













Manual

Diamond Systems

DS-MPE-CAN2L

PCIe MiniCardI/O Expansion Module with Dual CANbus Ports



The information contained in this document has been carefully researched and is, to the best of our knowledge, accurate. However, we assume no liability for any product failures or damages, immediate or consequential, resulting from the use of the information provided herein. Our products are not intended for use in systems in which failures of product could result in personal injury. All trademarks mentioned herein are property of their respective owners. All specifications are subject to change without notice.



DS-MPE-CAN2L

PCIe MiniCard Dual CAN 2.0 Port Module

Rev A.1 April 2015



Revision	Date	Comment
A.0	4/25/2014	Initial release
A.1	4/09/2015	Updated Windows installation procedure

FOR TECHNICAL SUPPORT PLEASE CONTACT:

support@diamondsystems.com

© Copyright 2015 Diamond Systems Corporation 555 Ellis Street Mountain View, CA 94043 USA Tel 1-650-810-2500 Fax 1-650-810-2525 www.diamondsystems.com



CONTENTS

1.	Important Safe Handling Information	3
2.	Introduction	4
2.	1 Description	4
2.	2 Features	4
2.	3 Operating System Support	4
2.	4 Mechanical, Electrical, Environmental	4
3.	Packing List	4
4.	Functional Overview	5
4.	1 Functional Block Diagram	5
4.	2 Mechanical Board Drawing	6
4.	3 CAN Controllers	6
4.	4 Transceivers	7
4.	5 Isolation	7
4.	6 Power Supply	7
5.	Installation	7
6.	Connector Pinout and Pin Description	8
6.	1 PCIe MiniCard Edge Connector (J1)	8
6.	2 CAN Ports (J4, J7)	8
7.	Jumper Configuration	9
8.	Linux Driver installation	10
8.	1 Installing the Software	10
8.	2 Setting the Baud Rate	11
8.	3 Setting the CAN ID and Message Length	12
8.	4 Writing a Message	13
8.	5 Viewing Messages	14
9.	Configure and Manage the Ports in Linux	16
9.	1 API to Configure and Manage CAN Ports	16
9.	2 Compiling User Application using CANLib Library	19
10.	Driver installation and Demo Application for Windows	20
10	0.1 Installing the PCI-CAN Driver	20
10	0.2 Run the Windows Application	24
10	0.3 Setting the Baud Rate	25
10	0.4 Setting the CAN ID and Message Length	26
10	0.5 Writing a Message	27
10	0.6 Viewing Messages	28
11.	API to Configure and Manage CAN Ports on Windows	29
12.	Specifications	33



1. IMPORTANT SAFE HANDLING INFORMATION



WARNING!

ESD-Sensitive Electronic Equipment

Observe ESD-safe handling procedures when working with this product.

Always use this product in a properly grounded work area and wear appropriate ESD-preventive clothing and/or accessories.

Always store this product in ESD-protective packaging when not in use.

Safe Handling Precautions

This board contains a high density connector with many connections to sensitive electronic components. This creates many opportunities for accidental damage during handling, installation and connection to other equipment. The list here describes common causes of failure found on boards returned to Diamond Systems for repair. This information is provided as a source of advice to help you prevent damaging your Diamond (or any vendor's) embedded computer boards.

ESD damage – This type of damage is usually almost impossible to detect, because there is no visual sign of failure or damage. The symptom is that the board eventually simply stops working, because some component becomes defective. Usually the failure can be identified and the chip can be replaced. To prevent ESD damage, always follow proper ESD-prevention practices when handling computer boards.

Damage during handling or storage – On some boards we have noticed physical damage from mishandling. A common observation is that a screwdriver slipped while installing the board, causing a gouge in the PCB surface and cutting signal traces or damaging components.

Another common observation is damaged board corners, indicating the board was dropped. This may or may not cause damage to the circuitry, depending on what is near the corner. Most of our boards are designed with at least 25 mils clearance between the board edge and any component pad, and ground / power planes are at least 20 mils from the edge to avoid possible shorting from this type of damage. However these design rules are not sufficient to prevent damage in all situations.

A third cause of failure is when a metal screwdriver tip slips, or a screw drops onto the board while it is powered on, causing a short between a power pin and a signal pin on a component. This can cause overvoltage / power supply problems described below. To avoid this type of failure, only perform assembly operations when the system is powered off.

Sometimes boards are stored in racks with slots that grip the edge of the board. This is a common practice for board manufacturers. However our boards are generally very dense, and if the board has components very close to the board edge, they can be damaged or even knocked off the board when the board tilts back in the rack. Diamond recommends that all our boards be stored only in individual ESD-safe packaging. If multiple boards are stored together, they should be contained in bins with dividers between boards. Do not pile boards on top of each other or cram too many boards into a small location. This can cause damage to connector pins or fragile components.

Power supply wired backwards – Our power supplies and boards are not designed to withstand a reverse power supply connection. This will destroy each IC that is connected to the power supply (i.e. almost all ICs). In this case the board will most likely will be unrepairable and must be replaced. A chip destroyed by reverse power or by excessive power will often have a visible hole on the top or show some deformation on the top surface due to vaporization inside the package. **Check twice before applying power!**

Overvoltage on digital I/O line – If a digital I/O signal is connected to a voltage above the maximum specified voltage, the digital circuitry can be damaged. On most of our boards the acceptable range of voltages connected to digital I/O signals is 0-5V, and they can withstand about 0.5V beyond that (-0.5 to 5.5V) before being damaged. However logic signals at 12V and even 24V are common, and if one of these is connected to a 5V logic chip, the chip will be damaged, and the damage could even extend past that chip to others in the circuit



2. INTRODUCTION

2.1 **Description**

DS-MPE-CAN2L implements a CAN protocol bus controller that performs serial communications according to the CAN 2.0A and CAN 2.0B specifications. The protocol uses a multi-master bus configuration for the transfer of frames between nodes of the network and manages error handling with no burden on the host processor.

