



## Beckhoff TwinCAT

The **W**indows **C**ontrol and **A**utomation **T**echnology

**TwinCAT R3IO & ADS**  
**User Manual**

Ver 2.0



## 내용

1. Introduction.....	4
2. TwinCAT 설치.....	5
3. TwinCAT System Service .....	11
3.1. TwinCAT System Properties.....	11
4. TwinCAT System 구성 및 실행.....	14
4.1. TwinCAT System Manager.....	15
4.2. Remote IO 연결 .....	19
4.2.1. EtherCAT Master Driver 설치 .....	19
4.2.2. EtherCAT Remote IO 연결.....	20
4.2.3. Remote IO 테스트 .....	25
4.3. TwinCAT IO Task 설정.....	27
4.3.1. Additional Task.....	27
4.3.2. Insert Variable .....	28
4.3.3. Add Variable Type.....	29
4.3.4. Auto start 체크.....	30
4.3.5. Link.....	31
4.3.6. TwinCAT Run Mode 실행 .....	34
4.3.7. TwinCAT Auto Boot Setting.....	34
5. ADS Connectivity .....	35
5.1. TwinCAT Device Concept.....	35
5.2. ADS Libraries.....	36
5.3. ADS Device Identification.....	37
5.3.1. ADS-AmsNetId.....	37
5.3.2. ADS-Port.....	39
5.3.3. Index - Group/Offset.....	40
5.4. ADS API 호출별 분류.....	41
5.4.1. Synchronous.....	41
5.4.2. Asynchronous.....	42
5.4.3. Notification.....	43
5.5. ADS Return Codes.....	44
5.5.1. Global Error Codes.....	44
5.5.2. General ADS Error Codes .....	45
5.6. TwinCAT ADS-OCX.....	47
5.6.1. ADS-OCX 의 접근 형태별 분류.....	47
5.6.2. ADS OCX 프로그래밍 (Visual Basic).....	48

5.7.	TwinCAT ADS-DLL .....	54
5.7.1.	ADS-DLL 프로그래밍 방법 .....	55
5.7.2.	샘플 소스 .....	57
5.8.	TwinCAT ADS.NET .....	65
5.8.1.	ADS.NET 프로그래밍 방법 .....	65
5.8.2.	샘플소스 .....	66
6.	EtherCAT 통신 진단 .....	73
6.1.	EtherCAT 프레임 상태 확인 .....	73
6.2.	EtherCAT Master 상태 확인 .....	74
6.3.	EtherCAT Slave 상태 확인 .....	76
6.4.	EtherCAT 토폴로지 .....	78

## **TwinCAT I/O & ADS Connectivity 교육 교재**

### **1. Introduction**

TwinCAT R3IO & ADS User Manual 에서는 Windows NT 계열의 사용자 어플리케이션을 작성하는데 필요한 라이브러리 파일 및 헤더 파일, DLL 파일을 제시하며, TwinCAT I/O 인터페이스 함수에 대해서 설명한다. 또한 TwinCAT System Manager에서 Interface Card 및 Bus Coupler, Bus Terminal 를 구성하는 방법과 Windows NT 계열의 사용자 어플리케이션에서 접근할 Bus Terminal의 논리 주소와 실제 Bus Terminal 물리 주소의 맵핑 방법에 대해서도 설명한다. 더 나아가 TwinCAT 시스템에서 지원하는 ADS 통신 설정 방법에 대해서도 설명한다.

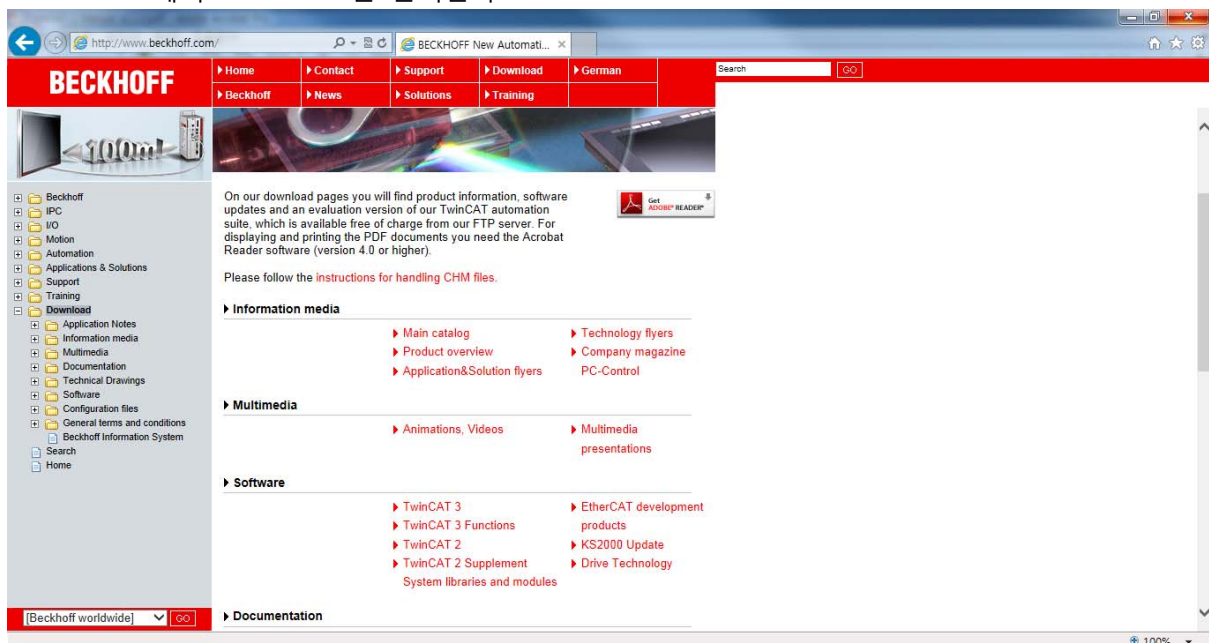
본 매뉴얼은 Beckhoff 에서 개발한 산업용 이더넷 통신 시스템인 EtherCAT 을 중심으로 설명하며, TwinCAT 시스템 구성은 EtherCAT Bus Coupler 인 EK1100 를 기준으로 한다.

## 2. TwinCAT 설치

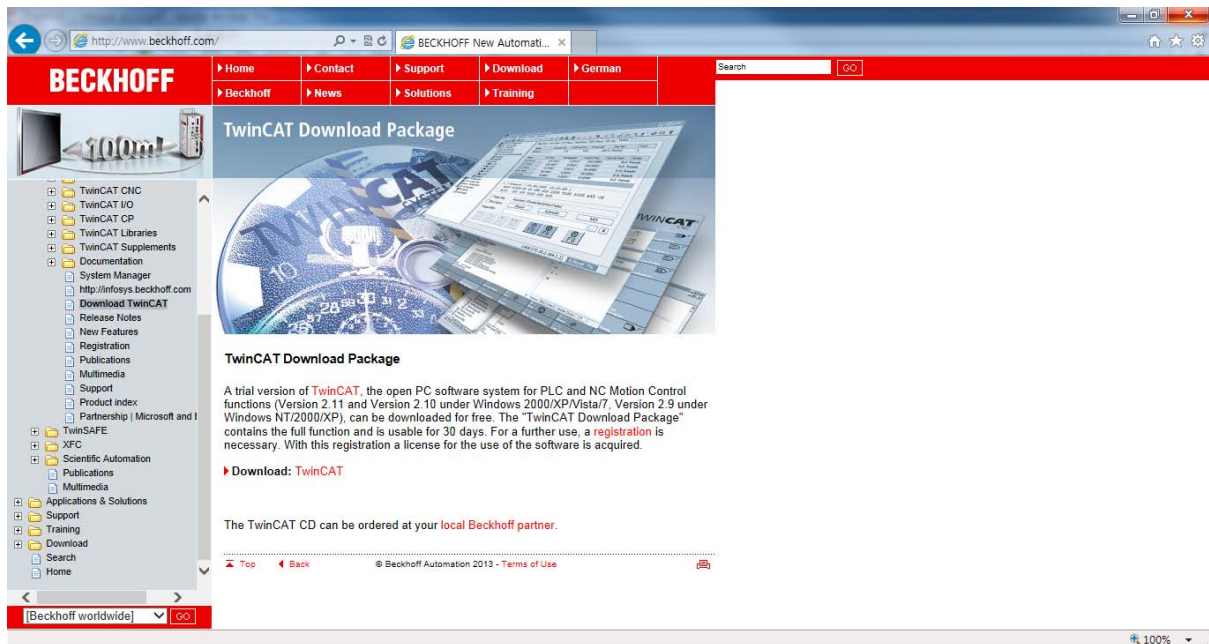
- [www.beckhoff.com](http://www.beckhoff.com) 에서 TwinCAT 소프트웨어를 다운로드 받을 수 있다.
- 페이지 상단의 'Download'를 선택한다.



- Software 에서 TwinCAT 2 를 클릭한다.



- Download 부분의 TwinCAT 을 클릭한다.



- 이메일 주소와 각 정보들을 기입한 후 하단에서 원하는 TwinCAT version 과 build 를 선택한다. [Registration] 버튼을 선택하면 기입한 이메일 주소로 TwinCAT 소프트웨어를 다운받을 수 있는 링크 메일이 발송된다.

Please fill in the following formular in order to download the "TwinCAT Download Package"

(▶) required fields

Mr. ▼

First Name:

Name: ▶

Company: ▶

Address: ▶

ZIP Code: ▶

City: ▶

Country: ▶

eMail: ▶


Phone: ▶

Fax:

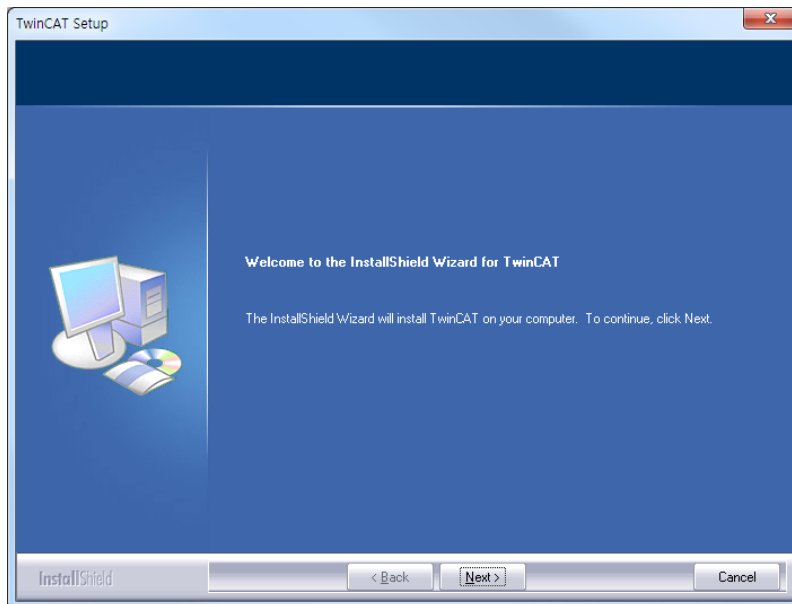
Version: ☒ TwinCAT 2.11 R3, Build 2234  
☐ TwinCAT 2.11 x64 Engineering, Build 2234  
☐ previous Version

Please select the TwinCAT version!

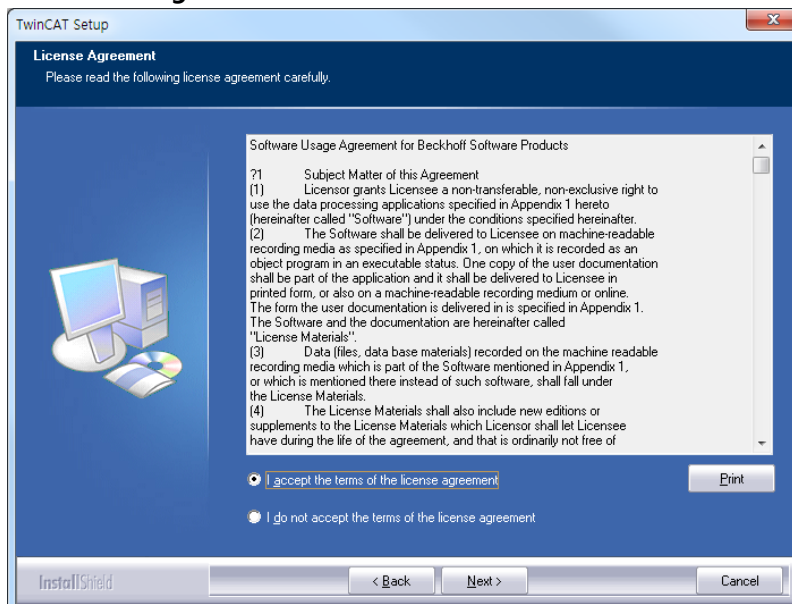
- 설치파일을 더블 클릭한다.

 tcat\_2110\_2232

- Next 를 클릭한다.



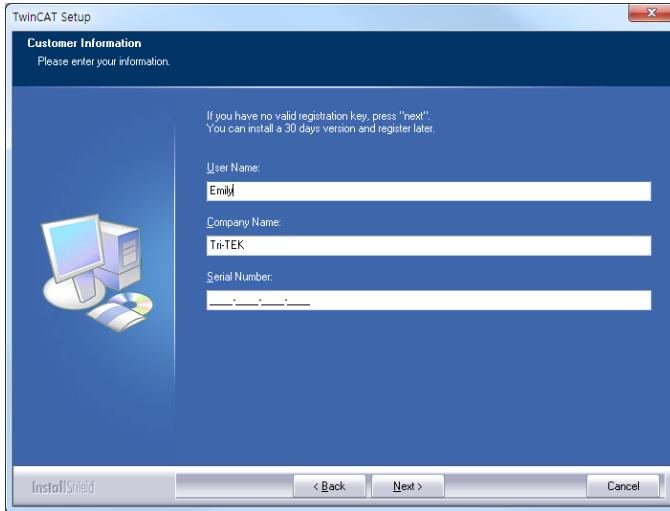
- License Argument



### • Customer Information

사용자명과 회사명을 기입한다. (30 일 데모버전 사용시 Serial Number 는 생략한다.)

Serial 정보는 사용자 시스템의 System ID 를 Beckhoff 에 보낸 후 확인 절차를 거쳐 부여 받아야 한다.



### • Select Installation Level

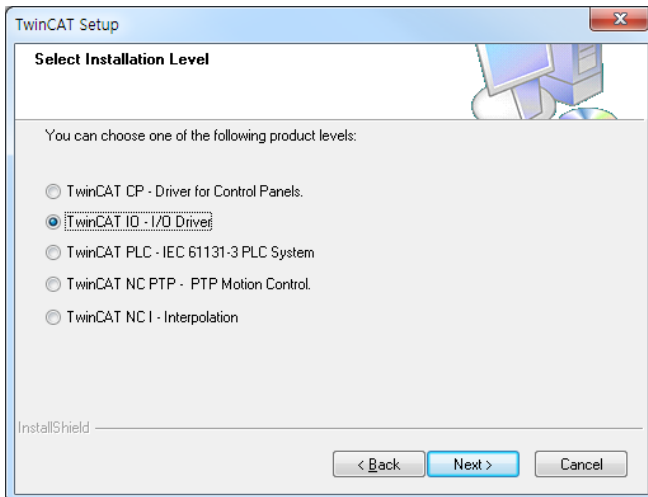
TwinCAT R3IO 를 사용하기 위해서 TwinCAT IO 레벨을 선택한다.

TwinCAT CP: Control Panels 에서 TwinCAT 시스템을 사용할 수 있도록 한다. 기본적인 TwinCAT 라우터 기능만 제공한다.

TwinCAT IO: TwinCAT R3 IO Task 를 사용할 수 있다.

TwinCAT PLC: IEC 61131-3 PLC 언어로 제어 프로그램을 작성할 수 있는 툴을 제공한다.

TwinCAT NC PTP, NC I: 모션 제어에 필요한 기능들을 포함하고 있다.





## • Select Components

TwinCAT System 에서 사용할 구성요소를 선택한다.

TwinCAT Scope View: 실시간 그래프를 통해 변수의 동작을 모니터링 할 수 있는 프로그램

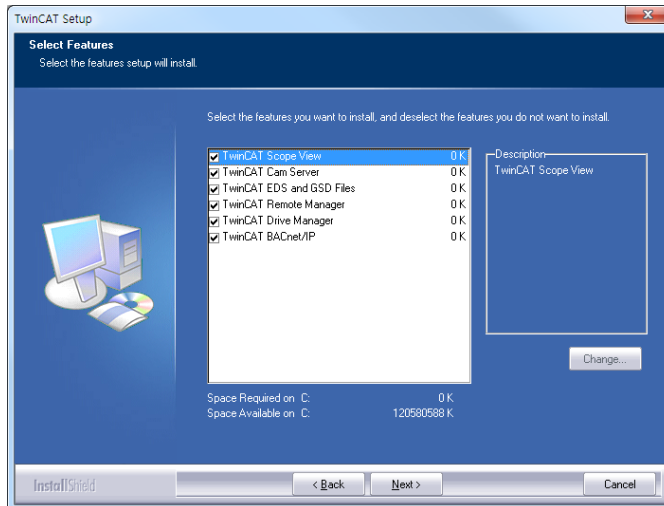
TwinCAT Cam Server: Motion 제어에서 Camming 을 사용할 경우 선택

TwinCAT EDS and GSD Files: DeviceNet, Profibus 용 EDS/GSD 파일을 사용할 경우 선택

TwinCAT Remote Manager: 다른 버전의 TwinCAT 이 설치된 Target PC 의 관리를 할 수 있는 툴

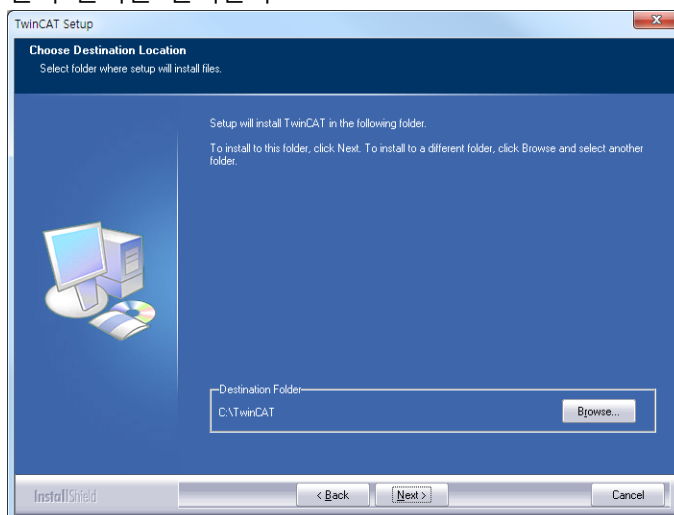
TwinCAT Device Manager: AX5000 서보 드라이브 구성을 할 경우 선택

TwinCAT BACnet/IP: BACnet 서버를 사용할 경우 선택



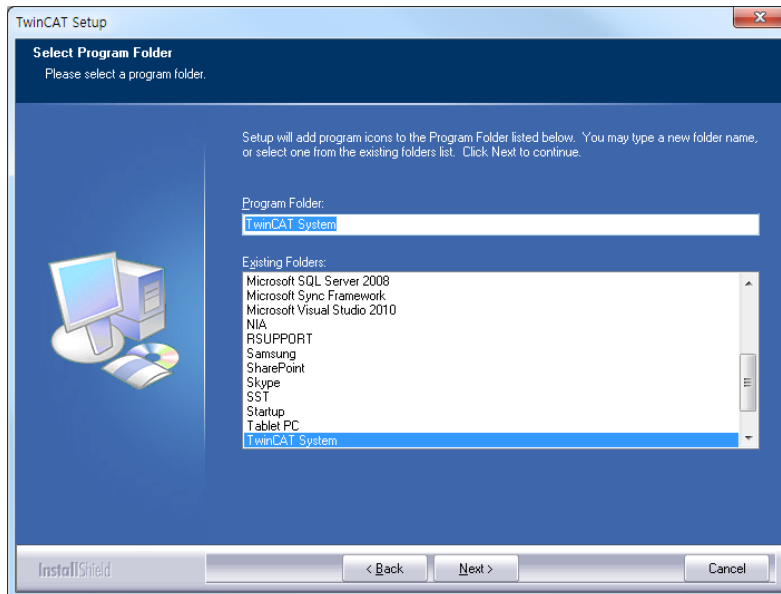
## • Choose Destination Location

설치 폴더를 선택한다.



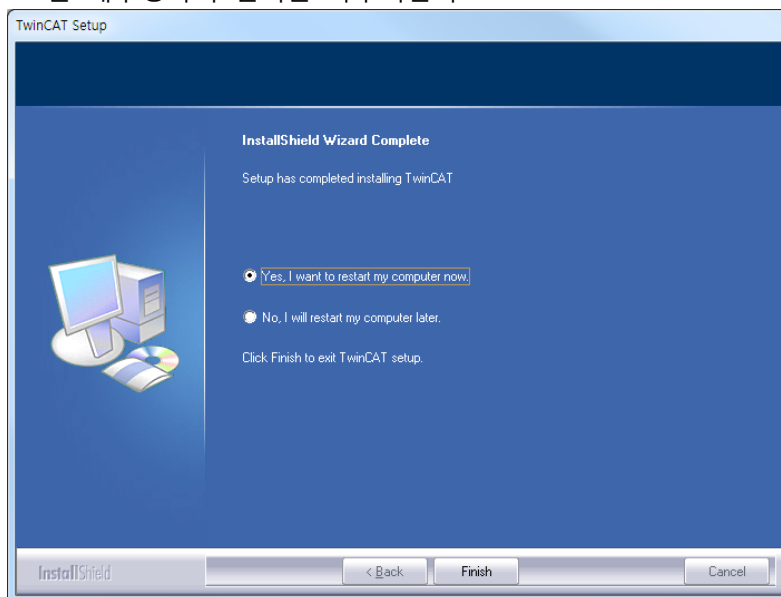
### • Select Program Folder

시작메뉴의 아이콘 설치 폴더를 선택한다.



### • 설치 완료

PC 를 재부팅하여 설치를 마무리한다..

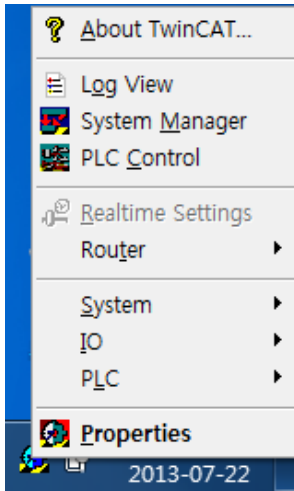


### 3. TwinCAT System Service

- Windows System tray 에 TwinCAT 아이콘이 생성 된다.



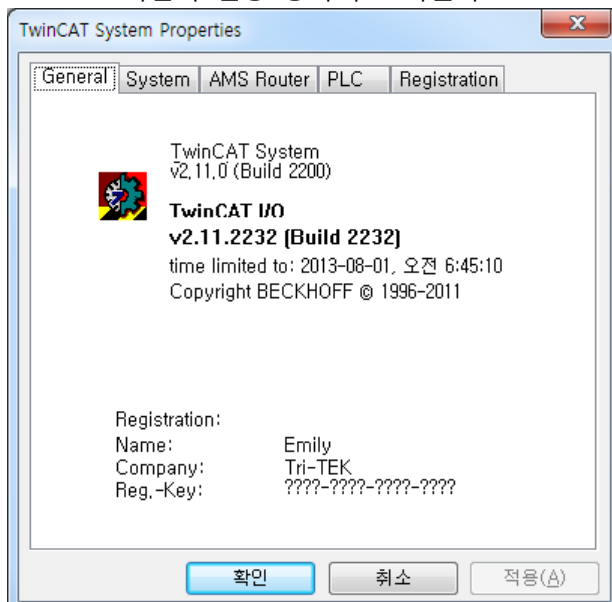
- TwinCAT 아이콘을 통해 TwinCAT System Service 에 접근할 수 있다.



