COMIZOA

Field Bus NETWORK LX412a/LX412b

TEST & MEASUREMENT & AUTOMATION

C-NET

APRIL 2008
P/N 0421-2008-4

Product Manual

CNETSDK Manual

Visit our web page at http://www.comizoa.com
For support requests, contact us at support@comizoa.com
For documentation suggestions, corrections, or requests, contact tech@comizoa.com

고객 기술 지원

전자 우편 : support@comizoa.com
파일 서버 : ftp.comizoa.com
웹 사이트 : http://www.comizoa.com

본사 안내
대전 광역시 유성구 탑립동 대덕테크노밸리 아이 2-2-2 블록 DTV-PostBI 센터

전화번호 : 042-936-6500
모사전송 : 042-936-6507

C-NET Product Manual

Table of Contents

Chapter

1

Introduction

본 장에서는 제품 보증안내를 비롯한 제품 보증 규정, 저작권, 상표안내, 주의사항을 설명하고 있습니다.

고객 중심에서 제품의 완성을 추구하는 ㈜커미조아는 고객님들을 위해 언제나 최선과 정성을 다하는 정성을 다하는 자세로 지금 이 시간에도 보다 나은 제품 개발을 위해 성실과 열정을 바탕으로 제품에 바탕으로 제품에 꿈을 담고 있습니다.

저희 ㈜ 커미조아 제품을 선택하여 주신 여러분들께 다시 한번 감사의 말씀을 드리며, 본 장에서 설명드리는 내용은 C-NET 라이브러리 운용과는 별개의 내용이오나, 제품의 보증과 규정 그리고 저작권에 대한 내용에 대해서 설명이므로 취급상 안내를 위해 참조해 주십시요..

1 C-NET사전 안내 사항

1.1 Overview

1.1.1 제품 보증 안내

저희 ㈜커미조아는 고객 여러분들께 가장 안정된 소프트웨어와 하드웨어를 공급함으로써, 고객 여러분들을 만족시켜드리는 것을 최우선의 목표로 하고 있습니다. 만약 구입하신 제품에 외관상의 하자, 동작이상 또는 불량이 발견되는 경우에는 언제든지 저희 ㈜커미조아를 통해 문의해주시기를 바라며, 가까운 대리점 혹은 총판점을 통해 구입하신 경우에는 해당 구입처를 통해 문의하시면, 더욱 빠른 기술 지원을 받으실 수 있습니다.

1.1.2 제품 보증 규정

구입하신 당사의 제품은 소비자의 과실 이외의 자체 결함 및 동작이상에 대해 2 년간 그 전체 혹은 일부에 대해서 보증하고 있습니다. 당사의 제품에 대한 자세한 제품 보증 규정은 별도로 관리되는 각 제품의 '제품 보증 규정' 에 의거하며, 자세한 보증 규정을 알기 원하시는 경우 ㈜ 커미조아 혹은 총판점 및 대리점 등 해당 구입처를 통해 문의해 주시기 바랍니다.

1.1.3 저작권

1.1.4 상표안내

CNETSDK 에서는 인용된 각 회사(Coporation) 의 제품 상표 및 저작권을 사전에 명시하며, 해당 사항은 대한민국 저작권법에 보호됩니다. 본 매뉴얼에서는 각 회사의 저작권에 대한 모든 권리나 이에 관계된 내용을 보호합니다.

Windows 는 Microsoft Corp. 의 등록상표입니다.

Visual C++ 는 Microsoft Corp. 의 등록상표입니다.

Visual Basic 은 Microsoft Corp. 의 등록상표입니다.

C++ Builder 는 Borland Software Corp. 의 등록상표입니다.

Delphi 는 Borland Software Corp. 의 등록상표입니다.

이외의 상표는 각 회사의 등록상표입니다.

1.1.5 주의사항

㈜ 커미조아의 제품군에는 제품의 특성에 따라서 하드웨어 및 소프트웨어 기술 지원이 필요한 경우가 있습니다. 필요하신 경우 본사 혹은 총판 및 대리점을 통해 제품 구입 이전에 점검 또는 요청해주시기 바랍니다.

㈜ 커미조아의 소프트웨어 및 하드웨어 제품군은 제품 성능 향상을 위해 예고 없이 변경될 수 있습니다. 고객님께서는 본 매뉴얼을 읽기전에 하드웨어 및 소프트웨어의 최신 변경사항에 대한 정보를 ㈜ 커미조아를 통해 요청하실 수 있습니다.

본 매뉴얼은 ㈜커미조아 CNETSDK 대한 정보와 CNETSDK 를 통한 C-NET 제어 사항에 대한 기반 설명을 포함하고 있습니다.

최신 내용 및 기술되지 않은 사항 혹은 누락된 사항은 ㈜ 커미조아 제품군 구입시에 고객 등록을 해주신 고객님의 정보를 통해 안내해 드릴 예정이며, 기술 지원 요청시에 보다 자세하고 정확한 방법과 내용을 통해 도움 드릴 것을 약속드립니다.

본 매뉴얼에 시작하기 앞서, 본 장(Chapter) 에서는 본 매뉴얼을 보다 정확하게 빠르게 이해하실 수 있도록 CNETSDK 의 전체 구성과 형식에 대해서 안내해 드리겠습니다.

1.1.6 매뉴얼 용어 안내

본 매뉴얼에서는 필요에 따라 매뉴얼에 기술된 내용을 설명하기 위해 함축된 의미의 용어나 현장 용어를 사용할 수도 있습니다. 최대한 보충 설명이 필요한 내용에 대해서는 해당 단원(Chapter)에서 설명하도록 하겠습니다.

전체 매뉴얼에서 다루는 내용 중 다음 용어는 최신 커미조아 라이브러리를 의미하며, 다음과 같은 명칭은 모두 같은 의미를 갖습니다.

매뉴얼 전체 범위에서 사용되는 범용적인 용어의 의미는 다음 표 1-1 매뉴얼 용어 정리 같이 미리 안내하여 드립니다.

명칭	의미
디지털 입출력 보드 (Digital Input Output Board)	㈜ 커미조아의 범용 디지털 입출력 제어를 위한 제품으로서 SD4xx Series 제품을 일부 혹은 종합적으로 일컫는 말입니다.
마스터 디바이스 (Master Device)	㈜ 커미조아의 디바이스넷 계열의 제품인 LX412a/b 보드를 일컫는 말입니다.
링(RING)	마스터 디바이스에 장착되어 있는 RJ-45 단자를 통해서 형성 되는 네트워크를 뜻합니다.
슬레이브 모듈 (Slave Module)	㈜ 커미조아의 cnD8-8, cnD16-16, cnD32-00, cnD00-32 제품군으로서 마스터 디바이스의 하위에서 링형의 네트워크를 구성하는 장치들을 일컫는 말입니다.
로컬 디바이스 (Local Device)	마스터 디바이스 보드에 달려 있는 범용 DIO 를 뜻합니다.

표 1-1 매뉴얼 용어 정리

1.1.7 매뉴얼 아이콘의 설명

매뉴얼에서 안내되는 내용을 보다 신속하고 정확하게 알 수 있도록 하며, 그 의미가 바르게 전달 되고 이해를 돕기 위해 원하는 의미에서 아래와 같은 매뉴얼 아이콘을 사용하고 있습니다.

	이해하기 어려운 용어의 해설 및 사전에 설명 없이 새로 나온 용어를 설명합니다.
	고객님의 편의와 기능의 자세한 내용에 대해 부가적으로 안내되는 사항으로서, 기억해 두면 편리한 기능을 나타냅니다.

	이 동작이나 실행 함수(Function) 에 있어서 그 동작의 주의를 요망하는 내용을 나타냅니다.
	이 동작이나 실행 함수(Function) 에 있어서 그 동작의 이상이나 문제가 발생할 수 있습니다.

표 1-2 매뉴얼 아이콘의 설명

1.2 Features

㈜ 커미조아의 CNETSDK 가 가지고 있는 장점은 다음과 같습니다.

1.2.1 독립성

CNETSDK 는 Microsoft 社의 표준라이브러리인 DLL(Dynamic Link Library) 형태의 독립된 라이브러리 인터페이스를 제공하며, 라이브러리 자체의 독립된 장치 관리의 편의성을 제공하고 있으며, DLL 형태의 라이브러리 장점을 통해 유지 보수와 귀사의 제품 구현에 보다 간편하게 하고 신뢰성 있는 독립형 동적 연결 라이브러리를 제공합니다

1.2.2 호환성

우수한 소프트웨어 개발 도구를 이용하여 전통적인 개발 방법보다 더 적은 시간과 비용으로 더 좋은 품질의 소프트웨어를 개발하는 방법을 이야기하는 최신 RAD(Rapid Application Development) 를 지향하고 있으며, 이에 맞는 최신 소프트웨어 개발 환경을 지원하고 있습니다.

고객님께서는 언제나 CNETSDK 를 통하여 귀사의 제품에 보다 신속하고 정확한 시스템을 구현하실 수 있습니다.

1.2.3 편의성

인터페이스 함수 명명 규칙의 통일화와 의사 코드 주제어를 매뉴얼과 라이브러리 인자(Parameter) 에 부각시켜, 보다 빠른 시간내에 숙련된 라이브러리 사용자로 만들어드립니다. 특히, CNETSDK 의 모든 함수 명에는 의미적 명명 규칙을 내포하였으며, 이것은 분명 실무 개발 환경에서 많은 부분 이점으로 작용할 것입니다.

1.2.4 신뢰성

㈜ 커미조아의 라이브러리 제품군은 오랜 시간 산업현장에서의 현장 경험을 바탕으로 형성된 신뢰성있는 제품군이며, 각 인터페이스에 걸쳐 사용하기 쉽도록 안내해 놓았습니다.

또한, 응용프로그램에 기반하지 않는 자체 디버그 기능을 바탕으로 응용프로그램에 의한 오류를 최단시간내에 해결할 수 있도록 합니다. 디버그 모드의 지원과 향상된 디버그 정보를 바탕으로 오류발생에 대한 원인을 신속히 분석하실 수 있도록 도움을 드리며, 저희 ㈜ 커미조아는 고객님께 항상 정직과 신뢰를 드릴 수 있도록 최선을 다할 것입니다.

1.2.5 풍부한 예제와 신속한 기술 지원

지금 이시간에도 ㈜ 커미조아는 타사에 비해 보다 많은 개발 선상의 활용가능한 라이브러리 예제를 지원해드리려고 노력하고 있으며, 예제간 중복 코드와 라이브러리 구성 이외의 부분을 최소화한 예제로 빠른 시간내에 예제의 코드를 바로 적용시켜드릴 수 있도록 노력하고 있습니다. 특히 Smart Update 기능을 통해 라이브러리의 최신 버전을 인터넷을 통해 다운로드 받아 바로 업그레이드할 수 있는 기능을 제공하고 있습니다.

저의 ㈜ 커미조아는 최신 .NET Framework 상의 C Sharp(C#) 부터, Visual Basic 및 현존하는 RAD(Rapid Application Development) 환경을 지원하는 CNETSDK 로 이제 보다 향상된 고객 지원과 기술 지원으로 고객여러분들의 성원에 이바지하겠습니다.

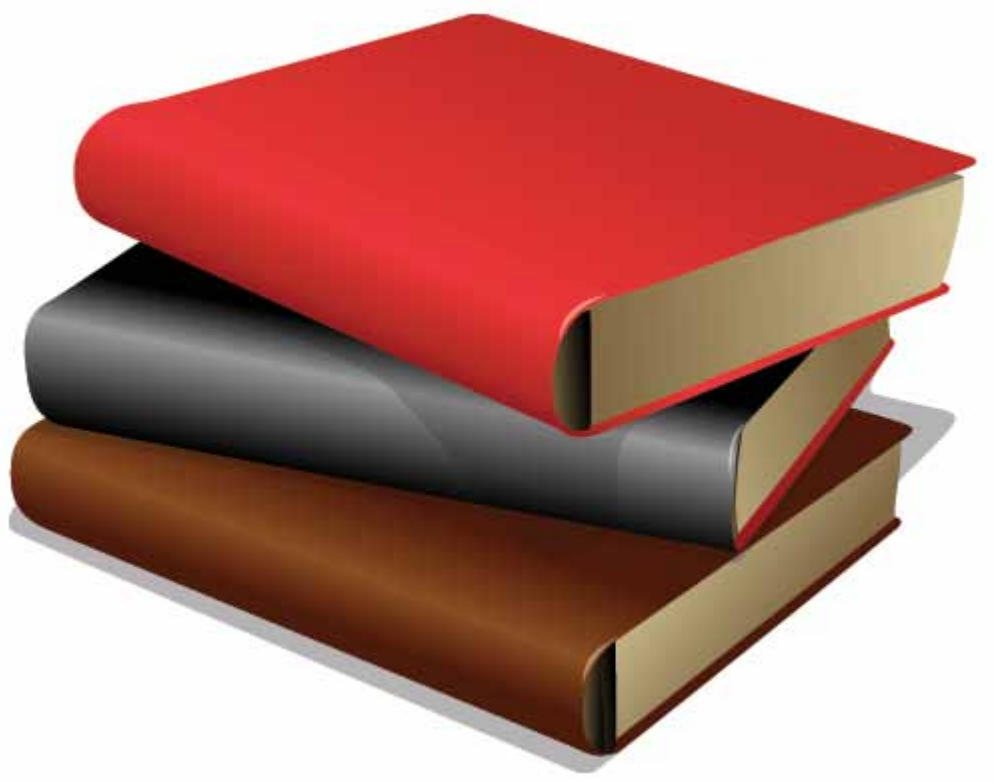
Chapter

2

About COMIZOA C-NET

안정성과 빠른 통신 속도는 필드버스방식의 DIO 솔루션이 갖추어야 하는 가장 기본적인 요소입니다.커미조아의 C-NET 은 이 두가지 요소를 모두 만족시키는 최고의 솔루션입니다.

가장 안정적이고, 빠른 DIO 솔루션으로 통하는 C-NET 은 하드웨어적인 성능을 최대한 활용할 수 있는 활용할 수 있는 최적의 소프트웨어를 제공합니다. CNETSDK 와 함께 최적의 속도와 개발의 편리성도 개발의 편리성도 함께 경험해 보십시오.



2 About COMIZOA C-NET

C-NET 은 갈수록 대형화되고 복잡해지는 산업환경에서 고객님들의 편리한 이용을 위해 제작된 원격 I/O 장치 제어 솔루션입니다.

C-NET 은 하나의 마스터 디바이스에서 여러 개의 슬레이브 모듈을 제어하여 다수의 접점(최대 2048 개)을 쉽게 제어할 수 있으며, 배선의 편리함과 원가절감의 효과도 누릴 수 있습니다.

2.1 Hardware Features

COMIZOA C-NET 을 사용하면 배선을 획기적으로 줄일 수 있으며, 모든 슬레이브 모듈에 통신회로가 장착되어 있어 원격 I/O 의 장점인 분산장착에 최적화 되어있습니다.

㈜커미조아의 C-NET 통신기법은 20Mbps 의 빠른 통신속도를 지원하여 고속으로 감시하거나 제어해야 할 경우 탁월한 성능을 보여줍니다.

1 개의 Ring 에 64 개의 슬레이브 모듈을 장착할 수 있어 접점의 확보가 용이하며, LX412a 의 경우 1 개의 마스터 디바이스에서 최대 2 개의 Ring 을 지원합니다.

모든 입,출력 접점에 안전회로가 장착되어 있어 사용자의 실수에 의한 피해를 최소화 시킬 수 있습니다.

여러 종류의 슬레이브 모듈로 사용자가 원하는 곳에 원하는 만큼의 접점을 사용할 수 있도록 지원합니다.

2.2 Software Features

제품 개별적인 하드웨어 인터페이스에 대한 사항들을 직접 인지하고 있지 않아도 CNETSDK 를 통해 신속하고 안정적인 응용프로그램 개발이 가능합니다.

빠른 속도의 장치 접근성을 소프트웨어 적으로도 보장합니다. 소프트웨어적인 고속 접근 기술을 사용하여 최대한 빠른 속도로 디바이스를 찾고 제어 합니다.

채널을 통합적으로 관리함으로 장치들의 순서나 위치에 관계없어 선형적인 접근이 가능 합니다.

사용자의 디버깅이 편리하도록 별도의 디버깅 기능을 제공합니다.

2.3 용어설명

명칭	의미
Open-collector 출력 방식	Transistor 의 collector 단을 floating 시켜놓은 방식. Floating(High impedance)과 Closed 두가지 state 를 가지게 된다. NPN 출력방식이라고도 한다.
Open-drain 출력방식	Open-collector 과 비슷한 개념으로 MOSFET 의 Drain 단을 floating 시켜놓은 방식.
Current sink	외부 전위를 받아들이는 형태로 구동하는 방식.
Current source	외부에 전위를 제공하는 형태로 구동하는 방식.
Surge 전압	단시간 내에 심하게 변화하는 과도적인 전압을 말한다.
Low-pass filter	낮은 주파수 성분은 통과시키고 높은 주파수 성분은 제거하는 필터. 단시간에 변화하는 신호를 제거하는 목적으로 사용된다.

Chapter

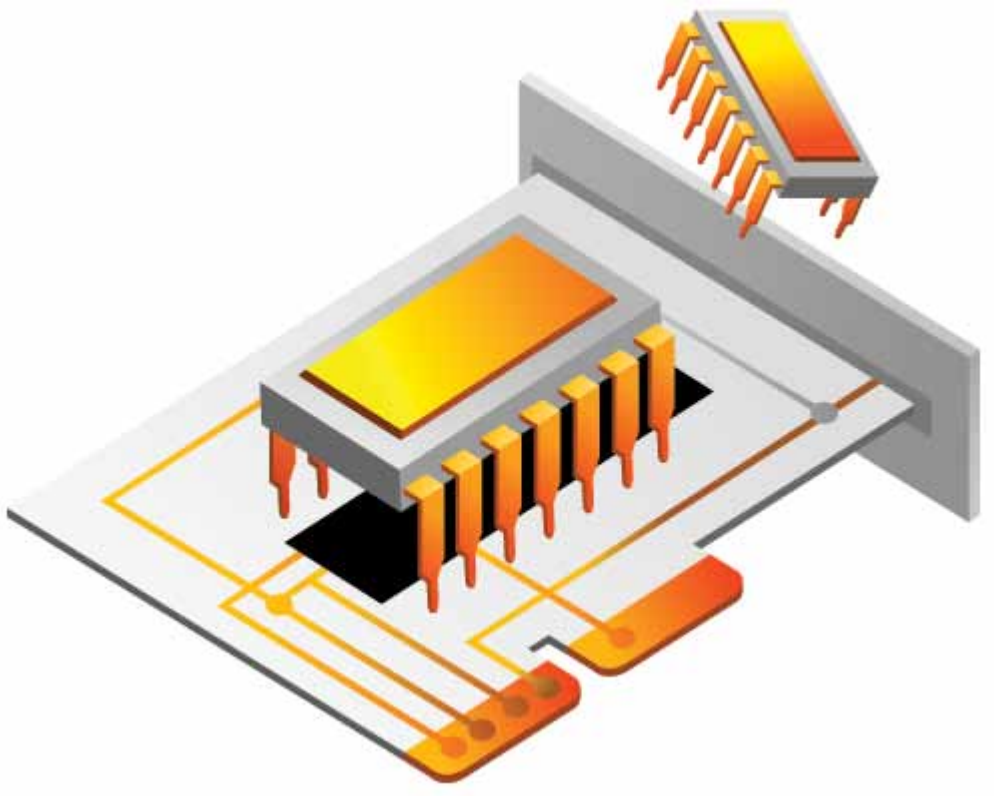
3

Hardware reference

최고의 성능을 자랑하는 솔루션인 C-NET 의 기본은 안전한 하드웨어에 있습니다. 안정적이며 편리한 사용이 가능한 하드웨어를 통해서 C-NET 의 강력함과 편리함,을 느껴보십시오.

본 장에서는 기본적인 하드웨어지식과 사용시에 도움이 되는 내용들을 담고 있습니다. 이 장을 통해서 장을 통해서 다른 하드웨어들과는 차별화되는 커미조아의 C-NET 의 특징을 확신 하실 수 있습니다.

수 있습니다.



3 HARDWARE REFERENCE

3.1 Master Device – COMI-LX412a/b

COMI-LX412a/b 는 C-NET 을 제어하는 마스터 디바이스입니다.

COMI-LX412a 와 COMI-LX412b 는 지원하는 Ring 의 개수로 구분하며 그 외의 기능상의 차이점은 없습니다.

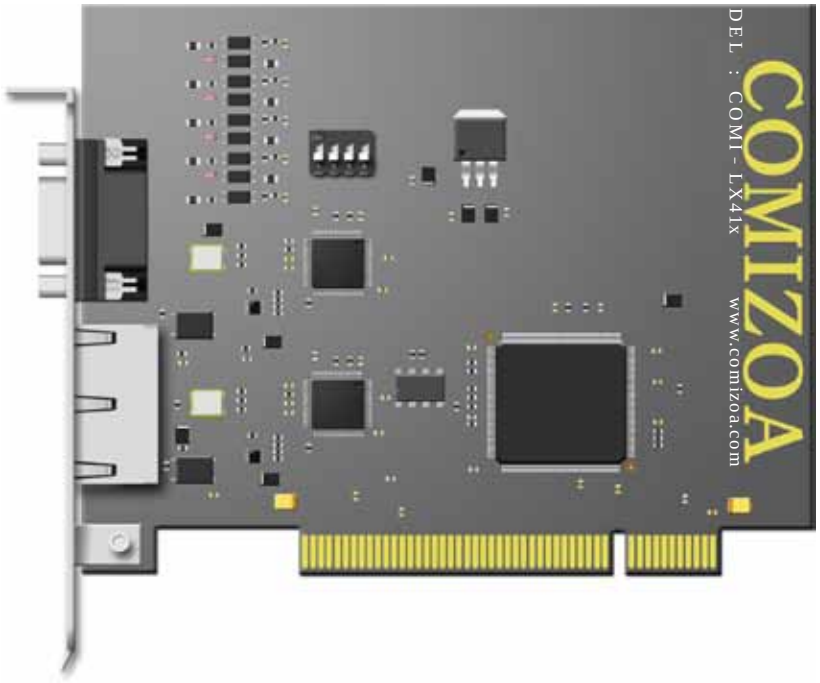


그림 3-1 Layout of the COMI-LX412a

3.1.1 Product specification

Features

▷ C-NET Master controller

▷ Standard PCI bus type

▷ Max. 128(64) C-NET digital slave modules support

▷ Baud-rate up to 20Mbps transfer rate

▷ Easy installation with RJ45 phone jack

▷ 4 Isolated digital input

▷ 4 Isolated digital output

Specifications

▷ C-NET Master

항 목	사 양
Number of rings	2(COMI-LX412a)
	1(COMI-LX412b)
Transmission Speed	2.5, 5, 10, 20 Mbps with automatic data flow control
Serial Interface	Half duplex RS-485 with transformer isolation
Cable Type	CATS UTP/STP Ethernet cable
Communication Distance	Max. 100m (20Mbps / 32 Slavemodules)
Communication Slave Module capacity	128 slaves(COMI-LX412a)
	64 slaves(COMI-LX412b)
Response time	20 usec
Isolation Voltage	2500Vrms

▷ Digital Input

항목	사양
Number of channels	Isolated 4 channels
Input type	Current sinking, sourcing type
Input Range	DC 0~24V
Input Level Tolerant	Low (Min. 0V ~ Max.1.5V)
	High(Min. 5V ~ Max.24V)
Isolation	1000V Channel to Channel
Input Impedance	3.6 KOhm

▷ Digital Output

항목	사양
Number of channels	Isolated 4 channels
Output type	Darington-TR Open-collector
Rds(On state resistance)	Max. 500mOhm
Sink Current	Max. 500mA / channel

▷ Common

항목	사양
Board size	132 x 100 (mm)
Serge Protection	10KV
Over Temperature Shutdown	165˚C

3.1.2 Functional block diagram

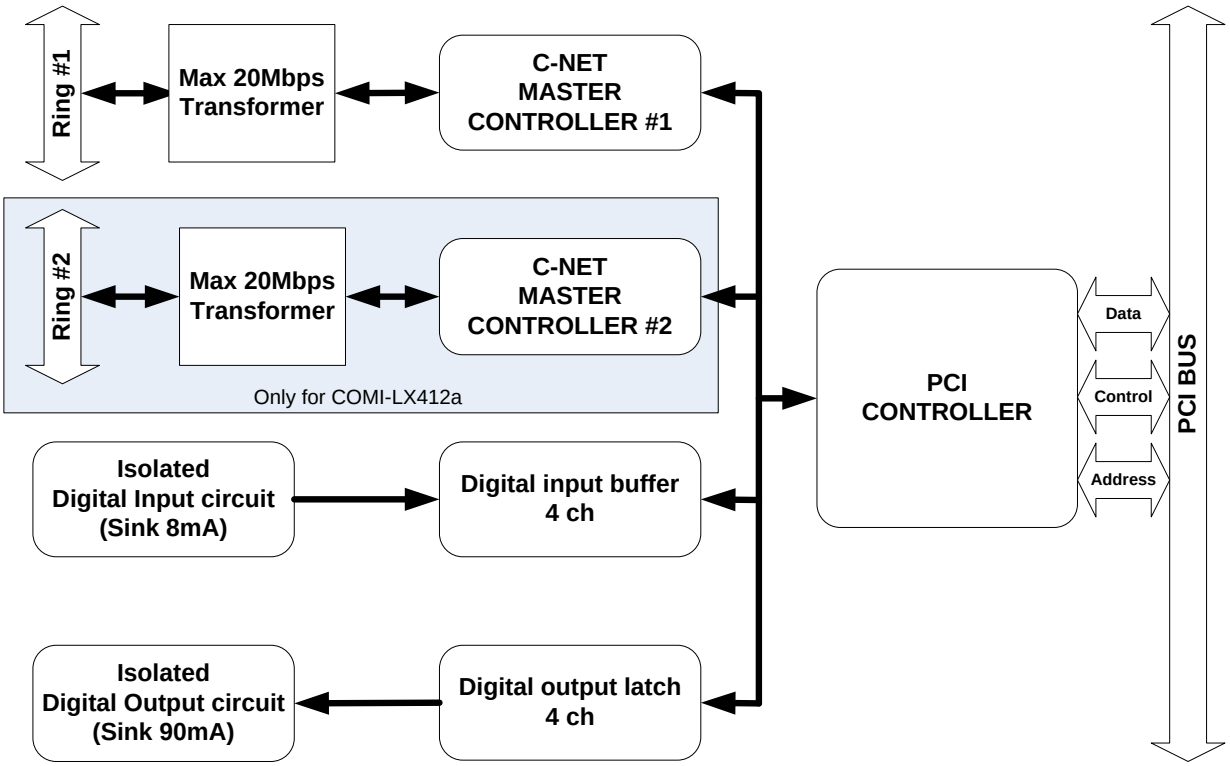


그림 3-2 Functional block diagram of the LX412a

3.1.3 디지털 입력 회로

COMI-LX412a/b 의 디지털 입력 회로는 모두 “Bidirectional Photo-coupler”로 구성되어 있습니다.

모든 입력회로는 외부 GND 와 절연되어 있고, 각 입력 회로마다 Low-pass filter 를 내장하여 Noise 나 Surge 전압 등의 외부 요인에 의한 시스템의 오동작이나 손상을 방지하며, Hardware 접점 사용시 발생할 수 있는 Chattering 을 제거하게 됩니다.

모든 입력은 Current sink 방식과 source 방식을 모두 사용할 수 있게 설계되어 있습니다.

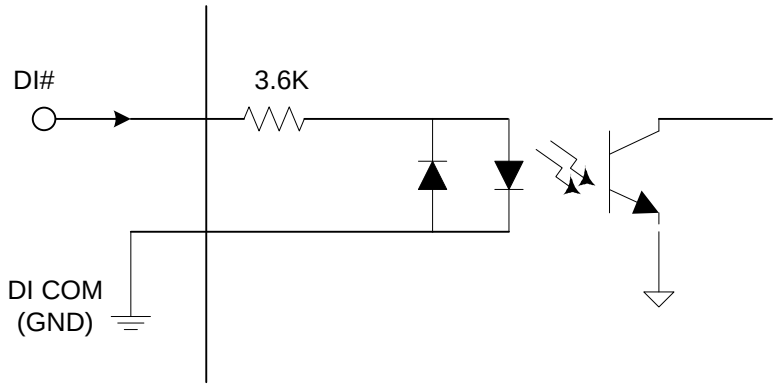


그림 3-3 COMI-LX412a/b 의 디지털 입력 회로 및 신호 연결 방법(Current Source mode)

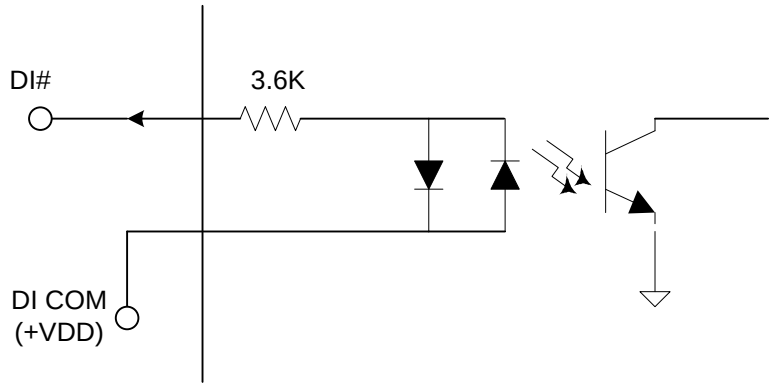


그림 3-4 COMI-LX412a/b 의 디지털 입력 회로 및 신호 연결 방법(Current Sink mode)

Current Source 방식 사용 시에는 DI COM 단자에 GND 를 입력해야 합니다. 이때 Digital Input 단자의 전압이 GND+5V 이상으로 올라가게 되면 Digital Input 단자에서 GND 로 전류가 흐르면서 Photo-coupler 가 “ON”되게 됩니다.