2.2 Features

- 2 CAN 2.0B ports with a 1Mbps data rate and programmable interrupts
- 31 receive buffers for improved performance
- 1 high priority transmit buffer and 16 standard priority transmit buffers
- 16 programmable acceptance filters
- 11-bit and 29-bit identifiers
- 500V port-to-port and input-to-output isolation
- Driver supports dual-independent and dual-redundant modes
- Latching connectors for increased ruggedness

2.3 **Operating System Support**

- Linux 2.6.16, 2.6.27, 2.6.31 and 2.6.32
- Windows 7

2.4 Mechanical, Electrical, Environmental

- PCIe MiniCard full size format
- Dimensions: 50.95mm x 30mm (2" x 1.18")
- -40°C to +85°C ambient operating temperature
- Power input requirements: +3.3VDC +/- 5%

3. PACKING LIST

The DS-MPE-CAN2L product comes with the PCIe MiniCard hardware assembly, a cable kit with two dual serial cables, and a hardware kit containing jumpers and mounting screws.

Quantity	Part Number	Description
1	9150500	DS-MPE-CAN2L hardware assembly
1	6800500	Hardware Kit with jumpers and screws
1	CK-CAN2L	Cable Kit with two CAN cables





4. FUNCTIONAL OVERVIEW

4.1 Functional Block Diagram

The DS-MPE-CAN2L block diagram is shown below.





4.2 Mechanical Board Drawing

The DS-MPE-CAN2L conforms to the PCIe MiniCard electromechanical specification revision 1.2, full size format. Overall dimensions are 50.95mm L x 30.00mm W.

The two mounting holes are isolated from the CPU ground and not connected to any ground lines.



4.3 CAN Controllers

The module offers two CAN controllers implemented as FPGA cores inside a Xilinx Spartan 6 FPGA. The core provides the following key features:

- Conforms to the ISO 11898 -1, CAN 2.0A, and CAN 2.0B standards
- Supports both standard (11-bit identifier) and extended (29-bit identifier) frames
- Supports bit rates up to 1Mbps
- Transmit message FIFO with a user-configurable depth of up to 64 messages
- Transmit prioritization through one High-Priority Transmit buffer
- Automatic re-transmission on errors or arbitration loss
- Receive message FIFO with a user-configurable depth of up to 64 messages
- Acceptance filtering with a user-configurable number of up to 16 acceptance filters
- Sleep Mode with automatic wake-up
- Loop Back Mode for diagnostic applications
- Maskable Error and Status Interrupts
- Readable Error Counters



4.4 Transceivers

The transceivers are Analog Devices ADM3053 combination isolation and transceiver. It provides isolated +5V to power the isolated side of the transceiver. This isolated +5V is available on the I/O connector.

4.5 **Isolation**

The module supports 500V isolation between each CAN port and the host, and between each CAN port and the other, via the ADM3053 isolated transceiver. An optional high-voltage resistor can be installed across each isolation barrier to enable leakage current flow between the isolated transceiver grounds and the host ground.

4.6 **Power Supply**

The module is powered by +3.3V from the PCIe MiniCard socket. It provides all other required voltages on board, including +5V for the CAN transceivers and the FPGA core voltages.

5. INSTALLATION

The DS-MPE-CAN2L plugs in to any socket meeting the PCIe MiniCard specifications. It has two connectors, one for each pair of serial ports, a protocol configuration jumper block, and a pair of mounting holes. To install the DS-MPE-CAN2L, fully insert the board into a PCIe MiniCard connector and secure in place by inserting one screw from the hardware kit into each of the mounting holes, see the diagram below.





6. CONNECTOR PINOUT AND PIN DESCRIPTION

6.1 PCIe MiniCard Edge Connector (J1)

The DS-MPE-CAN2L module is compatible with the standard Mini PCIe socket pinout as shown below.

WAKE#	1	2	+3.3VAUX_3
COEX1	3	4	GND9
COEX2	5	6	+1.5V_1
CLKREQ#	7	8	UIM_PWR
GND1	9	10	UIM_DATA
REFCLK-	11	12	UIM_CLK
REFCLK+	13	14	UIM_RESET
GND2	15	16	UIM_VPP
	K	ΞY	
RSVD(UIM_C8)	17	18	GND10
RSVD(UIM_C4)	19	20	W_DISABLE#
GND3	21	22	PERST#
PERN0	23	24	+3.3VAUX_4
PERP0	25	26	GND11
GND4	27	28	+1.5V_2
GND5	29	30	SMB_CLK
PETN0	31	32	SMB_DATA
PETP0	33	34	GND12
GND6	35	36	USB_D-
GND7	37	38	USB_D+
+3.3VAUX_1	39	40	GND13
+3.3VAUX_2	41	42	LED_WWAN#
GND8	43	44	LED_WLAN#
RSVD1	45	46	LED_WPAN#
RSVD2	47	48	+1.5V_3
RSVD3	49	50	GND14
RSVD4	51	52	+3.3VAUX_5

6.2 CAN Ports (J4, J7)

Each of the two CAN ports has its own 4-pin latching connector with the following pin out.

