

PCI-M512/M512U

User Manual

Warranty

All products manufactured by ICP DAS are warranted against defective materials for a period of one year from the date of delivery to the original purchaser.

Warning

ICP DAS assume no liability for damages consequent to the use of this product. ICP DAS reserves the right to change this manual at any time without notice. The information furnished by ICP DAS is believed to be accurate and reliable. However, no responsibility is assumed by ICP DAS for its use, nor for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Copyright 2002 by ICP DAS. All rights are reserved.

Trademark

The names used for identification only may be registered trademarks of their respective companies.

Tables of Contents

1. INTRODUCTION.....	4
1.1 FEATURES	4
1.2 SPECIFICATIONS	5
1.3 PRODUCT CHECK LIST	7
1.4 INSTALLATION QUICK START.....	8
1.4.1 Step 1: Software Installation.....	8
1.4.2 Step 2: get demo software and manual.....	10
1.4.3 Step 3: Hardware Installation	11
1.4.4 Step 4: Hardware Diagnostic	11
1.4.5 Step 5: Muti-Board Diagnostic.....	16
2. HARDWARE CONFIGURATION	17
2.1 BOARD LAYOUT.....	17
2.2 IDs OF PCI-M512.....	18
2.3 BLOCK DIAGRAM OF D/I/O.....	19
2.4 BATTERY STATUS INDICATORS	20
2.5 BLOCK DIAGRAM OF SRAM.....	22
2.6 DAUGHTER BOARDS	23
2.6.1 DB-16P Isolated Input Board.....	23
2.6.2 DB-16R Relay Board	24
2.6.3 DB-24PR, DB-24POR, DB-24C	25
2.7 PIN ASSIGNMENT	26
3. DLL DRIVER.....	27
3.1 FIND THE BOARD NUMBER.....	31
3.2 FUNCTIONS OF TEST.....	32
3.2.1 PCIM512_FloatSub	32
3.2.2 PCIM512_ShortSub	32
3.2.3 PCIM512_IntSub	33
3.2.4 PCIM512_GetDllVersion	33
3.3FUNCTIONS OF DRIVER INITIALIZATION.....	34
3.3.1PCIM512_DriverInit	34
3.3.2PCIM512_OpenBoard	34
3.3.3PCIM512_DetectBoards.....	35
3.3.4PCIM512_ReadBoardId	36
3.3.5PCIM512_ReadBoardStatus.....	37
3.3.6PCIM512_CloseBoard.....	38
3.3.7PCIM512_CloseAll.....	38

3.4 FUNCTIONS OF SRAM READ/WRITE.....	39
3.4.1 <i>PCIM512_WriteSramByte</i>	39
3.4.2 <i>PCIM512_WriteSramWord</i>	40
3.4.3 <i>PCIM512_WriteSramDword</i>	41
3.4.4 <i>PCIM512_ReadSramByte</i>	42
3.4.5 <i>PCIM512_ReadSramWord</i>	43
3.4.6 <i>PCIM512_ReadSramDword</i>	44
3.5 FUNCTIONS OF D/I/O READ/WRITE.....	45
3.5.1 <i>PCIM512_WriteToDo</i>	45
3.5.2 <i>PCIM512_ReadFromDi</i>	46
4. DEMO PROGRAM.....	47
4.1 PROGRAM ARCHITECTURE	48
4.2 PROBLEMS REPORT	49

1. Introduction

The PCI-M512/M512U boards are battery-backup SRAM and digital I/O boards. The PCI-M512 is for installing in a 5 V PCI slot only; the PCI-M512U is a new PCI-M512 for Universal PCI (3.3 and 5 V). The PCI-M512 and the PCI-M512U use the same driver and software; users do not need to modify their control program when replacing the PCI-M512 with the PCI-M512U.

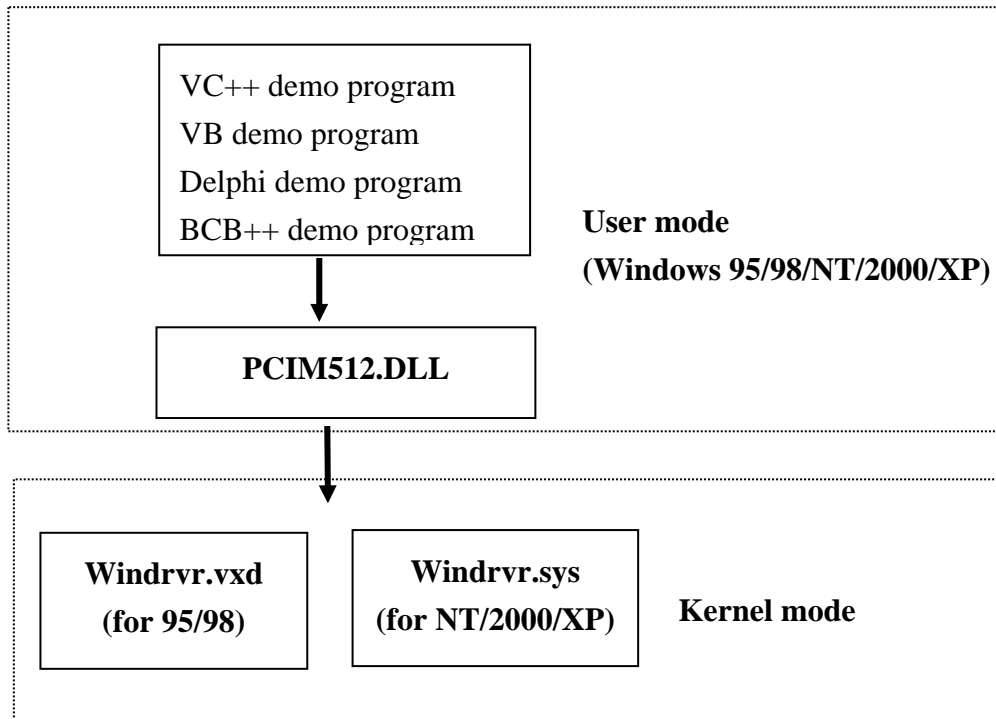
1.1 Features

- On-board 512 KB SRAM
- Two Li-batteries, BT1 & BT2, for battery-backup the data of SRAM
- Two indicators, low-battery & bad-battery, for battery BT1.
- Another two indicators, low-battery & bad-battery, for battery BT2.
- 16 bits general purpose TTL-compatible D/O or relay (with daughter board DB-16R or DB-24PR)
- 12 bits general propose TTL-compatible D/I or isolated input (with daughter board DB-16P)
- DLL library for windows 95/98/NT/2000/XP
- Demo Program for VC, VC++, VB, Delphi, BCB++

1.2 Specifications

Model Name	PCI-M512	PCI-M512U
Digital Input		
Channels	12	
Compatibility	5 V/TTL	
Input Voltage	Logic 0: 0.8 V max. Logic 1: 2.0 V min.	
Response Speed	1.4 MHz(Typical)	
Digital Output		
Channels	16	
Compatibility	5 V/TTL	
Output Voltage	Logic 0: 0.4 V max. Logic 1: 2.4 V min.	
Output Capability	Sink: 0.8 mA @ 0.8 V Source: -2.4 mA @ 2.0 V	
Response Speed	1.4 MHz(Typical)	
Special		
SRAM Size	512 KB	
Li-Battery	BT1 & BT2	
Battery Status bits	BT1 Low, BT1 Bad, BT2 Low, BT2 Bad(low voltage=2.3 V, bad voltage=2.1 V)	
LED Indicators	BT1 Low (Green), BT1 Bad (Red) BT2 Low (Green), BT2 Bad (Red)	
General		
Bus Type	5 V PCI, 32-bit, 33 MHz	3.3 V / 5 V Universal PCI, 32-bit, 33 MHz
Data Bus	16-bit(DI/DO) ,32-bit(Memory)	
I/O Connector	Male 20-bit ribbon x 2	
Dimensions (L x W)	140 mm x 90 mm	
Power Consumption	420 mA @ +5 V	
Operating Temperature	-20 ~ 60 °C	
Storage Temperature	-40 ~ 85 °C	
Humidity	0 ~ 90% RH, non-condensing	

There are many demo programs, written in VC++, VB, Delphi, and BCB++, given in the companion CD. These demo programs call the DLL, PCIM512.DLL, to access the PCI-M512 hardware. The PCIM512.DLL will call the kernel driver, Windrvr.vxd or Windrvr.sys as follows:



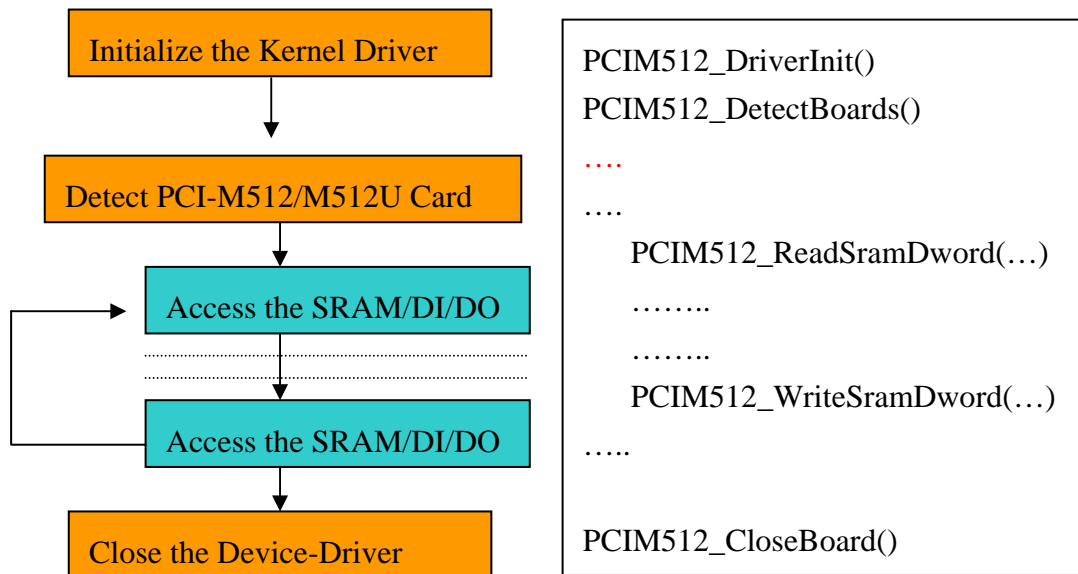
The install shields installs kernel driver and DLL driver system. **All demo program & DLL are the same for 95/98/NT/2000/XP.** However, the kernel driver is different for different system as follows:

- for windows 95/98 → will copy **WINDRVR.VXD** to
C:\WIN95\SYSTEM\VMM32
- for windows NT/2000/XP → will copy **WINDRVR.SYS** to
C:\WINNT\SYSTEM32\DRIVERS

All DLL & demo program will not work if the kernel driver is not installed correctly. The install shields will copy the correct kernel driver to the correct position if you select the correct O.S. (Windows 95/98, NT, 2000, XP).

Refer to **“Calling DLL functions in VB_VC_Delphi_BCB.pdf”** for more information about how to call the DLL functions with VC++6, VB6, Delphi3 and Borland C++ Builder 3. You can find it in the companion CD:\NAPDOS\PCI\Manual.

The software architecture is given as follows:



1.3 Product Check List

Your package includes the following items:

- One PCI-M512/PCI-M512U board
- One companion CD for software driver
- One Quick Start Guide

It is recommended to read the Quick Start Guide first. All the necessary and essential information are given in the Quick Start Guide as follows:

- Where to get the software driver, demo programs and other resources.
- How to install the software.
- How to test the card.

Attention !

If any of these items are missing or damaged, contact the dealer from whom you purchased the product. Save the shipping materials and carton in case you want to ship or store the product in the future.