Current Sink 방식 사용 시 DI COM 단자에는 +VDD 를 입력해야 합니다. 이때 Digital Input 단자의 전압이 VDD–5V 이하로 떨어지게 되면 VDD 에서 Digital Input 단자로 전류가 흐르면서 Photo-coupler 가 “ON”되게 됩니다.

※ DI COM 단자와 Digital Input 단자간의 전압차가 5V 보다 작아도 Input 이 “ON”될 수 있습니다. 하지만 이 때는 Tolerant 전압이므로 확실한 동작을 보장할 수 없습니다.

3.1.4 디지털 출력 회로

COMI-LX412a/b 의 디지털 출력 회로는 “Photo-coupler”와 “Daringtion-transistor”로 구성되어 있습니다.

또한 그림과 같이 외부와 최대 2500 Vrms 까지 절연하는 회로가 내장되어 있어, 외부에서 유입되는 순간적인 고전압 또는 노이즈의 영향으로 시스템이 파손되는 것을 방지할 수 있습니다.

COMI-LX412a/b 의 출력회로는 Open-collector 출력 방식을 사용합니다. Current sink 방식만 지원하므로, Current source 방식만 입력받을 수 있는 회로와 연결할 경우에는 Relay 또는 기타 부가회로가 외부에 장착되어야 합니다.

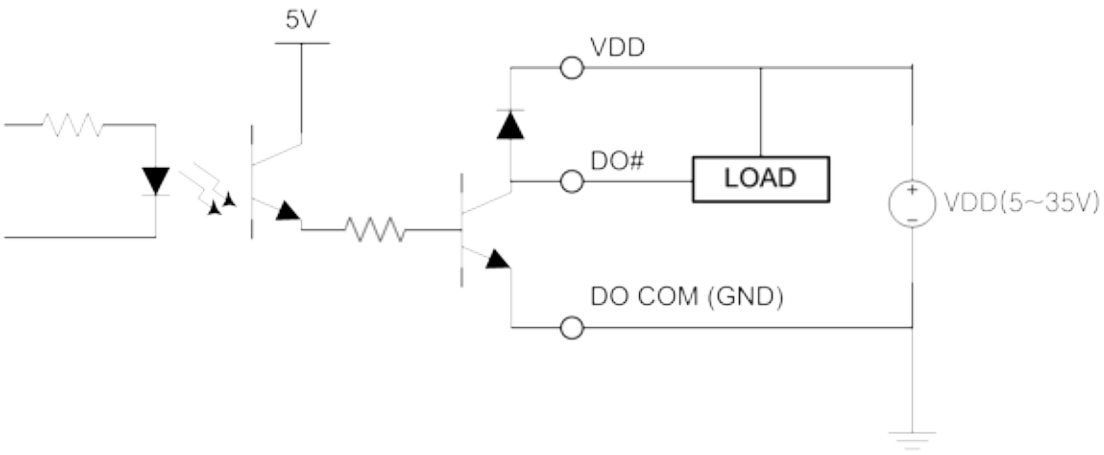


그림 3-5 LX412a/b 의 디지털 출력 회로 및 신호 연결 방법

결선시 DO COM 은 GND 에 연결되어 있어야 합니다. 디지털 출력이 “ON”상태가 되면 싱크 전류(Sink current)가 달링턴 TR 을 통해 전도되고, 디지털 출력이 “OFF” 상태가 되면 달링턴 TR 를 통해 전류가 전도되지 않습니다.

주의	릴레이, 코일 또는 모터등과 같이 인덕턴스(Inductance) 성질을 가지는 부하(Load)를 구동할 경우에는 외부 소스 전원을 VDD 핀에 연결해야 합니다. VDD 핀을 외부 소스전원에 연결하면, 그림과 같이 “Fly-wheel Diode”가 적절히 동작할 수 있는 회로가 구성됩니다. 역기전압이 발생하면 그림의“Fly-wheel Diode”가 동작되어 역기전압에 의한 달링턴 TR 의 손상을 방지할 수 있습니다.

3.1.5 Pin configuration

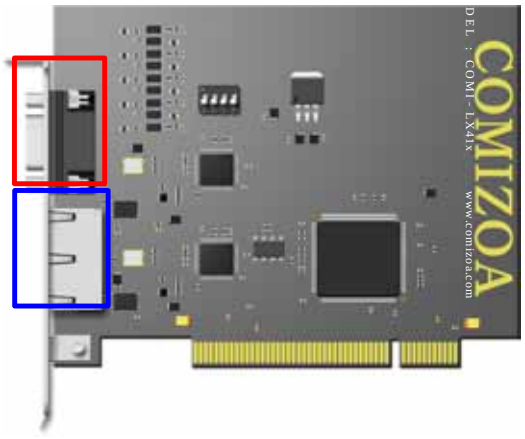
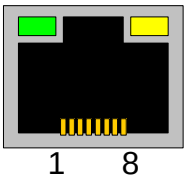


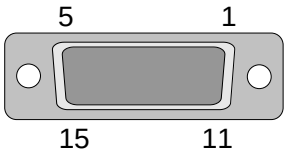
그림 3-6 Connector location

C-NET Ring Connector configuration



Pin	Label	Description
1	GND	GND
2	GND	GND
3	RS485_D+	High speed RJ485 Protocol +
4	GND	GND
5	GND	GND
6	RS485_D-	High speed RJ485 Protocol -
7	GND	GND
8	GND	GND

Digital I/O port configuration



Pin	Description	Pin	Description
1	Digital output 0	9	NC
2	Digital output 1	10	VDD
3	Digital output 2	11	Digital input 0
4	Digital output 3	12	Digital input 1
5	DO COM	13	Digital input 2
6	NC	14	Digital input 3
7	NC	15	DI COM
8	NC		

3.2 Slave modules

C-NET 의 Ring 을 구성하는 슬레이브 모듈입니다.

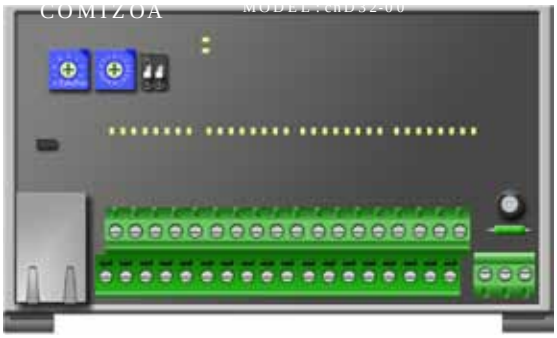


그림 3-7 Layout of the cnD32-00

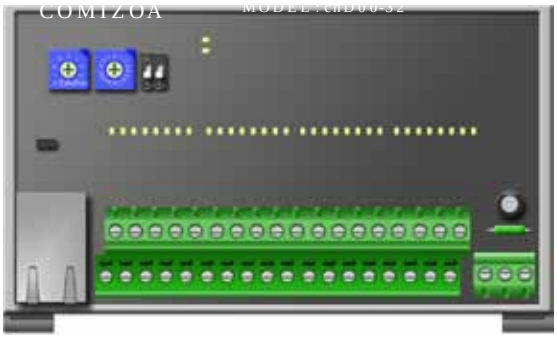


그림 3-8 Layout of the cnD00-32

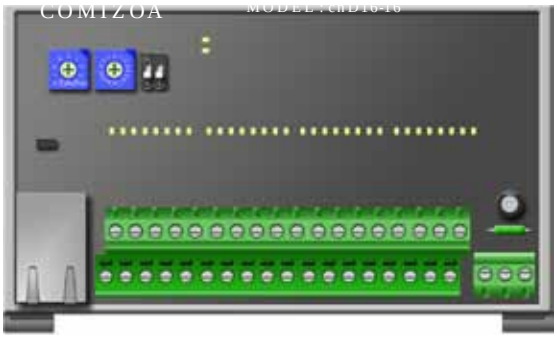


그림 3-9 Layout of the cnD16-16

그림 3-10 Layout of the cnD8-8

- Terminal board type cnD series slave modules -

그림 3-11 Layout of the cnDV8-8

그림 3-12 Layout of the cnDV16-00

그림 3-13 Layout of the cnDV00-16

그림 3-14 Layout of the cnDV32-00

그림 3-15 Layout of the cnDV00-32

- Vertical case type cnDV series slave modules -

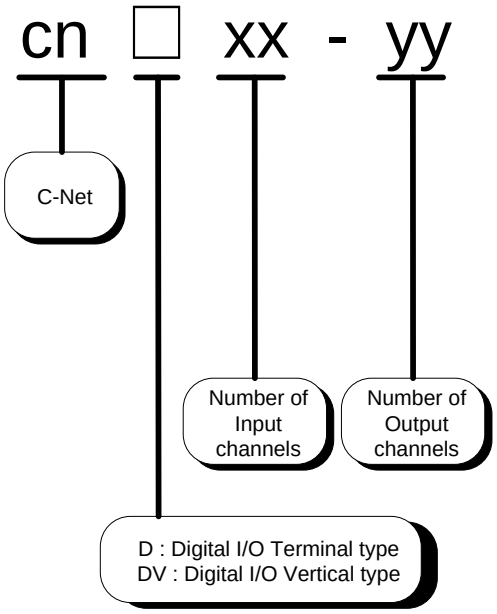


그림 3-16 C-NET Slave module 모델코드 표기법

3.2.1 Product specification

Features

▷ C-NET Slave module

▷ Easy installation with RJ45 phone jack

▷ LED diagnostic indicator

▷ Common industrial DIN Rail mount type

Specifications

▷ C-NET Slave

항 목	사 양
Number of Digital I/O channels	Isolated Input 32 channels (cnD32-00)
	Isolated Output 32 channels (cnD00-32)
	Isolated Input 16 channels, Isolated Output 16 channels(cnD16-16)
	Isolated Input 8 channels, Isolated Output 8 channels (cnD8-8)
	Isolated Input 32 channels (cnDV32-00)
	Isolated Output 32 channels (cnDV00-32)
	Isolated Input 16 channels (cnDV16-00)
	Isolated Output 16 channels (cnDV00-16)
	Isolated Input 8 channels, Isolated Output 8 channels (cnDV8-8)
Transmission Speed	2.5, 5, 10, 20 Mbps with automatic data flow control
Serial Interface	Half duplex RS-485 with transformer isolation
Cable Type	CATS UTP/STP Ethernet cable
Response time	20 usec
Isolation Voltage	2500Vrms

▷ Digital Input

항목	사 양
Input type	Current sinking, sourcing type
Input Impedance	3.6KOhm
Input Range	DC 0~24V
Input Level Tolerant	Low (Min. 0V ~ Max.1.5V)
	High(Min. 5V ~ Max.24V)
Isolation	1000V Channel to Channel
Input Impedance	3.6 KOhm

▷ Digital Output

항목	사 양
Output type	MOSFET with common ground
Output Device	IPS042G (Common ground)
Output Voltage	Min. 5V ~ Max. 35V
Sink Current	Max. 500mA / channel
Rds(On state resistance)	Max. 500mOhm
Throughput	10Khz (0.1ms)
Shutdown	Over current(1.1A Min ~ 2A Max)

▷ Common

항목	사 양
Board size	132 x 75 (cnD32-00, cnD00-32, cnD16-16)
	95 x 75 (cnD8-8)
Power Input	Isolated DC 24V
Serge Protection	10KV
Over Temperature Shutdown	165˚C

3.2.2 Functional block diagram

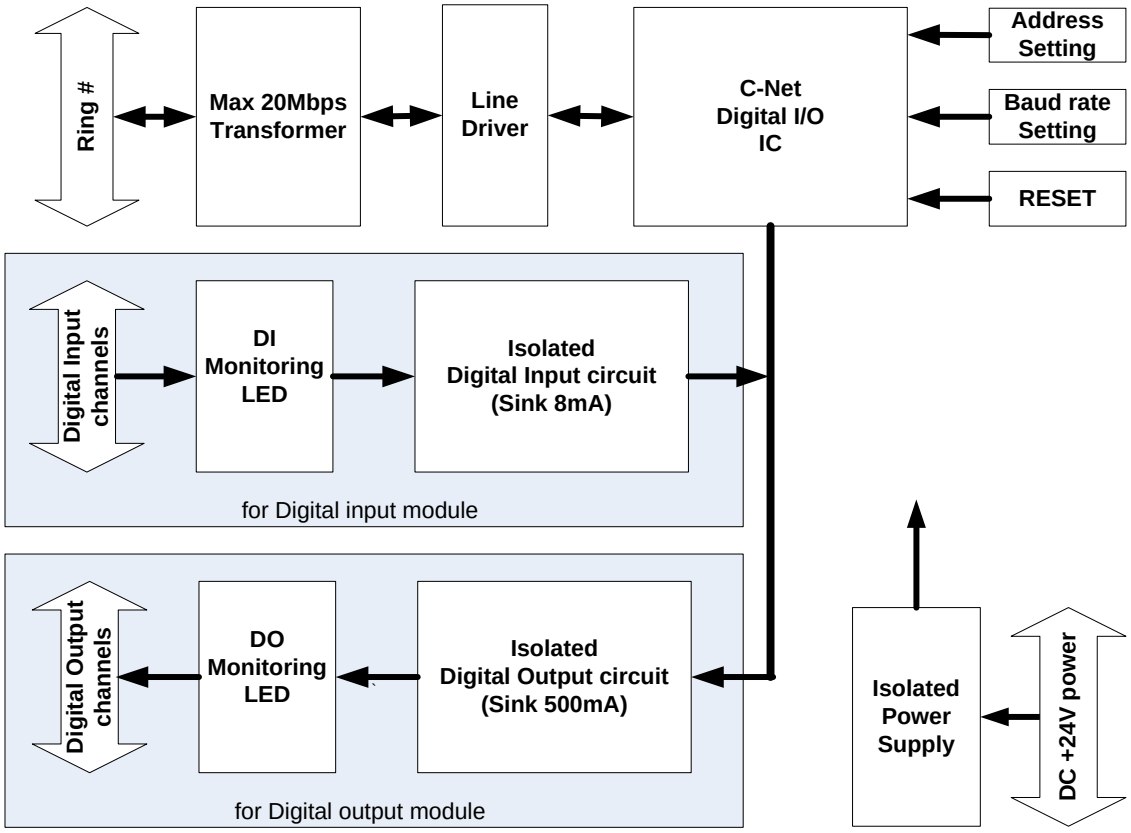


그림 3-17 Functional block diagram of slave modules

3.2.3 디지털 입력 회로

COMI-LX412a/b 의 디지털 입력 회로는 모두 “Bidirectional Photo-coupler”로 구성되어 있습니다.

모든 입력회로는 외부 GND 와 절연되어 있고, 각 입력 회로마다 Low-pass filter 를 내장하여 Noise 나 Surge 전압 등의 외부 요인에 의한 시스템의 오동작이나 손상을 방지하며, Hardware 접점 사용시 발생할 수 있는 Chattering 을 제거하게 됩니다.

모든 입력은 Current sink 방식과 source 방식을 모두 사용할 수 있게 설계되어 있습니다.

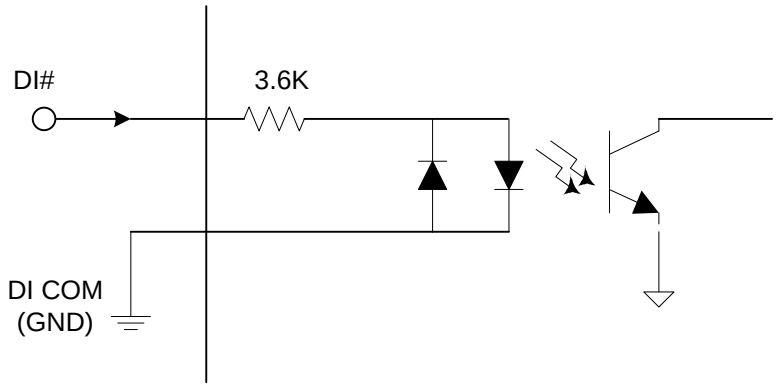


그림 3-18 C-NET slave module 의 디지털 입력 회로 및 신호 연결 방법(Current Source mode)

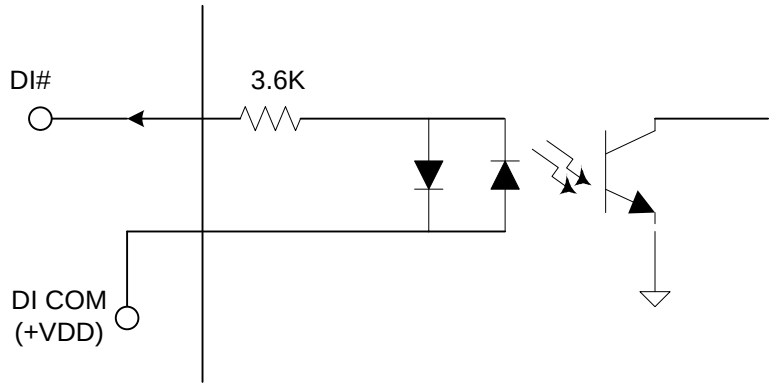


그림 3-19 C-NET slave module 의 디지털 입력 회로 및 신호 연결 방법(Current Sink mode)

Current Source 방식 사용 시에는 DI COM 단자에 GND 를 입력해야 합니다. 이때 Digital Input 단자의 전압이 GND+5V 이상으로 올라가게 되면 Digital Input 단자에서 GND 로 전류가 흐르면서 Photo-coupler 가 “ON”되게 됩니다.

Current Sink 방식 사용 시 DI COM 단자에는 +VDD 를 입력해야 합니다. 이때 Digital Input 단자의 전압이 VDD–5V 이하로 떨어지게 되면 VDD 에서 Digital Input 단자로 전류가 흐르면서 Photo-coupler 가 “ON”되게 됩니다.

※ DI COM 단자와 Digital Input 단자간의 전압차가 5V 보다 작아도 Input 이 “ON”될 수 있습니다. 하지만 이 때는 Tolerant 전압이므로 확실한 동작을 보장할 수 없습니다.

3.2.4 디지털 출력 회로

C-NET slave module 의 디지털 출력 회로는 “Photo-coupler”와 “MOSFET”로 구성되어 있습니다.

그림과 같이 외부와 최대 2500 Vrms 까지 절연하는 회로가 내장되어 있어, 외부에서 유입되는 순간적인 고전압 또는 노이즈의 영향으로 시스템이 파손되는 것을 방지할 수 있습니다.

또한 과전류 사용시 MOSFET 이 자동으로 출력을 중단 할 수 있도록 설계되어있어, 외부 배선 단락등의 원인으로 시스템이 파손되는 것을 방지할 수 있습니다.

C-NET 슬레이브 모듈의 출력회로는 Open-drain 출력 방식을 사용합니다. Current sink 방식만 지원하므로, Current source 방식만 입력받을 수 있는 회로와 연결할 경우에는 릴레이 또는 기타 부가회로가 외부에 장착되어야 합니다.

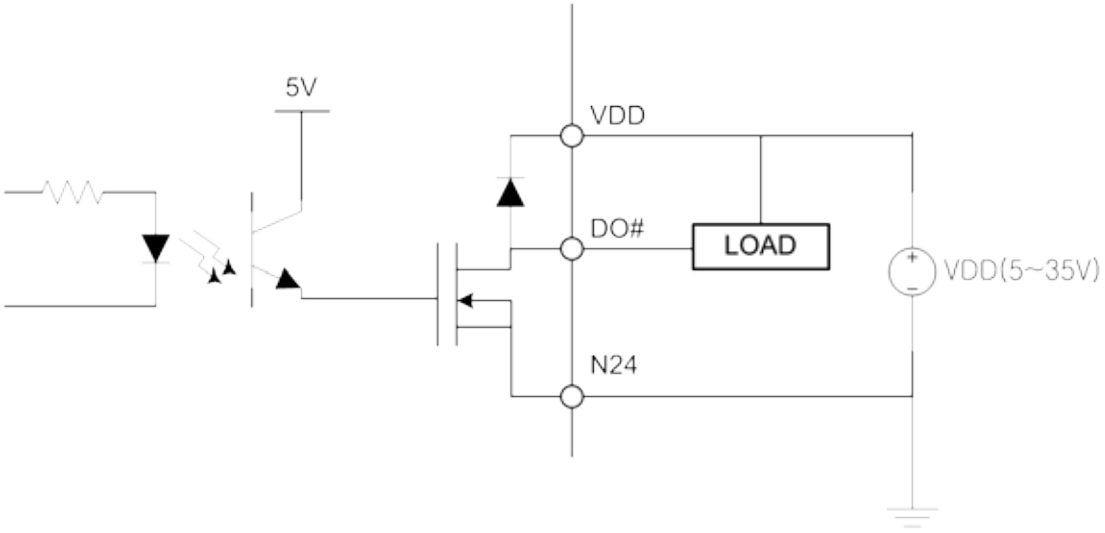


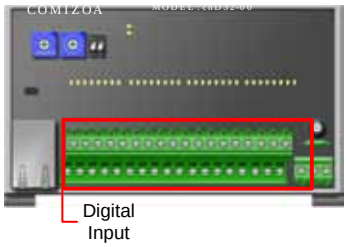
그림 3-20 C-NET 슬레이브 모듈의 디지털 출력 회로 및 신호 연결 방법

디지털 출력이 “ON”상태가 되면 싱크 전류(Sink current)가 MOSFET 을 통해 전도되고, 디지털 출력이 “OFF” 상태가 되면 MOSFET 를 통해 전류가 전도되지 않습니다.

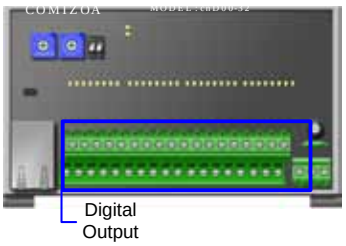
주의	릴레이, 코일 또는 모터등과 같이 인덕턴스(Inductance) 성질을 가지는 부하(Load)를 구동할 경우에는 외부 소스 전원을 VDD 핀에 연결해야 합니다. VDD 핀을 외부 소스전원에 연결하면, 그림과 같이 “Fly-wheel Diode”가 적절히 동작할 수 있는 회로가 구성됩니다. 역기전압이 발생하면 그림의“Fly-wheel Diode”가 동작되어 역기전압에 의한 MOSFET 의 손상을 방지할 수 있습니다.

3.2.5 Pin configuration – D Type

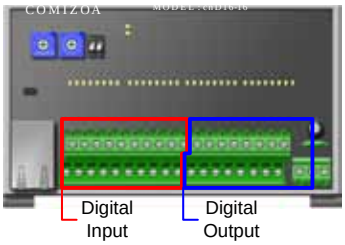
cnD32-00



cnD00-32



cnD16-16



cnD8-8

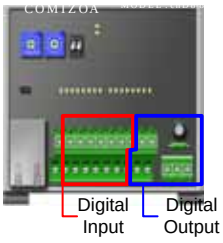


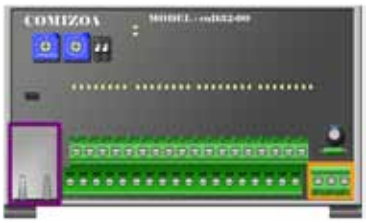
그림 3-21 Terminal location

Digital I/O terminal pin configuration

Digital Input terminal				Digital Output terminal			
Label	Description	Label	Description	Label	Description	Label	Description
0	Digital Input 0	16	Digital Input 16	0	Digital Output 0	16	Digital Output 16
1	Digital Input 1	17	Digital Input 17	1	Digital Output 1	17	Digital Output 17
2	Digital Input 2	18	Digital Input 18	2	Digital Output 2	18	Digital Output 18
3	Digital Input 3	19	Digital Input 19	3	Digital Output 3	19	Digital Output 19
4	Digital Input 4	20	Digital Input 20	4	Digital Output 4	20	Digital Output 20
5	Digital Input 5	21	Digital Input 21	5	Digital Output 5	21	Digital Output 21
6	Digital Input 6	22	Digital Input 22	6	Digital Output 6	22	Digital Output 22
7	Digital Input 7	23	Digital Input 23	7	Digital Output 7	23	Digital Output 23
COM	DI_COM	N24	GND	VDD	Tie to P24 for flywheel	N24	GND
8	Digital Input 8	24	Digital Input 24	8	Digital Output 8	24	Digital Output 24
9	Digital Input 9	25	Digital Input 25	9	Digital Output 9	25	Digital Output 25
10	Digital Input 10	26	Digital Input 26	10	Digital Output 10	26	Digital Output 26
11	Digital Input 11	27	Digital Input 27	11	Digital Output 11	27	Digital Output 27
12	Digital Input 12	28	Digital Input 28	12	Digital Output 12	28	Digital Output 28
13	Digital Input 13	29	Digital Input 29	13	Digital Output 13	29	Digital Output 29
14	Digital Input 14	30	Digital Input 30	14	Digital Output 14	30	Digital Output 30
15	Digital Input 15	31	Digital Input 31	15	Digital Output 15	31	Digital Output 31
COM	DI_COM	N24	GND	VDD	Tie to P24 for flywheel	N24	GND

안내	※ Common 단자 (COM, VDD)는 채널에 따라 분리되어있지 않습니다.

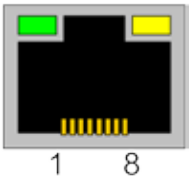
cnD 32channel module



cnD 16channel module

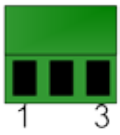
그림 3-22 Connector location

C-NET Ring Connector configuration



Pin	Label	Description
1	GND	GND
2	GND	GND
3	RS485_D+	High speed RJ485 Protocol +
4	GND	GND
5	GND	GND
6	RS485_D-	High speed RJ485 Protocol -
7	GND	GND
8	GND	GND

Power Connector configuration



Power connector

Pin	Label	Description
1	+24V	P24
2	GND	N24
3	F.G	Field Ground

3.2.6 Pin configuration – DV Type

cnDV rear side 16channel module

cnDV rear side 32channel module

그림 3-23 Terminal location

Digital I/O terminal pin configuration (cnDV16-00, cnDV00-16, cnDV08-08)

Common		Digital I/O				Common	
Pin No.	Description	Label	cnDV16-00	cnDV00-16	cnDV08-08	Label	Description
1	P24	18	Digital Input 0	Digital Output 0	Digital Input 0	34	N24
2	P24	19	Digital Input 1	Digital Output 1	Digital Input 1	35	N24
3	P24	20	Digital Input 2	Digital Output 2	Digital Input 2	36	N24
4	P24	21	Digital Input 3	Digital Output 3	Digital Input 3	37	N24
5	P24	22	Digital Input 4	Digital Output 4	Digital Input 4	38	N24
6	P24	23	Digital Input 5	Digital Output 5	Digital Input 5	39	N24
7	P24	24	Digital Input 6	Digital Output 6	Digital Input 6	40	N24
8	P24	25	Digital Input 7	Digital Output 7	Digital Input 7	41	N24
9	P24	26	Digital Input 8	Digital Output 8	Digital Output 0	42	N24
10	P24	27	Digital Input 9	Digital Output 9	Digital Output 1	43	N24
11	P24	28	Digital Input 10	Digital Output 10	Digital Output 2	44	N24
12	P24	29	Digital Input 11	Digital Output 11	Digital Output 3	45	N24
13	P24	30	Digital Input 12	Digital Output 12	Digital Output 4	46	N24
14	P24	31	Digital Input 13	Digital Output 13	Digital Output 5	47	N24
15	P24	32	Digital Input 14	Digital Output 14	Digital Output 6	48	N24
16	P24	33	Digital Input 15	Digital Output 15	Digital Output 7	49	N24
17	P24					50	N24

※ Common 단자 (COM, VDD)는 채널에 따라 분리되어있지 않습니다.

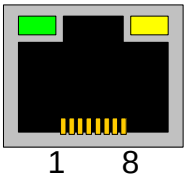
Digital I/O terminal pin configuration (cnDV32-00, cnDV00-32)

Digital I/O						Common	
Pin No.	cnDV32-00	cnDV00-32	Label	cnDV32-00	cnDV00-32	Label	Description
1	Digital Input 0	Digital Output 0	18	Digital Input 17	Digital Output 17	34	P24
2	Digital Input 1	Digital Output 1	19	Digital Input 18	Digital Output 18	35	P24
3	Digital Input 2	Digital Output 2	20	Digital Input 19	Digital Output 19	36	P24
4	Digital Input 3	Digital Output 3	21	Digital Input 20	Digital Output 20	37	P24
5	Digital Input 4	Digital Output 4	22	Digital Input 21	Digital Output 21	38	P24
6	Digital Input 5	Digital Output 5	23	Digital Input 22	Digital Output 22	39	P24
7	Digital Input 6	Digital Output 6	24	Digital Input 23	Digital Output 23	40	P24
8	Digital Input 7	Digital Output 7	25	Digital Input 24	Digital Output 24	41	P24
9	Digital Input 8	Digital Output 8	26	Digital Input 25	Digital Output 25	42	NC
10	Digital Input 9	Digital Output 9	27	Digital Input 26	Digital Output 26	43	N24
11	Digital Input 10	Digital Output 10	28	Digital Input 27	Digital Output 27	44	N24
12	Digital Input 11	Digital Output 11	29	Digital Input 28	Digital Output 28	45	N24
13	Digital Input 12	Digital Output 12	30	Digital Input 29	Digital Output 29	46	N24
14	Digital Input 13	Digital Output 13	31	Digital Input 30	Digital Output 30	47	N24
15	Digital Input 14	Digital Output 14	32	Digital Input 31	Digital Output 31	48	N24
16	Digital Input 15	Digital Output 15	33	NC	NC	49	N24
17	Digital Input 16	Digital Output 16				50	N24

※ Common 단자 (COM, VDD)는 채널에 따라 분리되어있지 않습니다.

