# PCM-CAN 200/200P
# PISO-CAN 200E - D/T
# PISO-CAN 200/400 - D/T
# PISO-CAN 100U/200U/400U - D/T
# DeviceNet Master Library V3.0

User's Manual

## Revision

| Version | DeviceNet X-Server Version | Date | CAN driver version | Author | Description |
|---|---|---|---|---|---|
| 3.1 | 1.0 | 2012 06/12 | 2.0 | Johney | Fix some error descriptions and add C# demos. |
| 3.0 | 1.0 | 2009 05/11 | 2.0 | Johney | Improve the efficiency.<br>New API functions.<br>Add search functions.<br>Support PCI boards<br>1. PISO-CAN 200/400<br>2. PISO-CAN 200/400U<br>3. PISO-CAN 200E<br>4. PCM-CAN 200 |
| 2.0 | None | 2008 12/15 | 1.0 | Johney | Fix the bugs of the read/write I/O functions. |
| 1.0 | None | 2006 02/15 | 1.0 | Johney | This manual is for the PISO-CAN 200/400 D/T board. |

# Contents

# 1.General Information

## *1.1 DeviceNet Introduction*

The CAN (Controller Area Network) is a serial communication protocol, which efficiently supports distributed real-time control with a very high level of security. It is an especially suited for networking "intelligent" devices as well as sensors and actuators within a system or sub-system. In CAN networks, there is no addressing of subscribers or stations in the conventional sense, but instead, prioritized messages are transmitted. DeviceNet is one kind of the network protocols based on the CAN bus and mainly used for machine control network, such as textile machinery, printing machines, injection molding machinery, or packaging machines, etc. DeviceNet is a low level network that provides connections between simple industrial devices (sensors, actuators) and higher-level devices (controllers), as shown in Figure 1.1.

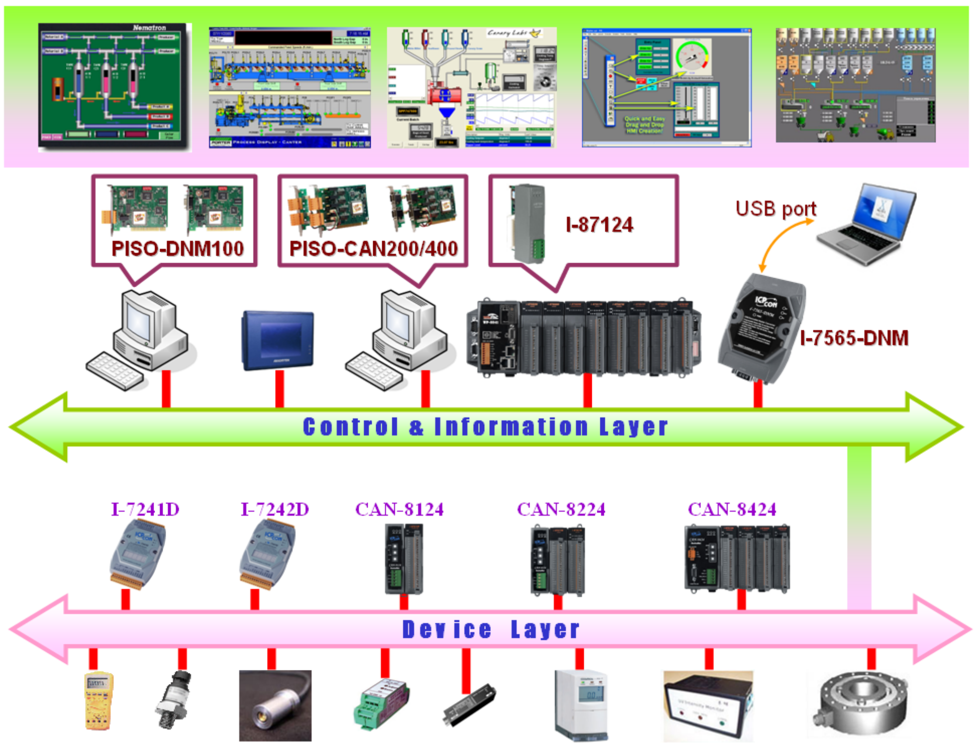


Figure 1.1-1 Example of the DeviceNet network

DeviceNet is a cost effective solution to one kind application of control area network. It reduces the connection wires between devices and provides rapid troubleshooting function. The transfer rate can be up to 500Kbps within 100 meters. The transfer distance can be up to 500 meters in 125Kbps (See Table 1.1). It allows direct peer to peer data exchange between nodes in an organized and, if necessary, deterministic manner. Master/Slave connection model can be supported in the same network. Therefore, DeviceNet is able to facilitate all application communications based on a redefine a connection scheme. However, DeviceNet connection object strands as the communication path between multiple endpoints, which are application objects that is needed to share data.

| Baud rate (bit/s) | Max. Bus length (m) |
|---|---|
| 500 K | 100 |
| 250 K | 250 |
| 125 K | 500 |

Table 1.1 The Baud rate and the Bus length

## 1.2 DeviceNet Applications

DeviceNet is the standardized network application layer optimized for factory automation. It is mainly used in low- and mid-volume automation systems. Some users have also implemented DeviceNet for machine control systems. The main DeviceNet application fields include the following application area (For more information, please refer to www.odva.org):

- Production cell builds and tests CPUs
- Dinnerware production
- Equipment for food packing
- Textile machines
- Fiberglass twist machine
- Trawler automation system
- Sponge production plant
- LCD manufacturing plant
- Isolation wall manufacturing
- Rolling steel door production
- Overhead storage bin production
- Bottling line

## *1.3 DeviceNet Master Library Characteristics*

ICP DAS DeviceNet Master Library (DLL functions) provides users to establish DeviceNet network rapidly by Master/Slave connection model. The DeviceNet master library is for PCI CAN interface card, which are PISO-CAN200/400, PISO-CAN200/400U, PCM-CAN200 and PISO-CAN200E. Using the library, users don't need to take care of the detail of the DeviceNet protocol. The library will implement the DeviceNet protocol automatically. It can reduce the complexity of user's DeviceNet master software. The library mainly supports the Predefined Master-Slave Connection Set functions to allow users to merge PISO-CAN200/400, PISO-CAN200/400U, PCM-CAN200 or PISO-CAN200E into DeviceNet network. It can help users to establish the connection with DeviceNet slave devices easily. The general application architecture is demonstrated as Figure 1.3-1.

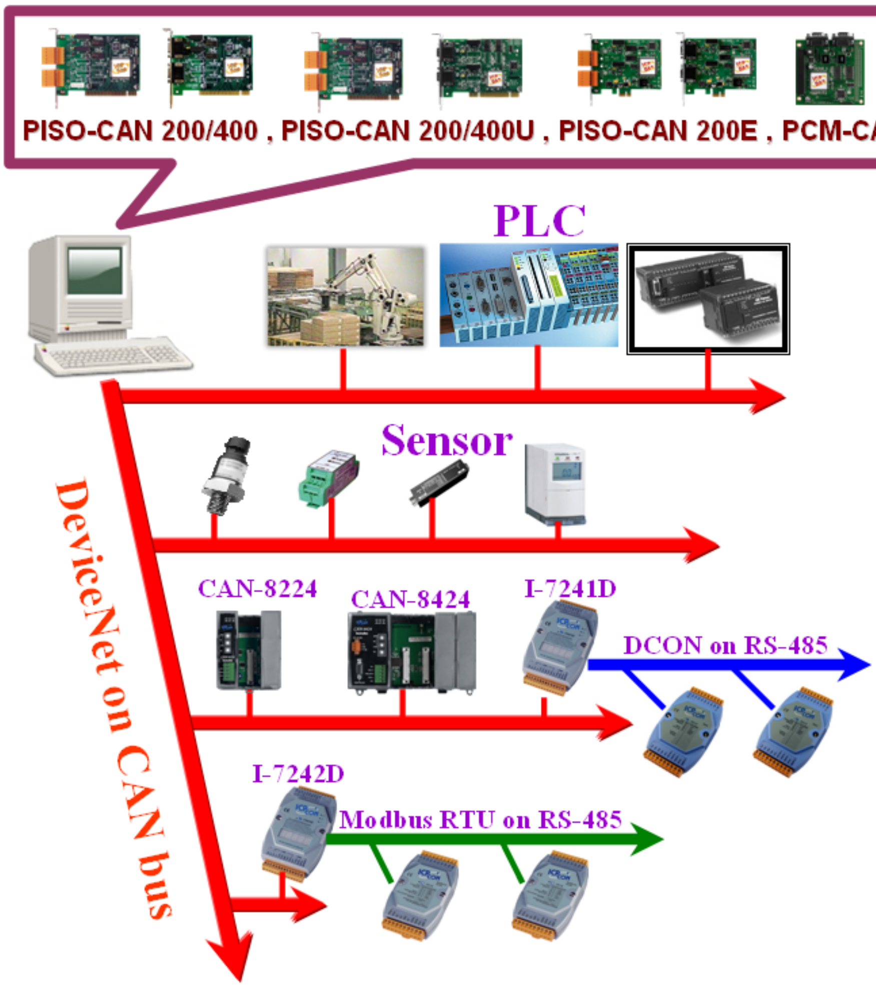


Figure 1.3-1 Application architecture

The library only provides the DeviceNet Master mechanism to communicate with slave devices by the Predefined Master/Slave Connection Set, which can be clarify as two forms: One is the Explicit Message and others are I/O Messages. Note that before communicating I/O data with DeviceNet slave devices, the Master device must connect to slave devices by explicit message connection to define the connection object. Here, we only provide one explicit message connection and four I/O connections as depicted in Figure 1.3-2.

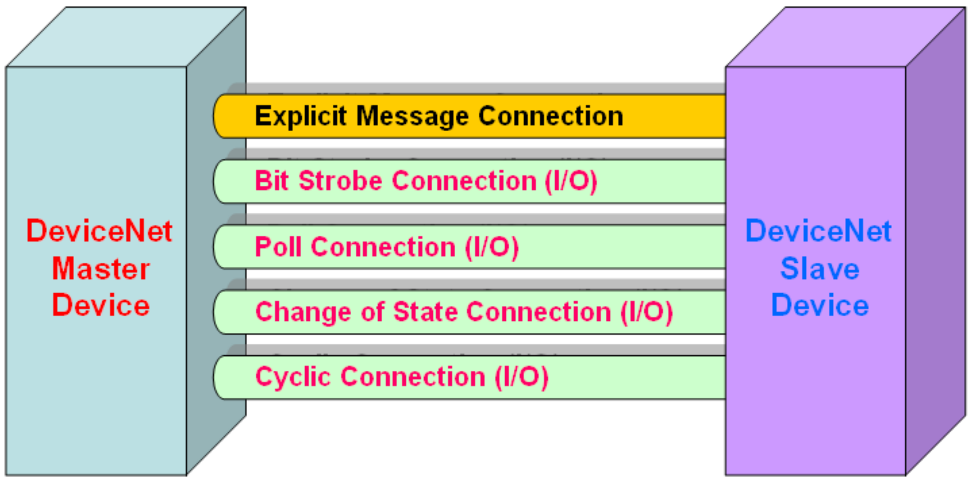


Figure 1.3-2 DeviceNet Messaging

The DeviceNet communication protocol is based on the concept of connections method. Master should create connections with slave devices based on the command of exchanging information and I/O data. To establish master control mechanism, there are only four main steps to be followed. Figure 1.3-3 demonstrates the basic process for the DeviceNet master communication. The every step function is described in below:

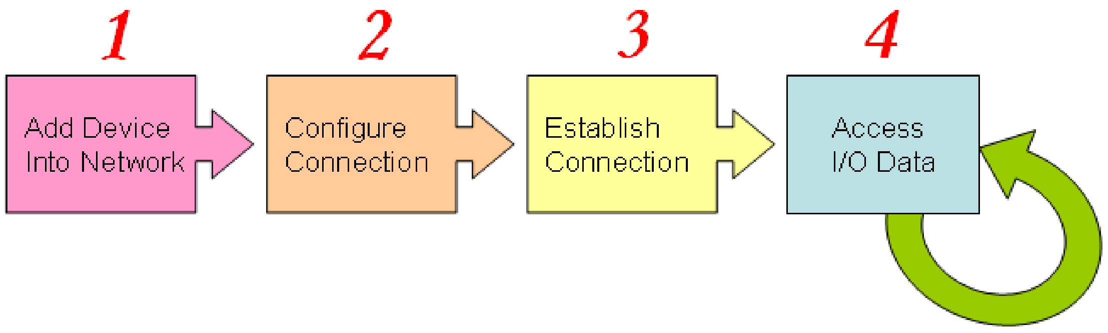


Figure 1.3-3 Four steps to establish connection

1. **Add device into network**
   You should provide the slave device's MAC ID to add into network.
2. **Configure connection**
   You can check the slave device's I/O connection type and the I/O data length. When configuring the I/O connection, you should provide these parameters.
3. **Establish connection**
   After configuring connections, users can start communicating with slave devices.
4. **Access I/O data**
   After communicating with slave devices, you can access the I/O data with corresponding read/write function.