1.4 Installation Quick Start

The PCI-M512 software supports Windows 95/98/NT/2000/XP. The recommended installation steps are given in Sec 1.2.1 ~ Sec. 1.2.4

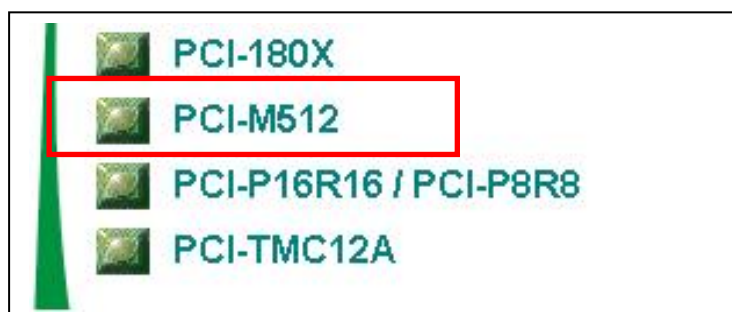
1.4.1 Step 1: Software Installation

Step 1: insert the companion CD into the CD-ROM driver. It will auto run as follows:

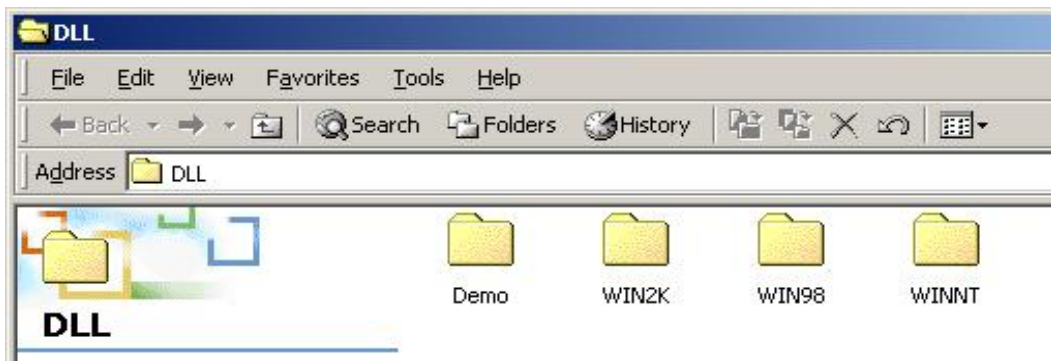


Step 2: click the first item, **PCI Bus DAQ Card**

Step 3: click the item, **PCI-M512**

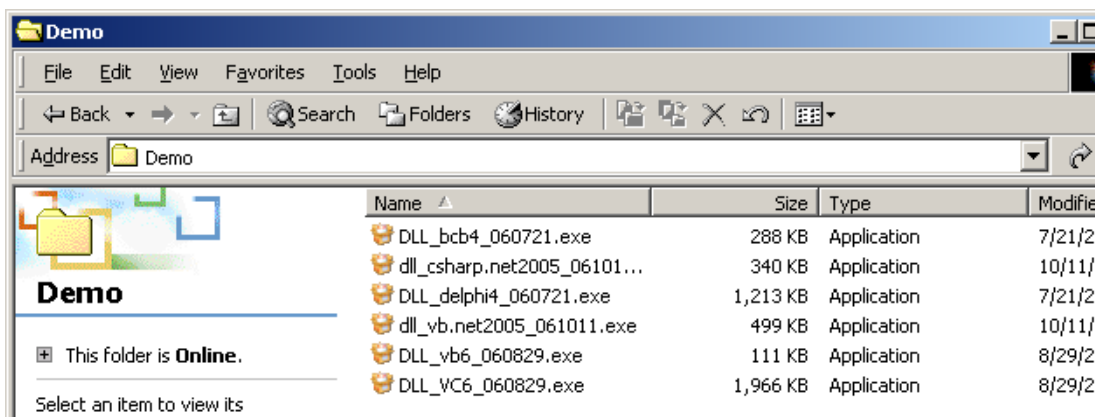


Step 4: select the driver for your OS and install it.



Step 5: reboot your PC.

Step 6: click into demo sub-directory to get the demo program



Then the install shields will install the kernel driver and DLL driver to system. **All demo programs & DLL are the same for Windows 95/98/NT/2000/XP.** However, the kernel driver is different for different systems as follows:

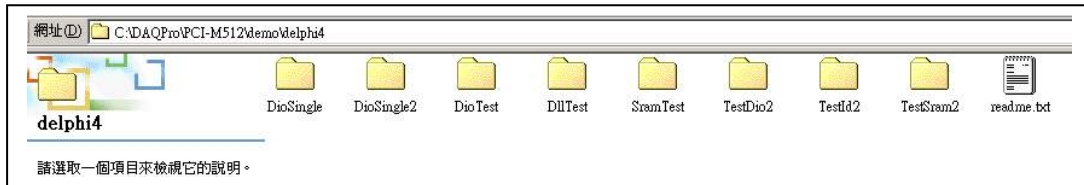
- for windows 95/98 → will copy **WINDRVR.VXD** to
C:\WIN95\SYSTEM\VMM32
- for windows NT/2000/XP → will copy **WINDRVR.SYS** to
C:\WINNT\SYSTEM32\DRIVERS

All DLL & demo programs will not work if the kernel driver is not installed correctly. The install shields will copy the correct kernel driver to the correct position if you select the correct O.S. (95/98, NT, 2000, XP).

1.4.2 Step 2: get demo software and manual

Now you had installed the driver and rebooted your PC. The Plug and Play had run automatically and you see the PCI-M512 installation information in “Device Manager”.

You also got the demo programs already. For example, after running self-extracting archive “dll_Delphi4_060721.exe”, the demo programs will show as follows:

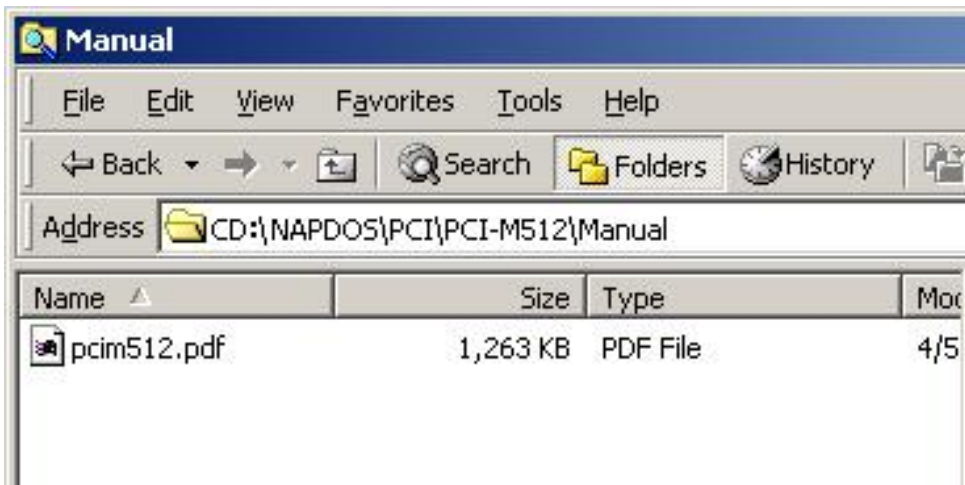


DioSingle → Test D/I/O of the PCI-M512 (only one program can access this board)
DioSingle2 → Test D/I/O of the PCI-M512 (one time only one program can occupy one board)
DioTest → Test D/I/O of the PCI-M512
DllTest → Test DLL driver & detect the PCI-M512

SramTest → Test NVSRAM of the PCI-M512
TestDio2 → Write D/O then read D/I of two PCI-M512 boards
TestId2 → Show IDs of two PCI-M512 boards
TestSram2 → Show Sram of two PCI-M512 boards

Get the PCI-M512 hardware manual:

The hardware manual “pcim512.pdf” is in the companion CD
CD:\NAPDOS\PCI\PCI-M512\Manual



Get the FAQ documentation:

If you have any problem about install hardware, driver and software, please refer to the companion CD:\NAPDOS\PCI\Manual. Those documentations include “Software Installation Guide”, “Trouble Shooting in Win32 Resource Conflict” and solutions of software frequently asked questions.

1.4.3 Step 3: Hardware Installation

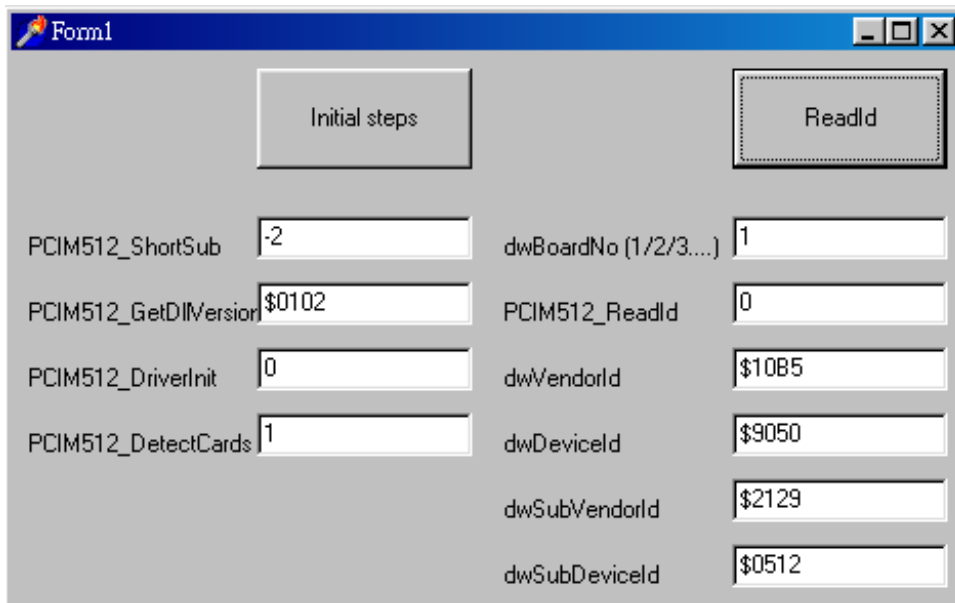
Step 1: Install your PCI-M512/M512U to PC

Step 2: power on your PC

Step 3: Now 98/2000/XP will find a PCI-M512/M512U card & ask you to provide a software driver. Refer to ”[PCI_ISA_PnP_Driver_Installation_in_Win9x_2K_XP.pdf](#)” for more information. You can find the documentation in the companion CD:\NAPDOS\PCI\Manual

1.4.4 Step 4: Hardware Diagnostic

Step 1: run **DllTest** of Delphi demo program as follows: (Sec. 1.2.2)



Initial steps	ReadId
PCIM512_ShortSub: -2	dwBoardNo (1/2/3....): 1
PCIM512_GetDllVersion: \$0102	PCIM512_ReadId: 0
PCIM512_DriverInit: 0	dwVendorId: \$10B5
PCIM512_DetectCards: 1	dwDeviceId: \$9050
	dwSubVendorId: \$2129
	dwSubDeviceId: \$0512

- Click **Initial Steps** first to check the kernel driver, DLL & PCIM512-DetectBoards()
- Check that the value of **PCIM512_DriverInit** is 0
- Click **ReadBoardId** to show the IDs of selected PCI-M512 in this PC

- Key-in new *dwBoardNo* to show IDs of another PCI-M512 as follows:.

The screenshot shows a software window titled "Form1" with a blue title bar. Inside the window, there are two buttons: "Initial steps" and "ReadId". Below the buttons are several input fields arranged in two columns. The left column contains: "PCIM512_ShortSub" with value "-2", "PCIM512_GetDIMVersion" with value "\$0102", "PCIM512_DriverInit" with value "0", and "PCIM512_DetectCards" with value "2". The right column contains: "dwBoardNo (1/2/3...)" with value "2", "PCIM512_ReadId" with value "0", "dwVendorId" with value "\$10B5", "dwDeviceId" with value "\$9050", "dwSubVendorId" with value "\$2129", and "dwSubDeviceId" with value "\$0512".