그림 3-24 Connector location

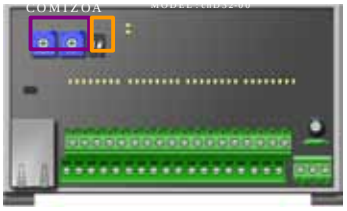
C-NET Ring Connector configuration



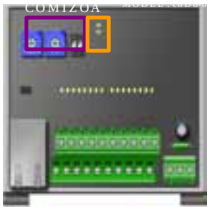
Pin	Label	Description
1	GND	GND
2	GND	GND
3	RS485_D+	High speed RJ485 Protocol +
4	GND	GND
5	GND	GND
6	RS485_D-	High speed RJ485 Protocol -
7	GND	GND
8	GND	GND

3.2.7 Slave module setting

cnD 32channel module



cnD 16channel module



cnDV 16channel module front side

cnDV 32channel module front side

그림 3-25 Address setting, Baudrate setting switch location

Address setting

	MSB	LSB	Address
 MSB  LSB	10 진 BDC	16 진 HEX	MSB x 16 + LSB

Baudrate setting

	DIP SW 1	DIP SW 2	BAUD-RATE
	ON	ON	2.5Mbps
	ON	OFF	5Mbps
	OFF	ON	10Mbps
	OFF	OFF	20Mbps

Chapter

4

HOW TO INSTALL

. 쉬운 설치는 제품의 가용성을 더욱더 증대 시키게 됩니다. C-NET 의 모든 제품군들에서는 쉽고 빠른 설치, 그리고 안정적인 설치를 위해 많은 분야에 접목될 수 있는 구조로 설계되어 있습니다. 이제 C-NET 을 통해서 경쟁력있는 제품 개발에 효과를 거두시기 바랍니다.

본 장에서는 C-NET 디바이스들의 최선의 성능을 발휘하는데에 도움이 되는 내용들을 포함하고 있는 포함하고 있는 장입니다. 처음 제품을 사용하시는 분들과 초보자들께서는 반드시 정독하여 주시기 주시기 바랍니다.

4 How to install

4.1 안전상의 주의

취급을 잘못한 경우, 인명피해 및 물적 손해 발생이 예상되는 경우입니다.

	설치 및 배선작업 시 반드시 주전원을 차단하고 설치하십시오.
	습기찬 곳에 보관, 설치하지 마십시오.
	젖은 손으로 제품을 조작하지 마십시오.
	부식성, 인화성 가스 또는 가연성 물질 근처에서 사용하지 마십시오.

취급을 잘못한 경우, 부상 및 물적 손해 발생이 예상되는 경우입니다.

	설치 및 배선작업 시 배선끼리 서로 단락되지 않도록 주의하십시오.
	단자 접속 및 극성에 오류가 없도록 하십시오.
	제품의 표면에 전도성 물질이 접촉되지 않도록 주의하십시오.

	사용자 임의로 제품을 분해, 개조하지 마십시오.

반드시 사용자 매뉴얼에 따라서 설치, 사용하시기 바랍니다. 매뉴얼의 지시사항을 위반하여 사용시 사후 무상 서비스나 제품의 무상 교환의 결격사유가 될 수 있습니다.

4.2 HARDWARE INSTALL

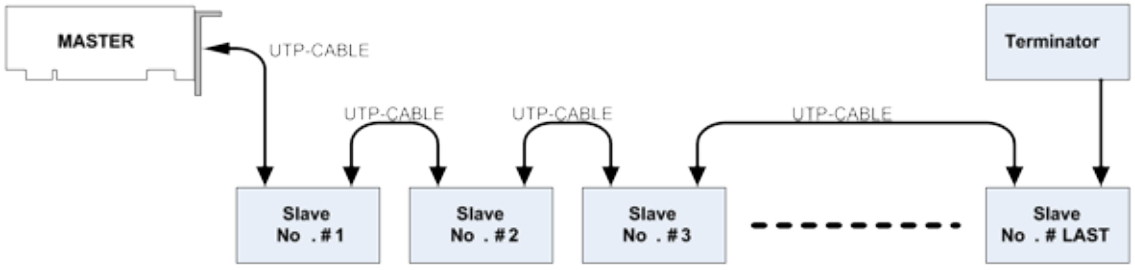


그림 4-1 C-NET Application 구성도

주의	※ Ring 의 마지막 Slave module 은 반드시 Terminater 를 장착해야 합니다. Terminator 는 UTP 케이블처럼 RJ-45 커넥터에 장착하는 방식이므로 별도의 케이블이 필요치 않습니다.

4.3 Address, Baudrate setting

4.3.1 Address setting

동일한 Ring 을 구성하는 모든 C-NET 슬레이브 모듈은 독립된 Address 를 가지고 있어야 합니다.

만약 Address 가 충돌할 경우 정상적인 동작을 보장할 수 없습니다.

통합 DIO 제어 시 접점은 Address 의 순서대로 배치되게 됩니다.

Example

MSB	LSB	Address
10 진 BDC	16 진 HEX	MSB x 16 + LSB

MSB 가 2, LSB 가 A 로 설정되어 있을 경우 해당슬레이브 모듈의 address 는 2 x 16 + 10 = 42 가 됩니다.

유효한 Address 의 범위는 0(MSB:0, LSB:0) ~ 63(MSB:3, LSB:F) 입니다.

구동중 Address 설정을 변경하지 마십시오.

4.3.2 Baudrate setting

모든 C-NET Slave module 은 동일한 baudrate 로 설정되어 있어야 합니다.

구동중 Baudrate 설정을 변경하지 마십시오.

4.4 전원 인가 및 통신 확인

배선이 완료된 상태에서 슬레이브 모듈에 전원을 인가하면 PWR LED 가 점등되게 됩니다.

동봉된 C-NET 전용 Application 인 COMI-LINK 를 실행시켜서 Ring 이 활성화 되면 슬레이브 모듈의 ACT LED 가 점등되게 됩니다.

COMI-LINK 의 Indicater 에서 장착한 슬레이브 모듈이 모두 활성화 되었는지 확인하십시오.

Chapter

5

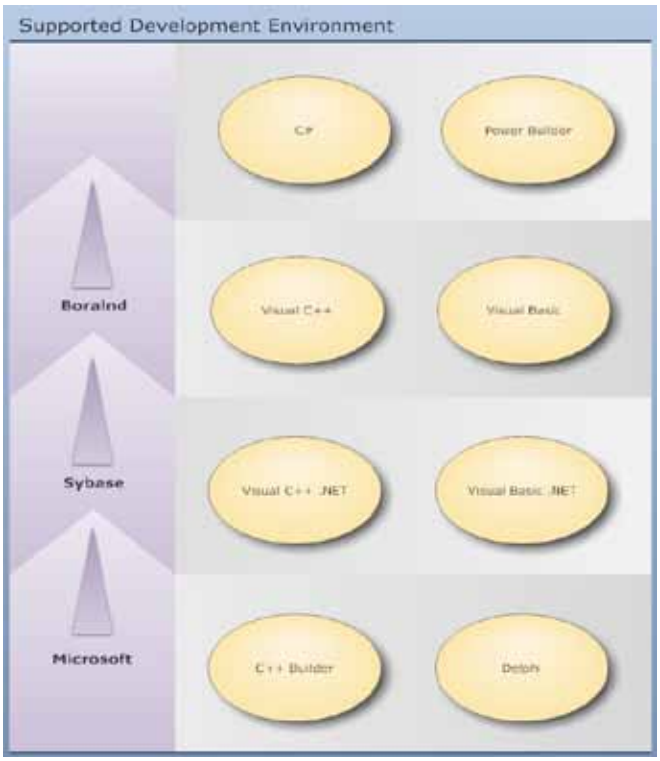
Development Environment for CNETSDK

㈜커미조아는 CNETSDK 를 통해 다양한 최신 개발환경을 지원하기 위해 노력하고 있습니다. 본 장에서 다루지 않는 개발 환경을 이용하시는 고객님께서는 저희 ㈜커미조아를 통해 문의해주시면 신속히 대처해 드리도록 하겠으며, 제공되는 라이브러리 인터페이스를 통해 보다 편리하고 빠르게 사용할 수 있도록 지원하여 드립니다.

커미조아 C-NET 라이브러리는 사용의 편리성과 다중 플랫폼의 지원을 목표로 개발되었습니다. 편리한 개발되었습니다. 편리한 사용환경과 다중플랫폼의 지원으로 좀더 빠른 어플리케이션 개발을 경험해 개발을 경험해 보십시오.

5 개발 환경 별 CNETSDK 사용 안내

5.1 개발 환경 지원 안내



개발 환경 별	언어별	버전 별	제품 별
Microsoft 社 Visual Studio	Visual C++	6.0 및 이하 버전을 포함한 VS2003, VS2005 환경	Enterprise Edition 을 포함한 전제품
	Visual Basic		
	C# (Sharp)		
Borland International 社 Borland Development Studio	C++ Builder	Builder 5 및 Builder 6 및 하위 버전 포함한 2006 버전	Architect, Professional, Enterprise
	Delphi	Delphi 5 및 Delphi 6, Delphi 7 및 하위 버전 포함한 2006 버전	
	C# (Sharp)	BDS 2006 버전	
Borland International 社 Borland Turbo Series	C++ Builder	Turbo C++	전 제품
	Delphi	Turbo Delphi / Turbo Delphi for .NET	전 제품
	C# (Sharp)	Turbo C#	전 제품
Sybase PowerBuilder	PowerBuilder	PowerBuilder 8, PowerBuilder 9, PowerBuilder 10.5	전 제품

표 5-1 지원 개발 환경

5.2 CNETSDK 구성

CNETSDK 는 아래 그림과 같이 하드웨어를 추상화하여, 공통 API 구조를 제공하고 있습니다.

API 는 “Application Programming Interface” 라 명명하며, 그 의미는 저희 ㈜커미조아 제품군의 기능과 그 기능을 사용하는 방법을 정의한 함수들(Functions)의 집합이라고 말할 수 있습니다. 고객님께서는 이제 저희 ㈜커미조아 API 를 통해 C-NET 이 가지고 있는 다양한 기능을 사용할 수 있습니다.

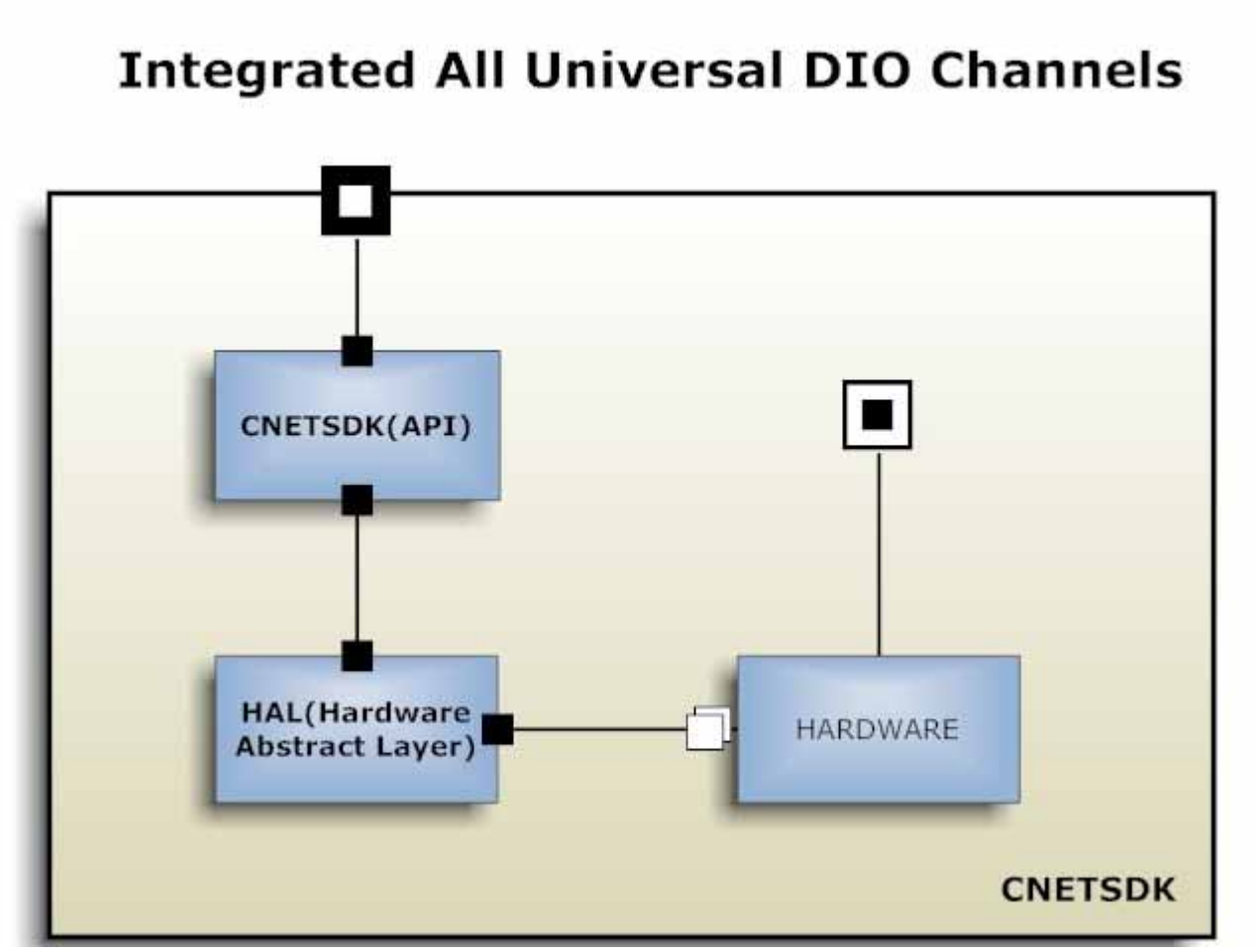


그림 5-1 CNETSDK 의 구조

5.2.1 HARDWARE Layer

하드웨어 구조를 추상화하여, 일관적인 API 인터페이스를 제공하게 해주는 CNETSDK 를 통해 편리하게 모든 장치를 일관적으로 제어하실 수 있습니다.

5.2.2 HAL(Hardware Abstract Layer)

CNETSDK 는 고급 HAL(Hardware Abstract Layer)기능과 최상의 속도와 안정적인 동작을 보장하기 위한 다양한 기법으로 구조화 되어 있습니다. 고객님께서는 CNETSDK 를 통하여 편리하고 안정적인 경험을 하실 수 있습니다.

5.2.3 CNETSDK Layer (API Layer)

마이크로소프트社의 Windows 98/ME/2000/XP/Vista 등의 제품군에서 동적 연결 라이브러리 방식을 지원하는 라이브러리를 의미합니다. 사용자는 제품들의 개별적인 하드웨어 인터페이스를 직접

인지하고 있지 않아도 CNETSDK 는 고객여러분들의 요구에 맞는 인터페이스를 제공하여, 신속하고 안정적인 응용프로그램 개발에 도움을 드립니다.

5.2.4 CNETSDK의 인터페이스 구성 파일

개발 환경 / 항목	MS VC++	Borland C++ Builder	Borland Delphi	MS Visual Basic	MS C Sharp(C #)
CNETSDK 파일 명	CNETSDK.H	CNETSDK.H	CNETSDK.PAS	CNETSDK.BAS	CNETSDK.CS
	CNETSDK.CPP	CNETSDK.CPP			
CNETSDK 의 상수(常數)정의 파일	CNETSDKDef.H	CNETSDKDef.H			

표 5-2 CNETSDK 인터페이스 구성 파일 안내

Microsoft 社의 윈도우 운영체제에서 동작하는 DLL(Dynamic Link Library) 형태의 CNETSDK 는 상기 표에서 나타낸 것 처럼, 라이브러리 인터페이스를 위해 "라이브러리 인터페이스" 파일과 라이브러리의 반환값 및 전달인자, 각 데이터 표기등을 위한 "라이브러리 상수 정의 파일"을 제공하고 있습니다.

㈜ 커미조아 고객님께서는 "라이브러리 인터페이스 파일" 을 통해 CNETSDK 의 함수를 명시적으로 응용프로그램에 포함시킬 수 있으며, 이에 따라 상기 표기한 개발 환경을 지원해드리고 있습니다.

5.3 각 개발 환경 별 안내

저희 ㈜ 커미조아에서는 고객님들의 개발환경에 대한 고민을 덜어드리기 위하여, 범용적인 객체지향 언어인 C++ 부터 Borland International 社의 Object Pascal 의 Delphi 제품군, 그리고 **최신 .NET 환경까지 고려한 라이브러리 인터페이스를 제공**하고 있습니다. 특히 이번 CNETSDK 에서는 개발 환경의 새 패러다임인 C Sharp(C#) 언어를 본격 지원하고 있으며, 보다 새로운 개발환경에서 저희 ㈜ 커미조아 제품 군을 경험하실 수 있도록 만전을 기하고 있습니다.

필요한 인터페이스(Interface) 파일은 실제 개발언어 및 환경에서 헤더파일 또는 프로젝트 구성파일로 반드시 사용이 됩니다.

인터페이스 파일에 대해서 자세히 설명해 주십시요.

먼저, 저희 ㈜ 커미조아에서 제공하는 CNETSDK 는 DLL(순 우리말로 동적 연결 라이브러리라고 함) 형태이며, 이것은 당사의 제품의 기능을 담당하는 가장 중요한 함수의 집합입니다.

고객님들께서는 이러한 함수, 즉 기능을 통해서 저희 ㈜ 커미조아의 제품 군을 이용하시게 됩니다.
그러나 마이크로소프트의 DLL 라이브러리 형태는 이 함수들을 어떻게 써야 하는 지의 정보를 정확하게 알 수 없습니다. 다시 말씀 드려서, 함수의 매개변수의 개수나 매개변수의 형태, 다시 말해 함수의 사용법을 알 수가 없습니다.

CNETSDK 에서는 해당 DLL 라이브러리에 대해서 그 사용방법을 각 개발환경에 맞게 제공해드리기 위해 [인터페이스] 파일을 제공하고 있습니다. 각 개발환경(VC++, Delphi) 등의 환경에서 개발을 하시는 고객 여러분들께서는 반드시 이 [인터페이스] 파일을 사용하셔야만, CNETSDK 를 사용할 수 있습니다.

5.3.1 Visual C++ 6.x 개발자를 위한 안내

Microsoft Visual C++ 6.0 에서 CNETSDK 를 사용하시려면 다음의 절차에 따라 사용하시면 됩니다.

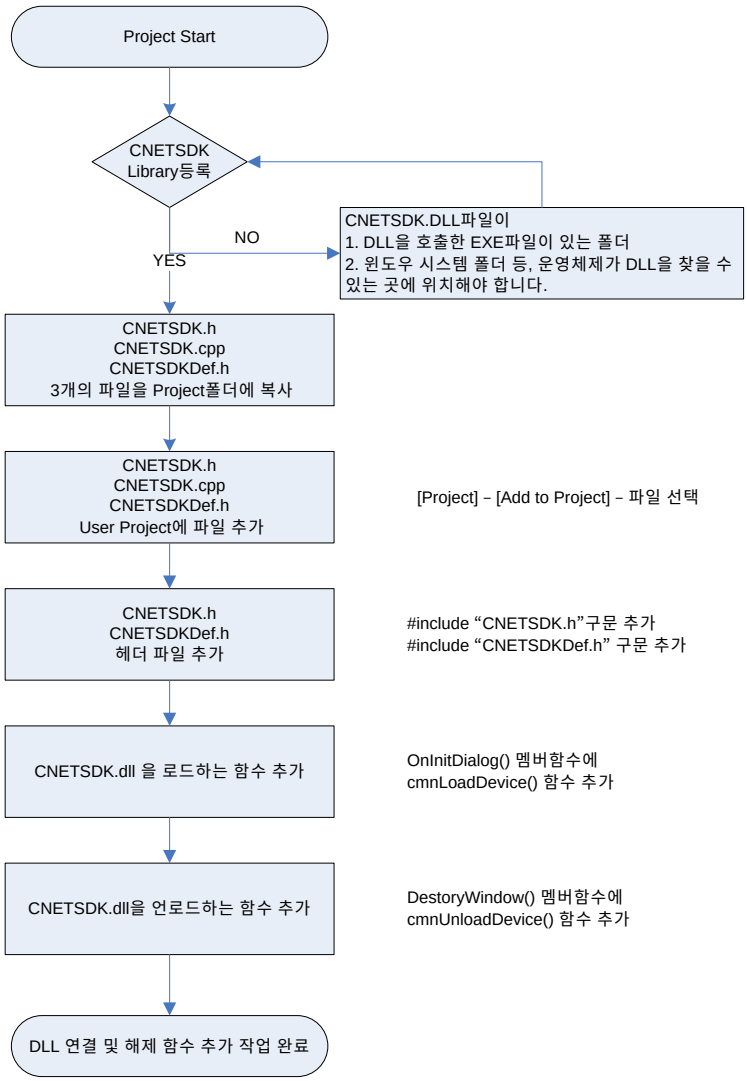


그림 5-2 Visual Studio 6.x 에서의 CNETSDK 사용 순서도

Visual C++ 6.x 을 실행합니다. 메뉴에서 'File'->'New' 를 선택하여 새로운 프로젝트 생성합니다.

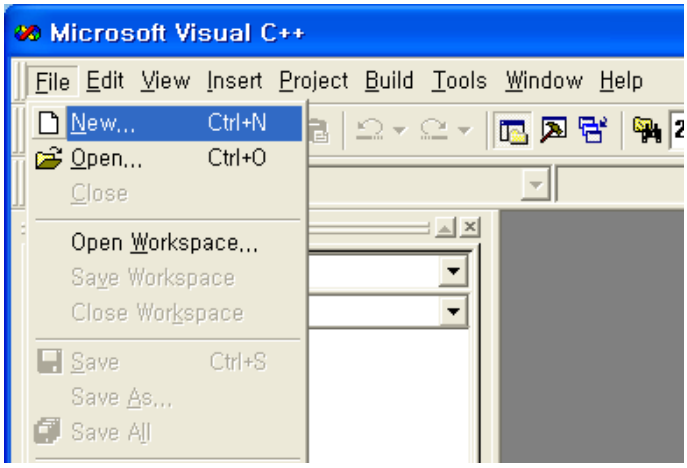


그림 5-3 Visual C++ 6.x 의 새로운 프로젝트 생성 화면

MFC AppWizard(exe)를 선택하고, 프로젝트를 생성할 위치와 프로젝트 이름을 입력한 후 [OK]버튼을 클릭 합니다.

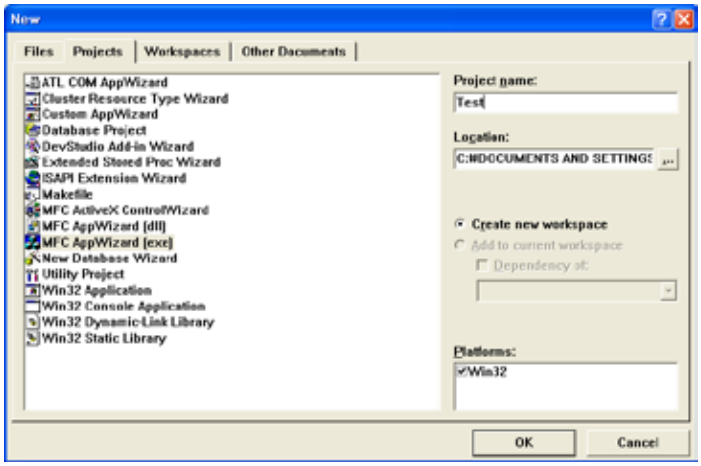


그림 5-4 새로운 프로젝트 생성 화면

MFC AppWizard 창이 나타나면 [Dialog based]를 선택하고 [Finish]버튼을 클릭합니다.

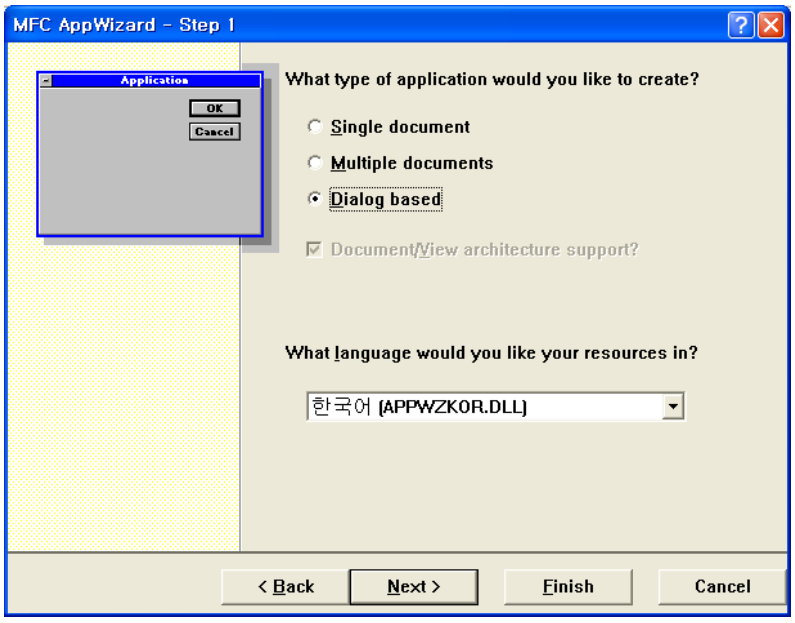


그림 5-5 MFC AppWizard 의 Application Type 선택 화면

VC++ 용 인터페이스 정의 파일인 CNETSDK.h, CNETSDK.cpp, CNETSDKDef.h 파일을 신규로 생성한 프로젝트 폴더로 복사 합니다.

메뉴에서 [Project]->[Add To Project]->[Files]를 선택합니다.

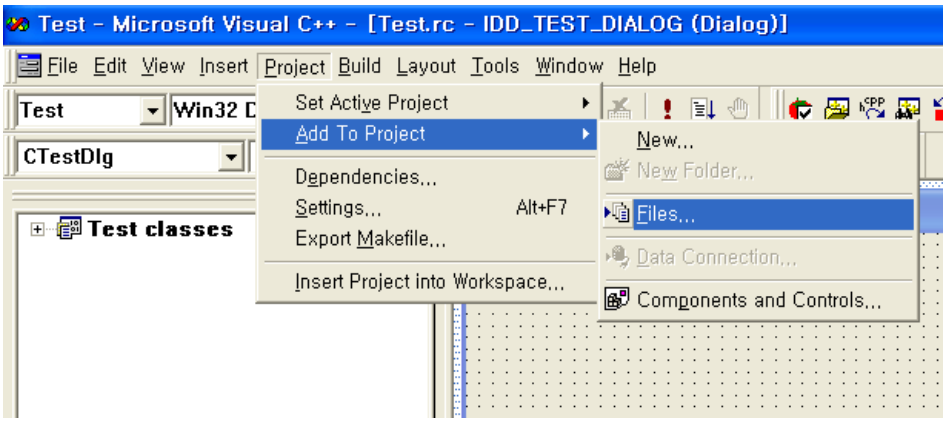


그림 5-6 프로젝트에 새로운 파일 추가 선택화면

추가될 파일을 선택한 후 [OK]버튼을 클릭하여 C-NET 의 인터페이스 파일인 세 개의 파일을 프로젝트에 추가 합니다.

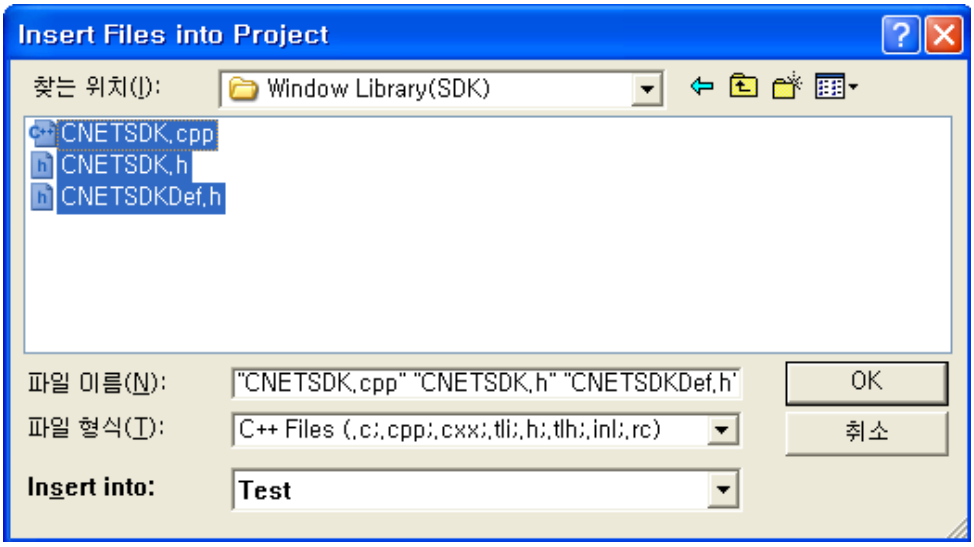


그림 5-7 프로젝트에 추가할 파일 선택 화면

WorkSpace 창의 FileView 탭에서 [생성한 프로젝트 이름]+Dlg.cpp 파일을 선택합니다.

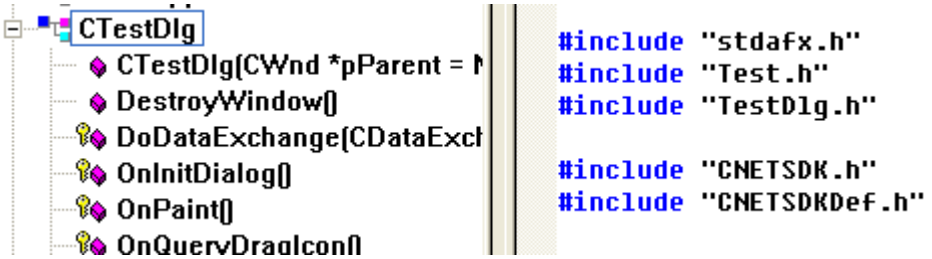


그림 5-8 사용자가 MFC AppWizard 를 통해서 생성한 소스코드에 CNETSDK 파일을 추가 함

([생성한 프로젝트 이름]+Dlg.cpp) 파일의 OnInitDialog()함수 내부의 “TODO”아래에 “CNETLoadDll();”을 추가 합니다.