After establishing the explicit connection, the connection path is then used to exchange the general information from one node to the others. And then users can create the I/O connections in the next step. Once I/O connections have been created, I/O data may be exchanged among devices in the DeviceNet network according to master device demand. Therefore, the master device can access I/O data of the slave devices by one of the four I/O connection methods. The library is not only easy to use but also providing a lot of the DeviceNet Master functions to retrieve and deliver the slave's I/O data. For more information, please refer to function description and demo programs in section 4.

## 1.4 DeviceNet Master X-Server Characteristics

The DeviceNet Master X-Server is a high-performance DeviceNet master engine. The DeviceNet master engine implements DeviceNet protocol automatically when the engine is active. The engine always listens to the bus and receives the message at the same time. It works as shown in Figure 1.4-1.

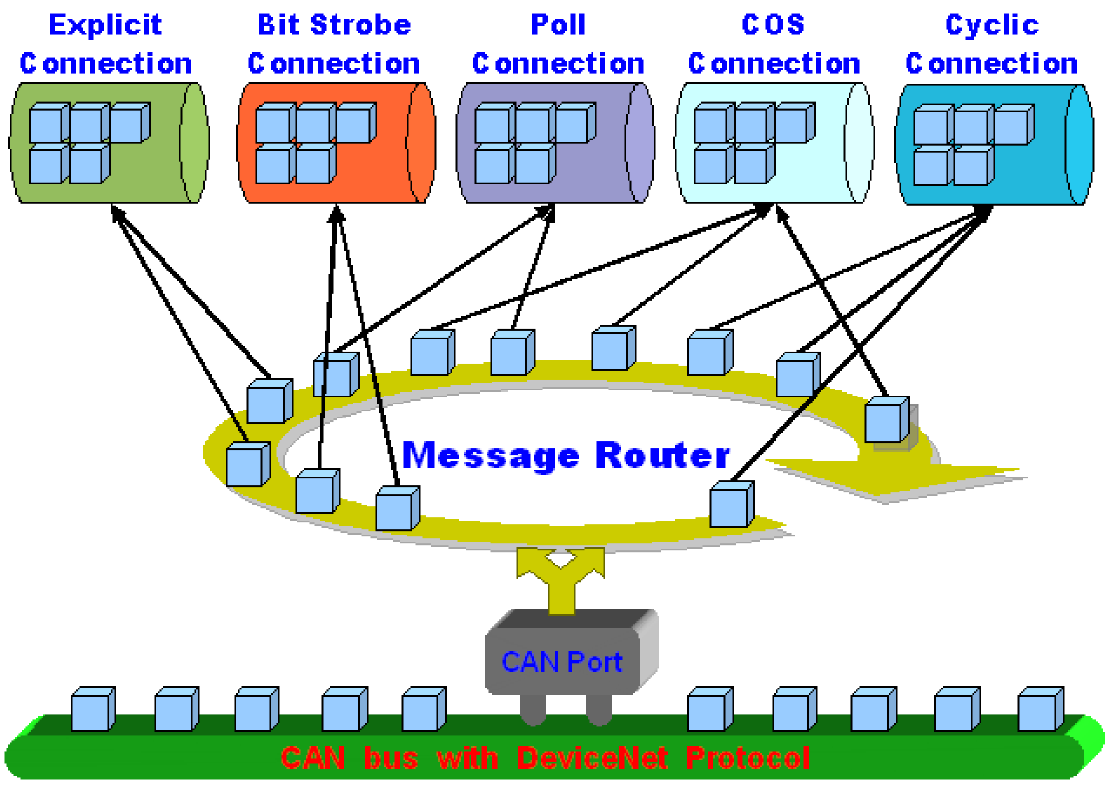


Figure 1.4-1 Message Router

The DeviceNet Master X-Server has a “ScanList” to store the remote slave devices information. After power off, the information still exists in the configuration file. When the users turn on the PC next time, the “ScanList” will be loaded from the configuration file. The users can easily use the DLL functions to configure it, including adding devices or removing devices. It works as shown in Figure 1.4-2. There is more information about the library functions in chapter 3.

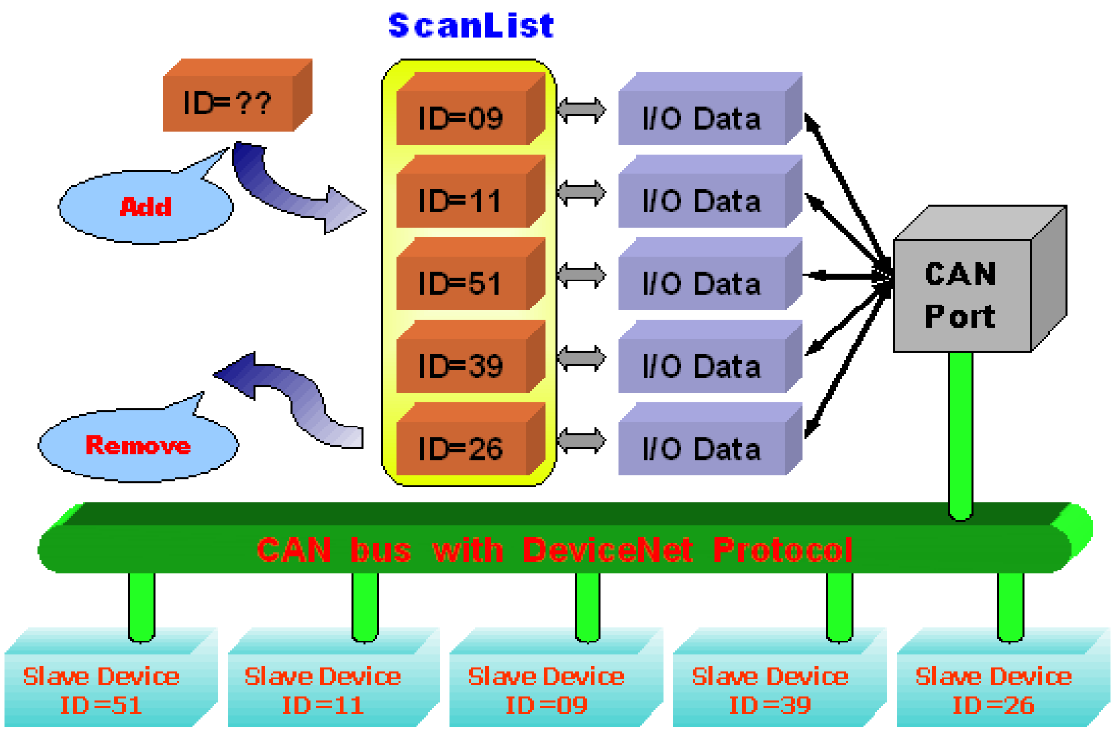


Figure 1.4-2 ScanList data structure

## *1.5 DeviceNet Library Features*

- Driver supported for Windows 98/ME/NT/2000/XP
- Programmable Master MAC ID.
- Programmable transfer-rate 125K, 250K, 500K.
- Each port support maximum nodes up to 64
- Support Group 2 Only Client functions
- Support UCMM functions
- Predefined Master-Slave Connection Set
- The maximum Fragment number is (Input/Output) up to 64
- Support I/O Operation Mode: Poll, Bit-Strobe and Change Of State/Cyclic
- Support Auto-Scan slave device function.
- Support on-line adding and removing devices.
- Support Auto-Reconnect when the connection is broken.

# 2. Driver Installation and Software Application

The DeviceNet DLL library is the collection of function calls for the PISO-CAN 200/400, PISO-CAN200/400U, PCM-CAN200 or PISO-CAN200E cards used in Windows 98/Me/NT/2000/XP systems. The application structure is presented in the following figure. The user's DeviceNet application programs can be developed by the following designated tools: VB, Delphi and Borland C++ Builder…etc. In these tools, the application program can call the PISOCANDNM.DLL driver to implement DeviceNet network application, as shown in the following Figure 2.0-1.

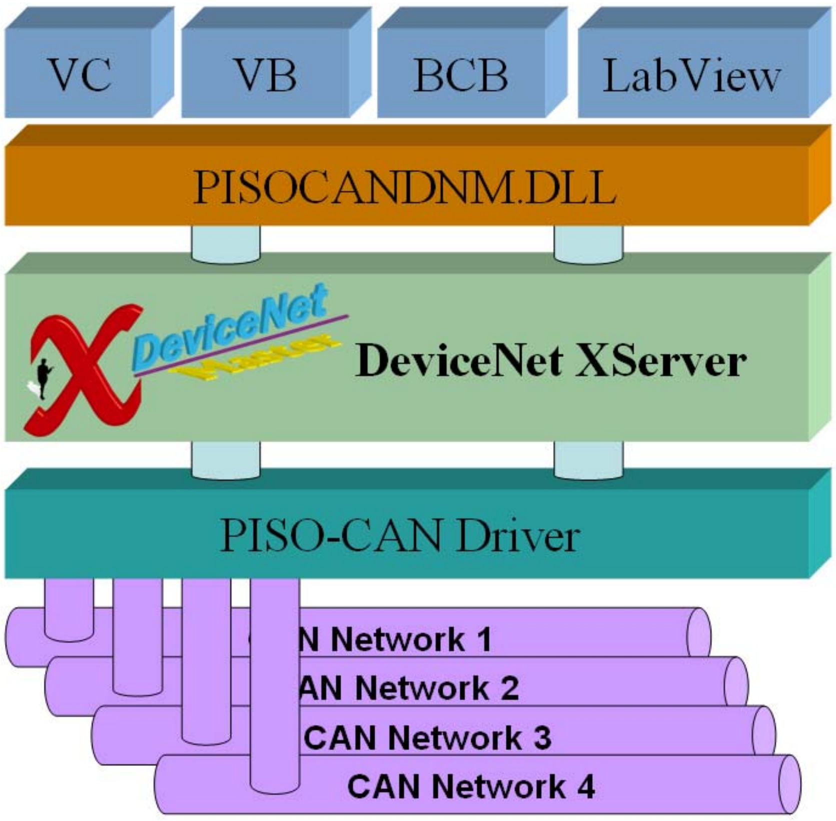


Figure 2.0-1 DeviceNet Driver skeleton

In the following sub-section, we show some flow diagrams to describe how to apply the DeviceNet protocol (PISOCANDNM.DLL) to build a master device. Section 2.3 ~ 2.7 show the flow diagram for users to understand easily. Note that users need to follow the operation principle of the DeviceNet protocol correctly and easily to communicate with the remote nodes by these connection methods.

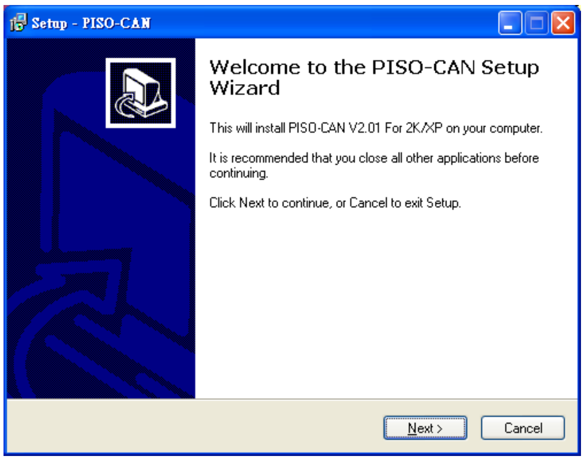
## *2.1 Driver Installation of the PISO-CAN 200/400 Card*

The software Installation for DeviceNet application has two main steps. First, the users should install the PISO-CAN 200/400, PISO-CAN200/400U, PCM-CAN200 or PISO-CAN200E card driver. Next, the users should install the DeviceNet Library. After finishing the installation process, the user can refer to demo program and follow its hardware application structure to testing the basic functions of master/slave connection. For the advanced application, users can refer to the basic demo program to develop the DeviceNet master applications.

The driver of PISO-CAN can be used in 98/Me/NT/2K/XP Windows environments. Users can find the driver in the path of "\CAN\PCI\PISO-CAN200_400" in the Fieldbus_CD. Execute the PISO-CAN.exe file to start install the driver. For these Windows operation systems, the recommended installation procedure is given as follows:

Step 1: Insert the product CD into the CD-ROM and find the path \CAN\PCI\PISO-CAN200_400\Win2K_XP\Setup (ex: the OS is Windows 2000 or XP).Then executes the PISO-CAN.exe to install the PISO-CAN card driver.

Step 2: Click the "Next" button to start the installation.



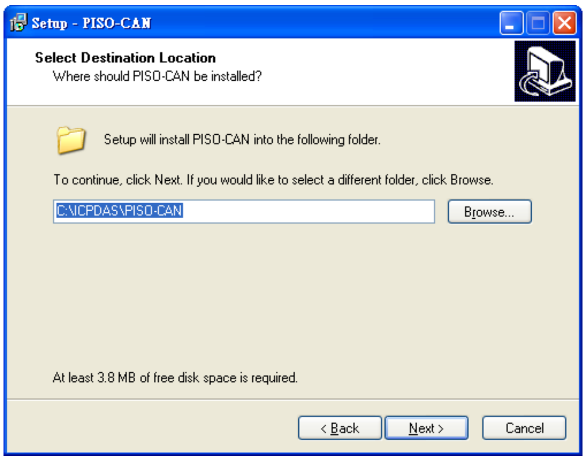
Welcome to the PISO-CAN Setup Wizard

This will install PISO-CAN V2.01 For 2K/XP on your computer.