Refer to Sec. 2.2 for more information about IDs of PCI-M512 as follows:

- **Vendor ID** = **10B5**
- **Device ID** = **9050**
- **Sub-vendor ID** = **2129**
- **Sub-device ID** = **0512**

Step 2: run **DioTest** of Delphi demo program as follows: (**Sec. 1.2.2**)

The screenshot shows a Delphi form titled 'Form1'. It contains several controls: two buttons at the top labeled 'Digital Output DDDD' and 'Digital Output \$5555'; a text box for 'dwBoardNo (1/2/3/....)' with the value '1'; a text box for 'PCIM512_DriverInIt' with the value '0'; a text box for 'PCIM512_DetectCards' with the value '1'; a text box for 'Digital Output Data, DDDD' with the value '\$AAAA'; and a text box for 'Digital Input' which is currently empty.

- Click **Digital Output DDDD** to write to D/O & Read D/I as follows: (write-data is given in **Digital Output Data DDDD**)

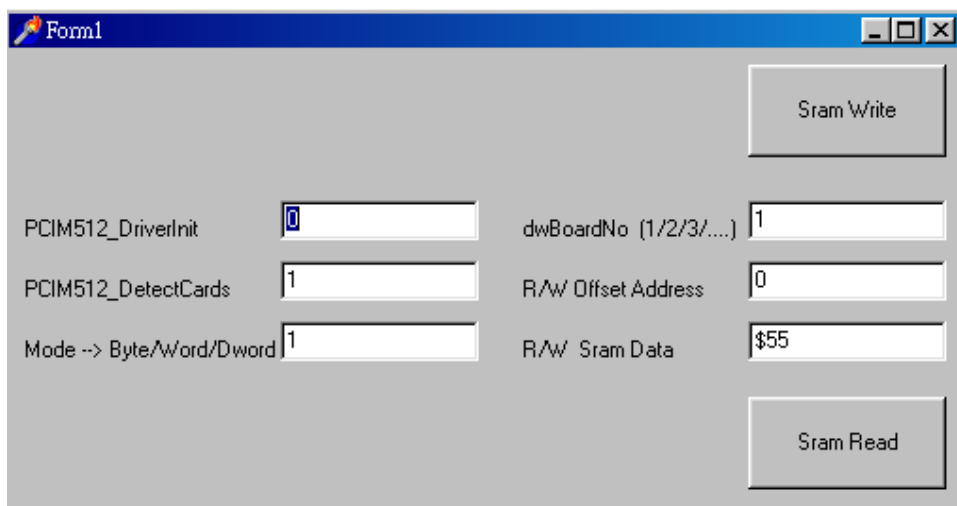
The screenshot shows the same Delphi form 'Form1' after the 'Digital Output DDDD' button has been clicked. The 'Digital Output DDDD' button is now highlighted with a dotted border. The 'Digital Input' text box now contains the value '\$AAA0'.

- Check that lowest 4-bits equal 0. These 4-bits are battery status bits. Refer to Sec. 2.4 for more information.
- Click **Digital Output \$5555** to write 0x5555 to D/O & Read D/I as follows:

The screenshot shows the Delphi form 'Form1' after the 'Digital Output \$5555' button has been clicked. The 'Digital Output \$5555' button is now highlighted with a dotted border. The 'Digital Input' text box now contains the value '\$5550'.

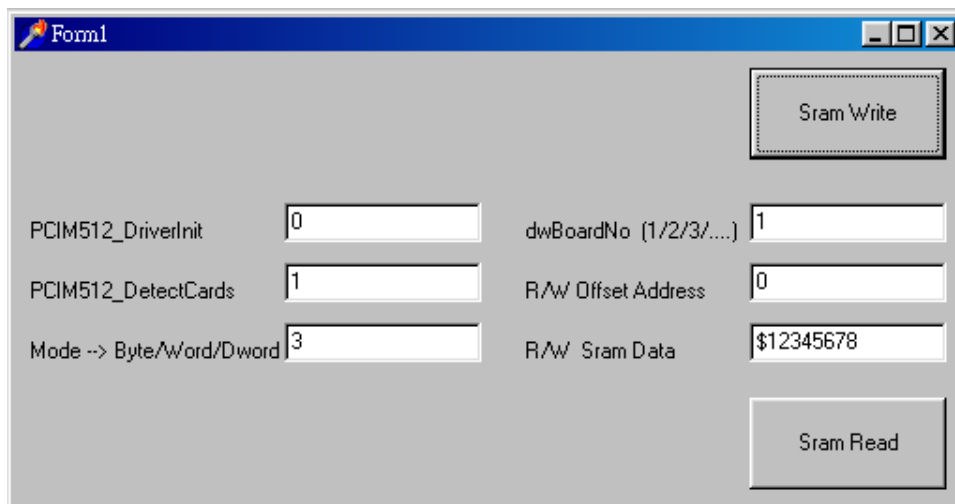
- Key-in new **dwBoardNo** to read/write to other PCI-M512. Refer to Sec. 3.1 for more information.

Step 3: run **SramTest** of Delphi demo program as follows: (Sec. 1.2.2)



PCIM512_DriverInit	0	dwBoardNo (1/2/3/....)	1
PCIM512_DetectCards	1	R/W Offset Address	0
Mode --> Byte/Word/Dword	1	R/W Sram Data	\$55

- Click **Sram Write** to write to SRAM (write-data is given in **R/W Sram Data**, offset address of SRAM is given in **R/W Offset Address**, byte/word/dword read/write is given in **Mode → Byte/Word/Dword**)
- Click **Sram Read** to read SRAM (read-data is given in **R/W Sram Data**, offset address of SRAM is given in **R/W Offset Address**, byte/word/dword read/write is given in **Mode → Byte/Word/Dword**)
- Key-in new **dwBoardNo** to read/write to other PCI-M512/M512U. Refer to Sec. 3.1 for more information.
- Write 0x12345678 to offset address 0 of SRAM as follows:



PCIM512_DriverInit	0	dwBoardNo (1/2/3/....)	1
PCIM512_DetectCards	1	R/W Offset Address	0
Mode --> Byte/Word/Dword	3	R/W Sram Data	\$12345678

- Read one byte of SRAM at offset address 0 as follows:

The screenshot shows the Fom1 application window with the following configuration:

PCIM512_DriverInit	0	dwBoardNo (1/2/3/...)	1
PCIM512_DetectCards	1	R/W Offset Address	0
Mode --> Byte/Word/Dword	1	R/W Sram Data	\$0078

Buttons: "Sram Write" (top right), "Sram Read" (bottom right, highlighted with a dashed border).

- Read one word of SRAM at offset address 0 as follows:

The screenshot shows the Fom1 application window with the following configuration:

PCIM512_DriverInit	0	dwBoardNo (1/2/3/...)	1
PCIM512_DetectCards	1	R/W Offset Address	0
Mode --> Byte/Word/Dword	2	R/W Sram Data	\$5678

Buttons: "Sram Write" (top right), "Sram Read" (bottom right, highlighted with a dashed border).

- Read one dword of SRAM at offset address 0 as follows:

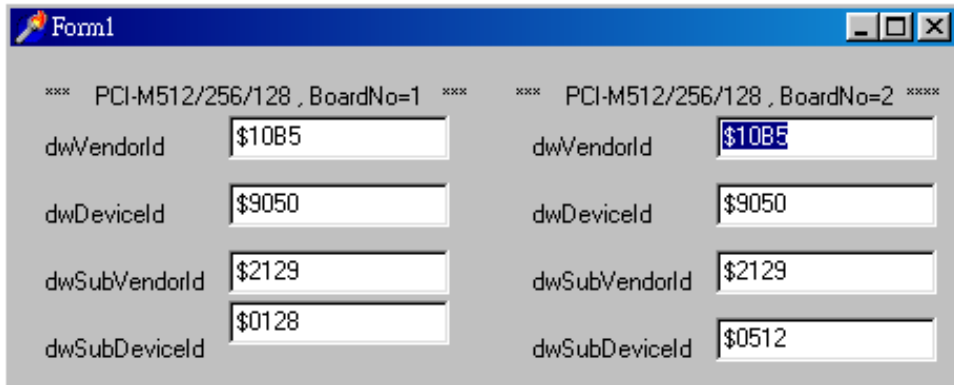
The screenshot shows the Fom1 application window with the following configuration:

PCIM512_DriverInit	0	dwBoardNo (1/2/3/...)	1
PCIM512_DetectCards	1	R/W Offset Address	0
Mode --> Byte/Word/Dword	3	R/W Sram Data	\$12345678

Buttons: "Sram Write" (top right, highlighted with a dashed border), "Sram Read" (bottom right).

1.4.5 Step 5: Multi-Board Diagnostic

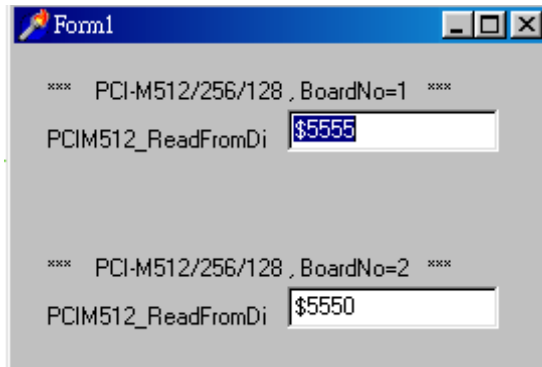
Step 1: run **TestId2** of Delphi demo program to read & show IDs of two PCI-M512s as follows: (Sec. 1.2.2)



The screenshot shows a Delphi application window titled 'Form1'. It displays two columns of data for PCI-M512 devices on two different boards. The first column is for BoardNo=1 and the second for BoardNo=2. Each column contains four text boxes representing different ID fields: dwVendorId, dwDevicId, dwSubVendorId, and dwSubDevicId. The values are as follows:

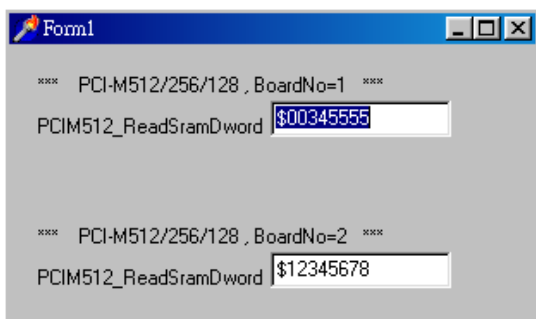
Field	BoardNo=1	BoardNo=2
dwVendorId	\$10B5	\$10B5
dwDevicId	\$9050	\$9050
dwSubVendorId	\$2129	\$2129
dwSubDevicId	\$0128	\$0512

Step 2: run **TestDIO2** of Delphi demo program to read/write D/I/O of two PCI-M512s as follows: (Sec. 1.2.2)



The screenshot shows the 'Form1' window with two sections. The first section is for BoardNo=1 and shows a text box labeled 'PCIM512_ReadFromDi' with the value '\$5555'. The second section is for BoardNo=2 and shows a text box labeled 'PCIM512_ReadFromDi' with the value '\$5550'.