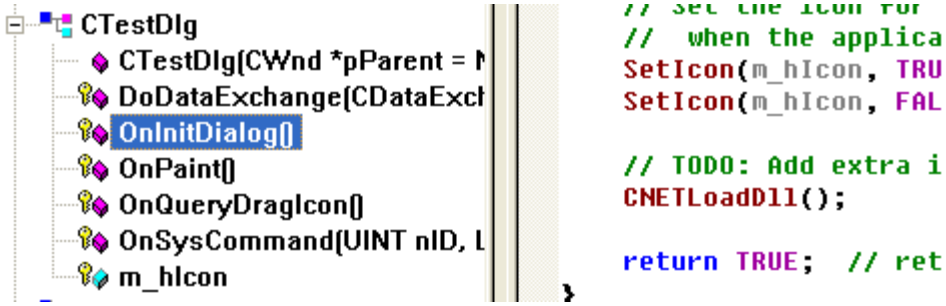


그림 5-9 DLL 로드 함수 호출 화면

사용자 작성 프로그램이 종료되면 DLL 을 Unload 시켜야 합니다. DLL 의 Unload 는 사용자 작성 프로그램의 종료시 이루어져야 하며 CNETUnloadDll()이라는 함수를 통해서 이루어 집니다. CNETUnloadDll ()을 추가 하는 방법은 다음과 같습니다.

Class View 창에서 [(생성한 프로젝트 이름)+Dlg] 클래스를 마우스 오른쪽 버튼으로 클릭 합니다. 팝업 메뉴에서 [Add Virtual Function]을 선택합니다.

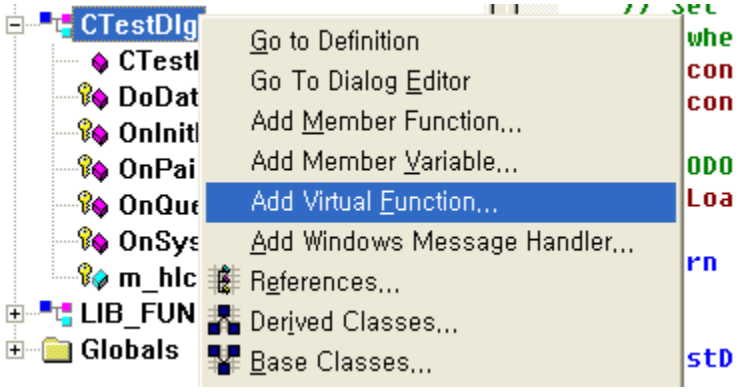


그림 5-10 가상 함수 추가

'New Virtual Functions'항목에서 'DestroyWindow'를 선택한 다음 [Add and Edit]버튼을 클릭합니다.

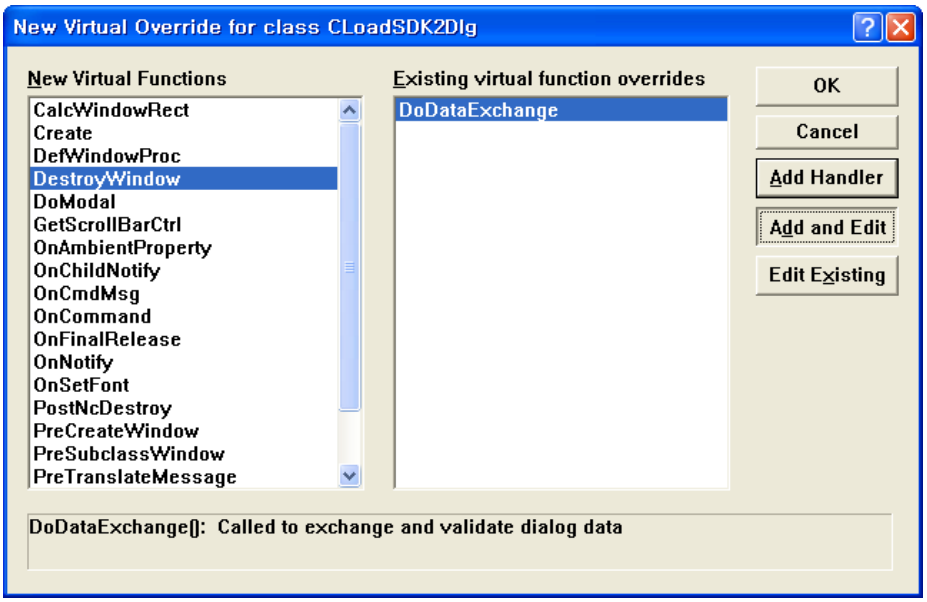


그림 5-11 DestroyWindow 함수 추가

(생성한 프로젝트 이름)+Dlg 클래스의 멤버함수인 'DestroyWindow()'에 'CNETUnloadDll ();'을 추가 합니다. 'CNETUnloadDll ()'함수를 추가 하면 윈도우가 종료 될 때 자동으로 DLL 도 해제 됩니다.

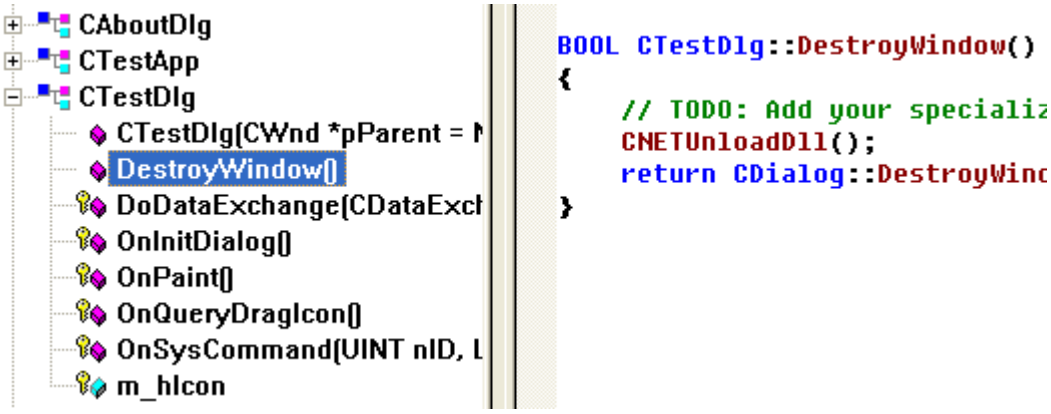


그림 5-12 DLL 로드 및 언로드 코드 추가

5.3.2 Visual C++ 7.x 개발자를 위한 안내

Microsoft 社의 Visual C++ 7.x(Visual Studio 2003)에서 CNETSDK 를 사용하시려면 다음의 절차에 따라 사용하시면 됩니다.

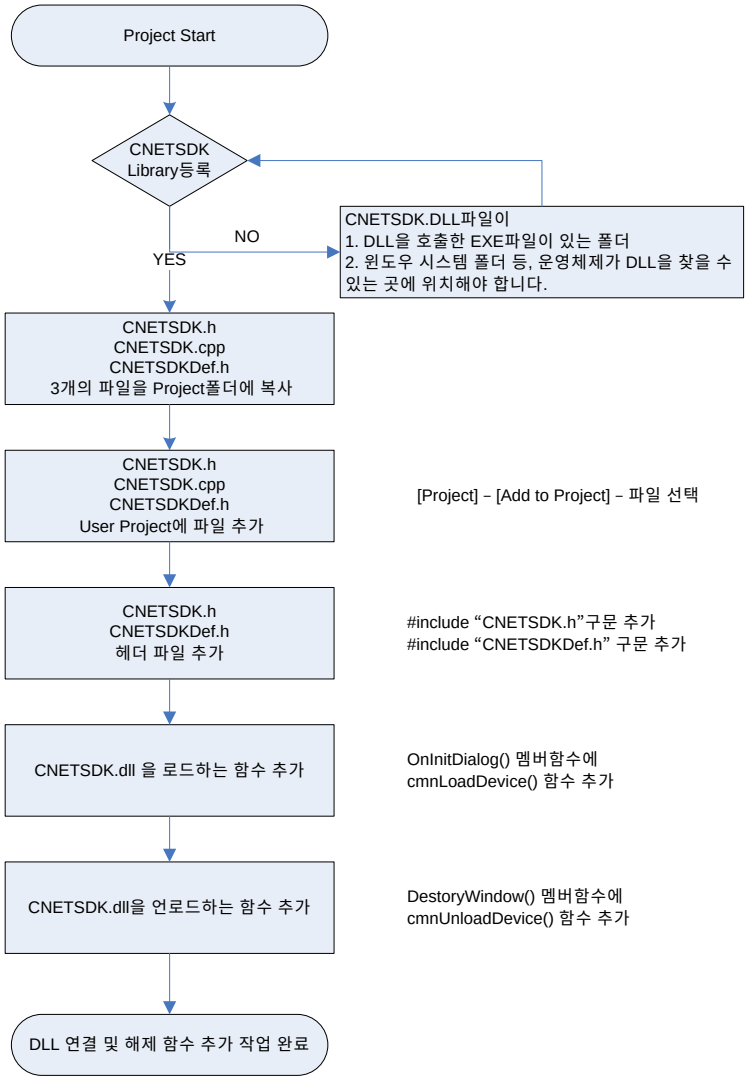


그림 5-13 Visual Studio 7.x 에서의 CNETSDK 사용 순서도

Microsoft 社의 Visual Studio 2003 을 실행합니다.

메뉴에서 [File]->[New]를 선택하여 Project 를 시작 합니다

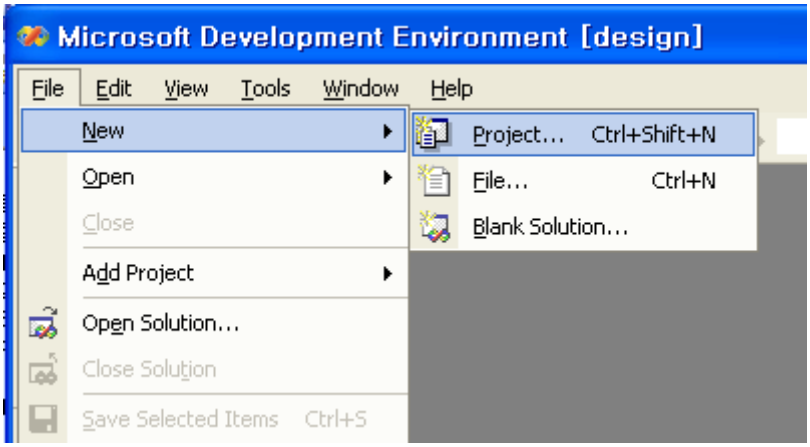


그림 5-14 프로젝트 생성의 시작

[New Project]창의 [Project Types]에서 [Visual C++ Project]를 선택하고, [Templates]에서 [MFC Application]을 선택합니다. 그리고 프로젝트를 생성할 위치와 프로젝트 이름을 입력한 후 [OK]버튼을 클릭 합니다.

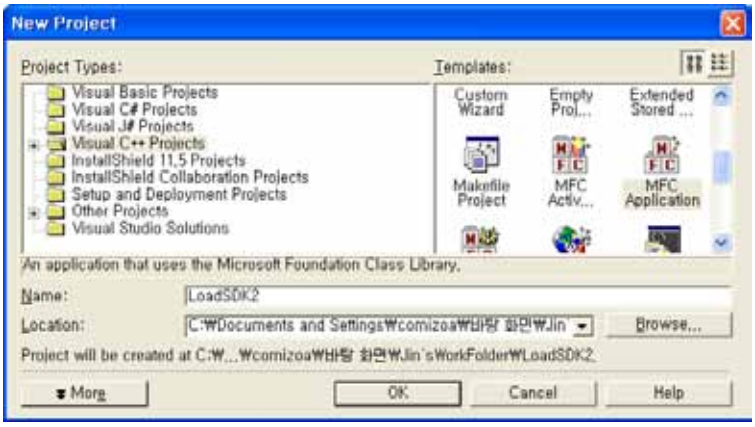


그림 5-15 프로젝트 경로 입력

[MFC Application Wizard]창이 나타나면, [Application Type]를 선택합니다.

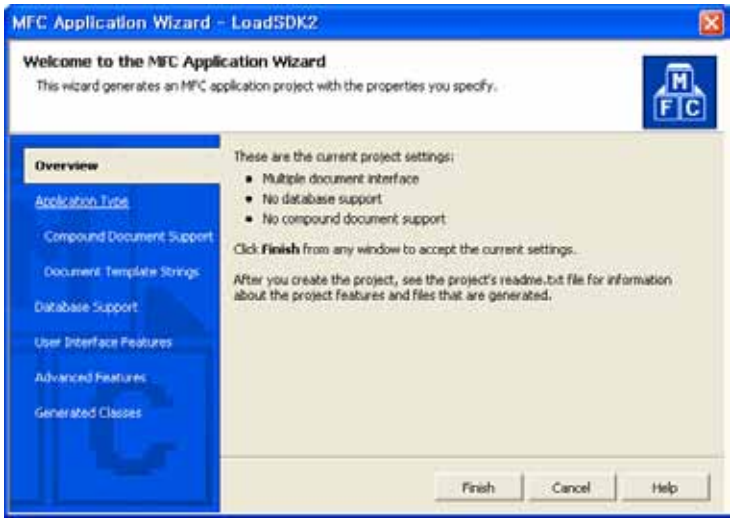


그림 5-16 응용 프로그램 종류 선택

[Application Type]카테고리에서 [Dialog based]를 선택한 후 마침을 클릭합니다.

그림 5-17 대화상자 기반 프로젝트 생성

VC++ 용 인터페이스 정의 파일인 CNETSDK.h, CNETSDK.cpp, CNETSDKDef.h 파일을 신규로 생성한 프로젝트 폴더로 복사 합니다.

메뉴에서 [Project]->[Add Existing Item]를 선택합니다.

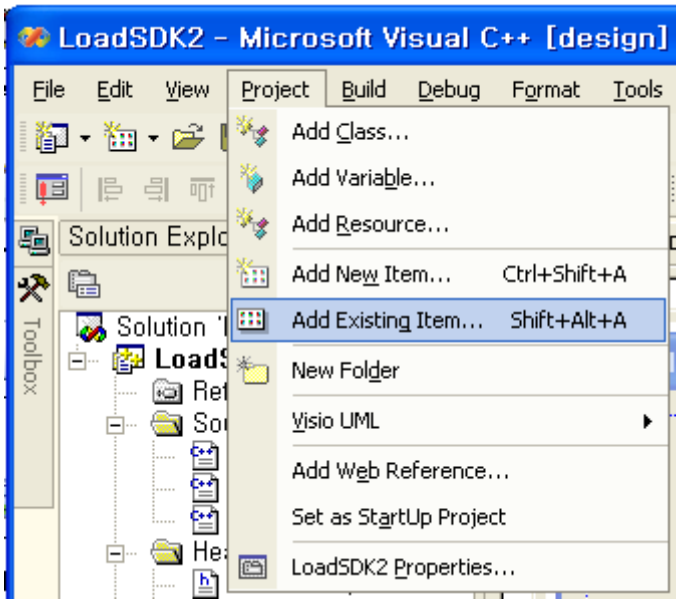


그림 5-18 기존 항목 추가 선택

추가될 파일을 선택한 후 [OK]버튼을 클릭하여 제공되는 헤더 및 소스 코드 파일을 프로젝트에 추가 합니다.

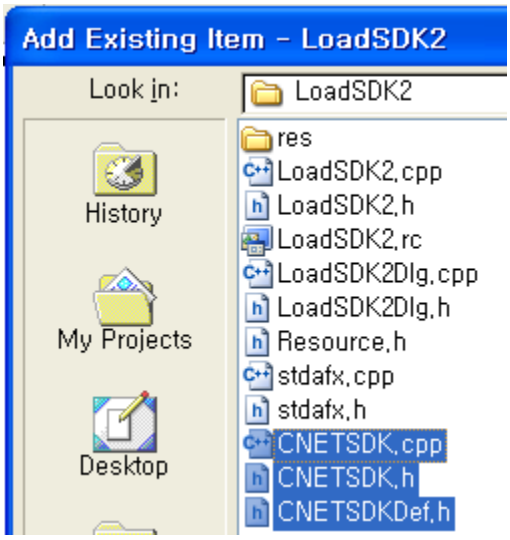


그림 5-19 헤더파일 및 소스 파일 추가

WorkSpace 창의 FileView 탭에서 ([생성한 프로젝트 이름]+Dlg.cpp) 파일을 선택합니다. 파일의 가장 위쪽에 인터페이스 파일을 추가합니다.

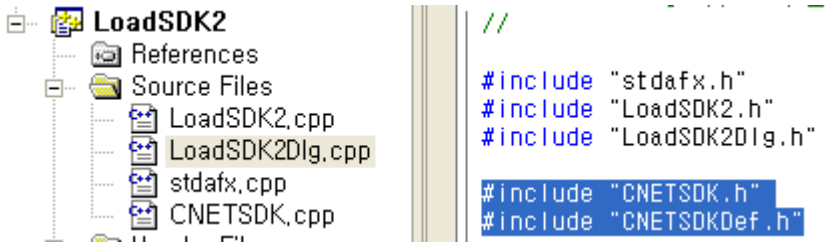


그림 5-20 인터페이스 헤더 파일 추가

(생성한 프로젝트 이름)+Dlg.cpp 파일의 OnInitDialog()함수 내부의 “TODO”아래에 “CNETLoadDll ();”을 추가 합니다.

```
// TODO: 여기에 추가 초기화 작업을 추가합니다.
CNETLoadDll();
```

그림 5-21 CNETLoadDll

사용자 작성 프로그램이 종료되면 DLL 을 Unload 시켜야 합니다. DLL 의 Unload 는 사용자 작성 프로그램의 종료시 이루어져야 하며, CNETUnloadDll ()이라는 함수를 통해서 이루어 집니다. CNETUnloadDll ()을 추가 하는 방법은 다음과 같습니다.

Class View 창에서 ([생성한 프로젝트 이름]+Dlg) 클래스를 선택합니다.

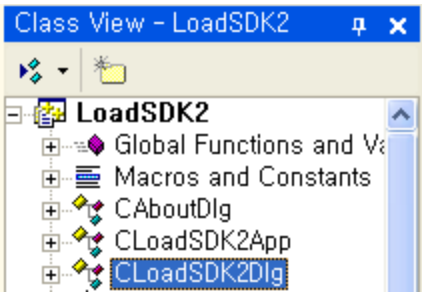


그림 5-22 사용자 생성 Dialog Class 선택

([생성한 프로젝트 이름]+Dlg) 클래스가 선택된 상태에서 Properties 창의 ‘Overrides’를 선택합니다.

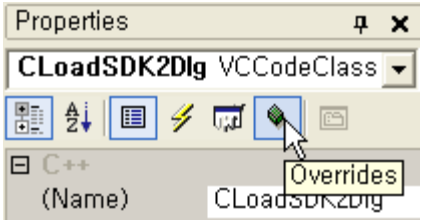


그림 5-23 Overrides 선택

‘DestroyWinodw’항목 옆의 콤보 박스를 클릭하여 ‘<Add>DestroyWindow’를 선택합니다. ([생성한 프로젝트 이름]+Dlg) 클래스에 ‘DestroyWindow’라는 이름의 오버라이드된 함수가 추가 됩니다.

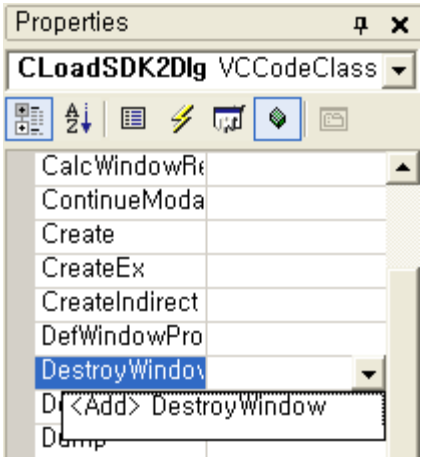


그림 5-24 Destroy Window 함수 추가

(생성한 프로젝트 이름)+Dlg 클래스의 멤버함수인 DestroyWindow()에 ‘CNETUnloadDll ();’을 추가 합니다. ‘CNETUnloadDll ()’함수를 추가 하면 윈도우가 종료 될 때 자동으로 동적으로 연결된 CNETSDK 가 함께 해제 됩니다.

```
BOOL CLoadSDK2Dlg::DestroyWindow()
{
    // TODO: Add your specialized code
    CNETUnloadDll();

    return CDialog::DestroyWindow();
}
```

그림 5-25 CNETUnloadDll 추가

CNETSDK 의 라이브러리 방식인 DLL 의 이점은 어떤 것이 있습니까?

예를 들어 드리겠습니다. 일반적인 C 나 C++ 개발환경에서 기본적인 Standard Input/Output Header 파일을 포함해서 사용합니다. 여기서 분명, 고객님들께서 잘 아시는 <stdio.h> 인터페이스 파일을 사용하게 됩니다.

하지만 이러한 stdio.h 에 정의된 함수들을 우리가 다 사용하는 것은 아닙니다. 그러나 컴파일러는 이러한 함수의 내용들을 실제 고객님께서 개발하시는 응용프로그램에 컴파일 시간(엄밀히 말하면 Linking 시간)에 모두 포함시켜버립니다. 이러한 형태는 실제 사용되지 않는 함수들 때문에 실행파일의 크기가 커져버리며, 그만큼 메모리를 많이 차지하게 됩니다.

DLL 은 이러한 라이브러리를 실행시간때에 연결 시키는 것을 가능하게 합니다. 즉, 필요한 기능을 실행시에 연결시킬 수 있도록 만들어 놓은 것이 DLL 이며, 각 응용프로그램의 프로세스(Process) 의 코드와 데이터들은 메모리 영역의 각기 별도의 영역을 갖지만 라이브러리 부분과 공유 데이터 영역은 단지 한 번만 올라왔을 뿐이고 그 이후부터는 공유해서 사용합니다.
DLL 라이브러리는 함수를 다른 프로세스와 공유해서 사용하기 위해 개발되었으며, ㈜커미조아 라이브러리 자체는 공유영역의 메모리에 올라가서 상주하게 됩니다. 인텔社의 80386 이상의 CPU 에서는 프로세서(Processor) 자체에서 공유 영역을 구현할 수 있는 특수한 기법을 제공합니다.

5.3.3 Visual C++ 8.x 개발자를 위한 안내

Microsoft 社의 Visual C++ 8.x (Visual Studio 2005)에서 CNETSDK 를 사용하시려면 다음의 절차에 따라 사용하시면 됩니다.

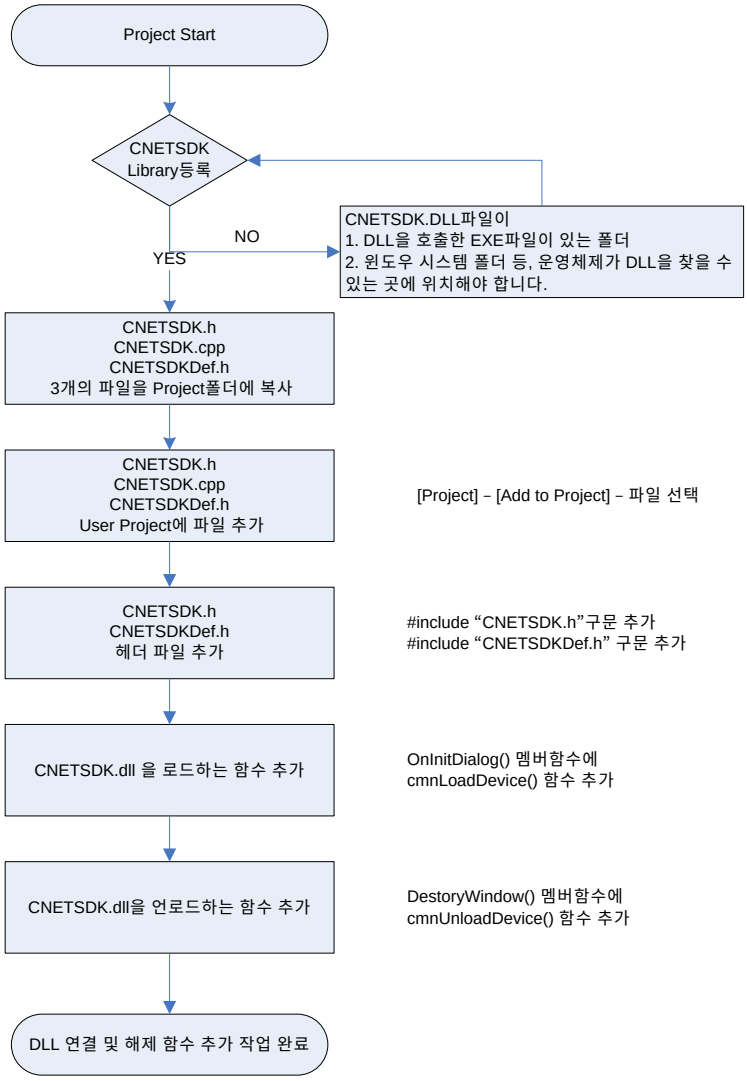


그림 5-26 Visual Studio 8.x 에서의 CNETSDK 사용 순서도

Microsoft 社의 Visual Studio 2005(이하 VS2005)를 실행 합니다.

메뉴에서 [File]->[New]->[Project]를 선택하여 새로운 프로젝트를 시작합니다.

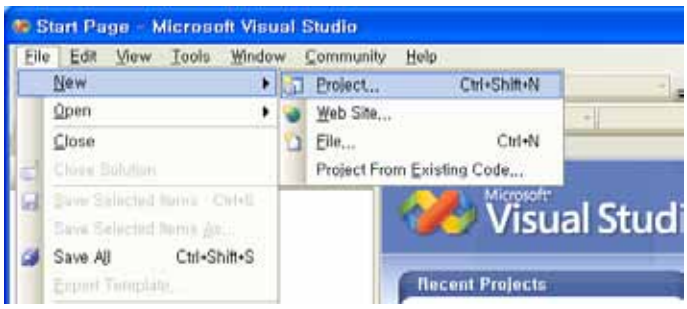


그림 5-27 새로운 프로젝트 생성 시작

[New Project]창이 화면에 나타나면, [Project types]에서는 [Visual C++]을 선택하고, [Templates]에서 [MFC Application]을 선택합니다. 그리고 프로젝트를 생성할 위치와 프로젝트 이름을 입력한 후 [OK]버튼을 클릭 합니다.

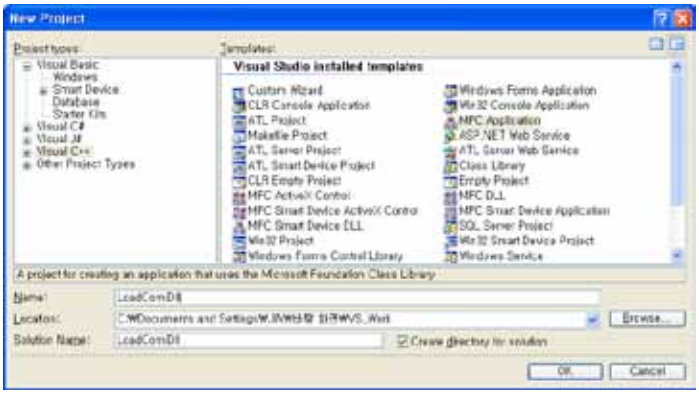


그림 5-28 새 프로젝트 옵션 선택화면

[MFC Application Wizard] 창이 화면에 나타나면, [Next]를 클릭합니다.

그림 5-29 MFC Application Wizard 의 Overview 화면

[Application Type]에서 [Dialog based]를 선택하고, [Use Unicode[1] Libraries]를 해제(Uncheck) 한 다음 [Finish]를 클릭합니다.

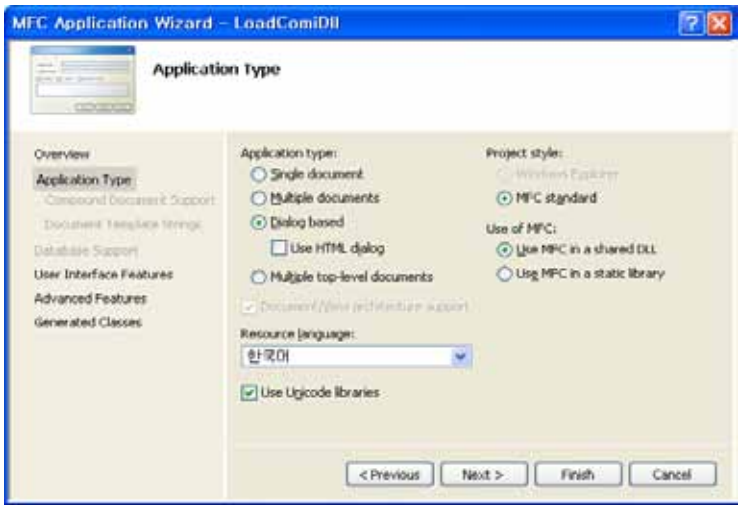


그림 5-30 MFC Application Wizard 의 Application Type 화면

VC++ 용 인터페이스 정의 파일인 CNETSDK.h, CNETSDK.cpp, CNETSDKDef.h 파일을 신규로 생성한 프로젝트 폴더로 복사합니다.

메뉴에서 [Project]->[Add Existing Item]을 선택합니다.

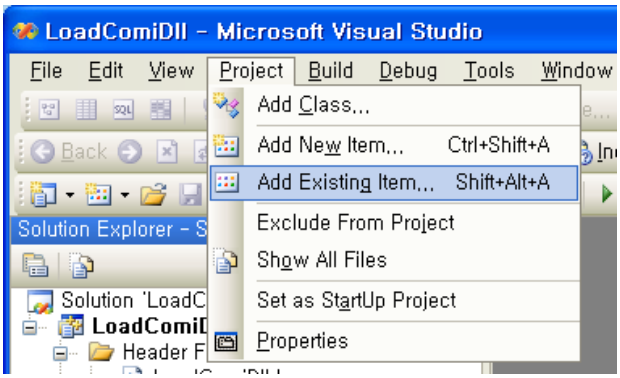


그림 5-31 메뉴에서 Add Existing Item 선택 화면

1 국제표준으로 제정된 2 바이트계의 만국 공통의 국제 문자부호 체계(UCS: Universal Code System)를 말한다. 애플컴퓨터·IBM·마이크로소프트 등이 컨소시엄으로 설립한 유니코드(Unicode)가 1990 년에 첫 버전을 발표하였고, ISO/IEC JTC1 에서 1995 년 9 월 국제표준으로 제정하였다. 공식 명칭은 ISO/IEC 10646-1(Universal Multiple-Octet Coded Character Set)이다.