It is recommended that you close all other applications before continuing.

Click Next to continue, or Cancel to exit Setup.

Step 3: Select the folder where the PISO-CAN setup would be installed and click "Next" button to continue.


Setup - PISO-CAN

**Select Destination Location**

Where should PISO-CAN be installed?

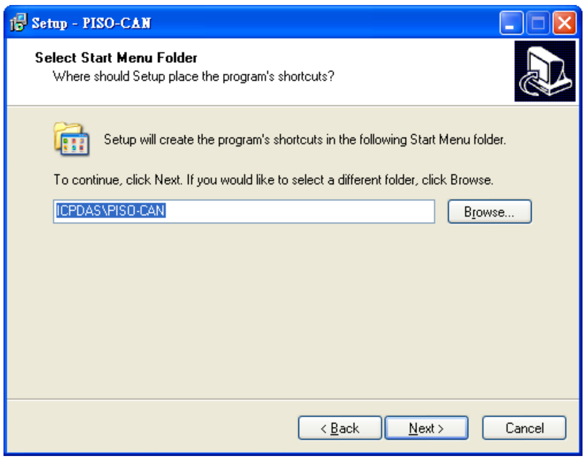
Setup will install PISO-CAN into the following folder.

To continue, click Next. If you would like to select a different folder, click Browse.

C:\ICPDAS\PISO-CAN

At least 3.8 MB of free disk space is required.

Step 4: Click the button "Next" to continue.


Setup - PISO-CAN

**Select Start Menu Folder**

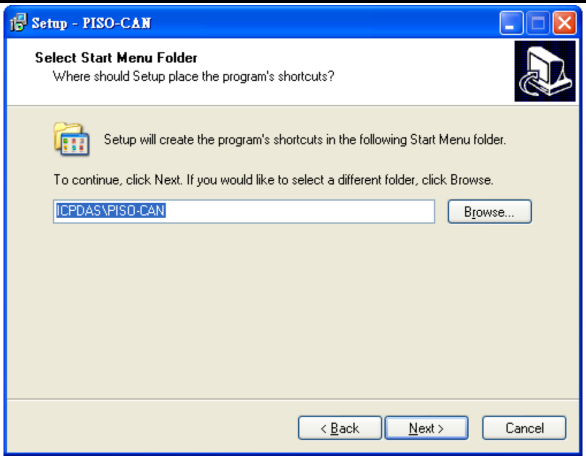
Where should Setup place the program's shortcuts?

Setup will create the program's shortcuts in the following Start Menu folder.

To continue, click Next. If you would like to select a different folder, click Browse.

ICPDAS\PISO-CAN

Step 5: Click the button "Next" to continue.



Select Start Menu Folder

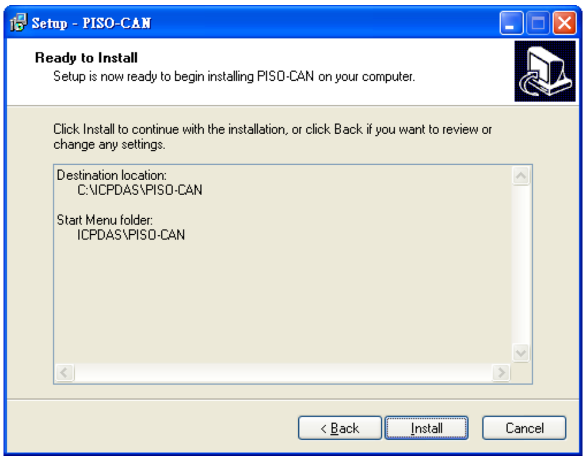
Where should Setup place the program's shortcuts?

Setup will create the program's shortcuts in the following Start Menu folder.

To continue, click Next. If you would like to select a different folder, click Browse.

ICPDAS\PISO-CAN

Step 6: Click the button "Install" to continue.



Ready to Install

Setup is now ready to begin installing PISO-CAN on your computer.

Click Install to continue with the installation, or click Back if you want to review or change any settings.

Destination location:
    C:\ICPDAS\PISO-CAN

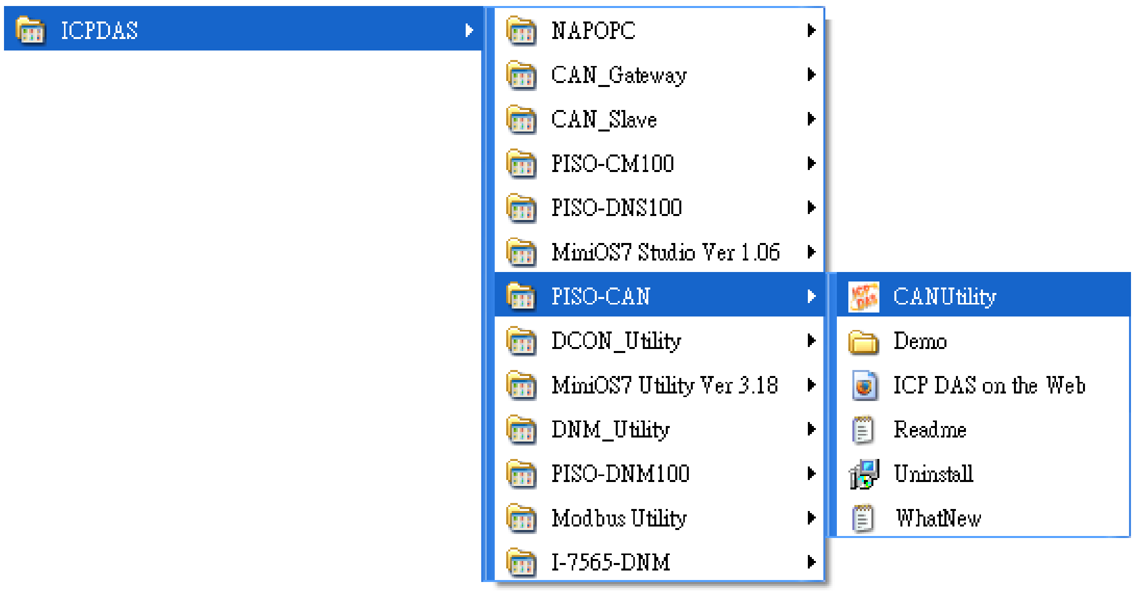
Start Menu folder:
    ICPDAS\PISO-CAN

Then the setup file will start the driver installation process and copy the related material to the indicated directory and then register the driver on your computer. The driver target directory is as below for the different systems.

Windows NT/2000 – **WINNT\SYSTEM32\DRIVERS**

Windows 98/Me/XP – **WINDOWS\SYSTEM32\DRIVERS**

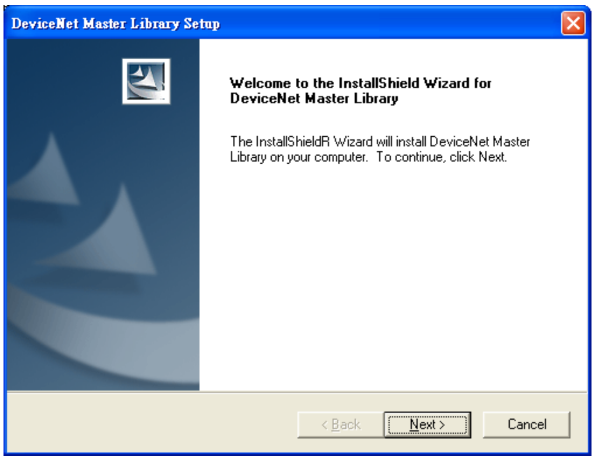
PISO-CAN Driver – **C:\ICPDAS\PISO-CAN**


ICPDAS
NAPOPC
CAN_Gateway
CAN_Slave
PISO-CM100
PISO-DNS100
MiniOS7 Studio Ver 1.06
PISO-CAN
DCON_Utility
MiniOS7 Utility Ver 3.18
DNM_Utility
PISO-DNM100
Modbus Utility
I-7565-DNM
CANUtility
Demo
ICP DAS on the Web
Readme
Uninstall
WhatNew

## *2.2 Install the DeviceNet Master Library*

Step 1: Insert the product CD into the CD-ROM and find the path \DeviceNet\Master\PISO-CAN200_400\Setup\.Then executes the DeviceNet Master Library v3.0.exe to install the DeviceNet master library.

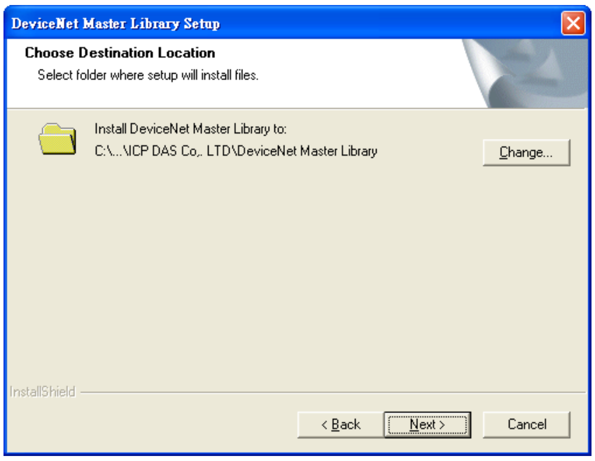
Step 2: Click the “Next” button to continue.



Welcome to the InstallShield Wizard for DeviceNet Master Library

The InstallShieldR Wizard will install DeviceNet Master Library on your computer. To continue, click Next.

Step 3: Click the “Next” button to continue.



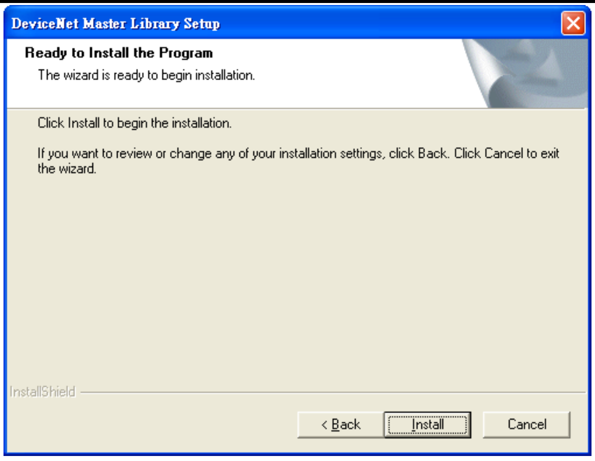
Choose Destination Location

Select folder where setup will install files.

Install DeviceNet Master Library to:

C:\...\ICP DAS Co,. LTD\DeviceNet Master Library

Step 4: Click “Install” to install the library.



DeviceNet Master Library Setup

Ready to Install the Program

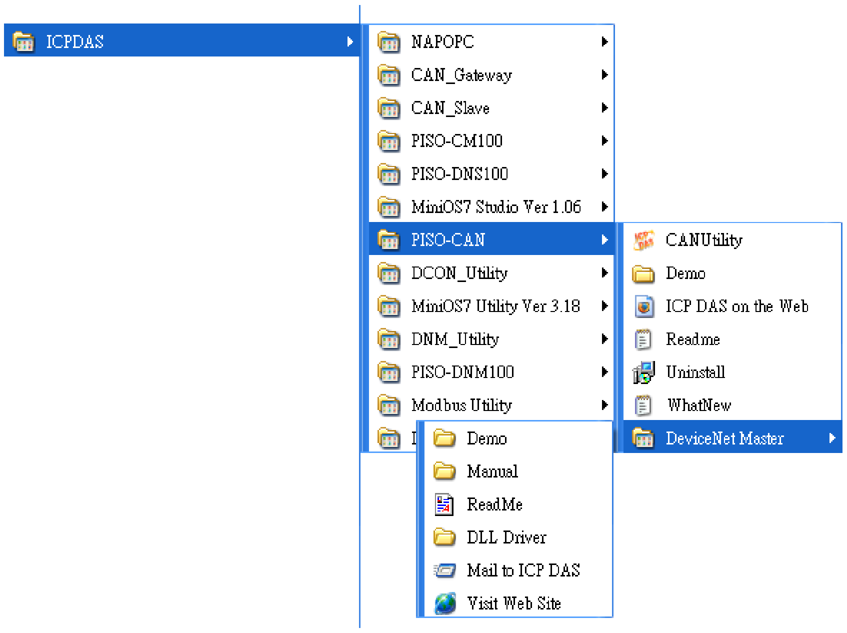
The wizard is ready to begin installation.

Click Install to begin the installation.

If you want to review or change any of your installation settings, click Back. Click Cancel to exit the wizard.

Then the setup file will start the driver installation process and copy the related material to the indicated directory and then register the driver on your computer. The driver target directory is default as below.