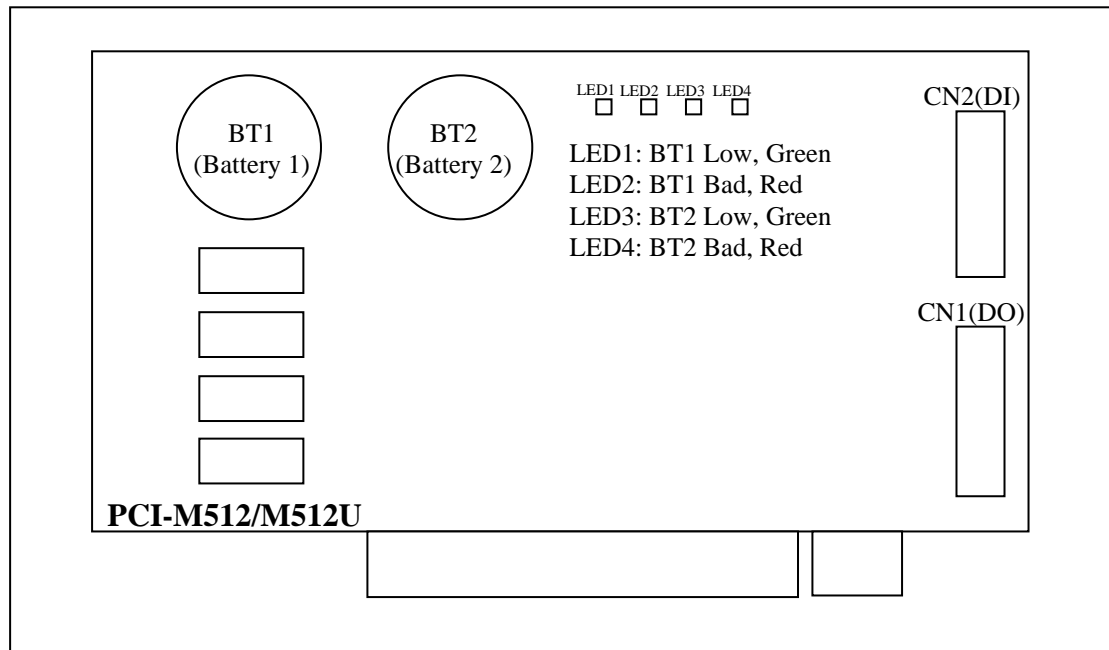
Step 3: run **TestSram2** of Delphi demo program to read/write SRAM of two PCI-M512s as follows: (Sec. 1.2.2)



The screenshot shows the 'Form1' window with two sections. The first section is for BoardNo=1 and shows a text box labeled 'PCIM512_ReadSramDword' with the value '\$00345555'. The second section is for BoardNo=2 and shows a text box labeled 'PCIM512_ReadSramDword' with the value '\$12345678'.

2. Hardware configuration

2.1 Board Layout



Note:

1. If BT1 & BT2 are both OK, LED1 ~ LED4 will be OFF.
2. If BT1 is lower than 2.3 V, the green LED1 will be ON.
3. If BT1 is lower than 2.1 V, the green LED1 & red LED2 will be ON.
4. If BT2 is lower than 2.3 V, the green LED3 will be ON.
5. If BT2 is lower than 2.1 V, the green LED3 & red LED4 will be ON.
6. If the PC power is off, the power control circuit will **select the battery with the higher voltage** to backup SRAM. If both BT1 & BT2 are bad, the data stored in SRAM may be lost.
7. SRAM can keep all stored data if either BT1 or BT2 is higher than 2 V.
8. **If either BT1 or BT2 is bad, it is recommended to replace both BT1 & BT2 with new batteries.**

2.2 IDs of PCI-M512

The IDs of the PCI-M512/M512U are given as follows:

- **Vendor ID = 10B5**
- **Device ID = 9050**
- **Sub-vendor ID= 2129**
- **Sub-device ID = 0512**

The plug&play BIOS will assign proper resources to every PCI-M512 card in the power-on stage. The software driver of the PCI-M512 will use these IDs to access the hardware.

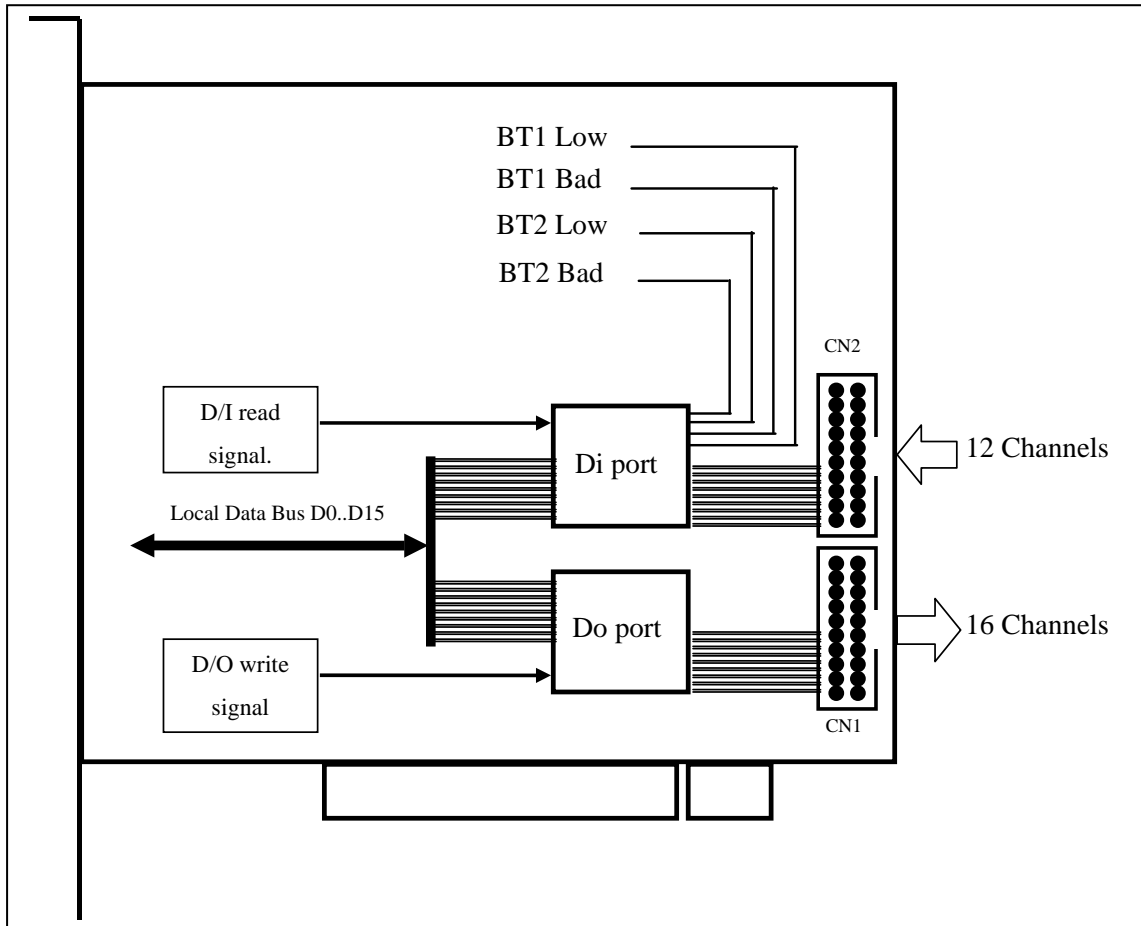
Users must use **PCIM512_DetectBoards()** to detect all PCI-M512 boards first. Then user can use the following commands to access SRAM or D/I/O of detected board.

	Read Function	Write Function
SRAM R/W Byte	PCIM512_ReadSramByte(...)	PCIM512_WriteSramByte(...)
SRAM R/W Word	PCIM512_ReadSramWord(...)	PCIM512_WriteSramWord(...)
SRAM R/W Dword	PCIM512_ReadSramDword(...)	PCIM512_WriteSramDword(...)
D/I/O R/W Word	PCIM512_ReadFrom Di(...)	PCIM512_WriteToDo(...)

PCIM512_ReadBoardId(dwBoardNo,*dwVendorId, *dwDeviceId, *dwSubVendorId, *dwSubDeviceId) is designed to read back the IDs of detected PCI-M512/M512U boards.

2.3 Block Diagram of D/I/O

The PCI-M512/M512U provides 16 channels of digital input and 16 channels of digital output. All levels are TTL compatible. The connections diagram and block diagram are given as follows:

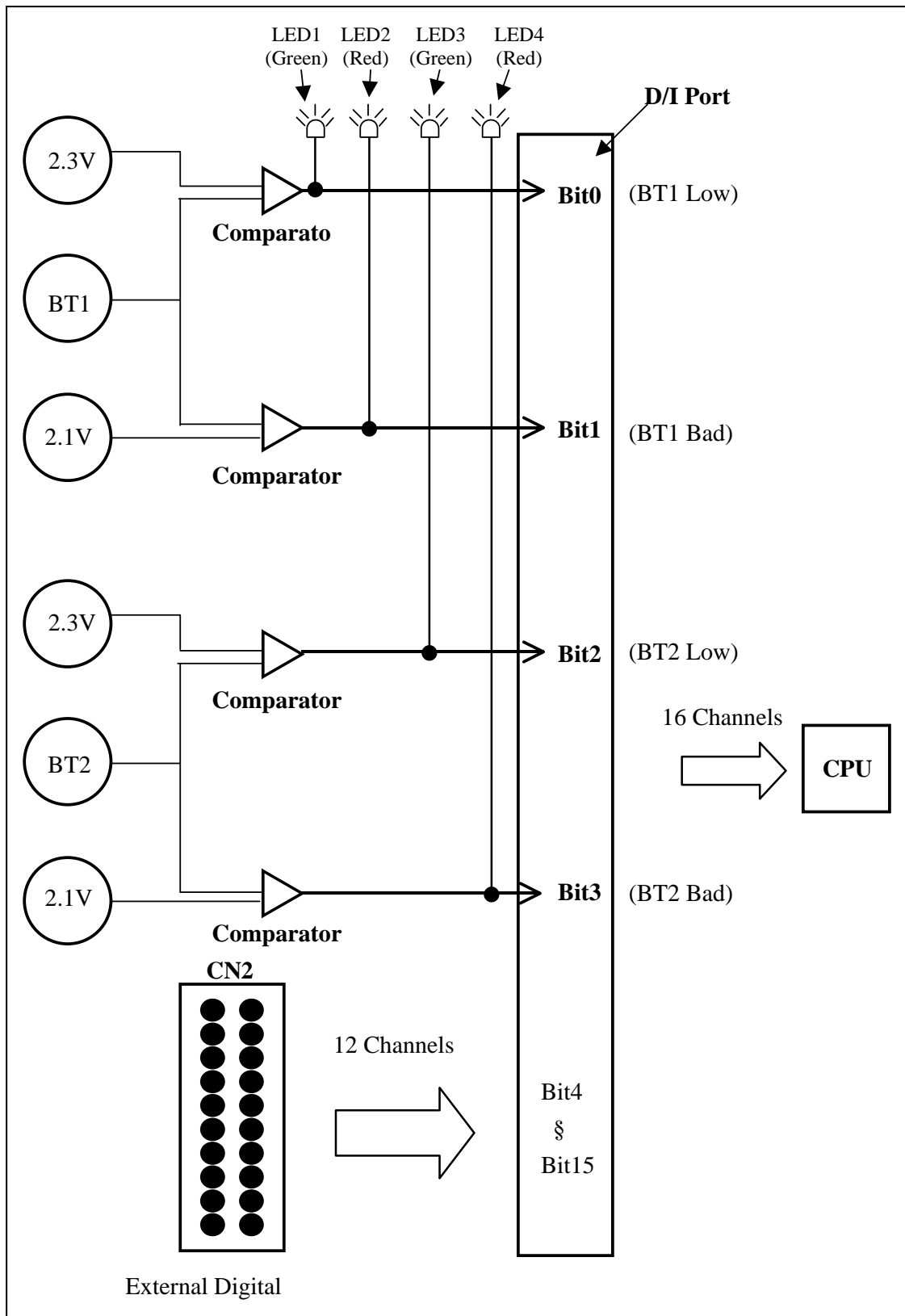


The D/O port can be connected to the DB-16R or DB-24PR. The DB-16R is a 16 channel Relay output board. The DB-24R is a 24 channel Power Relay output board. (note: only 16 channels of these 24 channels are valid).

The D/I port can be connected to the DB-16P. The DB-16P is a 16-channel isolated digital input daughter board. **Note: the starting 4 channels are used by battery status bits as the above diagram shows.**

All D/I & D/O are TTL compatible.

