit platforms.

CP-PWM-1012 Features:
- FPGA based PWM board
- PCIe form factor
- TTL outputs
- 2 channel, 3 phase complementary PWM outputs (12 outputs) or
- 1 channel, 6 phase complementary PWM outputs (12 outputs) or
- 12 channel straight PWM outputs (12 outputs)
- 12 bit PWM signal resolution
- In sine wave mode, programmable sine frequency, phase angles, dead-band, and PWM frequency
- In straight PWM mode, programmable PWM frequency and duty cycle
- 20 MHz PWM base frequency i.e. timing resolution of 50 ns
- 66 MHz board frequency

This version of the driver and library also provides support for the 64-bit Opteron architecture.

2. Requirements

- CP-PWM-1012 PCIe board installed.
- Selected versions of RedHawk Revision 5.4, 6.0, 6.3, 6.5, 7.0, 7.2, and 7.3. Actual supported versions depend on the individual driver.

3. Documentation

- 12-Channel Pulse Width Modulation (PWM) Output PCIe card datasheet by Concurrent Real-Time.

4. Installation and Removal

4.1. Hardware Installation

The CCURPWM 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 ***ccurpwm*** driver is designed to support IRQ sharing, however, since it does not use interrupts, it should not matter which PCIe slot is placed in.

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

```
# lspci –v -d 1542:9272
   05:04.0 System peripheral: Concurrent Computer Corporation Device 9272 (rev 01)
           Subsystem: PLX Technology, Inc. Device 9056
           Flags: bus master, 66MHz, medium devsel, latency 96, IRQ 19
           Memory at c4b08000 (32-bit, non-prefetchable) [size=512]
           Memory at c4b00000 (32-bit, non-prefetchable) [size=32K]
           Capabilities: <access denied>
# lsirq
   19      05:04.0 Concurrent Computer Corporation Unknown device (rev 01)
```

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 1542:9272
```

For each CCURPWM-1012 (Output) or CCURPWMIN-1112 (Input) PCIe board installed, a line like 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 9272 (rev 01)

If a line like 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. The individual driver detects whether the board is a CCURPWM-1012 (Output) or CCURPWMIN-1112 (Input) based on the firmware register. If a ***ccurpwm*** driver is installed and only *input* cards are present, or if a ***ccurpwmin*** driver is installed and only *output* cards are present, then the driver loading will fail and appropriate message will be generated in the kernel log file that can be viewed by the ***dmesg*** command.

4.2. Software Installation

Concurrent Real-Time's port of the ***ccurpwm*** software is distributed in RPM and DEB 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/CCRT/drivers/ccurpwm** 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 ***ccurpwm*** 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** or **/run/media** directory based on the CD-ROM's volume label – in this case **ccurpwm_driver**. The example's ***[user_name*]** may be ***root***, or the logged-in user. Then enter the following commands from a shell window:

```
=== as root ===
      --- on RedHawk 6.5 and below ---
# cd /media/ccurpwm_driver
      --- or on RedHawk 7.0 and above ---
# cd /run/media/[user_name]/ccurpwm_driver

# rpm –ivh ccurpwm_RedHawk_driver*.rpm   (on an RPM based system)
      --- or ---
# dpkg –i ccurpwm_RedHawk_driver*.deb    (on a Debian based system)

# cd /
# eject
```

On successful installation, the source tree for the ***ccurpwm*** package, including the loadable kernel module, API libraries, and test programs is extracted into the **/usr/local/CCRT/drivers/ccurpwm** 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 ***ccurpwm*** 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/CCRT/drivers/ccurpwm*** directory, they need to be backed up prior to invoking the removal; otherwise, all changes will be lost.