추가될 파일을 선택한 후 [OK]버튼을 클릭하여 세 개의 파일을 프로젝트에 추가합니다.

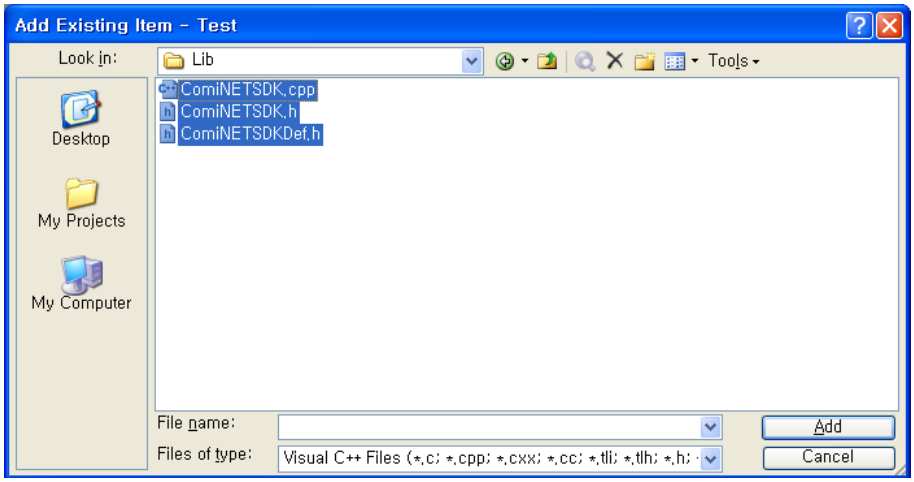


그림 5-32 새로 추가할 파일들 선택 화면

WorkSpace 창의 FileView 탭에서 ([생성한 프로젝트 이름]+Dlg.cpp) 파일을 선택합니다.

파일의 가장 위쪽에 인터페이스 파일을 추가합니다.

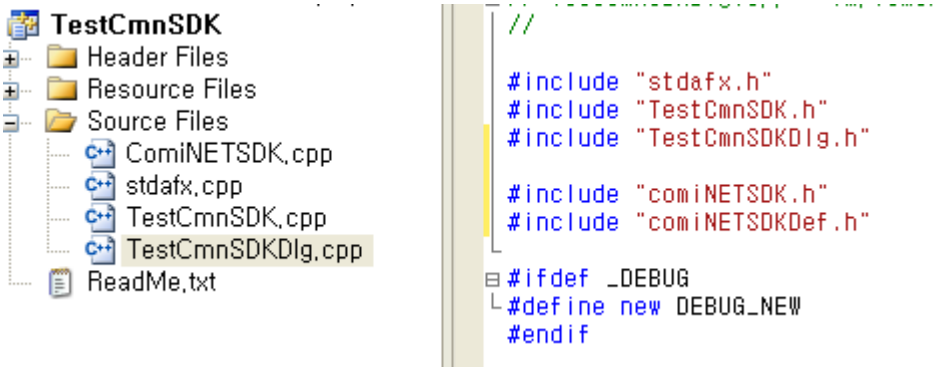


그림 5-33 사용자 생성 CPP 파일에 헤더 추가 화면

(생성한 프로젝트 이름)+Dlg.cpp 파일의 OnInitDialog() 함수 내부의 “TODO”아래에 “CNETLoadDll ();”을 추가 합니다.

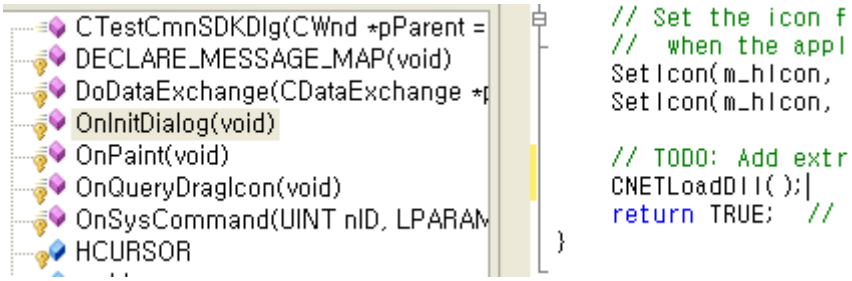


그림 5-34 LoadDll 추가

사용자 작성 프로그램이 종료되면 DLL 을 Unload 시켜야 합니다. DLL 의 Unload 는 사용자 작성 프로그램의 종료시 이루어져야 하며 CNETUnloadDll ()이라는 함수를 통해서 이루어 집니다. CNETUnloadDll ()을 추가 하는 방법은 다음과 같습니다.

Class View 창에서 ([생성한 프로젝트 이름]+Dlg) 클래스를 선택합니다.

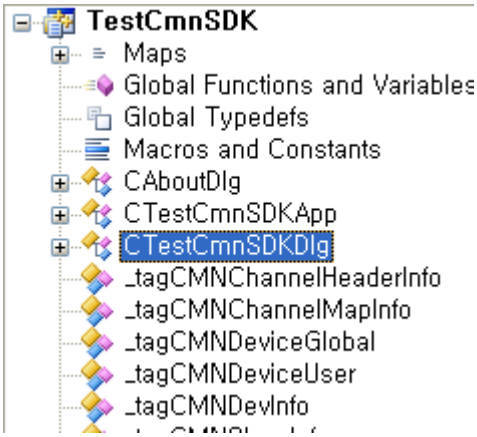


그림 5-35 사용자 생성 Dialog Class 선택

(생성한 프로젝트 이름)+Dlg 클래스가 선택된 상태에서 Properties 창의 'Overrides'를 선택합니다.

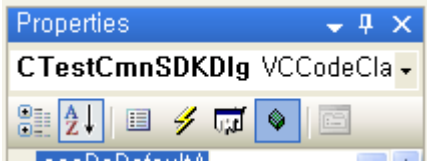


그림 5-36 Overrides 선택

'DestroyWinodw'항목 옆의 콤보 박스를 클릭하여 '<Add>DestroyWindow'를 선택합니다. (생성한 프로젝트 이름)+Dlg 클래스에 'DestroyWindow'라는 이름의 오버라이드된 함수가 추가 됩니다.

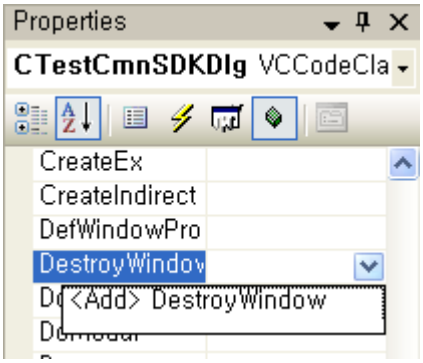


그림 5-37 Destroy Window 함수 추가

([생성한 프로젝트 이름]+Dlg) 클래스의 멤버함수인 DestroyWindow()에 'CNETUnloadDll ();'을 추가 합니다. 'CNETUnloadDll ()'함수를 추가 하면 윈도우가 종료 될 때 자동으로 DLL 도 해제 됩니다.

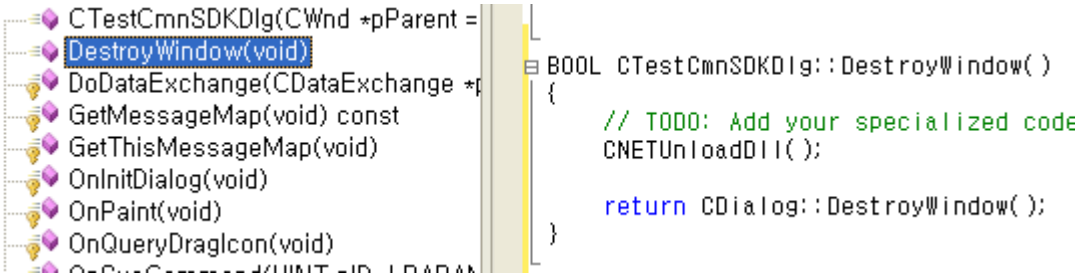


그림 5-38 CNETUnloadDll 함수 추가

5.3.4 Borland C++ Builder 개발자를 위한 안내

Borland C++ Builder 는 해당 개발 환경 버전인 BCB 5, BCB 6 및 2006 버전에서 CNETSDK 의 인터페이스 연결 방법이 매우 유사하기 때문에 공통적인 부분으로서 안내를 해드립니다.

전체 버전(Version)의 Borland C++ Builder 에서 CNETSDK 를 사용하시려면 다음의 절차를 통해 안내 받으시기 바랍니다.

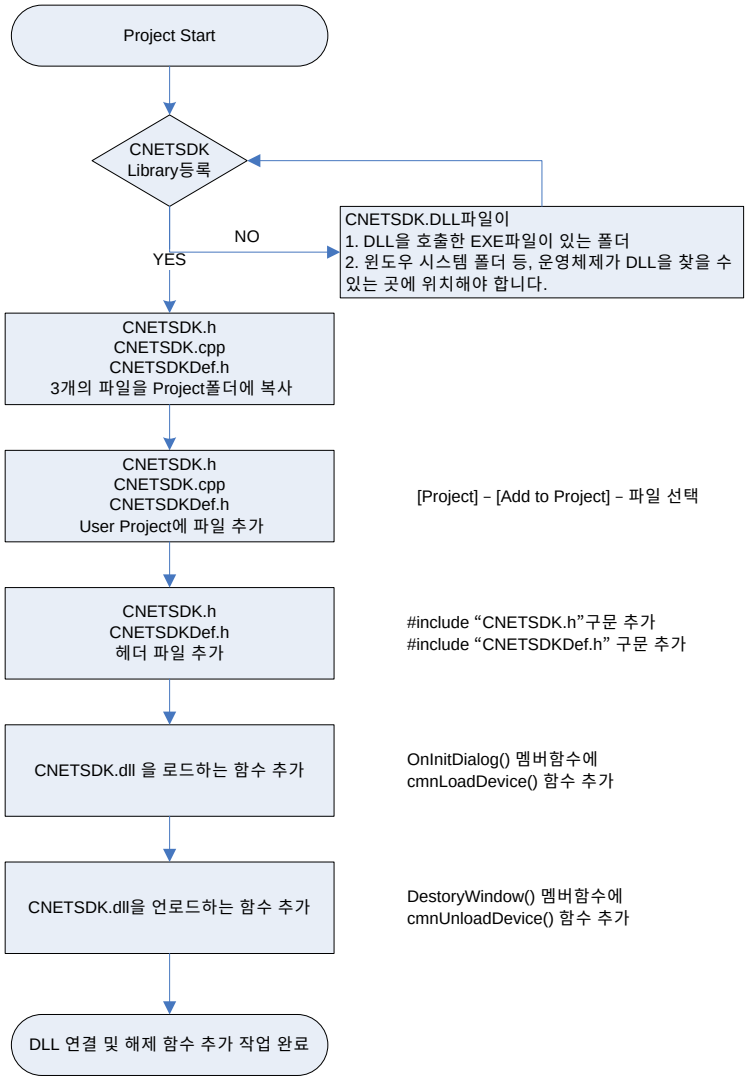


그림 5-39 Borland C++ Builder 에서 CNETSDK 사용 순서도

본 개발자를 위한 실제 안내에서는 다양한 버전의 Borland C++ Builder 의 화면을 통해 안내 해드리도록 하겠습니다.

Borland C++ Builder 를 실행합니다. 메뉴에서 [File]->[New]->[Application]을 선택하여 새로운 프로젝트를 시작합니다.

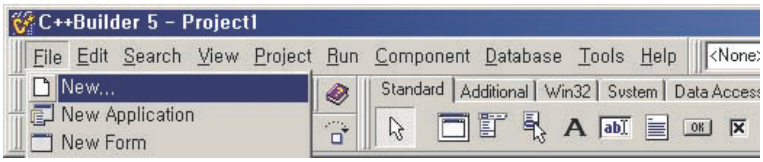


그림 5-40 BCB 5 에서 새로운 프로젝트 생성

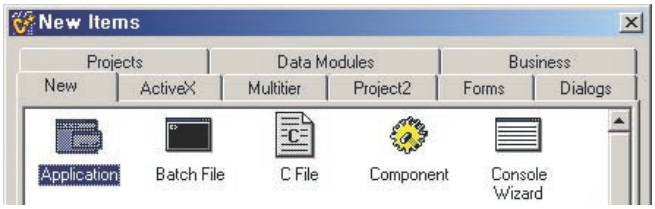


그림 5-41 BCB 5 에서 새로운 프로젝트 생성

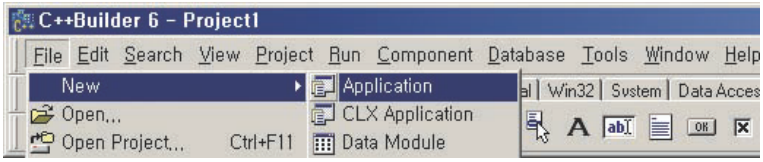


그림 5-42 BCB 6 에서 새로운 프로젝트 생성

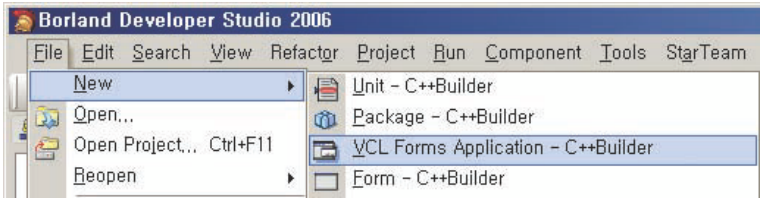


그림 5-43 BDS 2006 에서 새로운 프로젝트 생성

Borland C++ 및 VC++ 용 공용 인터페이스 정의 파일인 CNETSDK.h, CNETSDK.cpp, CNETSDKDef.h 파일을 신규로 생성한 프로젝트 폴더로 복사 합니다.

그림 5-44 CNETSDK 사용시 공통으로 사용되는 파일

	참고 사항
안내	Borland 社의 C++ Builder 에서는 [File]-[Save] or [Save All]을 해주어야 프로젝트 폴더 및 프로젝트 관련 파일들을 생성합니다.

그림과 같이 C++ Builder 에서 추가할 인터페이스 파일을 실제 사용자 프로젝트에 추가합니다. Project 의 메뉴의 Add to Project 를 사용하시면 됩니다.

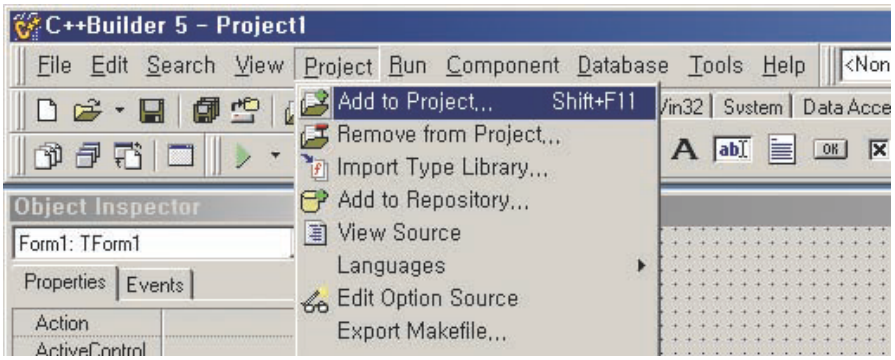


그림 5-45 C++ Builder 에서 프로젝트에 파일 추가 단계

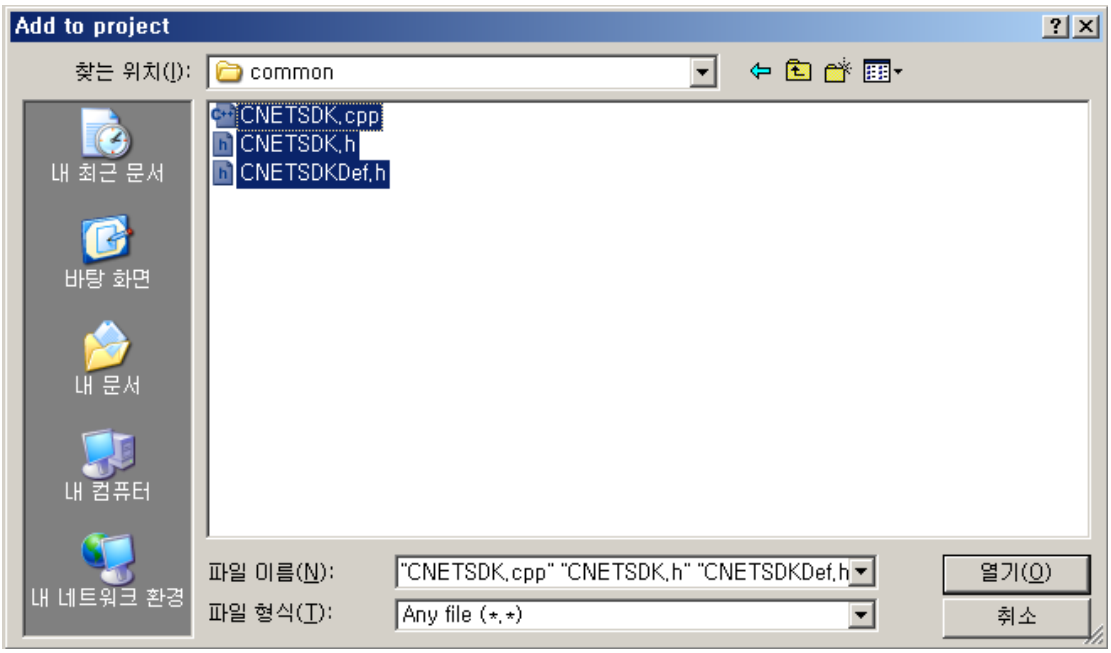


그림 5-46 C++ Builder 에서 프로젝트에 파일 추가

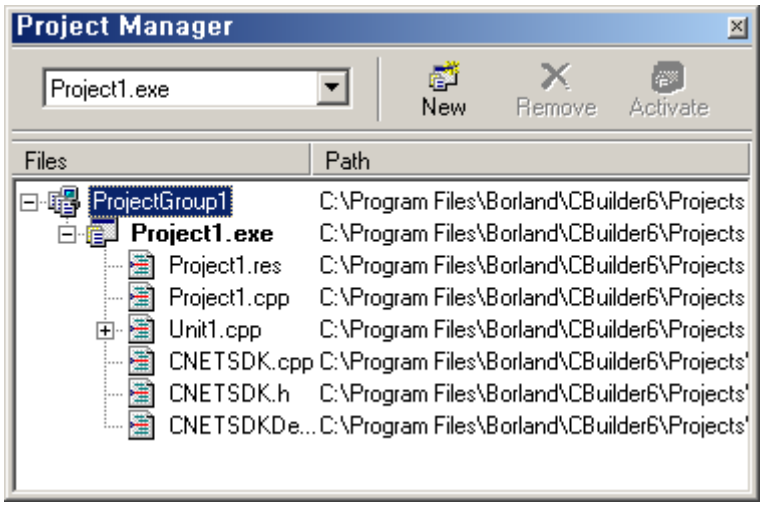


그림 5-47 Project Manager 에서 추가된 파일 확인

라이브러리 함수를 사용하고자 하는 대상 구현 부 응용프로그램 파일에 인터페이스 파일을 선언합니다.

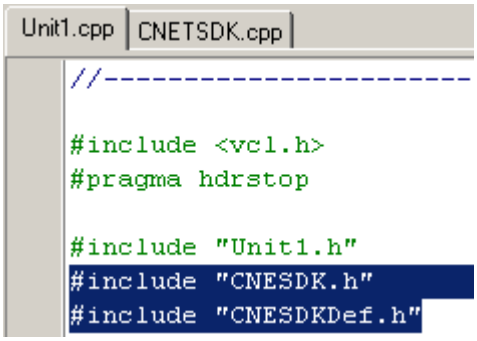


그림 5-48 라이브러리 사용시 필요한 헤더 파일 선언

CNETSDK.dll 파일을 cmmLoadDll()함수를 이용하여 로드할 수 있도록 FormCreate 함수 또는 응용프로그램 시작 부분에 추가합니다.

```
void __fastcall TForm1::FormCreate(TObject *Sender)
{
    CNETLoadDll();
}
```

그림 5-49 CNETSDK.dll 파일 로드

CNETSDK.dll 파일을 CNETLoadDll ()함수를 이용하여 로드합니다. CNETLoadDll ()을 추가 하는 방법을 FormCreate 를 통해 할 수 있으며, 그 예를 소개해 드립니다.

[Object Inspector] – [Events]탭의 OnCreate 에서 더블클릭합니다.

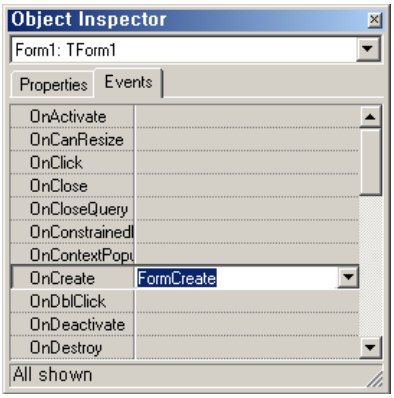


그림 5-50 OnCreate Event 추가하여 FormCreate 함수와 연결

추가된 FormCreate() 또는 응용프로그램 종료 함수 안에 CNETLoadDll ()함수를 추가합니다.

```
void __fastcall TForm1::FormCreate(TObject *Sender)
{
    CNETLoadDll();
}
```

[그림 12] FormCreate 함수에 cmmLoadDll 함수 추가

고객님이 작성하신 프로그램이 종료되면 DLL 을 명시적으로 Unload 시켜야 합니다.
DLL 의 Unload 시점은 고객님께서 작성하신 응용프로그램이 종료되는 시점에 반드시 이루어져야 하며 CNETUnloadDll()이라는 함수를 통해서 이루어 집니다. CNETUnloadDll()을 추가 하는 방법은 다음과 같습니다.

[Object Inspector] – [Events]탭의 OnDestroy 에서 더블클릭합니다.

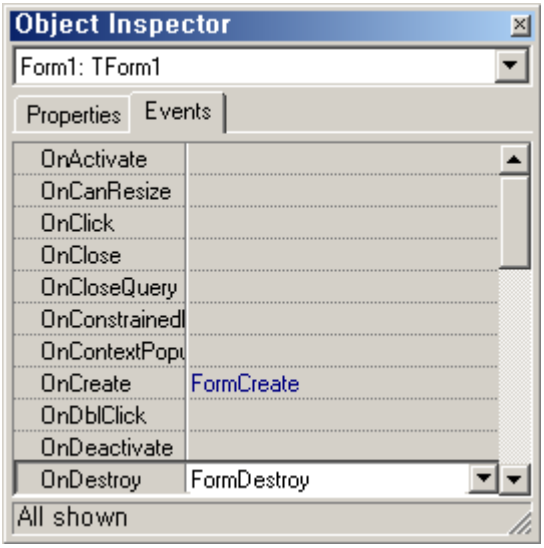


그림 5-51 응용프로그램의 종료시 DLL 이 명시적으로 Unload 될 수 있도록 OnDestroy Event 와 함수의 연결

추가된 FormDestroy()또는 응용프로그램 종료 함수에 CNETUnloadDll () 함수를 추가합니다.

```
void __fastcall TForm1::FormDestroy(TObject *Sender)
{
        CNETUnloadDll();
}
```

그림 5-52 FormDestroy 함수를 통하여 명시적인 UnloadDll 함수 구현

5.3.5 Borland Delphi 개발자를 위한 안내

Borland Delphi 는 해당 개발 환경 버전인 Delphi 5, Delphi 6 및 Delphi 7, 2006 버전에서 CNETSDK 의 인터페이스 연결 방법이 매우 유사하기 때문에 공통적인 부분으로서 안내를 해드립니다.

전체 버전(Version)의 Delphi 에서 CNETSDK 를 사용하시려면 다음의 절차를 통해 안내 받으시기 바랍니다.

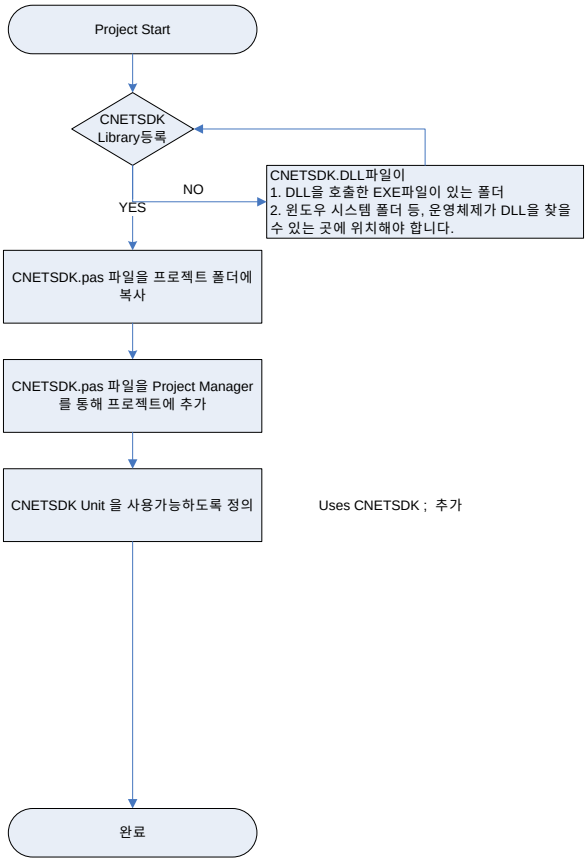


그림 5-53 Borland Delphi 에서 CNETSDK 사용 순서도

Delphi C-NET 응용프로그램을 개발에 필요한 환경인 Borland Delphi 를 실행합니다.

그림 5-54 Borland 社의 Delphi 5 화면

이 안내는 Delphi 5 를 기준으로 설명드리겠습니다. 만약 안내드리는 도중에 다른 개발 환경과 구분이 되어야 할 내용은 별도로 설명 드리겠습니다.

프로젝트 시작 전에 ㈜ 커미조아 CNETSDK 의 Delphi 용 공용 인터페이스 파일을 프로젝트 폴더에 복사합니다. 이 파일은 ㈜ 커미조아 CNETSDK 의 DLL(Dynamic Link Library) 와 고객님의 응용프로그램과의 인터페이스를 정의하여 놓은 파일입니다.

델파이(Delphi)는 명시적으로 프로젝트파일 간의 상호 변환이 필요가 없습니다. 따라서 Delphi 5 나, 6 그리고 7 버전에서 몇가지 기본적인 컴포넌트에 기반한 내용을 제외한 부분들을 그대로 사용할 수 있습니다. ㈜ 커미조아 CNETSDK 에서는 Delphi (5/6/7) 에 대한 풍부한 예제를 제공하고 있습니다.

새로운 프로젝트를 시작하기 위해서, “File” 메뉴의 “New” 에 대한 항목을 클릭하여, 새로운 응용프로그램 개발을 시작합니다.

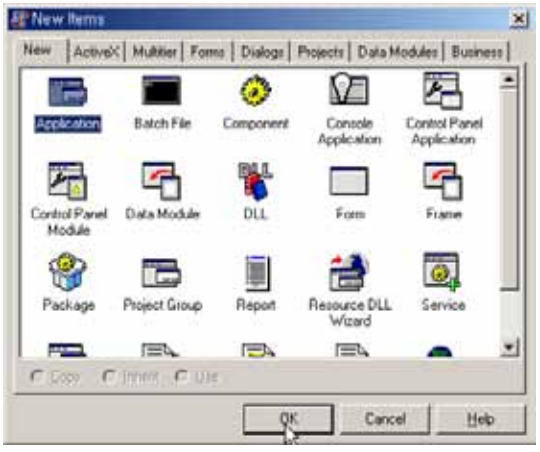


그림 5-55 Delphi 5 의 프로젝트 시작

프로젝트가 시작되면 화면상에 'Form1' 혹은 Delphi IDE 의 Project1 이 나타납니다.

인터페이스 파일을 추가하기 위한 작업으로서 'Project' 메뉴의 'Add to Project' 를 선택합니다.

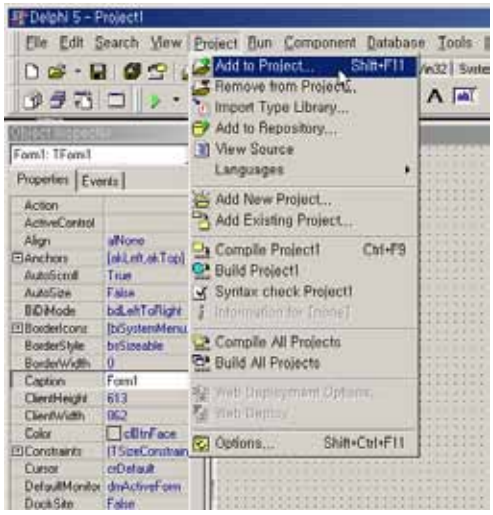


그림 5-56 Delphi 용 인터페이스 파일 추가

(주) 커미조아 CNETSDK 의 공용 인터페이스 정의 파일인 CNETSDK.PAS 파일을 프로젝트에 추가합니다.

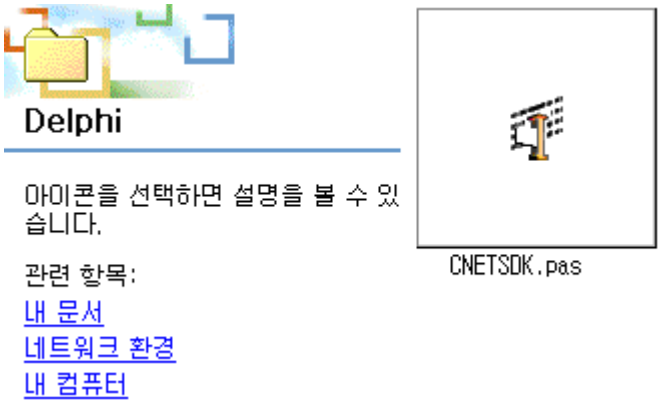


그림 5-57 Delphi 용 인터페이스 파일

Project Manager 를 통해 확인해 보면 CNETSDK.PAS 파일이 프로젝트에 등록된 것을 확인할 수 있습니다.