2.4 Battery Status Indicators



The initial voltage of BT1 will be larger than 3.0 V. If this voltage drops to 2.3 V, BT1 can still keep the stored data in SRAM for months. **It is recommended to replace both BT1 & BT2 when either BT1 or BT2 drops to 2.3 V.** If this voltage drops to 2.1 V, the BT1 can still keep the stored data in SRAM for weeks. **You should replace both BT1 & BT2 a.s.a.p. if either BT1 or BT2 drops to 2.1 V.**

The action table is given as follows:

Battery voltage status	LED status	D/I port status
BT1 > 2.3 V	LED1 OFF, LED2 OFF	Bit0=0, Bit1=0
2.3 V>BT1>2.1 V	LED1 ON, LED2 OFF	Bit0=1, Bit1=0
2.1 V>BT1	LED1 ON, LED2 ON	Bit0=1, Bit1=1
BT2 > 2.3 V	LED3 OFF, LED4 OFF	Bit2=0, Bit3=0
2.3 V>BT2>2.1 V	LED3 ON, LED4 OFF	Bit2=1, Bit3=0
2.1 V>BT2	LED3 ON, LED4 ON	Bit2=1, Bit3=1

You can call *PCIM512_ReadFromDi(DWORD dwBoardNo, WORD *Data)* to read the 16-bit data. Refer to Sec. 3.6 for more information.

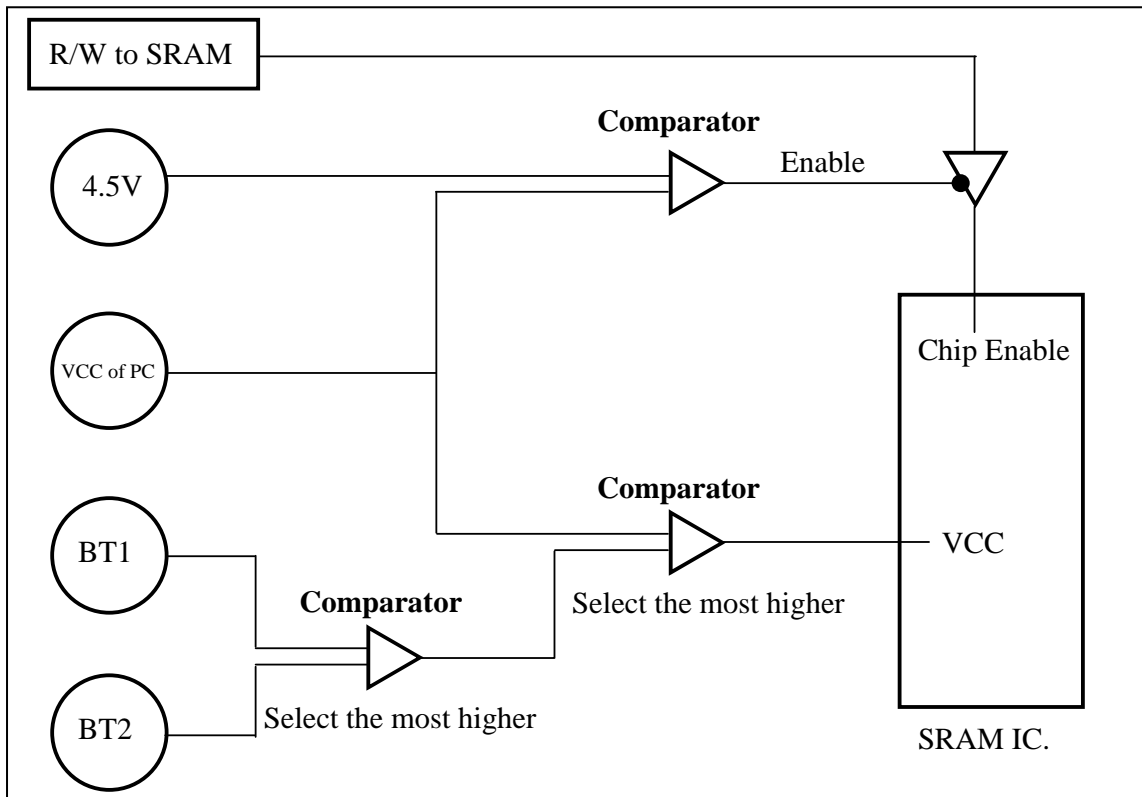
The lowest 4 bits, Bit0 ~ Bit3, are battery status bits. The other 12 bits, Bit4 ~ Bit15, are external D/I signals. You can connect a DB-16P to CN2 for sensor input. Refer to Sec. 2.5.1 for more information.

If you find that either BT1 or BT2 is in low-battery state, it is recommended to replace both BT1 & BT2 as follows:

1. prepare 2 new batteries for new BT1 & new BT2
2. **power up PC (not power off)**
3. replace the old BT1 with the new BT1
4. replace the old BT2 with the new BT2

Note: it is recommended to replace both BT1 & BT2 at the same time.

2.5 Block Diagram of SRAM



The power supply of SRAM is selected from the highest voltage of PC-VCC, BT1 & BT2. The initial voltage of BT1 & BT2 is about 3 V. If the PC is power on, the PC-VCC will be about 5 V. If the PC is off, the PC-VCC will be about 0 V. So when the PC is power on, the PC-VCC will supply power to SRAM. In this condition, BT1 & BT2 will preserve their battery for later usage.

If PC's power is off, the battery with higher voltage will supply power to SRAM. The stored data of SRAM will remain if the power is larger than 2.0 V. So, either BT1 or BT2 must be higher than 2.0 V to keep the SRAM data.

There is one low-battery indicator & one bad-battery indicator for both BT1 & BT2. Refer to Sec. 2.3 for more information. If you find that one of BT1 or BT2 is in low-battery state, it is recommended to replace both BT1 & BT2 as follows:

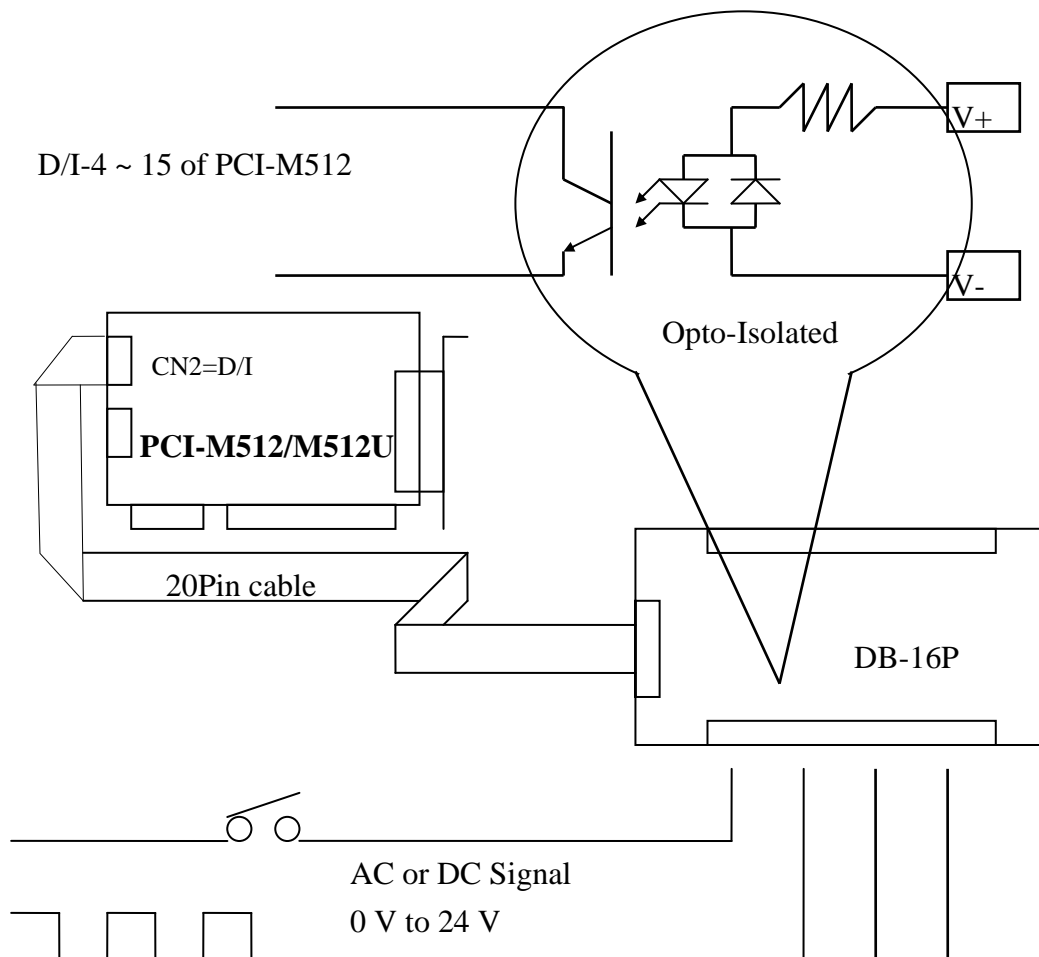
1. prepare 2 new batteries for new BT1 & new BT2
- 2. power up PC (not power off)**
3. replace the old BT1 with the new BT1
4. replace the old BT2 with the old BT2

Note: it is recommended to replace both BT1 & BT2 at the same time.

2.6 Daughter Boards

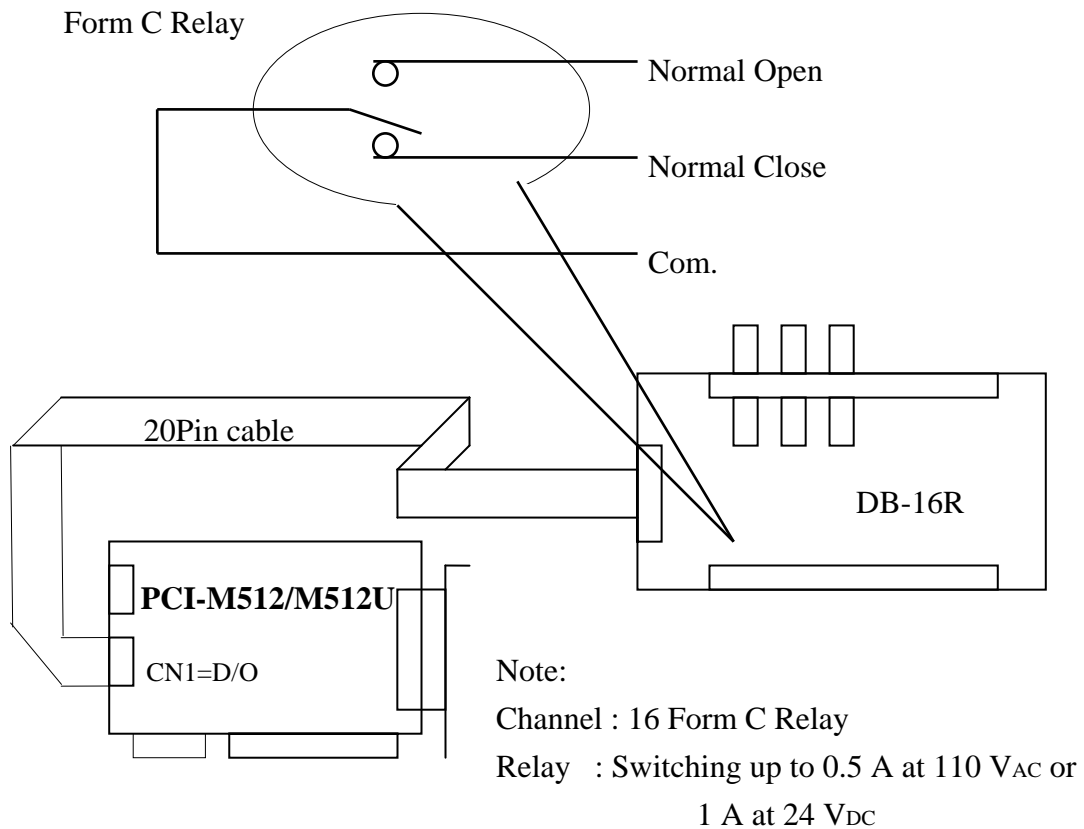
2.6.1 DB-16P Isolated Input Board

The DB-16P is a 16-channel isolated digital input daughter board. The optically isolated inputs of the DB-16P consist of a bi-directional optocoupler with a resistor for current sensing. You can use the DB-16P to sense DC signal from TTL levels up to 24 V or use the DB-16P to sense a wide range of AC signals. You can use this board to isolate the computer from large common-mode voltage, ground loops and transient voltage spike that often occur in industrial environments. **Note: the lowest nibble, bit_0 to bit_3, are used by PCI-M512, so only the highest 12-bits, bit_4 to bit_15, are available.**