```
=== as root ===
# rpm –e ccurpwm   (driver unloaded, uninstalled, and deleted – on an RPM based system)
--- or ---
# dpkg -P ccurpwm  (driver unloaded, uninstalled, and deleted – on a Debian based system)
```

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 ===
# cd /usr/local/CCRT/drivers/ccurpwm
# make unload          (unload the driver from the kernel)
      --- or on RedHawk 6.5 and below ---
# /sbin/service ccurpwm stop
      --- or on RedHawk 7.0 and above ---
# /usr/bin/systemctl stop ccurpwm
```

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

```
=== as root ===
# cd /usr/local/CCRT/drivers/ccurpwm
# 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/CCRT/drivers/ccurpwm*** directory at a later date to re-install and re-load the driver.

5. Auto-loading the Driver

The ***ccurpwm*** driver is a dynamically loadable driver. Once you install the package or perform the ***'make install'***, appropriate installation files are placed in the /usr/lib/system/systemd directory 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 ===
      --- on RedHawk 6.5 and below ---
# /sbin/chkconfig --add ccurpwm           (enable auto-loading of the driver)
# /sbin/chkconfig --del ccurpwm           (disable auto-loading of the driver)
      --- or on RedHawk 7.0 and above ---
# /usr/bin/systemctl enable ccurpwm       (enable auto-loading of the driver)
# /usr/bin/systemctl disable ccurpwm      (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/CCRT/drivers/ccurpwm
# make test                           (build the test programs)
```

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

```
=== as root ===
# cd /usr/local/CCRT/drivers/ccurpwm
# make test                (build the test programs)
# ./test/ccurpwm_dump      (Display board registers including pci config and bridge register
                             and main control registers)
# ./test/ccurpwm_reg       (Display board registers)
# ./test/ccurpwm_tst       (Interactive test to check driver and board)
# ./test/ccurpwm_tst_lib   (Interactive test to check the library)
# ./test/ccurpwm_tst_reg   (Automated test to check registers)
# ./test/ccurpwm_rdreg     (Read registers by offset)
# ./test/ccurpwm_wreg      (Write registers by offset)
```

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/CCRT/drivers/ccurpwm
# 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/CCRT/drivers/ccurpwm
# 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 CCURPWM board.

```
=== as root ===
# cd /usr/local/CCRT/drivers/ccurpwm
# 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 installed *ccurpwm* and *ccurpwm_lib* man pages for more information on the product.

8.1. Device Configuration

After the driver is successfully loaded, the device to card association file ***ccurpwm_devs*** will be created in the ***/usr/local/CCRT/drivers/ccurpwm/driver*** directory, if it did not exist. Additionally, there is a symbolic link to this file in the ***/usr/lib/config/ccurpwm*** 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 ***ccurpwm_devs*** file is automatically generated every time the user issues the ***'make 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'*** process.

If the user deletes the ***ccurpwm_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 ***ccurpwm_devs*** configuration file.

Warning: If you edit the ***ccurpwm_devs*** file to associate a device to a card, you will need to re-issue the ***'make 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 installed *ccurpwm_lib* man page.

8.3. Firmware Updates

This board is capable of being re-programmed in the field as new firmware updates are made available by *Concurrent Real-Time™.* 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/CCRT/drivers/ccurpwm/driver*** should be edited to un-comment the statement (*remove the preceding '#'*):

```
# BUILD_TYPE=debug
```

Next, compile and install the driver

```
# cd /usr/local/CCRT/drivers/ccurpwm/driver
# make
# make install
```

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

```
# ccurpwm_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/CCRT/drivers/ccurpwm/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 "ccurpwm_debug_mask=0x00082047" > /proc/driver/ccurpwm
```

Following are the supported flags for the debug mask as shown in the ***ccurpwm_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_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 ***ccurpwm_debug_ctrl*** is also supplied in the ***ccurpwm_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 "ccurpwm_debug_ctrl=0x00001234" > /proc/driver/ccurpwm
```

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

9. Notes and Errata

- Currently the driver does not use interrupts.
- This card can be used in conjunction with the PWM Input card CP-PWM-1112 also supplied by Concurrent Real-Time.

Appendix A: Modes of Operation

The CP-PWM-1012 card has 2 primary modes of operation:

Straight PWM Mode*:* In this mode, the card outputs 12 individual PWM waveforms. The PWM frequency and the duty cycle are programmable for each individual channel.