1	Ground Iso
2	CAN L
3	CAN H
4	Ground Iso

Connector Part Number / Description

BM04B-GHS-TBT 4 pos, 1.25mm, vertical, latching, SMD



7. JUMPER CONFIGURATION

The DS-MPE-CAN2L module has two line termination jumper blocks, one for each port. Jumper block J3 is for port J4, and jumper block J6 is for port J7. Jumper blocks J3 and J6 are identical. The default is no jumpers installed. To add termination for a port's bias line (jumper position B), CAN-H line (jumper position H), or CAN-L line (jumper position L), add a jumper at B, H or L location respectively.







8. LINUX DRIVER INSTALLATION

8.1 **Installing the Software**

The following steps are used to install the CAN interface utility software under the Linux operating system.

Step-1:

Download the DSC_CAN2L_PCI_LINUX_V1.7.zip file from the DS-MPE-CAN2L webpage (<u>http://www.diamondsystems.com/products/dsmpecan2l</u>). Click on *Linux driver package v1.0.7* in the Downloads section of the webpage. Use the following command to unzip the files:

```
Unzip DSC_CAN2L_PCI_LINUX_V1.7.zip
```

A DSC_CAN2L_PCI_LINUX_V1.7 directory will be created where the zip file is extracted. The DSC_CAN2L_PCI_LINUX_V1.7 directory contains the following files.

ls -l

- 1. CAN_Monitor : CAN Monitor demo application directory
- 2. CANLib : CAN Linux shared library.
- 3. dsc can2 pci driver: Linux CAN driver.
- 4. qt-opensource-linux-x86-5.2.1.run : Qt Installer which is required by the PCI CAN Interface utility.

Step-2:

Install the Qt shared libraries using the Qt Installer. Execute the command below and follow the Qt Installer instructions. Use the command below to install the Qt shared libraries. Install Qt at the default locations.

```
cd DSC_CAN2L_PCI_LINUX_V1.7
```

```
./qt-opensource-linux-x86-5.2.1.run
```

Note: The Qt shared libraries should be installed only once.

Step-3:

PCI CAN Utility is based on the CANLib library. Copy the shared library to "/lib" directory.

```
cd CANLib
```

cp libCAN.so /lib

Please note : Step-3 should be done only once.

Step-4:

Load the PCI CAN interface driver using the command below from the dsc_can2_pci_driver directory where the zip file is extracted.

cd dsc_can2_pci_driver

insmod dsc_can2_pci.ko

Step-5:

Start the PCI CAN Utility using the command below from the CAN_Monitor directory where the zip file is extracted.

cd CAN_monitor

./CAN



This command will open the CAN interface utility. To start the CAN utility in the future, follow Steps 4 and 5 only.

8.2 Setting the Baud Rate

Using the CAN interface utility software, the baud rate for each port can be selected. On the desired CAN port, select the baud rate from the Baud Rate drop-down menu. After selecting the desired baud rate, press "**Connect**" to connect with specified baud rate as shown in below figure.

File CAN1 CAN2 Select Baud rate						
1						
Connection CAN1	CAN#1 Messages					
Baud Rate 500kbps Connect	Time	CanID	Len	Data		
Msg Type Standard Disconnected!						
Write Message Connect with	selected					
ID 0000 Len 8 Baud rate						
Data(Hex)						
00 00 00 00 00 00 00 00						
				_		
Add List Clear List Write Message					Save Log	Clear Log
Connection CAN2	CAN#2 Messages					
Baud Rate 500kbps Connect	Time	CanID	Len	Data		
Msg Type Standard						
Write Message						
ID 00000000 Len 8						
Data(Hex)						
00 00 00 00 00 00 00 00						
	\Im					
Add List Clear List Write Message					Save Log	Clear Log
CAN Monitor® 20)14 Diamond Systems Co	orp.	S	oftware Ve	ersion :1.0.4	
S Y S T E M S www.diamondsys	tems.com					

To change the baud rate, click on "**Disconnect**" and select a new baud rate.



8.3 Setting the CAN ID and Message Length

Set the CAN ID and CAN message length for each CAN port by entering the desired numbers into the ID and Len fields respectively for that port.

File CAN1 CAN2						
Connection CAN1	CAN#1 Messages					
Baud Rate 500kbps Disconnect	Time	CanID	Len	Data		
Msg Type Standard Connected!						
CAN ID Write Message ID 12 Len 8 Message Length Data(Hex) 1 2 3 4 5 6 7 8				\$		
Add List Clear List Write Message					Save Log	Clear Log
Connection CAN2	CAN#2 Messages					
Baud Rate 500kbps Connect	Time	CanID	Len	Data		
Msg Type Standard Disconnected!						
Write Message ID 00000000 Len 8 Data(Hex) 00 00 00 00 00 00 00 00						
Add List Clear List Write Message					Save Log	Clear Log
CAN Monitor® 2 www.diamondsy	2014 Diamond Systems (ystems.com	Corp.		Software \	/ersion :1.0.4	



8.4 Writing a Message

To write a message on a CAN port, define the CAN message by entering the desired data into the Data (Hex) fields. Then click on **"Write Message"** as shown in the below figure.