2.6.2 DB-16R Relay Board

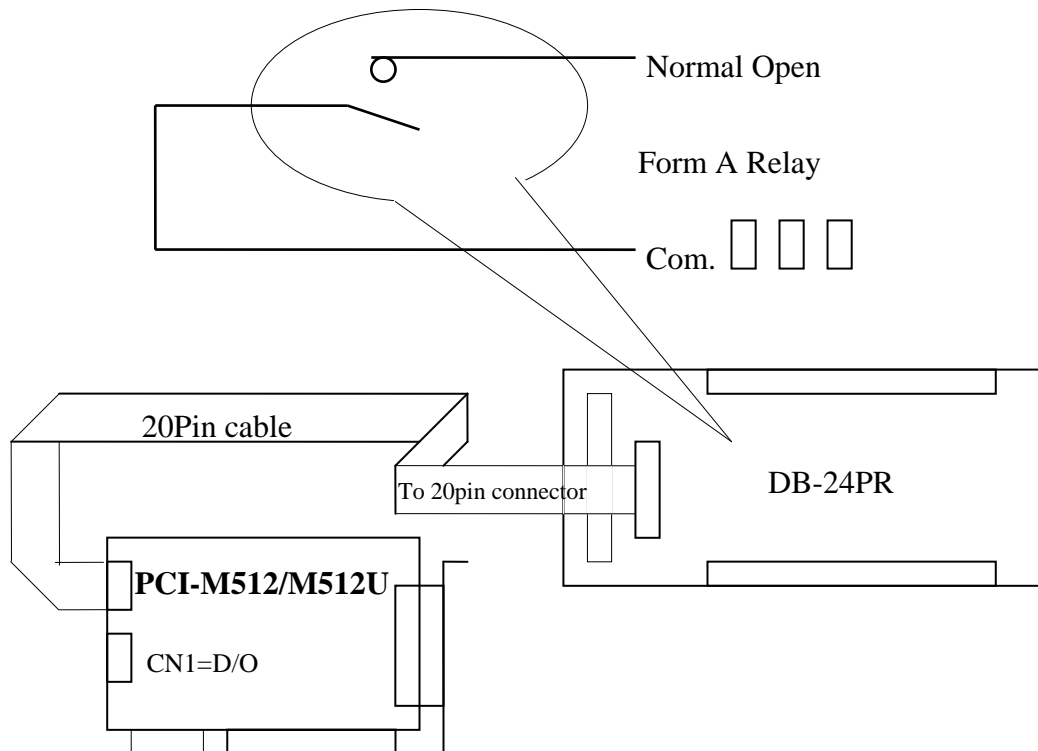
The DB-16R, 16-channel relay output board, consists of 16 Form C relays for efficient switching of loads by programmed control. It is a connector and functionally compatible with 785 series boards with industrial type terminal blocks. The relays are energized by applying a 5 volt signal to the appropriate relay channel on the 20-pin flat connector. There are 16 enunciator LEDs for each relay, they light when their associated relay is activated. To avoid overloading your PC's power supply, this board provides a screw terminal for external power supply.



2.6.3 DB-24PR, DB-24POR, DB-24C

DB-24PR	24*power relay, 5 A/250 V
DB-24POR	24*photo MOS relay, 0.1 A/350 V _{AC}
DB-24C	24*open collector, 100 mA per channel, 30 V max.

The DB-24PR, 24-channel power relay output board, consists of 8 Form C and 16 Form A electromechanical relays for efficient switching of loads by programmed control. The contact of each relay can control a 5 A load at 250 V_{AC}/30 V_{DC}. The relay is energized by applying a 5 volt signal to the appropriate relay channel on the 20-pin flat cable connector(just uses 16 relays) or 50-pin flat cable connector.(OPTO-22 compatible, for DIO-24 series). Twenty - four enunciator LEDs, one for each relay, light when their associated relay is activated. To avoid overloading your PC's power supply, this board needs a +12 V_{DC} or +24 V_{DC} external power supply.



Note:

50-Pin connector(OPTO-22 compatible), for DIO-24, DIO-48, DIO-144
 20-Pin connector for 16 channel digital output, A-82X, A-62X, DIO-64, ISO-DA16/DA8

Channel : 16 Form A Relay , 8 Form C Relay

Relay : switching up to 5 A at 110 V_{AC} / 5 A at 30 V_{DC}

2.7 Pin Assignment

CN2: pin assignment of digital input connector.

Pin	Name	Pin	Name
1	No Connection	2	No Connection
3	No Connection	4	No Connection
5	Digital input 4	6	Digital input 5
17	Digital input 6	8	Digital input 7
9	Digital input 8	10	Digital input 9
11	Digital input 10	12	Digital input 11
13	Digital input 12	14	Digital input 13
15	Digital input 14	16	Digital input 15
17	PCB ground	18	PCB ground
19	PCB +5 V	20	PCB +12 V

CN1: pin assignment of the digital output connector.

Pin	Name	Pin	Name
1	Digital output 0	2	Digital output 1
3	Digital output 2	4	Digital output 3
5	Digital output 4	6	Digital output 5
17	Digital output 6	8	Digital output 7
9	Digital output 8	10	Digital output 9
11	Digital output 10	12	Digital output 11
13	Digital output 12	14	Digital output 13
15	Digital output 14	16	Digital output 15
17	PCB ground	18	PCB ground
19	PCB +5 V	20	PCB +12 V

3. DLL Driver

The included software is a collection of subroutines for PCI-M512/M512U cards for Windows 95/98/NT/2000/XP applications. These subroutines are written with C language and perform a variety of digital I/O operations.

The subroutines in PCIM512.DLL are easy to understand as its name suggests for. It provides powerful, easy-to-use subroutines for developing your data acquisition application. Your program can easily call these DLL functions by VC++ VB, Delphi, and BORLAND C++ Builder. To speed-up your developing process, some demonstration source programs are provided.

Please refer to the following user manuals, you could find them in the companion CD:\NAPDOS\PCI\Manual.

- [PCI_ISA_PnP_Driver_Installation_in_Win9x_2K_XP.pdf](#)
Install the PnP (Plug and Play) driver for PCI card under Windows 95/98.
- [Software_Installation_Guide_in_Win32.pdf](#)
Install the software package under Windows 95/98/NT/XP.
- [Calling_DLL_functions_in_VB_VC_Delphi_BCB.pdf](#)
Call the DLL functions with VC++6, VB6, Delphi3 and Borland C++ Builder 3.
- [TroubleShooting_PCI_ISA_in_Win32_Resource_Conflict.pdf](#)
Check the resources I/O Port address, IRQ number and DMA number for add-on cards under Windows 95/98/NT.

The install shields will install kernel driver, DLL driver & application demo program to system. **All DLL driver & demo program are the same for all windows systems.** That is to say, the same DLL & demo program are same for 95/98/NT/2000/XP, but the kernel driver are different for different system as follows:

- for windows 95/98 → will copy **WINDRVR.VXD** to
C:\WIN95\SYSTEM\VMM32
- for windows NT/2000/XP → will copy **WINDRVR.SYS** to
C:\WINNT\SYSTEM32\DRIVERS

All DLL & demo program will not work if the kernel driver is not installed correctly. The install shields will copy the correct kernel driver to the correct position if you select the correct O.S.(95/98, NT, 2000, XP).

In this chapter, we use some keywords to indicate the attribute of Parameters.

Keyword	Set parameter by user before calling this function ?	Get the data/value from this parameter after calling this function ?
[Input]	Yes	No
[Output]	No	Yes
[Input, Output]	Yes	Yes

Note: All of the parameters need to be allocated spaces by the user.

The return codes of DLLs are defined as follows:

```
// return code
#define PCI_NoError 0
#define PCI_DriverOpenError 1
#define PCI_DriverNoOpen 2
#define PCI_GetDriverVersionError 3
#define PCI_InstallIrqError 4
#define PCI_ClearIntCountError 5
#define PCI_GetIntCountError 6
#define PCI_RegisterApcError 7
#define PCI_RemoveIrqError 8
#define PCI_FindBoardError 9
#define PCI_ExceedBoardNumber 10
#define PCI_ResetError 11
#define PCI_IrqMaskError 12
#define PCI_ActiveModeError 13
#define PCI_GetActiveFlagError 14
#define PCI_ActiveFlagEndOfQueue 15
#define PCI_BoardNoIsZero 16
#define PCI_BoardNoExceedFindBoards 17
```

The defined DLLs are given as follows:

Functions of test, Refer to Sec. 3.2

- float CALLBACK PCIM512_FloatSub(float fA, float fB);
- short CALLBACK PCIM512_ShortSub(short nA, short nB);
- int CALLBACK PCIM512_IntSub(int iA,int iB);
- DWORD CALLBACK PCIM512_GetDllVersion(void);

Functions of Driver Initialization, Refer to Sec. 3.3

- DWORD CALLBACK PCIM512_DriverInit(void);
- DWORD CALLBACK PCIM512_CloseBoard(DWORD dwBoardNo);
- DWORD CALLBACK PCIM512_DetectBoards(void);
- DWORD CALLBACK PCIM512_OpenBoard(DWORD dwBoardNo, DWORD dwIntEnable);
- DWORD CALLBACK PCIM512_ReadBoardStatus(DWORD dwBoardNo);
- DWORD CALLBACK PCIM512_CloseAll(void);

Functions of SRAM Read/Write, Refer to Sec. 3.4

- DWORD CALLBACK PCIM512_WriteSramByte(DWORD dwBoardNo, DWORD dwOffset, BYTE Data);
- DWORD CALLBACK PCIM512_WriteSramWord(DWORD dwBoardNo, DWORD dwOffset, WORD Data);
- DWORD CALLBACK PCIM512_WriteSramDword(DWORD dwBoardNo, DWORD dwOffset, DWORD Data);
- DWORD CALLBACK PCIM512_ReadSramByte(DWORD dwBoardNo, DWORD dwOffset, BYTE *Data);
- DWORD CALLBACK PCIM512_ReadSramWord(DWORD dwBoardNo, DWORD dwOffset, WORD *Data);
- DWORD CALLBACK PCIM512_ReadSramDword(DWORD dwBoardNo, DWORD dwOffset, DWORD *Data);

Functions of D/I/O Read/Write, Refer to Sec. 3.5

- DWORD CALLBACK PCIM512_WriteToDo(DWORD dwBoardNo, WORD Data);
- DWORD CALLBACK PCIM512_ReadFromDi(DWORD dwBoardNo, WORD *Data);

3.1 Find the Board Number

The plug&play BIOS will assign the proper base address to PCI-M512/M512U. If there is only one PCI-M512, users can identify this board as board_1. If there are two PCI-M512 boards in the system, it will be very difficult to identify which board is board_1. Our software driver can support 20 boards max. Therefore user can install 20 boards of PCI-M512 in one PC system.

The simplest way to find the board number is to use DioTest in Delphi4 demo program. This demo program will send a value to D/O and read back from D/I. The low 4 bits of D/I are battery status bits, they can be used as an indicator as follows:

Insert one piece of paper to BT1 of one PCI-M512

Install all PCI-M512 cards into this PC system

Power-on PC

You will find only one PCI-M512's LED1 & LED2 are ON

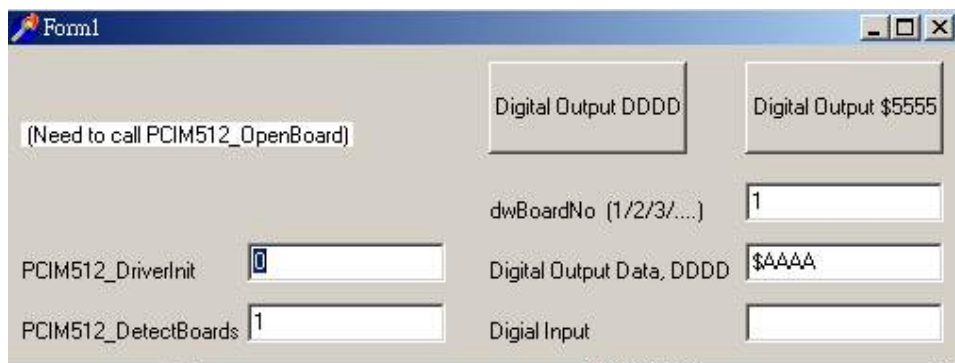
Run **DioTest of Delphi4 (Sec. 1.2.2)**

Key-in *board number* to 1

Click *Digital Output DD*

Check the value in *Digital Input*, if the LSB is 1, we find the target PCI-M512. Otherwise you can go to step 6 for next **board number**.

