User's Manual : I-7188XG & I-7188EG

This manual is intended for integrators, programmers, and maintenance personnel who will be installing and maintaining an I-7188XG & an I-7188EG controller system.

Legal Liability

ICP DAS CO., LTD. assumes no liability for any and all damages that may be incurred by the user as a consequence of this product. ICP DAS CO., LTD. reserves the right to change this manual at any time without notice.

ICP DAS CO., LTD. constantly strives to provide our customers with the most reliable and accurate information possible regarding our products. However, ICP DAS CO., LTD. assumes no responsibility for its use, or for any infringements of patents or other rights of third parties resulting from its use.

Trademark & Copyright Notice

The names of products are used for identification purposes only, and are the registered trademarks of their respective owners or companies.

Table of Contents

Reference Guide

This manual can also be found at
CD\NAPDOS\ISaGRAF\7188EG\English_Manu\User_Manual_I_7188XG_EG.pdf .

User's Manual Of The I-8417 / 8817 / 8437 / 8837 ISaGRAF Embedded controllers:

CD\NAPDOS\ISaGRAF\8000\English_Manu\User_Manual_I_8xx7.pdf .

ISaGRAF User's Guide:

For more extensive information regarding all of the capabilities of the ISaGRAF programming system, please refer to the "ISaGRAF USER'S GUIDE" manual which can be found from the CD ROM of the ISaGRAF workbench. Its file name is either "ISaGRAF.pdf" or "ISaGRAF.doc".

Installing ISaGRAF IO libraries:

Please refer to Chapter 1.

Hardware Manual:

I-7188XG: CD\NAPDOS\7188X\7188xb.htm .
I-7188EG: CD\NAPDOS\7188E\document\7188eh.pdf .

Set IP, Mask and Gateway address of I-7188EG:

Please refer to Appendix B.

CD on the Internet:

Newly updated ISaGRAF IO libraries, drivers and manuals can be found at ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/

Upgrade to New Hardware Driver:

You may upgrade the hardware driver to the new version. Refer to Appendix C

Technical Service:

Please contact the local agent or email to service@icpdas.com
New information can be found at www.icpdas.com

Specification

ISaGRAF Embedded Ethernet Controller: I-7188EG & I-7188EGD:

General environment

Operating temperature: -25°C to +75°C
Storage temperature: -40°C to +85°C
Humidity: 0 to 95 %
Built-in Watch Dog Timer (1.6 seconds)
Built-in power protection & network protection circuit
Program download from PC
Built-in I/O expansion bus interface

System

CPU: Am188™ES, 40M Hz, or compatiable
SRAM: 512K bytes
FLASH ROM: 512K bytes
COM port: COM1=RS-232, COM2=RS-485
Ethernet: 10 BaseT
Built-in RTC, NVRAM & EEPROM
Program download from COM1 & Ethernet
Built-in 64-bits hardware unique serial number
Supports ISaGRAF Programming languages: LD, FBD, SFC, ST, IL & FC

Real Time Clock

Year-2000 compliance
Gives seconds, minutes, hours, date of the month
Gives month and year, **from 1980 to 2079**
NVSRAM: 31 bytes, battery backup, data valid up to 10 years

EEPROM

2048 bytes (8 blocks, each block has 256 bytes)
Data retention > 100 years
1,000,000 erase/write cycles

Flash Memory

512K bytes
Erase unit is one sector(64K bytes)
100,000 erase/write cycles

COM1

RS-232: TXD,RXD,RTS,CTS,GND
Communication speed: 115200 max.
Supports Modbus Serial Protocol for Connecting PC/HMI & Touch panels

COM2

RS-485: Data+, Data-, self-tuner ASIC inside
Communication speed: 115200 max.
Supports Protocol of I-7000 & I-87K Remote I/O Modules
Supports Modbus Master Protocol for connecting other Modbus devices
Supports User self defined Protocol

Ethernet

10 BaseT Connector, 10M bps, NE2000 compatible
Supports Modbus TCP/IP Protocol for Connecting PC/HMI & Touch panels

Display

7-segment LED: 5-digit for 7188EGD

Power

Power requirements: 10 to 30VDC(non-regulated)
Power consumption: 2.0W for 7188EG
3.0W for 7188EGD

ISaGRAF Embedded Controller: I-7188XG & I-7188XGD:

General environment

Operating temperature: -25°C to +75°C
Storage temperature:-40°C to +85°C
Humidity: 0 to 95 %
Built-in Watch Dog Timer (1.6 seconds)
Built-in power protection & network protection circuit
Program downloadable from PC
Built-in I/O expansion bus interface

System

Module name: embedded controller
CPU: Am188™ES, 40M Hz
SRAM: 512K bytes
FLASH ROM: 512K bytes
COM port: COM1, COM2
Built-in RTC, NVRAM, EEPROM, D/I, D/O
Supports I/O Expansion Bus
Program download port: COM1
Supports ISaGRAF Programming languages: LD, FBD, SFC, ST, IL & FC

Real Time Clock

Year-2000 compliance
Seconds, minutes, hours, date of the month
Month, year, valid **from 1980 to 2079**
NVSRAM: 31 bytes, battery backup, data valid up to 10 years

EEPROM

2048 bytes (8 blocks, each block has 256 bytes)
Data retention > 100 years
1,000,000 erase/write cycles

Flash Memory

512K bytes
Erase unit is one sector(64K bytes)
100,000 erase/write cycles

D/I: 1 channel

High:3.5V ~ 30V, Low:0 ~ 1V

D/O: 1 channel

100 mA, 30V max.

COM1

RS-232 or RS-485
RS-232: TXD,RXD,RTS,CTS,GND
RS-485: D1+, D1-, self-tuner ASIC inside
Communication speed: 115200 max.
Supports Modbus Serial Protocol for Connecting PC/HMI & Touch panels

COM2

RS-485: D2+, D2-, self-tuner ASIC inside
Communication speed: 115200 max.
Supports Protocol of I-7000 & I-87K Remote I/O Modules
Supports Modbus Master Protocol for connecting other Modbus devices
Supports User self defined Protocol

Display

7-segmemt LED: 5-digit (for 7188XGD)

Power

Power requirements: 10 to 30VDC(non-regulated)
Power consumption: 2.0W for 7188XB
3.0W for 7188XBD

Chapter 1: Software & Hardware Installation

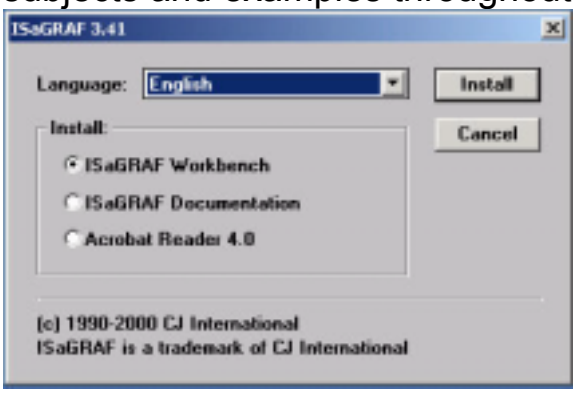
1.1: Installing The ISaGRAF Workbench Software Program

Before you can start programming the I-7188XG & I-7188EG embedded controller system, you must first install the ISaGRAF Workbench software program on a target PC.

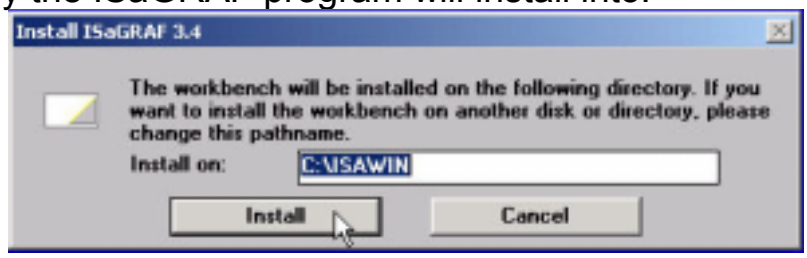
Steps To Installing The ISaGRAF Workbench Program

Insert the ISaGRAF Workbench CD into your CD-ROM drive. Normally the auto-start program will activate the "install.bat" file automatically. If your computer does not have the auto-start feature active, use the Windows Explorer and go to the CD-ROM drive where the Workbench CD is installed, then double-click on the "install.bat" file listed on the ISaGRAF CD. If the "install.bat" file is not found on your ISaGRAF CD, then double-click on the "ISaGRAF.exe" file to start the installation process.

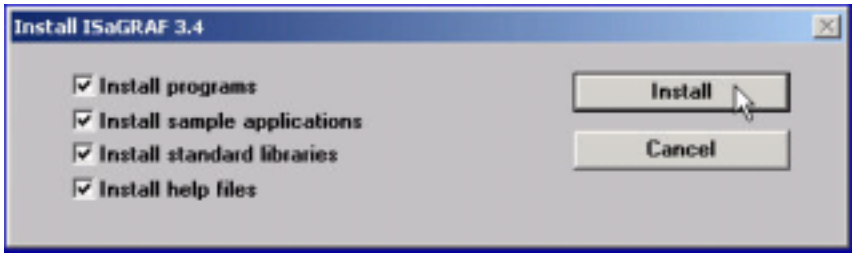
Once you have started the "install.bat" file, a dialog box will appear as shown on the next page. Select the language version of the ISaGRAF software program you would like to use and then press the "Install" button. The English version is used on all subjects and examples throughout this manual.



The first dialog box to appear allows the user to define what drive and subdirectory the ISaGRAF program will install into.



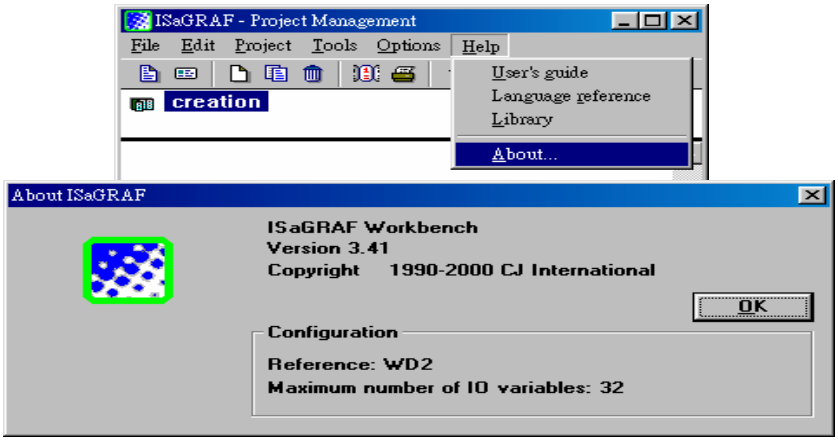
The next dialog box asks the user how much of the ISaGRAF program to you wish to install. By default, it is best to allow all of the ISaGRAF programs to install.



Once you have selected which programs and applications are to be installed, the installation process begins. Just wait the installation process to complete.

To begin the ISaGRAF 3.x software program, click on the Windows "Start" button, then on "Programs", and you should see the ISaGRAF program group as illustrated below.

NOTE: You must install the hardware protection device (dongle) provided with the ISaGRAF software on your computers parallel port to for the ISaGRAF program to achieve fully authorized functionality.
While using ISaGRAF and the dongle is plugged well, if the “Help” – “About” says “Maximum number of IO variables: 32”, it means ISaGRAF workbench cannot



find the dongle well. Please reset your PC and then check the “Help” – “About” again. If it still displays “Maximum number of IO variables: 32”, the dongle driver may not be installed well. Please execute the ISaGRAF CD_ROM \Sentinel5382\setup.exe for ISaGRAF-80 or \Sentinel\setup.exe for other ISaGRAF version and then reset the PC again.

Important Notice For Window NT Users

If your computer is using the Windows NT operating system, you will need to add one line to the "isa.ini" file in the ISaGRAF Workbench "EXE" subdirectory. If the ISaGRAF program is installed on your computer's "C" hard drive, you will find the required file in the following path:

C:\isawin\exe\isa.ini

You can use any ASCII based text editor (such as Notepad or UltraEdit32) to open the "isa.ini" file. Locate the [WS001] header in the "isa.ini" initialization file (it should be at the top of the file). Anywhere within the [WS001] header portion of the "isa.ini" initialization file, add the entry shown below within the [WS001] header:

[WS001]
NT=1
Isa=C:\ISAWIN
IsaExe=C:\ISAWIN\EXE
Group=Samples
IsaApl=c:\isawin\smp
IsaTmp=C:\ISAWIN\TMP

The [WS001] header should now look like the above example. The **NT=1** entry addition is absolutely required for the RS-232 communications to operate properly in the Windows NT operating environment.

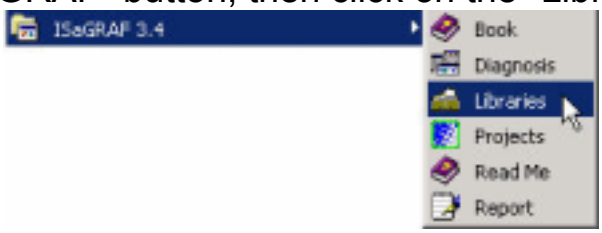
1.2: Installing The I-7188XG & I-7188EG I/O Libraries

The ISaGRAF Workbench software program must be installed before attempting to install the I-7188XG & I-7188EG I/O libraries. If you have not already installed the ISaGRAF Workbench program, please refer to section 1.1 before continuing.

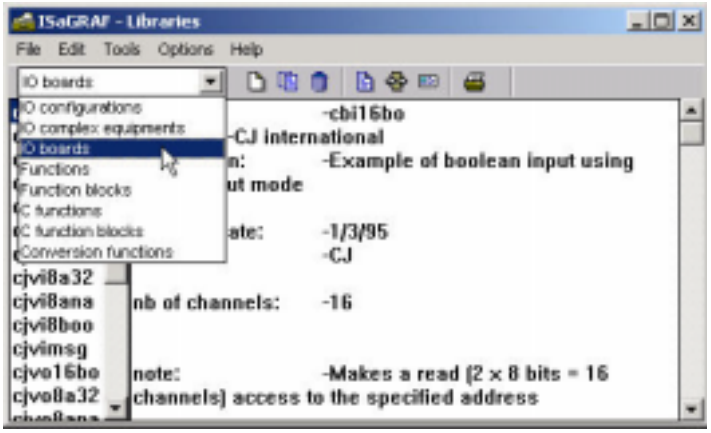
There is a CD-ROM supplied with each of the I-7188XG & I-7188EG controller with the appropriate I/O libraries. Please insert the I/O library CD into your CD-ROM drive. The following details the installation process for the operating systems that support the ISaGRAF Workbench program.

Setting Up The Libraries For Windows 95, 98, Windows NT & Windows 2000 Operating Systems

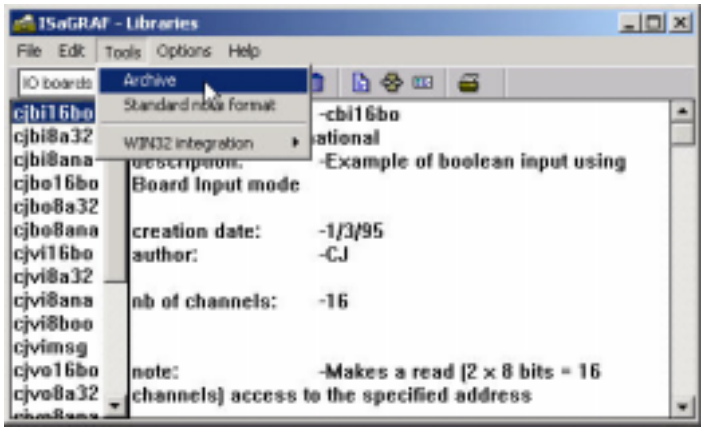
Click on the Windows "Start" button, then click on the "Program" button, then click on the "ISaGRAF" button, then click on the "Libraries" button as shown below.



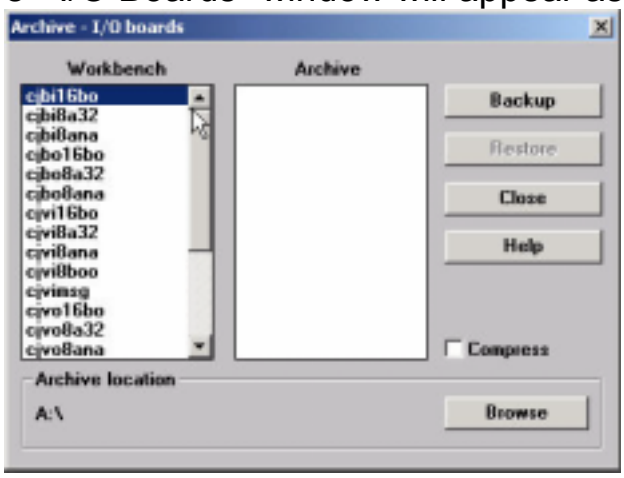
When you click on the "Libraries" icon, the following window will open:



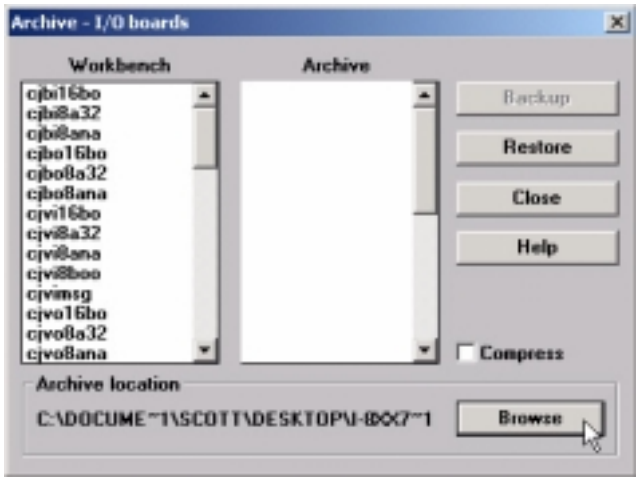
When the ISaGRAF Libraries window opens select "**IO Boards**" from the drop down menu just below the file menu bar. Next, select the "Tools" option from the main menu bar, then select "Archive" from the file menu bar as shown below.



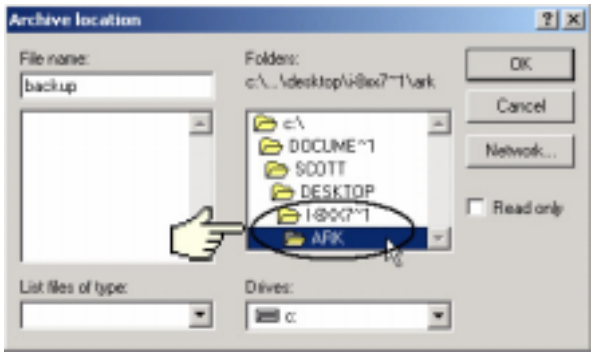
The "Archive - I/O Boards" window will appear as shown below.



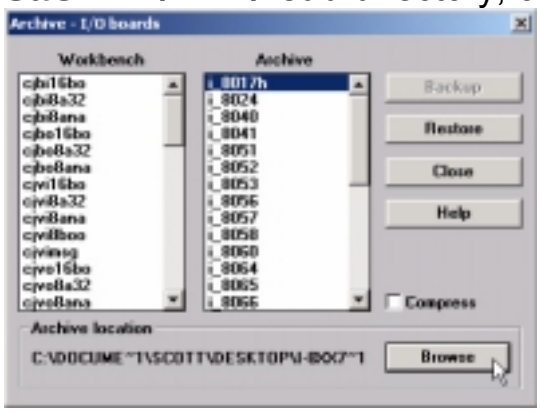
To find the drivers for the I-7188XG & I-7188EG controller system, click on the "Browse" button as shown below.



When you click on the "Browse" button, an "Archive Location" dialog box will open.



The important item to locate is the "**CD_ROM: \NAPDOS\ISaGRAF\ARK**" sub-directory (the location of these sub-directories) may be different depending on where you have the driver files located. After you have located the " **CD_ROM: \NAPDOS\ISaGRAF\ARK**" sub-directory, click on the "OK" button.



The I/O library for the I-7188XG & I-7188EG & I-8417/8817/8437/8837 control boards now appear in the "Archive" list box. Select all the items, and then click on the "Restore" button. The selected drivers will now install into the ISaGRAF sub-directory.

Follow the same procedure for loading the "**IO Complex Equipment**" library like you did to load the library for the "**IO Boards**". The same procedure should be used for loading other appropriate libraries for the I-7188XG / I-7188EG & I-8xx7 including "**C functions**" and "**C function blocks**"

1.3: Connecting Your PC To The I-7188XG / I-7188EG

1.3.1: Setting The NET-ID Address For The I-7188XG & I-7188EG

Each I-7188XG & I-7188EG F has a NET-ID No. The valid No. can be assigned is from 1 to 255. The default No. is 1.

To change the NET-ID No, please follows below steps.

1. Create a file folder named "7188" in your hard drive.
   For example, "c:\7188".

For Dos, Windows 95 & Windows 98 Users:
2. Copy \Napdos\IsaGRAF\7188EG\Driver\7188x.exe, 7188x.ini
   from the CD_ROM into your "7188" folder.
3. Run "\7188\7188x.exe" in your hard drive. A "7188x" screen
   will appear.

For Windows NT, Windows 2000 & Windows XP Users:
2. Copy \Napdos\ISaGRAF\7188EG\Driver\7188xw.exe, 7188xw.ini
   from the CD_ROM into your "7188" folder.
3. Run "\7188\7188xw.exe" in your hard drive. A "7188xw" screen
   will appear.

4. Link from COM1 of your PC to COM1 of the I-7188XG & I-7188EG
   controller by a RS232 cable.

5. Power off the I-7188XG & I-7188EG controller, connect pin "INIT*" to
   "GND", and then power it up.

6. If the connection is Ok, messages will appear on the 7188x screen.

   7188>

7. Type "isa7188 *s= " to set the NET-ID for I-7188**XG**
   8000> isa7188 *s=2

   Type "isa7188**e** *s= " to set the NET-ID for I-7188**EG**
   8000> isa7188e *s=3

8. Remove the connection between “INIT*” and “GND" .

1.3.2: Deleting The ISaGRAF Project Inside The I-7188XG / I-7188EG

If one ISaGRAF project has been download to the I-7188XG & I-7188EG controller. User may download a new ISaGRAF project to replace the old one by using ISaGRAF workbench. Or by some resons, user may want to delete the ISaGRAF project inside the I-7188XG & I-7188EG controller. To do this, please follows below steps.

1. Create a file folder named "7188" in your hard drive.
   For example, "c:\7188".

For Dos, Windows 95 & Windows 98 Users:
2. Copy \Napdos\IsaGRAF\7188EG\Driver\7188x.exe, 7188x.ini
   from the CD_ROM into your "7188" folder.
3. Run "\7188\7188x.exe" in your hard drive. A "7188x" screen
   will appear.