	CAN#1 Message	es			
ud Rate 500kbps Disconnect	Time	CanID	Len	Data	
g Type Standard Connected!	Wr ->57346045	12	8	12345678	
rite Message					
12 Len 8 CAN Messages					
ita(Hex)					
2 3 4 5 6 7 8	Click on write m	iessage			
	to transmit		N		
Add List Clear List Write Message			2	Save Log C	Clear Log
onnection CAN2	CAN#2 Massage	26			
ud Rate 500kbps Connect	Time	CanID	Len	Data	
g Type Standard					
rite Message					
00000000 Len 8					
ta(Hex)					
0 00 00 00 00 00 00 00					
Add Link Class Link Minite Manager				Save Log	Clear Log

To transmit to a different CAN ID, change the data in the CAN ID field, enter the desired data into the Data (Hex) fields, and click on "Write Message".

To change the message length, change the CAN message length to the new length, enter the desired data into the Data (Hex) fields, and click on "Write Message".

To transmit a different CAN message to the same CAN ID, change the CAN message to the desired data, and click on "Write Message".



8.5 Viewing Messages

Transmitted messages are listed in the CAN message box for the sending CAN port as shown in below figure.

Connection CAN1	CAN#1 Messag	es				
Baud Rate 500kbps Disconnect	Time	CanID	Len	Data		
Msg Type Standard Connected!	Wr ->57346045	12	8	12345678		
Write Message D 12 Len 8 Data(Hex)		Transn	nitted Messa	ige		
Add List Clear List Write Message			\$	Save	Log	Clear Log
Connection CAN2	CAN#2 Messag	es				
Baud Rate 500kbps Connect Msg Type Standard Disconnected!	Time	CanID	Len	Data		
Write Message						
D 00000000 Len 8						
Data/Haw						
00 00 00 00 00 00 00 00 00 00						



Received CAN messages are listed in the CAN message box for the CAN port receiving the message as shown in below figure.

Connection CAN1	CAN#1 Message	5			
aud Rate 500kbps Disconnect	Time	CanID	Len	Data	
Isg Type Standard Connected!	Rd ->57412959	111	8	11 22 33 44 55 66 77 88	
Write Message					
D 12 Len 8		Ň	Received CA	N Message	
Data(Hex)					
1 2 3 4 5 6 7 8					
Add List Clear List Write Message				Save Log	Clear Log
Connection CAN2	CAN#2 Message	s			
aud Rate 500kbps Disconnect	Time	CanID	Len	Data	
Isg Type Standard Connected!	Wr ->57412918	111	8	11 22 33 44 55 66 77 88	
Write Message		Carlo Carlo	Transmitte	d Message	
D 111 Len 8					
Data(Hex)					
11 22 33 44 55 66 77 88					
•					
Add List Clear List Write Message				Save Log	Clear Log
Add List Clear List Write Message				Save Log	Clear



9. CONFIGURE AND MANAGE THE PORTS IN LINUX

The CANLib library provides the set of APIs to configure and manage the CAN ports. The CANLib library can be used to build the CAN user application. It is a shared library built on top of Linux platform and using the driver provided functionality. To compile the CANLib shared library, use the below command

```
cd CANLib
make
```

All the CAN APIs prototypes are defined in the can.h file. This file is located in the CANLib directory. Include the can.h file in the user application to use all these APIs.

9.1 API to Configure and Manage CAN Ports

init_can0() & init_can1() : These function will initialize the CAN#0 & CAN#1 ports respectively.

Both these functions return the CAN file descriptor (fd). The return value of these functions should be retained for all subsequent APIs. Its prototypes are defined in the can.h file. Declare two CAN file descriptors and retains its return values.

```
#include ``can.h"
...
int
     can0 fd;
int
     can1 fd;
can0 fd = init can0() ;
if ( can0 fd < 0 )
{
   printf("Error while initializing the CAN#0\n") ;
   exit(0) ;
}
...
can1 fd = init can1() ;
if ( can1 fd < 0 )
{
   printf("Error while initializing the CAN#1\n") ;
      exit(0) ;
}
```



Baud Rate Configuration

set_baudrate() : This function configures the baud rate for the specified CAN port. By default it will not configure any baud rate. Below code snippet describes the usage of setting baud rate to 500kbps.

```
// Set 500k Baud rate for CAN#0
ret_val = set_baudrate(can0_fd, CAN_SPEED_500K ) ;
if ( ret_val < 0 )
{
    printf("Error while setting the baud rate \n") ;
    exit(0) ;
}
// Set 500k Baud rate for CAN#1
ret_val = set_baudrate(can1_fd, CAN_SPEED_500K ) ;
if ( ret_val < 0 )
{
    printf("Error while setting the baud rate \n") ;
    exit(0) ;
}</pre>
```

In the above code, can0_fd and can1_fd should contain the values returned by init can0 and init can1 function.

Use below macros for setting the different baud rates. These macros can also be found in can.h file.

CAN_SPEED_1M CAN_SPEED_800K CAN_SPEED_500K CAN_SPEED_250K CAN_SPEED_125K CAN_SPEED_100K CAN_SPEED_50K CAN_SPEED_20K CAN_SPEED_10K

CAN Transmit & Receive

can_tx() & can_rx() : These functions are be used to Transmit and Receive the CAN messages respectively.