DioTest



The screenshot shows a Windows-style application window titled "Form1". The window contains several controls:

- A text label: "(Need to call PCIM512_OpenBoard)"
- Two buttons: "Digital Output DDDD" and "Digital Output \$5555"
- A text field: "dwBoardNo (1/2/3/...)" with the value "1" entered.
- A text field: "PCIM512_DriverInIt" with the value "0" entered.
- A text field: "Digital Output Data, DDDD" with the value "\$AAAA" entered.
- A text field: "PCIM512_DetectBoards" with the value "1" entered.
- A text field: "Digital Input" which is currently empty.

Note: only one PCI-M512/M512U the board number will be 1.

3.2 Functions of Test

Note: All DLL libraries given in Sec 3.3 can be used before the kernel driver is installed. Refer to Sec. 1.2.1 for more information.

3.2.1 PCIM512_FloatSub

- **Description:**
To perform the subtraction as $fA - fB$ in float data type. This function is provided for testing DLL linkage purpose.
- **Syntax:**
float PCIM512_FloatSub(float fA, float fB)
- **Parameter:**
fA : [Input] 4 bytes floating point value
fB : [Input] 4 bytes floating point value
- **Return:**
The value of $fA - fB$

3.2.2 PCIM512_ShortSub

- **Description:**
To perform the subtraction as $nA - nB$ in short data type. This function is provided for testing DLL linkage purpose.
- **Syntax:**
short PCIM512_ShortSub(short nA, short nB)
- **Parameter:**
nA : [Input] 2 bytes short data type value
nB : [Input] 2 bytes short data type value
- **Return:**
The value of $nA - nB$.

3.2.3 PCIM512_IntSub

- **Description:**
To perform the subtraction as $iA - iB$ in int data type. This function is provided for testing DLL linkage purpose.
- **Syntax:**
short PCIM512_IntSub(int iA, int iB)
- **Parameter:**
iA :[Input] 4 bytes int data type value
iB :[Input] 4 bytes int data type value
- **Return:**
The value of $iA - iB$

3.2.4 PCIM512_GetDllVersion

- **Description:**
To get the version number of PCIM512.DLL
- **Syntax:**
DWORD PCIM512_GetDllVersion(void)
- **Parameter:**
None
- **Return:**
Return the DLL's version number.
For example: 102(hex) for version 1.02

3.3 Functions of Driver Initialization

3.3.1 PCIM512_DriverInit

- **Description :**
This subroutine will allocate the resources for the Windriver. This function must be called first before calling the DLL functions given in Sec 3.3 ~ Sec. 3.5.
- **Syntax :**
DWORD PCIM512_DriverInit();
- **Parameter :**
None
- **Return:**
PCI_NoError : OK
PCI_DriverOpenError: Windriver kernel not find, refer to Sec. 1.2.1 for more information.

3.3.2 PCIM512_OpenBoard

- **Description :**
This subroutine will open the PCI-M512 kernel driver and allocate the resource from the device. This function must be called first before calling the DLL functions
- **Syntax :**
void PCIM512_OpenBoard(DWORD dwBoardNo, DWORD dwIntEnable);
- **Parameter :**
dwBoardNo [Input] PCI-M512 board number
dwIntEnable [Input] PCI-M512 board interrupt enable/disable(1/0)
- **Return:**
PCI_NoError :OK
PCI_BoardOpenError :Board open kernel driver error
PCI_BoardNoExceedFindBoards :Not find the Board.

3.3.3 PCIM512_DetectBoards

- **Description :**

This subroutine will detect all installed PCI-M512/M512U boards. **This function must be called first before calling the DLL functions given in Sec 3.4 & Sec. 3.5.**

- **Syntax :**

DWORD PCIM512_DetectBoards();

- **Parameter :**

None

- **Return:**

0: No PCI-M512 is installed in this PC

1: only one PCI-M512/M512U is installed in this PC(board no.=1)

2: there are 2 PCI-M512/M512U installed in this PC(board no.=1/2)

N: Number of PCI-M512 installed in this PC

- **Note:**

1. call **PCIM512_DriverInit()** before calling this function

2. call **PCIM512_OpenBoard()** before calling this function

3. call **PCIM512_DetectBoards()** to detect all PCI-M512 boards.

4. call **PCIM512_ReadBoardId(...)** to identify the detected PCI-M512 boards. Refer to Sec. 2.2 for more information.

3.3.4 PCIM512_ReadBoardId

- **Description :**

This subroutine will show the IDs of detected PCI-M512/M512U boards. It is designed to identify PCI-M512/M512U.

- **Syntax :**

DWORD PCIM512_ReadBoardId(dwBoardNo, *dwVendorId, *dwDeviceId, *dwSubVendorId, *dwSubdeviceId);

- **Parameter :**

dwBoardNo : [Input] PCI-M512/M512U board number
dwVendorID : [output] vendor ID of this board
dwDeviceID : [output] device ID of this board
dwSubVendorID : [output] sub-vendor ID of this board
dwSubDeviceID : [output] sub-device ID of this board

- **Return:**

0: this is a valid board no. → all return IDs are valid
others: this is not a valid board no. → all return IDs are invalid

- **Note:**

1. call **PCIM512_DriverInit()** before calling this function
2. call **PCIM512_OpenBoard()** before calling this function
3. call **PCIM512_DetectBoards()** to detect all PCI-M512 boards.
4. call **PCIM512_ReadBoardId(...)** to identify the detected PCI-M512/M512U boards. Refer to Sec. 2.2 for more information.

3.3.5 PCIM512_ReadBoardStatus

- **Description :**

This subroutine will detect the DLL open status of PCI-M512/M512U boards.

- **Syntax :**

DWORD PCIM512_ReadBoardStatus(DWORD dwBoardNo);

- **Parameter :**

dwBoardNo [Input] PCI-M512/M512U board number

- **Return:**

0: The DLL of the board dwBoardNo is not opened.

1: The DLL of the board dwBoardNo is opened.

- **Note:**

1. call **PCIM512_DriverInit()** before calling this function

3.3.6 PCIM512_CloseBoard

- **Description :**

This subroutine will close the PCI-M512/M512U kernel driver and release the resource from the device.

- **Syntax :**

DWORD PCIM512_CloseBoard(DWORD dwBoardNo);

- **Parameter :**

dwBoardNo [Input] PCI-M512/M512U board number

- **Return:**

PCI_NoError : OK.
PCI_BoardIsNotOpen: This board is not opened.
PCI_BoardNoExceedFindBoards Not fined the board

3.3.7 PCIM512_CloseAll

- **Description :**

This subroutine will close all of PCI-M512/M512U kernel driver and release the resource from the device.

- **Syntax :**

DWORD PCIM512_CloseAll();

- **Parameter :**

None

- **Return:**

PCI_NoError : OK.

3.4 Functions of Sram Read/Write

3.4.1 PCIM512_WriteSramByte

- **Description:**

Write one byte, 8-bit data to SRAM of PCI-M512/M512U.

- **Syntax:**

DWORD PCIM512_WriteSramByte(dwBoardNo, dwOffset, Data)

- **Parameter:**

dwBoardNo : [Input] board number, from 1 to N

dwOffset : [Input] offset address of SRAM, from 0 to 0x7fff

Data : [Input] one byte of data (8-bit)

- **Return:**

0: Write OK

PCI_DriverNoOpen: kernel driver no find

PCI_BoardNolsZero: dwBoardNo is 0, it must be in the range of 1 ~ N

PCI_BoardNoExceedFindBoards: dwBoardNo > N

- **Note:**

1. call **PCIM512_DetectBoards()** before calling this function

2. call **PCIM512_ReadBoardId(...)** to identify the detected PCI-M512/M512U boards. Refer to Sec. 2.2 for more information.

3. This function can be used for PCI-M512/M512U.

3.4.2 PCIM512_WriteSramWord

- **Description:**

Write one word, 16-bit, of data to SRAM of PCI-M512/M512U.

- **Syntax:**

DWORD PCIM512_WriteSramWord(dwBoardNo, dwOffset, Data)

- **Parameter:**

dwBoardNo : [Input] board number, from 1 to N

dwOffset : [Input] offset address of SRAM, from 0 to 0x7ffe

Data : [Input] one word of data (16-bit)

- **Return:**

0: Write OK

PCI_DriverNoOpen: kernel driver no find

PCI_BoardNolsZero: dwBoardNo is 0, it must be in the range of 1 ~ N

PCI_BoardNoExceedFindBoards: dwBoardNo > N

- **Note:**

1. call **PCIM512_DetectBoards()** before calling this function

2. call **PCIM512_ReadBoardId(...)** to identify the detected PCI-M512/M512U boards. Refer to Sec. 2.2 for more information.

3.4.3 PCIM512_WriteSramDword

- **Description:**

Write one dword, 32-bit data to SRAM of PCI-M512/M512U.

- **Syntax:**

DWORD PCIM512_WriteSramDword(dwBoardNo, dwOffset, Data)

- **Parameter:**

dwBoardNo : [Input] board number, from 1 to N

dwOffset : [Input] offset address of SRAM, from 0 to 0x7ffc

Data : [Input] one dword of data (32-bit)

- **Return:**

0: Write OK

PCI_DriverNoOpen: kernel driver no find

PCI_BoardNolsZero: dwBoardNo is 0, it must be in the range of 1 ~ N

PCI_BoardNoExceedFindBoards: dwBoardNo > N

- **Note:**

1. call **PCIM512_DetectBoards()** before calling this function

2. call **PCIM512_ReadBoardId(...)** to identify the detected PCI-M512/M512U boards. Refer to Sec. 2.2 for more information.

3.4.4 PCIM512_ReadSramByte

- **Description:**

Read one byte, 8-bit data from SRAM of PCI-M512/M512U.

- **Syntax:**

DWORD PCIM512_ReadSramByte(dwBoardNo, dwOffset, *Data)

- **Parameter:**

dwBoardNo : [Input] board number, from 1 to N

dwOffset : [Input] offset address of SRAM, from 0 to 0x7fff

Data : [output] one byte of data (8-bit)

- **Return:**

0: Write OK

PCI_DriverNoOpen: kernel driver no find

PCI_BoardNolsZero: dwBoardNo is 0, it must be in the range of 1 ~ N

PCI_BoardNoExceedFindBoards: dwBoardNo > N

- **Note:**

1. call **PCIM512_DetectBoard()** before calling this function

2. call **PCIM512_ReadBoardId(...)** to identify the detected PCI-M512/M512U boards. Refer to Sec. 2.2 for more information.

3.4.5 PCIM512_ReadSramWord

- **Description:**

Read one word, 16-bit data from SRAM of PCI-M512/M512U.

- **Syntax:**

DWORD PCIM512_ReadSramWord(dwBoardNo, dwOffset, *Data)

- **Parameter:**

dwBoardNo : [Input] board number, from 1 to N

dwOffset : [Input] offset address of SRAM, from 0 to 0x7ffe

Data : [output] one word of data (16-bit)

- **Return:**

0: Write OK

PCI_DriverNoOpen: kernel driver no find

PCI_BoardNolsZero: dwBoardNo is 0, it must be in the range of 1 ~ N

PCI_BoardNoExceedFindBoards: dwBoardNo > N

- **Note:**

1. call **PCIM512_DetectBoards()** before calling this function

2. call **PCIM512_ReadBoardId(...)** to identify the detected PCI-M512/M512U boards. Refer to Sec. 2.2 for more information.