DeviceNet Master Library – **C:\ICPDAS\PISO-CAN\DeviceNet\**



Note：DeviceNet Master Utility is a useful tool for users to configure and test the DeviceNet slave devices. You can find the software in the path of /devicenet/Master/DNM_Utility/ in the Fieldbus_CD or from ftp://ftp.icpdas.com/pub/cd/fieldbus_cd/devicenet/master/DNM_Utility.
After installing the software, the utility is installed in the path below.
C:\ICPDAS\DNM_Utility\DNM_Utility.exe
Please refer to the manual of utility to know the detail.

## 2.3 Flow Diagram for Searching Devices

Before developing the DeviceNet applications, users should diagnose the connection between the slave devices. First, the users can search the slave devices in the network by using the searching functions. If the connection between the master with other slave devices is fine, the uses can find the information of the corresponding slave devices. When the users have no idea to communicate with the slave devices, users can follow these steps shown in figure 2.3-1. The following functions can help users to get the DeviceNet information of the slave devices. The users can find out the problem of the slave devices by using these functions. The detail information about those functions is in the next chapter.

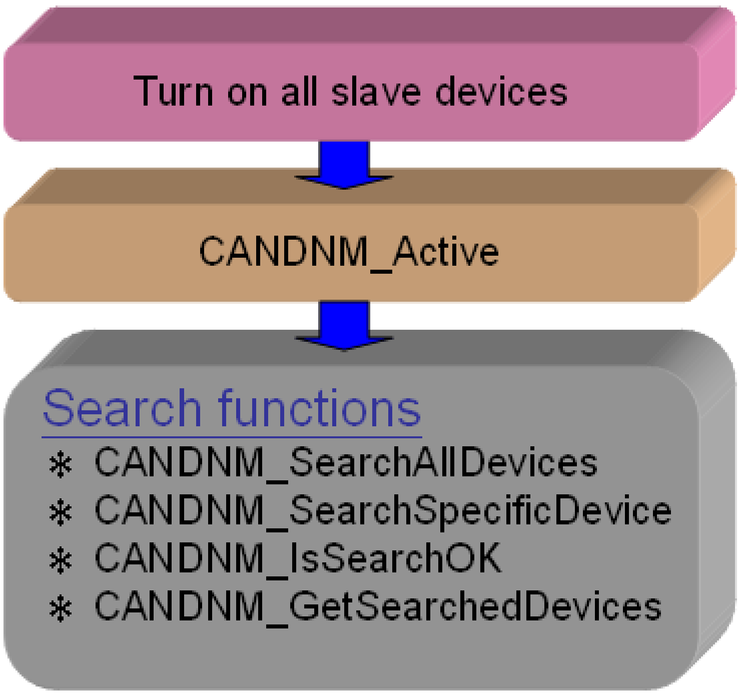

Turn on all slave devices
CANDNM_Active
Search functions
CANDNM_SearchAllDevices
CANDNM_SearchSpecificDevice
CANDNM_IsSearchOK
CANDNM_GetSearchedDevices

Figure 2.3-1 Searching Diagram

## 2.4 Flow Diagram for Slave Configuration

After getting the DeviceNet I/O information of the slave devices, users should save the parameters into the configuration file of the DeviceNet master library. The files will store the configuration data. The DeviceNet master library will load the previous configuration from the configuration file in the next boot-up. When the devices in the DeviceNet network are changed, the users must set the configuration data to fit the application. The configuration diagram is shown in Figure 2.4-1.  There is more information about those functions in the next chapter.

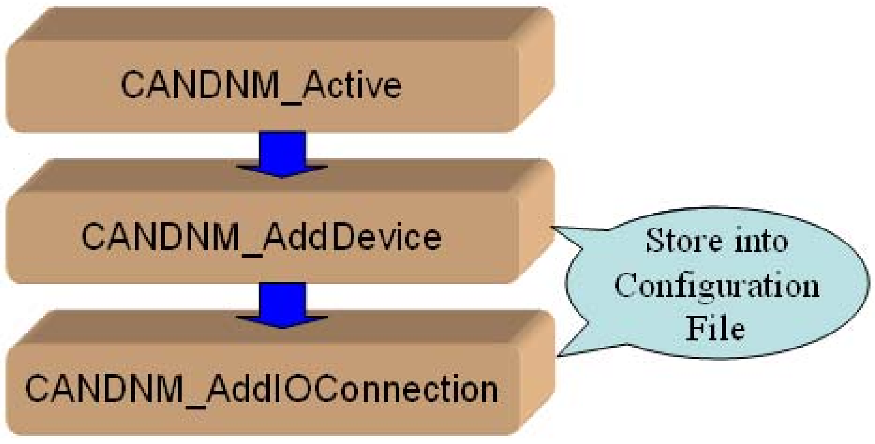


Figure 2.4-1 Slave Configuration Diagram

## *2.5 Flow Diagram for On-line Adding/Removing Device*

The DeviceNet master library provides the on-line adding/removing slave device functions. The users need not to break the communication between original slave devices when adding or removing the slave devices. The users can follow the steps to achieve this function. The steps are shown in Figure 2.5-1 and Figure 2.5-2.

### 1. <u>On-line Adding Devices :</u>

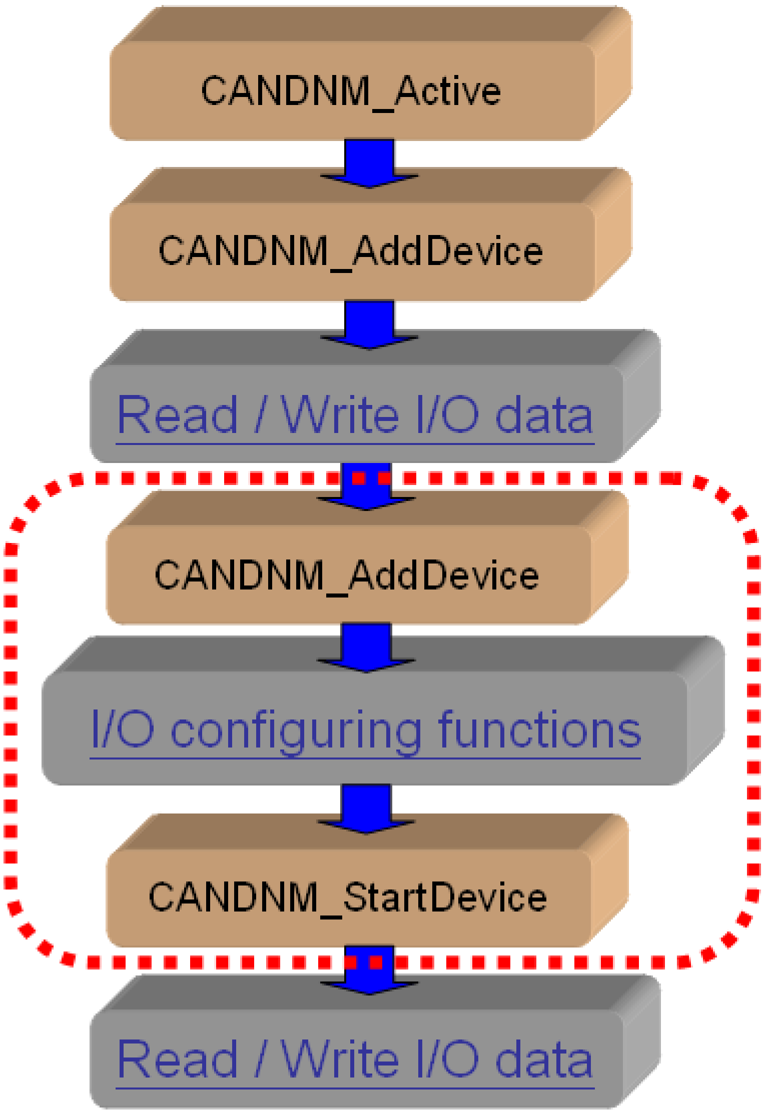


Figure 2.5-1 On-line Add Device Diagram

## 2. On-line Removing Devices :

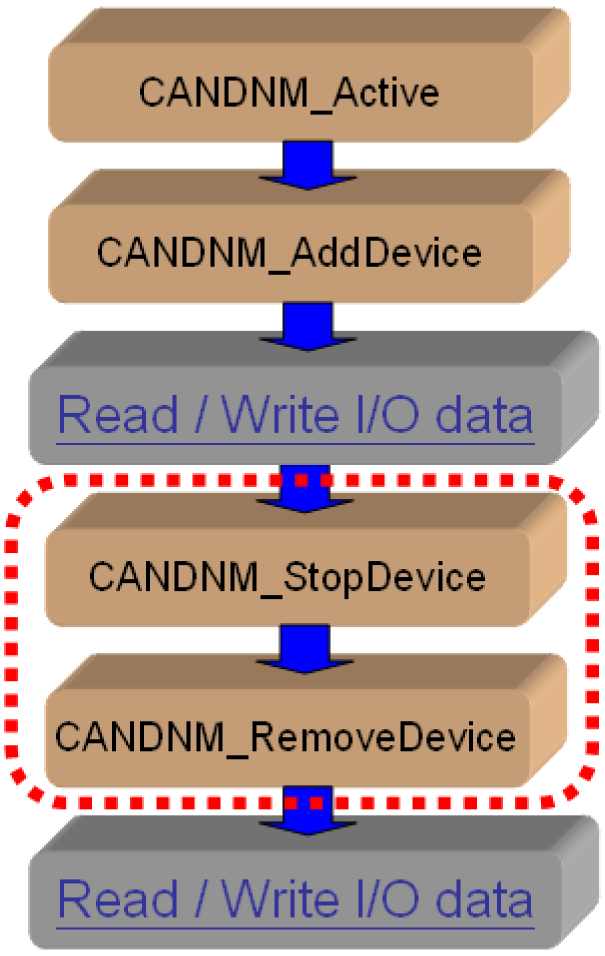


Figure 2.5-2 On-line Remove Device Diagram

## *2.6 Flow Diagram for “SetAttribute” and “GetAttribute”*

The users can set or get DeviceNet device's property via DeviceNet network. The DeviceNet master library provides these functions to set or get the properties of the remote devices easily. The steps are shown in Figure 2.6-1.

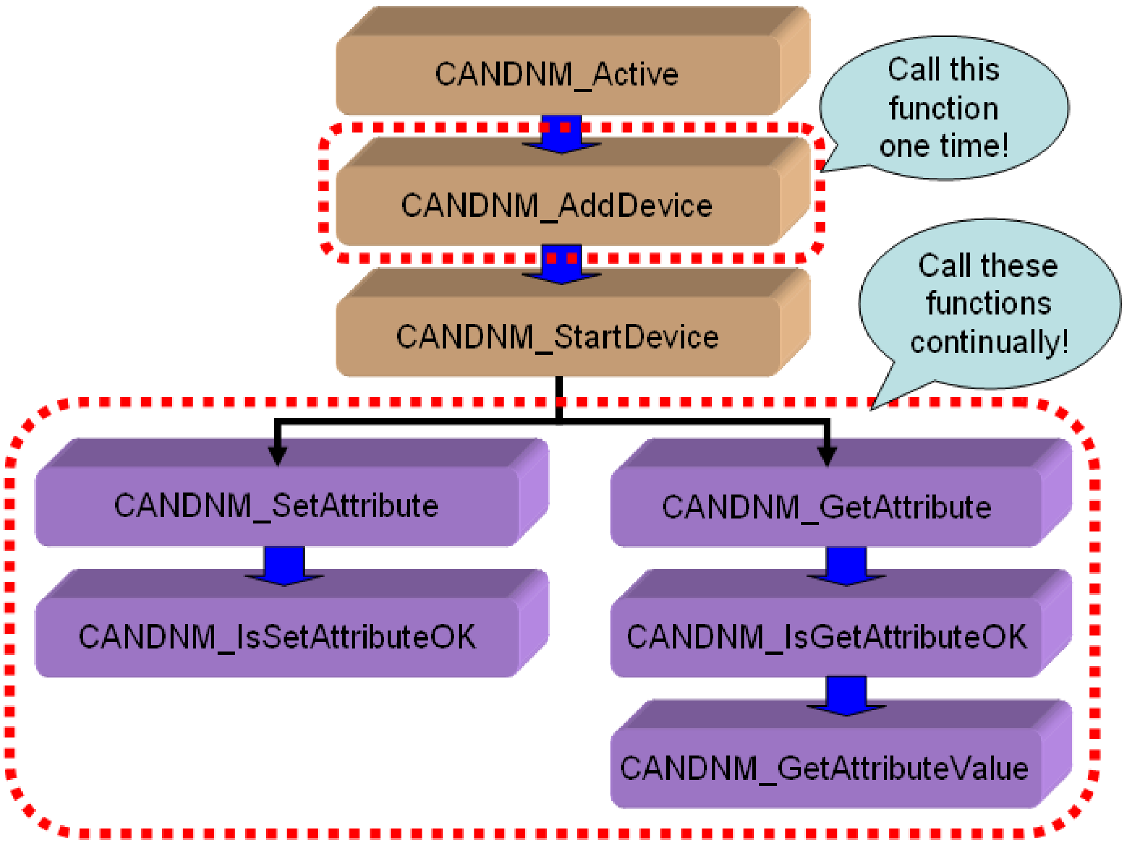


Figure 2.6-1 “SetAttribute” and “GetAttribute” Diagram

## 2.7 Flow Diagram for I/O Connection

The users can read or write device's I/O data via the DeviceNet I/O connections like Poll, Strobe, COS and Cyclic connection. There are four important steps to read and write the I/O data easily. Firstly, the users should know the device's I/O input length (in Byte) and output length (in Byte). Secondly, the users should set these two parameters by calling CANDNM_AddIOConnection. Thirdly, the users can set the initial output value by calling CANDNM_WriteOutputData before starting the specific slave device. If the users do not initialize the output value, the DeviceNet master library default output value is 0. Fourthly, the users can start communicating with devices to read or write I/O data. If the specific slave device doesn't have any output channel, the DeviceNet master library will start communicating with the device automatically. The Figure 2.7-1 shows the main steps to achieve this function. There are more functions described in chapter 4.