For Windows NT, Windows 2000 & Windows XP Users:
2. Copy \Napdos\ISaGRAF\7188EG\Driver\7188xw.exe, 7188xw.ini
   from the CD_ROM into your "7188" folder.
3. Run "\7188\7188xw.exe" in your hard drive. A "7188xw" screen
   will appear.

4. Link from COM1 of your PC to COM1 of the I-7188XG & I-7188EG
   controller by a RS232 cable.

5. Power off the I-7188XG & I-7188EG controller, connect pin "INIT*" to
   "GND", and then power it up.

6. If the connection is Ok, messages will appear on the 7188x screen.

   7188>

7. Type "isa7188 *d=" to delete ISaGRAF project for I-7188**XG**
   8000> isa7188 *d=

   Type "isa7188**e** *d=" to delete ISaGRAF project for I-7188**EG**
   8000> isa7188e *d=

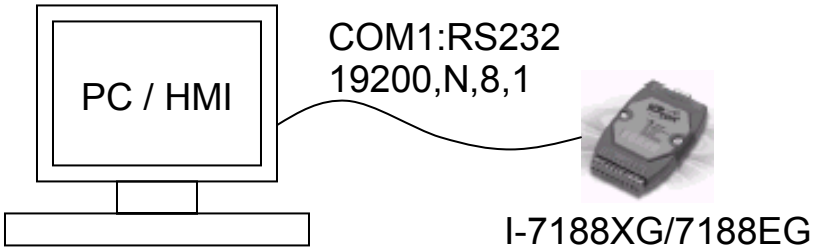
8. Remove the connection between “INIT*” and “GND" .

1.3.3: Connecting Your PC To The I-7188XG / I-7188EG COM1 Port

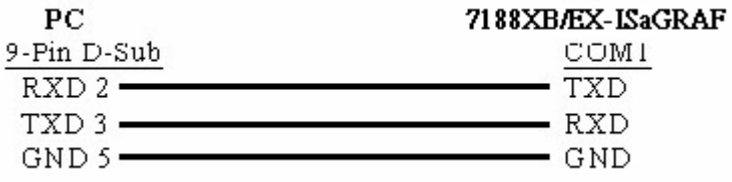
The COM1 port of the I-7188XG & I-7188EG is a Modbus slave port which can talk with **HMI softwares** or for the **ISaGRAF workbench** to download the ISaGRAF project.

COM1 of the I-7188EG is a pure RS232 port, while COM1 of the I-7188XG can be used as either a RS232 or a RS485 port.

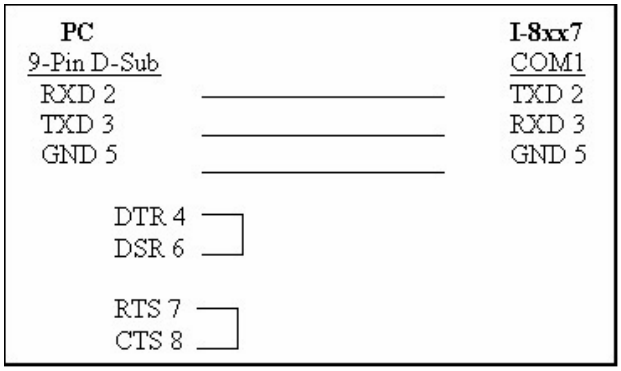
One PC/HMI can only link to COM1:RS232 port of **one** I-7188XG & I-7188EG.



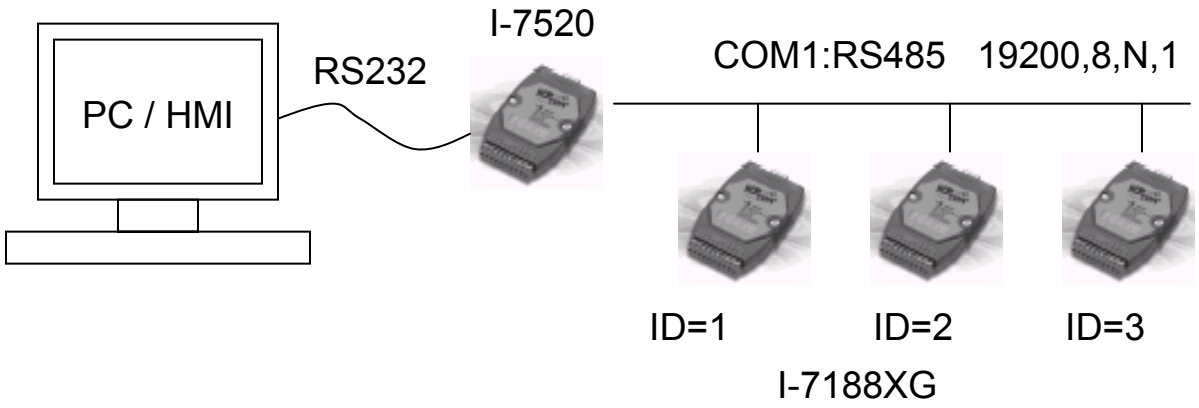
COM1:RS-232 Pin Wiring Assignments



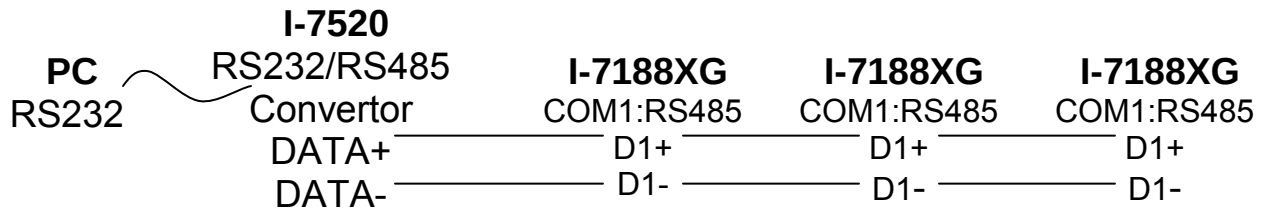
For the ISaGRAF Workbench RS-232 communication to operate properly, only the RXD, TXD, and the GND signals are used. If your PC is running a hardware device or software program that uses the CTS and DSR signals, you will need to wire the RTS-CTS and DTR-DSR signals together as shown below.



One PC or HMI can link to COM1:**RS485** port of **many** I-7188XG if each of them on the same RS485 network has a unique NET-ID.



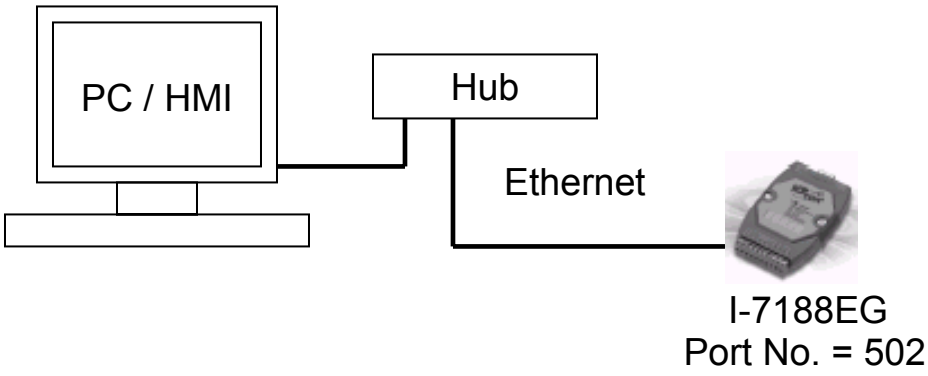
COM1:RS-485 Pin Wiring Assignments



Note:
Please make sure each I-7188XG on the same RS485 network has a different NET-ID. Please refer to Section 1.3.1 to set the NET-ID.

1.3.4: Connecting Your PC To The I-7188EG Ethernet Port

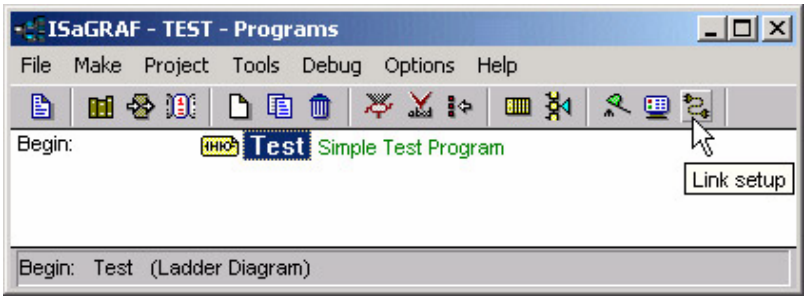
The ethernet port of the I-7188EG controller provides Modbus TCP/IP protocol. It can be used to connect to the PC or HMI software. Up to 5 PC/HMI can talk to one I-7188EG at the same time through the ethernet port.



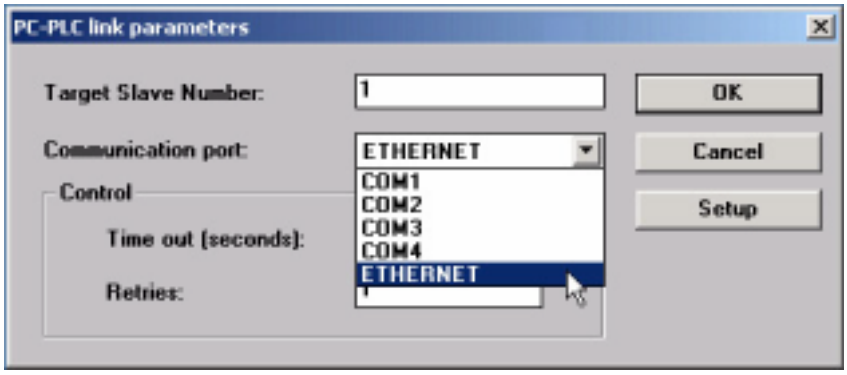
Before you can download an ISaGRAF application to the I-7188EG controller using the Ethernet port, you must first setup the Ethernet port to properly communicate with the host PC.

At the I-7188EG, Set IP, Mask and Gateway address:
Refer to **Appendix B** or CD_ROM:\NAPDOS\ISaGRAF\7188EG\driver\setip.txt

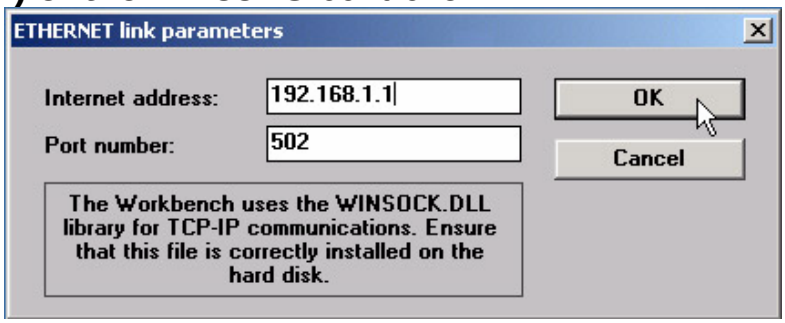
At your PC:
First open an ISaGRAF project and select a program you wish to communicate between your PC and the I-7188EG controller system. Next, select the "Link Setup" button on the project screen as shown below.



A "PC-PLC Link Parameters" dialog box will appear as shown below. From here select the "Ethernet" communications option and click on the "Setup" button.



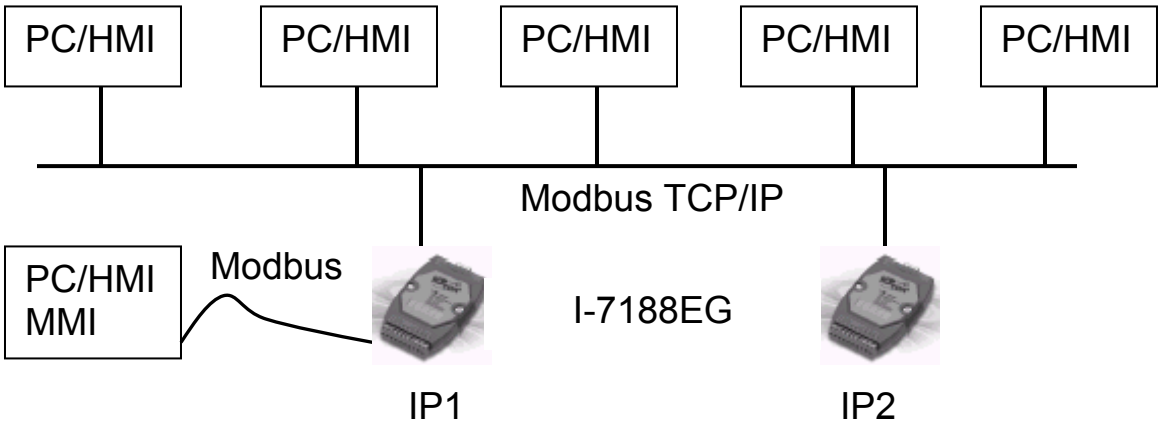
Once you have clicked on the "Setup" button, an "Ethernet Link Parameters" dialog box will appear. Set the "Port Number" to "**502**" and enter in the **Internet address (IP) of the I-7188EG** controller.



Once you have entered the appropriate information, click on the "OK" button, and now you have configured your PC to communicate with the I-7188EG through the Ethernet port.

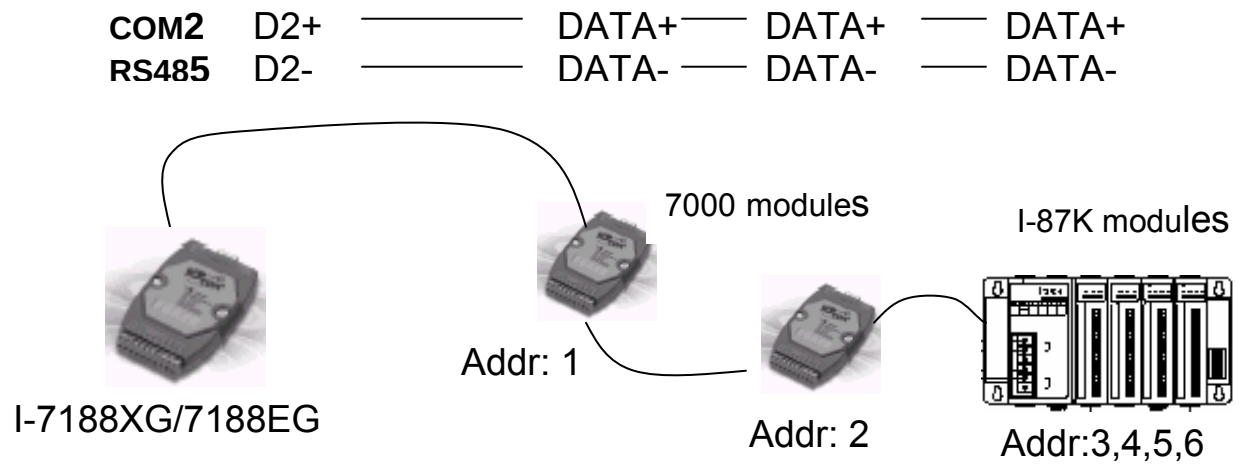
1.3.5: Multi-Clients Connection to The I-7188EG

Each I-7188EG has an IP address and with a fixed Ethernet port No. **502.** Up to 5 PCs can link to one I-7188EG throughout Ethernet (Modbus TCP/IP protocol). Another PC or MMI can link to COM1: RS232 port (Modbus protocol). Therefore the maximum number of clients can be linked is 6.



1.4: Linking I-7000 and I-87K Modules For Remote I/O

The I-7188XG and I-7188EG controller system can use its **COM2:RS485** port to link to ICP DAS's "I-7000" and “I-87K” series of remote I/O modules. This configuration can be very useful in applications that require distributed remote I/O throughout the system.



You can link up to 64 I-7000 or I-87K series remote modules to one I-7188XG / I-7188EG controller. You must remember to set each I-7000 and I-87K remote module must have a unique address, and be set to the same baud rate as the I-7188XG / I-7188EG controller system.

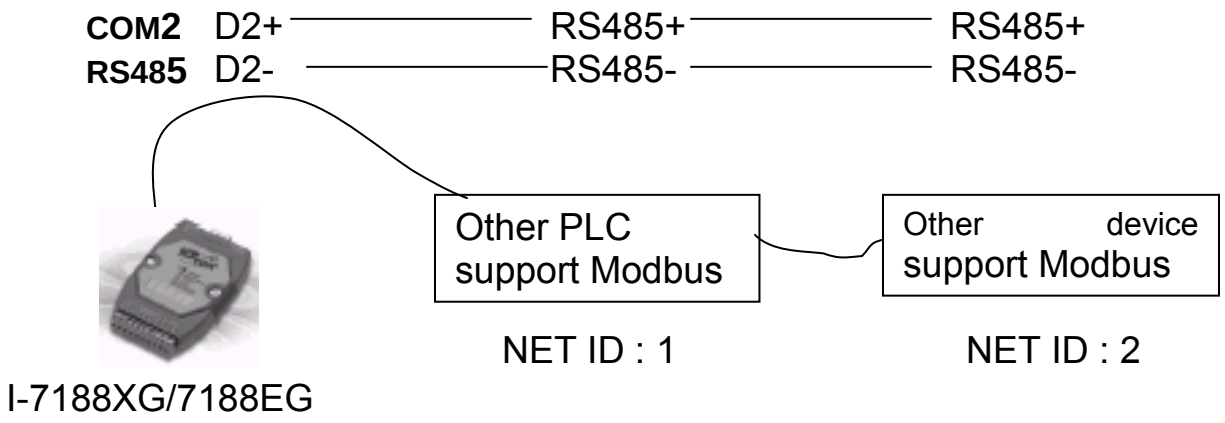
For more information regarding setting up and programming an I-7000 / I-87K remote module, please refer to Chapter 6 - "Linking To I-7000 and I-87K Modules".

1.5: Creating A Modbus Master Link

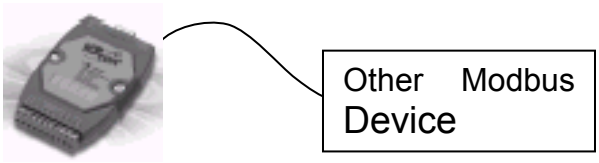
The I-7188XG and I-7188EG controller system can be a Modbus "Slave" and/or a Modbus "Master" controller depending on the application. Through this method you can use the COM1 port of the I-7188XG / I-7188EG controller system to link to a PC or other HMI products. In this type of configuration, the I-7188XG / I-7188EG controller becomes a Modbus slave controller. For more information about setting up and programming for Modbus slave, please refer to Chapter 4 .

Either **COM2 or COM3** can be used to link to other devices that support the Modbus protocol, then the I-7188XG / I-7188EG controller system will be the Modbus master controller. For more information about setting up and programming for Modbus master, please refer to Chapter 7 - "Linking A Modbus RTU Or Other Devices".

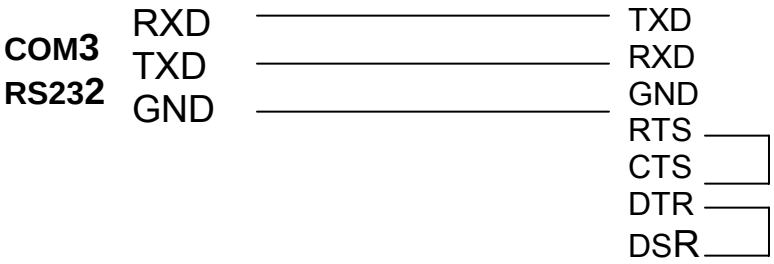
If **COM2:RS485** is used, one I-7188XG / I-7188EG can connect to many other Modbus devices. Each device on the link must have a unique NET ID (1 ~ 255) address, and communicate at same baud rate settings.

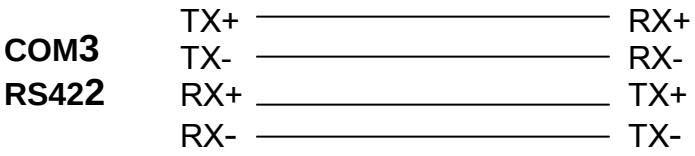


If **COM3:RS232**(with one of X503, X504, X505, X506 I/O expansion board plugged) or **COM3:RS422** (with X507 I/O expansion board plugged) is used, one I-7188XG / I-7188EG can connect to one Modbus device.



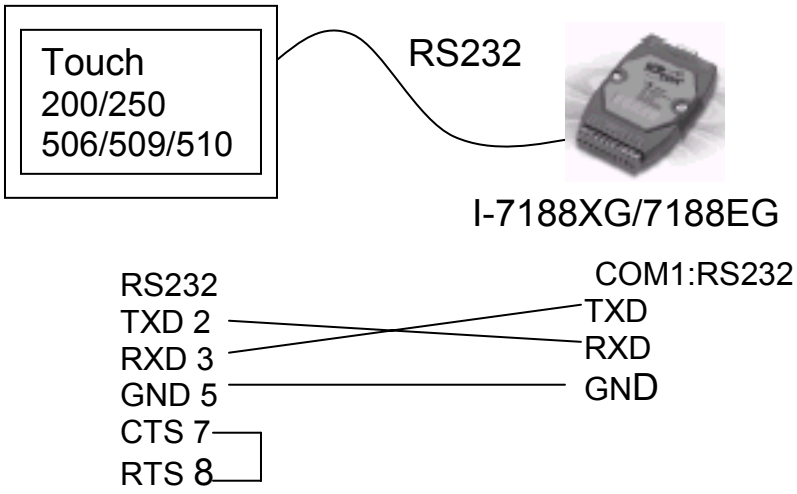
I-7188XG/7188EG





1.6: Linking To An MMI Interface Device

The **COM1:RS-232** port of the I-7188XG and I-7188EG controller system can be used to interface with additional Man Machine Interface (MMI) devices such as touch screen displays. ICP DAS provides a full line of touch screen displays, such as the "Touch" series screens. The models in the product line include the Touch 200, Touch 250, Touch 506, Touch 509 and Touch 510 MMI products.



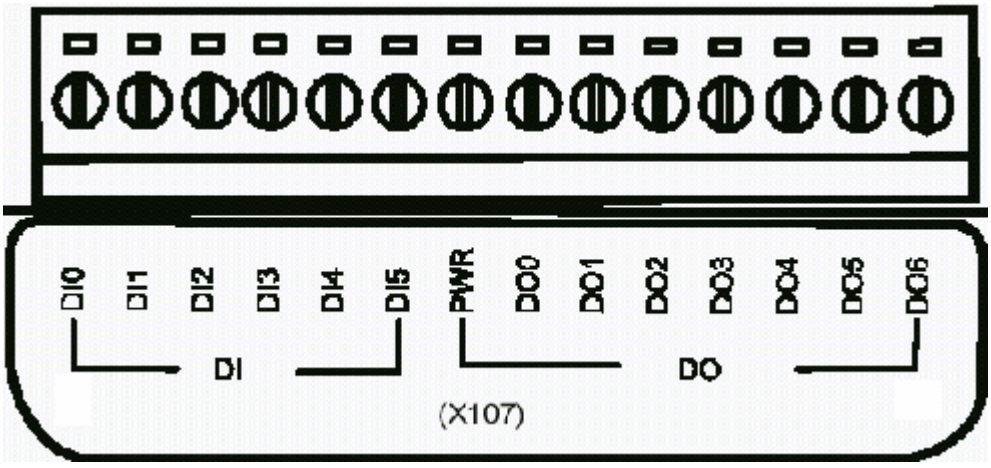
1.7: Using I-7188 I/O Expansion Boards

The I-7188XG / I-7188EG can plug an I/O Expansion board inside the main body. To install it, user have to loosen the screw and remove the shell of I-7188XG / I-7188EG. The supported I/O expansion boards are as below. It will be more.