CAN Transmit Prototype

```
int can tx( int can fd, unsigned char msgType, unsigned int can id, int
len,unsigned char *data) ;
Assign the appropriate values, before calling the can tx function.
can0 fd: CAN descriptor, return value from init can0() function
msgType = MSG STANDARD ; // or MSG EXTENDED.
can id = 0x12;
//if the msgType is MSG STANDARD then can id value should be in the range of 0 \times 0 to
0x7FF (11 bit value)
//Else if the msgType is MSG EXTENDED then can id value should be in the range 0 \times 0
to 0x1FFFFFFF (29 bit value)
len = 4 //This field can be of 0 to 8 ; // CAN Transmit Data Length (DLC)
data: This field depends on the above CAN Data length len field.
In this case, len is 4 then the CAN message data will be of 4 bytes long and each
byte can have values from 0x0 to 0xFF (8-bits )
      data[0] = 0x1A;
      data[1] = 0xAB;
      data[2] = 0x22;
      data[3] = 0x4D;
ret val = can tx(can0 fd, msgType, can id, dlc, data) ;
if ( ret val < 0 )
{
      printf("Error while transmitting the CAN message.\n") ;
      close(can1 fd) ;
      exit(0) ;
```

}

The above sample code will transmit the CAN standard message with CAN ID=0x12 of data length=4 and message data = $\{0x1A, 0xAB, 0x22, 0x4D\};$



CAN Receive Prototype

```
int can_rx(int can_fd, unsigned char *msgType, unsigned char *rx_data, unsigned int
*can_id, unsigned char *can_msg_len) ;
```

Pass the appropriate pointers for calling the can_rx function:

```
if ( can rx(can0 fd, &msgType, data, &can id, &dlc) )
      If (msgType == MSG STANDARD )
      {
            // Received message is CAN Standard Message.
      }
      else if
               (msgType == MSG EXTENDED)
      {
            // Received message is CAN Extended Message.
      }
      // dlc : Received CAN Data Length
      // can id : Will contain the CAN Message ID
      // Data of dlc length
      printf("ID=%x DLC=%d Data : ", can id, dlc) ;
      for (i=0; i< dlc; i++ )
            printf("%x ", data[i] ) ;
      printf("\n") ;
}
```

The sample example programs for both transmit and receive can be found in the CANLib directory for the reference.

9.2 Compiling User Application using CANLib Library

```
Export the library path using the following command:
export LD LIBRARY PATH=$LD LIBRARY PATH:/path-to-CANLib
```

To compile the user application, use the following command:

```
g++ can_app.c -lCAN -L/path-to-CANLib -o can_app
```



10. DRIVER INSTALLATION AND DEMO APPLICATION FOR WINDOWS

Download the DSC_CAN2_PCI_WIN_V1.0.3.zip file from the DS-MPE-CAN2L webpage (<u>http://www.diamondsystems.com/products/dsmpecan2l</u>). Click on *Windows driver package v1.0.3* in the Downloads section of the webpage. Use the following command to unzip the files:

Unzip DSC_CAN2L_PCI_WIN_V1.0.3.zip

The "DSC_CAN2_PCI_WIN_V1.0.3" directory contains the CAN application, library, and driver for testing the 2-CAN interfaces.

- 1. App : CAN Monitor demo application directory
- 2. dsc_can2_pci_src: Demo application source code directory.
- 3. dsc_can2_pci_driver: Windows CAN driver

10.1 Installing the PCI-CAN Driver

Step-1:

Open Windows command prompt with Administrator privileges. Click on Windows start button and type cmd in the search box and right click on the cmd.exe and click on "Run as administrator". Please refer to the screenshot given below.

(Lange Computer							
Programs (1)	·						
Documen	Open Run as administrator						
h cdc_c	Edit with Notepad++						
h cdc_c	Scan for Viruses						
h scsi_c 🇌	TortoiseSVN						
Microsoft GPIO AMOL	Pin to Taskbar Pin to Start Menu Restore previous versions						
🗟 RE: O 20141	Send to	e chart. rmware upgra					
🕙 OTAn	Cut	RE: OTA firmw					
Files (240	Сору						
or cmd	Delete						
🔗 cma	Open file location						
📕 cmd	Properties						
₽ See more rest	ults						
cmd	×	ut down 🕨					
🚳 🙆	📋 🝳 🚺		💁 🐗		1.1	- 🕅 🛱 🕼	16:23 30-12-2014

DS-MPE-CAN2L User Manual Rev A.1

www.diamondsystems.com



Step-2:

Change the working directory from the command prompt to the "DSC_CAN2_PCI_WIN_V1.0.3\ dsc_can2_pci_driver" directory where the software is copied.

Step-3:

Execute "install.bat". Please see the below screenshot for details. After executing, follow the next steps to install the driver.