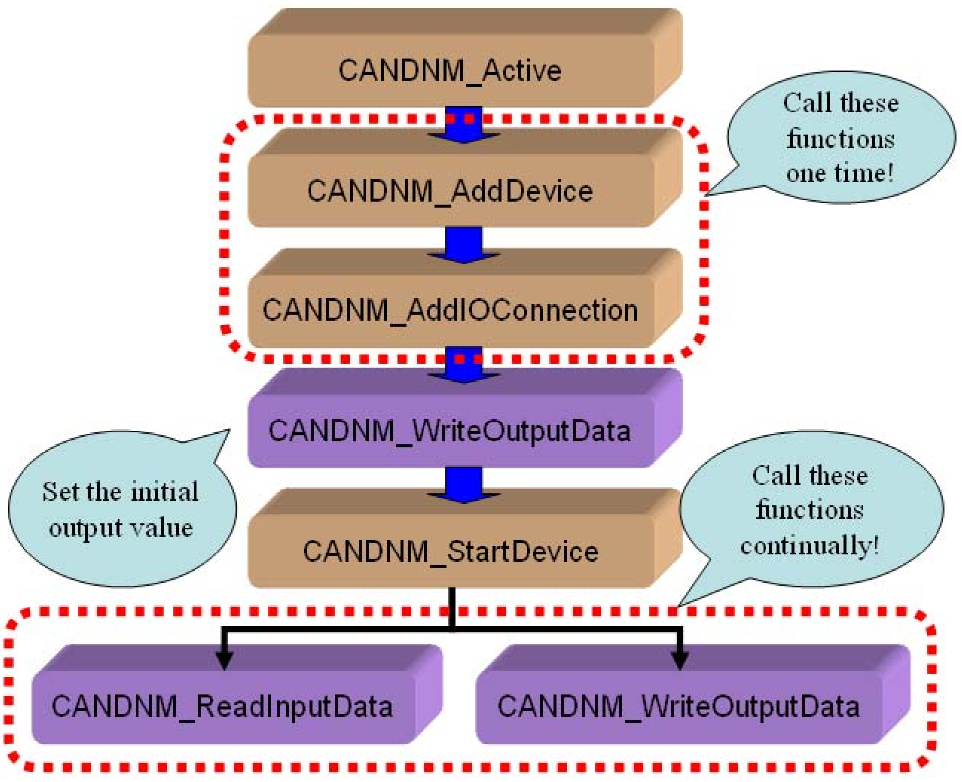


Figure 2.7-1 I/O Connection Diagram

Note: The Strobe connection doesn't support the output channel. The users can not use the CANDNM_WriteOutputData with Strobe connection.

# 3. Function description

All the functions of the PISOCANDNM.dll can be separated into four groups. The idea is shown Figure 3.0-1. There is more detail description in CH 3.1.

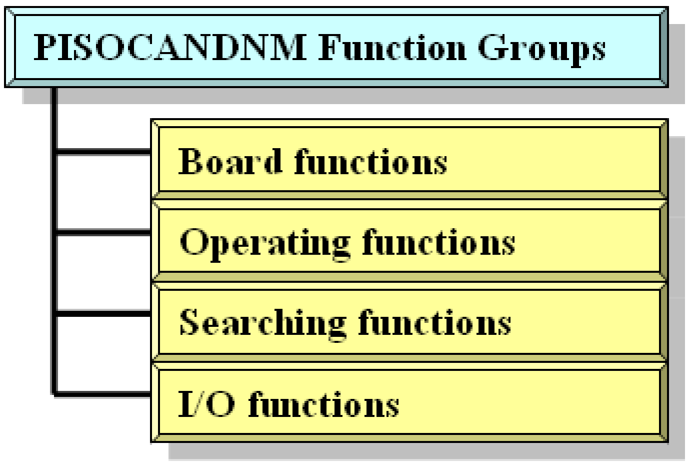


Figure 3.0-1 Four Function Groups

**[Board Functions]**

These functions in this group help users to find PISO-CAN 200/400 boards or get board's information. The users can use these functions to configure or manage the boards in the PC.

**[Operating Functions]**

These operating functions are the important operation of the DeviceNet master. They help users to configure the whole network.

**[Searching Functions]**

These searching functions can help user to debug the network, including the wire connection, the slave device's setting, and etc. When building the DeviceNet network, the user can use these functions to make sure that the network or the slave devices are fine.

**[I/O Functions]**

These functions help user to read or to write the I/O data from or to the remote slave devices.

## 3.1 DLL Function Definition and Description

All the functions provided in the PISOCANDNM.dll are listed in the following table and detail information for every function is presented in the next sub-section. However, in order to make the descriptions more simply and clear, the attributes for the both the input and output parameter functions are given as **[input]** and **[output]** respectively, as shown in the following table.

| Keyword | Set parameter by user before calling this function? | Get the data from this parameter after calling this function? |
|---|---|---|
| **[ input ]** | Yes | No |
| **[ output ]** | No | Yes |

Table 3.1-1 Keyword Table

Table 3.1.1 Functions Table (Board Functions) 1/1

| **No.** | **Function Name** | **Description** |
|---|---|---|
| **1** | CANDNM_GetBoardInf | Get the driver information of the PISO-CAN 200/400 |
| **2** | CANDNM_TotalPISOCANDNMBoard | Get total PISO-CAN 200/400 boards in the PC |
| **2** | CANDNM_Active | Enable the driver of the PISO-CAN 200/400 and activate the DeviceNet X-Server |
| **3** | CANDNM_Close | Close driver of the PISO-CAN 200/400 and DeviceNet X-Server |
| **4** | CANDNM_GetDLLVersion | Get the DLL version of the PISOCANDNM.DLL |
| **5** | CANDNM_GetDNMXSPCVersion | Get the version of the DeviceNet Master X-Server. |

Table 3.1.2 Functions Table (Operating Functions) 1/2

| No. | Function Name | Description |
|---|---|---|
| **1** | CANDNM_SetMasterMACID | Set the MAC ID of the PISO-CAN 200/400 board (DeviceNet Master's MAC ID) |
| **2** | CANDNM_GetMasterMACID | Get the MAC ID of the PISO-CAN 200/400 board (DeviceNet Master's MAC ID) |
| **3** | CANDNM_GetBaudRate | Get the baud rate of the CAN bus |
| **4** | CANDNM_SetBaudRate | Set the baud rate of the CAN bus |
| **5** | CANDNM_GetMasterStatus | Get the status of the PISO-CAN 200/400 board (DeviceNet Master's status) at present |
| **6** | CANDNM_GetSlaveStatus | Get the specific slave device's status. |
| **7** | CANDNM_StartDevice | PISO-CAN 200/400 will start to communicate with the specific slave device |
| **8** | CANDNM_StopDevice | PISO-CAN 200/400 will stop to communicate with the specific slave device |
| **9** | CANDNM_StartAllDevice | PISO-CAN 200/400 will start to communicate with all slave devices |
| **10** | CANDNM_StopAllDevice | PISO-CAN 200/400 will stop to communicate with all slave devices |
| **11** | CANDNM_AddDevice | Add the specific slave device's information into the PISO-CAN 200/400 board (DeviceNet Master) |
| **12** | CANDNM_RemoveDevice | Remove the specific slave device's information from the PISO-CAN 200/400 board (DeviceNet Master) |
| **13** | CANDNM_AddIOConnection | Add I/O information of the specific slave device into the PISO-CAN 200/400 board (DeviceNet Master) |
| **14** | CANDNM_RemoveIOConnection | Remove the specific slave device's I/O information from the PISO-CAN 200/400 board (DeviceNet Master) |

Table 3.1.3 Functions Table (Operating Functions) 2/2

| No. | Function Name | Description |
|---|---|---|
| **16** | CANDNM_GetAttribute | Send the get attribute command to the specific slave device. |
| **17** | CANDNM_IsGetAttributeOK | Check whether the slave has replied for the getting attribute command or not. |
| **18** | CANDNM_GetAttributeValue | Get the attribute value of the slave from the CANDNM_GetAttribute function |
| **19** | CANDNM_SetAttribute | Send the set attribute command to the slave device. |
| **20** | CANDNM_IsSetAttributeOK | Check whether the slave has replied for the setting command or not. |
| **21** | CANDNM_GetDeviceInfoFromScanList | Get specific slave device's I/O information from the Scan List within the DeviceNet master X-Server. |
| **22** | CANDNM_GetScanList | Get the I/O information of all slave devices form the Scan List within the DeviceNet master X-Server. |
| **23** | CANDNM_ImportScanList | Write the I/O information of all slave devices into the configuration file. |
| **24** | CANDNM_ClearAllConfig | Clear all configurations in the configuration file. |

Table 3.1.4 Functions Table (Searching Functions) 1/1

| No. | Function Name | Description |
|---|---|---|
| **1** | CANDNM_SearchAllDevices | PISO-CAN 200/400 will search the DeviceNet devices in the network and find out the I/O information of all slave devices. |
| **2** | CANDNM_SearchSpecificDevice | PISO-CAN 200/400 will search the DeviceNet network to find out the I/O information of specific slave devices. |
| **3** | CANDNM_IsSearchOK | Check whether the PISO-CAN 200/400 has searched completely or not. |
| **4** | CANDNM_GetSearchedDevices | Get the result of the searching command and retrieve the slave's I/O information. |

Table 3.1.5 Functions Table (I/O Functions) 1/1

| No. | Function Name | Description |
|---|---|---|
| **1** | CANDNM_ReadInputData | Read the input data via I/O connection like Poll, Strobe, COS, Cyclic. |
| **2** | CANDNM_WriteOutputData | Write the output data via I/O connection like Poll, COS, Cyclic. The Strobe doesn't support this operation. |

## *3.2 Function Return Code*

Table 3.2.1 Interpretation of the return code (Hardware Error) 1/1

| Return Code | Error ID | Comment |
|---|---|---|
| **1** | CAN_DriverError | Driver error. |
| **2** | CAN_ActiveBoardError | This board can not be activated. |
| **3** | CAN_BoardNumberError | The Board number exceeds the total board numbers. |
| **4** | CAN_PortNumberError | The Port number is not correct. |
| **5** | CAN_ResetError | CAN chip hardware reset error. |
| **6** | CAN_SoftResetError | CAN chip software reset error. |
| **7** | CAN_InitError | CAN chip initiation error. |
| **8** | CAN_ConfigError | CAN chip configure error. |
| **9** | CAN_SetACRError | Acceptance Code Register error. |
| **10** | CAN_SetAMRError | Set to Acceptance Mask Register error. |
| **11** | CAN_SetBaudRateError | Set Baud Rate error. |
| **12** | CAN_EnableRxIrqFailure | Enable CAN chip receive interrupt failure. |
| **13** | CAN_DisableRxIrqFailure | Disable CAN chip receive interrupt failure. |
| **14** | CAN_InstallIrqFailure | Installing PCI board IRQ failure. |
| **15** | CAN_RemoveIrqFailure | Removing PCI board IRQ failure. |
| **16** | CAN_TransmitBufferLocked | Transmit buffer in CAN chip is locked |
| **17** | CAN_TransmitIncomplete | Previously transmission is not yet completed. |
| **18** | CAN_ReceiveBufferEmpty | CAN chip RXFIFO is empty. |
| **19** | CAN_DataOverrun | Data was lost because there was not enough space in CAN chip RXFIFO |
| **20** | CAN_ReceiveError | Receive data is not completed |
| **21** | CAN_SoftBufferIsEmpty | Software buffer in driver is empty |
| **22** | CAN_SoftBufferIsFull | Software buffer in driver is full |
| **23** | CAN_TimeOut | Function no response and timeout |
| **24** | CAN_InstallIsrError | Installing user ISR failure |