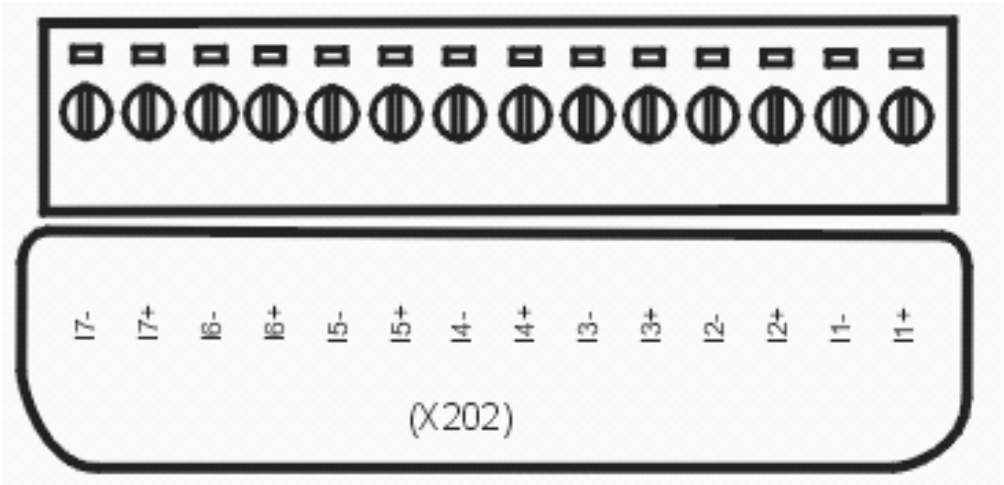
X107 :	6 D/I & 7 D/O
X202 :	7 A/D (0~20mA)
X203 :	2 A/D (0~20mA), 2 D/I & 6 D/O
X303 :	1 A/D (+/- 5V), 1 D/A (+/- 5V), 4 D/I & 6 D/O
X304 :	3 A/D (+/- 5V), 1 D/A (+/- 5V), 4 D/I & 4 D/O
X305 :	7 A/D (+/- 5V), 1 D/A (+/- 5V), 2 D/I & 2 D/O
X310:	2 A/D (0~10V or 0 ~40mA), 2 D/A (0~10V), 3 D/I & 3 D/O
X503 :	1 RS232 (5 wire,RTS,CTS,RXD,TXD,GND) : COM3
X504 :	1 RS232 (5 wire) : COM3 & 1 RS232 (9 wire) : COM4
X505 :	3 RS232 (5 wire) : COM3 , COM4 , COM5
X506 :	6 RS232 (3 wire, RXD, TXD, GND): COM3 ~ COM8
X507 :	1 RS422 (TXD+, TXD-, RXD+, RXD-) : COM3, 4 D/I & 4 D/O
X508 :	1 RS232 (5 wire) : COM3, 4 D/I & 4 D/O
X509 :	2 RS232 (3 wire) : COM3, 4 D/I & 4 D/O
X510 :	1 RS232/RS485: COM3, 5 D/I & 5 D/O

Pin assignment:

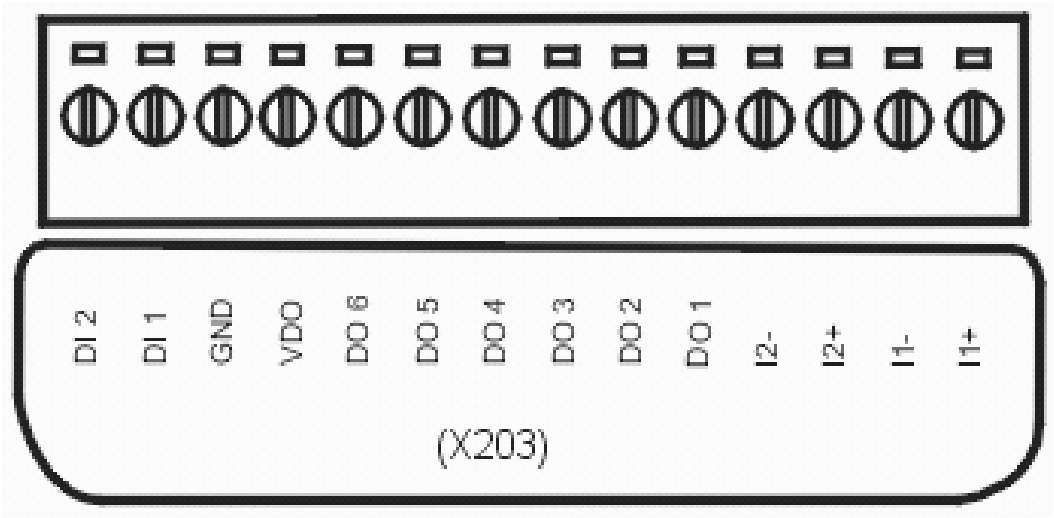
X107:



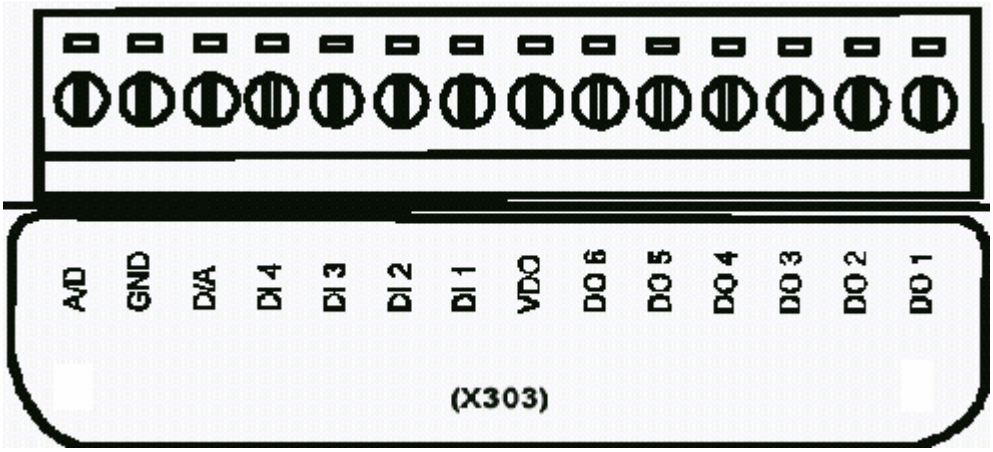
X202:



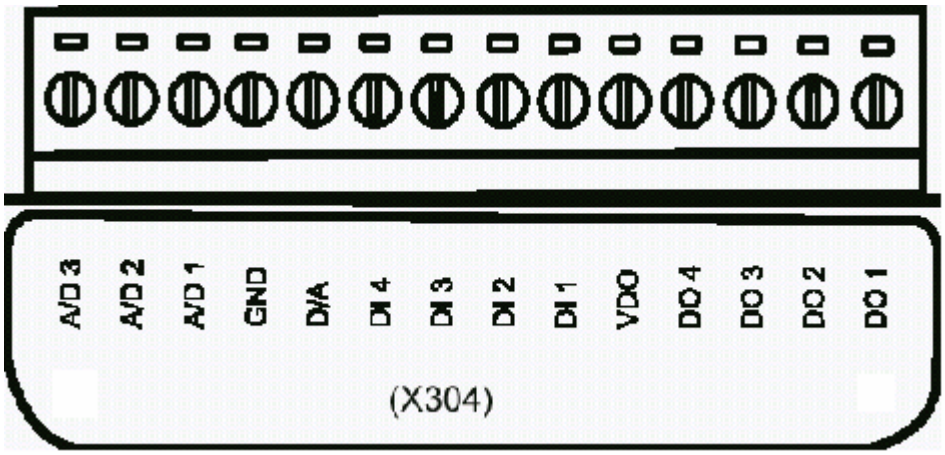
X203:



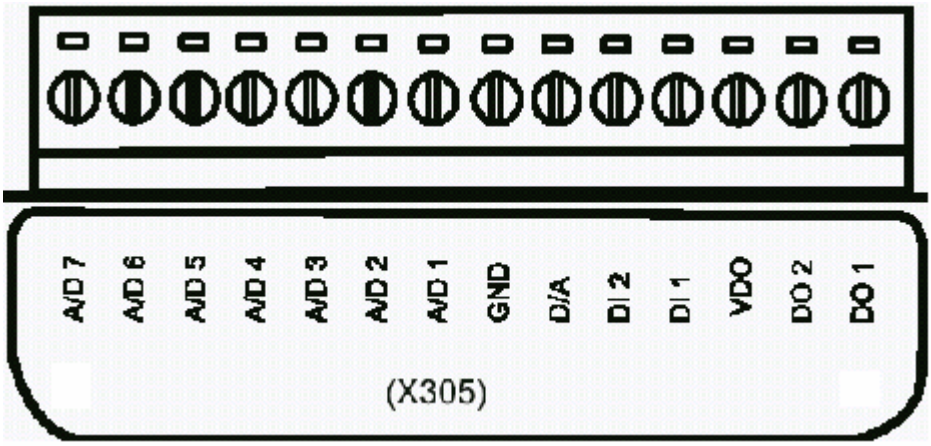
X303:



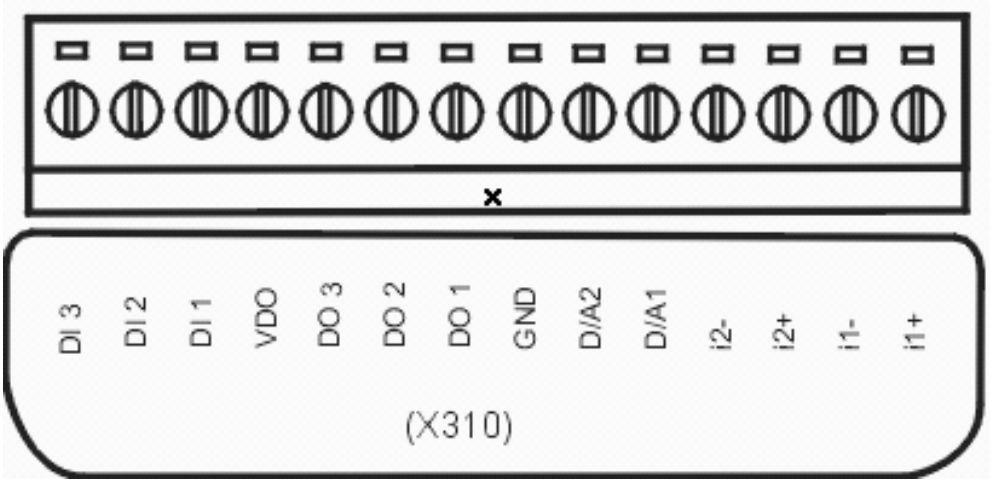
X304:



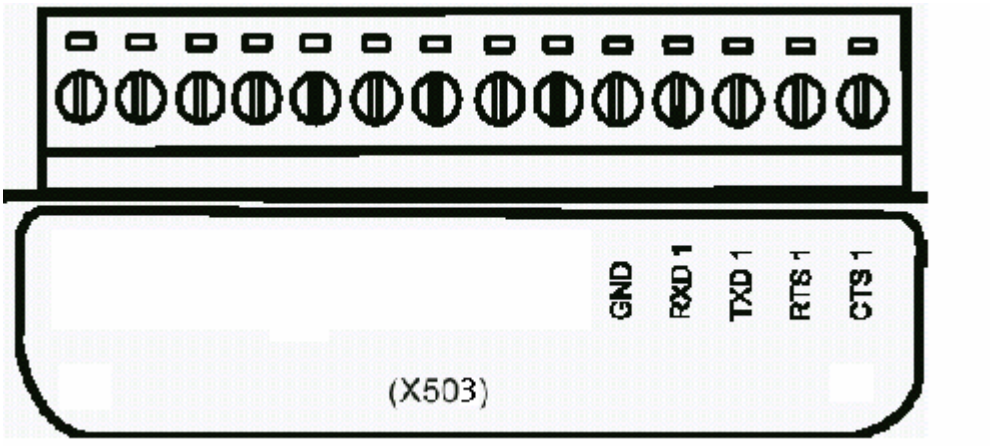
X305:



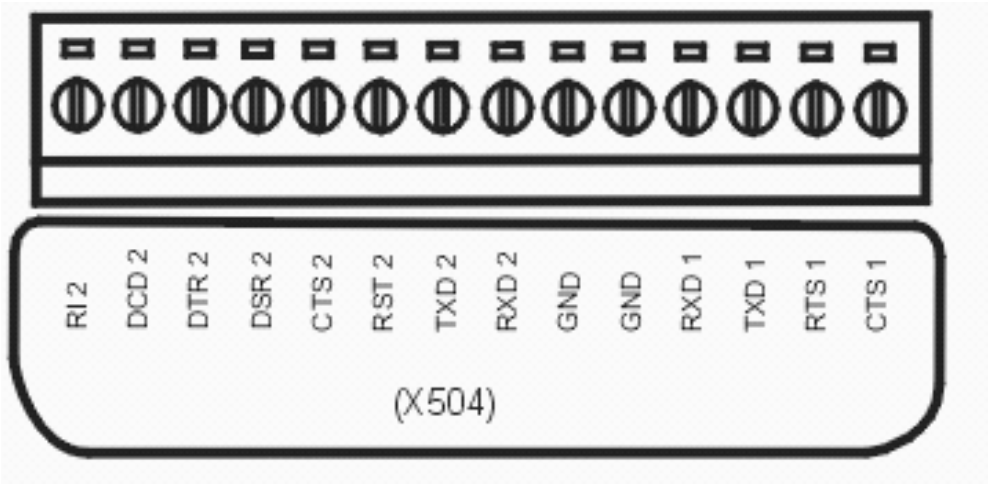
X310:



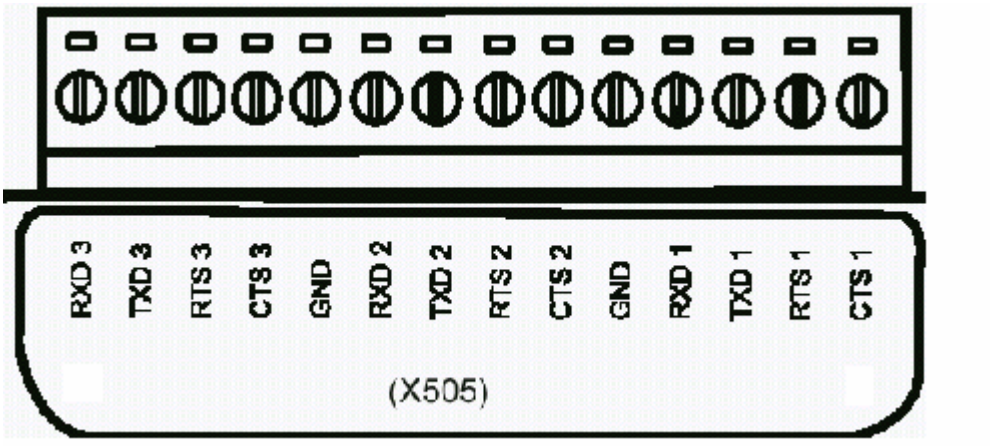
X503:



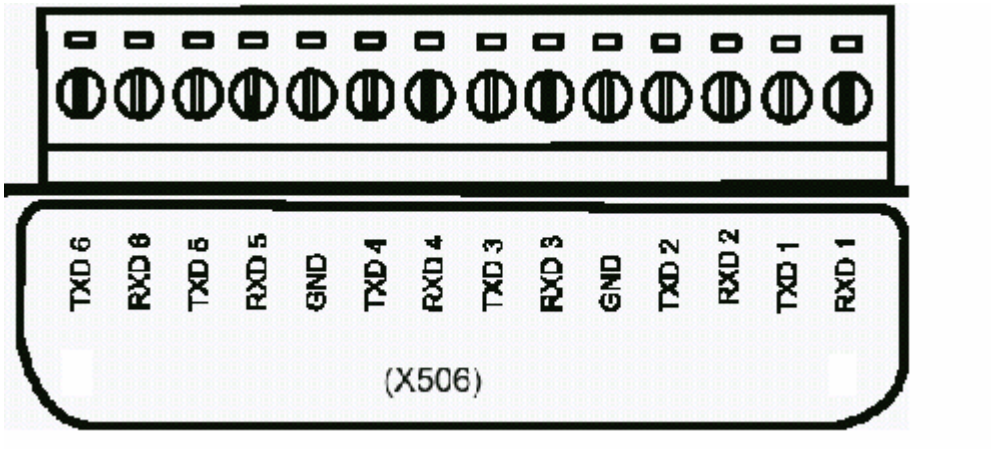
X504:



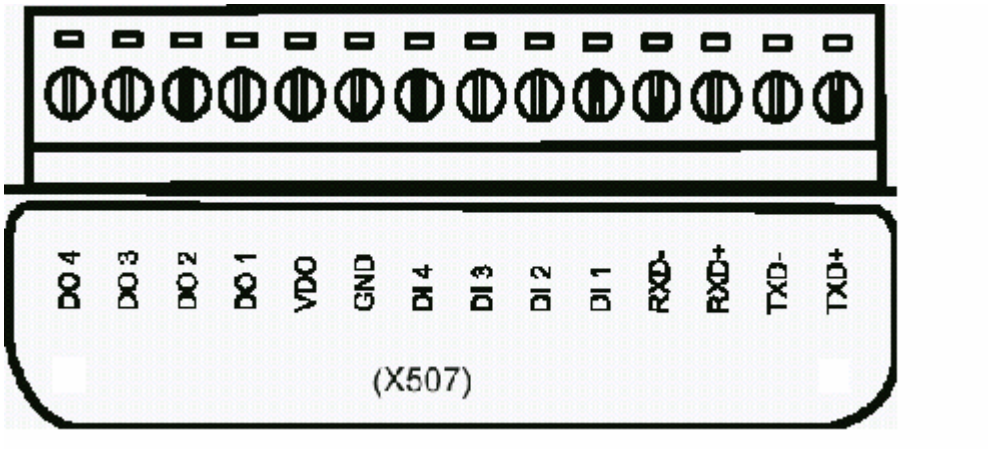
X505:



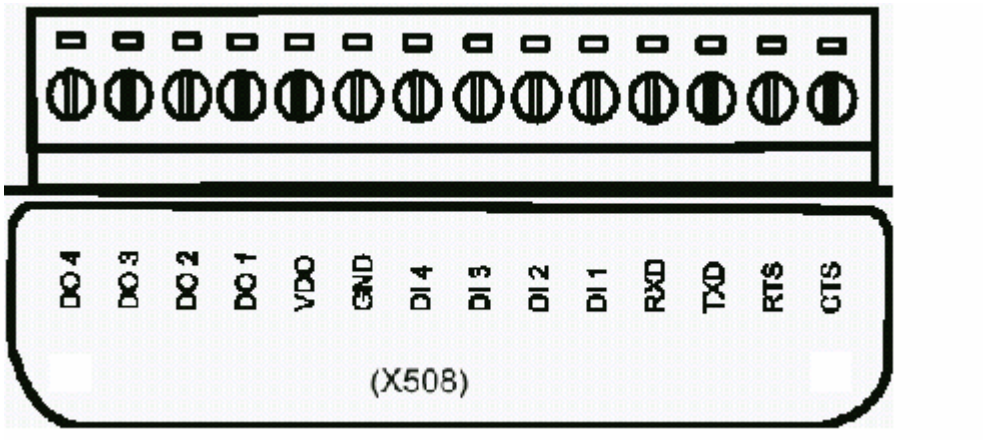
X506:



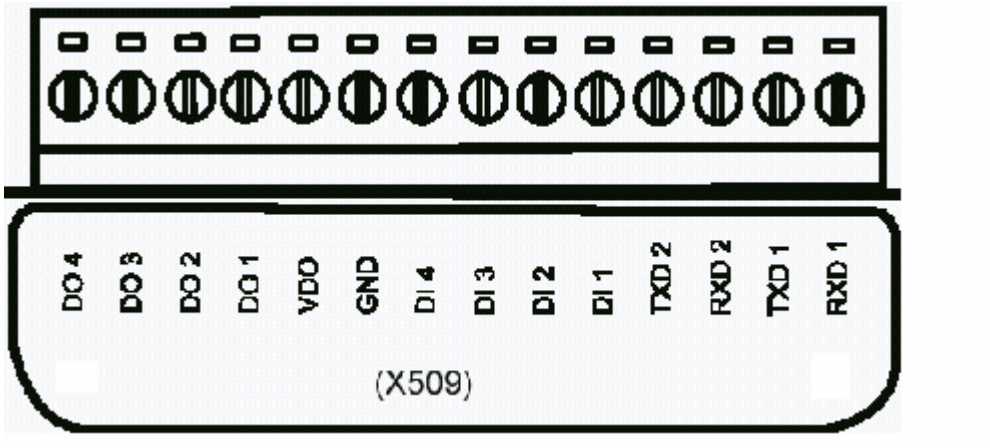
X507:



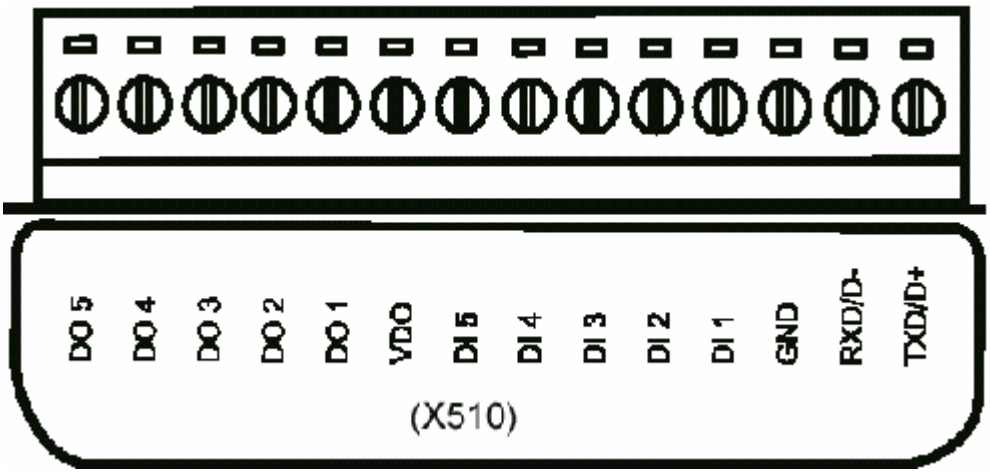
X508:



X509:


DO 4
DO 3
DO 2
DO 1
VDO
GND
DI 4
DI 3
DI 2
DI 1
TXD 2
RXD 2
TXD 1
RXD 1
(X509)

X510:


DO 5
DO 4
DO 3
DO 2
DO 1
VDO
DI 5
DI 4
DI 3
DI 2
DI 1
GND
RXD/D-
TXD/D+
(X510)

Chapter 2: ISaGRAF Programming Basics

2.1: A Simple Ladder Logic (LD) Program

This chapter provides simple yet effective program examples of how you can use the different ISaGRAF programming languages.