3.4.6 PCIM512_ReadSramDword

- **Description:**

Read one dword, 32-bit data from SRAM of PCI-M512/M512U.

- **Syntax:**

DWORD PCIM512_ReadSramDword(dwBoardNo, dwOffset, *Data)

- **Parameter:**

dwBoardNo : [Input] board number, from 1 to N

dwOffset : [Input] offset address of SRAM, from 0 to 0x7ffc

Data : [output] one dword of data (32-bit)

- **Return:**

0: Write OK

PCI_DriverNoOpen: kernel driver no find

PCI_BoardNolsZero: dwBoardNo is 0, it must be in the range of 1 ~ N

PCI_BoardNoExceedFindBoards: dwBoardNo > N

- **Note:**

1. call **PCIM512_DetectBoards()** before calling this function

2. call **PCIM512_ReadBoardId(...)** to identify the detected PCI-M512/M512U boards. Refer to Sec. 2.2 for more information.

3.5 Functions of D/I/O Read/Write

3.5.1 PCIM512_WriteToDo

- **Description:**

Write one word, 16-bit, of data to D/O of PCI-M512/M512U.

- **Syntax:**

DWORD PCIM512_WriteToDo(dwBoardNo, Data)

- **Parameter:**

dwBoardNo : [Input] board number, from 1 to N

Data : [Input] one word of data (16-bit)

- **Return:**

0: Write OK

PCI_DriverNoOpen: kernel driver no find

PCI_BoardNolsZero: dwBoardNo is 0, it must be in the range of 1 ~ N

PCI_BoardNoExceedFindBoards: dwBoardNo > N

- **Note:**

1. call **PCIM512_DetectBoards()** before calling this function

2. call **PCIM512_ReadBoardId(...)** to identify the detected PCI-M512 /M512U boards. Refer to Sec. 2.2 for more information.

3.5.2 PCIM512_ReadFromDi

- **Description:**

Read one word, 16-bit, of data from D/I & battery status bits of PCI-M512/M512U.

- **Syntax:**

DWORD PCIM512_ReadFromDi(dwBoardNo, *Data)

- **Parameter:**

dwBoardNo : [Input] board number, from 1 to N

Data: [output] one word of data (16-bit), Bit0 ~ Bit3 are battery status bits and Bit4 ~ Bit15 are external D/I bits as follows:

Bit0=1 → BT1 is low battery

Bit1=1 → BT1 is bad battery

Bit2=1 → BT2 is low battery

Bit3=1 → BT3 is bad battery

(refer to Sec. S.4 for more information)

- **Return:**

0: Write OK

PCI_DriverNoOpen: kernel driver no find

PCI_BoardNolsZero: dwBoardNo is 0, it must be in the range of 1 ~ N

PCI_BoardNoExceedFindBoards: dwBoardNo > N

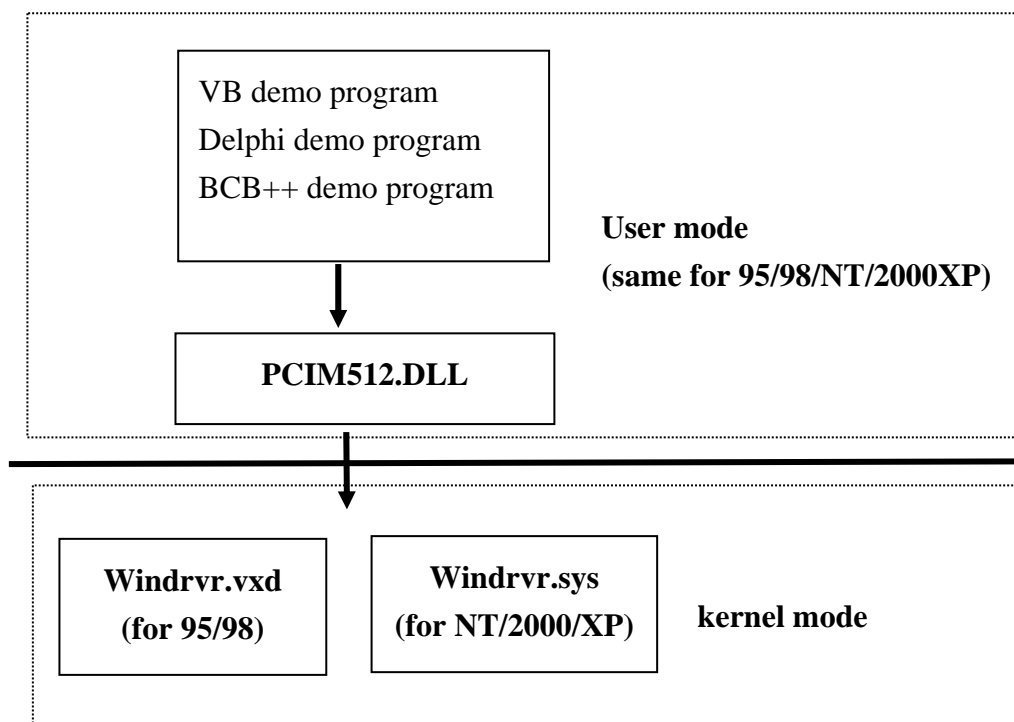
- **Note:**

1. call **PCIM512_DetectBoards()** before calling this function

2. call **PCIM512_ReadBoardId(...)** to identify the detected PCI-M512/M512U boards. Refer to Sec. 2.2 for more information.

4. Demo Program

There are many demo program, written in VC++, VB, Delphi, and BCB++, given in the companion CD. These demo programs will call the DLL, PCIM512.DLL, to access the hardware of PCI-M512/M512U. The PCIM512.DLL will call the kernel driver, Windrvr.vxd or Windrvr.sys as follows:



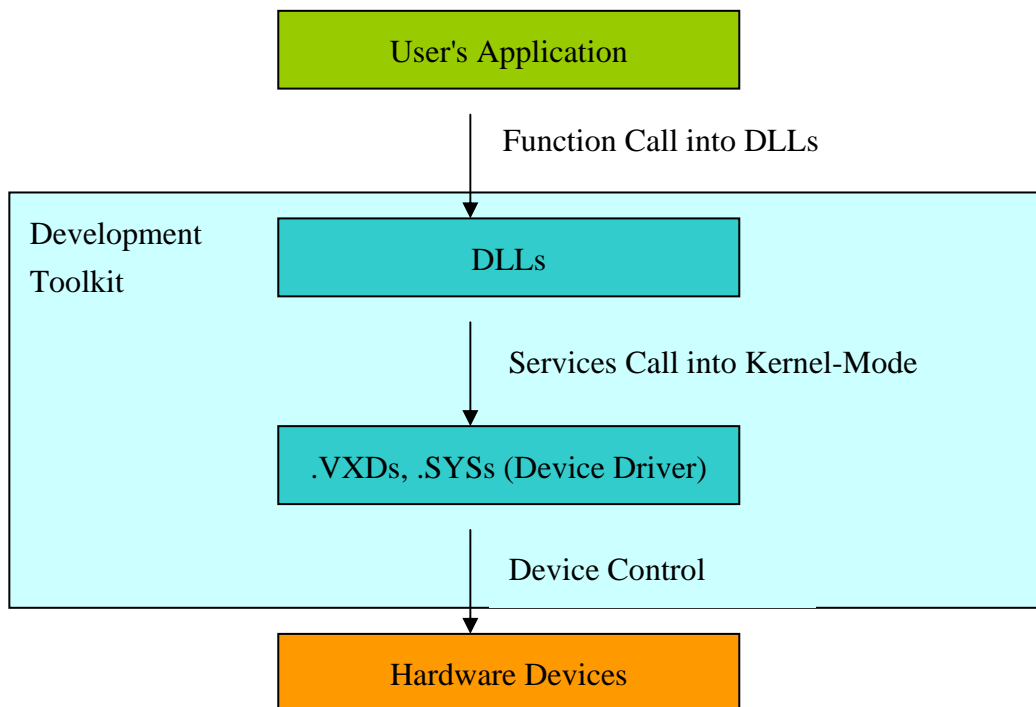
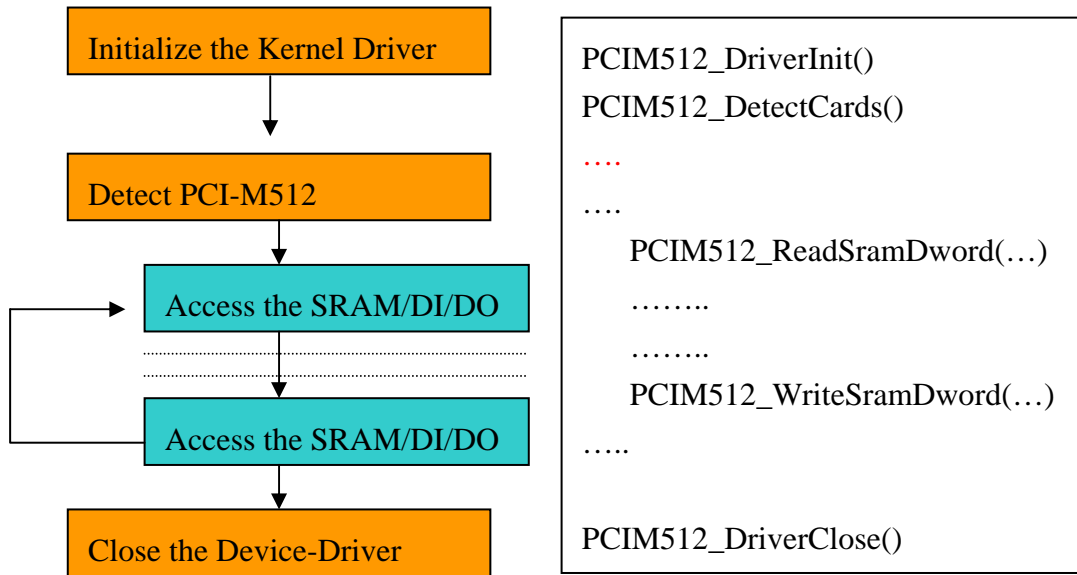
The install shields will install kernel driver, DLL driver & application demo programs to system. **All demo program & DLL are same for 95/98/NT/2000/XP**, but the kernel driver is different for different system as follows:

- for windows 95/98 → will copy **WINDRVR.VXD** to
C:\WIN95\SYSTEM\MM32
- for windows NT/2000/XP → will copy **WINDRVR.SYS** to
C:\WINNT\SYSTEM32\DRIVERS

All DLL & demo program will not work if the kernel driver is not installed correctly. The install shields will copy the correct kernel driver to the correct position if you select the correct O.S.(95/98/ME, NT, 2000, XP).

Refer to **Calling DLL functions in VB_VC_Delphi_BCB.pdf** in the companion CD:\NAPDOS\PCI\Manual for more information about how to call the DLL functions with VC++5, VB5, Delphi3 and Borland C++ Builder 3.

4.1 Program Architecture



4.2 Problems Report

Technical support is available at no charge as described below. The best way to report problems is to send electronic mail to

Service@icpdas.com

on the Internet.

When reporting problems, please include the following information:

- 1) Is the problem reproducible? If so, how?
- 2) What kind and version of **platform** that you using? For example, Windows 3.1, Windows 95, or Windows NT 4.0, etc.
- 3) What kinds of our **products** are you using? Please see the product's manual.
- 4) If a dialog box with an **error message** was displayed, please include the full text of the dialog box, including the text in the title bar.
- 5) If the problem involves **other programs** or **hardware devices**, what devices or version of the failing programs are you using?
- 6) **Other comments** relative to this problem or **any suggestions** will be welcomed.

After we had received your comments, we will take about two business days to test the problems that you related. Then will reply as soon as possible to you. Please check that we had received your comments. And please keep in contact with us.

ICP DAS

E-mail: Service@icpdas.com

Web Site: <http://www.icpdas.com>