Carrier frequency:	0.1Hz - 1MHz
Duty cycle:	0-100% in 0.1% increments

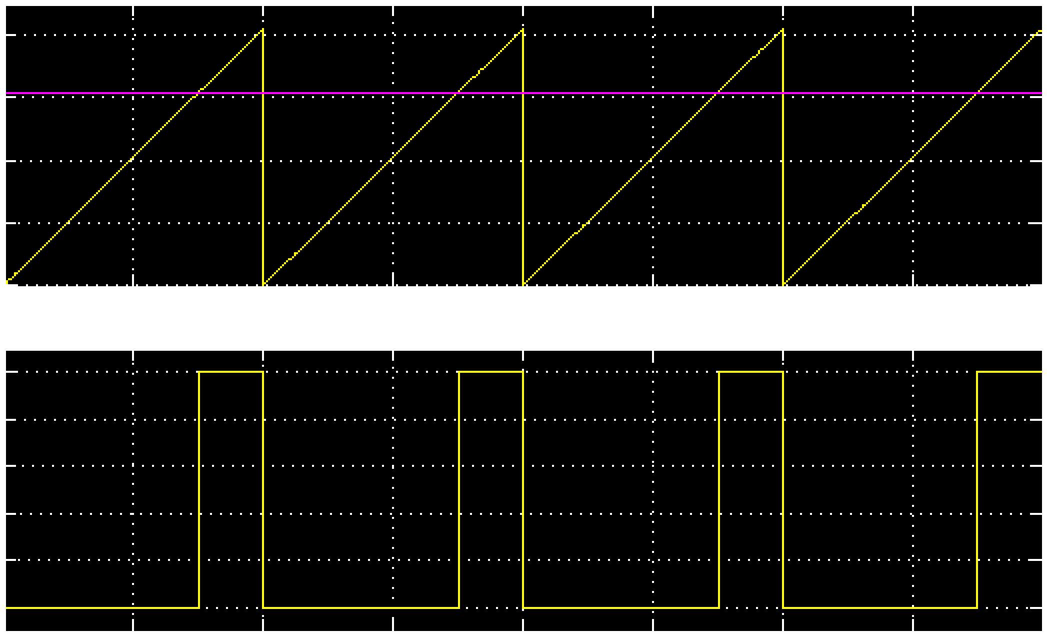


Figure 1: *Single channel PWM output with a 25% duty cycle*

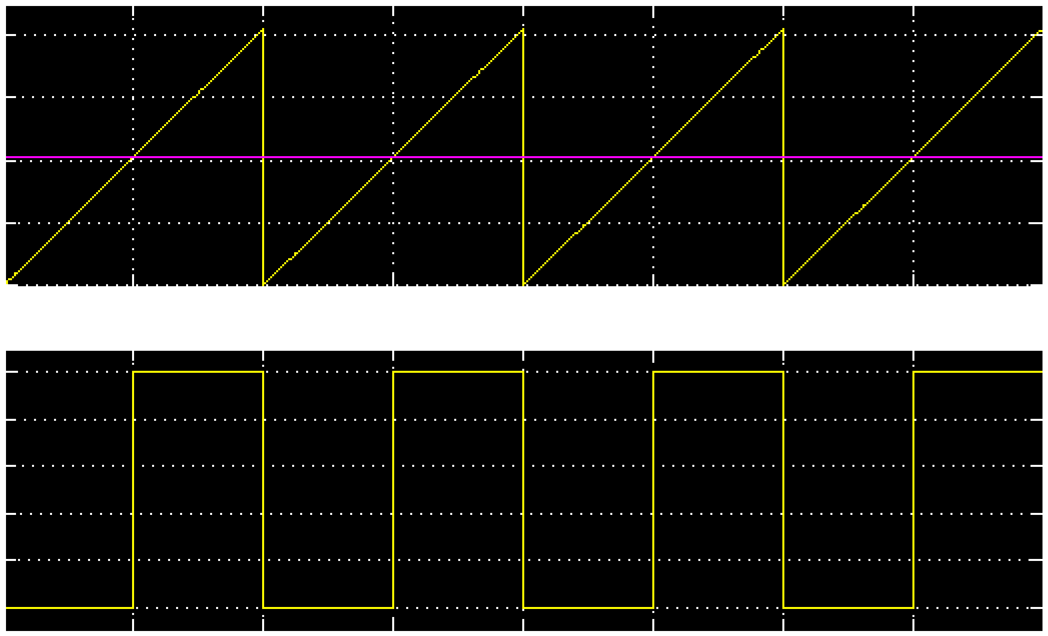


Figure 2: *Single channel PWM output with a 50% duty cycle*

3-Phase PWM (2 Channels): In this mode, the card outputs 2 channels of 3 phase complementary PWM outputs. Thus, for each channel there will be 6 outputs, 3 for each phase and 3 complementary outputs for each phase. The phases for the 2 channels can be setup to form a single channel 6 phase complementary output card. The sine wave frequencies, phase angles, dead-band, and carrier frequencies are all programmable.

6-Phase PWM (1 Channel): In this mode, the card outputs 1 channel of 6 phase complementary PWM outputs. Thus, for this channel there will be 12 outputs, 6 for each phase and 6 complementary outputs for each phase. The sine wave frequencies, phase angles, dead-band, and carrier frequencies are all programmable.

Carrier frequency:	0.1Hz – 1 MHz Must be twice the sine frequency to satisfy the Nyquist sampling theorem
Sine frequency:	0.01Hz – 100 KHz
Phase Angles:	0-360 degrees
Dead-band:	Minimum 1 (i.e. 50 ns dead-band)