Note:
Please refer to “User’s Manual Of I-8417 / 8817 / 8437 / 8837 ISaGRAF Embedded Controllers” for simple programs of FDB, ST, IL & SFC language, or refer to “Napdos\ISaGRAF\8000\English_Manu\user_manu_I_8xx7.pdf”.

For more extensive information regarding all of the capabilities of the ISaGRAF programming system, please refer to the “ISaGRAF USER’S GUIDE” manual which can be found from the CD_ROM of the ISaGRAF workbench. Its file name is either “ISaGRAF.pdf” or “ISaGRAF.doc”.

Ladder Logic Basics

"Ladder Logic" programming (LD) is a graphical representation of Boolean equations, combining **contacts** (input arguments) and **coils** (output results). Ladder Logic most closely resembles the electrical schematics that an electrician or technician may use to diagnose and troubleshoot an industrial process controller system.

The LD language enables the programmer to describe the conditions and modifications to Boolean data by placing "graphical symbols" to represent hardware devices used in a process control application.

A Simple Ladder Example Program

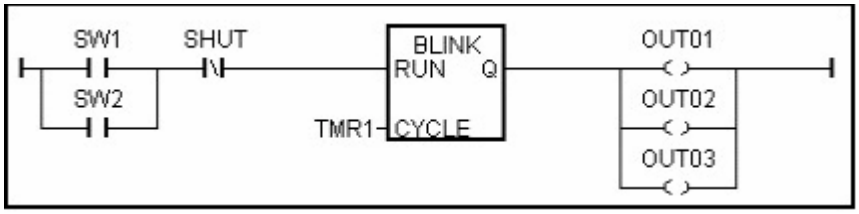
The following is a step-by-step example on how to create a ladder logic (hence forth referred as "LD") program using the ISaGRAF Workbench software program provided with the I-7188XG / I-7188EG controller system.

In the example, two normally open switches are programmed in parallel, illustrating a Boolean "OR" operation, a normally closed switch acting as a shutdown or emergency stop switch followed by a timer set to one second. When the logic flow becomes true starting at the left power rail through all the input path logic then three (3) outputs are turned on.

Variables Used In The Example LD Program:

Name	Type	Attribute	Description
SW1	Boolean	Input	Switch input 1
SW2	Boolean	Input	Switch input 2
SHUT	Boolean	Input	Shutdown input
OUT01	Boolean	Output	Output 1
OUT02	Boolean	Output	Output 2
OUT03	Boolean	Output	Output 3
TMR1	Timer	Internal	Time Period of blinking, initial value is set at "T#1s"

Ladder Logic Program Outline:



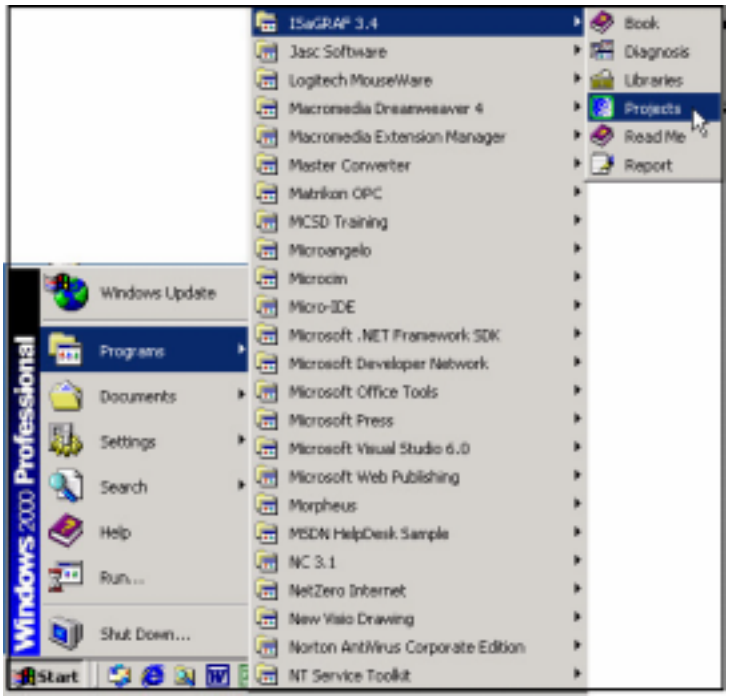
Process Operation Actions:

1. Monitor Switch 1 (Normally Open) & Switch 2 (Normally Open)
2. Monitor Shutdown Switch (Normally Closed)
3. If Either Switch 1 OR 2 Is True, AND Shutdown Switch Is Closed, Active "Blink" Timer
4. Turn Outputs 1, 2, & 3 On And Off At One Second Interval Rate

2.1.1: Programming LD

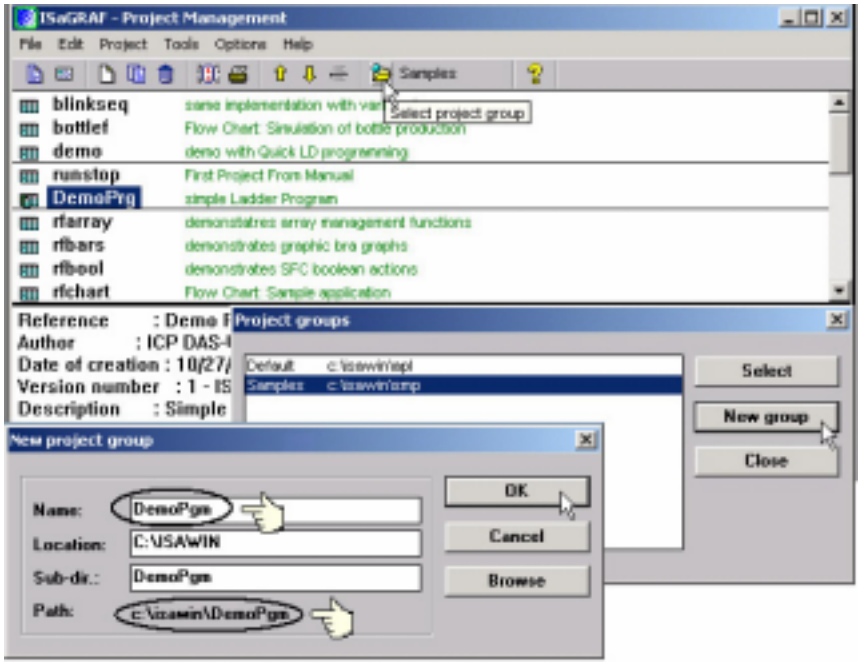
Starting & Running The ISaGRAF Workbench Program

Click on the Windows "Start" button, then click on "Programs", then click on "ISaGRAF 3.4", then click on "Projects" as shown below.



2.1.1.1: Creating An ISaGRAF User's Group

Click on the "Select Program Group", and then click on "New Group", then type in the name for the new user's group you wish to create, and last click on "OK".

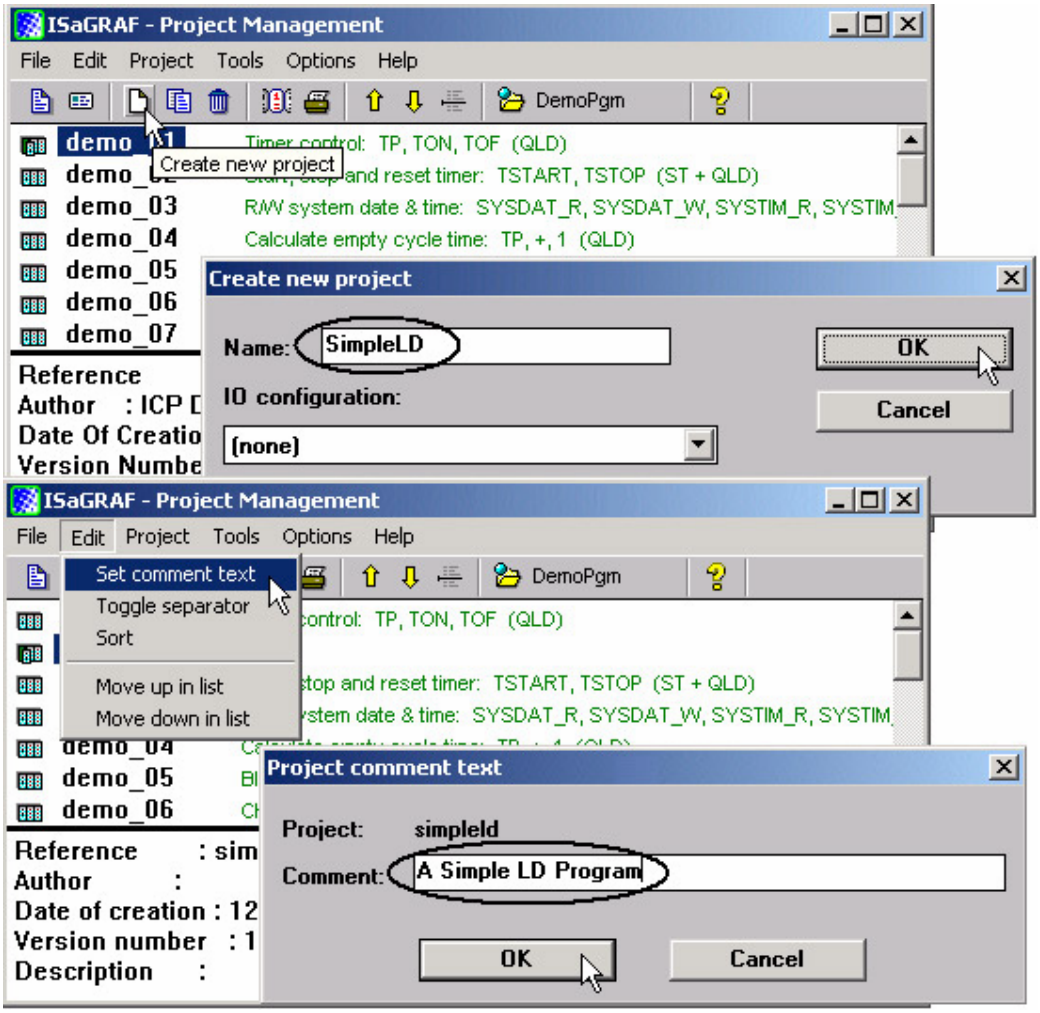


Note that the name that you give the "New Project Group" also creates a new sub-directory corresponding to the project group name in the "c:\isawin" sub-directory.

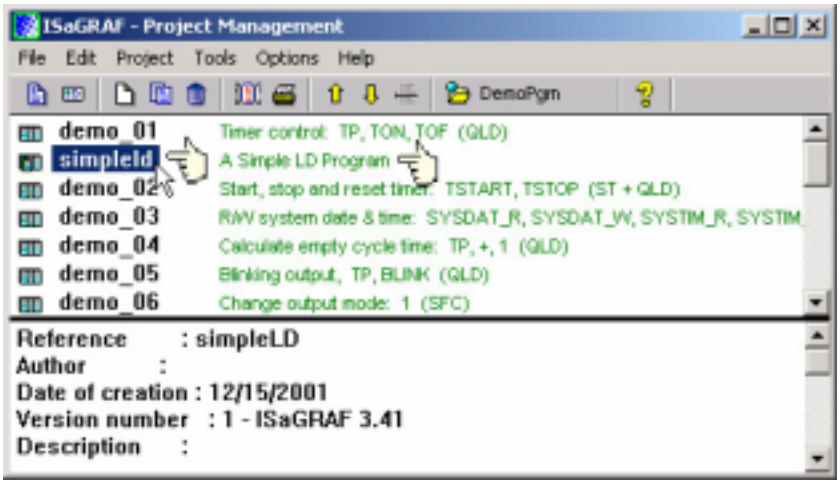
To start the new project, either double click on the new project name, or click on the new project name (the name will be highlighted) to select the new project group and click on the "Select" button.

2.1.1.2: Creating A New ISaGRAF Project

To start a new ISaGRAF project, click on the "Create New Project" icon and then enter in the name for the new project. You can then enter additional information for your project by clicking on the "Edit" and then "Set Comment Text" menu as illustrated below.

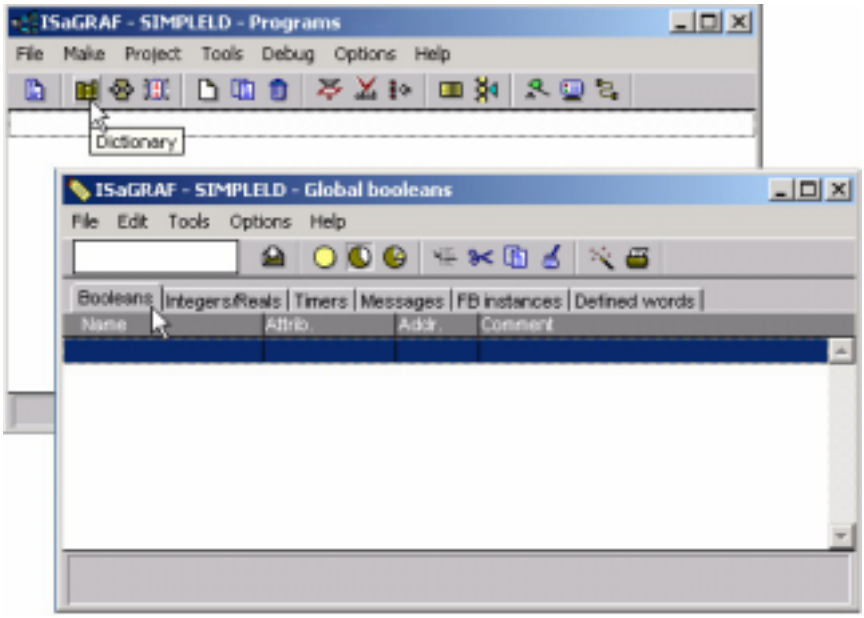


You will now see the name of the new project in the "Project Management" window. Double click on the name of the new project to open the new project.

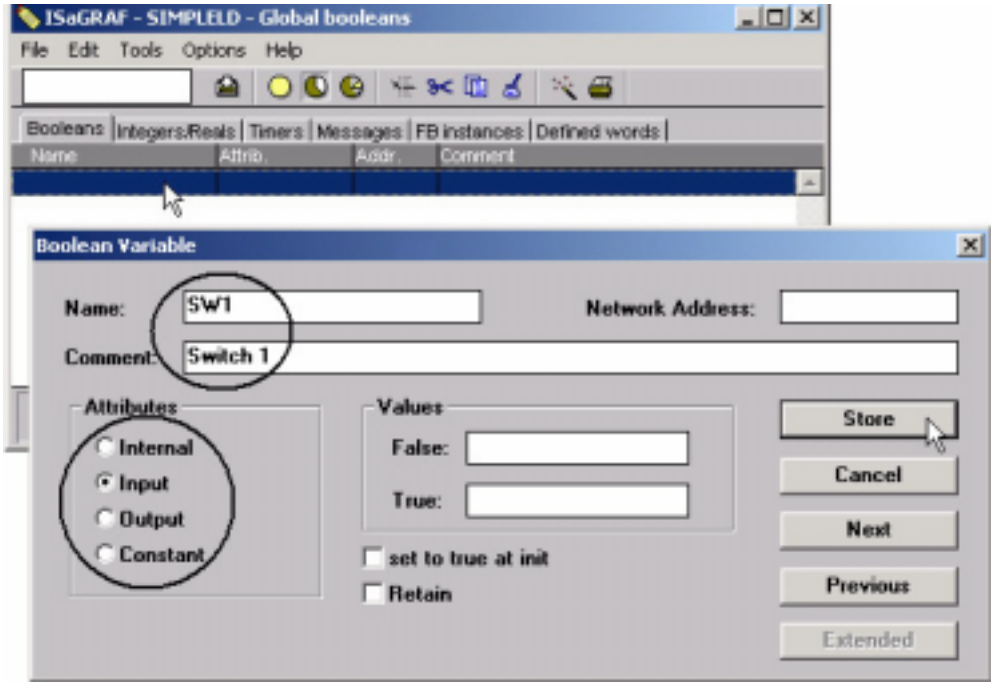


2.1.1.3: Declaring The ISaGRAF Project Variables

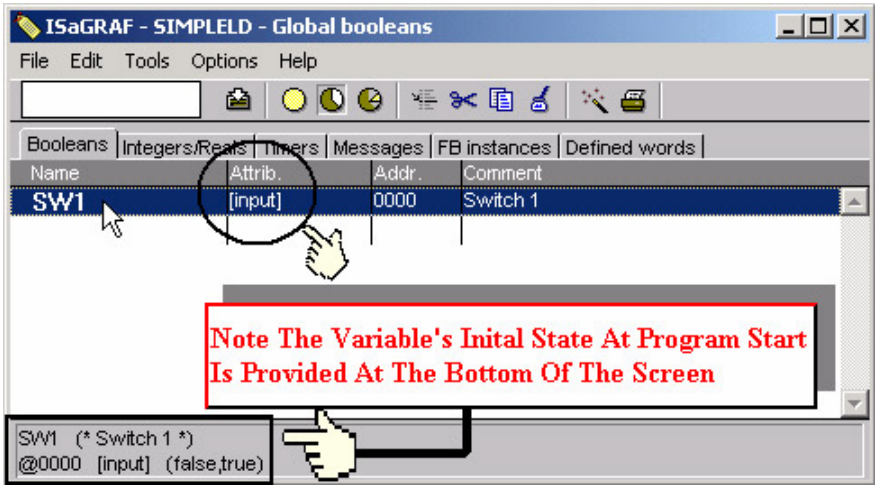
Before you can start creating an ISaGRAF program, you must first declare the variables that will be used in the ISaGRAF program. To begin this process, first click on the "Dictionary" icon and then click on the "Boolean" tab to declare the Boolean variables that will be used in our example program.



To declare the program variables for the ISaGRAF project, double click on the colored area below the "Boolean" tab, and a "Boolean Variable" window will open. Enter in the name of the variable to be used in the project. For the purpose of this example program the variable "Boolean Variable Name" is "SW1", and "Switch 1" is added to the "Comment Section". The next item that must be declared is what type of "Attribute" the variable will possess. In this example program, SW1's attribute will be an "Input". Lastly, press the "Store" button to save the Boolean variable that has been created.

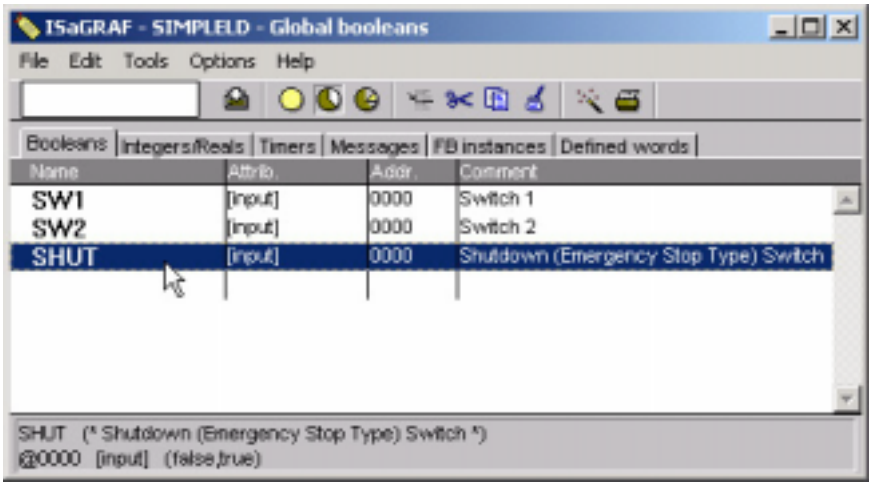


The new Boolean variable has now been declared. Note the other information areas that are provided for the programmer to fully explain how the variable will be handled.

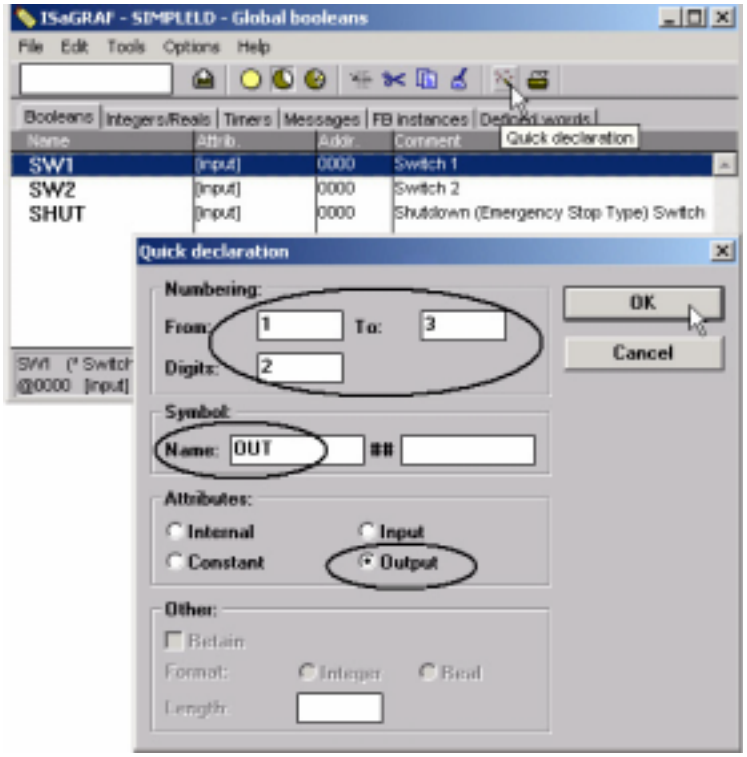


NOTE: You MUST make sure that the variable you have declared has the desired Attribute assigned. If you decide that you want to change a project variable's attribute, just double click on the variable name and you can reassign the attribute for the variable.

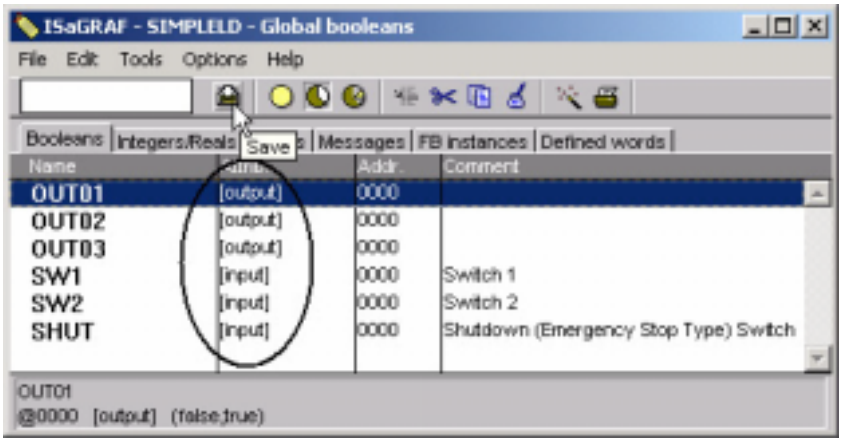
Using the same method described above, declare the additional Boolean variables for this example program, "SW2" and "SHUT". When you have completed the Boolean variable assignments, the Global Boolean window should look like the example below.