D:\Diamond\IR1_2014_12_30\Driver>dir Volume in drive D is New Volume Volume Serial Number is B454-91CB Directory of D:\Diamond\IR1_2014_12_30\Driver 30-12-2014 16:37 (DIR) 30-12-2014 16:37 (DIR) 18-12-2012 11:21 77,824 devcon.exe 18-12-2012 11:21 221,257 DSCDIOFunctions.exe 18-12-2012 11:21 3,921 fpgpio96.inf 18-12-2012 11:21 15,872 fpgpio96.sys 18-12-2012 11:21 58 install.bat 18-12-2012 11:21 58 install.bat 18-12-2012 11:21 1,461,992 WdfCoInstaller01009.dll 6 File(s) 1,780,924 bytes 2 Dir(s) 95,328,833,536 bytes free D:\Diamond\IR1_2014_12_30\Driver>install.bat_	Administrator: C:\Windows\System32\cmd.exe
Directory of D:\Diamond\IR1_2014_12_30\Driver 30-12-2014 16:37 \DIR> . 30-12-2014 16:37 \DIR> . 18-12-2012 11:21 77.824 devcon.exe 18-12-2012 11:21 221,257 DSCDIOFunctions.exe 18-12-2012 11:21 3,921 fpgpio96.inf 18-12-2012 11:21 15.872 fpgpio96.sys 18-12-2012 11:21 58 install.bat 18-12-2012 11:21 58 install.bat 18-12-2012 11:21 1,461,992 WdfCoInstaller01009.dll 6 File(s) 1,780,924 bytes 2 Dir(s) 95,328,833,536 bytes free D:\Diamond\IR1_2014_12_30\Driver>install.bat_	D:\Diamond\IR1_2014_12_30\Driver>dir Volume in drive D is New Volume Volume Serial Number is B454-91CB
	Directory of D:\Diamond\IR1_2014_12_30\Driver 30-12-2014 16:37 (DIR) 30-12-2014 16:37 (DIR) 18-12-2012 11:21 77,824 devcon.exe 18-12-2012 11:21 221,257 DSCDIOFunctions.exe 18-12-2012 11:21 3,921 fpgpi096.inf 18-12-2012 11:21 15,872 fpgpi096.sys 18-12-2012 11:21 58 install.bat 18-12-2012 11:21 58 install.bat 18-12-2012 11:21 1,461,992 WdfCoInstaller01009.dll 6 File(s) 1,780,924 bytes 2 Dir(s) 95,328,833,536 bytes free D:\Diamond\IR1_2014_12_30\Driver>install.bat_

Step-4: Restart the system.



Step-5:

Check whether the driver is installed properly or not by opening the device manager.

Right Click on My Computer => Click on Properties => Device Manager. Please refer to the screenshot below.

Control Panel 🕨	All Control Panel Items 🕨 Systen	n 🔻 😽	Search Control Panel	٩
Control Panel Home Control Panel Home Control Panel Home Remote Settings System protection Advanced system settings	View basic information Windows edition Windows 7 Professional N Copyright © 2009 Microso Service Pack 1 Get more features with a n	about your computer ft Corporation. All rights reserve ew edition of Windows 7	ed.	
See also Action Center Windows Update	System Rating: Processor: Installed memory (RAM): System type: Pen and Touch: Computer name; Gomputer name: Full computer name: Computer description: Workgroup: Windows activation	System rating is not available Intel(R) Atom(TM) CPU N2600 2.00 GB 32-bit Operating System No Pen or Touch Input is avail workgroup settings diamond-PC diamond-PC WORKGROUP) @ 1.60GHz 1.60 GHz lable for this Display	ange settings
Tools	You must activate toda Droduct ID: 55041-050-045	ay. Activate Windows now	Îrav	+



If the driver is installed properly then the device manager will show the device as "PCIe Sample device" under "Sample Device" as shown in the screenshot below.

🚔 Device Manager	
File Action View Help	
diamond-PC	*
> 📲 Computer	
Disk drives	
🔉 📲 Display adapters	
🖓 🖓 Human Interface Devices	
De la IDE ATA/ATAPI controllers	
> - Keyboards	
🖟 📲 Mice and other pointing devices	
D - Monitors	H.
Network adapters	
Processors	
🔺 🎡 Sample Device	
PCIe Sample device	
⊿ -₁∰ System devices	
ACPI Fixed Feature Button	
ACPI Sleep Button	
ACPI Thermal Zone	
Intel(R) 82801G (ICH7 Family) PCI Express Root Port - 27D0	
Intel(R) 82801G (ICH7 Family) PCI Express Root Port - 27D2	
Intel(R) 82801G (ICH7 Family) PCI Express Root Port - 27D4	
Intel(R) 82801G (ICH7 Family) SMBus Controller - 27DA	
Intel(R) 82802 Firmware Hub Device	



10.2 Run the Windows Application

The application is stored in the "App" directory. Double click on "dsc_can2_pci.exe". The application window will open as shown in the below screenshot.

Connection CON1	CAN1 # Messages	
	Time Op Type ID Len Data	
Baud 500kbps Connect		
Msg Type Standard		
Write Message		
0 Len 8		
ata(Hex)		
•		
Add List Clear List Write Message	Save Log Clear Log	
Connection CAN2	CAN2 # Messages	
Baud 500kbns - Connect	Time Op Type ID Len Data	
Msg Type Standard		
Write Message		
0 Len 8		
ta(Hex)		
0 0 0 0 0 0		
-	< III	
	Save Log Gear Log	
Add Liet Clear Liet Write Message		



10.3 Setting the Baud Rate

Using the CAN interface utility software, the baud rate for each port can be selected. On the desired CAN port, select the baud rate from the Baud Rate drop-down menu. After selecting the desired baud rate, press "**Connect**" to connect with specified baud rate as shown in below figure.

	CAN1	# Mess	sages				
Connection CAN1	Time	Op	Туре	ID	Len	Data	_
Baud 500kbps Connect							
Msg Type Standard - Disconnected							
Write Message Connect	vith the Selecte	ed					
D 0 Len 8							
Data(Hex)							
0 0 0 0 0 0 0							
							
Add List Clear List Write Message			Save	Log	Clear Log	1	2
							_
	0.0110						
Connection CAN2	Time	# Hess	Ture		llon	Data	
Baud 500kbps Connect	Time	Op	Туре		Len		
Msg Type Standard - Disconnected							
Write Message							
D 0 Len 8							
Data(Hex)							
o lo lo lo lo lo lo							
•	•			1	11		
Add List Clear List Write Message			Save l	og	Clear Log	1	
						4	
CA	N Monitor@ 20	14 Diamo	nd System	s Corp.	Softw	are Version: 1.0.1	
	nu diamondevete	me com					

To change the baud rate, click on "**Disconnect**" and select a new baud rate.