Table 3.2.2 Interpretation of the return code (General Error) 1/1

| Return Code | Error ID | Comment |
|---|---|---|
| **5000** | DNMXS_UnKnowError | The DeviceNet has some unknown errors. |
| **1000** | DNMXS_BoardNotActive | The PISO-CAN 200/400 has not been activated. |
| **1001** | DNMXS_OnlineError | The master MAC ID collides with other slave device in the DeviceNet network. |
| **1002** | DNMXS_CANBusError | The CAN port can't send message. Please check the baud rate or the port of the CAN bus. |
| **1003** | DNMXS_Booting | The PISO-CAN 200/400 is still booting. |
| **1050** | DNMXS_MACIDError | The MAC ID is exceed the range(0 ~ 63) |
| **1051** | DNMXS_BaudRateError | The baud rate is exceed the range(0 ~ 2) |
| **1052** | DNMXS_ConnectionTypeError | The connection type is exceed the range (0 ~ 4) |
| **1053** | DNMXS_DuplicMasterMACID | The MAC ID is the same with the master’s ID. |
| **1054** | DNMXS_ConfigError | The configuration is out of order. |
| **1055** | DNMXS_NowScanning | The PISO-CAN 200/400 is searching the slave. |
| **1056** | DNMXS_ScanListError | The Scan List has some errors. |
| **1057** | DNMXS_DeviceExist | The information of the slave device already exists. |
| **1058** | DNMXS_DeviceNotExist | The information of the slave device doesn’t exist. |
| **1059** | DNMXS_MapTableError | The MapTable has some errors. |

Table 3.2.3 Interpretation of the return code (I/O Error) 1/1

| Return Code | MapTable Error | Comment |
|---|---|---|
| **1100** | DNMXS_ExplicitNotAllocate | The Explicit connection is not established. |
| **1101** | DNMXS_PollNotAllocate | The Poll connection is not established. |
| **1102** | DNMXS_BitStrobeNotAllocate | The Strobe connection is not established. |
| **1103** | DNMXS_COSNotAllocate | The COS connection is not established. |
| **1104** | DNMXS_CyclicNotAllocate | The Cyclic connection is not established. |
| **1105** | DNMXS_PollAlreadyExist | The Poll connection has been established. |
| **1106** | DNMXS_BitStrobeAlreadyExist | The Poll connection has been established. |
| **1107** | DNMXS_COSAlreadyExist | The COS connection has been established. |
| **1108** | DNMXS_CyclicAlreadyExist | The Cyclic connection has been established. |
| **1109** | DNMXS_CommunicationPause | The communication between PISO-CAN 200/400 and all slave devices has been suspended. |

Table 3.2.4 Interpretation of the return code (Slave Error) 1/1

| Return Code | DeviceNet Error | Comment |
|---|---|---|
| **1150** | DNMXS_SlaveNoResp | The slave has no any response. |
| **1151** | DNMXS_WaitForSlaveResp | The PISO-CAN 200/400 is waiting for the response form the slave device. |
| **1152** | DNMXS_SlaveRespError | The slave replied some errors. |
| **1153** | DNMXS_OutputDataLenError | The output length of the I/O connection doesn't match the device's output length. |
| **1154** | DNMXS_InputDataLenError | The input length of the I/O connection doesn't match the device's input length. |

## *3.3 Function Description*

### 3.3.1 CANDNM_GetBoardInfo

- **Description:**

  This function is used to obtain the driver information of PISO-CAN 200/400 board.

- **Syntax:**

```
DWORD CANDNM_GetBoardInf (BYTE BoardNo, DWORD *dwVID,
                          DWORD *dwDID, DWORD *dwSVID,
                          DWORD *dwSDID, DWORD *dwSAuxID,
                          DWORD *dwIrqNo)
```

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **dwVID**: [output] The address of a variable which is used to receive the vendor ID.
  **dwDID**: [output] The address of a variable used to receive device ID.
  **dwSVID**: [output] The address of a variable applied to receive sub-vendor ID.
  **dwSDID**: [output] The address of a variable applied to receive sub-device ID.
  **dwSAuxID**: [output] The address of a variable used to receive sub-auxiliary ID.
  **dwIrqNo**: [output] The address of a variable used to receive logical interrupt number.

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

## 3.3.2 CANDNM_TotalPISOCANDNMBoard

- **Description:**

  The function can get the count of total PISO-CAN 200/400 boards in the user's PC.

- **Syntax:**

```
DWORD CANDNM_TotalPISOCANDNMBoard (BYTE *TotalBoards ,
                                   BYTE *BoardIDList,
                                   BYTE *PortNumList)
```

- **Parameter:**

  **TotalBoards:** [output] The amount of total PISO-CAN 200/400 boards.
  **BoardIDList:** [output] The list of all board No. in each boards.
  **PortNumList:** [output] The port amount of all boards.

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

### 3.3.3 CANDNM_Active

- **Description:**

  The function is used to activate the PISO-CAN 200/400 board and the DeviceNet master X-Server. It must be called once before using the other functions of PISO-CAN 200/400 APIs.

- **Syntax:**

  DWORD CANDNM_Active (BYTE BoardNo, BYTE PortNo)

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

### 3.3.4 CANDNM_Close

- **Description:**

  The function is used to stop and close the kernel driver and release the device resource from computer device resource. The DeviceNet Master X-Server would also be closed at the same time. This method must be called once before exiting the user's application program.

- **Syntax:**

  DWORD CANDNM_Close (BYTE BoardNo, BYTE PortNo)

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

### 3.3.5 CANDNM_GetDLLVersion

- **Description:**

  The function can obtain the version information of PISOCANDNM.DLL.

- **Syntax:**

  DWORD CANDNM_GetDLLVersion (void)

- **Parameter:**

  None

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

## 3.3.6 CANDNM_GetDNMXSPCVersion

- **Description:**

  The function can obtain the version information of the DeviceNet Master X-Server.

- **Syntax:**

  DWORD CANDNM_GetDNMXSPCVersion (void)

- **Parameter:**

  None

- **Return:**

  The version information. For example: If 100(hex) is return, it means the version is 1.00.

- **Error Return:**

  Please refer to the chapter 3.2 for the function return code.

### 3.3.7 CANDNM_GetMasterMACID

- **Description:**

  The function can get the MAC ID of the DeviceNet master (PISO-CAN 200/400).

- **Syntax:**

  DWORD CANDNM_GetMasterMACID (BYTE BoardNo, BYTE Port)

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

## 3.3.8 CANDNM_SetMasterMACID

- **Description:**

  The function can set the MAC ID of the DeviceNet master (PISO-CAN 200/400). After calling this function, the users must restart your application to make the change enabled. It will save the information in the configuration file.

- **Syntax:**

```
    DWORD CANDNM_SetMasterMACID (BYTE BoardNo, BYTE PortNo
                                 BYTE MasterMACID)
```

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)
  **MasterMACID:** [input] The new MAC ID of the master. (0 ~ 63)

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

## 3.3.9 CANDNM_GetBaudRate

- **Description:**

  This function can help you to get the DeviceNet baud rate information of PISO-CAN 200/400.

- **Syntax:**

  DWORD CANDNM_GetBaudRate (BYTE BoardNo, BYTE PortNo)

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)

- **Return:**

  The CAN bus baud rate information in the PISO-CAN 200/400.
  If the value is 0, the baud rate is 125Kbps.
  If the value is 1, the baud rate is 250Kbps.
  If the value is 2, the baud rate is 500Kbps.

- **Error Return:**

  Please refer to the chapter 3.2 for the function return code.

## 3.3.10 CANDNM_SetBaudRate

- **Description:**

  This function can set the DeviceNet baud rate of the PISO-CAN 200/400. After calling this function, you must restart your application to make change enabled. The baud rate information will be saved in the configuration file. It will be loaded in the next boot-up.

- **Syntax:**

```
DWORD CANDNM_SetBaudRate (BYTE BoardNo, BYTE PortNo,
                          BYTE BaudRate)
```

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board’s port number (0~3)
  **BaudRate:** [input] The new baud rate value.
  0 : 125K bps
  1 : 250K bps
  2 : 500K bps

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

### 3.3.11 CANDNM_GetMasterStatus

- **Description:**

  The function is used to obtain the DeviceNet master status. The users can call this function to make sure that the DeviceNet master is working fine.

- **Syntax:**

  DWORD CANDNM_GetMasterStatus (BYTE BoardNo, BYTE PortNo)

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

### 3.3.12 CANDNM_GetSlaveStatus

- **Description:**

  This function is to get the remote slave device's communication status.

- **Syntax:**

  ```
  DWORD CANDNM_GetSlaveStatus (BYTE BoardNo, BYTE PortNo,
                               BYTE DesMACID)
  ```

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)
  **DesMACID:** [input] The remote slave's MAC ID. (0~63)

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

### 3.3.13 CANDNM_StartDevice

- **Description:**

  This function is used to start to communicate with the specific device that the users applying to.

- **Syntax:**

```
DWORD CANDNM_StartDevice (BYTE BoardNo, BYTE PortNo,
                          BYTE DesMACID)
```

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)
  **DesMACID:** [input] The remote slave's MAC ID. (0~63)

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

### 3.3.14 CANDNM_StopDevice

- **Description:**

  This function is used to stop to communicate with the destination device that the users appointed to.

- **Syntax:**

```
DWORD CANDNM_StopDevice (BYTE BoardNo, BYTE PortNo,
                         BYTE DesMACID)
```

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)
  **DestMACID:** [input] The remote slave device's MAC ID (0~63)

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

### 3.3.15 CANDNM_StartAllDevice

- **Description:**

  This function is used to start to communicate with all slave devices in ScanList.

- **Syntax:**

  DWORD CANDNM_StartAllDevice (BYTE BoardNo, BYTE PortNo)

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

### 3.3.16 CANDNM_StopAllDevice

- **Description:**

  This function is used to stop to communicate with all destination devices in ScanList.

- **Syntax:**

  DWORD CANDNM_StopAllDevice (BYTE BoardNo, BYTE PortNo)

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

### 3.3.17 CANDNM_AddDevice

- **Description:**

  This function can add the slave devices into the ScanList of the PISO-CAN 200/400 and save the information into the configuration file. Before communicating with the slave devices, the users should call this function.

- **Syntax:**

```
DWORD CANDNM_AddDevice (BYTE BoardNo, BYTE PortNo,
                        BYTE DesMACID, WORD Explicit_EPR)
```

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)
  **DestMACID:** [input] The remote slave device's MAC ID (0~63)
  **Explicit_EPR:** [input] The Expected Packet Rate. (Usually is 2500).

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

## 3.3.18 CANDNM_RemoveDevice

- **Description:**

  This function is used for removing the specified slave device from the ScanList. And the information of the device in configuration file is erased at the same time.

- **Syntax:**

  DWORD CANDNM_RemoveDevice (BYTE BoardNo, BYTE PortNo, BYTE DesMACID)

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)
  **DestMACID:** [input] The remote slave device's MAC ID (0~63)

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

### 3.3.19 CANDNM_AddIOConnection

- **Description:**

  This method is used to configure the I/O connection of the specific MAC ID device. The PISO-CAN 200/400 can get/set the data via the connection to the specific slave, according to the produced / consumed connection path of this slave device. This configuration data will be saved into the configuration file.

- **Syntax:**

```
DWORD CANDNM_AddIOConnection (BYTE BoardNo, BYTE PortNo,
                              BYTE DesMACID, BYTE ConType,
                              WORD DeviceInputLen,
                              WORD DeviceOutputLen,
                              WORD EPR)
```

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)
  **DestMACID:** [input] The remote slave device's MAC ID (0~63)
  **ConType:** [input] The remote slave device's I/O connection type
  0 : Explicit connection type
  1 : Poll connection type
  2 : Bit-Strobe connection type
  3 : COS connection type
  4 : Cyclic connection type
  **DeviceInputLen:** [input] The remote slave device's input length. (Byte)
  **DeviceOutputLen:** [input] The remote slave device's output length. (Byte)
  **EPR:** [input] The expected packet rate. (mSec)
  (Don't less than 50 mSec)

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

## 3.3.20 CANDNM_RemoveIOConnection

- **Description:**

The function is used to remove the I/O connection configuration.

- **Syntax:**

```
DWORD CANDNM_RemoveIOConnection (BYTE BoardNo,
                                 BYTE PortNo,
                                 BYTE DesMACID,
                                 BYTE ConType)
```

- **Parameter:**

**BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
**PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)
**DestMACID:** [input] The remote slave device's MAC ID (0~63)
**ConType:** [input] The remote slave device's I/O connection type

0 : Explicit connection type
1 : Poll connection type
2 : Bit-Strobe connection type
3 : COS connection type
4 : Cyclic connection type

- **Return:**

Please refer to the chapter 3.2 for the function return code.

## 3.3.21 CANDNM_GetAttribute

- **Description:**

  This function is used to send the request command to retrieve the attribute value of the specific device's instance. Before calling this function, you must start communicating with the device by CANDNM_StartDevice function. After calling CANDNM_GetAttribute function, you should execute the "CANDNM_GetAttributeValue" to get the response message returned from remote slave device.

- **Syntax:**

```
DWORD CANDNM_GetAttribute (BYTE BoardNo, BYTE PortNo,
                           BYTE DesMACID, BYTE ClassID,
                           BYTE InstanceID, BYTE AttributeID)
```

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)
  **DestMACID:** [input] The remote slave device's MAC ID (0~63)
  **ClassID:** [input] The remote slave device's ClassID
  **InstanceID:** [input] The remote slave device's InstanceID
  **AttributeID:** [input] The remote slave device's AttributeID

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

## 3.3.22 CANDNM_IsGetAttributeOK

- **Description:**

  This function is used to check whether the PISO-CAN 200/400 has received the response message or not. After checking the response message, you should execute the "CANDNM_GetAttributeValue" to get the response message returned from remote slave device.