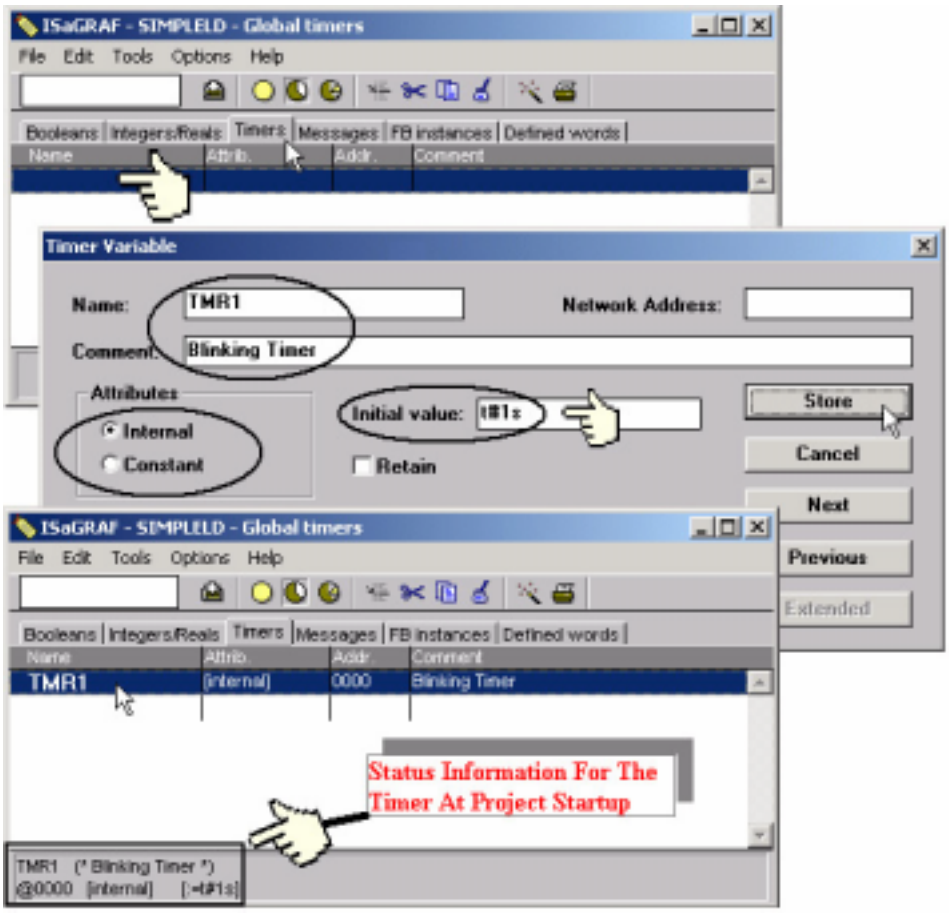
There are three outputs used in this example program named "OUT01, OUT02, and OUT03". ISaGRAF provides a quick and easy way to declare like variables that are sequentially ordered. To begin this process, click on the "Quick Declaration" icon, and enter in the output number that you will start with in the "Numbering" from and "To" field (this example uses from 1 to 3). Enter the "Symbol" name for the output variables being declared, and lastly, set the attribute to "Output".



When you click on the "OK" button, all three outputs will be immediately added to the "Global Boolean" window.



To declare the timer (TMR1) variable used in this example program, click on the "Timers" tab in the Global project setup screen. Double click on the colored area and enter the Name as "TMR1", set the "Attributes" to "Internal", the "Initial Value" to "T#1s", then click on the "Store" button.

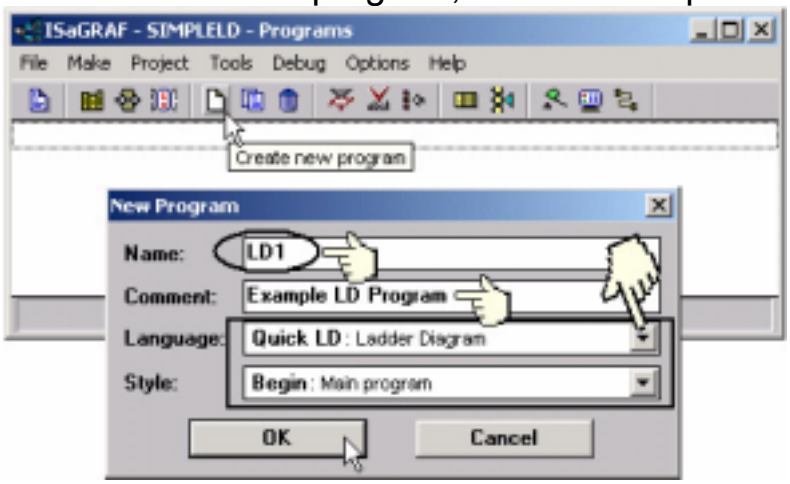


Once all of the timer variable characteristics have been properly setup, click on "X" at the top right of the Global timers window to close the variable dictionary for this example project.

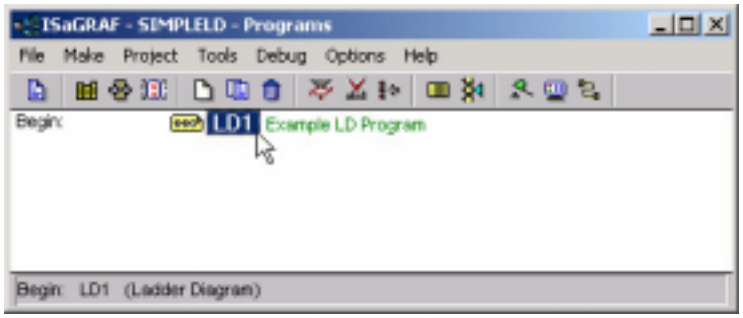
2.1.1.4: Creating The Example LD Program

Once all of the variables have been properly declared, you are now ready to create the example LD program. To start this process, click on the "Create New Program" icon and the "New Program" window will appear.

Enter the "Name" as "LD1" (the name of our example program), next, click on the "Language" scroll button and select "Quick LD: Ladder Diagram", and make sure the "Style" is set to "Begin: Main Program". You can add any desired text to the "Comment" section for the LD program, but it isn't required.

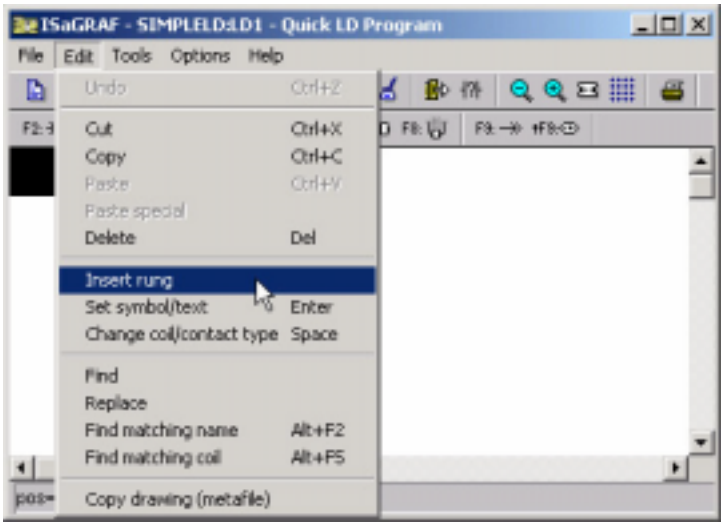


The "LD1" program has now been created. To open the "LD1" program, double click on the "LD1" name.

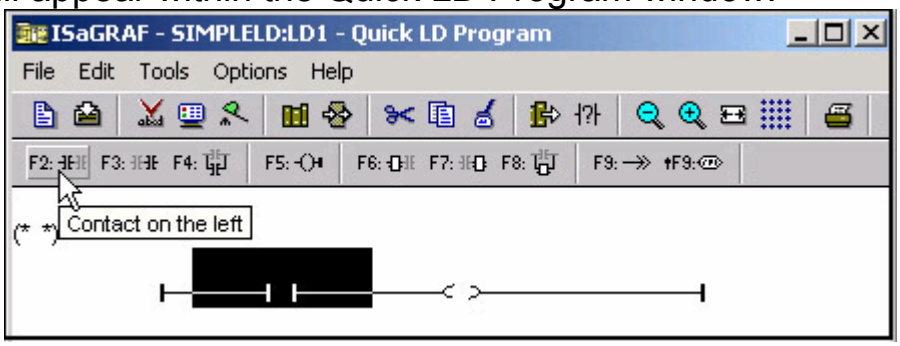


2.1.1.5: Editing The Example "LD1" Program

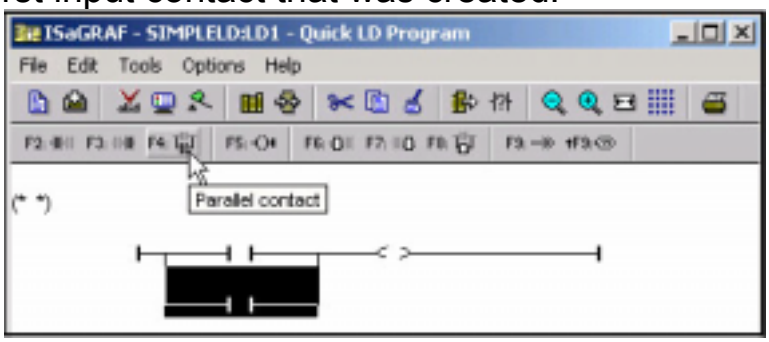
When you double click on the "LD1" name the "Quick LD Program" window will appear. To start programming our LD program, click on "Edit" from the main menu bar, then click on "Insert Rung" as shown below. "Insert Rung" means to insert a basic LD rung just above the current position.



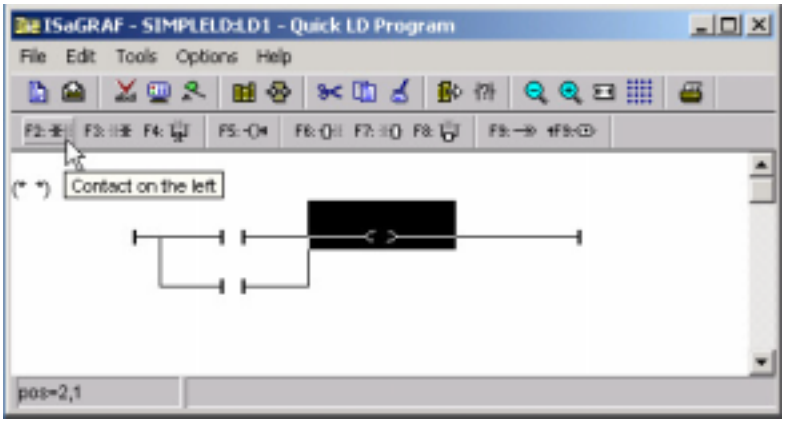
Or, you may just simply click on the "F2 (Contact On The Left)" icon, and the following will appear within the Quick LD Program window.



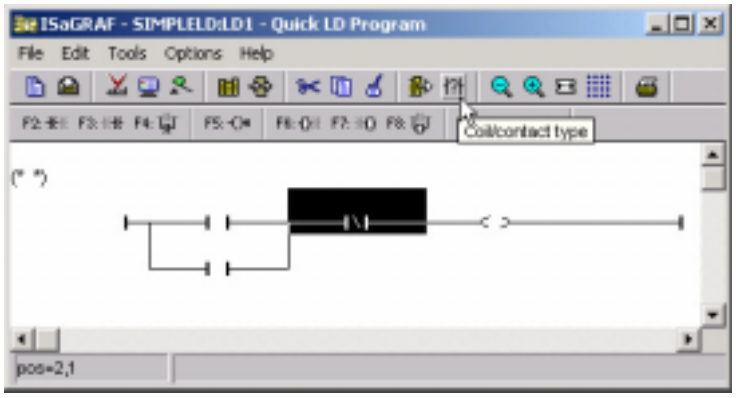
Click on the "F4 (Parallel Contact)" icon and you will add a parallel input contact below the first input contact that was created.



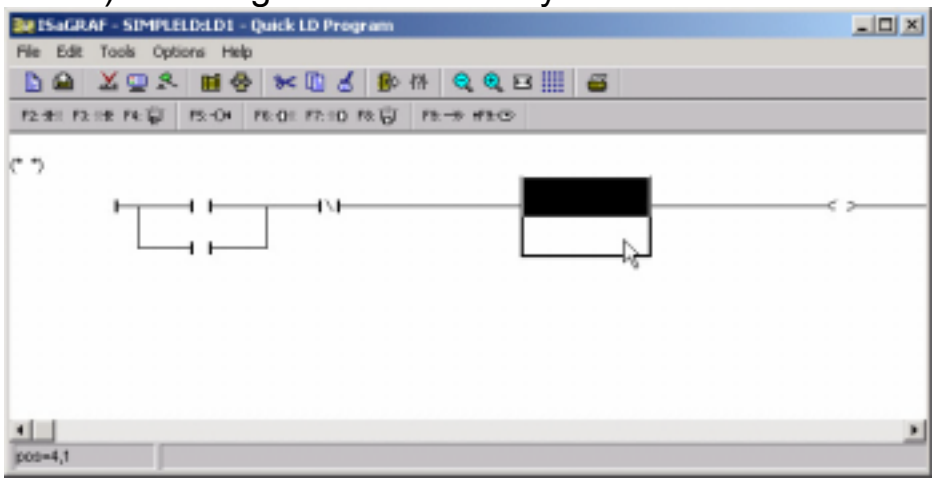
Click on the coil contact at the end of the LD rung and then click on the "F2 (Contact On The Left)" icon.



A new normally open input contact to the left of the output coil now appears. Click on the "Coil/Contact Type" icon to change the normally open contact to a normally closed contact.

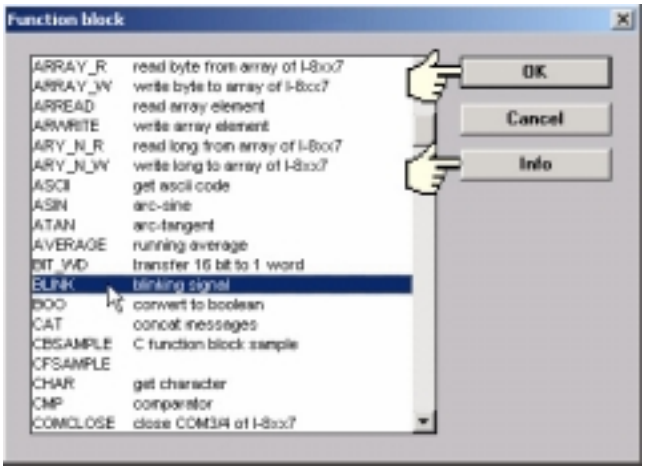


Click on the "F7 (Block On The Right)" icon to add a function block (which will be used for the timer) to the right of the normally closed contact.

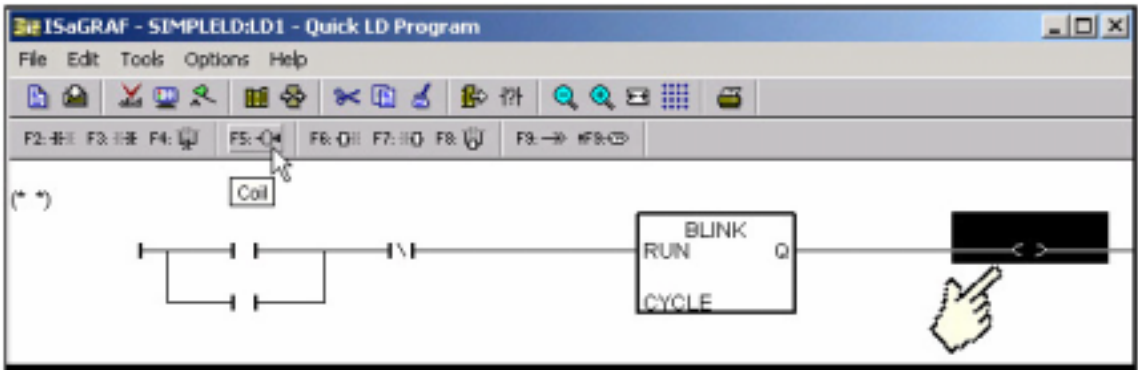


Double click anywhere inside of the new function block and the "Function Block" assignment window appears. Select the "BLINK" type function block for the type of timer we are using in our example program. To learn how the "BLINK" function

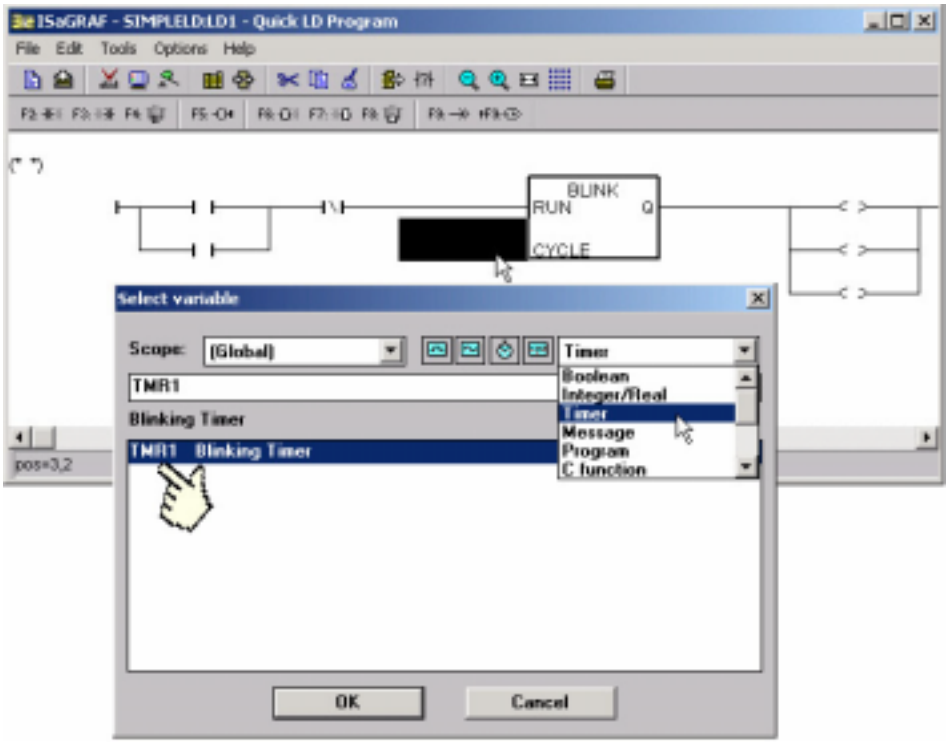
operates you can click on the "Info" button for a detailed explanation of its functionality.



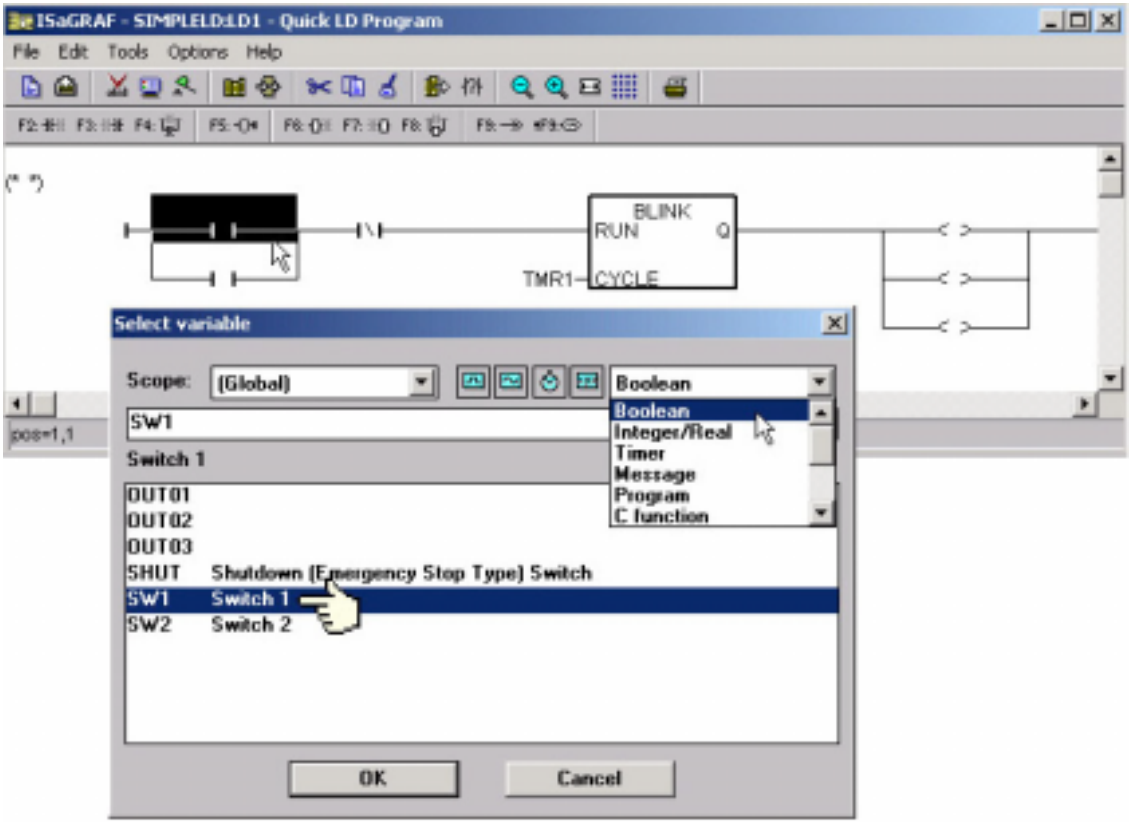
Now move your cursor to the output coil on the right side of the LD program. Click on the "F5 (Coil) icon two times to add two additional outputs in parallel with the first output.