10.4 Setting the CAN ID and Message Length

Set the CAN ID and CAN message length for each CAN port by entering the desired numbers into the ID and Len fields respectively for that port.

Connection CON4	CAN1	# Mes	sages			
	Time	Op	Туре	ID	Len	Data
Baud 500kbps Connect						
Msg Type Standard - Disconnected						
CANUD		th				
Write Message	incorage cong					
0 Len 8						
ata(Hex)						
· · · · · · ·						
			Save	Log	Clear Log	1
Add List Clear List Write Message						J .
	1					
Connection CAN2	CAN2	# Mes	sages			
Baud 500kbps - Connect	Time	Op	Туре	ID	Len	Data
Misg Type Standard						
	-					
Write Message						
0 Len 8						
ita(Hex)						
	•)	
						20 C
			C	32 I S	CI	



10.5 Writing a Message

To write a message on a CAN port, define the CAN message by entering the desired data into the Data (Hex) fields. Then click on **"Write Message"** as shown in the below figure.

Connection CAN1	CAN1	# Mes	sages			
	Time	Op	Туре	ID	Len	Data
Baud 500kbps Connect						
Msg Type Standard 💌 Disconnected						
Write Message						
) 0 Len 8	Can Messag	es				
ata(Hex)						
	Click on to Tr	Write Me ansmit	essage			
1 Alexandress of the second se						
Add List Clear List Write Message			Save	Log	Clear Log] .
Connection CAN2	CAN2	# Mes	sages			
	Time	Op	Туре	ID	Len	Data
Baud 500kbps _ Connect						
Msg Type Standard						
	4					
Write Message						
0 Len 8						
sta(Hex)						
				m	1	
	6		Savel	00	Clear Log	1
Add List Clear List Write Message			Javel	~ <u> </u>	Gear Luy	1
	CAN Monitor@ 201	4 Diamo	nd System	s Com	Softw	vare Version: 1.0.1
	www.diamondsyste	ms.com			- Sitte	

To transmit a different CAN ID, change data in the CAN ID field, enter the desired data into the Data (Hex) fields, and click on "Write Message".

To change the message length, change the CAN message length to the new length, enter the desired data into the Data (Hex) fields, and click on "Write Message".

To transmit a different CAN message to the same CAN ID, change the CAN message to the desired data, and click on "Write Message".



10.6 Viewing Messages

Transmitted and received messages are listed in the CAN message box for the sending CAN port as shown in below figure.

Connection CAN1	CANT # Messages
	Time Op Type ID Len Data 14/29500000 W/r Std 122 9 12/24 / 45 50 67 79 and PD
Baud 500kbps v Disconnect	1428580971 Wr Std 123 8 12 34 45 56 67 78 aa BB
Msg Type Standard - Connected	1428580972 Wr Std 123 8 12 34 45 56 67 78 aa BB
Write Message	
ID 123 Len 8	
Deta/Haw)	
· ·	
Add List Clear List Write Message	Save Log Clear Log
	CAND # Massage
Connection CAN2	
Baud 500kbps v Disconnect	
May Tage Standard and Connected	
Write Message	
ID 0 Len 0	
Data(Hex)	
Add List Clear List Write Message	Save Log Clear Log
	W Monitor@ 2014 Diamond Sustame Com Software Version: 1.0.6



11. API TO CONFIGURE AND MANAGE CAN PORTS ON WINDOWS

The source code for all windows user APIs to configure and manage the CAN ports are stored in "dsc_can2_pci_src" directory. User can include following files into their project directory. These API are dependent on CAN dll. User has to include these dll into their project directory.

common.h dscud.h dscud_os.h mpedaq0804.h pci_fpga.h public.h



The APIs are dependent on lib and dll. User has include the below APIs into their project.

dscud.dll dscud.lib

Project Settings -> Configuration Properties -> Linker -> Input ->

Modify Additional Dependecies : dscud.lib;setupapi.lib;%(AdditionalDependencies)

Configuration: Activ	/e(Release)	▼ Platfor	rm: Active(Win32)	•	Configuration Manager
 Common Propert Configuration Pro General Debugging VC++ Directo C/C++ Linker General Input Manifest F Debugging System Optimizati Embedded Advanced Command Manifest Tool Resources XML Docume Browse Inform Build Events Custom Build 	iies pperties ries iile g J IDL J Line nt Generator nation Step	Additional Dep Ignore All Defa Ignore Specific Module Definit Add Module to Embed Manag Force Symbol F Delay Loaded I Assembly Link	vendencies ult Libraries Default Libraries tion File o Assembly ed Resource File References DIIs Resource	dscud.lib;setupapi.lib;%(Ad	IditionalDependencies)
٠ III	•	Additional Depend Specifies additiona	dencies Il items to add to the link (command line [i.e. kernel32.lib]	

Opening the CAN device

OpenDevice(&GUID_DEVINTERFACE_FP_GPIO96, FILE_FLAG_OVERLAPPED);

This function should be called only once during initiation. It returns a valid handle if the driver detects the CAN device.

can_init(int can_ch): This API accepts the CAN interface number as an argument for the initialization. This will initialize CAN#0 or CAN#1. The value of can_ch should 0 for CAN#0 and 1 for CAN#1.

// Example : To initialize CAN#0 channel, pass the argument value as 0 to can_init function as shown below.