- **Syntax:**

```
DWORD CANDNM_IsGetAttributeOK (BYTE BoardNo, BYTE PortNo,
                               BYTE DesMACID)
```

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)
  **DestMACID:** [input] The remote slave device's MAC ID (0~63)

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

## 3.3.23 CANDNM_GetAttributeValue

- **Description:**

  This function is used to get the attribute value of the specific device's instance from the remote slave device. Before calling this function, the users should call CANDNM_GetAttribute to send request command first.

- **Syntax:**

```
DWORD CANDNM_GetAttributeValue (BYTE BoardNo, BYTE PortNo,
                                BYTE DesMACID,
                                BYTE *DATA, WORD *DataLen)
```

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)
  **DestMACID:** [input] The remote slave device's MAC ID (0~63)
  **DATA:** [output] The attribute value that returned from the slave device.
  **DataLen:** [output] The length of the attribute value (in byte).

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

### 3.3.24 CANDNM_SetAttribute

- **Description:**

  The method is used to set the attribute of the specific device's instance. Before calling this function, you must start communicating with the device by CANDNM_StartDevice function. After calling CANDNM_SetAttribute function, you should execute the "CANDNM_IsSetAttributeOK" to check the response message returned from the remote slave device.

- **Syntax:**

```
DWORD CANDNM_SetAttribute (BYTE BoardNo, BYTe PortNo,
                           BYTE DesMACID, BYTE ClassID,
                           BYTE InstanceID, BYTE AttributeID,
                           WORD DataLen, BYTE *DATA)
```

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)
  **DestMACID:** [input] The remote slave device's MAC ID (0~63)
  **ClassID:** [input] The remote slave device's ClassID
  **InstanceID:** [input] The remote slave device's InstanceID
  **AttributeID:** [input] The remote slave device's AttributeID
  **DataLen:** [input] The length of the attribute value (in byte).
  **DATA:** [input] The attribute value that the users want to send.

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

## 3.3.25 CANDNM_IsSetAttributeOK

- **Description:**

  This function is used to check the response value after executing the "CANDNM_SetAttribute" function.

- **Syntax:**

```
DWORD CANDNM_IsSetAttributeOK (BYTE BoardNo, BYTE PortNo,
                               BYTE DesMACID)
```

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)
  **DestMACID:** [input] The remote slave device's MAC ID (0~63)

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

## 3.3.26 CANDNM_ClearAllConfig

- **Description:**

  This function will clear all data in the configuration file and reset the master ID to 0 and the baud rate to 125 Kbps.

- **Syntax:**

  DWORD CANDNM_ClearAllConfig (BYTE BoardNo, BYTE PortNo)

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board’s port number (0~3)

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

### 3.3.27 CANDNM_SearchAllDevices

- **Description:**

  This function is used to retrieve all devices in DeviceNet network. Attention! This function will terminate all communications with remote devices. This function is usually used for developing or debugging applications.

- **Syntax:**

  DWORD CANDNM_SearchAllDevices (BYTE BoardNo, BYTE PortNo)

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

## 3.3.28 CANDNM_SearchSpecificDevice

- **Description:**

  This function is used to retrieve some devices which specified by the users. Attention! This function will terminate all communications with remote devices. This function is usually used for developing or debugging applications.

- **Syntax:**

```
DWORD CANDNM_SearchSpecificDevice (BYTE BoardNo,
                                    BYTE PortNo,
                                   WORD ListCount,
                                   BYTE *DesMACIDList)
```

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)
  **ListCount:** [input] The amount of the slave's ID.
  **DestMACIDList:** [input] The list of all slave's MAC ID.

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

## 3.3.29 CANDNM_IsSearchOK

- **Description:**

  This function will check whether the searching process has finished or not.

- **Syntax:**

  DWORD CANDNM_IsSearchOK (BYTE BoardNo, BYTE PortNo)

- **Parameter:**

  **BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
  **PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)

- **Return:**

  Please refer to the chapter 3.2 for the function return code.

## 3.3.30 CANDNM_GetSearchedDevices

- **Description:**

This function will get the information of the slave devices searched in the network

- **Syntax:**

```
DWORD CANDNM_GetSearchedDevices (BYTE BoardNo,
                                 BYTE PortNo,
                                 WORD *TotalDevices,
                                 BYTE *DesMACID,
                                 BYTE *Type,
                                 WORD *DeviceInputLen,
                                 WORD *DeviceOutputLen)
```

- **Parameter:**

**BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
**PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)
**TotalDevices:** [output] The amount of all slave device which are found.
**DesMACID:** [output] The list of slave's MAC ID which are found.
**Type:** [output] The list of slave's connection type which are found.
0 : Explicit connection type
1 : Poll connection type
2 : Bit-Strobe connection type
3 : COS connection type
4 : Cyclic connection type

**DeviceInputLen:** [output] The list of slave's input length which are found.
**DeviceOutputLen:** [output] The list of slave's output length which are found.

- **Return:**

Please refer to the chapter 3.2 for the function return code.

### 3.3.31 CANDNM_GetDeviceInfoFromScanList

- **Description:**

This function will get the ScanList data of the certain device from the configuration file.

- **Syntax:**

```
DWORD CANDNM_GetDeviceInfoFromScanList
                    (BYTE BoardNo, BYTE PortNo, BYTE DesMACID,
                    WORD *ListCount, BYTE *ConnectionTypeList,
                    WORD *InputDataLenList, WORD *OutputDataLenList,
                    WORD *EPRList)
```

- **Parameter:**

**BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
**PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)
**DesMACID:** [input] The remote slave's MAC ID. (0~63).
**ListCount:** [output] The amount of all information items.
**ConnectionTypeList:** [output] The list of slave's connection type.

0 : Explicit connection type
1 : Poll connection type
2 : Bit-Strobe connection type
3 : COS connection type
4 : Cyclic connection type

**InputDataLenList:** [output] The list of slave's input length.
**OutputDataLenList:** [output] The list of slave's output length.
**EPRList:** [output] The list of slave's expected packet rate.

- **Return:**

Please refer to the chapter 3.2 for the function return code.

### 3.3.32 CANDNM_GetScanList

- **Description:**

This function will get all the ScanList data from the configuration file.

- **Syntax:**

```
DWORD CANDNM_GetScanList (BYTE BoardNo, BYTE PortNo,
                          WORD *TotalDevices,
                          BYTE *DesMACIDList,
                          BYTE *ConnectionTypeList,
                          WORD *InputDataLenList,
                          WORD *OutputDataLenList,
                          WORD *EPR_List)
```

- **Parameter:**

**BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
**PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)
**TotalDevices:** [output] The data count of all the information.
**DestMACIDList:** [output] The MAC ID of all the slave devices in the ScanList.
**ConnectionTypeList:** [output] The connection type of all the slave devices in the ScanList.

0 : Explicit connection type
1 : Poll connection type
2 : Bit-Strobe connection type
3 : COS connection type
4 : Cyclic connection type

**InputDataLenList:** [output] The input data length of all the slave devices in the ScanList.
**OutputDataLenList:** [output] The output data length of all the slave devices in the ScanList.
**EPR_List:** [output] The EPR value of all the slave devices in the ScanList.

- **Return:**

Please refer to the chapter 3.2 for the function return code.

## 3.3.33 CANDNM_ImportScanList

- **Description:**

This function will write all specific devices' information in the ScanList to the configuration file.

- **Syntax:**

```
DWORD CANDNM_ImportScanList (BYTE BoardNo, BYTE PortNo,
                             WORD ListCount,
                             BYTE *DesMACIDList,
                             BYTE *ConnectionTypeList,
                              WORD *InputDataLenList,
                              WORD *OutputDataLenList,
                              WORD *EPR_List)
```

- **Parameter:**

**BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
**PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)
**ListCount:** [input] The data count of all the information.
**DestMACIDList:** [input] The MAC ID of all the slave devices in the ScanList.
**ConnectionTypeList:** [input] The connection type of all slave devices.

0 : Explicit connection type
1 : Poll connection type
2 : Bit-Strobe connection type
3 : COS connection type
4 : Cyclic connection type

**InputDataLenList:** [input] The input data length of all slave devices.
**OutputDataLenList:** [input] The output data length of all slave devices.
**EPR_List:** [input] The EPR value of all slave devices.

- **Return:**

Please refer to the chapter 3.2 for the function return code.

## 3.3.34 CANDNM_ReadInputData

- **Description:**

This function is to get the data according with the produced connection path of the specific MAC ID device via the I/O connection.

- **Syntax:**

```
DWORD CANDNM_ReadInputData (BYTE BoardNo, BYTE PortNo,
                            BYTE DesMACID, BYTE ConType,
                            WORD *IOLen, BYTE *IODATA)
```

- **Parameter:**

**BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
**PortNo**: [input] PISO-CAN 200/400 board’s port number (0~3)
**DestMACID:** [input] The remote slave device’s MAC ID (0~63)
**ConType:** [input] The connection type of the remote slave.
0 : Explicit connection type
1 : Poll connection type
2 : Bit-Strobe connection type
3 : COS connection type
4 : Cyclic connection type

**IOLen:** [output] The length of the I/O data (In byte).
**IODATA:** [output] The remote I/O data.

- **Return:**

Please refer to the chapter 3.2 for the function return code.

### 3.3.35 CANDNM_WriteOutputData

- **Description:**

The function will set the data according with the consumed connection path of the specific MAC ID device via the I/O connection.

- **Syntax:**

```
DWORD CANDNM_WriteOutputData (BYTE BoardNo, BYTE PortNo,
                              BYTE DesMACID, BYTE ConType,
                              WORD IOLen, BYTE *IODATA)
```

- **Parameter:**

**BoardNo**: [input] PISO-CAN 200/400 board number (0~15)
**PortNo**: [input] PISO-CAN 200/400 board's port number (0~3)
**DestMACID:** [input] The remote slave device's MAC ID (0~63)
**ConType:** [input] The connection type of the remote slave.

0 : Explicit connection type
1 : Poll connection type
2 : Bit-Strobe connection type
3 : COS connection type
4 : Cyclic connection type

**IOLen:** [Input] The length of the I/O data (In byte).
**IODATA:** [Input] The remote I/O data.

- **Return:**

Please refer to the chapter 3.2 for the function return code.

# 4.Demo Programs for Windows

All of demo programs will not work normally if PISO-CAN200/400 driver would not be installed correctly. During the installation process of the driver, the install-shields will register the correct kernel driver to the operation system and copy the DLL driver and demo programs to the correct position based on the driver software package you have selected (Win98,Me,NT,win2000,XP). After completing the driver and DeviceNet library installation, the related demo programs, development library and declaration header files for different development environments are installed in the system as follows.

## **The list of demo programs**:

| | |
|---|---|
| DEMO1: | Devices communicate by the Explicit Message Connection. |
| DEMO2: | Devices communicate by the I/O Connection. |

## A brief introduction to the demo programs

### DEMO1:

Demo1 is the example used for starting the DeviceNet communication. This demo program is designed to communicate with slave device through Explicit Message Connection. Users apply the GetAttribute and SetAttribute function to access the information data of the slave devices. Before exercising this demo, the user needs to finish the wire connection between the Master and slave device. (See figure 4.1)