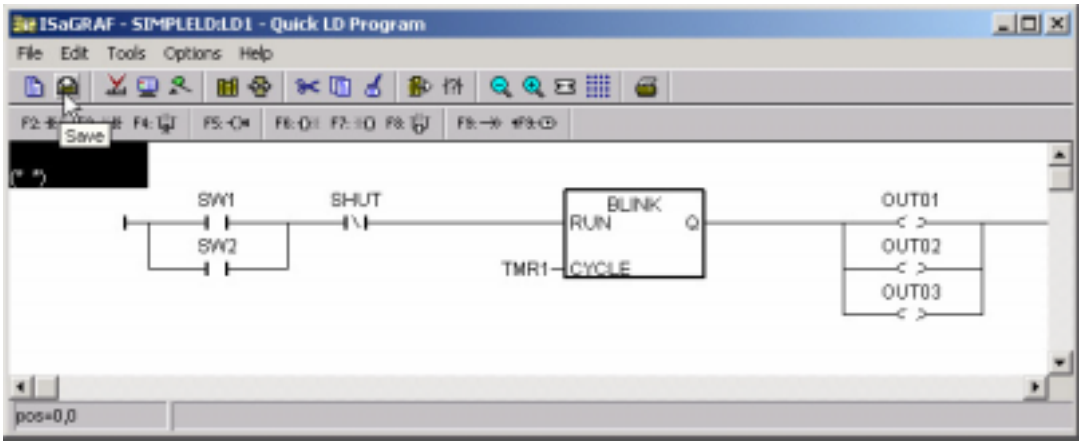
After adding the two additional outputs, move your cursor to the left of the timer function block to where the word "CYCLE" is and double click at that position.



Now we are ready to assign our program variables to each of the program components. Place the cursor over the first normally open switch as shown below then double click on the contact. A "Select Variable" window will now open.



Using the same method as described above, now assign the rest of the program variables to the contacts and coils in the example program. Lastly, remember to click on the "Save" button to complete the programming of the example LD program. Your program should now look like the below illustration.

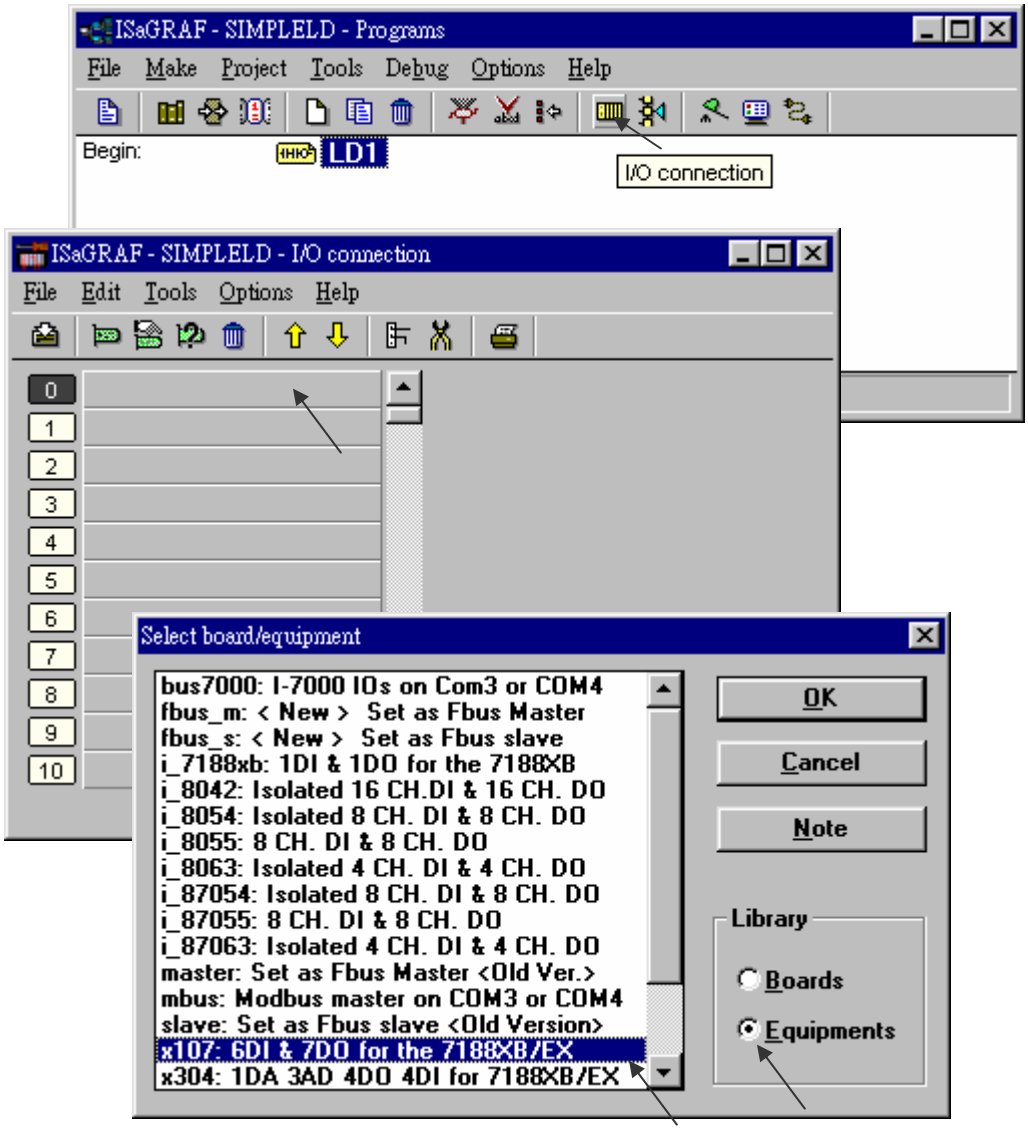


2.1.2: Connecting The I/O

The ISaGRAF Workbench software program is an open programming system. This allows the user to create an ISaGRAF program that can operate a large number of different PLC controller systems. It is the responsibility of the PLC hardware manufacturer to embed the ISaGRAF "kernel" in their respective controller for the ISaGRAF program to operate properly. The ICP DAS product line of I-7188XG , I-7188EG & I-8417 / 8817 / 8437 / 8837 series of controllers has the ISaGRAF kernel embedded, creating a powerful and flexible industrial controller system.

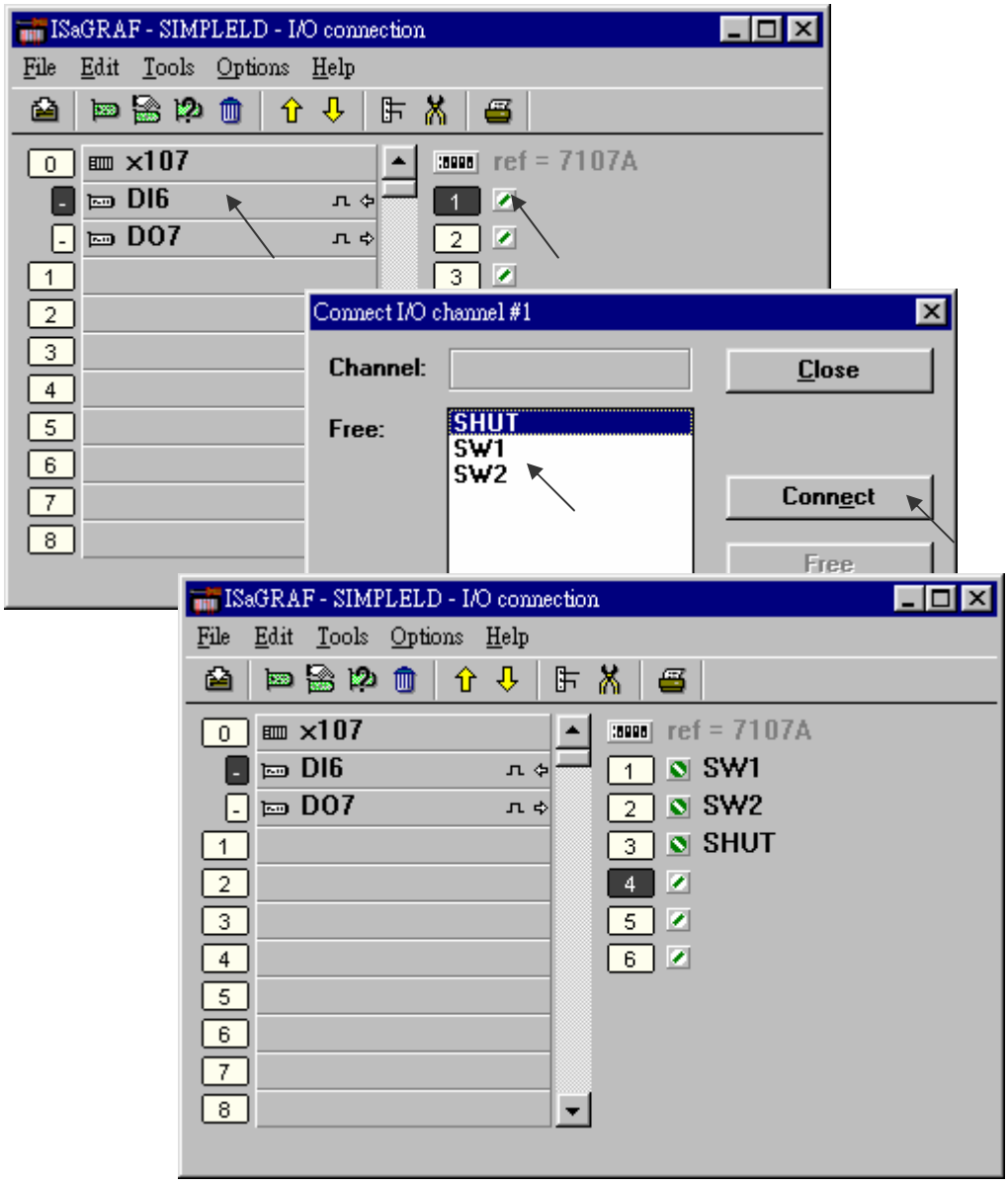
Now that you have created the ISaGRAF example program, now you must connect the "LD1" example program to the I-7188XG / I-7188EG I/O controller system.

Click on the "I/O Connection" icon as shown in the below picture and the "I/O Connection" window will appear as shown in the next illustration. In this example, if you have a “X107” I/O expansion board (please refer to cataloge or section 1.7), you should double click on the "**0**" slot for “X107”, however If you don’t have “X107”, just double click on any slot for “xboo_io” (simulate boolean I/O), then "Set Board/Equipment" window will appear. Select “Equipment” and double click on “X107” or “xboo_io”.



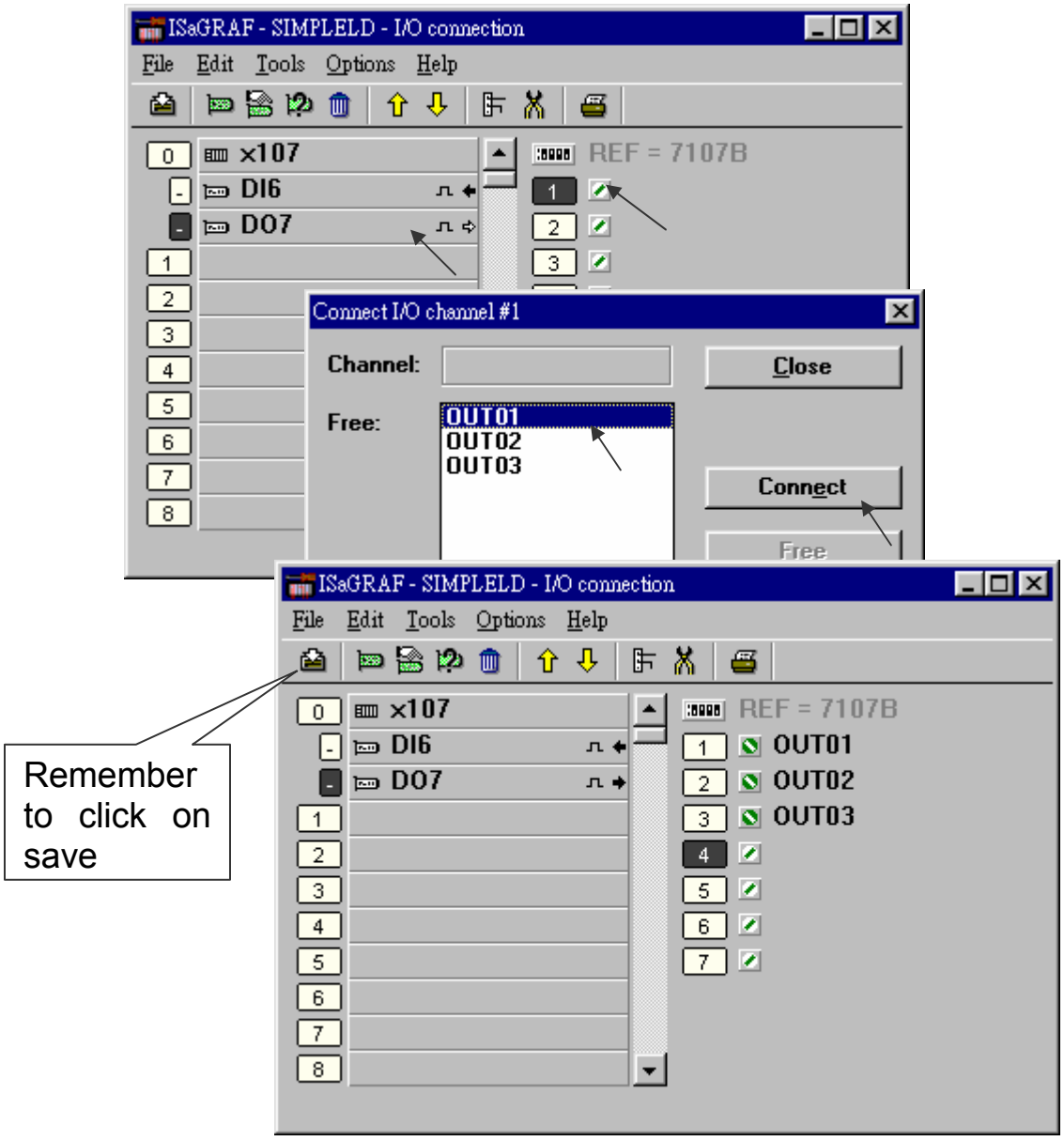
IMPORTANT NOTICE: Slot 0 is reserved for I/O expansion boards (please refer to section 1.7) . You can use other slots for additional function.

To connect the Input attributed variables to “X107”, click on “DI6” and then double click on channel 1 on the right. Then select the name and click on “Connect”.



To connect the Output attributed variables to “X107”, click on “DO7” and then double click on channel 1 on the right. Then select the name and click on “Connect”.

Once you have completed making the input I/O connections, remember to click on the "SAVE" icon to save the I/O connections that have been created for the example program.

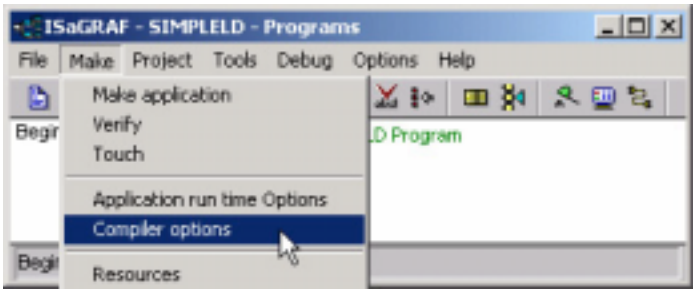


IMPORTANT NOTE: All of the Input and Output variables MUST be connected through the I/O connection as described above for any program to be successfully compiled. Only the Input and Output variables will appear in the "I/O Connections" window.

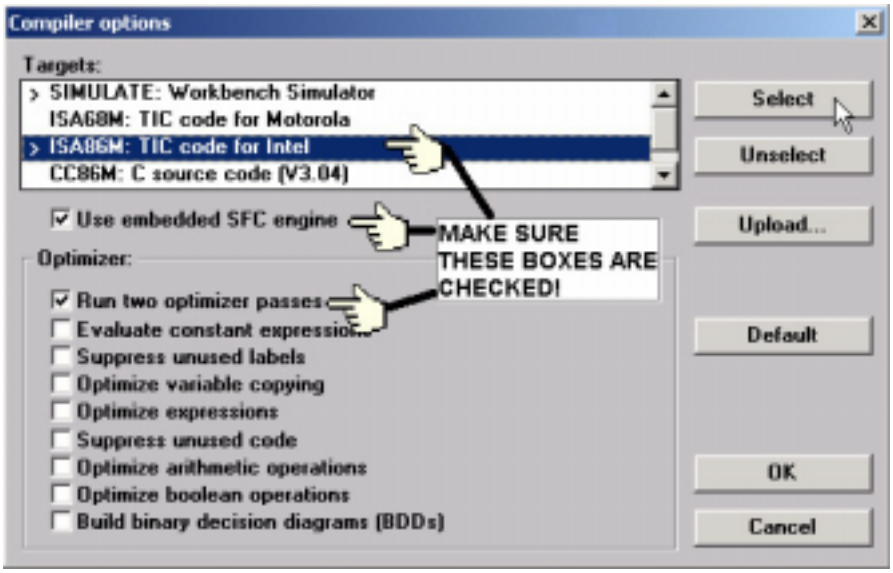
2.1.3: Compiling The Example LD Project

For ANY AND EVERY ISaGRAF program to work properly with any of the I-7188XG , I-7188EG & I-8417 / 8817 / 8437 / 8837 controller systems, it is the responsibility of the programmer to properly select the correct "Compiler Options". You MUST select the "ISA86M: TIC Code For Intel" option as described below.

To begin the compilation process, first click on the "MAKE" option from the main menu bar, and then click on "Compiler Options" as shown below.



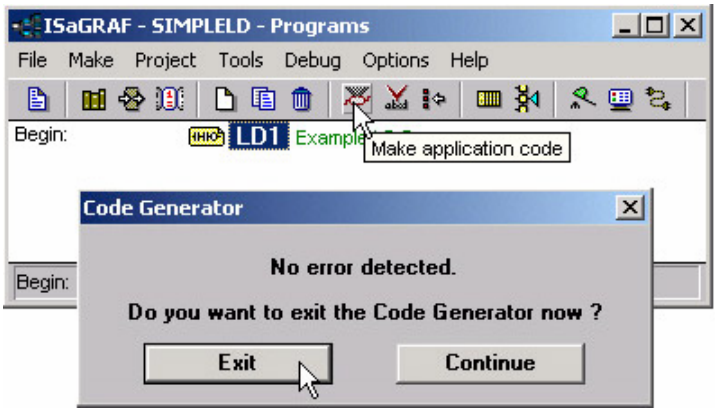
The "Compiler Options" window will now appear. Make sure to select the options as shown below then press the "OK" button to complete the compiler option selections.



TIME TO COMPILE THE PROJECT!

Now that you have selected the proper compiler options, click on the "Make Application Code" icon to compile the example LD project. If there are no compiler errors detected during the compilation process, CONGRATULATIONS, you have successfully created our example LD program.

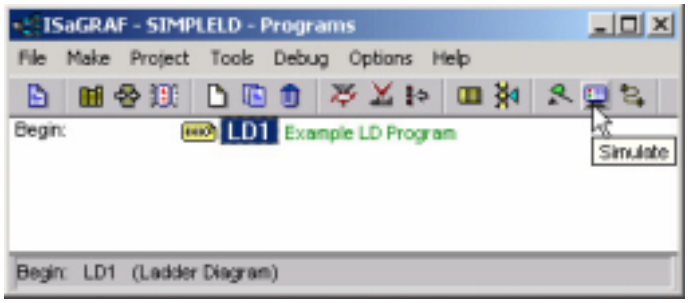
If errors are detected during the compilation process, just click on the "CONTINUE" button to review the error messages. Return to the Project Editor and correct the errors as outlined in the error message window.



2.1.4: Simulating The LD Project

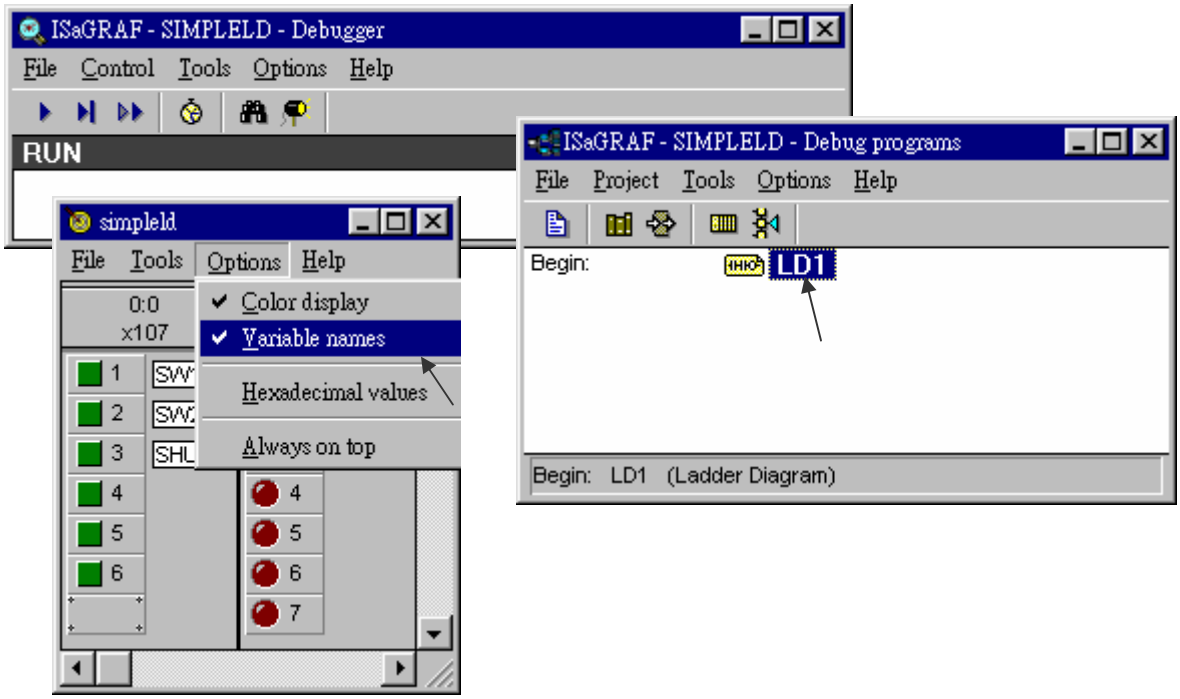
A powerful program-debugging feature of the ISaGRAF software program is the ability to "SIMULATE" the program you have developed before loading it into the I-7188XG / I-7188EG controller system.

After successfully compiling the example LD program, click on the "SIMULATE" icon as shown below.