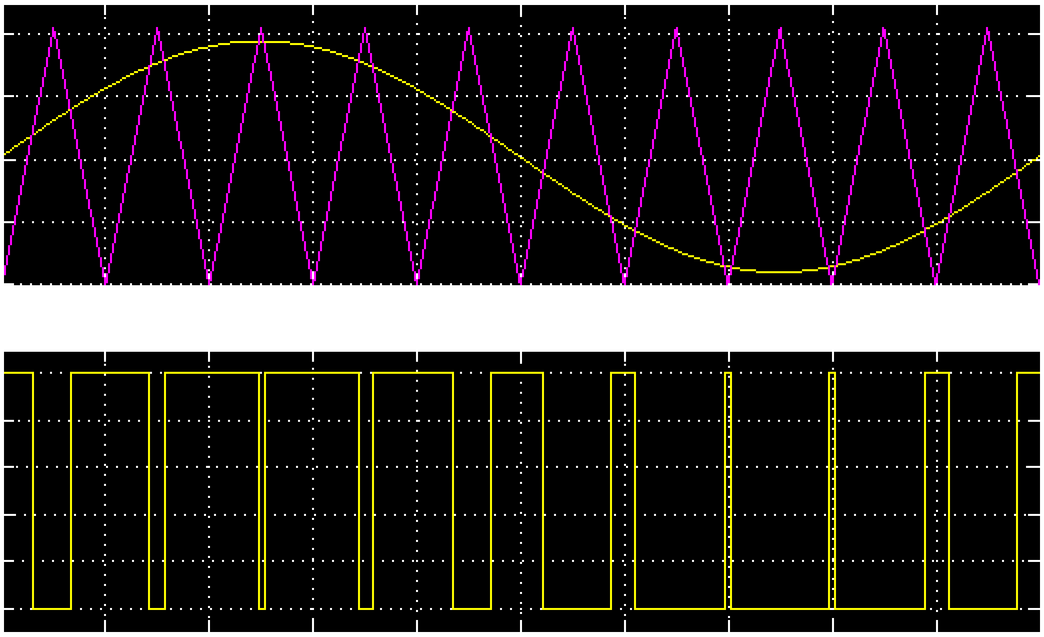


Figure 4: *PWM output of a single phase sine wave*