#### 3.1. TwinCAT System Properties

- General

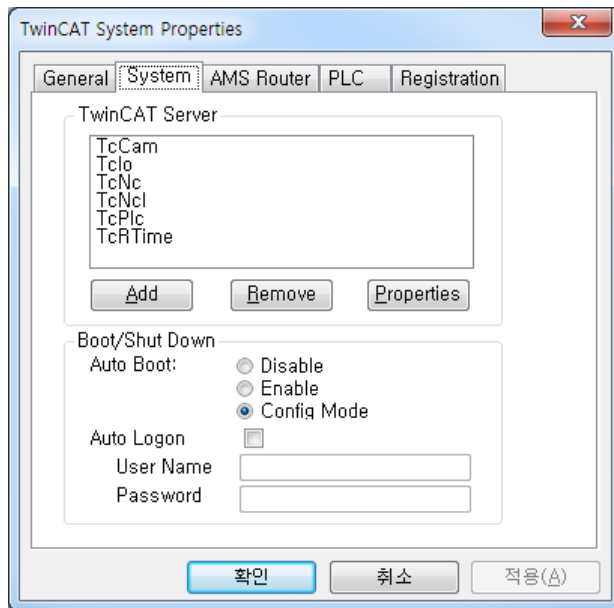
TwinCAT 버전과 인증 등록이 표시된다.



- **System**

상단 부분: Local PC 에 설치된 TwinCAT Server 를 확인할 수 있다.

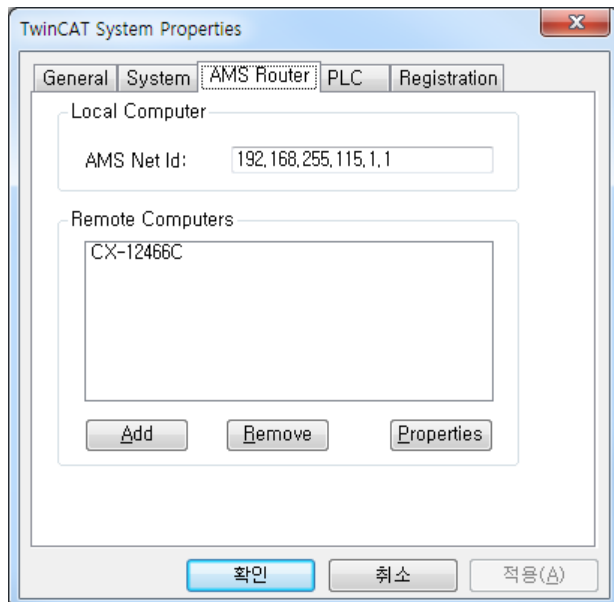
하단 부분: Windows 실행 시 TwinCAT Auto Boot 및 Windows Auto Logon 을 설정 할 수 있다.



- **AMS Router** – TwinCAT 의 통신 라우터

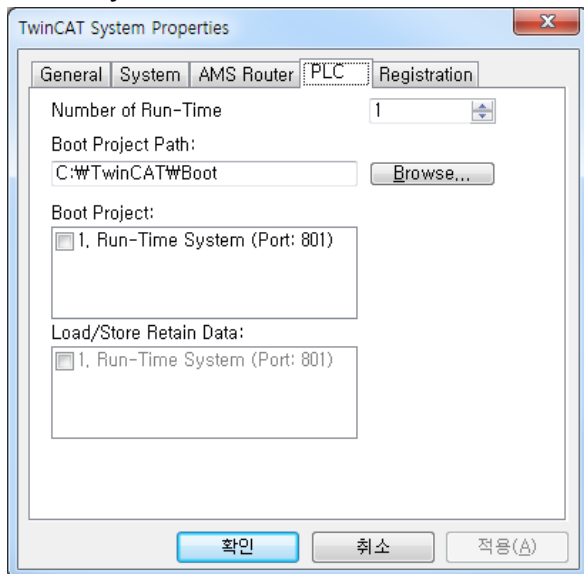
Local Computer: Local TwinCAT Service 의 고유 주소인 AMS Net Id 를 설정 및 표시한다.

Remote Computers: Local PC 와 연결했던 Remote PC 의 목록을 표시하며, 추가 등록 및 삭제가 가능하다.



## • PLC

최대 4 개의 PLC Run-Time 까지 구성 가능하며 Boot Project 의 경로를 설정할 수 있다. 또한, Boot Project 와 Retain Data 의 Enable 여부를 선택 할 수 있다.

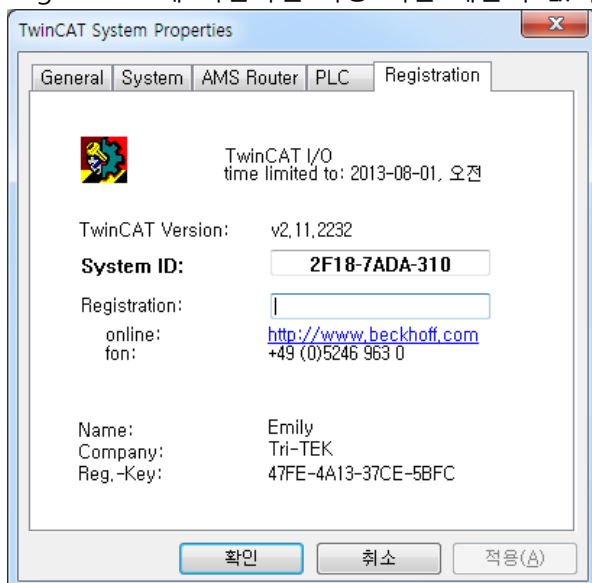


## • Registration

TwinCAT 버전 및 System ID 를 표시한다.

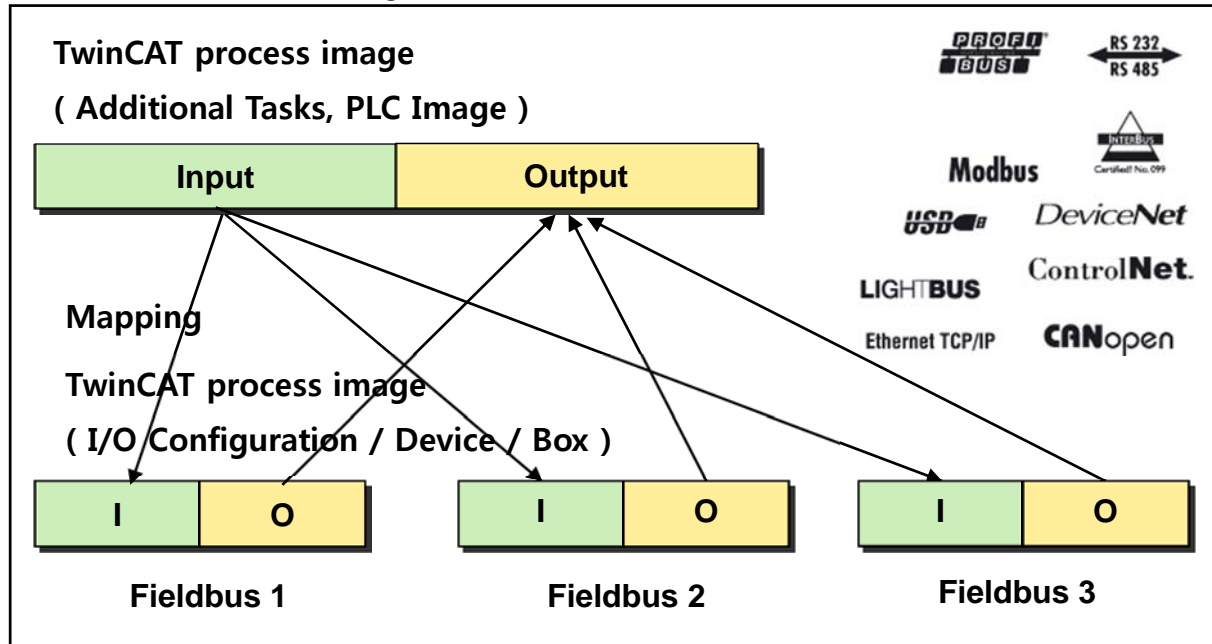
System ID: TwinCAT 이 설치된 하드디스크의 ID 와 CPU 의 ID 조합으로 되어 있으므로 TwinCAT 이 설치된 모든 PC 의 System ID 는 서로 다르다.

해당 PC 의 System ID 를 Beckhoff 에 보내면 Registration 키를 보내주며, 받은 라이선스를 Registration 에 기입하면 사용 기간 제한이 없어진다.



#### 4. TwinCAT System 구성 및 실행

TwinCAT System Manager 는 13 종류의 필드버스를 구성할 수 있다. 통합되는 필드버스 통신 데이터는 TwinCAT Process Image 에 재배열된다.



▶ TwinCAT I/O User Manual에서 사용된 Interface Card 및 Bus Coupler, Bus Terminal

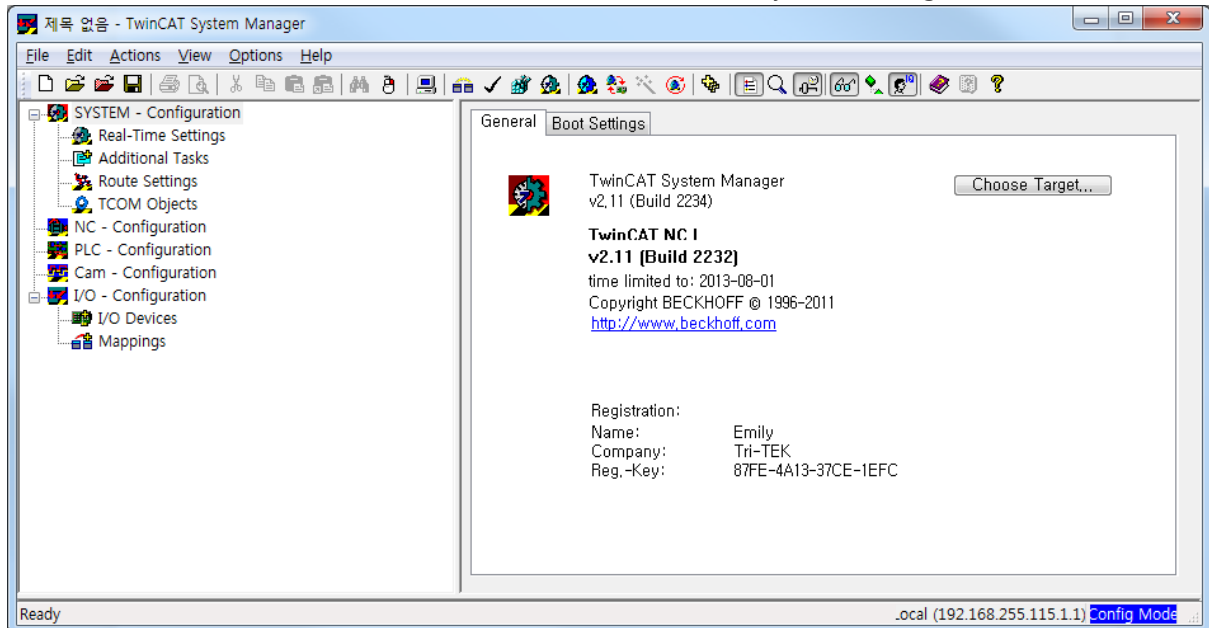
Vendor	Product	Type	slot
Intel	Intel Pro 100	Network Interface Card	
Beckhoff	EK1100	EtherCAT Coupler	1
Beckhoff	EL1088	8 Ch. Digital Input (24V, negative switching)	2
Beckhoff	EL2088	8 Ch. Digital Output (24V, negative switching)	3
Beckhoff	EL3058	8 Ch. Analog Input (4-20mA, single-ended)	4
Beckhoff	EL9010	End Terminal	6

본 문서는 EtherCAT 시스템을 설명하였다. 다른 필드버스 시스템도 본 문서에서 설명하는 절차와 방법으로 진행된다.

주의 사항 : EtherCAT 시스템을 구성하기 위해서는 TwinCAT 버전이 2.10 이상 되어야 한다.

## 4.1. TwinCAT System Manager

TwinCAT 시스템 구성은 필드버스 종류에 관계없이 TwinCAT System Manager 에서 이루어 진다.



- **SYSTEM – Configuration**

- General:** TwinCAT 버전, 사용자 정보, 인증 등록을 표시한다.

- Boot Settings:** 해당 PC 부팅 시 TwinCAT System 상태를 설정한다.

- \* **Real-Time Setting:** TwinCAT 의 기본 사이클 타임 및 CPU 점유율을 설정한다.

- \* **Additional Tasks:** TwinCAT IO 의 Task 및 변수들을 정의한다.

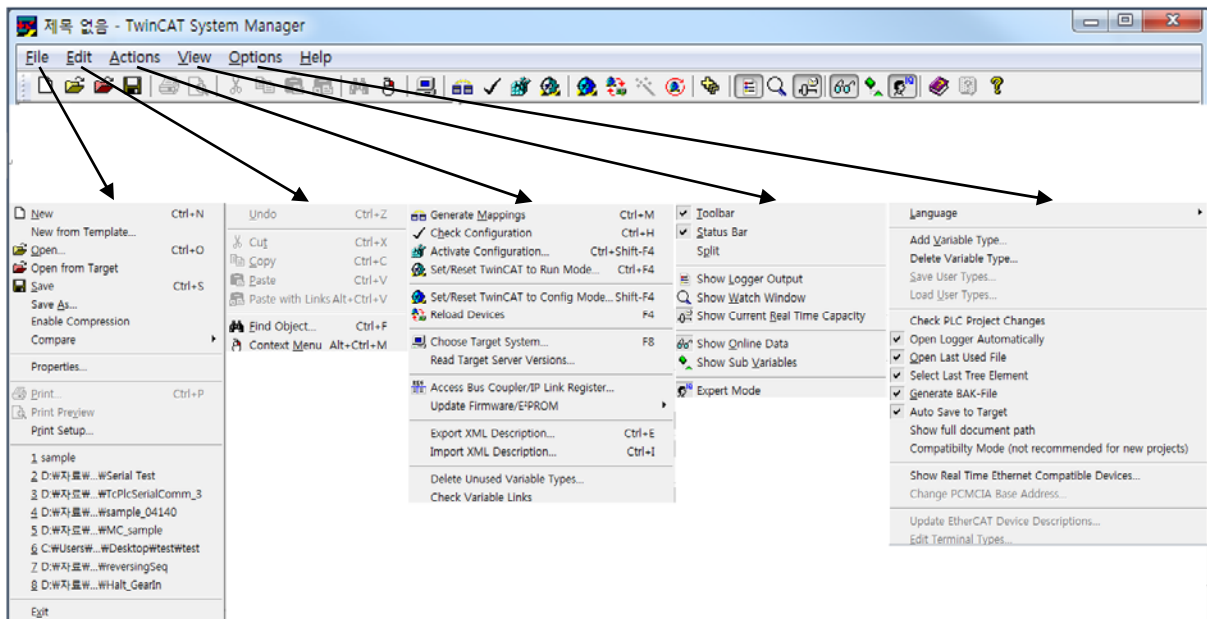
- \* **Router Settings:** 원격 PC 및 컨트롤러 연결을 설정한다.

- **I/O - Configuration**

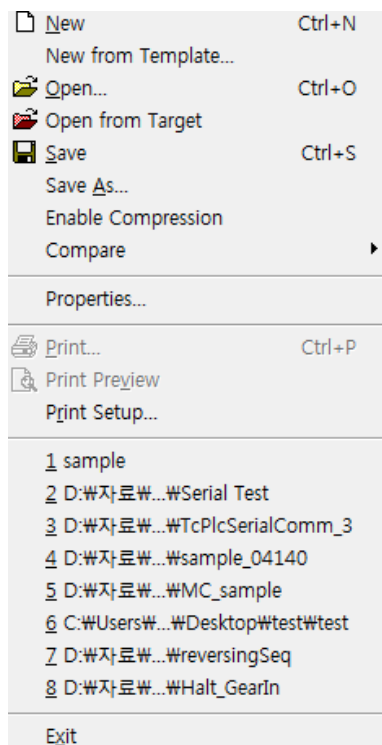
- \* **I/O Devices:** PC 와 연결된 필드버스 Slave 장치를 구성한다.

- \* **Mappings:** I/O Devices 와 연결된 Additional Task 의 변수 정보를 표시한다.

## • Menu bar



## • File



- New: 새 파일을 생성한다.
- New from Template: BC, BX 모델을 연결하기 위해 미리 정의된 template 을 불러온다.
- Open: 저장된 System Manager 파일을 불러온다.
- Open from Target: 최근에 사용한 System Manager 파일을 업로드 한다.
- Save: 구성된 System Manager 를 저장한다.
- Save As: 다른 이름으로 저장한다.
- Compare: 다른 System Manager 파일과 비교한다.



## • Action

	Generate <u>M</u> appings	Ctrl+M
	✓ Check Configuration	Ctrl+H
	Activate Configuration...	Ctrl+Shift-F4
	Set/Reset TwinCAT to Run Mode...	Ctrl+F4
	Set/Reset TwinCAT to Config Mode...	Shift-F4
	Reload Devices	F4
	Choose Target System...	F8
	Read Target Server Versions...	
	Access Bus Coupler/IP Link Register...	
	Update Firmware/E²PROM	
	Export XML Description...	Ctrl+E
	Import XML Description...	Ctrl+I
	Delete Unused Variable Types...	
	Check Variable Links	

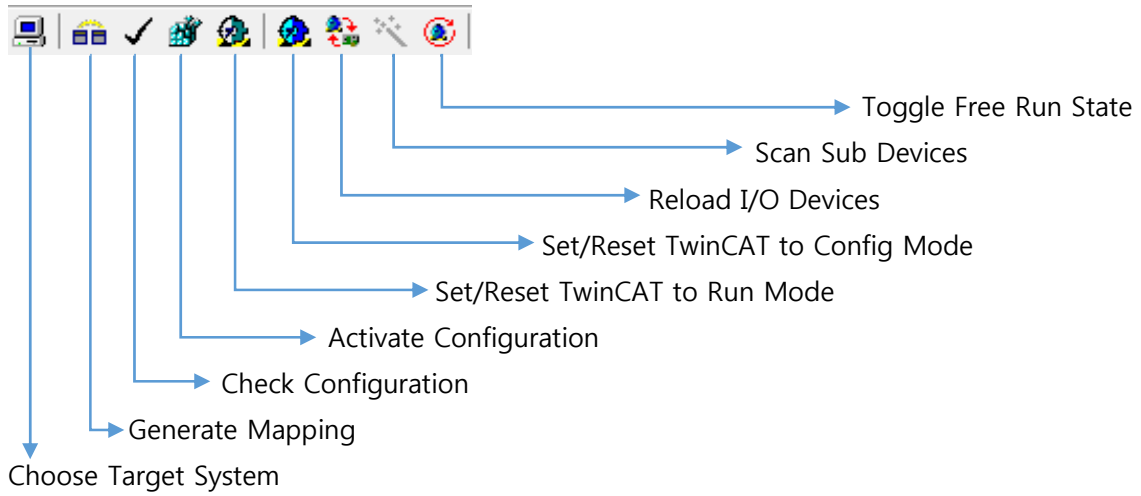
- Generate Mappings  
: 링크된 변수의 맵핑 정보를 생성한다.
- Check Configuration: 생성된 맵핑정보를 확인한다.
- Activate Configuration  
: System Manager 의 구성을 윈도우 레지스터에 저장하고 TwinCAT 을 RUN 모드로 변경한다.
- Set/Reset TwinCAT to Run Mode  
: TwinCAT 을 RUN 모드로 변경한다.
- Set/Reset TwinCAT to Config Mode  
: TwinCAT 을 Config 모드로 변경한다.

## • Option

<u>L</u> anguage	▶
Add <u>V</u> ariable Type...	
Delete Variable Type...	
<u>S</u> ave User Types...	
Load <u>U</u> ser Types...	
Check PLC Project Changes	
<input checked="" type="checkbox"/> Open Logger Automatically	
<input checked="" type="checkbox"/> <u>O</u> pen Last Used File	
<input checked="" type="checkbox"/> Select Last Tree Element	
<input checked="" type="checkbox"/> <u>G</u> enerate BAK-File	
<input checked="" type="checkbox"/> Auto Save to Target	
Show full document path	
Compatibilty Mode (not recommended for new projects)	
Show Real Time Ethernet Compatible Devices...	
Change PCMCIA Base Address...	
Update EtherCAT Device Descriptions...	
<u>E</u> dit Terminal Types...	

- Language: 사용할 언어를 설정한다.
- Add Variable Type  
: TwinCAT IO 에서 사용할 변수 타입(배열, 구조체)을 설정한다.
- Show Real Time Ethernet Compatible Devices  
: EtherCAT 및 Real Time Ethernet 드라이버를 설치 한다.
- Reload Devices  
: TwinCAT 에 연결된 필드버스 Slave 장치를 다시 로드 한다
- Choose Target System  
: 원격 제어할 PC 를 설정 및 선택한다.

• 메뉴 바

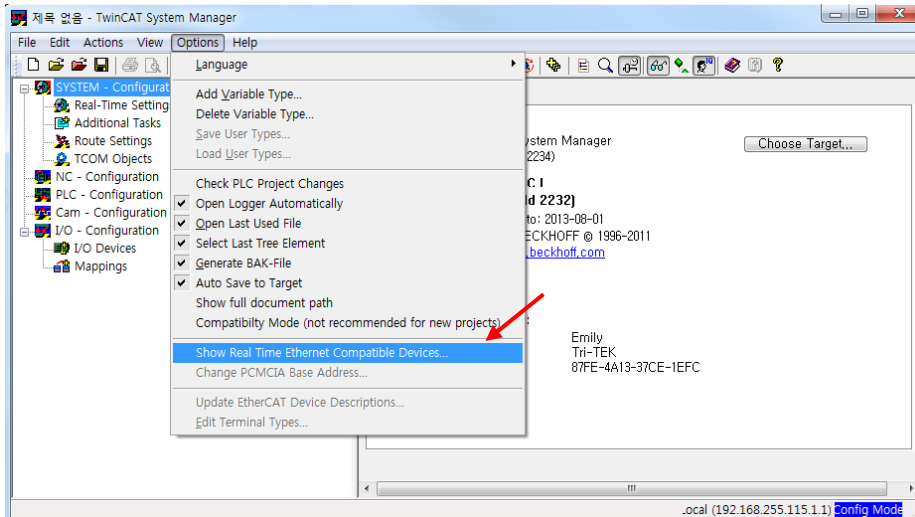


\* TwinCAT System Manager 의 메뉴에서 가장 많이 사용하는 부분만을 설명함.

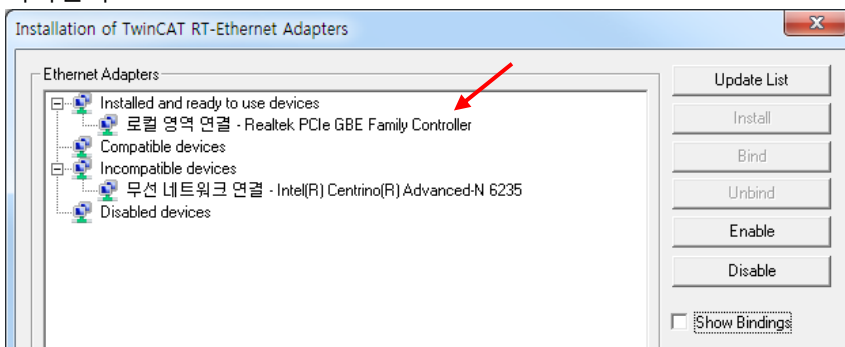
## 4.2. Remote IO 연결

### 4.2.1. EtherCAT Master Driver 설치

- [Option] – [Show Real Time Ethernet Compatible Devices]을 실행한다.