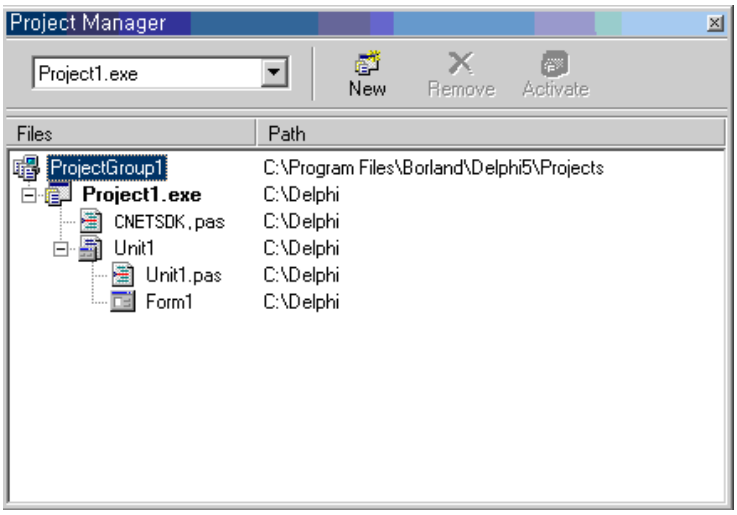


그림 5-58 Delphi 용 인터페이스 파일을 프로젝트 매니저에 추가

보충	Delphi 의 CNETSDK 인터페이스 파일의 사용에 대한 부가 안내
	저희 ㈜ 커미조아의 CNETSDK 의 CNETSDK.Pas 파일은 다른 개발환경(VC++, C++ Builder)와 달리 DLL 의 명시적인 로드와 언로드가 자동으로 이루어집니다. 이것은 델파이가 가지고 있는 Initialization 과 Finalization 을 이용한 것으로 프로젝트에 별다른 LoadDll 함수 호출 없이 자동으로 DLL 이 로드가 됩니다. 또한 응용프로그램 종료시에 UnloadDll 이 명시적으로 되지 않아도, 자동적으로 응용프로그램이 종료시에 UnloadDll 이 실행됩니다.

실제 Unit1.pas 혹은 구현 부의 코드를 에디터를 통해 확인합니다.

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialog
  StdCtrls;

type
  TForm1 = class(TForm)

  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation
////////////////////////////////////////////////////////////////////////
// COMZIOA SDK Library 를 위한 인터페이스 파일을 사용합니다.
uses CNETSDK;
////////////////////////////////////////////////////////////////////////