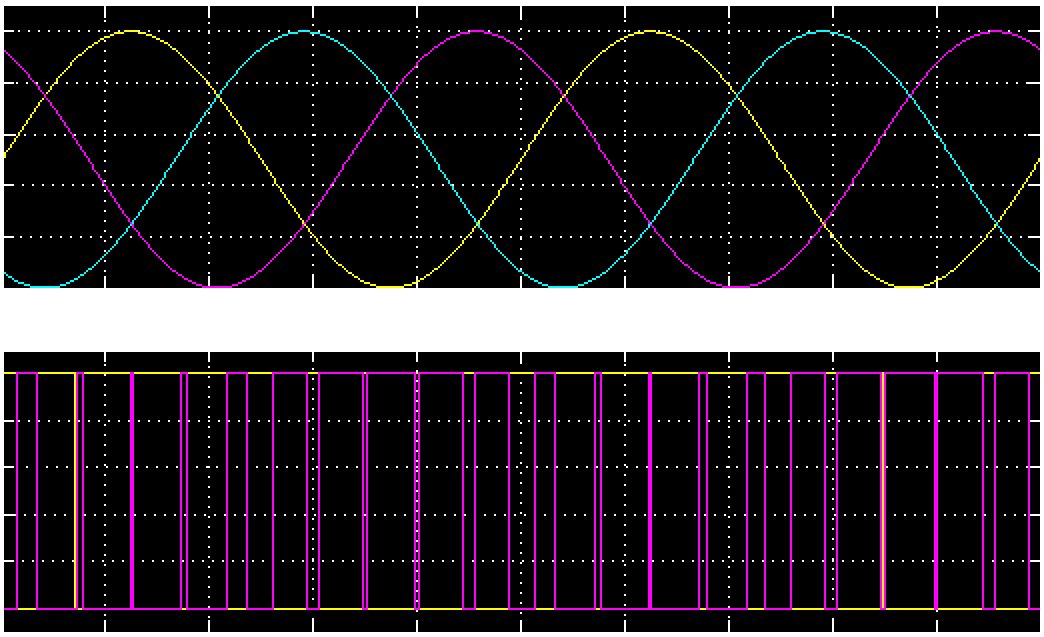
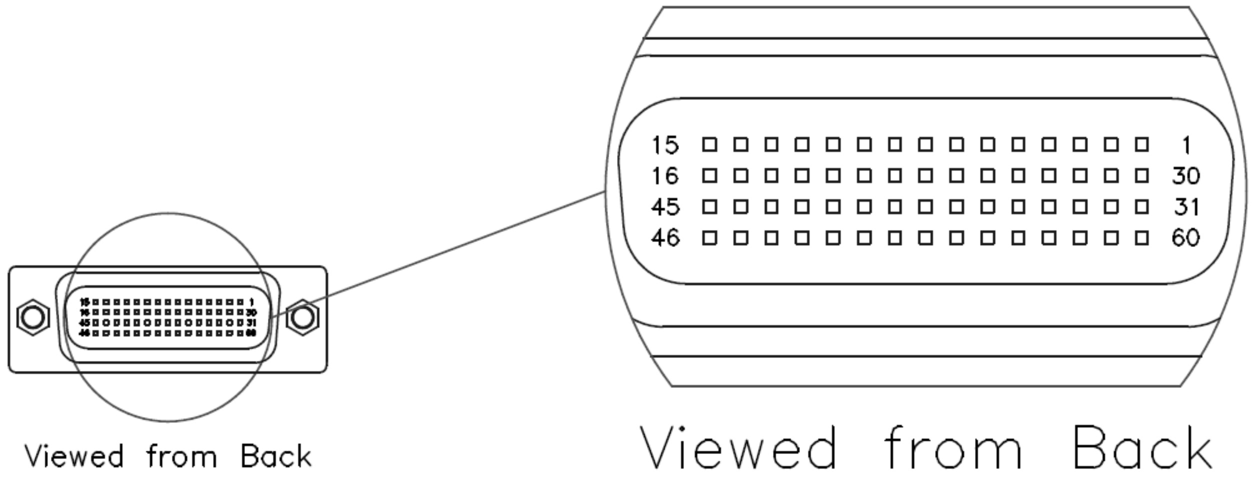