...

can_init (0) ; // Will initialize CAN#0 channel

Similarly, CAN#1 can be initialized by passing '1' as argument to can_init function as shown below



can_init (1); // Will initialize CAN#1 channel

Baud rate configuration.

set_baudrate(int can_ch, int baud rate): This API accepts the CAN interface number and the baudrate as arguments for configuring the baud rate of the specified CAN port.

// Set 500k Baud rate for CAN#0, Corresponding CAN channel values should be passed.

ret_val = set_baudrate(0, CAN_SPEED_500K);

Use below macros for setting the different baud rates. These macros can also be found in can.h file.

CAN_SPEED_1M CAN_SPEED_800K CAN_SPEED_500K CAN_SPEED_250K CAN_SPEED_125K CAN_SPEED_100K CAN_SPEED_50K CAN_SPEED_20K CAN_SPEED_10K

CAN Transmit & Receive :

FrameCANTxMsg() & check_rx_msg() : These function will be used to Transmit and Receive the CAN messages respectively.

CAN Transmit Prototype.

void FrameCANTxMsg(int can_ch, unsigned char msgType, unsigned int can_id, int len, unsigned char *data): Assign the appropriate values, before calling the FrameCANTxMsg function.

can_ch: CAN port number, 0 for CAN#0 and 1 for CAN#1.

msgType = MSG_STANDARD ; // or MSG_EXTENDED .

can_id = 0x12; //if the msgType is MSG_STANDARD then the value should be in the range of **0x0 to 0x7FF** (11 bit value)

//Else if the msgType is MSG_EXTENDED then the value should be in range 0x0 to 0x1FFFFFFF (29 bit value)

len = 4 //This field can be of 0 to 8 bytes length ; // CAN Transmit Data Length

data: This field depends on the len field.

If len is 4 then the CAN message data will be of 4 bytes long and each byte can range from 0x0 to 0xFF (8-Bit data)

data[0] = 0x1A ; data[1] = 0xAB ; data[2] = 0x22 ; data[3] = 0x4D ;

FrameCANTxMsg(0, msgType, can_id, dlc, data) ;



The above sample code will transmit the CAN Standard message with CAN ID=0x12 of data length=4 and message data = $\{0x1A, 0xAB, 0x22, 0x4D\}$;

CAN Receive Prototype.

int check_rx_msg(int can_ch, unsigned char *msgType, unsigned char *rx_data, unsigned int *can_id, unsigned char *can_msg_len);

Pass the appropriate pointers for calling the check_rx_msg function.

```
if ( check_rx_msg(0, &msgType, data, &can_id, &dlc) )
{
        If (msgType == MSG STANDARD)
        {
               // Received message is CAN Standard Message.
        }
        else if (msgType == MSG_EXTENDED)
        {
               // Received message is CAN Extended Message.
        }
// dlc : Received CAN Data Length
// can id : Will contain the received CAN Message ID
// Data of dlc length
printf("ID=%x DLC=%d Data : ", can_id, dlc) ;
for (i=0; i< dlc; i++ )
printf("%x ", data[i] ) ;
printf("\n") ;
}
```

The sample example programs for both transmit and receive can be found in the DSC_CAN2_PCI_V1.0.2_2015_01_13 directory for the reference.

Compile and build the CAN Application using VC++



12. SPECIFICATIONS

Number of ports	2 CAN 2.0B
Data rate	1Mbps
Number of receive buffers	31
Number of transmit buffere	1 high priority
	16 standard priority
Acceptance filters	16 programmable, 29-bit
Identifiers	11-bit and 29-bit
Modes	Dual-independent
	Dual-redundant
Isolation	500V port-to-port and input-to-output
Input power	+3.3VDC +/-5%
Power consumption	0.462W @ 3.3V
Software drivers	Windows XP
	Linux 2.6.16, 2.6.27, 2.6.31, and 2.6.32
Operating temperature	-40°C to +85°C
MTBF	1,583,210 hours at 20°C
Dimensions	50.95mm x 30mm (2" x 1.18")
Weight	8.5g (0.3oz)
RoHS Compliant	Yes



Our company network supports you worldwide with offices in Germany, Austria, Switzerland, the UK and the USA. For more information please contact:

Headquarters





FORTEC Elektronik AG Augsburger Str. 2b 82110 Germering

Phone: E-Mail: Internet: +49 89 894450-0 info@fortecag.de www.fortecag.de

Distec GmbH Office Vienna Nuschinggasse 12

1230 Wien Phone:

E-Mail:

Internet:

+43 1 8673492-0 <u>info@distec.de</u> <u>www.distec.de</u>

Distec GmbH Augsburger Str. 2b 82110 Germering

Phone: E-Mail: Internet:

+49 89 894363-0 info@distec.de www.distec.de

ALTRAC AG

Bahnhofstraße 3 5436 Würenlos

Phone: E-Mail: Internet: +41 44 7446111 info@altrac.ch www.altrac.ch

Display Technology Ltd.

Osprey House, 1 Osprey Court Hichingbrooke Business Park Huntingdon, Cambridgeshire, PE29 6FN

Phone: E-Mail: Internet: +44 1480 411600 info@displaytechnology.co.uk www. displaytechnology.co.uk

Apollo Display Technologies, Corp.

87 Raynor Avenue, Unit 1Ronkonkoma, NY 11779

Phone: E-Mail: Internet: +1 631 5804360 info@apollodisplays.com www.apollodisplays.com



Germany

Austria

Fortec Group Members







United Kingdom





A FORTEC GROUP MEMBER