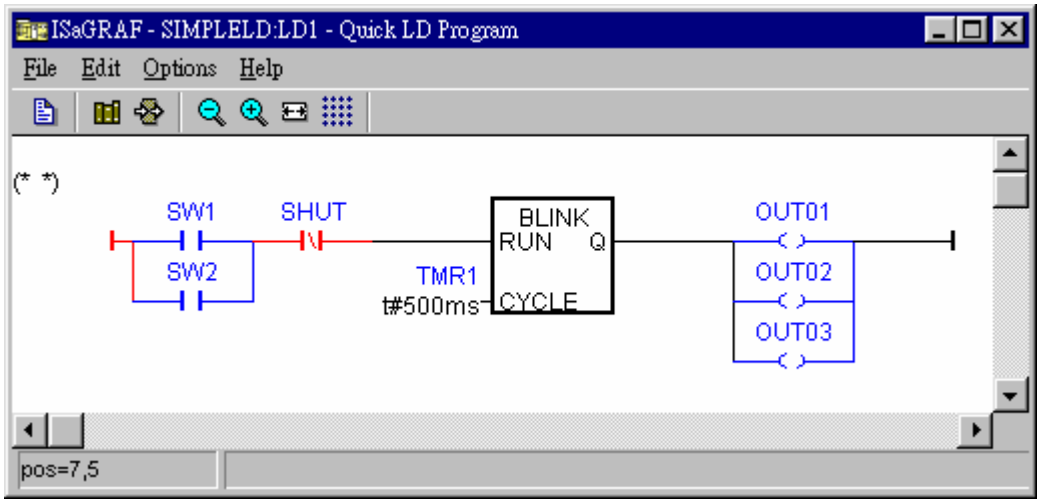
When you click on the "Simulate" icon three windows will appear. The windows are the "ISaGRAF Debugger", the "ISaGRAF Debug Programs", and the "I/O Simulator" windows. If the I/O variable names you have created DO NOT appear in the I/O simulator window, just click on the "Options" and "Variable Names" selection and the variable names you have created will now appear next to each of the I/O's in the simulator window.

In the "ISaGRAF Debug Program" window, double click on the "LD1" where the cursor below is positioned. This will open up the ISaGRAF Quick LD Program window and you can see the LD program you have created.



Running The Simulation Program

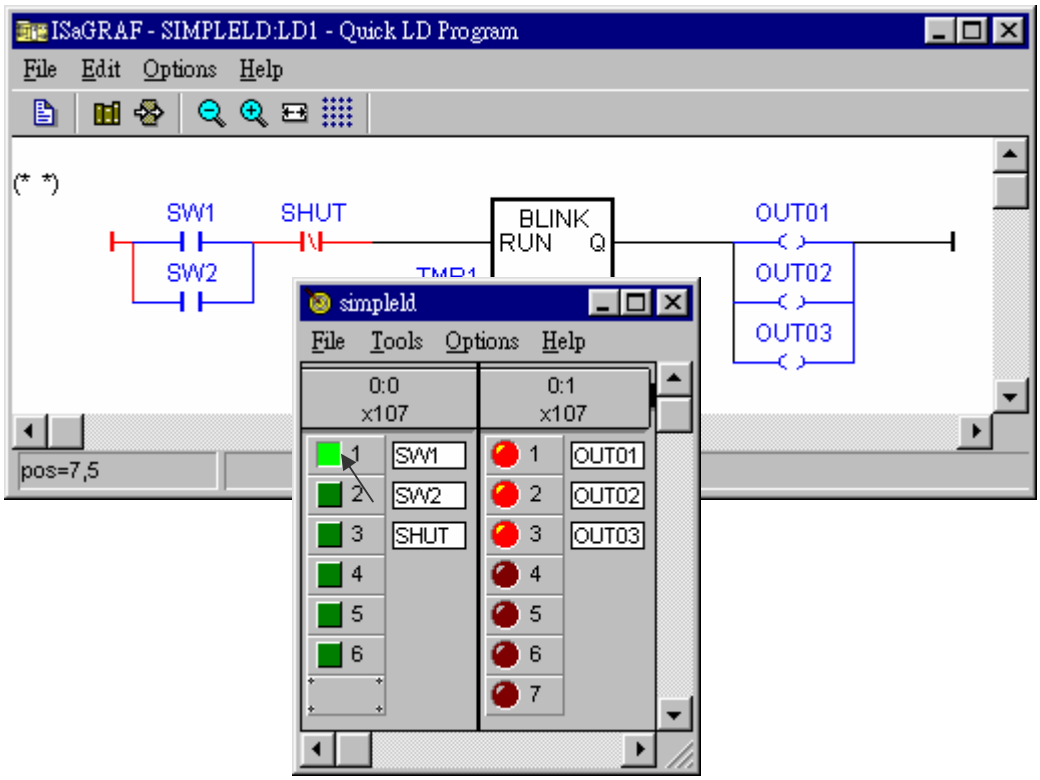
When you double click on "LD1" in the "ISaGRAF Debug Programs" window, the follow window should appear.



IMPORTANT TIP

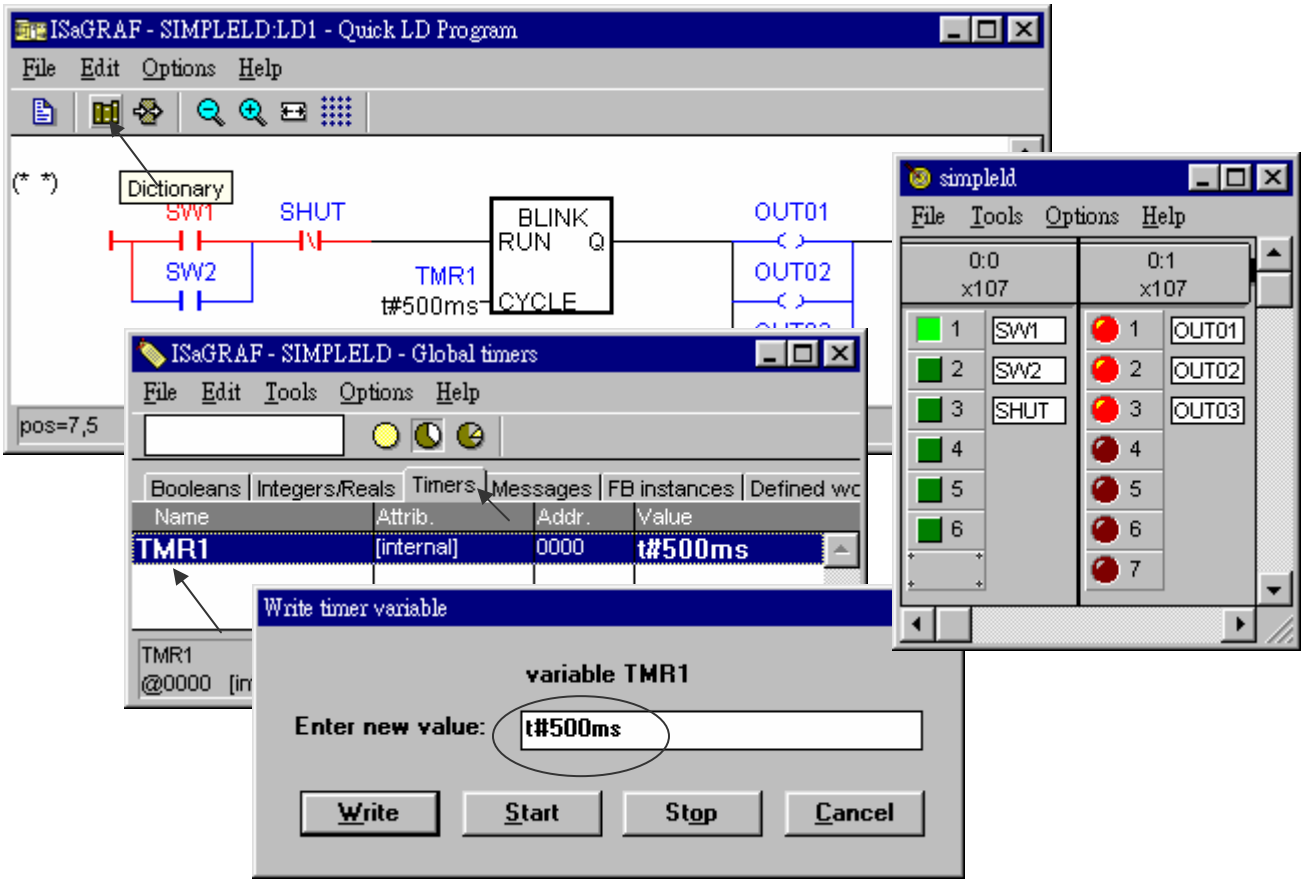
Note the colors of the I/O in the following example above. "SW1" and "SW2" are normally open switches that have not been energized so their color is blue, but the "SHUT" is a normally closed switch and its color is red because it is energized by default.

To see the example LD program run in the simulator window, click on either the "SW1" or "SW2" button in the "I/O Simulator" window.



In the example above you see that "SW1" button has been turned on which allows the logic (power flow) to go true for the example LD program. When either "SW1" or "SW2" IS energized (their respective green buttons are pushed in the "I/O Simulator" window), and the "SHUT" switch button IS NOT on (button 3 remains off), this creates a true state for the logic to flow through the example LD circuit. Now "OUT1", "OUT2", and "OUT3" will now turn on and off in one-second intervals as defined by the "TMR1" variable.

You can adjust the "TMR1" variable while the program is running. To accomplish this, click on the "Dictionary" icon in the "ISaGRAF Quick LD Program" window which will open the "ISaGRAF Global Variables" window as shown in the first two pictures below.

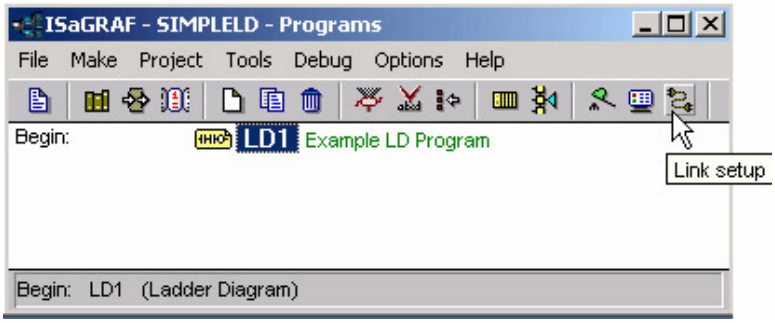


When the "ISaGRAF Global Variables" window opens, click on the "Timers" tab, and then double click on the "TMR1" name, this will open the "Write Timer Variable" window. Change the "Enter New Value:" from "t#1s" to t#500ms" and click on the "Write" button.

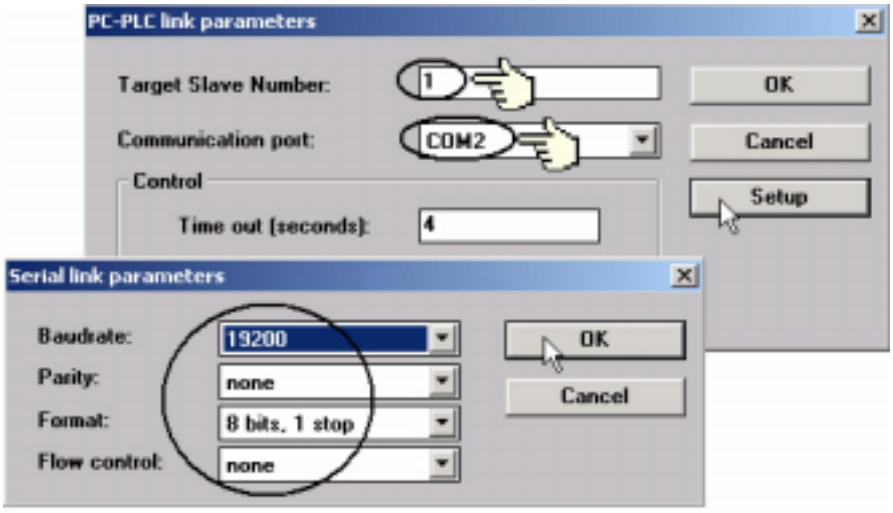
Now when you click on either "SW1" or "SW2" button in the I/O simulator the outputs will be turned on and off every 500 milliseconds (1/2 second) versus the previous setting of every 1-second.

2.1.5: Downloading & Debugging The Example LD Project

The last step required to running the example LD program on the I-7188XG & I-7188EG controller systems is to download the project to the controller (frequently referred to as the "Target" platform"). Before this download can be accomplished you must first establish communications between your development PC and the I-7188XG/I-7188EG controller.



To begin this process, click on the "Link Setup" icon in the "ISaGRAF Programs" window. When you click on the "Link Setup" icon, the following window will appear.



The "Target Slave Number" is the Node-ID address for the I-7188XG / I-7188EG controller system, Default NET-ID is 1, to change the NET-ID, please refer to Section 1.3.1 . The "Communication Port" is the serial port connection on your development PC, and this is normally either COM1 or COM2.

The communication parameters for the target controller MUST be set to the same serial communication parameters for the development PC. For I-7188XG / I-7188EG controllers (serial port communications), the default parameters for COM1 port are:

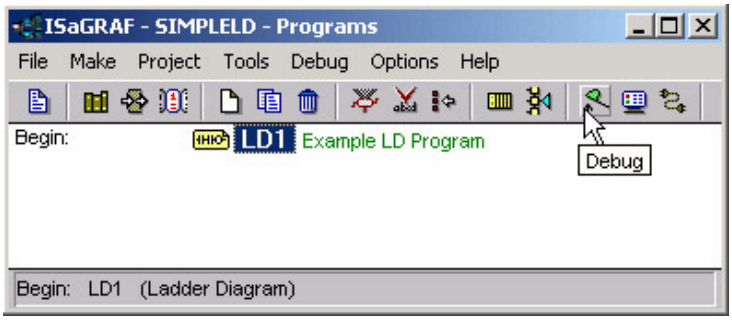
Baudrate:	19200
Parity:	none
Format:	8 bits, 1 stop
Flow control:	none

IMPORTANT NOTE

It may be necessary to change the COM port settings for the development PC. Depending on which computer operating system you are using, you will need to make sure that the COM port can properly communicate to the I-7188XG / I-7188EG controller system.

DOWNLOADING THE EXAMPLE LD PROJECT

Before you can download the LD project to the I-7188XG / I-7188EG controller system, you must first verify that your development PC and the I-7188XG / I-7188EG controller system are communicating with each other. To verify proper communication, click on the "Debug" icon in the "ISaGRAF Programs" window as shown below.

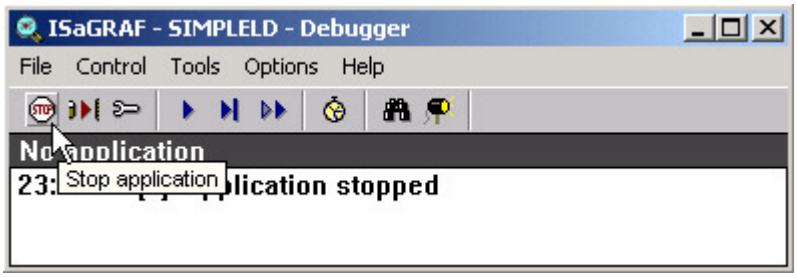


If the development PC and the I-7188XG / I-7188EG controller system are communicating properly with each other, the following window displayed below will appear (or if a program is already loaded in the I-7188XG / I-7188EG controller system, the name of the project will be displayed with the word "Active" following it.

If the message in the "ISaGRAF Debugger" says "Disconnected", it means that the development PC and the I-7188XG / I-7188EG controller system have not established communications with each other.

The most common causes for this problem is either the serial port cable not being properly configured, or the development PC's serial port communications DO NOT match that of the I-7188XG / I-7188EG controller system.

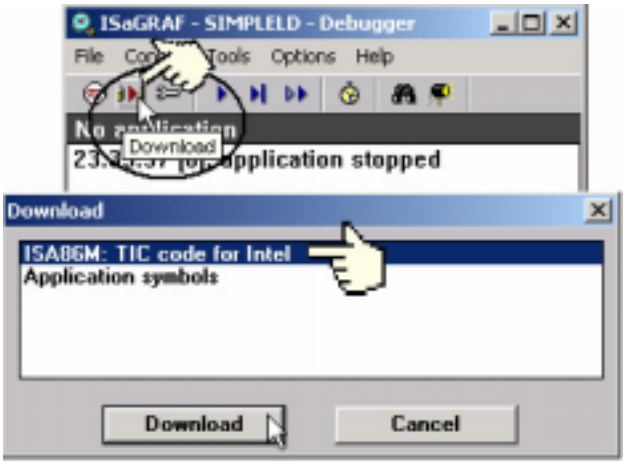
You may have to either change the serial port communication settings for the development PC (which may require changing a BIOS setting) or change the "Serial Link Parameters" in the ISaGRAF program.



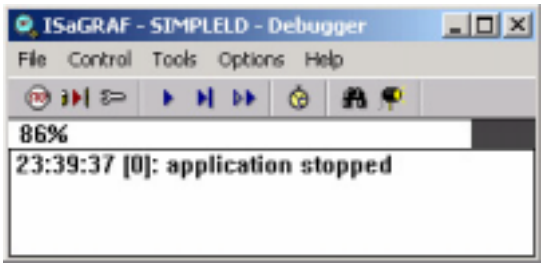
If there is a program already loaded in the I-7188XG / I-7188EG controller system you will need to stop that program before you can download the example LD program. Click on the "STOP" icon as illustrated above to halt any applications that may be running.

STARTING THE DOWNLOADING PROCESS

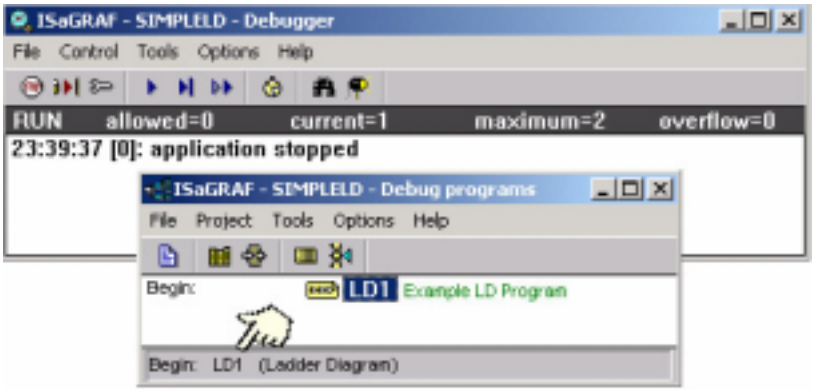
From the "ISaGRAF Debugger" window click on the "Download" icon, then click on "ISA86M: TIC Code For Intel" from the "Download" window as shown below.



The example LD program will now start downloading to the I-7188XG / I-7188EG controller system. A progress bar will appear in the "ISaGRAF Debugger" window showing the program downloading progress.

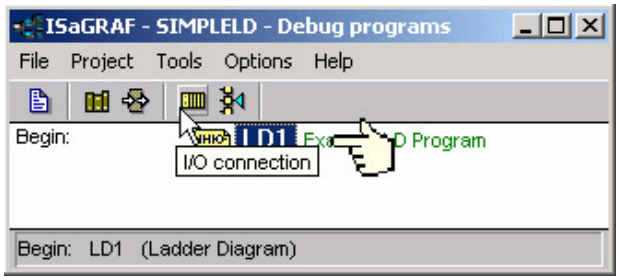


When the example LD program has successfully completed the downloading process to the I-7188XG / I-7188EG controller system, the following two windows will appear.

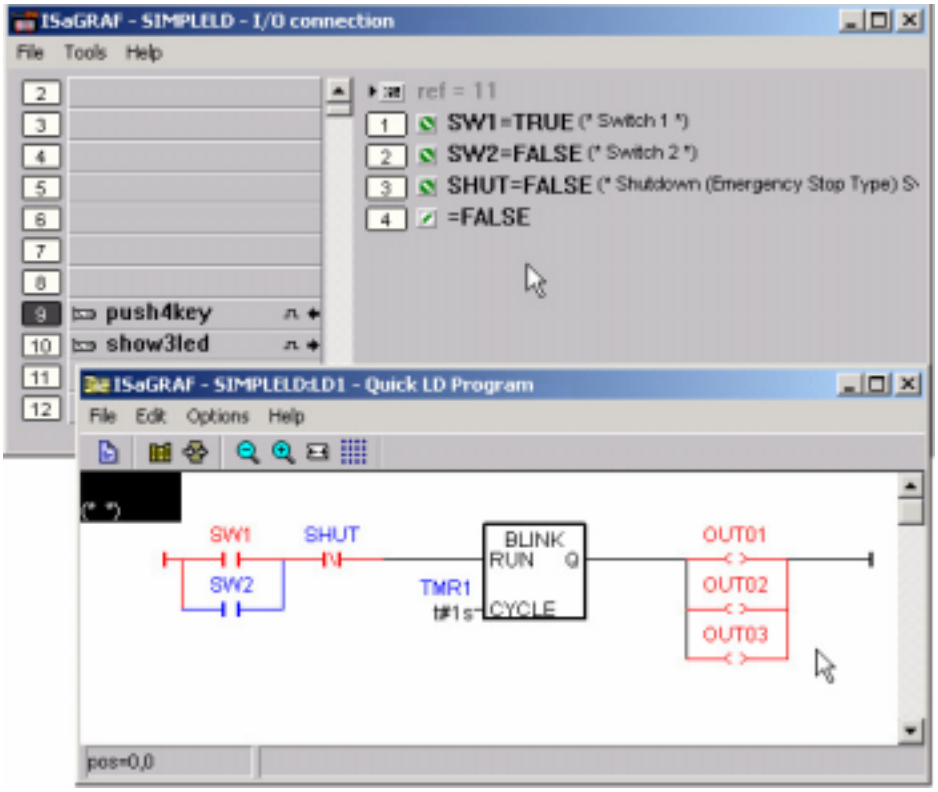


RUNNING THE EXAMPLE LD PROGRAM

You can observe the real time I/O status from several ISaGRAF windows while you are running the example LD program. One of the windows is the "I/O Connections" window, which shows each of the inputs and outputs as assigned. Click on the "I/O Connections" icon in the ISaGRAF Debugger window to open the "I/O Connections" screen. Another VERY helpful window you can open is the "Quick LD Program" window. From this window you can observe the LD program being executed in real time.



In the window below, the "SW1" switch is pressed which is creating a true logic state for the outputs to be turned on and off (blinked) at a one second interval. The "Quick LD Program" window shows the entire ladder logic program in REAL TIME and is an excellent diagnostic tool for development and troubleshooting.



Though there are numerous steps involved in creating and downloading an ISaGRAF program, each step is quick and easy to accomplish, and the end result is a powerful and flexible control development environment for the I-7188XG / I-7188EG controller systems.

PRACTICE, PRACTICE, PRACTICE!

Now that you have successfully created and ran your first ISaGRAF program with the I-7188X / I-7188EG controller system, you should practice creating more elaborate and powerful programs. Like any other computer development environment, practice and experimentation is the key to understanding and success, GOOD LUCK!

2.2: A Simple Function Block Diagram (FBD) Program

2.3: A Simple Structured Text (ST) Program

2.4: A Simple Instruction List (IL) Program

2.5: A Simple Sequential Function Chart (SFC) Program

Note:
Please refer to “User’s Manual Of I-8417 / 8817 / 8437 / 8837 ISaGRAF Embedded Controllers” for simple programs of FDB, ST, IL & SFC language, or refer to “Napdos\ISaGRAF\8000\English_Manu\user_manu_I_8xx7.pdf”.