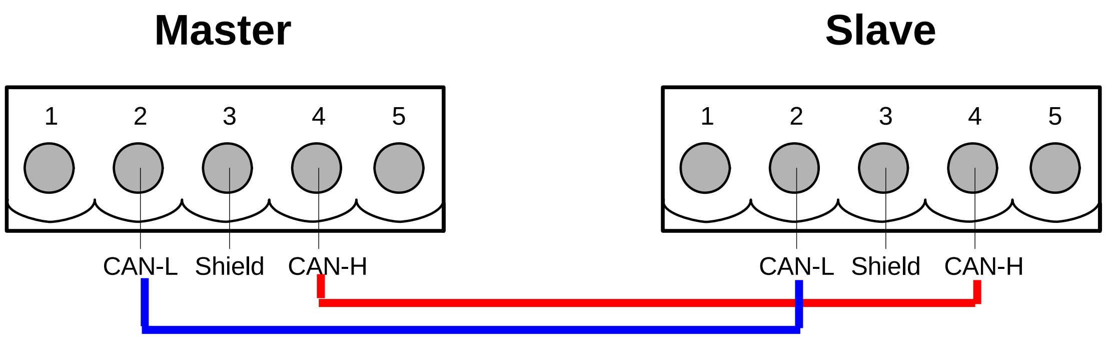
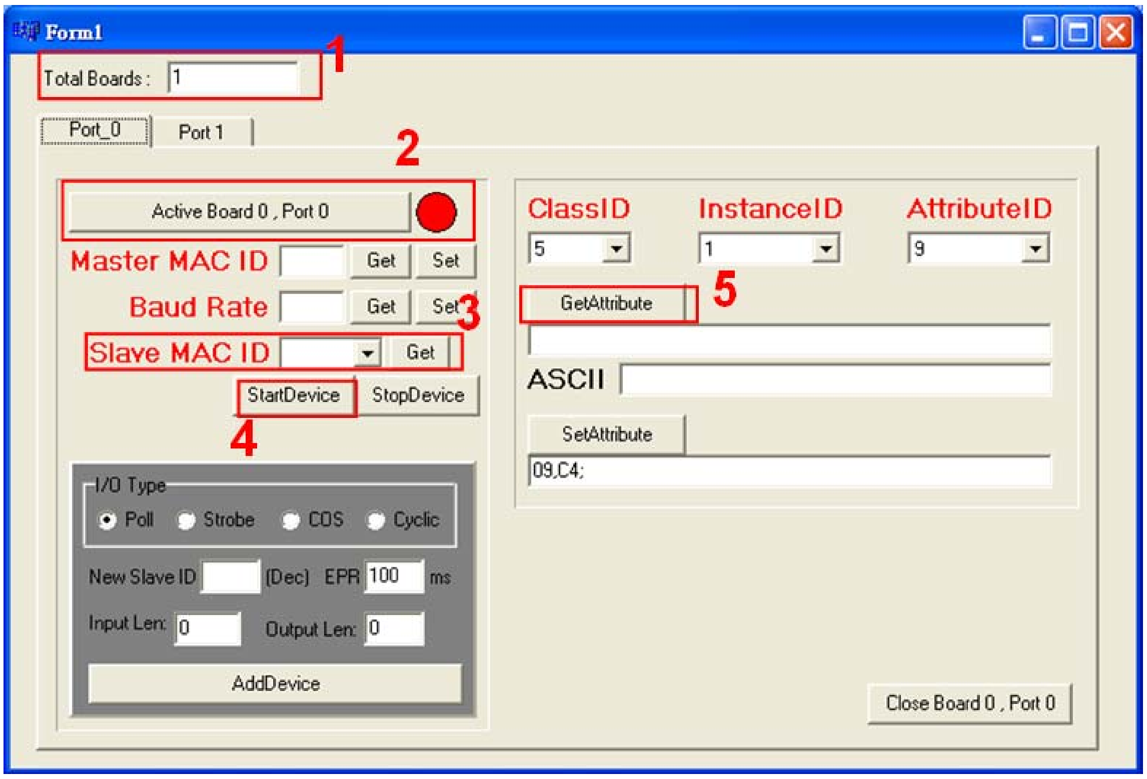


Figure 4.1: The wire connection



Total Boards : 1

Master MAC ID

Baud Rate

Slave MAC ID

I/O Type: [✓ Poll] Strobe COS Cyclic

New Slave ID (Dec) EPR 100 ms

Input Len: 0 Output Len: 0

ClassID: 5

InstanceID: 1

AttributeID: 9

ASCII

09,C4;

Figure 4.2: the form of demo1

Step 1 : The application will scan the PISO-CAN 200/400 in the user’s PC.

Step 2 : Active the Port 0 of the PISO-CAN 200/400 and DeviceNet Master X-Server. The PISO-CAN 200/400 will send two “Duplicate Check Message” to the CAN bus. Please wait for 2 seconds to make sure that the DeviceNet Master X-Server boot up.

Step 3 : After the DeviceNet Master X-Server has boot-up, the users can get all slave IDs. If the users have no any device information, you need to add at least one slave device. After clicking the “Get” button, it will put all slave IDs into the drop down list.

Step 4 : Before clicking the “StartDevice” button, the users should turn on the corresponding slave device. After clicking the “StartDevice” button, the DeviceNet Master will communicate with the slave device. The users can add you own device by using the “AddDevice” button which located at the left and the bottom of this form.

Step 5 : The users can select the ClassID, InstanceID and AttributeID and then click “GetAttribute” to get the corresponding attribute value form the remote slave device.

Based on this demo (See figure 4.2), the users can get the attribute value of the slave device. And users can set the attribute value of the slave device by keying the DeviceNet message into the SetAttribute frame area and then click the “SetAttribute” button to send out the DeviceNet message to the slave device through the DeviceNet network. For example, you can select ClassID = 1, InstanceID = 1, AttributeID = 7, then press “GetAttribute” button; you will get the slave product name in the ASCII field.

## DEMO2:

In demo 2, we provide a demonstration on how to communicate with slave device through the I/O Connection. The users can read or write the I/O data.

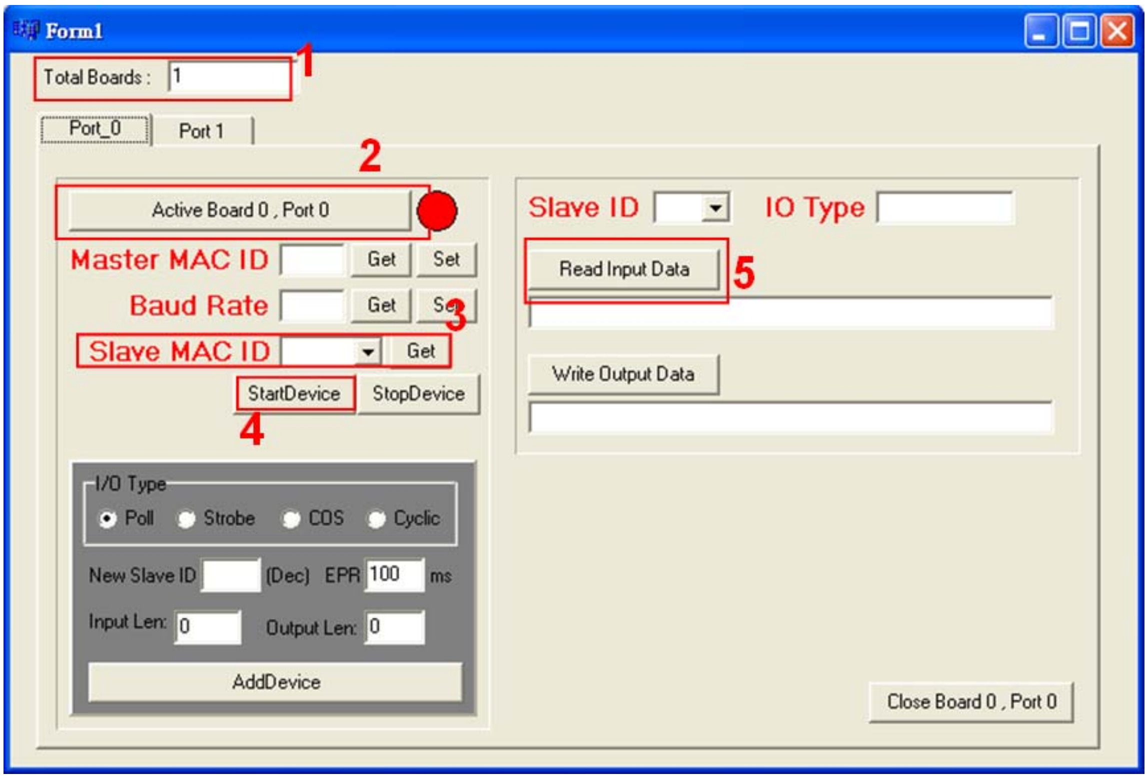


Figure 4.3: The form of demo2 program

Step 1 : The application will scan the PISO-CAN 200/400 in the user's PC.

Step 2 : Active the Port 0 of the PISO-CAN 200/400 and DeviceNet Master X-Server. The PISO-CAN 200/400 will send two "Duplicate Check Message" to the CAN bus. Please wait for 2 seconds to make sure that the DeviceNet Master X-Server has booted up.

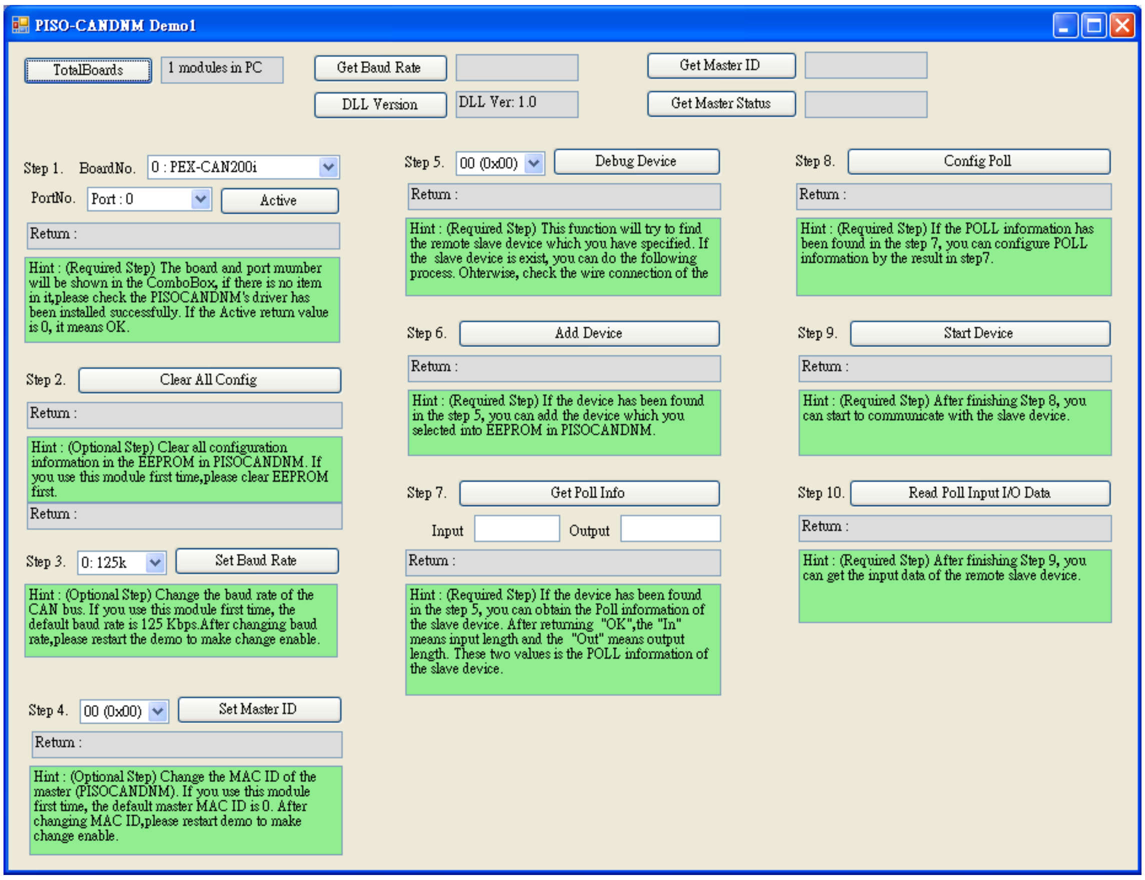
Step 3 : After the DeviceNet Master X-Server has boot-up, the users can get all slave IDs. After clicking the "Get" button, it will put all slave IDs into the drop down list.

Step 4 : Before clicking the "StartDevice" button, the users should turn on the corresponding slave device. After clicking the "StartDevice" button, the DeviceNet Master will communicate with the slave device. The users can add you own device by using the "AddDevice" button which located at the left and the bottom of this form.

Step 5 : The users can select the "Slave ID" which want to access the I/O data. And then the user can press "Read Input Data" to get the corresponding input data value form the remote slave device. If the users have the output length of the slave device, you can send the output data to the slave device.

## C# DEMO:

In the C# demo, we provide a demonstration on how to communicate with slave device. Before using the demo, the users need one DeviceNet slave I/O module.



1 modules in PC

DLL Ver: 1.0

Step 1. BoardNo. 0 : PEX-CAN200i

PortNo. Port : 0

Return :

Hint : (Required Step) The board and port number will be shown in the ComboBox, if there is no item in it,please check the PISOCANDNM's driver has been installed successfully. If the Active return value is 0, it means OK.

Step 2.

Return :

Hint : (Optional Step) Clear all configuration information in the EEPROM in PISOCANDNM. If you use this module first time,please clear EEPROM first.

Return :

Step 3. 0: 125k

Hint : (Optional Step) Change the baud rate of the CAN bus. If you use this module first time, the default baud rate is 125 Kbps.After changing baud rate,please restart the demo to make change enable.

Step 4. 00 (0x00)

Return :

Hint : (Optional Step) Change the MAC ID of the master (PISOCANDNM). If you use this module first time, the default master MAC ID is 0. After changing MAC ID,please restart demo to make change enable.

Step 5. 00 (0x00)

Return :

Hint : (Required Step) This function will try to find the remote slave device which you have specified. If the slave device is exist, you can do the following process. Otherwise, check the wire connection of the

Step 6.

Return :

Hint : (Required Step) If the device has been found in the step 5, you can add the device which you selected into EEPROM in PISOCANDNM.

Step 7.

Input Output

Return :

Hint : (Required Step) If the device has been found in the step 5, you can obtain the Poll information of the slave device. After returning "OK",the "In" means input length and the "Out" means output length. These two values is the POLL information of the slave device.

Step 8.

Return :

Hint : (Required Step) If the POLL information has been found in the step 7, you can configure POLL information by the result in step7.

Step 9.

Return :

Hint : (Required Step) After finishing Step 8, you can start to communicate with the slave device.

Step 10.

Return :

Hint : (Required Step) After finishing Step 9, you can get the input data of the remote slave device.