- 사용할 드라이버를 선택하고 Install 을 클릭하여 드라이버 추가가 완료되면 아래 화면과 같이 나타난다.



Installed and ready to use devices: 드라이버 설치가 완료되었고 사용이 가능한 디바이스를 표시한다.

Compatible devices: 드라이버 설치가 안된 인텔 랜 카드를 표시한다.

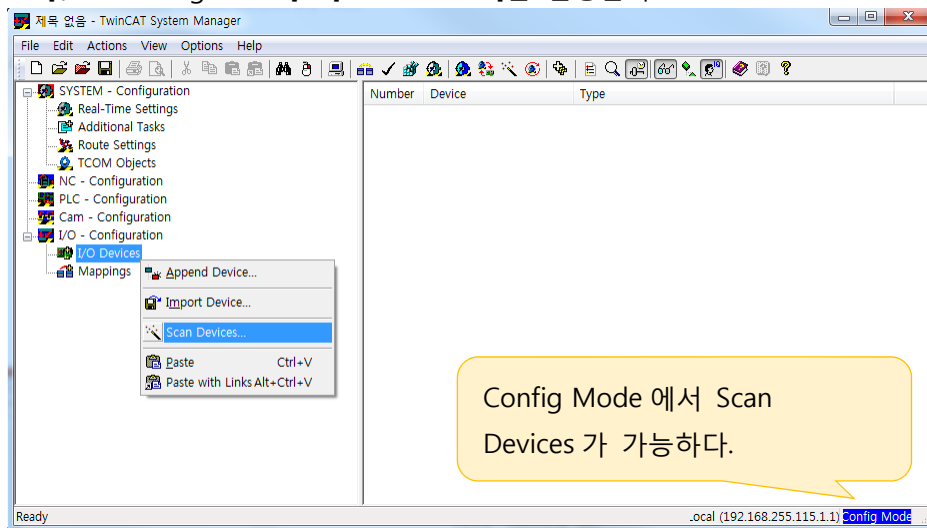
Incompatible devices: 드라이버 설치가 안된 인텔 외 다른 제품의 랜 카드를 표시한다.

Disabled devices: Disable 된 랜 카드를 표시한다.

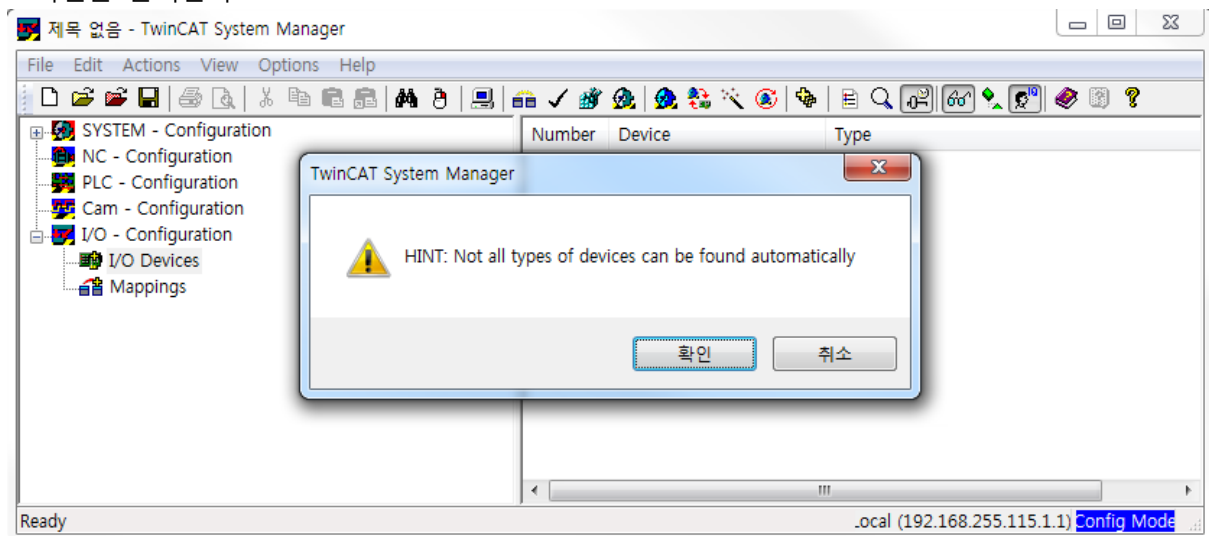
(\* EtherCAT 은 인텔 랜 카드를 사용하여 개발되었으므로 EtherCAT 의 성능을 100%사용하기 위해서는 반드시 인텔 랜 카드를 사용해야 한다. \*)

#### 4.2.2. EtherCAT Remote IO 연결

- [I/O – Configuration] – [Scan Devices]를 실행한다.

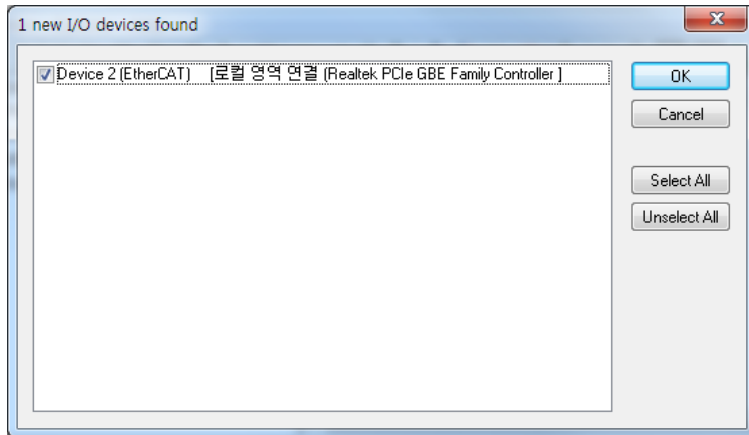


- 확인을 클릭한다.

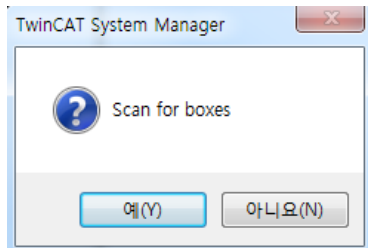


• 설치되어 있는 Interface Card 의 리스트가 표시된다. 해당 네트워크 카드를 체크한 뒤, OK 버튼을 클릭한다.

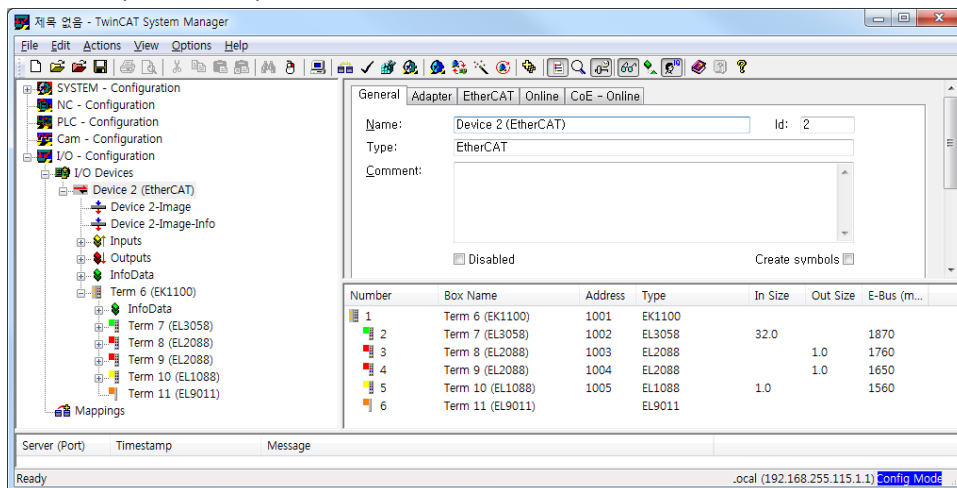
EtherCAT Slave 가 연결되지 않으면 'RT Ethernet'으로 표시된다.



• 네트워크 카드를 설치한 후 EtherCAT Slave 를 검색한다.



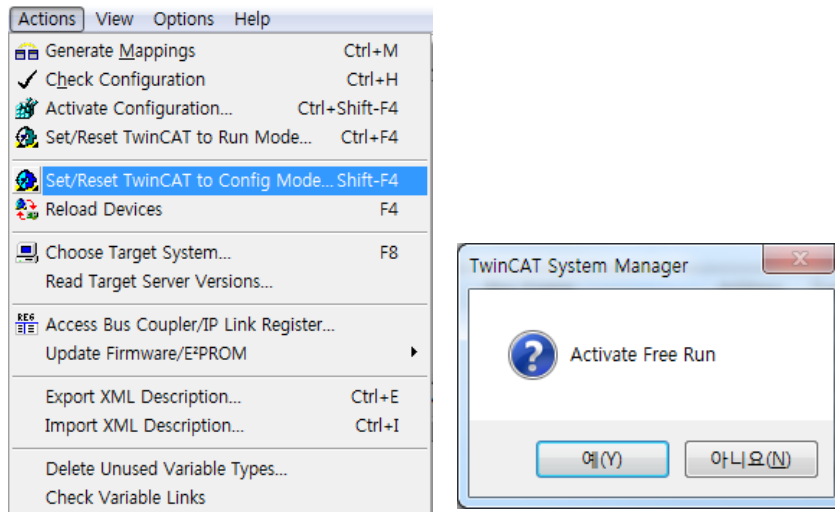
• Scan 이 완료된 화면



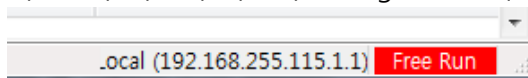
- **Free Run Mode**

: 제어를 할 수는 없지만, 간단한 IO 테스트를 위해 통신이 연결된 상태

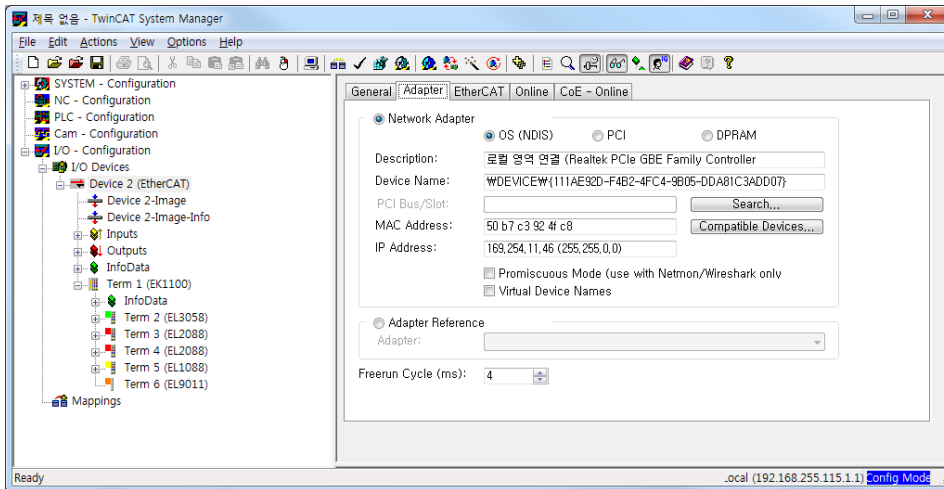
[Action]-[Set/Reset TwinCAT to Config Mode]



TwinCAT System Manager 우측 하단 부분에서 다음 그림과 같이 빨강색 바탕의 Free Run 텍스트와 파란색 바탕의 Config Mode 텍스트가 번갈아 보이게 된다.



## • EtherCAT Adapter tab



**Description:** 사용중인 EtherCAT Master Device 를 표시한다.

**MAC Address:** EtherCAT Master Device 의 MAC Address 를 표시한다.

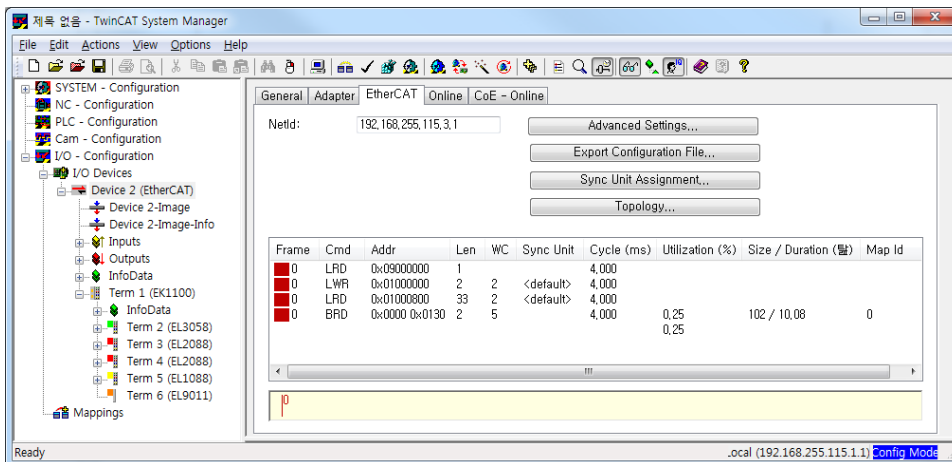
**Search:** EtherCAT Master Device 선택창을 화면에 표시한다.

**IP Address:** EtherCAT Master Device 의 IP Address 를 표시한다.

**Compatible Devices:** EtherCAT Master Device 드라이버 설치 창을 화면에 표시한다.

**Freerun Cycle:** TwinCAT 이 Freerun 모드일 때 사용하는 Cycle 타임이다.

## • EtherCAT EtherCAT tab

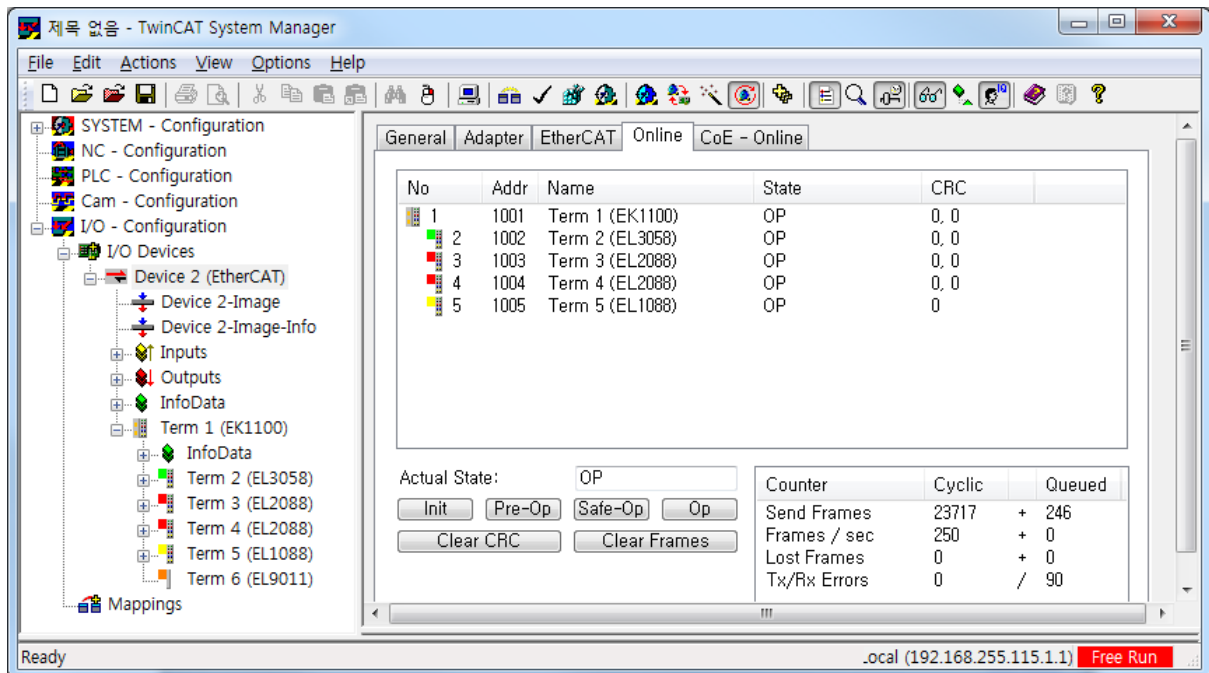


**NetID:** EtherCAT 마스터의 NetID 를 표시한다.

**Advanced Settings:** Advanced Settings 창을 화면에 표시한다.

**Export Configuration File:** EtherCAT 마스터 구성 XML 파일을 만든다.

• EtherCAT Online tab



**NO:** EtherCAT Slave 의 순서를 표시한다.

**Addr:** EtherCAT Slave 의 address 를 표시한다.

**State:** EtherCAT Slave 의 상태를 표시한다.

**Actual State:** EtherCAT Slave 의 상태를 표시한다.

**Send Frames:** EtherCAT 프레임 을 표시한다.

**Frames:** EtherCAT 초당 프레임 수를 표시한다.

**Init:** EtherCAT Slave 의 상태를 Init 으로 설정한다.

**Pro-OP:** EtherCAT Slave 를 Pro-OP 로 설정한다.

**Safe-OP:** EtherCAT Slave 를 Safe-OP 로 설정한다.

**OP:** EtherCAT Slave 를 OP 로 설정한다.

**Clear CRC:** CRC 카운터를 클리어 한다.

**ClearFrames:** 모든 프레임 값을 초기화 한다.

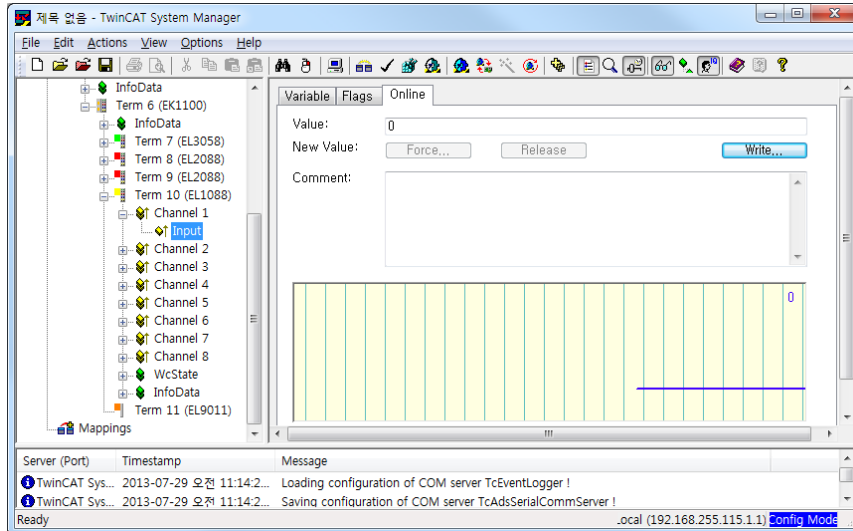
(\* 로스트 프레임 또는 CRC 에 값이 존재한다면 EtherCAT 통신상태가 불량을 의미하며, 대다수의 불량 원인은 불량 Ethernet 케이블 사용에 있다. \*)



### 4.2.3. Remote IO 테스트

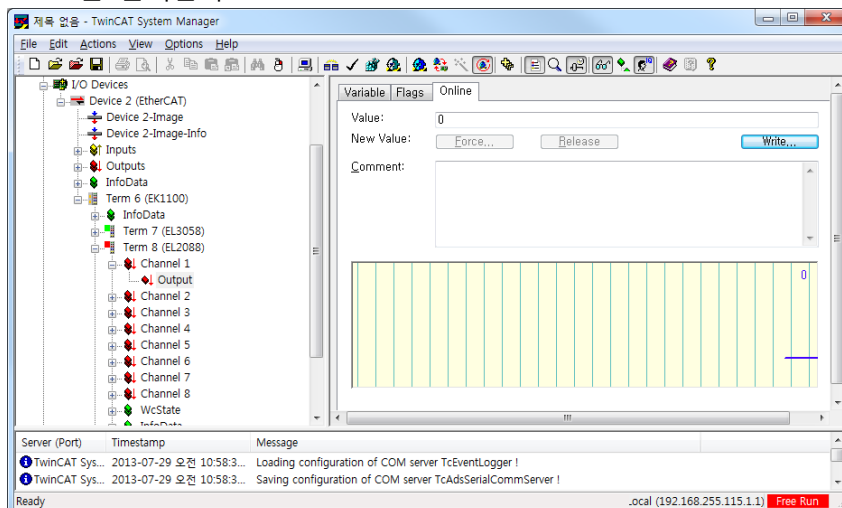
#### A. Input Test

TwinCAT 이 Free Run 모드일 때 입력이 인가되는 터미널을 선택하고 확인하고자 하는 채널의 input 을 선택하여 입력신호 값을 확인한다.

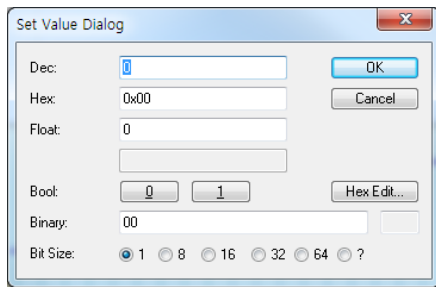


#### B. Output Test

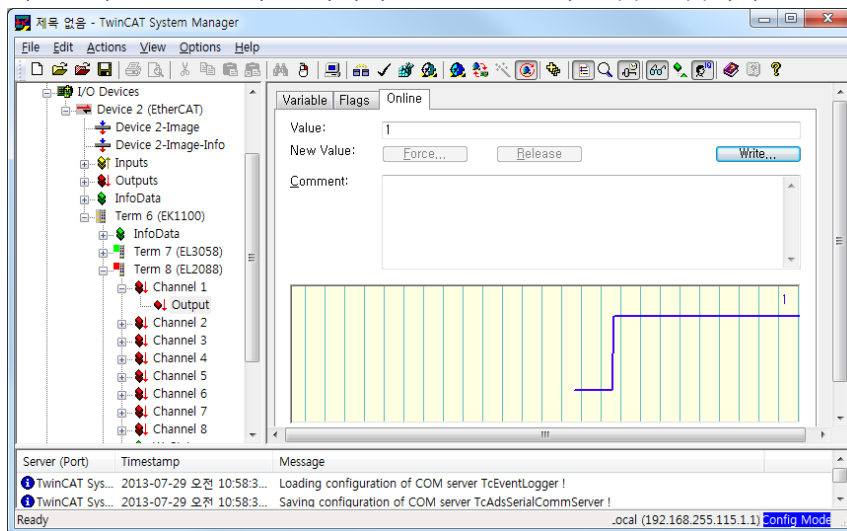
TwinCAT 이 Free Run 모드일 때 출력 값을 입력할 터미널의 Output 을 선택하고 Online 탭의 Write 를 클릭한다.



다음 화면에서 Bool 값을 1로 선택한다.



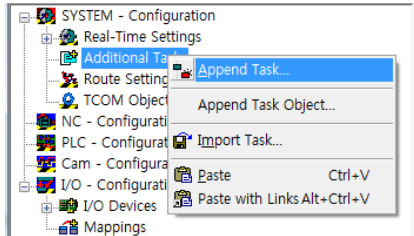
모니터 화면에 1의 값이 출력되는 것을 확인 할 수 있다. 실제 디지털 출력 모듈의 출력 여부를 확인한다. LED 신호가 출력되지 않으면 설정이 잘못된 것이다.