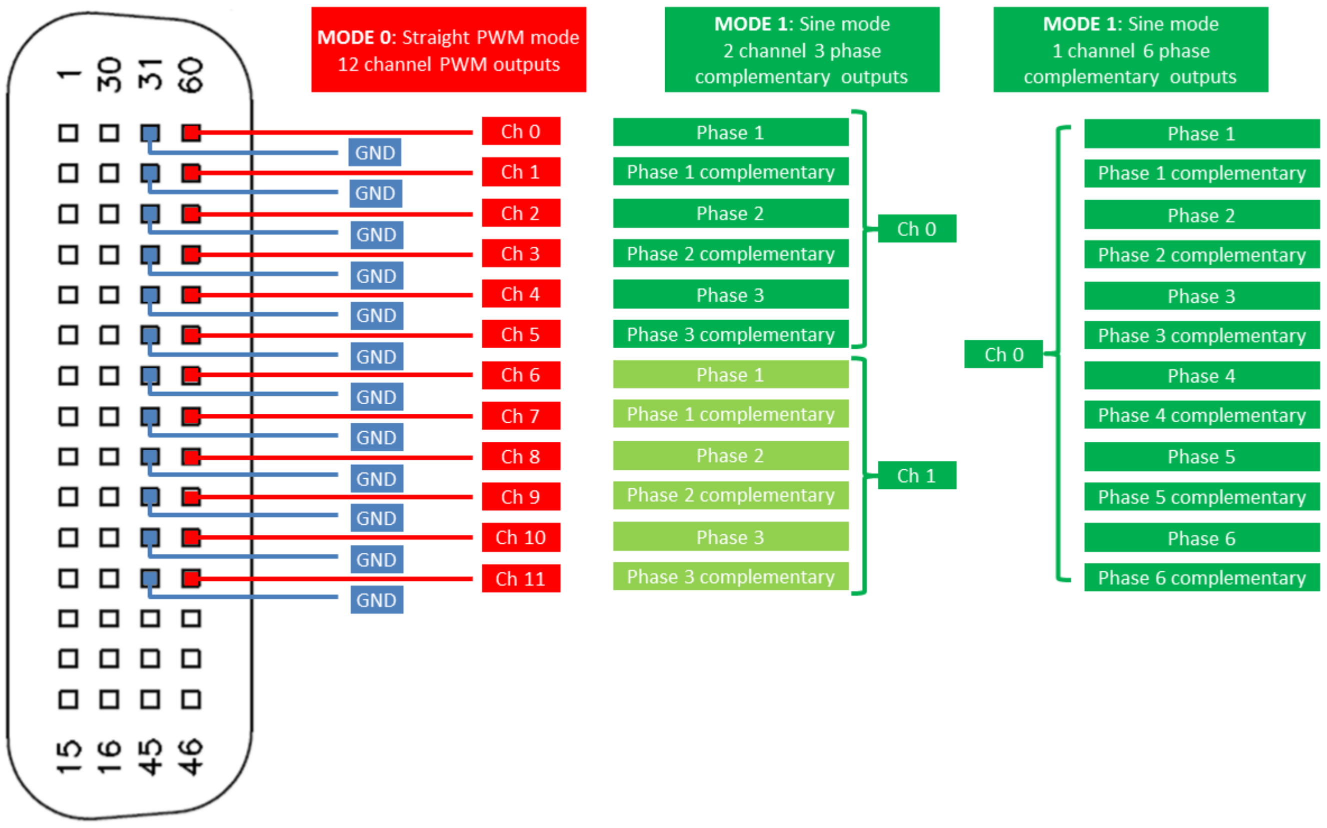
Figure 5: *PWM and complementary PWM output of a three phase sine wave*