{$R *.DFM}


end.
```

그림 5-59 uses 구문을 통해 CNETSDK Unit 사용

위와 같이 implementation 부에 **uses 를 통해 라이브러리Unit 을 사용할 수 있도록 반드시 지정**해 주십시오. (상단의 uses에 선언하여도 무방합니다) 이후, 델파이 에서는 다른 개발과 동일하게 DLL 라이브러리를 사용하실 수 있습니다.

5.3.6 Visual Basic 개발자를 위한 안내

Visual Basic 6.0 은 마이크로소프트의 컴포넌트 기반 응용프로그램 개발을 위해 태어난 뛰어난 개발 환경입니다. CNETSDK 는 Visual Basic 6.0 를 완벽히 지원하며, 인터페이스 파일을 제공하고 있습니다. Visual Basic 고객님들께서도 응용프로그램 개발에 편의성을 드리기위해 저희 ㈜ 커미조아는 언제나 노력하고 있습니다.

실제 Visual Basic 6.0 의 프로젝트 시작 전에, 프로젝트 디렉토리에 ㈜ 커미조아 CNETSDK 인터페이스 파일인 CNETSDK.BAS 파일과 CNETSDK 파일 'CNETSDK.DLL' 파일을 반드시 프로젝트 디렉토리에 복사해주시기 바랍니다.

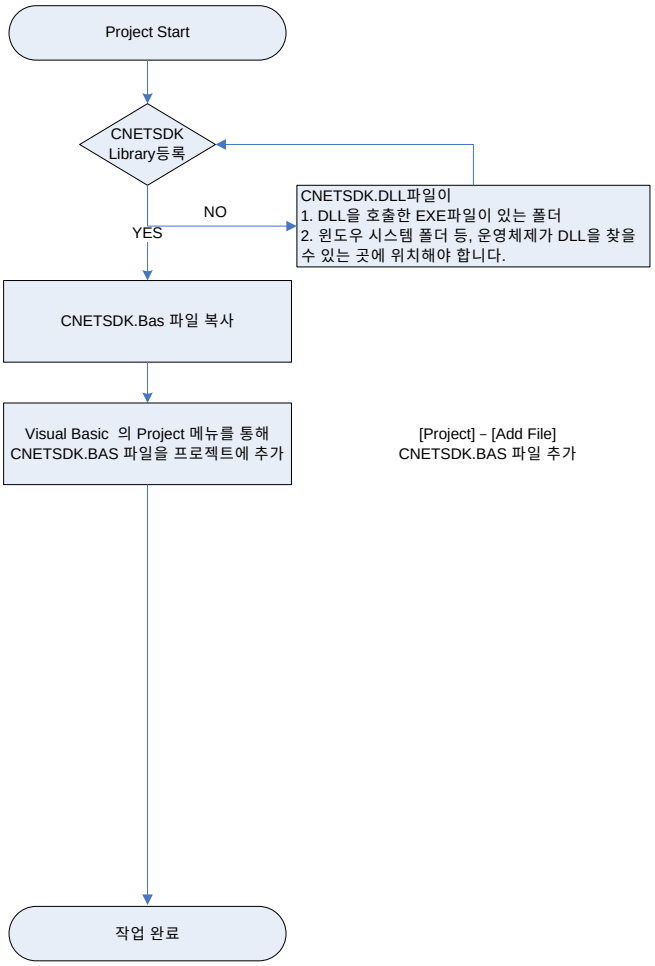


그림 5-60 Visual Basic 에서 CNETSDK 사용 순서도

Visual Basic 을 실행합니다. Visual Basic 이 시작되면 다음과 같은 신규 프로젝트 화면이 표시됩니다.

그림 5-61 새로운 프로젝트 생성

안내	Visual Basic 에 대한 CNETSDK 지원 사항은 어떠합니까?
	CNETSDK 는 .NET 환경의 Visual Basic 까지 지원을 하고 있습니다. 특히 미리 구성된 CNETSDK 인터페이스 파일은 간단히 프로젝트에 추가만 하시면, 바로 CNETSDK 를 사용하실 수 있도록 도움을 드리고 있습니다. 만약, Visual Basic 사용 고객님께서 CNETSDK 사용에 어려움이 있으시다면, 언제든지 저희 ㈜ 커미조아 고객 지원팀으로 연락주시기 바랍니다. 최선을 다해 지원해 드릴 것을 약속드립니다.

신규프로젝트 창에서 'Standard EXE' 를 통해 표준 응용프로그램 개발을 시작합니다. '열기' 버튼을 누릅니다. 만약 이 화면이 나타나지 않으면, 아래의 화면과 같이 'File' 메뉴의 'New Project' 항목을 통해 신규 프로젝트를 시작합니다.

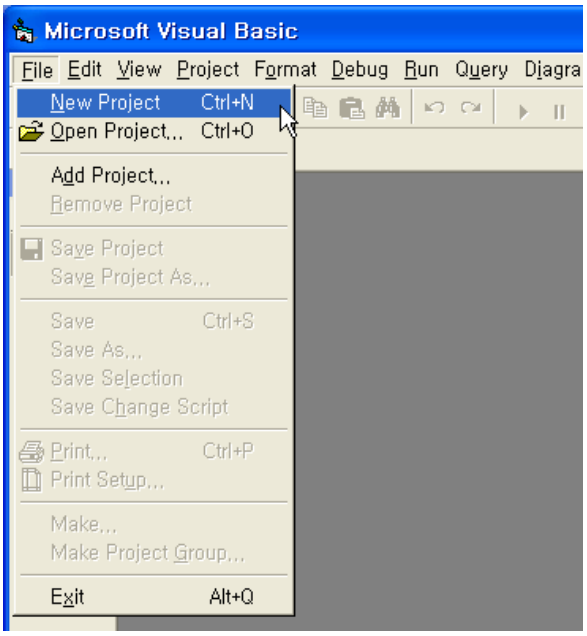


그림 5-62 신규 프로젝트의 시작

시작된 표준 EXE 응용프로그램 개발 메뉴에서 아래 그림과 같이 Project 메뉴를 통해 'Add File…' 을 선택합니다.

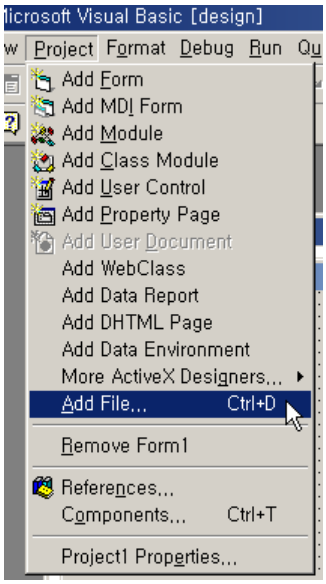


그림 5-63 프로젝트에 인터페이스 파일을 추가하기 위한 과정

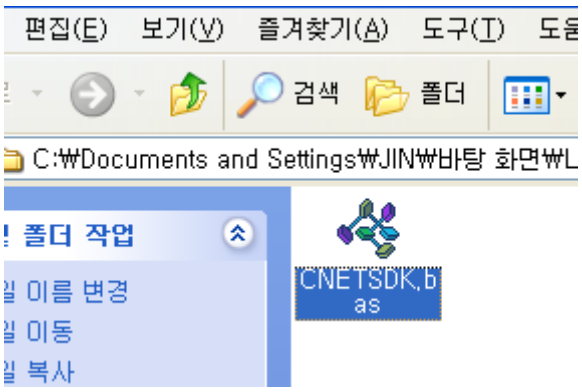


그림 5-64 프로젝트에 추가 대상이 되는 CNETSDK.BAS 파일

CNETSDK.BAS 파일을 프로젝트에 추가해 주시면, 명시적인 라이브러리 로드가 이루어지게 되며, Visual Basic 의 프로젝트에서 함께 사용하실 수 있습니다.

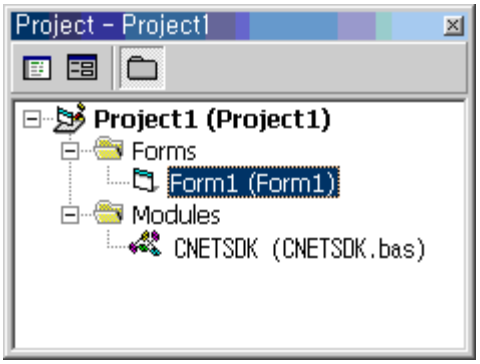


그림 5-65 프로젝트에 모듈로 추가된 CNETSDK.BAS 파일

추가된 인터페이스 파일을 통해 응용프로그램 구현이 가능합니다.

Visual Basic 의 명시적인 CNETSDK 인터페이스에 대한 부가 안내
CNETSDK 에 포함된 Visual Basic 용 인터페이스 파일은 별도의 DLL 라이브러리 (CNETSDK) 의 로드(Load) 및 언로드(Unload) 가 필요 없습니다. 따라서, 고객님의 프로젝트의 Form1 에 추가된 인터페이스 파일을 통하여, 응용프로그램 구현을 바로 시작하실 수 있습니다.

Chapter

6

CNETSDK Introduction

다년간의 보다 강력하고 편리한 라이브러리 기술개발을 통해 자신있게 제공하여 드리는CNETSDK 는 편리한 함수명의 규칙을 통해 사용자 편의성을 극대화 하였습니다. ㈜커미조아 CNETSDK 의 최신 기능과 기술은 결코 흉내낼수 없는 커미조아의 기술입니다. 지금 확인하십시요.

본 장에서는 CNETSDK 가 제공하는 라이브러리 인터페이스에 대한 자세한 설명을 수록하였습니다. 수록하였습니다. CNETSDK 는 자사의 CNETSDK 의 기능을 보다 강력하고 효율적으로 지원할 수 있는 지원할 수 있는 다양한 런타임(Run-time) 인터페이스와 라이브러리의 다양한 기능을 직관적으로 직관적으로 제공합니다. 본 매뉴얼에서는 CNETSDK 에서 제공하는 라이브러리 함수에 대한 설명을 대한 설명을 기능에 따라 그룹별로 수록하였습니다.

6 CNETSDK 소개

6.1 함수의 명명 규칙

CNETSDK 에서 제공하는 모든 함수는 다른 API 함수와 이름이 중복되는 것을 피하기 위하여 아래의 예와 같이 “cmn”이라는 첨두어가 붙습니다.

cmnGnLoadDevice(), cmnGnUnloadDevice(),...

그리고 “cmn” 첨두어 바로 다음에는 해당 함수가 속하는 기능의 그룹을 대표하는 첨두어가 이어집니다. 이렇게 한 이유는 동일한 기능 그룹에 속한 함수들을 쉽게 구분할 수 있고, 함수가 리스트될 때에 일반적으로 알파벳순으로 정렬되므로 동일 기능 그룹에 속한 함수들이 함께 리스트될 수 있도록 하기 위함입니다. 아래는 몇 가지 기능 그룹을 대표하는 첨두어가 적용된 함수들의 예입니다.

○. General Functions (Gn): cmnGnLoadDevice(), cmnGnUnloadDevice(), ...

○. Digital Input Functions (Di): cmnDiGetOne(), cmnDiGetMulti(), ...

○. Digital Output Functions (Do): cmnDoGetOne(), cmnDoGetMulti(), ...

○. Local Channel Functions (Local): cmnLocalDiGetOne(), cmnLocalDoGetOne(), ...

6.2 데이터형 표기

당사의 CNETSDK 인터페이스는 매뉴얼에서 명시한 윈도우 표준 Dynamic Link Library 를 지원하는 어떠한 개발 환경에서도 사용 가능합니다. 하지만 데이터형에 대한 이름은 개발환경에 따라서 서로 다릅니다. 따라서 본 매뉴얼에서는 데이터의 형 표기를 표 6-1 과 같이 통일하여 표기합니다. 이에 대한 각 컴파일러의 대응되는 데이터형 표기는 표 6-1 을 참조하여 사용하시기 바랍니다.

그리고 본 매뉴얼에서는 “[in]”과 “[out]” 표기를 사용하여 매개변수가 함수에 전달되는 것인지 아니면 전달받는 것인지를 명시하였습니다. “[in]”은 함수에 값을 전달함을 의미하고, “[out]은 함수로부터 값을 전달받는다는 것을 의미합니다. 단, 이 표기는 본 매뉴얼에서만 사용되는 것이며, 실제 헤더파일에는 표기되어 있지 않습니다.

Data type	Description	C/C++	VB 6.0	Delphi	C#
VT_EMPTY	반환값이 없는 데이터 표현형	void	-	-	void
VT_HANDLE	운영체제가 특정한 정보를 유지하기 위해서, 메모리에 유지하는 정보 블록에 붙은 고유 번호	void *	Long (ByRef)	THandle	IntPtr
VT_I4	4 바이트 부호있는 정수 표현형	long	Long (ByVal)	LongInt	Int
VT_PI4	4 바이트 부호있는 정수 변수의 주소 값 (포인터) 또는 배열주소 표현형	long *	Long (ByRef)	PLongInt	Int[]
VT_R4	4 바이트 부호있는 실수 표현형	float	Double (ByVal)	Double	Float
VT_PR4	4 바이트 부호있는 실수 변수의 주소 값(포인터) 또는 배열주소 표현형	float *	Double (ByRef)	PDouble	float[]
VT_R8	8 바이트 부호있는 실수 표현형	double	Double (ByVal)	Double	double
VT_PR8	8 바이트 부호있는 실수변수의	double *	Double (ByRef)	PDouble	double[]

Data type	Description	C/C++	VB 6.0	Delphi	C#
	주소 값(포인터) 또는 배열주소 표현형				
VT_STR	선형 메모리 상의 문자열 선두 주소를 지시하는 4 바이트 주소 표현형	char *	String (ByVal)	PChar	String

표 6-1 언어독립적 데이터 표기 및 각 언어별 대응 데이터 형

Chapter

7

C-NET General

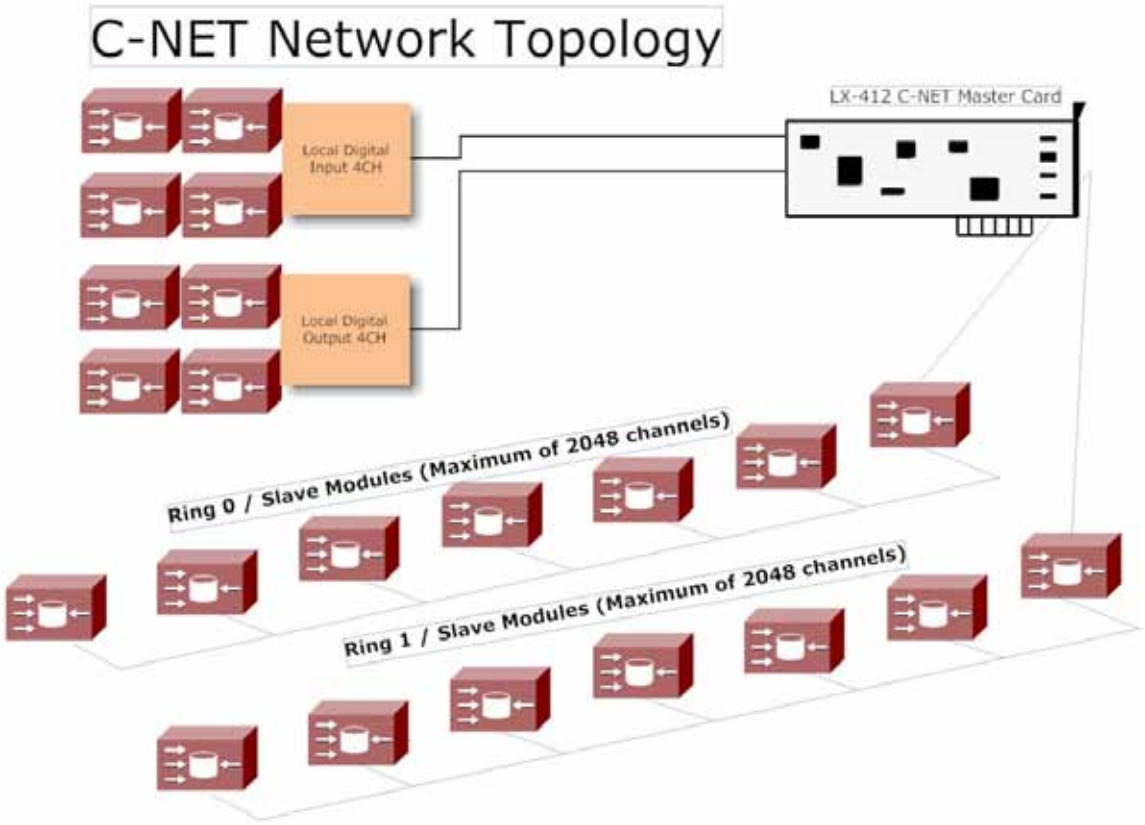
표준 함수와 관계된 C-NET 의 기능을 통해 가장 강력하고 편리한 통신 기반 제어를 하실 수 있습니다. 뛰어난 성능을 기반으로 한 C-NET 라이브러리의 즐거움을 이제 함께 하십시요.

이 단원에서는 장치의 로드/언로드 또는 장치의 초기화와 관련된 가장 일반적인 함수들을 소개합니다. 고객 소개합니다. 고객 여러분들께서는 라이브러리의 초기화와 사용을 위해서는 반드시 본 장을 읽어주시기 읽어주시기 바랍니다. 구성되는 함수에는 통합 라이브러리의 로드 및 언로드, 매개 변수(媒介變數)초기화 변수(媒介變數)초기화 및 디바이스 리셋(Reset)에 대한 내용들로 구성되어 있습니다.

7 C-NET General Functions

7.1 C-NET

㈜커미조아 의 C-NET 은 Master Device 와 Slave Module 을 통해서 네트워크를 구성하며, 마스터 장치의 Ring Connector 의 개수에 따라서 구성 할 수 있는 Ring 네트워크의 개수가 달라집니다. LX412a 의 경우는 2 개의 네트워크를 구성 할 수 있으며, LX412b 의 경우는 한 개의 네트워크를 구성 할 수 있습니다. 한 개의 Ring 네트워크에 추가 할 수 있는 슬레이브 모듈의 개수는 최대 64 개이며, 최대 접점의 수는 2048(64 * 32)개 입니다.



7.2 함수 요약

Summary of Functions
r BOOL CNETLoadDll ([none] VT_EMPTY) CNETSDK 를 응용프로그램의 메모리 공간으로 호출합니다.
r VT_EMPTY CNETUnloadDll([none] VT_EMPTY) CNETSDK 를 응용프로그램의 메모리 공간에서 해제합니다.
r VT_I4 cmnGnLoadDevice ([in] VT_I4 IsResetDevice, [out] VT_PI4 nTotalMasterDevices) 시스템에 설치된 하드웨어 장치를 로드하고 장치를 초기화 합니다.
r VT_EMPTY cmnGnUnloadDevice ([none] VT_EMPTY) 시스템에 설치된 하드웨어 장치를 해제하고, 초기화 합니다.
r VT_I4 cmnGnIsResetDevice([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo,[out] VT_PI4 IsReset) 이 함수는 지정된 하드웨어 장치가 리셋된 상태를 반환합니다.
r VT_I4 cmnGnResetDevice ([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo) 장치를 리셋한 후에 초기화합니다.
r VT_I4 cmnGnDlgSetup([in] BOOL IsEnable, [out] DWORD nLevel, [in] DWORD nDebugType, [in] VT_HANDLE Handler, [in] LPVOID lParam, [in] VT_STR szDebugFileName) 디버깅로그를 출력할 방법을 설정 합니다.
r VT_I4 cmnGnDlgGetLastMessage ([in] VT_I4 nBufferSize, [out] VT_STR szBuffer, [out] VT_PI4 StoredMsg) 디버깅 메시지 큐에 있는 메시지를 반환해 주는 함수 입니다.
r VT_I4 cmnGnStartRingAll (VT_EMPTY) 모든 링에 대해서 통신을 시작합니다.
r VT_I4 cmnGnStartRing([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo, [in] VT_I4 nSlaveIp) 단일 링에 대해서 통신을 시작합니다.
r VT_I4 cmnGnResetRing([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo) 선택한 링과의 통신을 리셋합니다.
r VT_I4 cmnGnStopRing([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo)

7.3 함수 설명

NAME

CNETLoadDll

- 라이브러리(Library) 로드

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r BOOL CNETLoadDll ([none] VT_EMPTY)

DESCRIPTION

CNETSDK 를 응용프로그램의 메모리 공간으로 호출합니다. 이 함수는 전체 프로그램에서 CNETSDK 를 사용하기 위한 수순으로서는 가장 먼저 호출되어야 합니다.

RETURN VALUE

* 이 리턴값은 불 형(Boolean Type) 을 가지고 있습니다.

Value	Meaning
FALSE	DLL 을 로드하는데 실패하였음을 의미합니다.
TRUE	DLL 을 성공적으로 로드하였음을 의미합니다.

NAME

CNETUnloadDll

- 라이브러리(Library) 로드 해제

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_EMPTY CNETUnloadDll ([none] VT_EMPTY)

DESCRIPTION

CNETSDK 를 응용프로그램의 메모리 공간에서 해제합니다. 이 함수가 호출되면 CNETSDK 는 응용프로그램에서 제거 됩니다. 이함수는 전체 프로그램에서 CNETSDK 를 사용을 종료하기 위한 수순으로는 가장 나중에 호출되어야 합니다.

NAME

cmnGnLoadDevice

- 장치 로드

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnLoadDevice ([in] VT_I4 IsResetDevice, [out] VT_PI4 nTotalMasterDevices)

DESCRIPTION

시스템에 설치된 하드웨어 장치를 로드하고 장치를 초기화 합니다. 이 함수는 CNETSDK 의 다른 함수가 호출되기전에 반드시 한번은 수행되어야 합니다. 일반적으로 프로그램의 시작부분에서 수행해주면 됩니다. 본 함수는 시스템에 설치되어 있는 LX412a/b 보드를 초기화 합니다.

장치가 초기화되면 마스터 카드내부에 저장되어 있는 슬래이브 장치의 정보가 모두 초기화 됩니다. 따라서 두개의 서로 다른 응용프로그램에서 장치를 로드할 경우에 주의해주시기 바랍니다.

PARAMETER

▶IsResetDevice : 장치 초기화를 수행할 것인지를 결정합니다.

Value	Meaning
0 또는 FALSE	장치로드시에 장치를 초기값으로 초기화하지 않습니다.
1 또는 TRUE	장치 로드시에 장치를 리셋(Reset)한 후에 장치의 여러가지 속성들을 기본값으로 초기화 합니다.

▶nTotalMasterDevices : 현재 설치되어 있는 Master 장치의 개수를 반환합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnGnUnloadDevice

- 장치로드 해제

INFORMATION
1 General Function
! VC++/BCB/.NET
: Level 1
J 위험 요소 없음

SYNOPSIS

r VT_EMPTY cmnGnUnloadDevice ([none] VT_EMPTY)

DESCRIPTION

시스템에 설치된 하드웨어 장치를 해제하고, 초기화 합니다. 이 함수는 C-NET 디바이스의 사용을 종료하는 시점에서 호출되어야 하며, CNETUnloadDll 보단 앞서 호출 되어야 합니다.

NAME

cmnGnIsResetDevice

- 장치 리셋상태 확인

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnIsResetDevice([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo,[out] VT_PI4 IsReset)

DESCRIPTION

이 함수는 지정된 마스터 디바이스에 속하는 통신 링(Ring)의 리셋여부를 반환합니다.

PARAMETER

▶nDeviceNo : 리셋 상태를 확인할 링이 있는 마스터 디바이스를 지정합니다.
▶nRingNo : 리셋 상태를 확인할 링을 지정합니다.
▶nIsReset : 장치의 리셋 여부를 전달합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnGnResetDevice

- 장치 리셋

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnResetDevice ([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo)

DESCRIPTION

장치를 리셋한 후에 초기화합니다. 일반적으로 cmnGnLoadDevice 함수에서 장치초기화를 수행하므로 초기에 장치를 따로 리셋할 필요가 없습니다. 이 함수는 장치가 비정상적으로 동작하는 상황에서 장치를 초기화 후 다시 초기화하고자 할 때 사용할 수 있습니다.

이 함수를 통해 초기화되면, 검색된 장치의 모든 정보도 함께 초기화 됩니다. 따라서, 슬래이브의 장치 검색을 새로 해주셔야 합니다.

PARAMETER

▶nDeviceNo : 리셋할 링이 있는 마스터 디바이스를 지정합니다.
▶nRingNo : 리셋할 링을 지정합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnGnDlgSetup

- 로그창 설정

INFORMATION
1 General Function
! VC++/BCB/.NET
: Level 1
J 위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnDlgSetup([in] BOOL IsEnable, [out] DWORD nLevel, [in] DWORD nDebugType, [in] VT_HANDLE Handler, [in] LPVOID lParam, [in] VT_STR szDebugFileName)

DESCRIPTION

디버깅로그를 출력할 방법을 설정 합니다. 디버깅 로그는 출력을 위한 여러가지 방법을 지원합니다. 선택된 디버깅 방법에 따라 사용되는 인자가 다르기 때문에 주의를 요합니다. 디버그 메시지 출력에 DEBUG_OUT_CALLBACK 모드를 사용하실 경우에는 cmnGnDlgGetLastMessage 함수를 콜백 함수내부에 추가하셔서 사용하시면 됩니다.

PARAMETER

▶IsEnable : 디버그 다이얼로그의 사용여부를 정합니다.

Value	Meaning
DEBUG_DISABLE	디버그 모드를 사용하지 않습니다.
DEBUG_ENABLE	디버그 모드를 사용합니다.

▶nLevel : 디버깅 레벨을 설정합니다.

Value	Meaning
DEBUG_LEVEL_SUMMERY	기본적인 함수 시작과 끝에 대한 메시지를 출력합니다.
DEBUG_LEVEL_DETAIL	함수의 매개변수를 포함한 메시지를 출력합니다.
DEBUG_LEVEL_DEBUG	라이브러리 인터페이스의 디버그 정보를 출력합니다.
DEBUG_LEVEL_ALL	전체 디버그 메시지를 출력합니다.

▶nDebugType : 디버그 메시지를 표시할 방법을 설정합니다.

Value	Meaning
DEBUG_OUT_WINDOW	윈도우 메시지로 디버그 메시지를 출력합니다.
DEBUG_OUT_LOCALFILE	디버그 메세지를 파일로 출력합니다.
DEBUG_OUT_CONSOLE	디버그 메세지를 콘솔창을 통해서 출력합니다.
DEBUG_OUT_SOCKET	디버그 메시지를 TCP/UDP 를 통해서 출력합니다.
DEBUG_OUT_CALLBACK	디버그 메세지 콜백 함수를 통해서 출력합니다.

▶Handler : 콜백 함수를 호출하기 위한 핸들러입니다. VT_EMPTY (*pDebugHandler) (LPVOID) 형의 함수 포인터를 대입합니다.

▶lParam : 콜백 함수를 호출할 때 파라미터에 넣어줄 값입니다.
▶szDebugFileName : 파일로 디버그 메시지를 저장할 경우 지정하는 파일명입니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.

cmnERR_NONE	수행 성공.

NAME

cmnGnDlgGetLastMessage

- 디버그 메시지를 반환

INFORMATION
1 General Function
! VC++/BCB/.NET
: Level 1
J 위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnDlgGetLastMessage ([in] VT_I4 nBufferSize, [out] VT_STR szBuffer, [out] VT_PI4 StoredMsg)

DESCRIPTION

디버깅 메시지 큐에 있는 디버깅 메시지를 szBuffer 인자를 통해서 VT_STR 형으로 반환해 주는 함수 입니다.

PARAMETER

▶nBufferSize : 메시지를 받을 버퍼의 사이즈를 지정합니다.
▶szBuffer : 메시지를 받을 VT_STR 버퍼를 지정합니다.
▶StoredMsg : 큐에 저장되어 있는 메시지의 수를 반환합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnGnStartRingAll

- 모든 링들의 통신 시작

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnStartRingAll (VT_EMPTY)

DESCRIPTION

모든 링에 속한 슬레이브 모듈과 마스터 디바이스간의 통신을 시작합니다. 디바이스를 로드한 후 반드시 호출되어야 하는 함수 입니다. 이 함수가 호출되어야만 마스터 디바이스와 슬레이브 모듈간의 통신이 시작됩니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnGnStartRing

- 단일링의 통신시작

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnStartRing([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo, [in] VT_I4 nSlaveIp)

DESCRIPTION

선택한 링에 속한 슬레이브 모듈과 마스터 디바이스간의 통신을 시작합니다. 디바이스를 로드한 후 반드시 호출되어야 하는 함수 입니다. 이 함수가 호출되어야만 마스터 디바이스와 슬레이브 모듈간의 통신이 시작됩니다.

PARAMETER

▶nDeviceNo : 통신을 시작할 슬레이브 모듈이 속해 있는 마스터 디바이스를 지정합니다.
▶nRingNo : 통신을 시작할 슬레이브 모듈이 속해 있는 링을 지정합니다.
▶nSlaveIp : 통신을 시작할 슬레이브 모듈의 IP 를 지정합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnGnResetRing

- 단일링 리셋

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnResetRing([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo)

DESCRIPTION

선택한 링의 통신상태를 리셋합니다. 이 함수는 마스터 디바이스와 슬레이브 모듈간에 비정상적인 통신이 이루어질 경우 리셋하고 다시 초기화하고자 할 때 사용할 수 있습니다.

PARAMETER

▶nDeviceNo : 리셋할 링이 속한 마스터 디바이스를 지정합니다.
▶nRingNo : 리셋할 링을 지정합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnGnStopRing

- 단일링의 통신 중단

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnStopRing([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo)

DESCRIPTION

마스터 디바이스와 선택한 링에 속한 슬레이브 모듈간의 통신을 중단합니다.

PARAMETER

▶nDeviceNo : 통신을 중단 할 링이 속한 마스터 디바이스를 지정합니다.
▶nRingNo : 통신을 중단 할 링을 지정합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

Chapter

8

C-NET Communication Control

고속과 저속의 통신제어의 다양한 실제 환경은 많은 이유에 있어, 통신 라인 및 설비의 길이, 배치와 노이즈등이 변수들이 작용하고 있습니다. C-NET 은 현존하는 필드버스 제품 군중 가장 빠른 통신속도를 보장하며, 안전성을 보장합니다.

이 단원에서는 C-NET 의 통신 제어에 대한 함수를 다루고 있습니다. C-NET 은 다양한 통신 지원함수를 지원함수를 통해서 보다 다양한 기능으로 일관성있는 C-NET 통신제어를 할 수 있도록 안내하여 드립니다.

안내하여 드립니다.

8 C-NET Communication Control Functions

8.1 함수 요약

Summary of Functions
r VT_I4 cmnGnSetComSpeed ([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo, [in] VT_I4 nStatus) 선택된 링에 속한 슬레이브 모듈과 마스터 디바이스간의 통신 속도를 설정합니다.
r VT_I4 cmnGnGetComSpeed ([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo, [out] VT_PI4 pStatus) 선택된 링에 속한 슬레이브 모듈과 마스터 디바이스 간의 통신 속도를 확인합니다.
r VT_I4 cmnGnGetComStatus ([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo, [out] VT_PI4 nStatus) 선택된 링의 통신 상태를 확인합니다.
r VT_I4 cmnGnGetGlobalChannelMap ([in] VT_STR szFileName) 기본으로 설정되어 있는 사용자 채널 맵을 szFileName 의 파일명으로 생성합니다.
r VT_I4 cmnGnPutUserChannelMap ([in] VT_STR szFileName) 사용자 정의 채널을 기본 채널맵 설정으로 사용합니다.
r VT_I4 cmnGnGetMasterInfo ([in] VT_I4 nDeviceNo, [out] VT_PI4 pDeviceId) 마스터 디바이스의 종류를 확인 합니다. 인자를 통한 전달값은 LX412b 의 경우는 0xA413, LX412a 의 경우는 0xA412 값을 반환합니다.
r VT_I4 cmnGnGetSlaveInfoAll ([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo, [out] TCMNDevInfo * pDevInfo) 선택된 링 하위의 슬레이브 장치들에 대한 정보를 TCMNDevInfo 구조체를 통해서 전달 받습니다.
r VT_I4 cmnGnGetSlaveInfo ([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo, [in] VT_I4 nDevIp, [out] VT_PI4 pDeviceType, [out] VT_PI4 pUse) 선택된 링 하위의 슬레이브 장치들에 대한 정보를 TCMNDevInfo 구조체를 통해서 전달 받습니다.
r VT_I4 cmnGnGetMasterTotal ([out] VT_PI4 pTotalDeviceNum) 마스터 디바이스의 총 개수를 확인 합니다.
r VT_I4 cmnGnGetSlaveTotal ([out] VT_PI4 pTotalDeviceNum) 슬레이브 디바이스의 총 개수를 확인합니다.
r VT_I4 cmnGnGetDITotal ([out] VT_I4 pTotalDiChannelNum) 디지털 입력 채널(접점)의 총 개수를 확인합니다.
r VT_I4 cmnGnGetDOTotal ([out] VT_PI4 pTotalDOChannelNum) 디지털 출력 채널(접점)의 총 개수를 확인합니다.

8.2 함수 설명

NAME

cmnGnSetComSpeed

- 링의 통신 속도설정

INFORMATION
1 General Function
! VC++/BCB/.NET
: Level 1
J 위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnSetComSpeed ([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo, [in] VT_I4 nStatus)

DESCRIPTION

선택된 링에 속한 슬레이브 모듈과 마스터 디바이스간의 통신 속도를 설정합니다. 통신 속도는 4 단계로 설정 할 수 있으며, 선택된 링의 통신속도에 맞추어 슬레이브 모듈들의 통신 속도도 동일하게 설정되어야 합니다.

PARAMETER

▶nDeviceNo : 통신속도를 설정할 링이 속한 마스터 디바이스를 지정합니다.
▶nRingNo : 통신속도를 설정할 링을 지정합니다.
▶nStatus : 통신 속도를 설정합니다. 입력 할 수 있는 통신 속도는 다음과 같습니다.

Value	Meaning
SPEED_1X	2.5Mbps 로 통신 속도를 설정합니다.
SPEED_2X	5Mbps 로 통신 속도를 설정합니다.
SPEED_3X	10Mbps 로 통신 속도를 설정합니다.
SPEED_4X	20Mbps 로 통신 속도를 설정합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnGnGetComSpeed

- 링의 통신 속도확인

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnGetComSpeed ([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo, [out] VT_PI4 pStatus)

DESCRIPTION

선택된 링에 속한 슬레이브 모듈과 마스터 디바이스 간의 통신 속도를 확인합니다.

PARAMETER

▶nDeviceNo : 통신속도를 확인할 링이 속한 마스터 디바이스를 지정합니다.
▶nRingNo : 통신속도를 확인할 링을 지정합니다.
▶nStatus : 설정되어 있는 통신속도를 전달합니다.

Value	Meaning
SPEED_1X	2.5Mbps
SPEED_2X	5Mbps
SPEED_3X	10Mbps
SPEED_4X	20Mbps

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnGnGetComStatus

- 링의 통신 상태표시

INFORMATION
1 General Function
! VC++/BCB/.NET
: Level 1
J 위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnGetComStatus ([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo, [out] VT_PI4 nStatus)

DESCRIPTION

선택된 링의 통신 상태를 확인합니다. 통신 상태는 nStatus 인자와 통신상태 비트 값을 AND 연산해서 확인 할 수 있습니다.

PARAMETER

▶nDeviceNo : 통신상태를 확인할 링이 속한 마스터 디바이스를 지정합니다.
▶nRingNo : 통신 상태를 확인할 링을 지정합니다.
▶nStatus : 선택된 링의 통신상태를 전달받습니다. 링의 통신상태는 비트 연산을 통해서 확인 할 수 있습니다.

Value	Meaning
RING_DISCONNECTED	링의 통신의 단절된 상태입니다.
RING_CONNECTED	링의 통신이 연결된 상태입니다.
SLAVE_ERROR	슬레이브 모듈과의 통신 에러입니다.
RING_STOP	링의 통신이 중단된 상태입니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnGnGetGlobalChannelMap

- 기본 설정 채널 맵 반환

INFORMATION
1 General Function
! VC++/BCB/.NET
: Level 1
J 위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnGetGlobalChannelMap ([in] VT_STR szFileName)

DESCRIPTION

기본으로 설정되어 있는 사용자 채널 맵을 szFileName 의 파일명으로 생성합니다. 생성된 파일을 cmnGnPutuserChannelMap 함수에 대입해서, 채널맵을 기본 설정상태로 변경하는 것도 가능 합니다.

PARAMETER

▶ szFileName : 생성시킬 파일의 이름을 지정합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnGnPutUserChannelMap

- 사용자 정의 채널맵 대입

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnPutUserChannelMap ([in] VT_STR szFileName)

DESCRIPTION

사용자 정의 채널을 기본 채널맵 설정으로 사용합니다. 기본적으로 장치를 로드 하였을 경우에는 마스터 디바이스 순서, 링의 순서, 슬레이브의 IP 에 의해서 채널의 순서가 정해집니다. 기본적으로 로드된 채널의 순서를 바꿀 경우에 이 함수를 사용합니다.

PARAMETER

▶ szFileName : 사용자 설정 채널정보를 저장한 파일로 채널 설정을 변경합니다.
사용자 정의 파일은 TCMNHeaderInfo 와 TCMNChannelMapInfo 구조체로 구성됩니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnGnGetMasterInfo

- 마스터 장치 확인

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnGetMasterInfo ([in] VT_I4 nDeviceNo, [out] VT_PI4 pDeviceId)

DESCRIPTION

마스터 디바이스의 종류를 확인 합니다. 인자를 통한 전달값은 LX412b 의 경우는 0xA413, LX412a 의 경우는 0xA412 값을 반환합니다.

PARAMETER

▶nDeviceNo : 마스터 디바이스의 종류를 확인할 디바이스의 번호를 넣습니다. 디바이스의 번호는 슬롯의 순서대로 정해집니다.
▶pDeviceId : 디바이스의 종류를 나타내는 ID 값을 반환합니다. LX412a 은 0xA412, LX412b 는 0xA413 의 값을 반환합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnGnGetSlaveInfoAll

- 슬레이브 모듈 정보 획득

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnGetSlaveInfoAll ([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo, [out] TCMNDevInfo * pDevInfo)

DESCRIPTION

선택된 링 하위의 슬레이브 장치들에 대한 정보를 TCMNDevInfo 구조체를 통해서 전달 받습니다. TCMNDevInfo 구조체 내의 nTotalDevNumber 값은 nRingNo 에 의해서 선택된 링에 속한 슬레이브 모듈의 개수입니다.

PARAMETER

▶nDeviceNo : 링의 슬레이브 정보를 얻어올 마스터 디바이스를 번호를 지정합니다. 마스터 디바이스의 번호는 CPU 에 가까운 순으로 정해집니다.
▶nRingNo : 슬레이브의 정보를 얻어올 링의 번호를 지정합니다.
▶pDevInfo : TCMNDevInfo 구조체를 통해서 슬레이브의 정보를 반환 받습니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnGnGetSlaveInfo

- 슬레이브 모듈 정보 획득

INFORMATION
1 General Function
! VC++/BCB/.NET
: Level 1
J 위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnGetSlaveInfo ([in] VT_I4 nDeviceNo, [in] VT_I4 nRingNo, [in] VT_I4 nDevIp, [out] VT_PI4 pDeviceType, [out] VT_PI4 pUse)

DESCRIPTION

선택된 단일 슬레이브에 대한 타입과 사용여부를 확인 합니다.

PARAMETER

▶nDeviceNo : 정보를 얻을 슬레이브가 속한 마스터 디바이스의 번호를 지정합니다.
▶nRingNo : 정보를 얻을 슬레이브가 속한 링을 지정합니다.
▶nDevIp : 정보를 얻을 슬레이브의 IP 를 지정합니다.
▶pDeviceType : 지정한 슬레이브 모듈의 디바이스 타입을 반환합니다. 반환값에 따른 슬레이브 모듈의 구분은 다음과 같습니다.

Value	Meaning
0	Digital 32 Output Slave Module
1	Digital 8 Input 8 Output Slave Module
2	Digital 16 Input 16 Output Slave Module
4	Digital 32 Input Slave Module

▶pUse : 지정한 슬레이브 모듈의 사용여부를 반환합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnGnGetMasterTotal

- 마스터의 개수를 확인

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnGetMasterTotal ([out] VT_PI4 pTotalDeviceNum)

DESCRIPTION