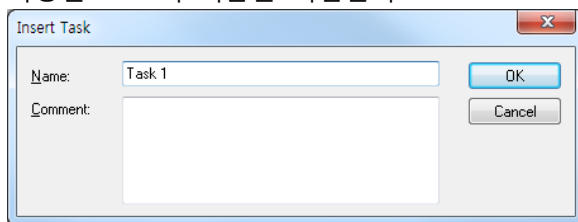
### 4.3. TwinCAT IO Task 설정

#### 4.3.1. Additional Task

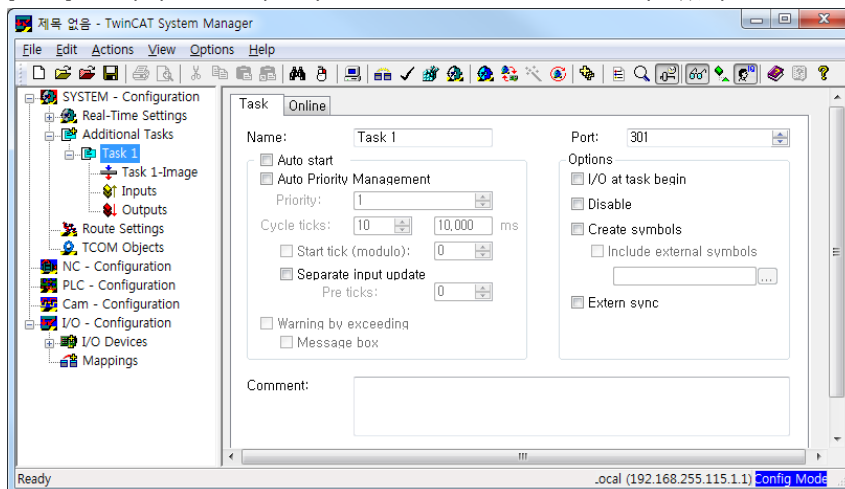
[SYSTEM – Configuration] - [Additional Tasks]를 선택하고 우 클릭하여 [Append Task] 를 선택한다.



사용할 Task 의 이름을 기입한다.



[Task] 탭에서 Task 이름과 Port Number 를 설정할 수 있다.



**Name:** Task 이름

**Port:** Task Port 를 설정 (TwinCAT IO 는 최대 5 Port(301 ~ 305)의 Port 를 사용할 수 있다.)

**Auto Start :** 설정한 Cycle Tick 의 간격으로 자동 업데이트 사용 유무를 설정

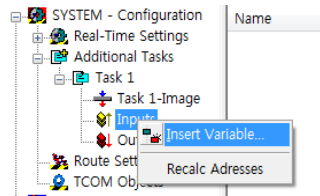
**Auto Priority Management :** Task 의 우선순위를 TwinCAT System 에서 자동으로 관리

**Cycle Tick :** Task 의 Cycle Tick 을 설정

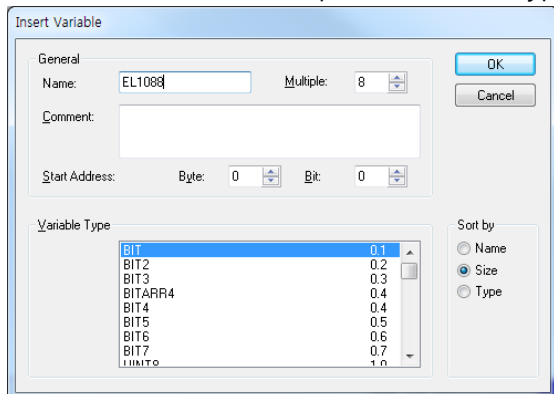
(\*TwinCAT IO 를 사용하기 위해서는 반드시 AutoStart 를 선택해야 한다.\*)

### 4.3.2. Insert Variable

[Additional Tasks]의 [Task1]의 Input 또는 Output 을 우 클릭하여 Insert Variable 을 클릭한다.



예 : Name :EL1088 , Multiple : 8, Variable Type : Bit, OK 클릭



**Name** : 선언할 변수의 이름을 입력한다.

**Multiple** : 선언할 변수의 수량을 입력한다.

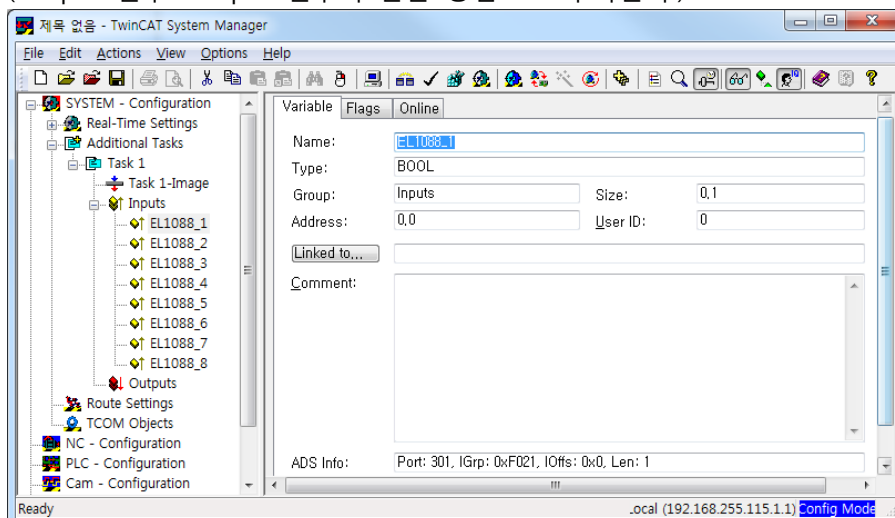
**Start Address** : 선언할 변수의 Address 를 설정한다.

**Variable Type** : 선언할 변수의 Type 을 선택한다.

EL1088 설정화면

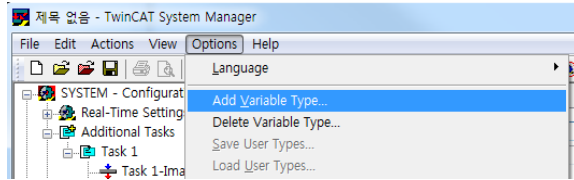
Task 1 의 Input 영역에 Address 가 0.0~0.3 을 사용하는 BOOL Type 의 논리 주소를 생성하게 된다.

(Output 변수도 Input 변수와 같은 방법으로 추가한다.)

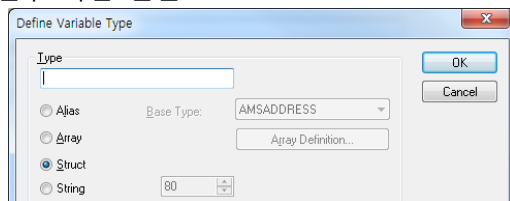


### 4.3.3. Add Variable Type

기본적으로 설정되어 있는 변수 타입은 사용자가 유연하게 사용할 수 없다. 그러므로 TwinCAT System Manager 의 메뉴바에 [Option] – [Add Variable Type]를 선택하여 사용자가 변수를 정의해 구성할 수 있다.



변수 타입 선언

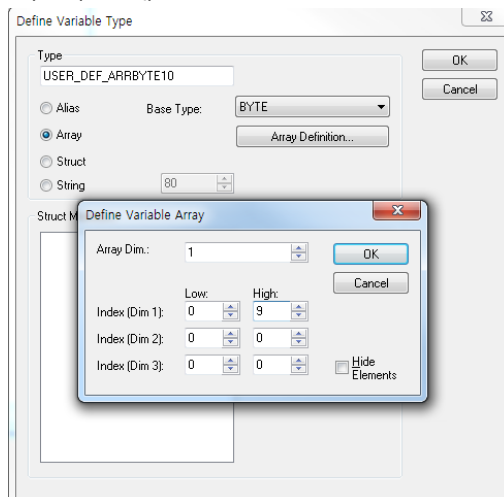


**Alias:** 사용자가 데이터타입을 설정

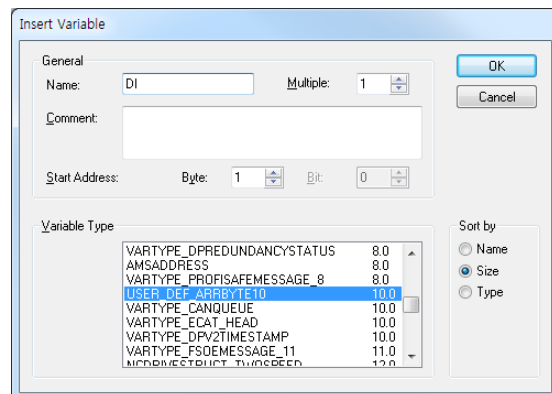
**Array:** 배열 선언

**Struct:** 구조체 선언

변수 사용 예:

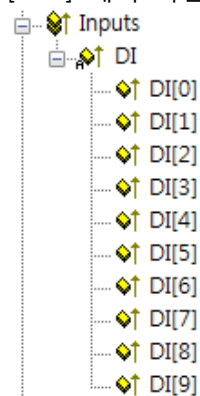


USER\_DEF\_ARRBYTE10 이라는 배열을 정의  
(크기 10 의 바이트 배열)



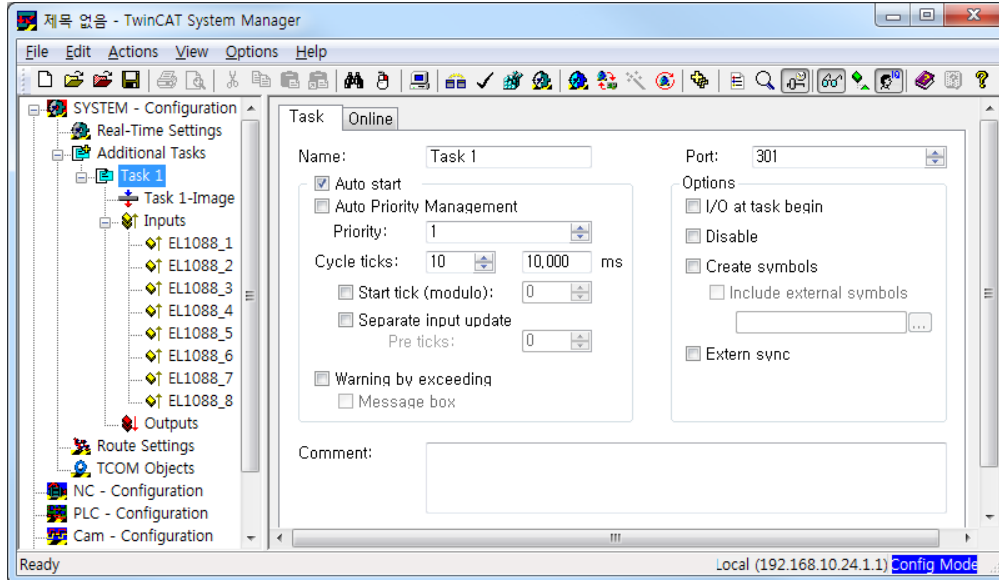
변수 선언 다이얼로그에서 Variable Type 의  
'USER\_DEF\_ARRBYTE10'을 선택한다.

[Task] 에서 확인.



#### 4.3.4. Auto start 체크

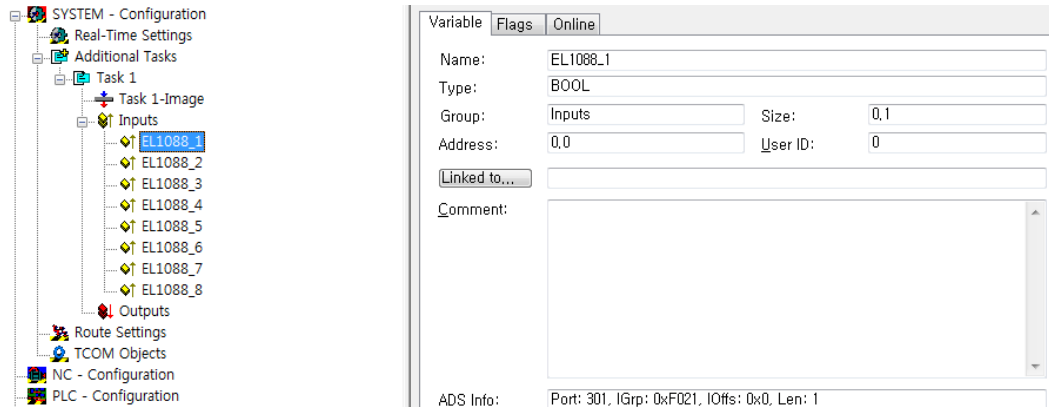
4.3.1 Additional Task 에서 설명한 'Auto Start' 체크박스를 선택하였을 때는 설정된 'Cycle Time'에 맞춰 TwinCAT System Manager 가 자동으로 I/O 업데이트를 한다. 그렇게 때문에 사용자 어플리케이션에서 I/O 데이터를 업데이트하는데 시간을 투자 할 필요가 없다.



### 4.3.5. Link

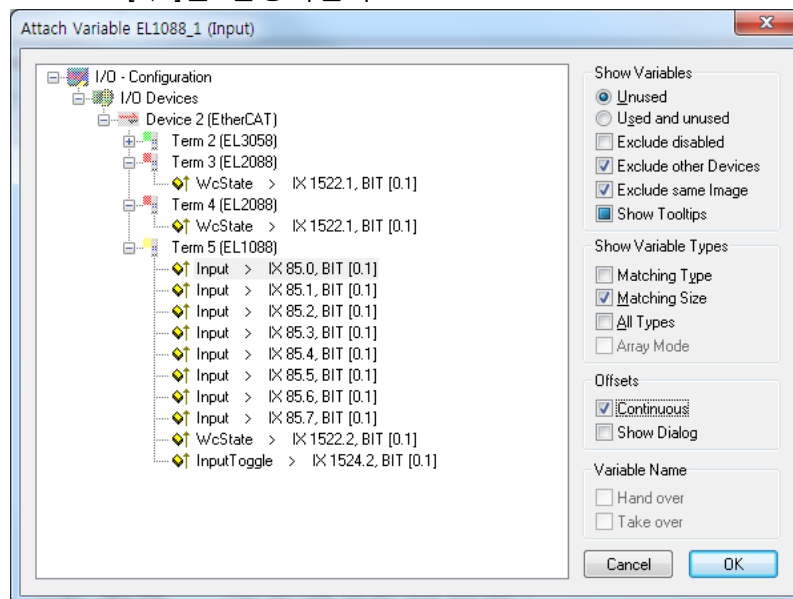
#### A. 단일 링크

Task 에 설정한 Input 영역의 논리 주소를 I/O Channel 에 링크하기 위해 'Linked to...' 버튼을 클릭한다. 또는 링크할 변수를 더블 클릭한다.



연결할 Remote IO 의 변수를 선택한다.

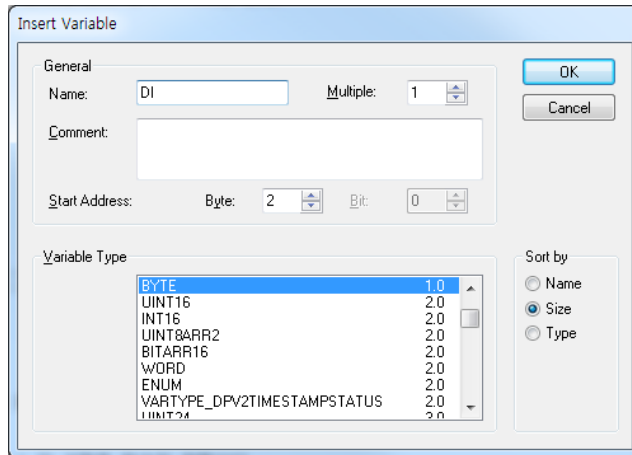
Additional Tasks 의 Inputs 영역인 EL1088\_1(논리 Address 0.0)과 EL1088 의 Input->IX 85.0 BIT [0,1]을 맵핑시킨다.



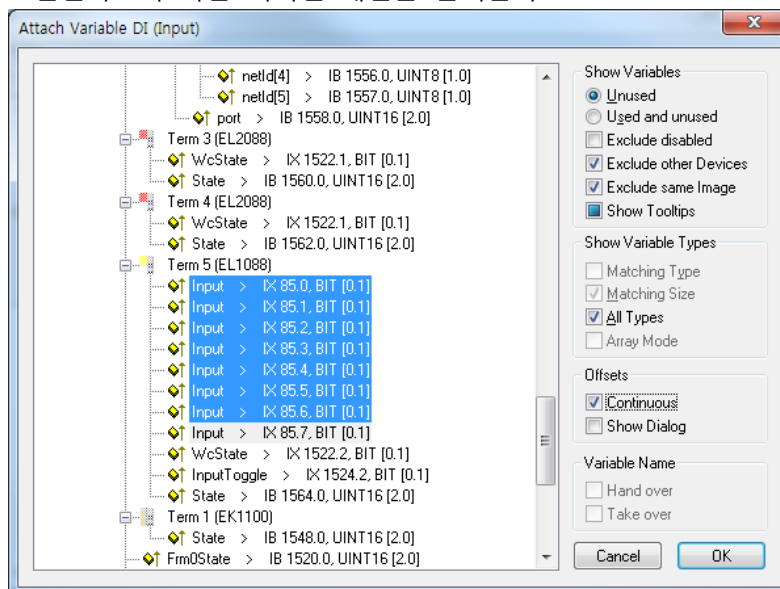
## B. 멀티 링크

TwinCAT 'Multi Link' 기능을 제공한다. 예를 들면 1 byte 형 데이터 타입에 8 개의 디지털 변수를 연결할 수 있다.

- 바이트 형 변수를 추가한다.



- 추가한 바이트 형 변수를 링크한다.
- 다음 다이얼로그화면에서 오른쪽 'Show Variable Types->All Type' 체크 박스 선택, 'Offsets->Continuous' 체크 박스 선택
- 연결하고자 하는 디지털 채널을 선택한다.




Additional Task 에서 구성되는 Input/Output Process Image 구성은 I/O Terminal 에 연결되어 있는 Channel 에 따라 본 절의 순서에 따라 계속 추가하면 된다.

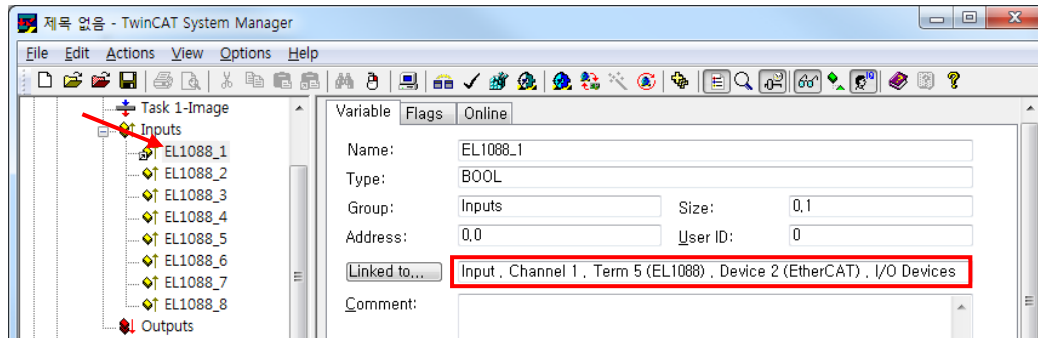


### C. Link 정보

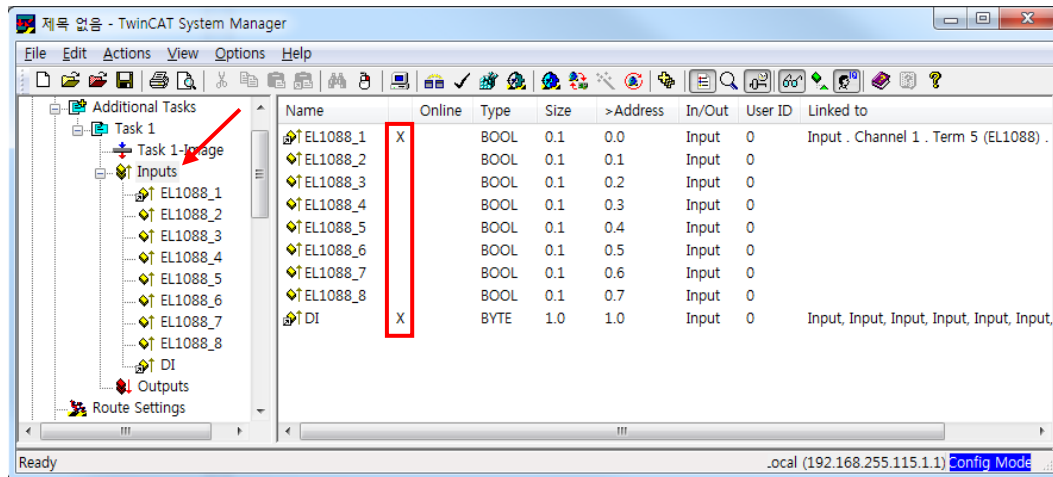
Link 정보를 설정한 변수는 리스트 아이콘에 화살표로 표시된다.

... EL1088\_1

[EL1088\_1]의 [Variable]탭에서 맵핑된 물리 주소의 정보를 확인할 수 있다.

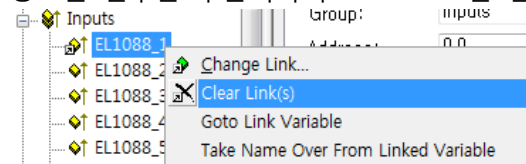


Task 변수와 IO 모듈의 채널이 링크되면 X로 표시된다.



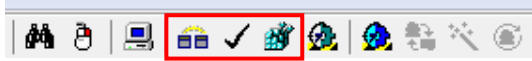
### D. Clear Link

링크된 변수를 우클릭하여 Clear Link를 클릭하면 링크가 해지 된다.

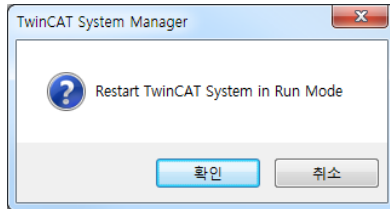


#### 4.3.6. TwinCAT Run Mode 실행

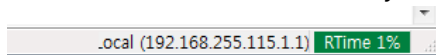
[Generate mappings] - [Check Configuration] - [Activate Configuration]을 순서대로 실행한다.



확인을 클릭한다.



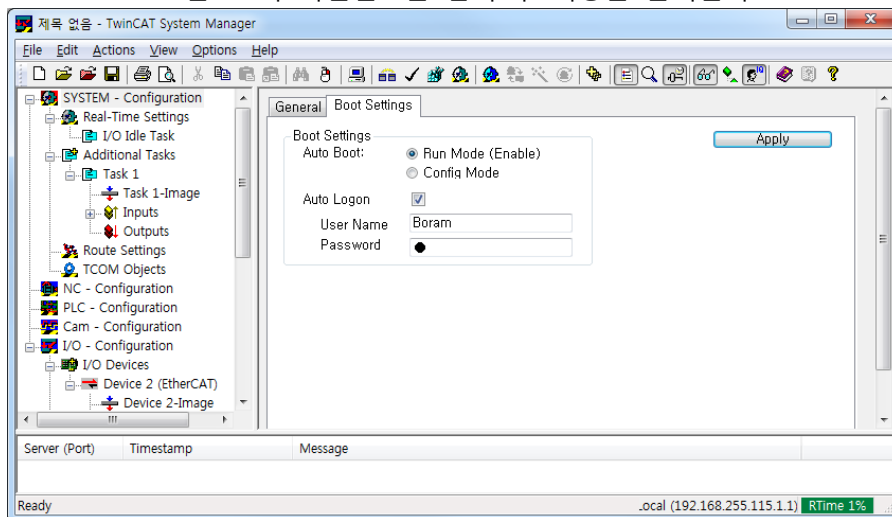
우측 하단의 녹색은 TwinCAT System 의 Run Mode 를 의미한다.



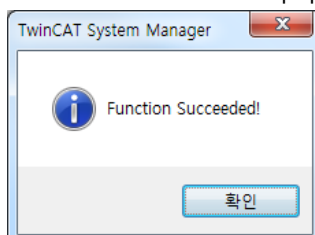
#### 4.3.7. TwinCAT Auto Boot Setting

TwinCAT System Manager 을 구성한 뒤, 로컬 PC 가 재부팅 될 때 자동적으로 Run Mode 로 전환될 수 있도록 설정하는 기능이다. PC 재부팅 시 실행되는 TwinCAT System Manager 파일은 가장 마지막에 구성하여 Activate Configuration 을 실행한 파일을 실행한다.