Appendix B: External Connections and Pin-outs

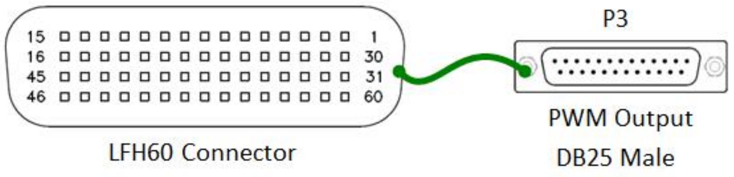
LFH60 Connector

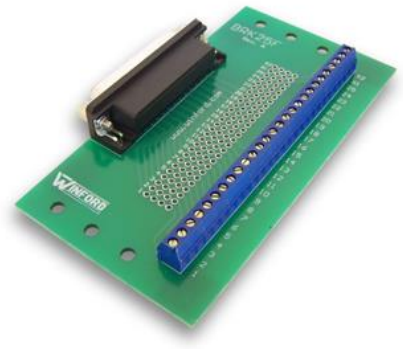


LFH60 PWM outputs



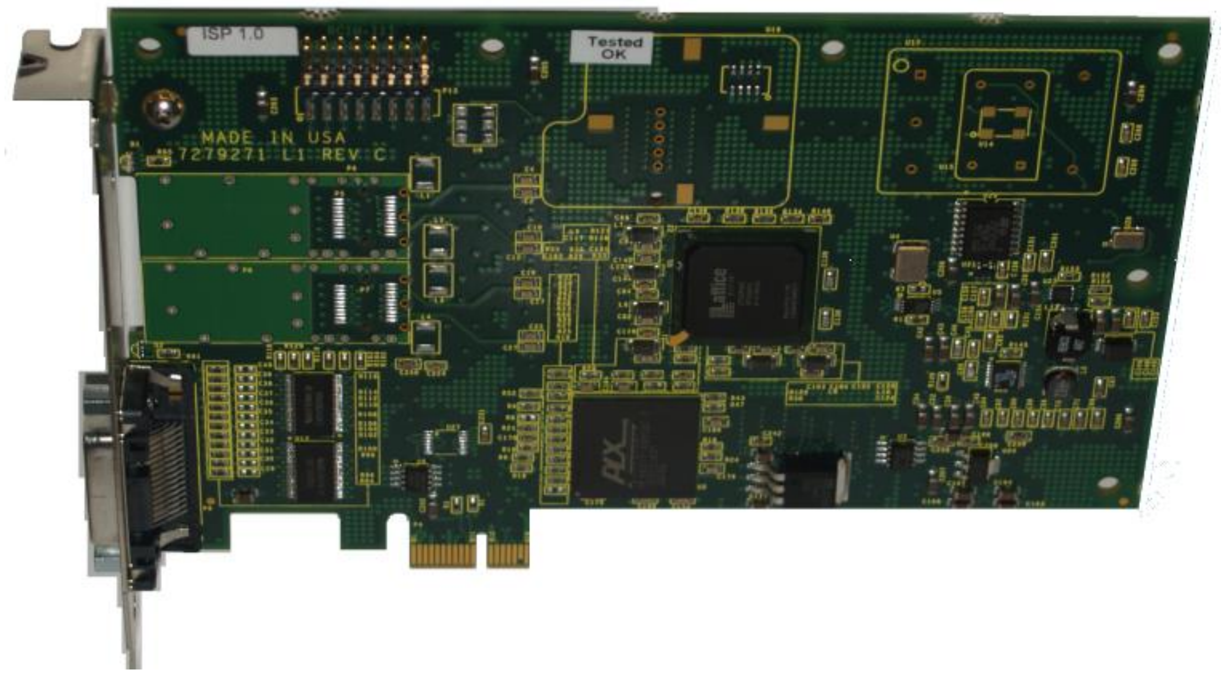
720-1831080-901 Cable and breakout board connections





Breakout Board Pinout	
Pin 1	Out
Pin 2	Gnd
Pin 3	Out
Pin 4	Gnd
Pin 5	Out
Pin 6	Gnd
Pin 7	Out
Pin 8	Gnd
Pin 9	Out
Pin 10	Gnd
Pin 11	Out
Pin 12	Gnd
Pin 13	Out
Pin 14	Gnd
Pin 15	Out
Pin 16	Gnd
Pin 17	Out
Pin 18	Gnd
Pin 19	Out
Pin 20	Gnd
Pin 21	Out
Pin 22	Gnd
Pin 23	Out
Pin 24	Gnd

Appendix C: The Board



CP-PWM-1012 Card