마스터 디바이스의 총 개수를 확인 합니다. 사용자의 컴퓨터 혹은 산업용 디바이스의 PCI 슬롯에 작착되어 있는 LX412a/b 보드의 개수를 반환합니다.

PARAMETER

▶pTotalDeviceNum : 현재 설치되어 있는 마스터 디바이스의 개수를 반환합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnGnGetSlaveTotal

- 슬레이브의 개수를 확인

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnGetSlaveTotal ([out] VT_PI4 pTotalDeviceNum)

DESCRIPTION

슬레이브 디바이스의 총 개수를 확인합니다. 사용자의 컴퓨터 혹은 산업용 디바이스에 장착되어 있는 모든 슬레이브 모듈의 개수를 반환합니다.

PARAMETER

▶pTotalDeviceNum : 현재 활성화 되어 있는 슬레이브의 개수를 반환합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnGnGetDITotal

- 입력채널의 개수를 확인

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnGetDITotal ([out] VT_I4 pTotalDiChannelNum)

DESCRIPTION

디지털 입력 채널(접점)의 총 개수를 확인합니다. 사용자의 컴퓨터 혹은 산업용 디바이스에 장착되어 있는 모든 디지털 입력 채널(접점)의 개수를 반환합니다.

PARAMETER

▶pTotalDiChannelNum : 현재 활성화 되어 있는 Digital Input Channel 의 개수를 반환합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnGnGetDOTotal

- 출력 채널의 개수를 확인

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnGnGetDOTotal ([out] VT_PI4 pTotalDOChannelNum)

DESCRIPTION

디지털 출력 채널(접점)의 총 개수를 확인합니다. 사용자의 컴퓨터 혹은 산업용 디바이스에 장착되어 있는 모든 디저털 출력 채널(접점)의 개수를 반환합니다.

PARAMETER

▶pTotalDOChannelNum : 현재 활성화 되어 있는 Digital Output Channel 의 개수를 반환합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

Chapter

9

Master Universal Digital I/O

통신 제어의 Slave 모듈과 별도로 Master 장치에는 별도의 범용 디지털 입출력 장치를 포함하고 있습니다. 이 범용 디지털 입출력 포트를 통해 마스터 카드상에서 제공하는 범용 디지털 포트를 편리하게 사용하실 수 있습니다.

이 장에서는 C-NET 마스터 카드가 제공하는 범용 디지털 입출력 포트에 대해서 설명합니다. C-NET 마스터 C-NET 마스터 카드는 각각 4 채널의 기본 입출력 포트를 가지고 있으며, 별도의 마스터 상 15 핀 연결 15 핀 연결 단자를 통해 자유롭게 사용하실 수 있습니다.

9 Master Universal Digital I/O Function

9.1 함수 요약

Summary of Functions
r VT_I4 cmnLocalDiSetInputLogic ([in] VT_I4 GlobalChNo, [in] VT_I4 InputLogic) 로컬디바이스의 채널(접점)에 대한 디지털 입력 로직을 설정합니다.
r VT_I4 cmnLocalDiGetInputLogic ([in] VT_I4 GlobalChNo, [out] VT_PI4 InputLogic) 로컬디바이스의 채널(접점)에 대한 디지털 입력 로직을 확인합니다.
r VT_I4 cmnLocalDoSetOutputLogic ([in] VT_I4 GlobalChNo, [in] VT_I4 OutputLogic) 로컬디바이스의 채널(접점)에 대한 디지털 출력 로직을 설정 합니다.
r VT_I4 cmnLocalDoGetOutputLogic ([in] VT_I4 GlobalChNo, [in] VT_PI4 OutputLogic) 로컬디바이스의 채널(접점)에 대한 디지털 출력 상태를 확인합니다.
r VT_I4 cmnLocalDiGetOne([in] VT_I4 GlobalChNo, [out] VT_PI4 InputStatus) 로컬디바이스의 채널(접점)에 대한 디지털 입력 상태를 확인합니다.
r VT_I4 cmnLocalDiGetMulti([in] VT_I4 IniChannel, [in] VT_I4 NumChannel, [out] VT_PI4 InputStates) 로컬디바이스의 다중채널(접점)에 대한 입력 상태를 확인합니다.
r VT_I4 cmnLocalDoPutOne([in] VT_I4 GlobalChNo, [out] VT_I4 OutState) 로컬 디바이스의 채널(접점)에 대한 출력 상태를 설정합니다.
r VT_I4 cmnLocalDoPutMulti([in] VT_I4 IniChannel, [in] VT_I4 NumChannel, [out] VT_I4 OutStates) 로컬 디바이스의 다중채널(접점)에 대한 출력 상태를 설정합니다.
r VT_I4 cmnLocalDoGetOne([in] VT_I4 GlobalChNo,[out] VT_PI4 OutState) 로컬 디바이스의 채널(접점)에 대한 출력 상태를 확인합니다.
r VT_I4 cmnLocalDoGetMulti([in] VT_I4 IniChannel, [in] VT_I4 NumChannel, [out] VT_PI4 OutStates) 로컬디바이스의 다중채널(접점)에 대한 출력 상태를 확인합니다.

9.2 함수 설명

NAME

cmnLocalDiSetInputLogic

- 디지털 입력 로직 설정

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnLocalDiSetInputLogic ([in] VT_I4 GlobalChNo, [in] VT_I4 InputLogic)

DESCRIPTION

로컬디바이스의 채널(접점)에 대한 디지털 입력 로직을 설정합니다. 마스터 디바이스의 채널(접점)은 선형적으로 관리되며, 0 번에 가까운 번호를 가진 장치에 속한 채널(접점)일수록 빠른 순서의 채널(접점)을 가지게 됩니다. 마스터 디바이스의 순서는 CPU 에 가까운 순으로 결정됩니다.

PARAMETER

▶GlobalChNo : 로직을 설정할 로컬디바이스의 채널을 지정합니다.
▶InputLogic : 로직을 설정합니다. 설정 가능한 로직은 아래와 같습니다.

Value	Meaning
cmnLOGIC_A	Normal Open
cmnLOGIC_B	Normal Close

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnLocalDiGetInputLogic

- 디지털 입력 채널로직 확인

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnLocalDiGetInputLogic ([in] VT_I4 GlobalChNo, [out] VT_PI4 InputLogic)

DESCRIPTION

로컬디바이스의 채널(접점)에 대한 디지털 입력 로직을 확인합니다. 마스터 디바이스의 채널(접점)은 선형적으로 관리되며, 0 번에 가까운 번호를 가진 장치에 속한 채널(접점)일수록 빠른 순서의 채널(접점)을 가지게 됩니다. 마스터 디바이스의 순서는 CPU 에 가까운 순으로 결정됩니다.

PARAMETER

▶GlobalChNo : 로직의 상태를 확인할 로컬디바이스의 채널을 지정합니다.
▶InputLogic : 설정되어 있는 입력 로직을 반환합니다. 로직의 상태는 다음과 같습니다.

Value	Meaning
cmnLOGIC_A	Normal Open
cmnLOGIC_B	Normal Close

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnLocalDoSetOutputLogic

- 디지털 출력 채널로직 설정

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnLocalDoSetOutputLogic ([in] VT_I4 GlobalChNo, [in] VT_I4 OutputLogic)

DESCRIPTION

로컬디바이스의 채널(접점)에 대한 디지털 출력 로직을 설정 합니다. 마스터 디바이스의 채널(접점)은 선형적으로 관리되며, 0 번에 가까운 번호를 가진 장치에 속한 채널(접점)일수록 빠른 순서의 채널(접점)을 가지게 됩니다.
마스터 디바이스의 순서는 CPU 에 가까운 순으로 결정됩니다.

PARAMETER

▶GlobalChNo : 로직을 설정할 로컬디바이스의 채널을 지정합니다.
▶OutputLogic : 지정한 로컬디바이스 채널의 로직을 설정합니다. 반환값에 따른 로직은 아래와 같습니다.

Value	Meaning
cmnLOGIC_A	Normal Open
cmnLOGIC_B	Normal Close

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnLocalDoGetOutputLogic

- 디지털 출력 채널 로직 확인

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnLocalDoGetOutputLogic ([in] VT_I4 GlobalChNo, [in] VT_PI4 OutputLogic)

DESCRIPTION

로컬디바이스의 채널(접점)에 대한 디지털 출력 상태를 확인합니다. 마스터 디바이스의 채널(접점)은 선형적으로 관리되며, 0 번에 가까운 번호를 가진 장치에 속한 채널(접점)일수록 빠른 순서의 채널(접점)을 가지게 됩니다. 마스터 디바이스의 순서는 CPU 에 가까운 순으로 결정됩니다.

PARAMETER

▶GlobalChNo : 로직을 설정할 로컬디바이스의 채널을 지정합니다.
▶OutputLogic : 지정한 로컬디바이스 채널의 로직을 반환합니다. 반환값은 다음과 같습니다.

Value	Meaning
cmnLOGIC_A	Normal Open
cmnLOGIC_B	Normal Close

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnLocalDiGetOne

- 단일 디지털 입력 채널상태 확인

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnLocalDiGetOne([in] VT_I4 GlobalChNo, [out] VT_PI4 InputStatus)

DESCRIPTION

로컬디바이스의 채널(접점)에 대한 디지털 입력 상태를 확인합니다.

PARAMETER

▶GlobalChNo : 로컬 디바이스의 채널을 지정합니다.
▶InputStatus : 지정한 로컬 디바이스 채널의 상태를 반환합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnLocalDiGetMulti

- 다중 디지털 입력 채널상태 확인

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnLocalDiGetMulti([in] VT_I4 IniChannel, [in] VT_I4 NumChannel, [out] VT_PI4 InputStates)

DESCRIPTION

로컬디바이스의 다중채널(접점)에 대한 입력 상태를 확인합니다. 시작채널(접점)부터 설정된 범위의 채널(접점)에 대한 Digital 입력 상태값을 얻습니다.

PARAMETER

▶IniChannel : 한번에 읽어들일 채널의 가장 앞채널을 지정합니다.
▶NumChannel : 한번에 읽어들일 채널의 수를 지정합니다. 최대 32 개까지 가능합니다.
▶InputStates : 최대 32 개의 채널의 상태를 32 비트 정수 형으로 반환합니다. 원하는 채널의 상태는 비트 연산으로 확인이 가능 합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnLocalDoPutOne

- 단일 출력 채널 상태 설정

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnLocalDoPutOne([in] VT_I4 GlobalChNo, [out] VT_I4 OutState)

DESCRIPTION

로컬 디바이스의 채널(접점)에 대한 출력 상태를 설정합니다.

PARAMETER

▶GlobalChNo : 로컬 디바이스의 채널을 지정합니다.
▶OutState : 출력 상태를 설정합니다. 출력 상태는 로직의 영향을 받습니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnLocalDoPutMulti

- 다중 출력 채널 상태 설정

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnLocalDoPutMulti([in] VT_I4 IniChannel, [in] VT_I4 NumChannel, [out] VT_I4 OutStates)

DESCRIPTION

로컬 디바이스의 다중채널(접점)에 대한 출력 상태를 설정합니다. 시작채널(접점)부터 설정된 범위의 채널(접점)에 대한 Digital 출력 상태를 설정합니다.

PARAMETER

▶IniChannel : 한번에 읽어들일 채널의 가장 앞채널을 지정합니다.
▶NumChannel : 한번에 상태를 설정할 채널의 수를 지정합니다. 최대 32 개까지 가능합니다.
▶OutStates : 최대 32 개의 채널의 상태를 32 비트 정수 형으로 설정합니다. 원하는 채널의 상태는 비트 연산으로 확인이 가능 합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnLocalDoGetOne

- 단일 디지털 출력 채널상태 확인

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnLocalDoGetOne([in] VT_I4 GlobalChNo,[out] VT_PI4 OutState)

DESCRIPTION

로컬 디바이스의 채널(접점)에 대한 출력 상태를 확인합니다.

PARAMETER

▶GlobalChNo : 로컬디바이스의 채널을 지정합니다.
▶OutState : 출력 상태를 설정합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnLocalDoGetMulti

- 다중 디지털 출력 채널상태 확인

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnLocalDoGetMulti([in] VT_I4 IniChannel, [in] VT_I4 NumChannel, [out] VT_PI4 OutStates)

DESCRIPTION

로컬디바이스의 다중채널(접점)에 대한 출력 상태를 확인합니다. 시작채널(접점)부터 설정된 범위의 채널(접점)에 대한 Digital 출력 상태값을 얻습니다.

PARAMETER

▶IniChannel : 한번에 읽어들일 채널들의 시작 채널을 지정합니다.
▶NumChannel : IniChannel 에서 시작 하여 한번에 읽어들일 채널의 수를 지정합니다. 최대 32 까지 지정가능 합니다.
▶OutState : 최대 32 개의 채널의 상태를 32 비트 정수 형으로 반환합니다. 원하는 채널의 상태는 비트 연산으로 확인이 가능 합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

Chapter

10

Slave Universal Digital I/O

C-NET 제품군에서 최대 4096 채널까지 가능한 초고속 필드퍼스 시스템인 C-NET 은 국내외는 물론 해외까지 그 기능과 안전성을 입증받고 있습니다. 개별 Ring 당 2048 채널, 2 개의 Ring 경우에 최대 4096 채널의 슬래이브 채널을 지원하는 고성능 C-NET 시스템으로 보다 저렴한 비용의 필드버스 시스템을 구축하십시요.

이 장에서는 C-NET 제품군을 통해서 제어가능한 슬래이브 모듈을 위한 함수를 안내하고 있습니다. C-NET 있습니다. C-NET 슬래이브 모듈은 많은 노력을 투자하지 않고 손쉽게 탈장착이 가능하며, 모듈 구성과 모듈 구성과 배선이 매우 용이 합니다. 각종 로직 설정 함수와 입출력 함수를 통해 슬래이브 모듈 제어에 모듈 제어에 편리함을 함께하십시요.

10 Slave Universal Digital I/O Function

10.1 함수 요약

Summary of Functions
r VT_I4 cmnDiSetInputLogic ([in] VT_I4 GlobalChNo, [in] VT_I4 InputLogic) 디지털 입력 채널(접점)에 대한 입력 로직을 설정합니다.
r VT_I4 cmnDiGetInputLogic ([in] VT_I4 GlobalChNo, [in] VT_PI4 InputLogic) 디지털 입력 채널(접점)에 대한 입력 로직을 확인합니다.
r VT_I4 cmnDoSetOutputLogic ([in] VT_I4 GlobalChNo, [in] VT_I4 OutputLogic) 디지털 출력 채널(접점)에 대한 출력 로직을 설정합니다.
r VT_I4 cmnDoGetOutputLogic ([in] VT_I4 GlobalChNo, [in] VT_PI4 OutputLogic) 디지털 출력 채널(접점)에 대한 출력 로직을 확인합니다.
r VT_I4 cmnDiGetOne([in] VT_I4 GlobalChNo, [out] VT_I4 InputStatus) 단일 디지털 입력 채널(접점)에 대한 디지털 입력 상태를 확인합니다.
r VT_I4 cmnDiGetMulti([in] VT_I4 IniChannel, [in] VT_I4 nChannels, [out] VT_PI4 InputStates) 다중채널(접점)에 대한 입력 상태를 확인합니다.
r VT_I4 cmnDoPutOne([in] VT_I4 GlobalChNo, [in] VT_I4 OutState) 단일 디지털 출력 채널(접점)에 대한 출력 상태를 설정합니다.
r VT_I4 cmnDoPutMulti([in] VT_I4 IniChannel, [in] VT_I4 nChannels , [in] VT_I4 OutStates) 다중채널(접점)에 대한 출력 상태를 설정합니다.
r VT_I4 cmnDoGetOne([in] VT_I4 GlobalChNo, [out] VT_PI4 OutState) 단일 디지털 출력 채널(접점)에 대한 출력 상태를 확인합니다.
r VT_I4 cmnDoGetMulti([in] VT_I4 IniChannel, [in] VT_I4 nChannels, [out] VT_PI4 OutStates) 다중채널(접점)에 대한 출력 상태를 확인합니다.

10.2 함수 설명

NAME

cmnDiSetInputLogic

- 디지털 입력 로직 설정

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnDiSetInputLogic ([in] VT_I4 GlobalChNo, [in] VT_I4 InputLogic)

DESCRIPTION

디지털 입력 채널(접점)에 대한 입력 로직을 설정합니다.

PARAMETER

▶GlobalChNo : 입력 로직을 설정할 채널을 지정합니다.
▶InputLogic : 지정한 채널의 입력 로직을 설정 합니다. 설정값은 아래와 같습니다.

Value	Meaning
cmnLOGIC_A	Normal Open
cmnLOGIC_B	Normal Close

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnDiGetInputLogic

- 디지털 입력 로직 확인

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnDiGetInputLogic ([in] VT_I4 GlobalChNo, [in] VT_PI4 InputLogic)

DESCRIPTION

디지털 입력 채널(접점)에 대한 입력 로직을 확인합니다.

PARAMETER

▶GlobalChNo : 입력 로직을 확인할 채널을 지정합니다.
▶InputLogic : 지정한 채널의 입력 로직을 반환 합니다. 반환값은 아래와 같습니다.

Value	Meaning
cmnLOGIC_A	Normal Open
cmnLOGIC_B	Normal Close

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnDoSetOutputLogic

- 디지털 출력 로직 설정

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnDoSetOutputLogic ([in] VT_I4 GlobalChNo, [in] VT_I4 OutputLogic)

DESCRIPTION

디지털 출력 채널(접점)에 대한 출력 로직을 설정합니다.

PARAMETER

▶GlobalChNo : 출력 로직을 설정할 채널을 지정합니다.
▶ OutputLogic : 지정한 채널의 출력 로직을 설정 합니다. 설정값은 아래와 같습니다.

Value	Meaning
cmnLOGIC_A	Normal Open
cmnLOGIC_B	Normal Close

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnDoGetOutputLogic

- 디지털 출력 로직 확인

INFORMATION
1 General Function
! VC++/BCB/.NET
: Level 1
J 위험 요소 없음

SYNOPSIS

r VT_I4 cmnDoGetOutputLogic ([in] VT_I4 GlobalChNo, [in] VT_PI4 OutputLogic)

DESCRIPTION

디지털 출력 채널(접점)에 대한 출력 로직을 확인합니다.

PARAMETER

▶GlobalChNo : 출력 로직을 확인할 채널을 지정합니다.
▶ OutputLogic : 지정한 채널의 출력 로직을 반환 합니다. 반환값은 아래와 같습니다.

Value	Meaning
cmnLOGIC_A	Normal Open
cmnLOGIC_B	Normal Close

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnDiGetOne

- 단일 디지털 입력 채널상태 확인

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnDiGetOne([in] VT_I4 GlobalChNo, [out] VT_I4 InputStatus)

DESCRIPTION

단일 디지털 입력 채널(접점)에 대한 디지털 입력 상태를 확인합니다.

PARAMETER

▶GlobalChNo : 로컬 디바이스의 채널을 지정합니다.
▶InputStatus : 지정한 로컬 디바이스 채널의 상태를 반환합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnDiGetMulti

- 다중 디지털 입력 채널상태 확인

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnDiGetMulti([in] VT_I4 IniChannel, [in] VT_I4 nChannels, [out] VT_PI4 InputStates)

DESCRIPTION

다중채널(접점)에 대한 입력 상태를 확인합니다. 시작채널(접점)부터 설정된 범위의 채널(접점)에 대한 Digital 입력 상태값을 얻습니다.

PARAMETER

▶IniChannel : 한번에 읽어들일 채널의 가장 앞채널을 지정합니다.
▶NumChannel : 한번에 읽어들일 채널의 수를 지정합니다. 최대 32 개까지 가능합니다.
▶InputStates : 최대 32 개의 채널의 상태를 32 비트 정수 형으로 반환합니다. 원하는 채널의 상태는 비트 연산으로 확인이 가능 합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnDoPutOne

- 단일 출력 채널 상태 설정

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnDoPutOne([in] VT_I4 GlobalChNo, [in] VT_I4 OutState)

DESCRIPTION

단일 디지털 출력 채널(접점)에 대한 출력 상태를 설정합니다.

PARAMETER

▶GlobalChNo : 로컬 디바이스의 채널을 지정합니다.
▶OutState : 출력 상태를 설정합니다. 출력 상태는 로직의 영향을 받습니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnDoPutMulti

- 다중 출력 채널 상태 설정

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnDoPutMulti([in] VT_I4 IniChannel, [in] VT_I4 nChannels , [in] VT_I4 OutStates)

DESCRIPTION

다중채널(접점)에 대한 출력 상태를 설정합니다. 시작채널(접점)부터 설정된 범위의 채널(접점)에 대한 Digital 출력 상태를 설정합니다.

PARAMETER

▶IniChannel : 한번에 읽어들일 채널의 가장 앞채널을 지정합니다.
▶NumChannel : 한번에 상태를 설정할 채널의 수를 지정합니다. 최대 32 개까지 가능합니다.
▶OutStates : 최대 32 개의 채널의 상태를 32 비트 정수 형으로 설정합니다. 원하는 채널의 상태는 비트 연산으로 확인이 가능 합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnDoGetOne

- 단일 디지털 출력 채널상태 확인

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnDoGetOne([in] VT_I4 GlobalChNo, [out] VT_PI4 OutState)

DESCRIPTION

단일 디지털 출력 채널(접점)에 대한 출력 상태를 확인합니다.

PARAMETER

▶GlobalChNo : 로컬디바이스의 채널을 지정합니다.
▶OutState : 출력 상태를 반환합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

NAME

cmnDoGetMulti

- 다중 디지털 출력 채널상태 확인

INFORMATION	
1	General Function
!	VC++/BCB/.NET
:	Level 1
J	위험 요소 없음

SYNOPSIS

r VT_I4 cmnDoGetMulti([in] VT_I4 IniChannel, [in] VT_I4 nChannels, [out] VT_PI4 OutStates)

DESCRIPTION

다중채널(접점)에 대한 출력 상태를 확인합니다. 시작채널(접점)부터 설정된 범위의 채널(접점)에 대한 Digital 출력 상태값을 얻습니다.

PARAMETER

▶IniChannel : 한번에 읽어들일 채널들의 시작 채널을 지정합니다.
▶NumChannel : IniChannel 에서 시작 하여 한번에 읽어들일 채널의 수를 지정합니다. 최대 32 까지 지정가능 합니다.
▶OutState : 최대 32 개의 채널의 상태를 32 비트 정수 형으로 반환합니다. 원하는 채널의 상태는 비트 연산으로 확인이 가능 합니다.

RETURN VALUE

Value	Meaning
음수	수행 실패. 자세한 내용은 '에러 코드' 편을 참고 합니다.
cmnERR_NONE	수행 성공.

Chapter

11

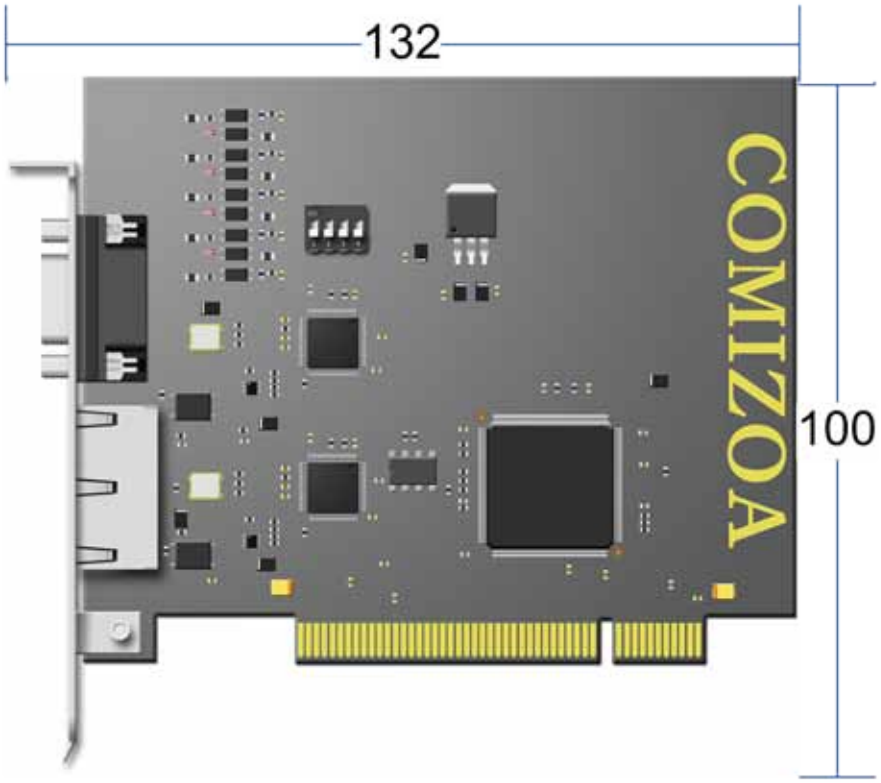
APPENDIX

C-NET 은COMI-LINK 와 같은 강력한 디버그 및 테스트 프로그램을 제공합니다.COMI-LINK 가 제공하는 다양한 기능은C-NET 라이브러리를 통해 개발이 가능합니다. 고급 사용자 인터페이스와 뛰어난 성능을 가진 COMI-LINK 는 무료로 배포되는C-NET 의고 수준 응용프로그램입니다.

이 단원에서는 전체 매뉴얼에서 다루지 못한 구성된 제품의 규격(Dimension)과 C-NET 디버그 및 테스트 및 테스트 응용프로그램인 COMI-LINK 에 대해서 안내합니다.

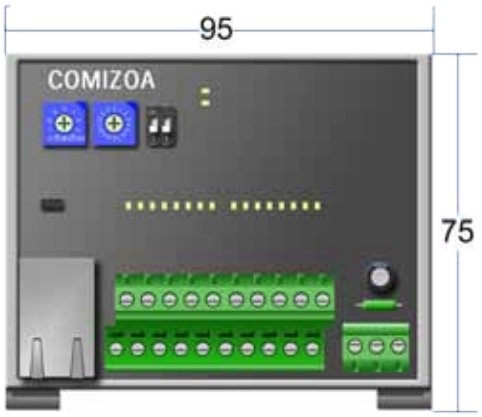
11 APPENDIX

11.1 Dimension of COMI-LX412a/b

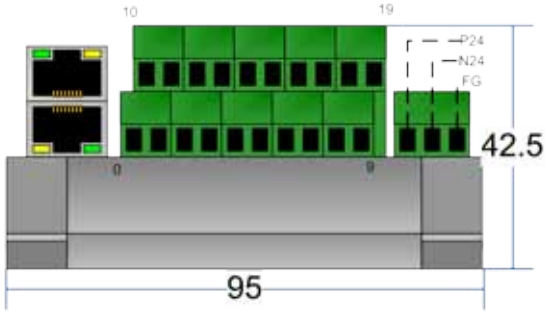


Dimension	(W)132 x (H)100

11.2 Dimension of C-NET Slave modules

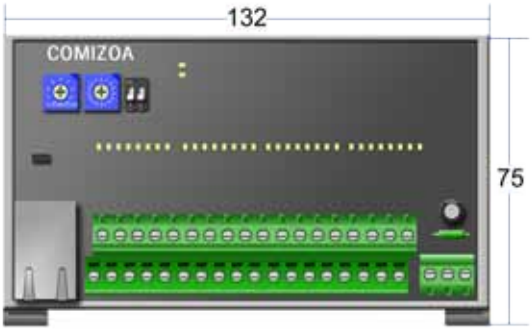


Dimension	(W)95 x (H)75

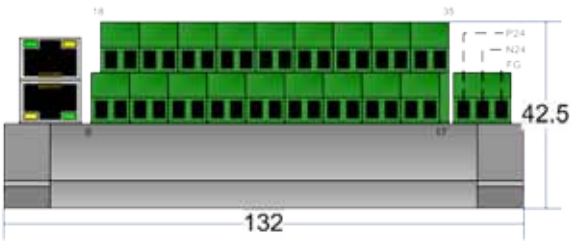


Dimension	(W)95 x (H)42.5

cnD series 16channel module



Dimension	(W)132 x (H)75

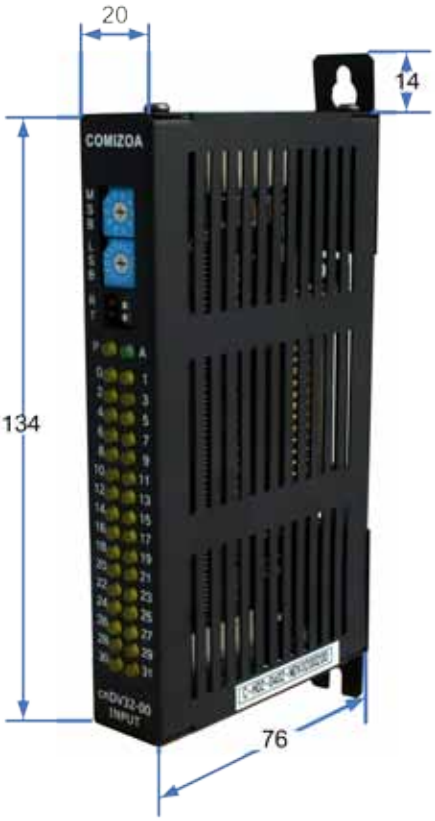


Dimension	(W)132 x (H)42.5

cnD series 32channel module

Dimension	(W)20 x (H)99 x (D)76

cnD series 16channel module



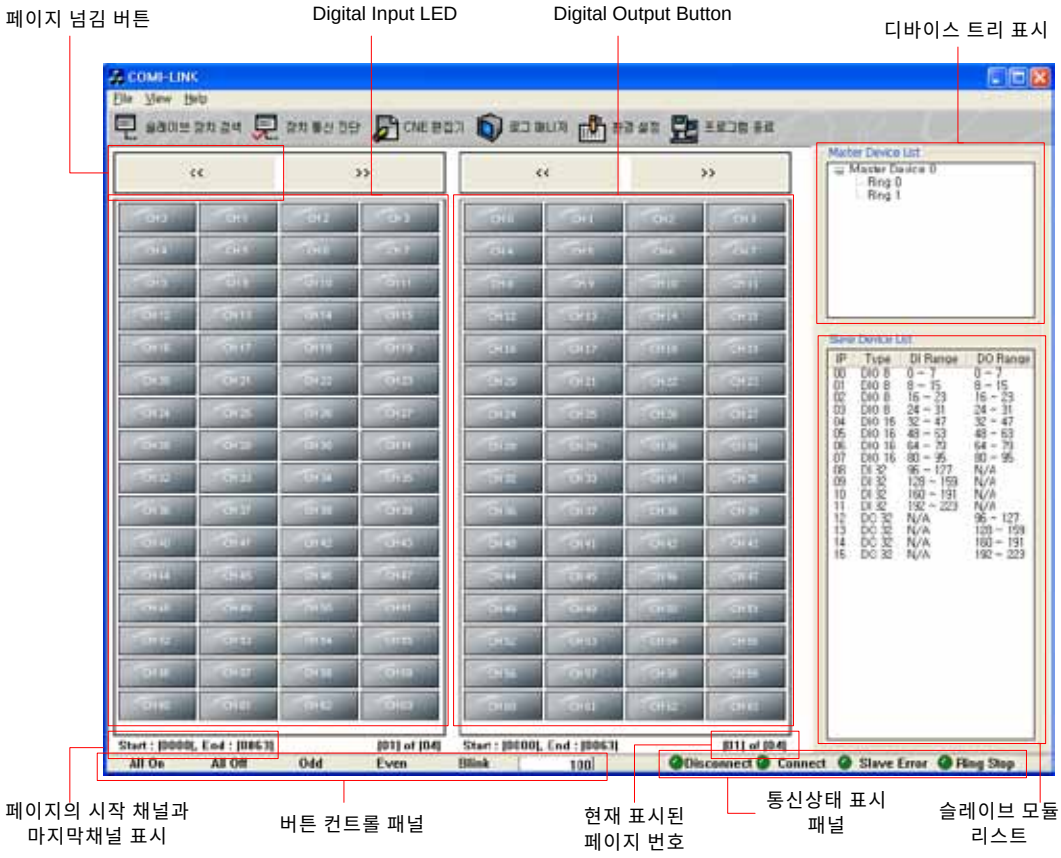
Dimension	(W)20 x (H)134 x (D)76

cnDV series 32channel module

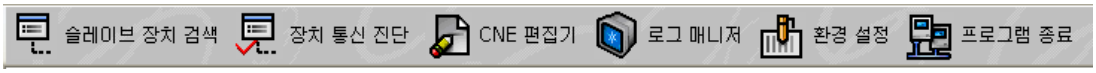
11.3 COMI-LINK

COMI-LINK 는 (주) 커미조아의 디바이스넷 제어 전용 유틸리티 프로그램으로서 C-NET 구입시 무상으로 제공됩니다. 본 프로그램은 C-NET 의 전반적인 기능을 사용 할 수 있도록 되어 있으며, 별도의 코딩없이도 장치를 테스트 하는 것이 가능합니다.

11.3.1 COMI-LINK UI



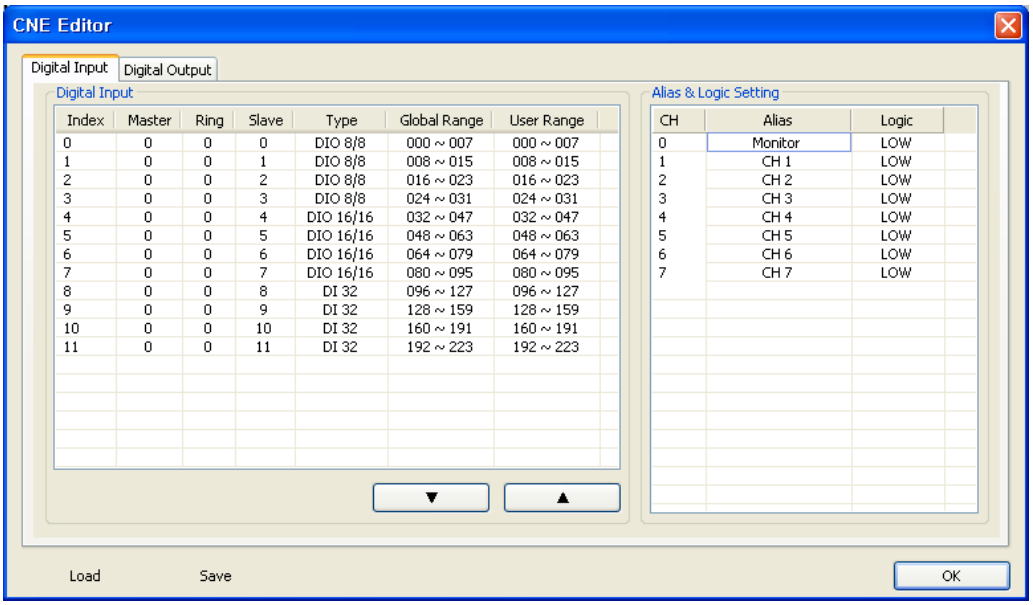
11.3.2 TOOL BAR



●슬레이브 장치 검색 : 현재 연결되 있는 장치들을 검색 합니다. 검색된 결과는 오른쪽에 있는 디바이스 트리 표시 창과 디바이스 모듈 리스트에 표시됩니다.

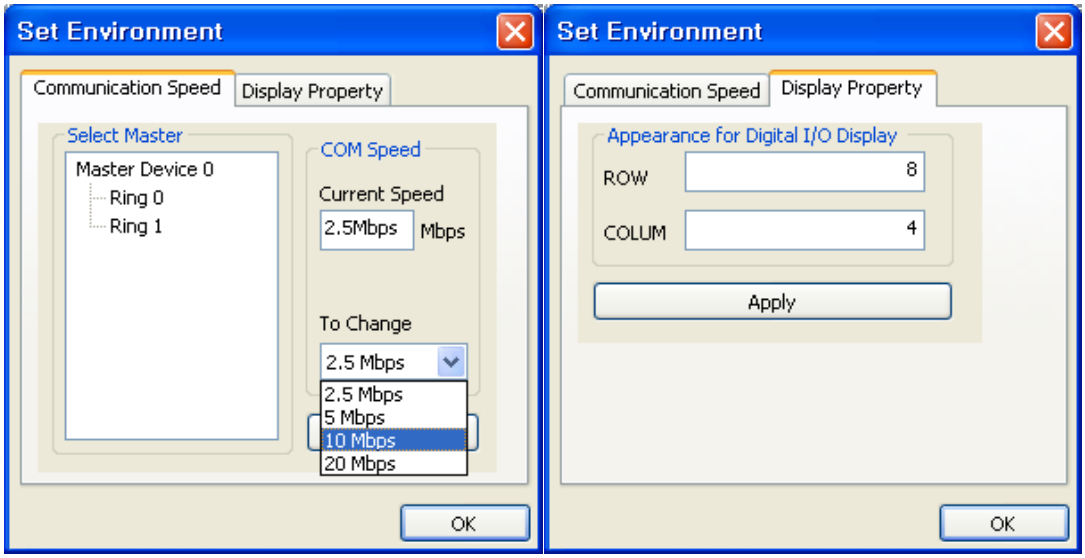
●장치 통신 진단 : 선택된 링의 통신 상태를 나타 냅니다.

●CNE 편집기 : 슬레이브 모듈의 순서, Alias 를 변경합니다. 통합적으로 관리되는 채널의 순서를 변경할 수 있으며 각 채널별로 별도의 이름을 지정하여 구분하기 쉽게 합니다. 변경한 내용을 파일로 저장하거나 로드하여 사용할 수 있습니다.

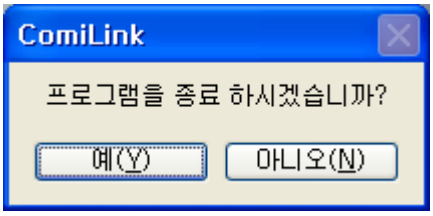


●로그매니져 : 디버그 메시지를 표시 해주는 기능을 합니다.

●환경 설정 : 통신 속도 및 화면에 보여줄 버튼의 개수를 조정합니다.



●프로그램 종료 : 프로그램을 종료 합니다.



11.3.3 COMI-LINK 실행

COMI-LINK 를 실행하면 현재 장착되어 있는 마스터 장치의 개수와 '0'번 마스터의 '0'번링의 통신 속도를 표시 한 후 자동으로 슬레이브 모듈을 검색합니다.

장치가 로드 되면 오른쪽의 상단의 트리 컨트롤에 마스터 장치와 링을 보여주는 트리가 표시 됩니다.

트리 컨트롤에 표시된 링을 더블클릭 하시면 더블클릭 된 링에 연결된 슬레이브 모듈들이 하단의 리스트에 표시 됩니다.

Slave Device List

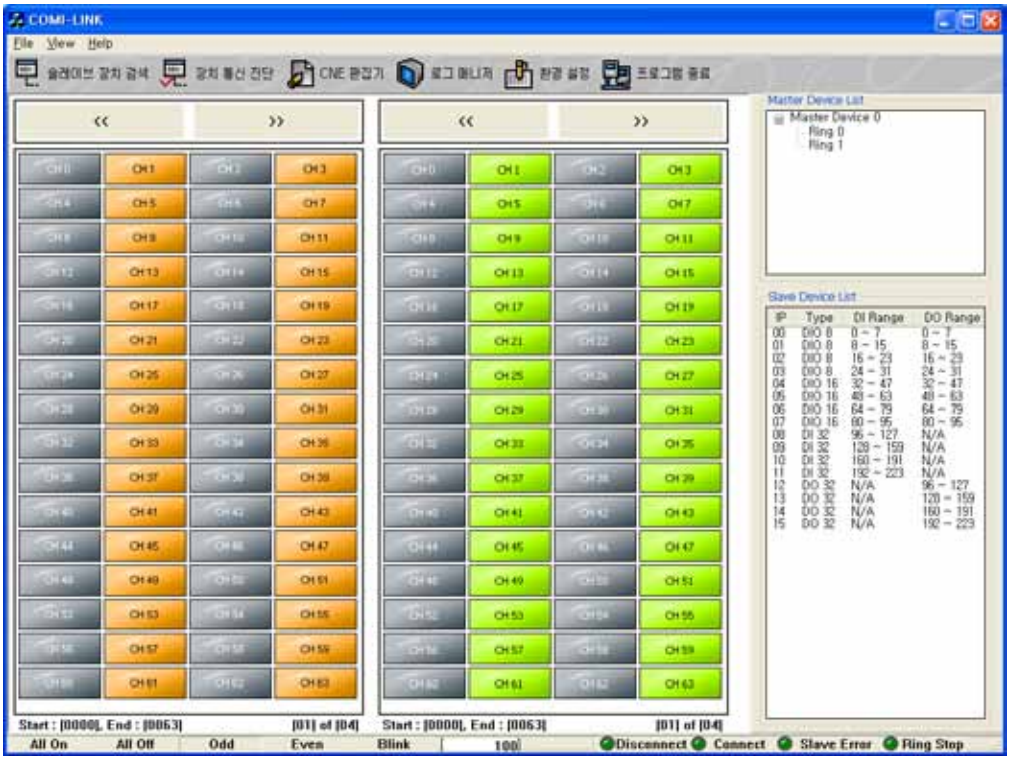
IP	Type	DI Range	DO Range
00	DIO 8	0 ~ 7	0 ~ 7
01	DIO 8	8 ~ 15	8 ~ 15
02	DIO 8	16 ~ 23	16 ~ 23
03	DIO 8	24 ~ 31	24 ~ 31
04	DIO 16	32 ~ 47	32 ~ 47
05	DIO 16	48 ~ 63	48 ~ 63
06	DIO 16	64 ~ 79	64 ~ 79
07	DIO 16	80 ~ 95	80 ~ 95
08	DI 32	96 ~ 127	N/A
09	DI 32	128 ~ 159	N/A
10	DI 32	160 ~ 191	N/A
11	DI 32	192 ~ 223	N/A
12	DO 32	N/A	96 ~ 127
13	DO 32	N/A	128 ~ 159
14	DO 32	N/A	160 ~ 191
15	DO 32	N/A	192 ~ 223

리스트에는 각 슬레이브 모듈의 IP, Device Type, Digital Input Range, Digital Output Range 에 대한 정보들이 표시 됩니다.

리스트에서 표시된 장치를 더블 클릭 하면 별도의 창으로 슬레이브 모듈의 정보를 확인하고 컨트롤 하는 것이 가능 합니다.

모듈이 로드된 상태에서는 COMI-LIN 의 버튼을 통하여 Digital Input 을 감시 하거나 Digital Output 출력을 내보낼 수 있습니다.

Status Bar 의 전체 버튼 컨트롤 패널을 이용하여 전체 버튼을 컨트롤 하는 것도 가능 합니다.



각 버튼의 기능은 다음과 같습니다.

●All On : 모든 Digital Output Channel 을 출력으로 설정합니다.

●All Off : 모든 Digital Output Channel 을 출력을 정지합니다.

●Odd : 홀수번의 Digital Ouput Channel 을 출력으로 설정합니다..

●Even : 짝수번의 Digital Output Channel 을 출력으로 설정합니다.

●Blink : Blink Interval 에 설정된 값을 주기로 Digital Output Channel 의 All On, All Off 혹은 Odd, Even 의 상태를 변경합니다.

11.4 에러코드

ERROR CODE	VALUE	MEANING
cmnERR_NONE	0	수행 성공
cmnERR_INVALID_PARAMETER	-190	인자가 무효함
cmnERR_MEM_ALLOC_FAIL	-290	메모리 할당 실패
cmnERR_GLOBAL_MEM_FAIL	-292	공유메모리 할당 실패
cmnERR_GN_LOAD_DEVICE_FAIL	-10000	디바이스 로드 실패
cmnERR_GN_UNLOAD_DEVICE_FAIL	-10001	디바이스 언로드 실패
cmnERR_GET_SLAVE_INFO_FAIL	-10002	슬레이브 정보 획득 실패
cmnERR_GET_SLAVE_INFO_ALL_FAIL	-10003	모든 슬레이브 정보 획득 실패
cmnERR_GET_GLOBAL_CHANNEL_MAP_FAIL	-10004	전역 채널 맵 획득 실패
cmnERR_GN_GET_GLOBAL_CHANNEL_MAP_FAIL	-10005	전역 채널 맵 획득 실패
cmnERR_GN_PUT_USER_CHANNEL_MAP_FAIL	-10006	사용자 채널 맵 설정 실패
cmnERR_GN_DLG_SETUP_FAIL	-10007	디버그 모드 설정 실패
cmnERR_LOCAL_DI_GET_ONE_FAIL	-10008	Local digital input channel 상태 획득 실패
cmnERR_LOCAL_DI_GET_MULTI_FAIL	-10009	Local digital input channels 상태 획득 실패
cmnERR_LOCAL_DO_PUT_ONE_FAIL	-10010	Local digital output channel 설정 실패
cmnERR_LOCAL_DO_PUT_MULTI_FAIL	-10011	Local digital output channels 설정 실패
cmnERR_LOCAL_DO_GET_ONE_FAIL	-10012	Local digital output channel 상태 획득 실패
cmnERR_LOCAL_DO_GET_MULTI_FAIL	-10013	Local digital output channels 상태 획득 실패
cmnERR_DI_GET_ONE_FAIL	-10014	Digital input channel 상태 획득 실패
cmnERR_DI_GET_MULTI_FAIL	-10015	Digital input channels 상태 획득 실패
cmnERR_DO_PUT_ONE_FAIL	-10016	Digital output channel 설정 실패
cmnERR_DO_PUT_MULTI_FAIL	-10017	Digital output channels 설정 실패
cmnERR_DO_GET_ONE_FAIL	-10018	Digital output channel 상태 획득 실패
cmnERR_DO_GET_MULTI_FAIL	-10019	Digital output channels 상태 획득 실패
cmnERR_GN_START_RING_ALL_FAIL	-10020	전체 링들의 통실 시작 실패
cmnERR_GN_START_RING_FAIL	-10021	선택된 링과의 통신 시작 실패
cmnERR_GN_RESET_RING_FAIL	-10022	리셋 링 실패
cmnERR_GN_STOP_RING_FAIL	-10023	링과의 통신 중단 실패
cmnERR_GN_RESET_DEVICE_FAIL	-10024	디바이스 리셋 실패
cmnERR_GN_ISRESET_DEVICE_FAIL	-10025	디바이스 리셋 여부 확인 실패
cmnERR_GN_GET_COM_STATUS_FAIL	-10026	통신상태 확인 실패
cmmERR_GN_GET_COM_SPEED_FAIL	-10027	통신속도 확인 실패
cmnERR_GN_SET_COM_SPEED	-10028	통신속도 설정 실패
cmnERR_GN_DLG_GET_LAST_MESSAGE_FAIL	-10029	마지막 에러 메시지 획득 실패
cmnERR_GN_DLG_GET_NO_MORE_MESSAGE_FAIL	-10030	에러메세지 버퍼에 더 이상 메시지가 없음
cmnERR_GN_DLG_GET_BUFFER_SMALL_FAIL	-10031	에러메세지 버퍼가 에러메세지 보다 작음

11.5 함수색인

TEST & MEASUREMENT & AUTOMATION / COMIZOA

CNETSDK Manual

매뉴얼 자료 번호 : 4.0.0.2

㈜커미조아
http://www.comizoa.com
Tel) 042 – 936 – 6500~6
Fax) 042 – 936 – 6507