[System – Configuration] – [Boot Settings] 탭에서 Auto Boot 를 Enable 로 선택, Auto Logon 에 Windows 로그인 ID 와 비밀번호를 입력 후 적용을 클릭한다.



Function Succeeded 다이얼로그의 확인을 클릭한다.

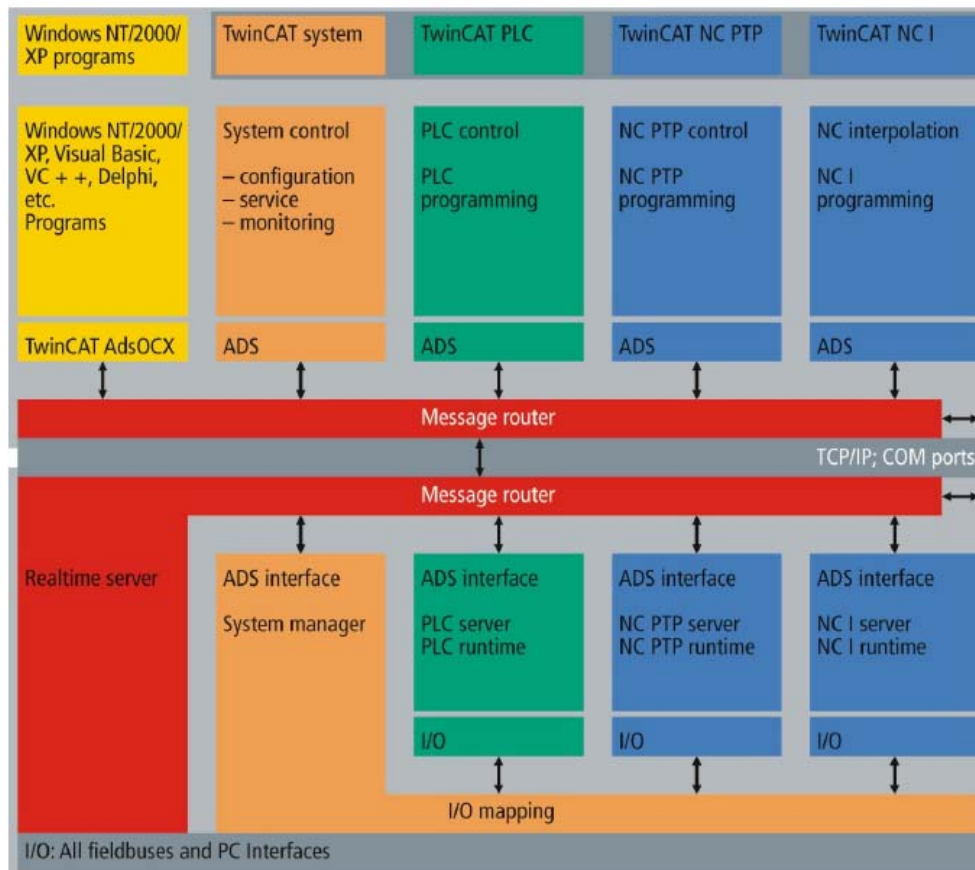


## 5. ADS Connectivity

Automation Device Specification(ADS)는 독립적인 디바이스와 필드버스에 의존하지 않고 ADS Device 에 접근 방식을 관리하는 인터페이스이다.

(\*TwinCAT ADS 통신을 사용하기 위해서는 반드시 TwinCAT 이 Run 모드일 때만 사용이 가능하다.\*)

### 5.1. TwinCAT Device Concept



TwinCAT 시스템 구조는 소프트웨어적인 모듈인 TwinCAT PLC, TwinCAT NC 등을 독립적인 디바이스(가상 디바이스 : Virtual Devices)로 취급한다. 소프트웨어적인 모듈은 서버/클라이언트 방식으로 연결되며, 모듈들은 'Message Router'을 거치는 ADS(Automation Device Specification) 인터페이스를 이용하여 메시지를 교환한다.

ADS 를 통해 모든 TwinCAT server 와 client program 의 command 와 데이터 교환이 가능하다.

## 5.2. ADS Libraries

TwinCAT Router 를 통해 ADS 디바이스들간의 데이터 교환에 필요한 메소드 및 이벤트들이 제공된다. 프로그램 환경에 맞도록 다양한 종류의 ADS 라이브러리를 지원한다.

- **"TcSystem.lib" PLC library**

TwinCAT PLC 에서 사용된다.

- **ADS-DLL**

ADS 통신을 사용하기 위한 2 가지(API, COM interfaces) Client Functions 라이브러리형식으로 지원한다.

(e.g. C/C++, Delphi, etc)

- **ADS.Net component**

ADS Device 통신 클래스를 지원하는 클래스 라이브러리이다.

(e.g. VB.NET, C#, Delphi.NET, Delphi Prism, etc)

- **ADS-OCX (ActiveX-Control)**

Microsoft 사의 COM 기술을 기반으로 ADS 통신을 사용하기 위한 다양한 종류의 메소드를 지원한다.

(e.g. Visual Basic, C++, Delphi, etc)

- **ADS-Script-DLL**

Script 언어에서 사용할 수 있는 라이브러리이다.

(e.g. VBScript, Jscript, etc)

- **ADS-WebService**

HTTP 를 통한 ADS 서비스이다.

(e.g. Visual C#, Delphi.NET, etc)

- **ADS-Java-DLL**

### 5.3. ADS Device Identification

- TwinCAT 시스템은 각 ADS 디바이스를 식별하기 위한 두 가지 식별자를 가진다.
  - ➔ AdsAmsNetId
  - ➔ AdsPortNr
- 해당 디바이스의 데이터에 접근하기 위해 세 가지 식별자가 더 필요하다.
  - ➔ Group Index (Input/Output/Memory)
  - ➔ Offset Index
  - ➔ Size

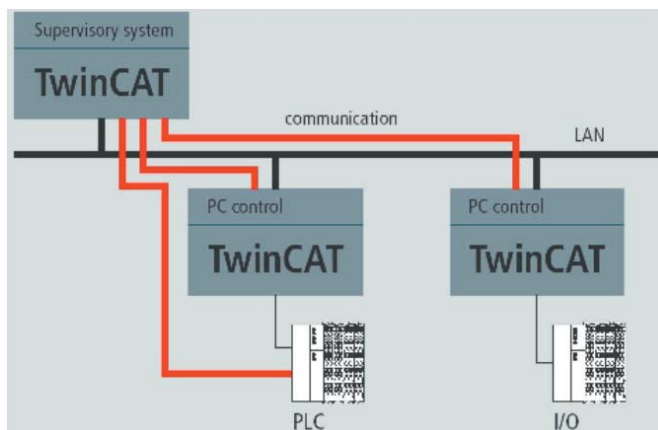
#### 5.3.1. ADS-AmsNetId

- TwinCAT 이 설치된 PC 의 식별자로 TCP/IP 를 확장한 형태이다.

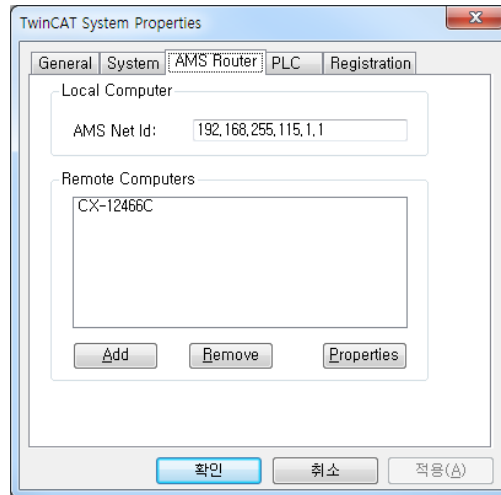
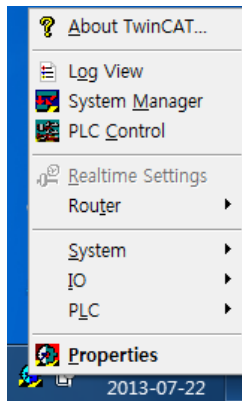
예) TCP/IP : 172.1.2.16

AmsNetId : 172.1.2.16.**1.1**

- 하나의 PC 에서 이루어지는 TwinCAT 모듈 간의 데이터 교환뿐만 아니라 ADS 를 통해 TwinCAT 이 설치된 여러 PC 들 간의 데이터 접근 또한 가능하다.



- TwinCAT-PC 의 ADS-AmsNetId 는 TwinCAT system service 에서 설정 가능하다.



### 5.3.2. ADS-Port

TwinCAT 시스템에 존재하는 소프트웨어적인 모듈(PLC, IO, NC 등)들의 접근 포트(AdsPortNr)들이 정해져 있다.

ADS-PortNr	ADS device description
100	Logger (only NT-Log)
110	Eventlogger
300	IO
301	additional Task 1
302	additional Task 2
303	additional Task 3
304	additional Task 4
305	additional Task 5
500	NC
801	PLC RuntimeSystem 1
811	PLC RuntimeSystem 2
821	PLC RuntimeSystem 3
831	PLC RuntimeSystem 4
900	Camshaft controller
10000	System Service
14000	Scope

### 5.3.3. Index - Group/Offset

#### A. Specification of the TwinCAT ADS system services

Index Group	Index Offset	Access	Data type	Description
0x0000F020	0x00000000- 0xFFFFFFFF	R/W	UINT8[n]	<b>READ_I - WRITE_I</b> PLC process diagram of the physical inputs(%I field). Offset is byte offset.
0x0000F021	0x00000000- 0xFFFFFFFF	R/W	UINT8	<b>READ_IX - WRITE_IX</b> PLC process diagram of the physical inputs(%IX field). The index offset contains the bit address which is calculated from byte number +8 + bit number
0x0000F030	0x00000000- 0xFFFFFFFF	R/W	UINT8[n]	<b>READ_Q - WRITE_Q</b> PLC process diagram of the physical outputs(%Q field). Offset is byte offset.
0x0000F031	0x00000000- 0xFFFFFFFF	R/W	UINT8	<b>READ_QX - WRITE_QX</b> PLC process diagram of the physical outputs(%QX field). The index offset contains the bit address which is calculated from the byte number *8 + bit number.

#### B. Specification of the PLC services

This section includes services to access the PLC memory range (%M field).

Index Group	Index Offset	Access	Data type	Description
0x00004020	0x00000000- 0x0000FFFF	R/W	UINT8[n]	<b>READ_M - WRITE_M</b> PLC memory range(%M field).Offset is byte offset.
0x00004021	0x00000000- 0xFFFFFFFF	R/W	UINT8	<b>READ_MX - WRITE_MX</b> PLC memory range (%MX field).The low word of the index offset is the byte offset. The index offset contains the bit address calculated from the byte number *8 + bit number



## 5.4. ADS API 호출별 분류

ADS Client 가 요청하면 ADS Server 가 응답하는 방식으로 통신하며, 접근방식은 Synchronous, Asynchronous, Notification 3 가지가 있다.

### 5.4.1. Synchronous

Client 가 Server 에게 메시지를 요청하였을 때 서버로부터 응답 받을 때까지 기다리는 방식이다.

- 변수 형태: ALL
- 접근 방식: By name, By address
- 이점: 사용자 어플리케이션에서 처리하기 편리하다.
- 단점: 이벤트 처리를 사용자 어플리케이션에서 직접 작성해야 한다. 이점은 데이터를 읽을 때 큰 단점이 될 수 있다.
- 비교: 로컬 통신에서 주로 사용되며, 빠르고, 초기화 단계가 필요하다

	Read	Write
By name	AdsSyncReadBoolVarReq AdsSyncReadIntegerVarReq AdsSyncReadLongVarReq AdsSyncReadSingleVarReq AdsSyncReadDoubleVarReq AdsSyncReadStringVarReq	AdsSyncWriteBoolVarReq AdsSyncWriteIntegerVarReq AdsSyncWriteLongVarReq AdsSyncWriteSingleVarReq AdsSyncWriteDoubleVarReq AdsSyncWriteStringVarReq
By address	AdsSyncReadBoolReq AdsSyncReadIntegerReq AdsSyncReadLongReq AdsSyncReadSingleReq	AdsSyncWriteBoolReq AdsSyncWriteIntegerReq AdsSyncWriteLongReq AdsSyncWriteSingleReq AdsSyncWriteDoubleReq AdsSyncWriteStringReq

### 5.4.2. Asynchronous

Client 에서 Server 에게 메시지를 요청하였을 때 서버로부터 응답을 기다리지 않고 콜백 함수를 지정하여 콜백 함수에서 반환 값을 처리하도록 하는 방식이다

- 변수 형태: All except BOOL
- 접근 방식: 일반적으로 어드레스 기반으로 사용하지만 이름 기반으로도 사용한다.
- 이점: 사용자 어플리케이션에서 응답을 기다리지 않고 주 메인 프로세스를 진행시킬 수 있다.
- 단점: 사용자 어플리케이션에서 응답 이벤트 처리 부분을 별도로 작성해야 하기 때문에 더 많은 소스 코드가 필요하다.
- 비고 : 많은 데이터 양을 빠르지 않게 통신하는데 주로 사용된다.

	Read	Write
<b>By address</b>	AdsReadIntegerReq AdsReadLongReq AdsReadSingleReq AdsReadDoubleReq AdsReadStringReq	AdsWriteIntegerReq AdsWriteLongReq AdsWriteSingleReq AdsWriteDoubleReq AdsWriteStringReq

### 5.4.3. Notification

업데이트할 데이터를 서버에 등록하여 서버에서 자동으로 업데이트 하는 방식이다.

- 변수 형태: All
- 접근 방식: By name, By address
- 이점: ADS Server 에서 데이터를 업데이트 시킨다.
- 단점: 필요 없는 높은 데이터 트래픽이 발생할 수 있다.
- 비고: 데이터 처리가 필요하지 않는 모니터링 시스템에 적합하다.

	Read	Write
<b>By name</b>	AdsReadBoolVarConnect AdsReadIntegerVarConnect AdsReadLongVarConnect AdsReadSingleVarConnect AdsReadDoubleVarConnect AdsReadStringVarConnect	AdsWriteBoolVarConnect AdsWriteIntegerVarConnect AdsWriteLongVarConnect AdsWriteSingleVarConnect AdsWriteDoubleVarConnect
<b>By address</b>	AdsReadBoolConnect AdsReadIntegerConnect AdsReadLongConnect AdsReadSingleConnect AdsReadDoubleConnect AdsReadStringConnect AdsReadBoolDisconnect AdsReadIntegerDisconnect AdsReadLongDisconnect AdsReadSingleDisconnect AdsReadDoubleDisconnect AdsReadStringDisconnect	AdsWriteIntegerConnect AdsWriteLongConnect AdsWriteSingleConnect AdsWriteDoubleConnect AdsWriteBoolDisconnect AdsWriteIntegerDisconnect AdsWriteLongDisconnect AdsWriteSingleDisconnect AdsWriteDoubleDisconnect

## 5.5. ADS Return Codes

### 5.5.1. Global Error Codes

Hex	Dec	Description
0x0	0	no error
0x1	1	Internal error
0x2	2	No Rtime
0x3	3	Allocation locked memory error
0x4	4	Insert mailbox error
0x5	5	Wrong receive HMSG
0x6	6	target port not found
0x7	7	target machine not found
0x8	8	Unknown command ID
0x9	9	Bad task ID
0xA	10	No IO
0xB	11	Unknown ADS command
0xC	12	Win 32 error
0xD	13	Port not connected
0xE	14	Invalid ADS length
0xF	15	Invalid ADS Net ID
0x10	16	Low Installation level
0x11	17	No debug available
0x12	18	Port disabled
0x13	19	Port already connected
0x14	20	ADS Sync Win32 error
0x15	21	ADS Sync Timeout
0x16	22	ADS Sync AMS error
0x17	23	ADS Sync no index map
0x18	24	Invalid ADS port
0x19	25	No memory
0x1A	26	TCP send error
0x1B	27	Host unreachable

### 5.5.2. General ADS Error Codes

Hex	Dec	Description
0x700	1792	error class <device error>
0x701	1793	Service is not supported by server
0x702	1794	invalid index group
0x703	1795	invalid index offset
0x704	1796	reading/writing not permitted
0x705	1797	parameter size not correct
0x706	1798	invalid parameter value(s)
0x707	1799	device is not in a ready state
0x708	1800	device is busy
0x709	1801	invalid context (must be in Windows)
0x70A	1802	out of memory
0x70B	1803	invalid parameter value(s)
0x70C	1804	not found (files, ...)
0x70D	1805	syntax error in command or file
0x70E	1806	objects do not match
0x70F	1807	object already exists
0x710	1808	symbol not found
0x711	1809	symbol version invalid
0x712	1810	server is in invalid state
0x713	1811	AdsTransMode not supported
0x714	1812	Notification handle is invalid
0x715	1813	Notification client not registered
0x716	1814	no more notification handles
0x717	1815	size for watch too big
0x718	1816	device not initialized
0x719	1817	device has a timeout
0x71A	1818	query interface failed
0x71B	1819	wrong interface required
0x71C	1820	class ID is invalid
0x71D	1821	object ID is invalid
0x71E	1822	request is pending
0x71F	1823	request is aborted
0x720	1824	signal warning
0x721	1825	invalid array index
0x722	1826	symbol not active

0x723	1827	access denied
0x724	1828	missing license
0x72C	1836	exception occurred during system start
0x740	1856	Error class <client error>
0x741	1857	invalid parameter at service
0x742	1858	polling list is empty
0x743	1859	var connection already in use
0x744	1860	invoke ID in use
0x745	1861	timeout elapsed
0x746	1862	error in win32 subsystem
0x747	1863	Invalid client timeout value
0x748	1864	ads-port not opened
0x750	1872	internal error in ads sync
0x751	1873	hash table overflow
0x752	1874	key not found in hash
0x753	1875	no more symbols in cache
0x754	1876	invalid response received
0x755	1877	sync port is locked

• 자세한 ADS Return Code 는 <http://infosys.beckhoff.com> 을 참고해 주시기 바랍니다.  
(<http://infosys.beckhoff.com> → TwinCAT2 → TwinCAT System → TwinCAT Connectivity  
→ TwinCAT ADS Sample → ADS Returncodes)

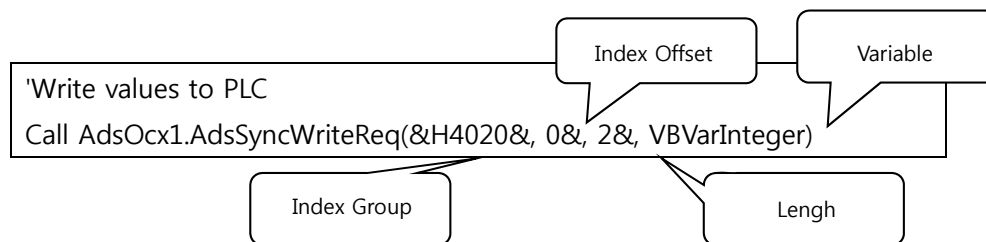
## 5.6. TwinCAT ADS-OCX

Microsoft 의 COM 기술을 기반으로 하는 Active X 컨트롤로 구현되었다. 모든 프로그램 환경에서 사용할 수 있으나, Active X 컨트롤은 비주얼 베이직에서 가장 많이 사용한다. 모든 통신 방식을 제공한다. (Synchronous, Asynchronous, Notification )

### 5.6.1. ADS-OCX 의 접근 형태별 분류

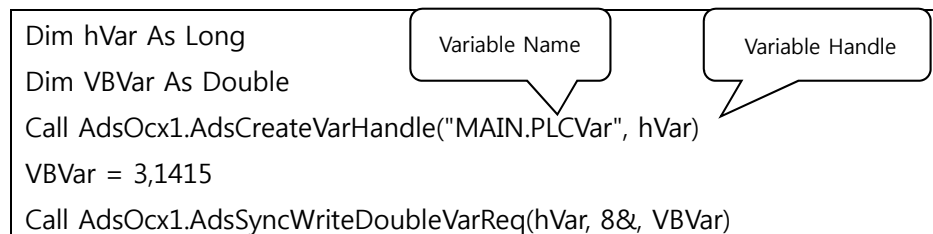
#### 1) 주소 기반.

- ADS 시스템에서 정의되어 있는 변수를 주소 기반으로 접근하는 방식.



#### 2) 이름 기반.

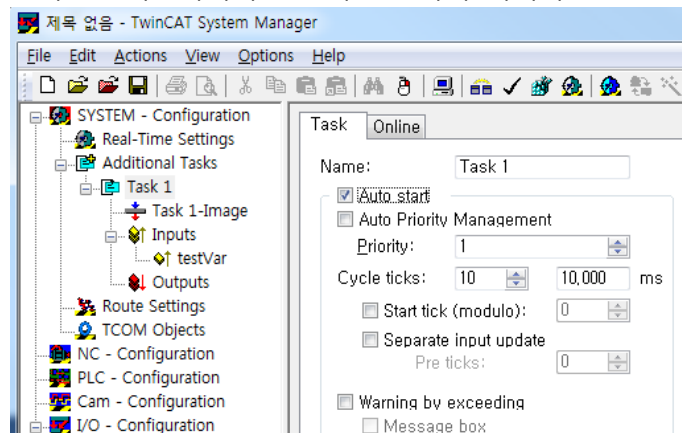
- ADS 시스템에서 정의되어 있는 변수를 이름 기반으로 접근하는 방식.



\* Index Group 의 식별자는 5.3.3 Index - Group/Offset 을 참고하기 바랍니다.

## 5.6.2. ADS OCX 프로그래밍 (Visual Basic)

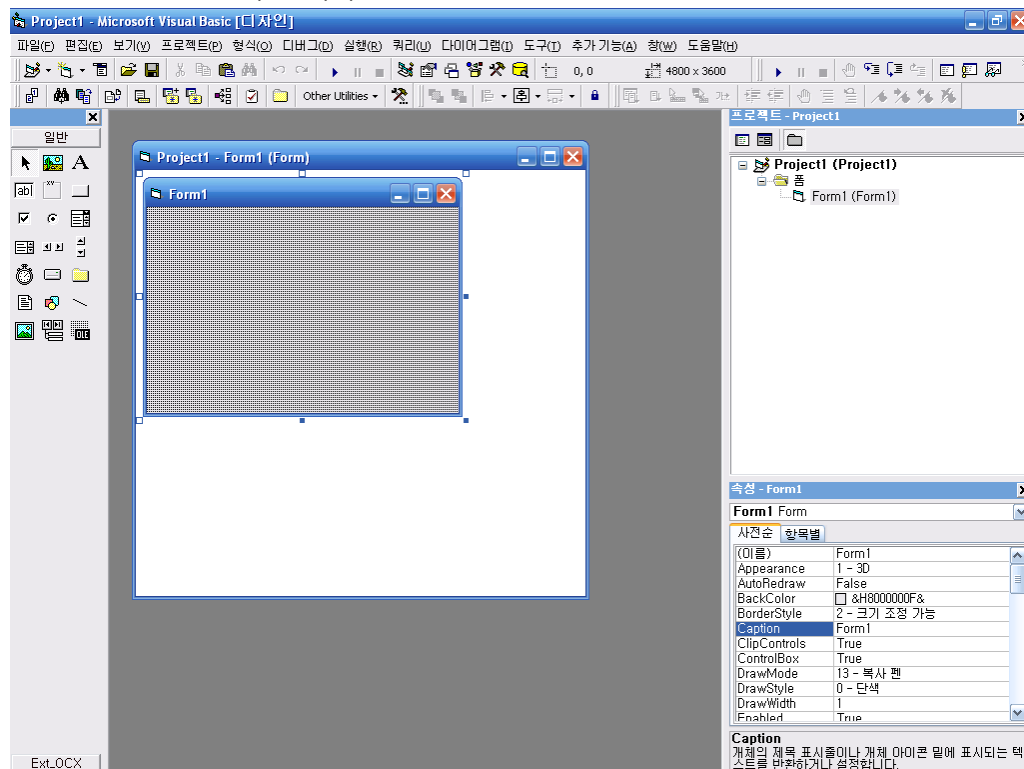
먼저, VB 어플리케이션을 테스트 하기 위해서는 TwinCAT I/O 를 Run 모드로 설정해야 한다.



- TwinCAT I/O 에서 'testVar' 변수 선언하고, TwinCAT System 을 Run 모드로 전환한다.

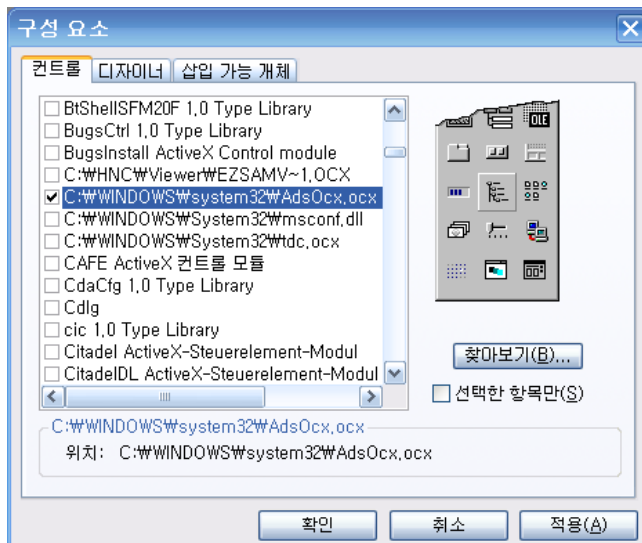
### 1) Synchronous – 이름 기반

- Visual Basic 프로젝트 시작.

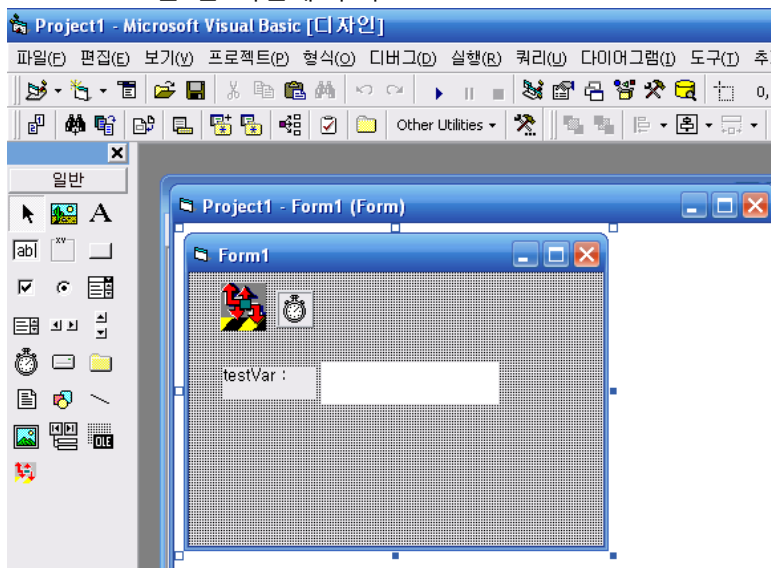




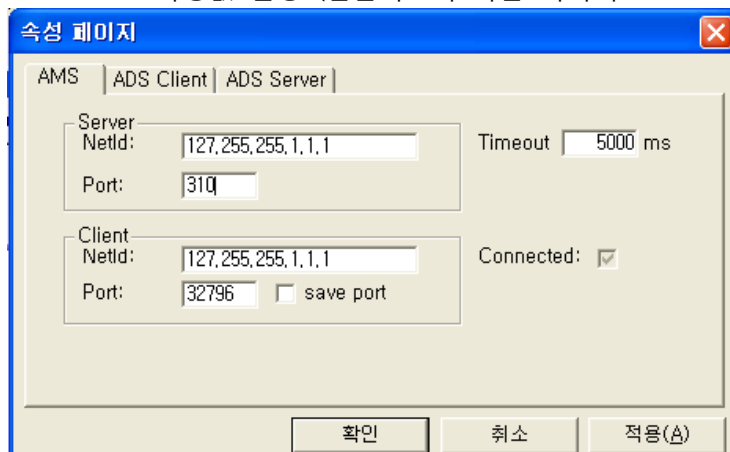
- ADS OCX 추가



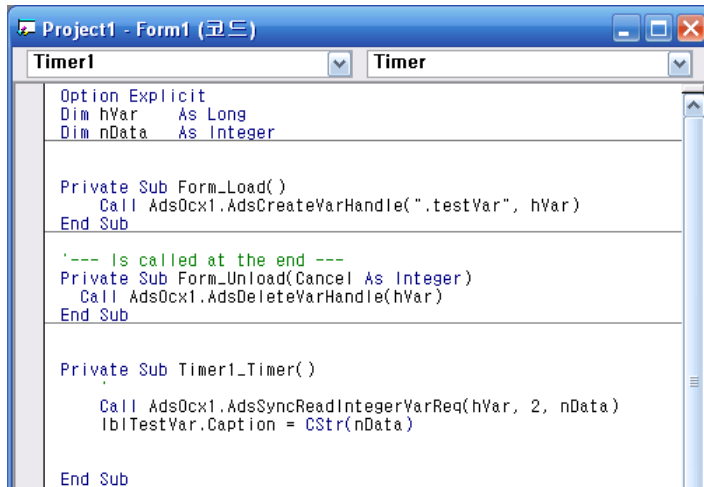
- ADS OCX 를 폼 화면에 추가.



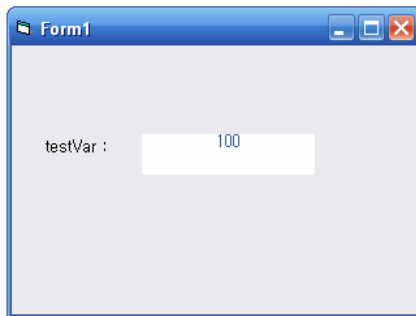
- ADS OCX 속성값 설정. (접근하고자 하는 서버의 NetID 와 Port 번호를 지정한다.)



- 프로그램을 작성한다.

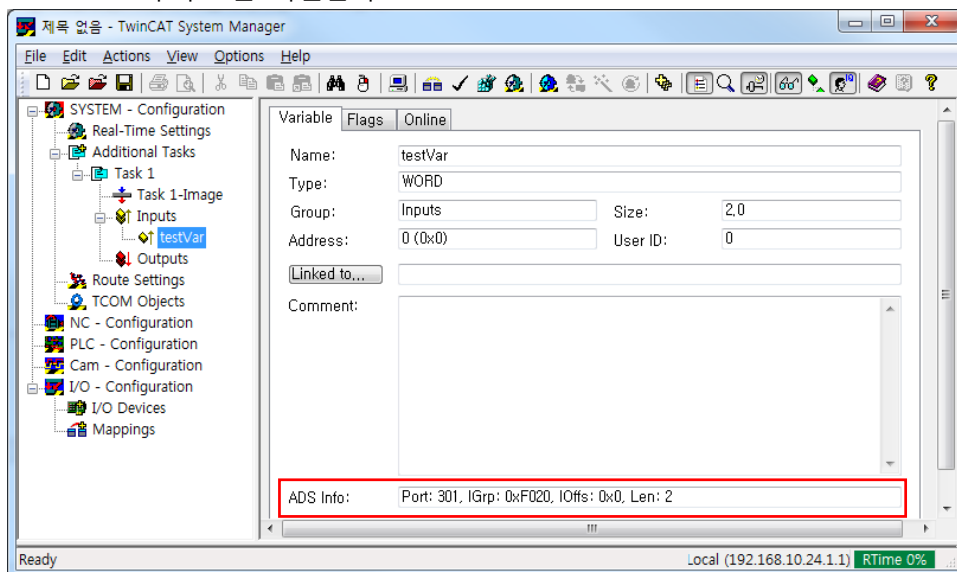


- 프로그램을 실행한다.

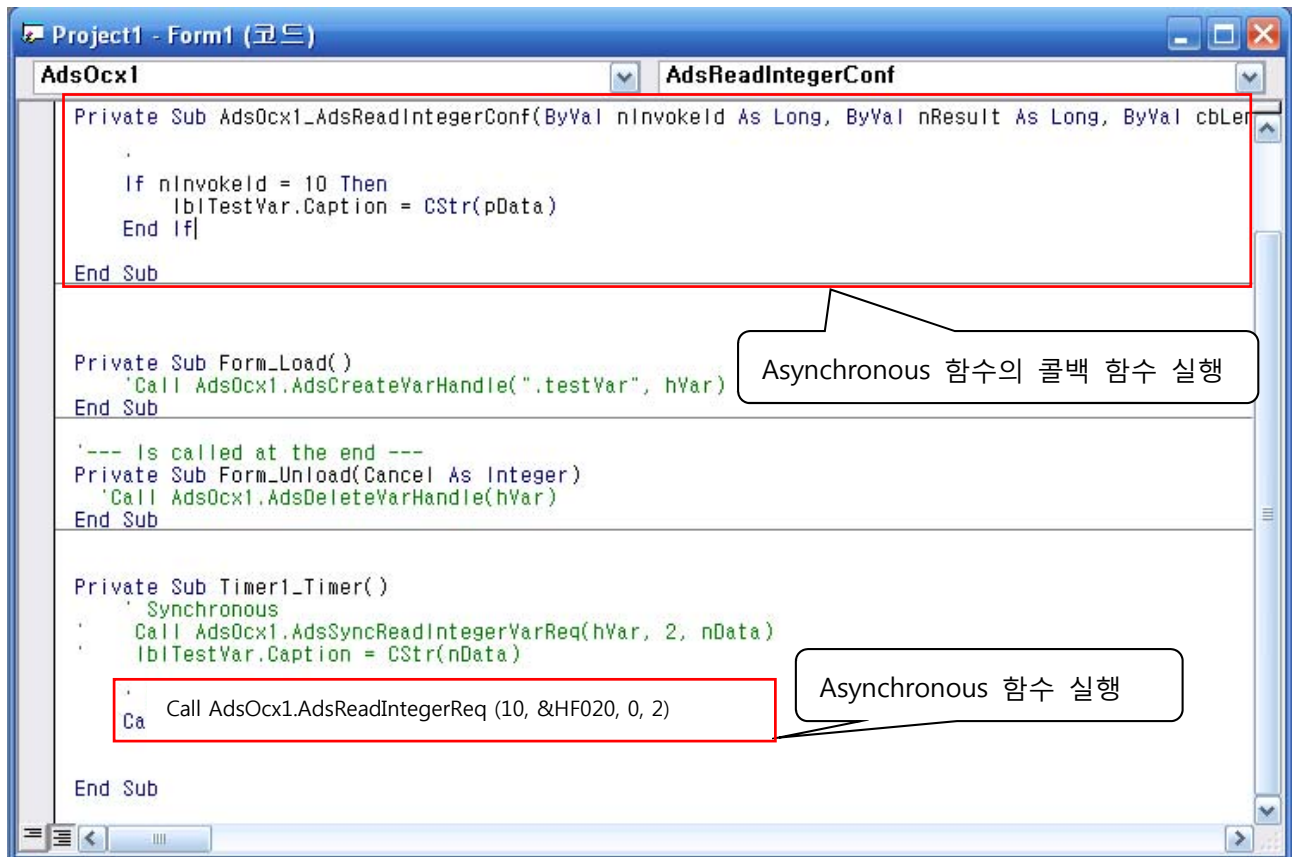


## 2) Asynchronous – 주소 기반

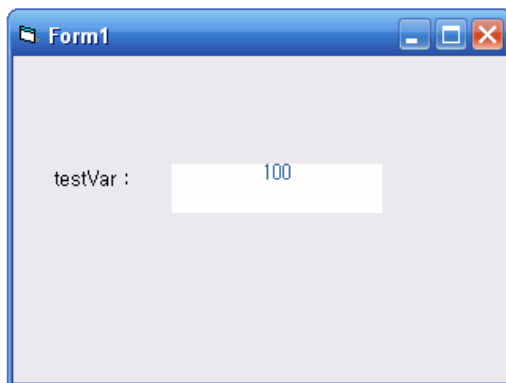
- 'testVar'의 주소를 확인한다.



- Visual Basic 프로그램 수정.



- 실행 화면.

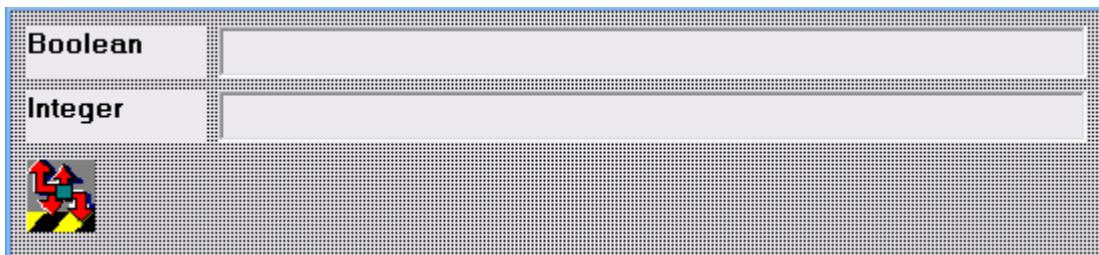


### 3) Notification

- TwinCAT I/O 변수

```
PLCVarBoolean : BOOL;  
PLCVarInteger : INT;
```

- VB Code



- 프로그램 로딩 및 PLC 변수 연결

```
Private Sub Form_Load()  
  
    Dim nErr As Long  
    nErr = AdsOcx1.AdsReadVarConnectEx(".PLCVarBoolean", _           // PLC 에서 선언된 변수 이름  
    If (nErr > 0) Then Call MsgBox("Error AdsReadVarConnectEx -> .PLCVarBoolean: " & nErr)  
    nErr = AdsOcx1.AdsReadVarConnectEx(".PLCVarInteger", _  
  
    If (nErr > 0) Then Call MsgBox("Error AdsReadVarConnectEx -> .PLCVarInteger: " & nErr)  
  
End Sub
```

- 이벤트 처리 함수

```
Private Sub AdsOcx1_AdsReadConnectUpdateEx(
    ByVal dateTime As Date,           ' Time stamp
    ByVal nMs As Long,               ' Milliseconds of time stamp
    ByVal hConnect As Long,          ' AdsReadVarConnectEx 에 의해 결정된 핸들
    ByVal data As Variant,           ' 값
    Optional ByVal hUser As Variant) ' 컴포넌트

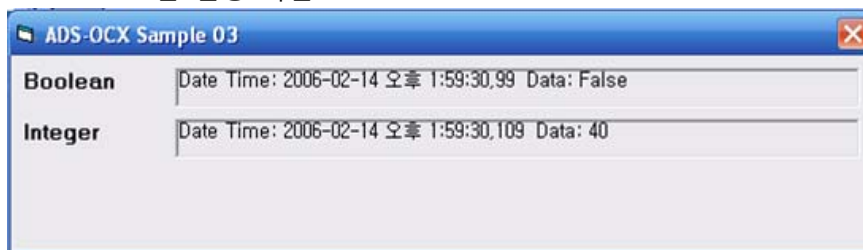
    hUser.Caption = ("Date Time: " & dateTime & ", " & nMs & " Data: " & data)

End Sub
```

- 프로그램 종료 및 연결된 PLC 변수 끊기

```
Private Sub Form_Unload(Cancel As Integer)
    Dim nIndex As Long
    For nIndex = 0 To 6
        Call AdsOcx1.AdsDisconnectEx(hConnect(nIndex))
    Next
End Sub
```

- VB 프로그램 실행 화면



- \* 주의 : AdsReadVarConnectEx() 의 사이클 타임을 목적에 맞게 잘 설정해야 한다. 모든 변수에 짧은 사이클 타임을 적용하면 사용자 어플리케이션이 느려진다.

- 지금까지 Visual Basic 에서 ADS OCX 에 대한 간단한 프로그램을 작성해 보았다.  
더 많은 자료는 [Infosys.beckhoff.com](http://infosys.beckhoff.com) 홈페이지에 있다.
- [http://infosys.beckhoff.com/content/1033/tcadsocx/html/tcadsocx\\_intro.htm](http://infosys.beckhoff.com/content/1033/tcadsocx/html/tcadsocx_intro.htm)

## 5.7. TwinCAT ADS-DLL

ADS DLL 에서 지원하는 연결은 'Synchronous' 방식과 'Notification' 방식을 지원한다.

ADS DLL API	설 명
AdsGetDllVersion	Returns the version number, revision number and build number of the ADS-DLL.
AdsPortOpen	Establishes a connection (communication port) to the TwinCAT message router.
AdsPortClose	The connection (communication port) to the TwinCAT message router is closed.
AdsGetLocalAddress	Returns the local NetId and port number.
AdsSyncWriteReq	Writes data synchronously to an ADS device.
AdsSyncReadReq	Reads data synchronously from an ADS server.
AdsSyncReadReqEx	Reads data synchronously from an ADS server.
AdsSyncReadWriteReq	Writes data synchronously into an ADS server and receives data back from the ADS device.
AdsSyncReadWriteReqEx	Writes data synchronously into an ADS server and receives data back from the ADS device.
AdsSyncReadDeviceInfoReq	Reads the identification and version number of an ADS server.
AdsSyncWriteControlReq	Changes the ADS status and the device status of an ADS server.
AdsSyncReadStateReq	Reads the ADS status and the device status from an ADS server.
AdsSyncAddDeviceNotificationReq	A notification is defined within an ADS server (e.g. PLC). When a certain event occurs a function (the callback function) is invoked in the ADS client (C program).
AdsSyncDelDeviceNotificationReq	A notification defined previously is deleted from an ADS server.
AdsSyncSetTimeout	Alters the timeout for the ADS functions. The standard value is 5000ms.
AdsAmsRegisterRouterNotification	The AdsAmsRegisterNotificationReq() function can be used to detect a change in the status of the TwinCAT router. The given callback function is invoked each time the status changes. Monitoring of the router's status is ended once more by the <a href="#">AdsAmsUnRegisterNotification()</a> function.
AdsAmsUnRegisterRouterNotification	Monitoring the router's status is ended by the AdsAmsUnRegisterNotification() function. See also <a href="#">AdsAmsRegisterNotificationReq()</a> .
PAmsRouterNotificationFuncEx	Type definition of the callback function required by the <a href="#">AdsAmsRegisterRouterNotification</a> function.
PAdsNotificationFuncEx	Type definition of the callback function required by the <a href="#">AdsSyncAddDeviceNotificationReq</a> function.

### 5.7.1. ADS-DLL 프로그래밍 방법

필요한 파일

TcAdsDll.dll - dynamic function library

TcAdsDll.lib - link library for the TcAdsDll.dll

TcAdsAPI.h - declarations of the ADS functions

TcAdsDef.h - declarations of the structures and constants

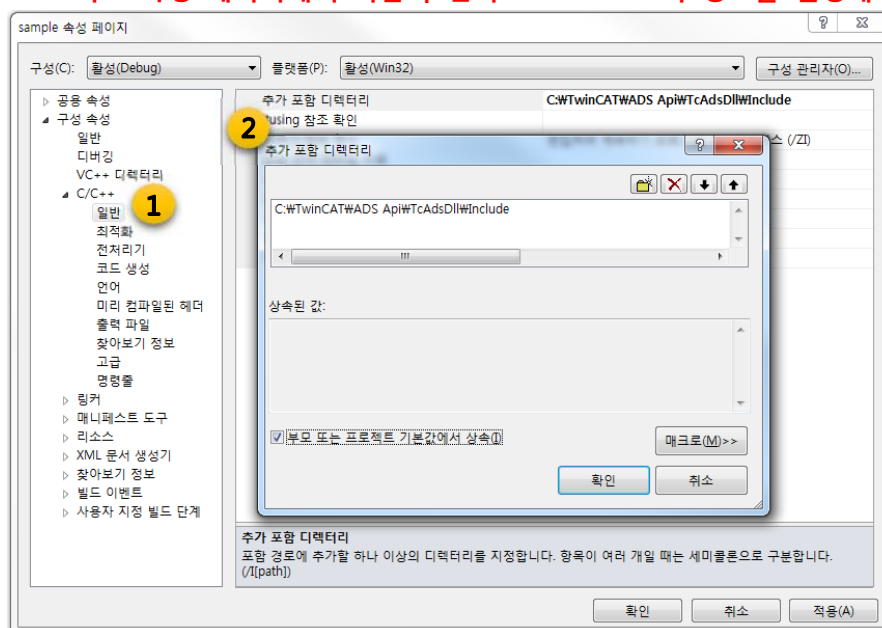
#### 5.7.1.1. Header files 추가

TcAdsDll 의 함수들을 사용하기 위해서는 TcAdsAPI.h 와 TcAdsDef.h 파일을 포함해야 한다.

```
#include "C:\TwinCAT\ADS Api\TcAdsDll\Include\TcAdsDef.h"
#include "C:\TwinCAT\ADS Api\TcAdsDll\Include\TcAdsAPI.h"
```

TcAdsDll.dll 은 시스템 폴더에 설치되어 있다.

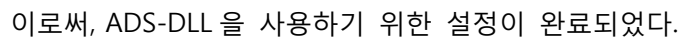
**\*프로젝트 속성 페이지에서 다음과 같이 Header file 의 경로를 설정해주면**



**\*코딩 창에서 Header file 의 경로를 작성할 필요가 없다.**

```
#include "TcAdsDef.h"
#include "TcAdsAPI.h"
```

(\*AdsDll.lib 는 C:\TwinCAT\ADS Api\TcAdsDll\Lib 폴더에 있다.\*)





## 5.7.2. 샘플 소스

예제①. 'Synchronous' 방식으로 Write 하는 Sourcecode

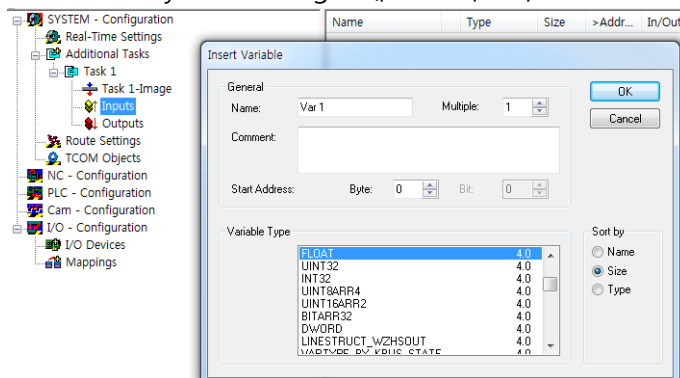
사용자가 입력하는 실수 값을 TwinCAT System Manager 의 Additional Tasks 에 있는 Input 변수의 값에 입력되는 소스코드이다

예제②. 'Synchronous' 방식으로 Read 하는 Sourcecode

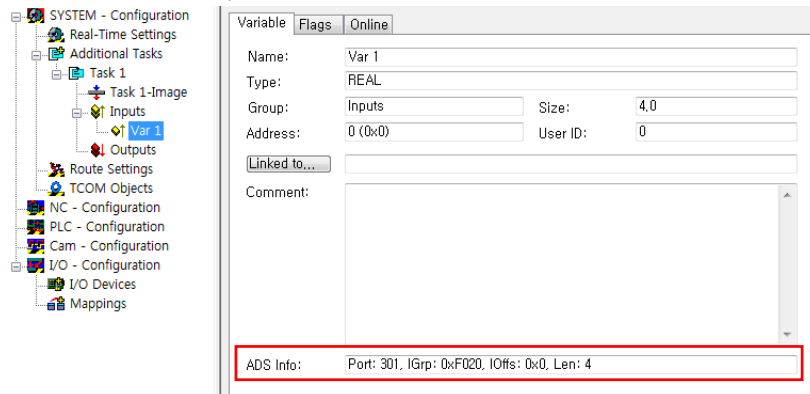
Real 타입의 Output 변수의 값을 Visual Studio 실행 창에서 읽어올 수 있는 소스코드이다.

**\*Visual C++와 ADS 를 통해 값을 Read/Write 할 데이터를 생성한다.**

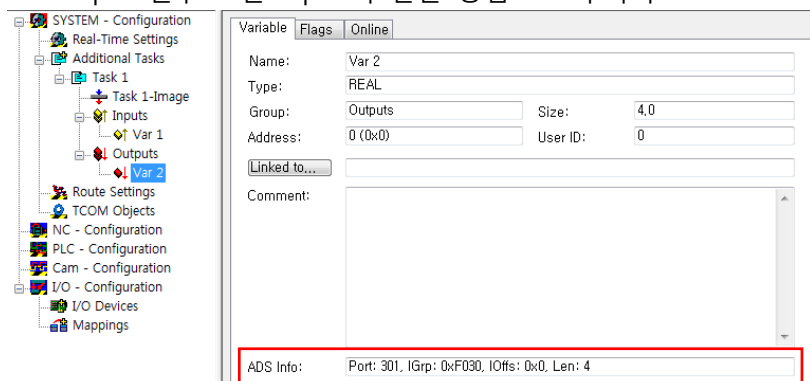
- TwinCAT System Manager 새 프로젝트의 Additional Tasks -Input 에 Float 타입의 변수를 만든다.



- 그 변수의 Group, Offset, size 를 확인한다.



- Output 변수 또한 Input 과 같은 방법으로 추가하고 ADS Info 를 확인한다.



### 5.7.2.1. 'Synchronous' 방식으로 Write 하는 Sourcecode

```
#include <iostream>
#include <windows.h>
#include <conio.h>
#include "c:\twincat\wads api\wtcadsdll\include\wtcadsdef.h"
#include "c:\twincat\wads api\wtcadsdll\include\wtcadsapi.h"

using namespace std;
void main()
{
    long            nErr, nPort;
    AmsAddr         Addr;
    PAmsAddr        pAddr = &Addr;
    FLOAT           fData;

    // ADS 라우터의 통신 포트를 연다.
    nPort = AdsPortOpen();
    nErr = AdsGetLocalAddress(pAddr);
    if (nErr) cerr << "Error: AdsGetLocalAddress: " << nErr << '\n';
    pAddr->port = 301;

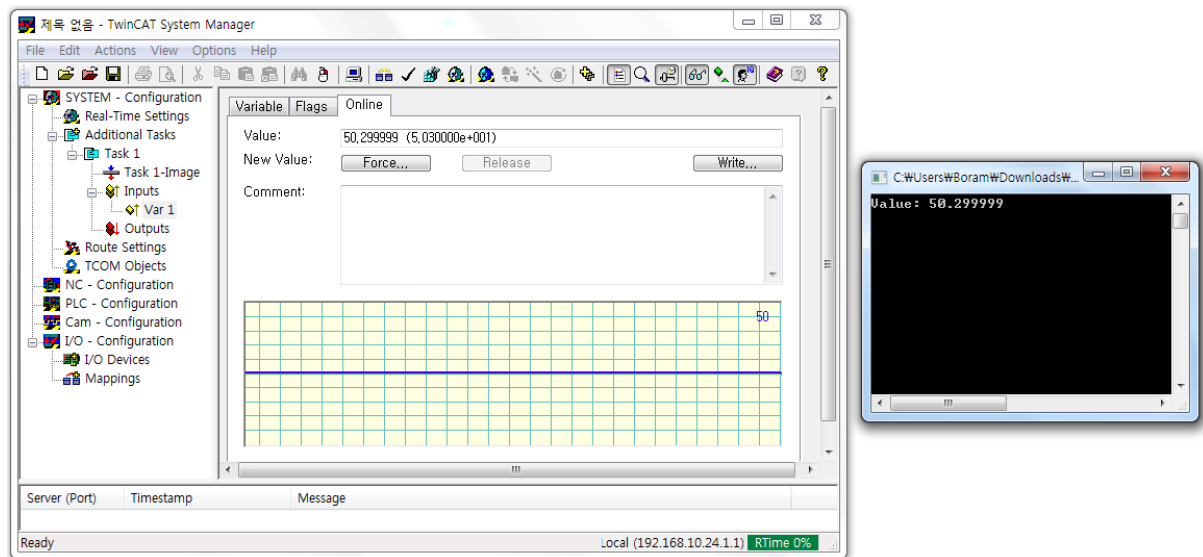
    // Additional Tasks의 Input 데이터에 입력될 값을 입력한다.
    cout << "Value: ";
    cin >> fData;

    // ADS 식별자를 통해 Additional Tasks의 Input 데이터에 값을 Write한다.
    nErr = AdsSyncWriteReq( pAddr, 0xF020, 0x0, 4, &fData );
    if (nErr) cerr << "Error: AdsSyncWriteReq: " << nErr << '\n';

    // Wait for key
    _getch();

    // 통신 포트를 닫는다.
    nErr = AdsPortClose();
    if (nErr) cerr << "Error: AdsPortClose: " << nErr << '\n';
}
```

## 실행 화면



## 함수 설명

- LONG AdsPortOpen( void ) // 로컬 컴퓨터의 사용 가능한 Port 를 연다.
- LONG AdsPortClose( void ) // Port 를 닫는다.
- LONG AdsGetLocalAddress( PAdsAddr pAddr ) // 로컬 컴퓨터의 AmsNetId 를 가져온다.

LONG AdsSyncWriteReq(

<a href="#">PAdsAddr</a>	pAddr,	[in] Structure with NetId and port number of the ADS server.
ULONG	nIndexGroup,	[in] Index Group.
ULONG	nIndexOffset,	[in] Index Offset.
ULONG	nLength,	[in] Length of the data in bytes.
PVOID	pData	[out] Pointer to a data buffer that will receive the data.

);

pAddr : NetID 와 Port 구조체 , = AdsGetLocalAddress 함수에 의해 자동적으로 읽어옴  
 0xF020 : Input 이미지를 배열로 읽고자 할 때 사용되는 그룹 인덱스.  
 ("Index-Group/Offset" Specification of the TwinCAT ADS system services 참조 )  
 4 : 이미지 사이즈  
 &fData : Input 이미지를 읽고 저장할 공간.

pAddr->port = 301; // Port 번호를 지정한다. (TwinCAT IO Port : 301)

### 5.7.2.2. 'Synchronous' 방식으로 Read 하는 Sourcecode

```
#include <iostream>
#include <conio.h>
#include <windows.h>
#include "c:\twincat\wadsapi\wtcadsdll\include\wtcadsdef.h"
#include "c:\twincat\wadsapi\wtcadsdll\include\wtcadsapi.h"

using namespace std;

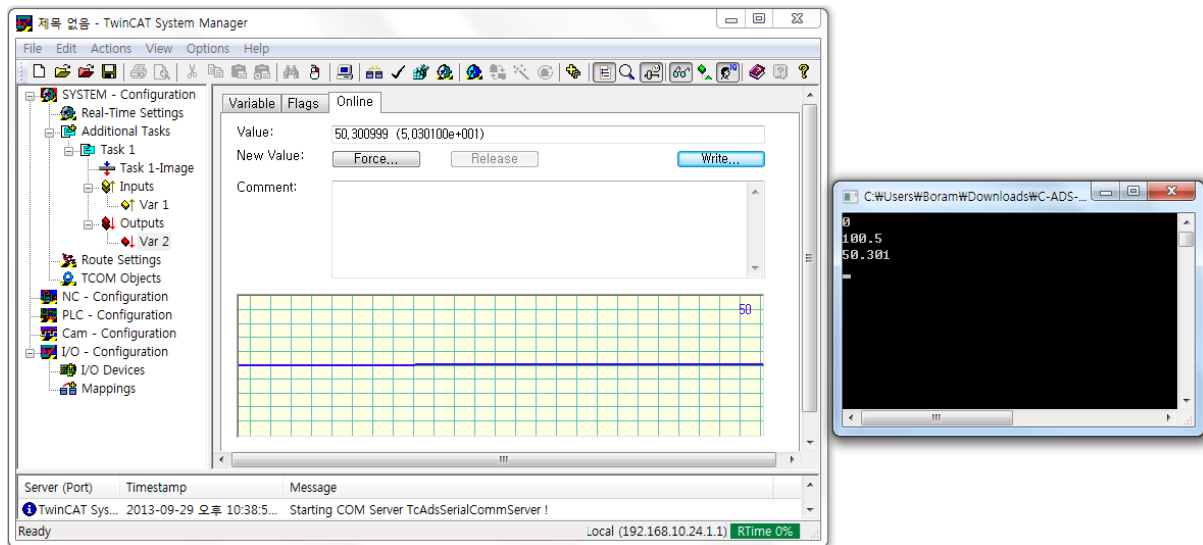
void main()
{
    Long          nErr, nPort;
    AmsAddr        Addr;
    PAmsAddr       pAddr = &Addr;
    FLOAT          fData;

    // ADS 라우터의 통신 포트를 연다.
    nPort = AdsPortOpen();
    nErr = AdsGetLocalAddress(pAddr);
    if (nErr) cerr << "Error: AdsGetLocalAddress: " << nErr << '\n';
    pAddr->port = 301;

    // ADS 식별자를 통해 Additional Tasks의 Output 데이터 값을 Read한다
    do
    {
        nErr = AdsSyncReadReq(pAddr, 0xF030, 0x0, 4, &fData);
        if (nErr) cerr << "Error: AdsSyncReadReq: " << nErr << '\n';
        else cout << fData << '\n';
        cout.flush();
    }
    while (getch() == 'Wr');    // Read next value (Carriage return is delimiter) , stop reading otherwise

    // 통신 포트를 닫는다
    nErr = AdsPortClose();
    if (nErr) cerr << "Error: AdsPortClose: " << nErr << '\n';
```

## 실행 화면



## 함수 설명

LONG AdsSyncReadReq(

<a href="#">PAmsAddr</a>	pAddr,	[in] Structure with NetId and port number of the ADS server.
ULONG	nIndexGroup,	[in] Index Group.
ULONG	nIndexOffset,	[in] Index Offset.
ULONG	nLength,	[in] Length of the data in bytes.
PVOID	pData	[out] Pointer to a data buffer that will receive the data.

)

pAddr : NetID 와 Port 구조체 , = AdsGetLocalAddress 함수에 의해 자동적으로 읽어옴  
 0xF030 : Output 이미지를 배열로 읽고자 할 때 사용되는 그룹 인덱스.  
 ("Index-Group/Offset" Specification of the TwinCAT ADS system services 참조 )  
 4 : 이미지 사이즈  
 &fData : Output 이미지를 읽고 저장할 공간.

### 5.7.2.3. 'Notification' 방식 – Event driven reading

#### C++ program

```
#include <iostream>
#include <conio.h>
#include <windows.h>
#include <winbase.h>
#include "C:\TwinCAT\WAds Api\WtCadsDll\Include\WtCadsDef.h"
#include "C:\TwinCAT\WAds Api\WtCadsDll\Include\WtCadsApi.h"

using namespace std;

// 이벤트가 발생했을 때 처리할 콜백함수 선언
void _stdcall Callback(AmsAddr*, AdsNotificationHeader*, unsigned long);

void main()
{
    // 기타 필요한 변수들을 선언
    long                nErr, nPort;
    AmsAddr              Addr;
    PAmsAddr             pAddr = &Addr;
    ULONG                hNotification, hUser;
    AdsNotificationAttrib adsNotificationAttrib;
    char                 szVar []={"MAIN.PlcVar"};

    // ADS 라우터 포트 오픈
    nPort = AdsPortOpen();
    nErr = AdsGetLocalAddress(pAddr);
    if (nErr) cerr << "Error: AdsGetLocalAddress: " << nErr << '\n';
    pAddr->port = 801;

    // 모니터링할 변수들 지정
    adsNotificationAttrib.cbLength = 4; // 데이터 변수 길이
    adsNotificationAttrib.nTransMode = ADSTRANS_SERVERONCHA; // 업데이트 방식
    adsNotificationAttrib.nMaxDelay = 0;
    adsNotificationAttrib.nCycleTime = 10000000; // 값을 체크하는 주기 1sec

    // Get variable handle
    nErr = AdsSyncReadWriteReq(pAddr, ADSIGRP_SYM_HNDBYNAME, 0x0, sizeof(hUser), &hUser, sizeof(szVar), szVar);
    if (nErr) cerr << "Error: AdsSyncReadWriteReq: " << nErr << '\n';

    // Initiate the transmission of the PLC-variable
    nErr = AdsSyncAddDeviceNotificationReq(pAddr, ADSIGRP_SYM_VALBYHND, hUser, &adsNotificationAttrib,
    Callback, hUser, &hNotification);

    if (nErr) cerr << "Error: AdsSyncAddDeviceNotificationReq: " << nErr << '\n';
    cout.flush();

    // Wait for user intraction (keystroke)
    _getch();
}
```

```

// Finish the transmission of the PLC-variable
nErr = AdsSyncDelDeviceNotificationReq(pAddr, hNotification);
if (nErr) cerr << "Error: AdsSyncDelDeviceNotificationReq: " << nErr << '\n';

// Release the variable handle
nErr = AdsSyncWriteReq(pAddr, ADSIGRP_SYM_RELEASEHND, 0, sizeof(hUser), &hUser);
if (nErr) cerr << "Error: AdsSyncWriteReq: " << nErr << '\n';

// 프로그램 종료부분에서 통신 포트 연결을 끊는다.
nErr = AdsPortClose();
if (nErr) cerr << "Error: AdsPortClose: " << nErr << '\n';
}

// 이벤트가 발생하였을 때 실행되는 콜백 함수
void __stdcall Callback(AmsAddr* pAddr, AdsNotificationHeader* pNotification, ULONG hUser)
{
    int                nIndex;
    static ULONG       nCount = 0;
    SYSTEMTIME         SystemTime, LocalTime;
    FILETIME           FileTime;
    LARGE_INTEGER       LargeInteger;
    TIME_ZONE_INFORMATION TimeZoneInformation;

    cout << ++nCount << ". Notification:\n";

    // print (to screen)) the value of the variable
    cout << "Value: " << *(ULONG *)pNotification->data << '\n';
    cout << "Notification: " << pNotification->hNotification << '\n';

    // Convert the timestamp into SYSTEMTIME
    LargeInteger.QuadPart = pNotification->nTimeStamp;
    FileTime.dwLowDateTime = (DWORD)LargeInteger.LowPart;
    FileTime.dwHighDateTime = (DWORD)LargeInteger.HighPart;
    FileTimeToSystemTime(&FileTime, &SystemTime);

    // Convert the time value Zeit to local time
    GetTimeZoneInformation(&TimeZoneInformation);
    SystemTimeToTzSpecificLocalTime(&TimeZoneInformation, &SystemTime, &LocalTime);

    // Print out the timestamp
    cout << LocalTime.wHour << ":" << LocalTime.wMinute << ":" << LocalTime.wSecond << "." << LocalTime.wMilliseconds;

    // Print out the ADS-address of the sender
    cout << "\nServerNetId: ";
    for (nIndex = 0; nIndex < 6; nIndex++)
        cout << (int)pAddr->netId.b[nIndex] << ".";
    cout << " Port: " << pAddr->port << "\n\n";
    cout.flush();
}

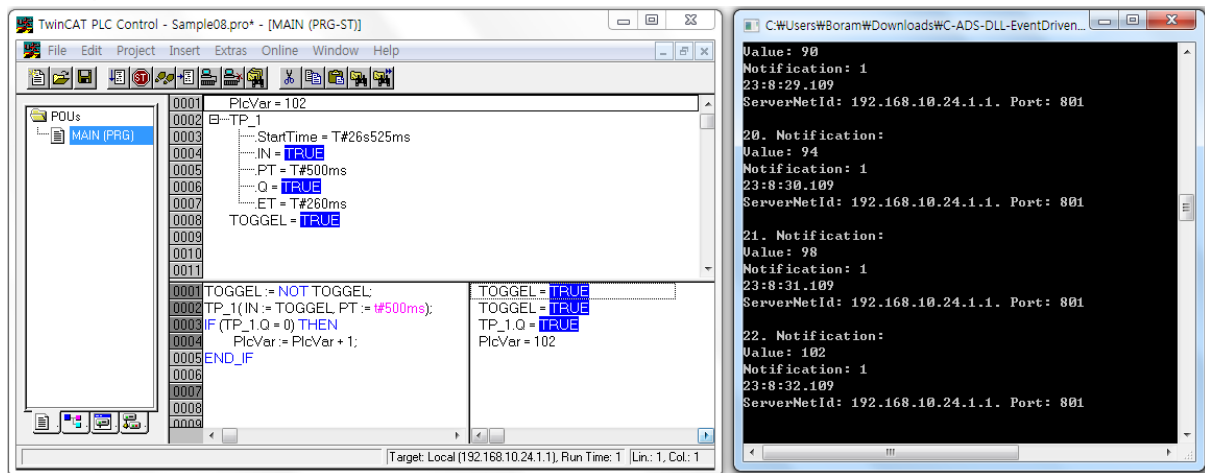
```

## PLC program

```
PROGRAM MAIN
VAR
    PlcVar AT %M* : DWORD;
    TP_1          : TP;
    TOGGEI: BOOL;
END_VAR
```

```
TOGGEI := NOT TOGGEI;
TP_1(IN := TOGGEI, PT := t#500ms);
IF (TP_1.Q = 0) THEN
    PlcVar := PlcVar + 1;
END_IF
```

## 실행 화면



## 함수 설명

### AdsSyncAddDeviceNotificationReq

(pAddr, : ADS Net ID  
0x4020, : Index Group – PLC 메모리 주소(M)  
0, : Index Offset  
&adsNotificationAttrib,  
Callback, : 콜백 함수  
hUser, : 식별자  
&hNotification);



## 5.8. TwinCAT ADS.NET

Microsoft Visual Studio 를 시작해서 새 프로젝트를 생성한다. (Windows Forms Application)

### 5.8.1. ADS.NET 프로그래밍 방법

#### 5.8.1.1. TwinCAT.Ads Dll

TwinCAT.Ads Dll 은 .NET class library 로써 ADS device 와 통신할 수 있는 class 들을 제공한다.

.NET framework version 1.0(Microsoft Visual Studio 2002)은

...₩TwinCAT₩ADS Api₩.NET₩v1.0.3705 폴더에 있는 TwinCAT.Ads.dll 을 사용한다.

.NET framework version 1.1(Microsoft Visual Studio 2003/Borland Developer Studio 2006)은

...₩TwinCAT₩ADS Api₩.NET₩v1.1.4322 폴더에 있는 TwinCAT.Ads.dll 을 사용한다.

.NET Compact Framework version 1.0 ( CE)은

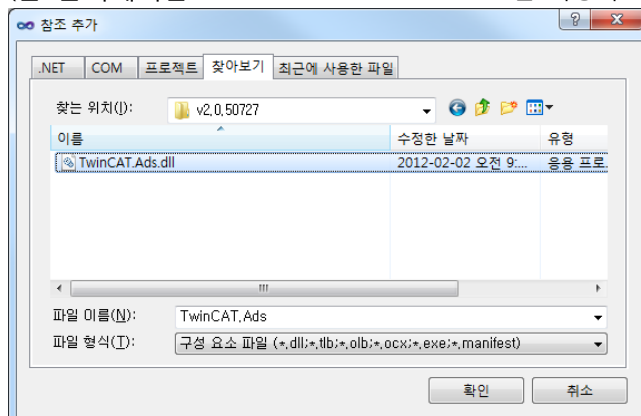
...₩TwinCAT₩ADS Api₩CompactFramework 폴더에 있는 TwinCAT.Ads.dll 을 사용한다.

#### 5.8.1.2. Add Reference

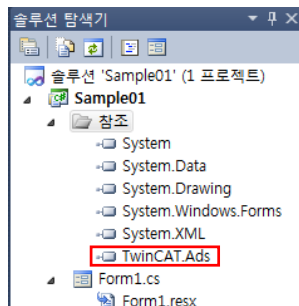
프로젝트 메뉴에서 참조추가를 선택한다.

찾아보기에서 .NET framework 버전에 맞는 TwinCAT.Ads.dll 을 추가한다.

(본 문서에서는 Microsoft Visual Studio 2010 을 사용하므로 v2.0.50727 을 사용하였다.)



솔루션 탐색기의 참조 폴더 아래에 TwinCAT.Ads 가 추가된 것을 확인할 수 있다.



TwinCAT.Ads 에 있는 모든 클래스와 구조체를 사용하기 위해 아래 코드를 선언해준다.

```
Using TwinCAT.Ads;  
Using System.IO;
```

## 5.8.2. 샘플소스

### 5.8.2.1. 'Synchronous' 방식 – Reading and writing of string variables

TwinCAT PLC Control로부터 string 형의 데이터 값을 읽고 쓸 수 있는 샘플소스이다.

#### C# program

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;
using TwinCAT.Ads;           // 'TwinCAT.Ads' Reference 추가
using System.IO;

namespace Sample04 {
    public class Form1 : System.Windows.Forms.Form {
        private System.ComponentModel.Container components = null;
        private System.Windows.Forms.TextBox textBox1;
        private System.Windows.Forms.Button btnRead;
        private System.Windows.Forms.Button btnWrite;
        private System.Windows.Forms.Label label1;
        private TcAdsClient adsClient;
        private int varHandle;

        public Form1() {
            InitializeComponent();
        }

        protected override void Dispose( bool disposing ) {
            ...
        }

        Private void InitializeComponent() {
            ...           // Windows Form Designer generated code
        }

        [STAThread]
        static void Main() {
            Application.Run(new Form1());
        }
    }
}
```

```

private void Form1_Load(object sender, System.EventArgs e)
{
    Try
    {
        adsClient = new TcAdsClient();
        // ADS Router Port open
        // (PLC1 Port: TrinCAT2=801, TwinCAT3=851)
        adsClient.Connect(801);
        // 'TwinCAT PLC Control' Main의 text 변수를 핸들로 생성한다.
        varHandle = adsClient.CreateVariableHandle("MAIN.text");
    }

    catch( Exception err)
    {
        MessageBox.Show(err.Message);
    }
}

// 프로그램 종료부분에서 통신 포트 연결을 끊는다.
private void Form1_Closing(object sender, System.ComponentModel.CancelEventArgs e)
{
    adsClient.Dispose();
}

// PLC Main.text 데이터 값을 Read한다.
private void btnRead_Click(object sender, System.EventArgs e) {
    try {
        AdsStream adsStream = new AdsStream(30);
        AdsBinaryReader reader = new AdsBinaryReader(adsStream);

        adsClient.Read(varHandle, adsStream);
        textBox1.Text = reader.ReadPlcString(30);
    }
    catch(Exception err) {
        MessageBox.Show(err.Message);
    }
}

```

```
// text창에 입력한 값을 PLC Main.text 데이터에 Write한다.
private void btnWrite_Click(object sender, System.EventArgs e)
{
    try {
        AdsStream adsStream = new AdsStream(30);
        AdsBinaryWriter writer = new AdsBinaryWriter(adsStream);

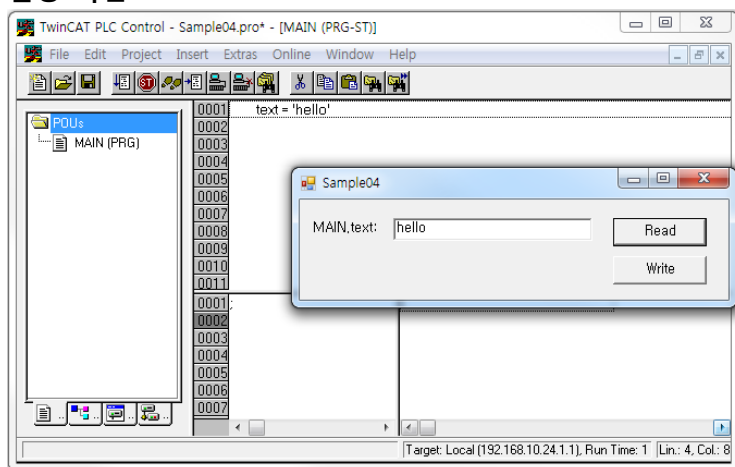
        writer.WritePlcString(textBox1.Text, 30);

        adsClient.Write(varHandle, adsStream);
    }
    catch(Exception err)
    {
        MessageBox.Show(err.Message);
    }
}
}
```

## PLC program

```
PROGRAM MAIN
VAR
    text : STRING[30] := 'hello';
END_VAR
```

## 실행 화면



### 5.8.2.2. 'Notification' 방식 – Event driven reading

PLC 에 있는 7 개의 전역변수는 서로 다른 타입의 데이터 형이다. 각 데이터 값이 바뀌는 이벤트가 발생하면 콜백함수가 실행된다. Visual Studio 실행창에 변한 데이터 값이 타임스탬프와 함께 자동으로 표시된다.

#### C# program

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;
using TwinCAT.Ads;
using System.IO;

namespace Sample03 {

    // Summary description for Form1.
    public class Form1 : System.Windows.Forms.Form {
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.Label label3;
        private System.Windows.Forms.Label label4;
        private System.Windows.Forms.Label label5;
        private System.Windows.Forms.Label label6;
        private System.Windows.Forms.Label label7;
        private System.Windows.Forms.Label label8;
        private System.Windows.Forms.Label label9;
        private System.Windows.Forms.Label label10;
        private System.Windows.Forms.TextBox tbInt;
        private System.Windows.Forms.TextBox tbDint;
        private System.Windows.Forms.TextBox tbSint;
        private System.Windows.Forms.TextBox tbLreal;
        private System.Windows.Forms.TextBox tbReal;
        private System.Windows.Forms.TextBox tbString;
        private System.Windows.Forms.Label label11;
        private System.Windows.Forms.TextBox tbBool;

        // Required designer variable.
        private System.ComponentModel.Container components = null;
        private TcAdsClient tcClient;
        private int[] hConnect;
        private AdsStream dataStream;
        private System.Windows.Forms.Label label11;
        private System.Windows.Forms.TextBox tbBool;
        private BinaryReader binRead;
```

```

public Form1() {
    InitializeComponent();
}

protected override void Dispose( bool disposing ) {
    ...
}

private void InitializeComponent() {
    ...    // Windows Form Designer generated code
}

// The main entry point for the application.
[STAThread]
static void Main() {
    Application.Run(new Form1());
}

private void Form1_Load(object sender, System.EventArgs e) {
    dataStream = new AdsStream(31);
    //Encoding is set to ASCII, to read strings
    binRead = new BinaryReader(dataStream, System.Text.Encoding.ASCII);
    tcClient = new TcAdsClient();    // Creaset instance of class TcAdsClient
    tcClient.Connect(801);          // PLC1 Port: TwinCAT2=801, TwinCAT3=851

    hConnect = new int[7];          // 7개 변수에 대한 배열 생성
    try {
        hConnect[0] = tcClient.AddDeviceNotification("MAIN.boolVal",dataStream,0,1,
            AdsTransMode.OnChange,100,0,tbBool);
        hConnect[1] = tcClient.AddDeviceNotification("MAIN.intVal",dataStream,1,2,
            AdsTransMode.OnChange,100,0,tbInt);
        hConnect[2] = tcClient.AddDeviceNotification("MAIN.dintVal",dataStream,3,4,
            AdsTransMode.OnChange,100,0,tbDint);
        hConnect[3] = tcClient.AddDeviceNotification("MAIN.sintVal",dataStream,7,1,
            AdsTransMode.OnChange,100,0,tbSint);
        hConnect[4] = tcClient.AddDeviceNotification("MAIN.lrealVal",dataStream,8,8,
            AdsTransMode.OnChange,100,0,tbLreal);
        hConnect[5] = tcClient.AddDeviceNotification("MAIN.realVal",dataStream,16,4,
            AdsTransMode.OnChange,100,0,tbReal);
        hConnect[6] = tcClient.AddDeviceNotification("MAIN.stringVal",dataStream,20,11,
            AdsTransMode.OnChange,100,0,tbString);

        tcClient.AdsNotification += new AdsNotificationEventHandler(OnNotification);
    }
}

```

```

        catch(Exception err) {
            MessageBox.Show(err.Message);
        }
    }

    // PLC 변수가 바뀌었을 때 실행되는 함수
    private void OnNotification(object sender, AdsNotificationEventArgs e) {
        DateTime time = DateTime.FromFileTime(e.TimeStamp);
        e.DataStream.Position = e.Offset;
        string strValue = "";

        if( e.NotificationHandle == hConnect[0])
            strValue = binRead.ReadBoolean().ToString();
        else if( e.NotificationHandle == hConnect[1] )
            strValue = binRead.ReadInt16().ToString();

        else if( e.NotificationHandle == hConnect[2] )
            strValue = binRead.ReadInt32().ToString();

        else if( e.NotificationHandle == hConnect[3] )
            strValue = binRead.ReadByte().ToString();

        else if( e.NotificationHandle == hConnect[4] )
            strValue = binRead.ReadDouble().ToString();

        else if( e.NotificationHandle == hConnect[5] )
            strValue = binRead.ReadSingle().ToString();

        else if( e.NotificationHandle == hConnect[6] )
            strValue = new String(binRead.ReadChars(11));

        ((TextBox)e.UserData).Text = String.Format("DateTime: {0},{1}ms; {2}"
            ,time,time.Millisecond,strValue);
    }

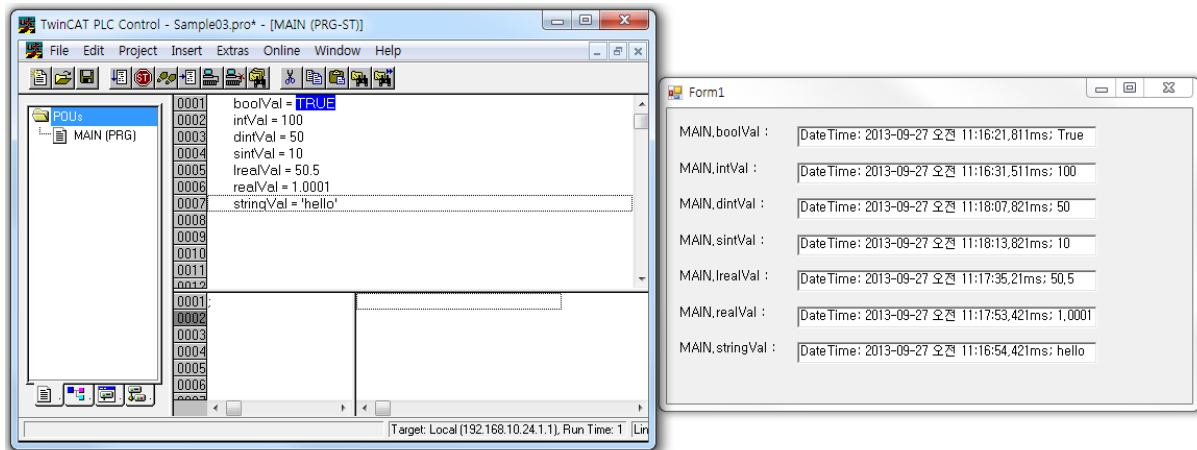
    // 프로그램이 종료되면 ADS 라우터 포트를 닫는다.
    private void Form1_Closing(object sender, System.ComponentModel.CancelEventArgs e) {
        try {
            tcClient.Dispose();
        }
        catch(Exception err) {
            MessageBox.Show(err.Message);
        }
    }
}

```

## PLC program

```
PROGRAM MAIN
VAR
    boolVal: BOOL;
    intVal : INT;
    dintVal : DINT;
    sintVal : SINT;
    lrealVal : LREAL;
    realVal : REAL;
    stringVal : STRING(10);
END_VAR
```

## 실행 화면





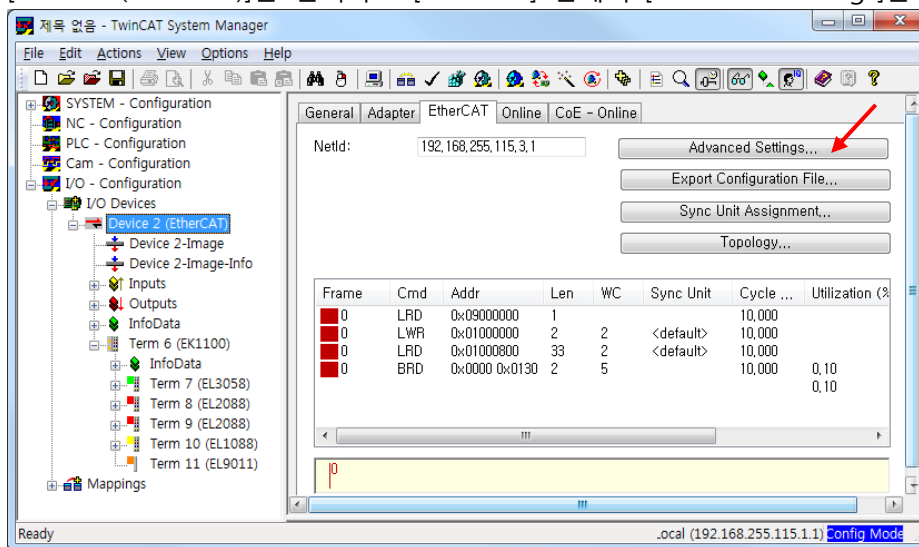
## 6. EtherCAT 통신 진단

TwinCAT System Manager 화면에서 EtherCAT 마스터와 슬레이브의 통신상태를 확인할 수 있다. 각 변수를 선택한 후 오른쪽 화면의 'Variable' 탭의 Comment 텍스트가 값에 대한 의미를 표시한다.

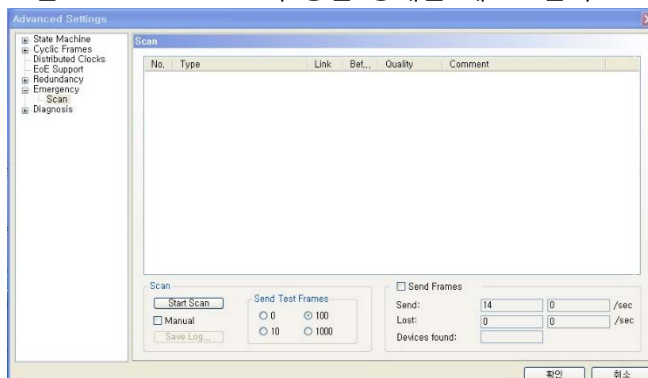
해당하는 변수의 ADS 정보는 'Variable' 탭의 'ADS Info' 에서 설명하고 있다. 이 정보를 이용하여 상위 어플리케이션에서 상태를 확인할 수 있으며, 또한 이 변수를 'Task' 변수와 연결하여 확인할 수도 있다

### 6.1. EtherCAT 프레임 상태 확인

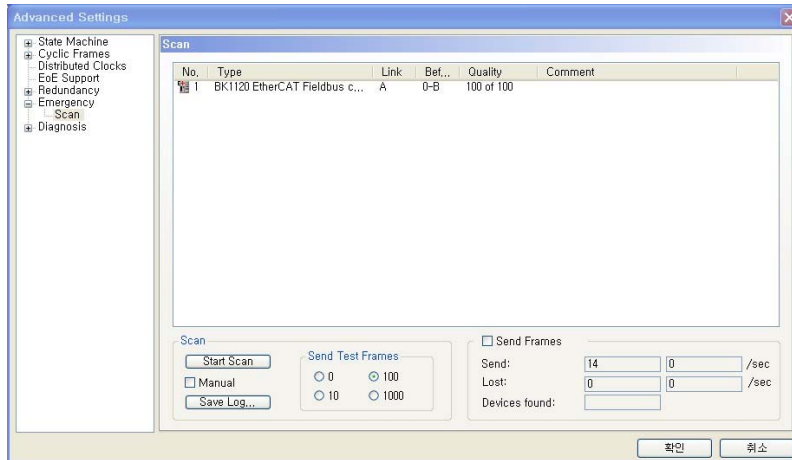
[Device 1(EtherCAT)]을 선택하고 [EtherCAT] 탭에서 [Advanced Settings]를 클릭한다.



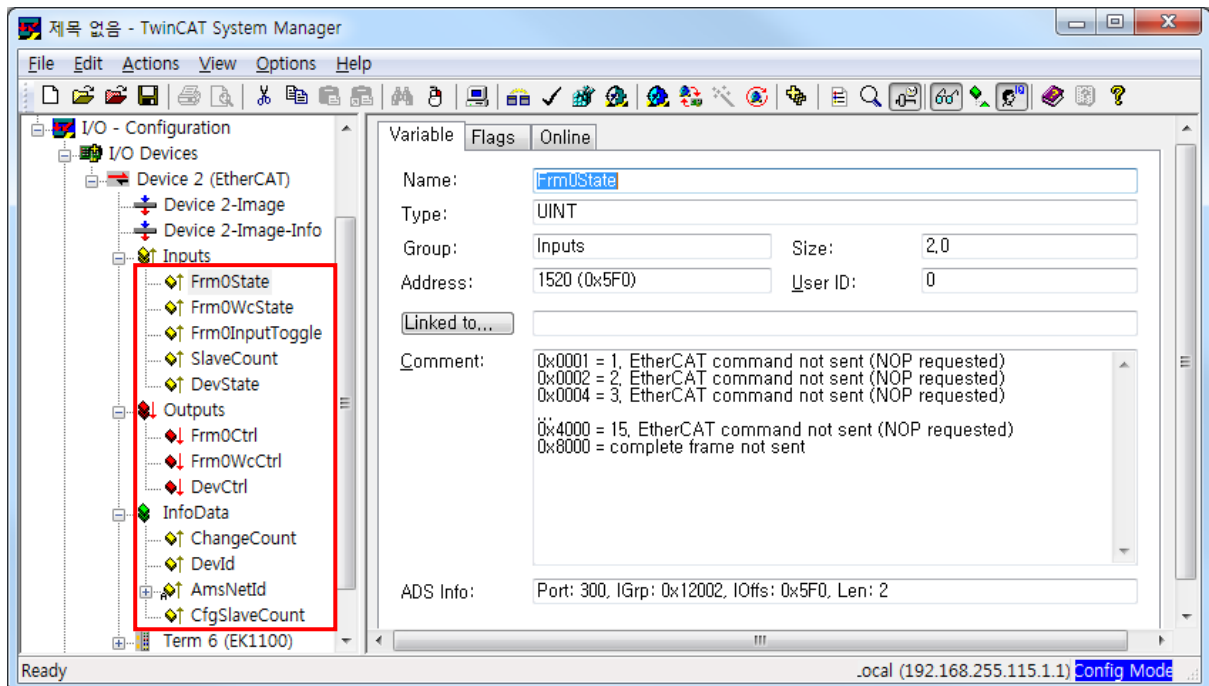
[Emergency]에 [Scan]을 선택하고, 사용할 Test Frames 을 선택, Start Scan 을 클릭하여, 연결된 모든 EtherCAT Slave 의 통신 상태를 테스트한다..



Quality 의 값이 부족하면 정상적인 EtherCAT 통신을 할 수 있는 상태가 아니다. 대다수의 EtherCAT 통신 문제는 Ethernet 케이블의 불량에 있으므로 Ethernet 케이블 교체를 해야 한다.



## 6.2. EtherCAT Master 상태 확인



**Frm0State:** EtherCAT 프레임 상태를 표시한다.

Bit	설명
0 (0x0001)	1. EtherCAT Command not sent (NOP requested)
1 (0x0002)	2. EtherCAT Command not sent (NOP requested)
2 (0x0004)	3. EtherCAT Command not sent (NOP requested)
...	
14 (0x4000)	15. EtherCAT Command not sent (NOP requested)
15 (0x8000)	Complete frame not sent

**Frm0WcState** : Master 가 Slave 로부터 정상상태의 프레임을 받지 못할 때 표시한다.

Bit	설명
0 (0x0001)	Wrong working counter of 1 EtherCAT command received
1 (0x0002)	Wrong working counter of 2 EtherCAT command received
2 (0x0004)	Wrong working counter of 3 EtherCAT command received
...	
14 (0x4000)	Wrong working counter of 15 EtherCAT command received
15 (0x8000)	Completed frame missing

**Slave Count** : EtherCAT Master 에 연결된 Slave 숫자를 표시한다.

Bit	설명
0 (0x0001)	Link error Detected
1 (0x0002)	I/O locked after link error (I/O Reset required)
2 (0x0004)	Link error (redundancy adapter)
3 (0x0008)	Missing one frame (redundancy mode)
4 (0x0010)	Out of send resources (I/O reset required)
5 (0x0020)	Watchdog triggered
6 (0x0040)	Ethernet driver (miniport) not found
7 (0x0080)	I/O reset active
8 (0x0100)	At least one device in "INIT" state
9 (0x0200)	At least one device in "PRE-OP" state
10 (0x0400)	At least one device in "SAFE-OP" state
11 (0x0800)	At least one device indicates an error state
12 (0x1000)	DC not in sync

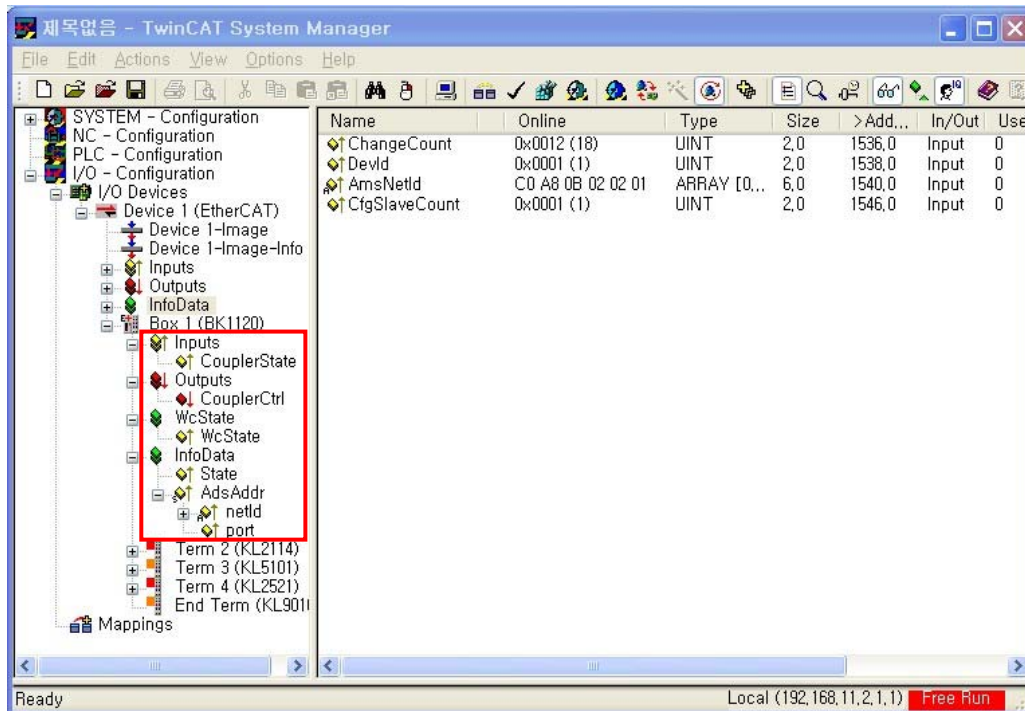
**DevState** : Word 타입의 변수로 EtherCAT Master 상태를 표시하며 아래 8-11 bits 는 연결된 Slave 의 상태를 표시한다.

**ChangeCnt**: EtherCAT Slave 의 상태 변경 횟수 이다.

**DevID** : EtherCAT Master Device 의 Station ID 이다.

**AmsNetID** : EtherCAT Master 의 AmsNetID 이다.

### 6.3. EtherCAT Slave 상태 확인



**CouplerState** : EtherCAT Slave 의 내부통신(K-Bus)상태를 표시한다.

Bit	설명
0 (0x0000)	No error
1 (0x0001)	K-Bus error
2 (0x0002)	Configuration error
5 (0x0010)	Outputs disabled
6 (0x0020)	K-Bus overrun
7 (0x0040)	Communication error (Inputs)
8 (0x0080)	Communication error (Outputs)

**CouplerCtrl** : EtherCAT Slave 를 컨트롤한다.

Bit	설명
5 (0x0010)	Disable outputs
6 (0x0020)	Show K-Bus overrun

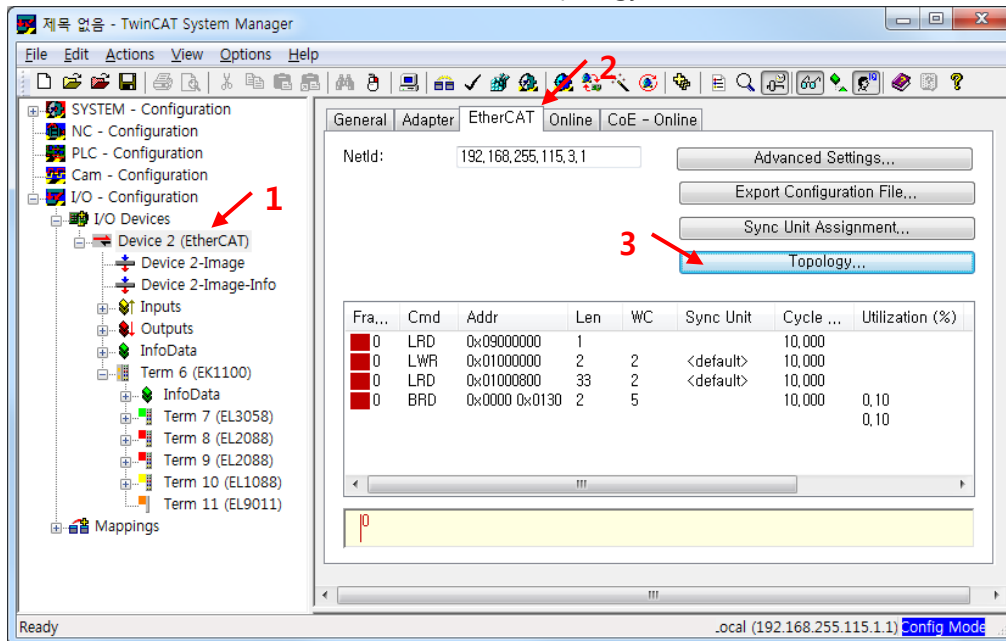
**State** : EtheCAT Slave Device 의 상태를 표시한다.

값	설명
0x__1	Slave in "INIT" State
0x__2	Slave in "PREOP" State
0x__3	Slave in "BOOT" State
0x__4	Slave in "SAFEOP" State
0x__8	Slave in "OP" State
0x0010	Slave signals error
0x0020	Invaild vendorId, ProductCode...Read
0x0100	Slave not present
0x0200	Slave signals link error
0x0400	Slave signals missing link
0x0800	Slave signals unexpected link
0x1000	Communication port A
0x2000	Communication port B
0x4000	Communication port C
0x8000	Communication port D

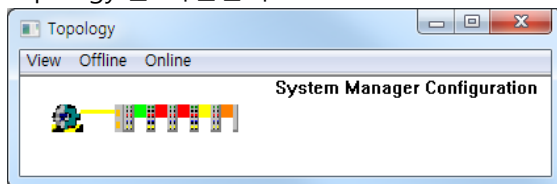
**AdsAddr** : EtherCAT Slave 의 AmsNetId 와 Port 를 표시한다.

## 6.4. EtherCAT 토폴로지

[Device 2(EtherCAT)]의 [EtherCAT] 탭에서 [Topology...]를 클릭한다.



Topology 를 확인한다.



- 기술 지원

서울 금천구 가산동 448 대륭테크노타운 3 차 717 호  
전화 : 02-2107-3242(대표전화)  
홈페이지 : [www.tritek.co.kr](http://www.tritek.co.kr)

마지막 수정 날짜 : 2013 년 9 월 30 일