Chapter 3: Establishing I/O Connections

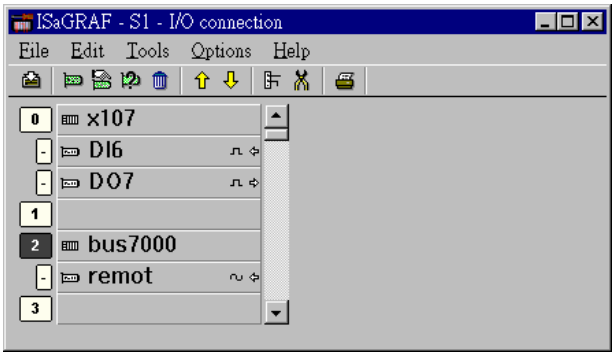
Before you can operate an ISaGRAF program with the I-7188XG / I-7188EG controller system, you must make sure that the I-7188XG / I-7188EG Library has been installed. If you haven't done so already, please refer to Section 1.2 ".

3.1: Linking I/O Boards To An ISaGRAF Project

The numbers on the left of the "I/O Connections" window indicate the slot number. **Slots 0 is only for I/O Expansion boards, such as X107, X304 & X507~X509** (refer to section 1.7). Slots 1 and above can be used for "I-7188xb" or "virtual" I/O boards such as the "bus7000" and "mbus" functions.

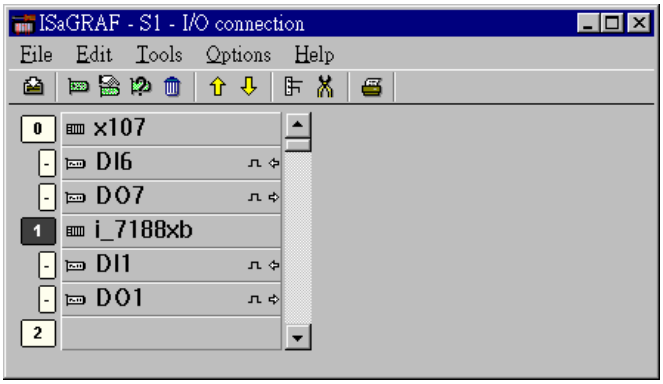
In the below example I/O connection we are using the I-7188EG controller system that has the X107 expansion board installed and has connected to some I-7000 modules(please refer to section 1.4 & chapter 6).

Slot 0: X107 expansion Board (6 digital inputs & 7 digital outputs)
Slot 2: bus7000 (for I-7000 & I-87k remote I/O modules)



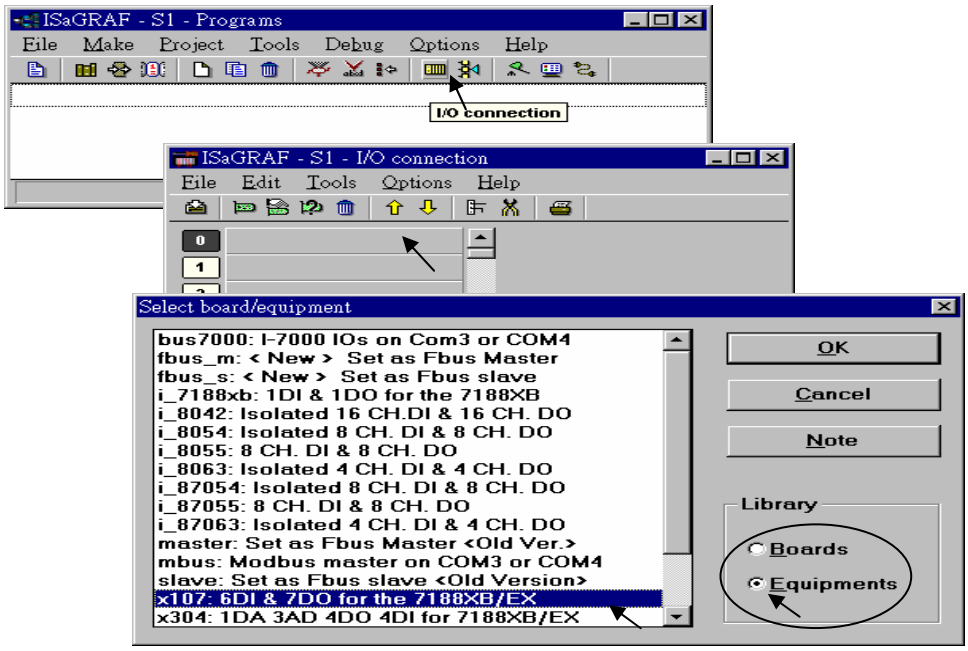
The second example is using the I-7188XG controller system has the the X107 expansion board installed, and the "I-7188xb" is the built-in one Ch. D/I and one Ch. D/O inside the I-7188XG controller.

Slot 0: X107 expansion Board (6 digital inputs & 7 digital outputs)
Slot 1: I-7188xb (built-in one Ch. D/I and one Ch. D/O inside the I-7188XG)



3.1.1: Linking I/O Boards

The I-7188XG / I-7188EG controller library defines two basic types of real I/O boards, "Boards" and "Equipments". The "Boards" selection is for I/O boards that are "single type", meaning that all of the channels on that board are of a single type and attribute. The "Equipments" selection is for I/O boards that are "multi-type", which means boards that have multiple types (such as the “X107” digital I/O expansion board that has 6 digital inputs and 7 digital outputs all on the same board). To begin the linking I/O board process, double click on the slot that you want to associate an I/O board to.



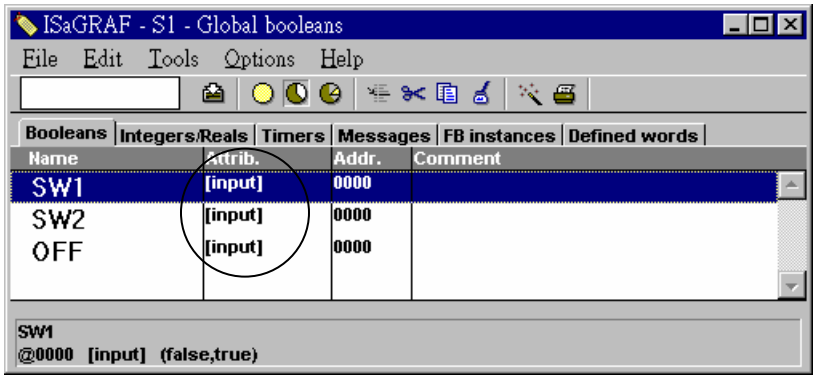
If you link an I/O board to an incorrect slot, first click on the slot number you wish to correct, then just click on the "Clear Slot" icon to delete the connection. The connection is now cleared, and now you can make a connection to the desired slot location.

3.1.2: Linking Input & Output Board Variables

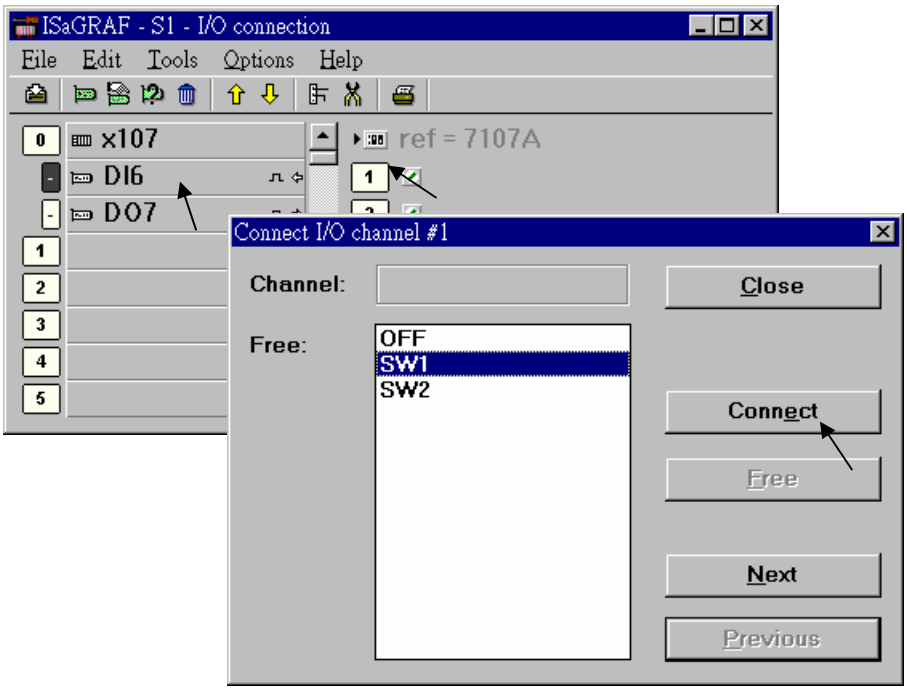
All of the **input and output** board "variables (or names)" must be linked (connected) in the "I/O Connection" window. Click on the slot you wish to link the variable to, then double click on the channel (or I/O point) number on the right hand portion of the "I/O Connection" window. Lastly, choose the variable name you wish to link to and then click on the "Connect" button.

IMPORTANT NOTE

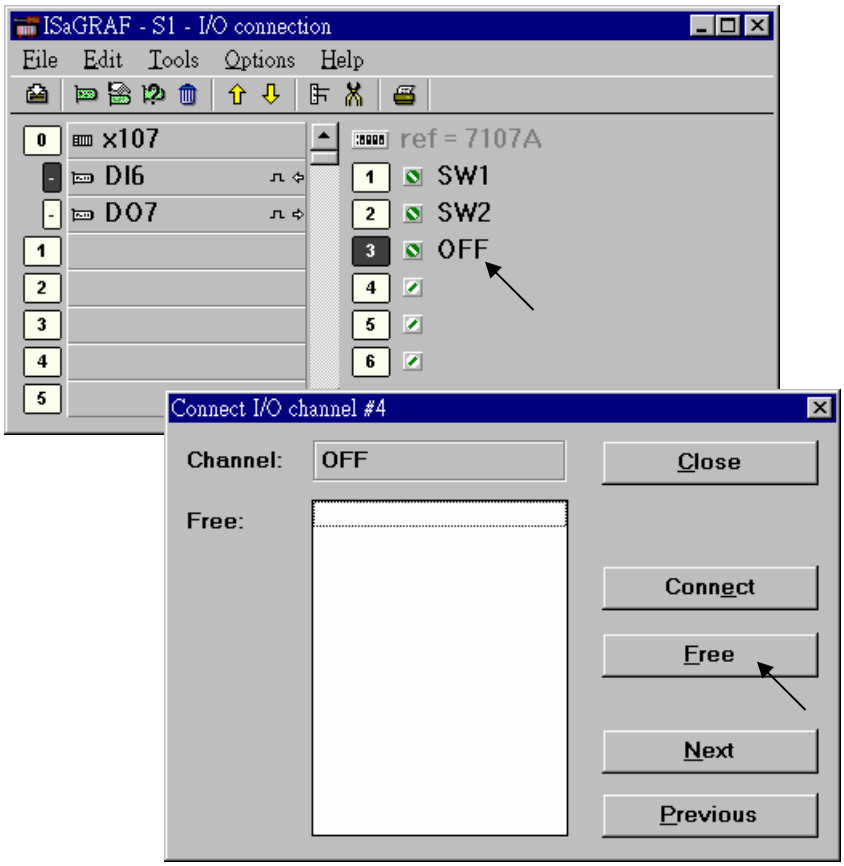
Remember that before you can assign any input or output, you must FIRST declare the variable in the "ISaGRAF Global Variables" window as shown below.



Click once on slot 0, then double click on "1" on the right hand side of the "ISaGRAF I/O Connection" window. With the "Connect I/O Channel #1" window now open, click on the "Connect" button to create the link between the variable "SW1" and channel number 1 of the "X107"-"DI6" input.



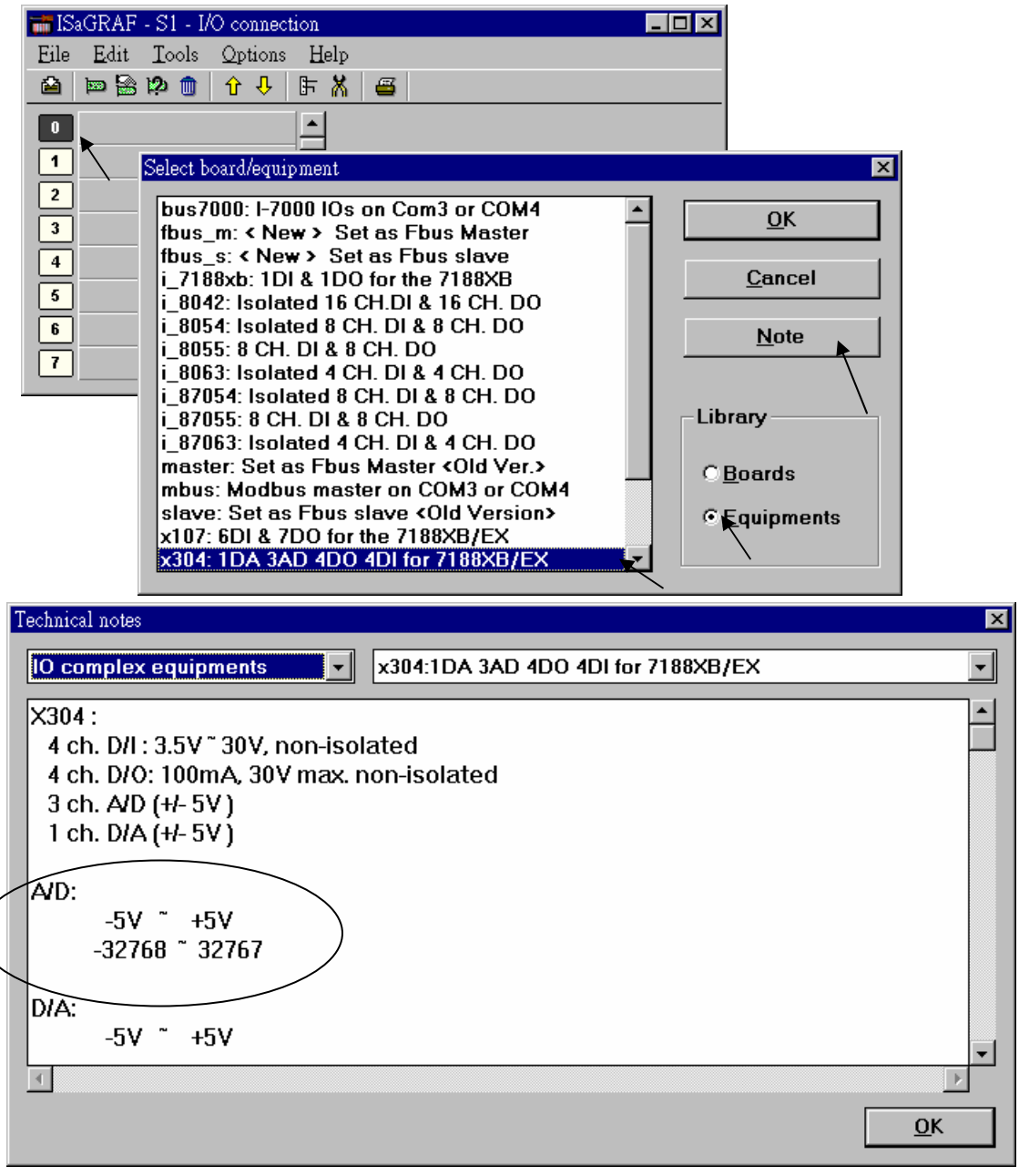
If you connect an input or an output variable to the wrong (or undesired) I/O location, double click on the I/O point you wish to remove. The "Connect I/O Channel #x" will open then click on the "Free" button to remove that variable from the I/O point.



When you click on the "Free" button you will see that the variable is removed from the I/O point in the "ISaGRAF I/O Connection" window and the variable is placed in the "Free" portion of the "Connect I/O Channel #x" window.

3.2: Linking Analog Type I/O Boards

The method to connect analog type I/O boards to the I-7188XG / I-7188EG controller system is very similar to that of connecting digital I/O boards. You may click on “Note” to see the A/D & D/A transfer table.



Chapter 4: Linking To An HMI Program

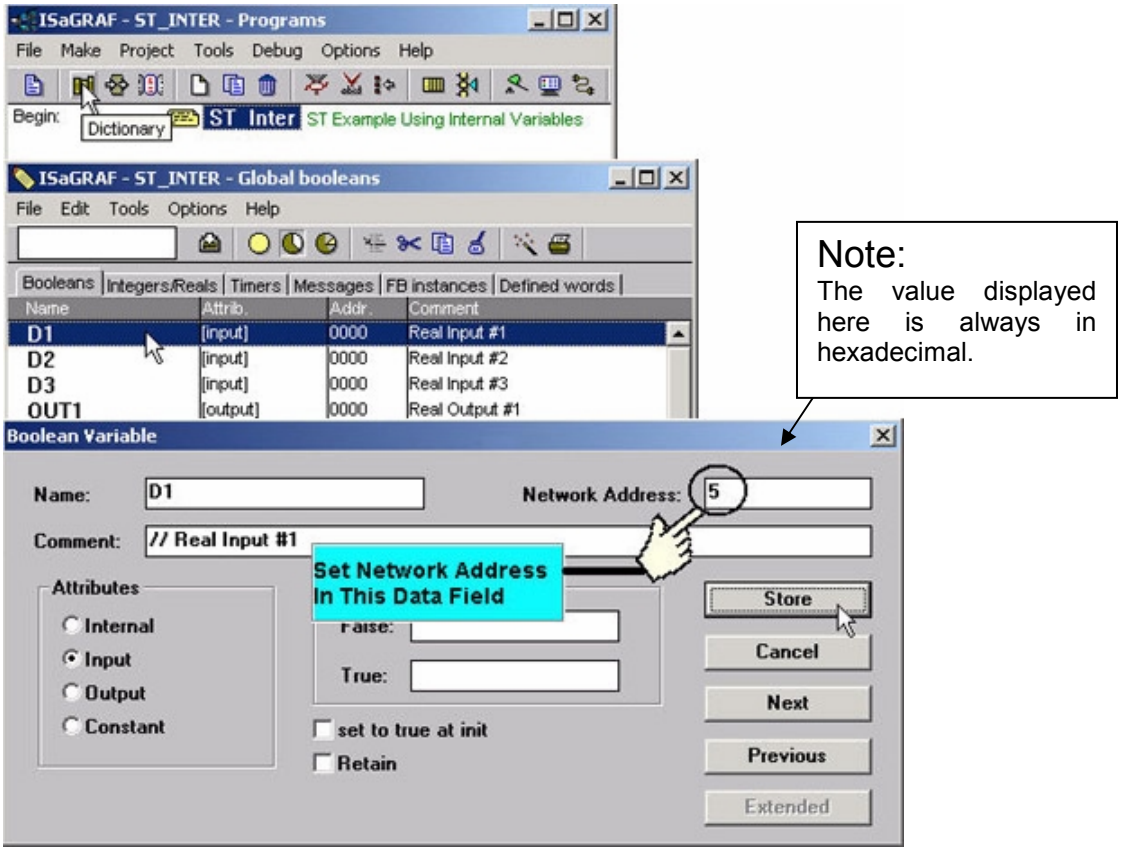
This chapter details how to make data from the I-7188XG / I-7188EG controller system available to Human Machine Interface (HMI) programs. This is a powerful feature that allows customers to create their own custom HMI programs and link them to the I-7188XG / I-7188EG controller system

After you realize the material described in section 4.1. Additionally there are "touch screen" monitors provided by ICP DAS that support the "Modbus" protocol, and these touch screen monitors can also access data from an I-7188XG / I-7188EG controller system. Please refer to Section 4.4 of “User Manual Of The I-8417/8817/8437/8837 ISaGRAF Embedded Controllers”.

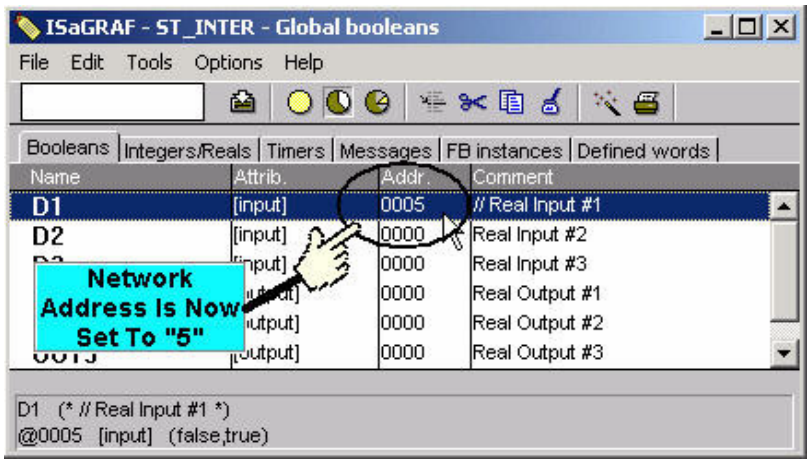
4.1: Declaring Variable Addresses For Network Access

To make data from an I-7188XG / I-7188EG controller system available to other software programs or HMI devices, you must first declare the variable with a "Network Address". The variable must be declared with a network address number that is in the "Modbus" format. **The valid network addresses for an I-7188XG / I-7188EG controller system is from 1 to FFF in hexadecimal (1 ~ 4095)**. Other software programs or HMI devices will access the I-7188XG / I-7188EG controller information through these network addresses.

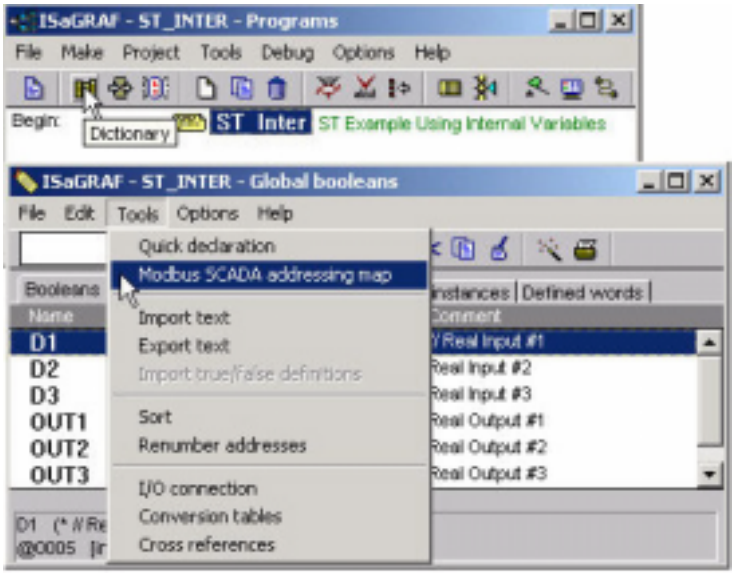
There are two methods available to declare a variable for network address access. The first method is described below. Open an "ISaGRAF Programs" windows and click on the "Dictionary" icon, then double click on the variable to assign a network address number.



When you click on the "Store" button you will see that "ISaGRAF Global Variables" window will now be updated with the new network address for the variable.

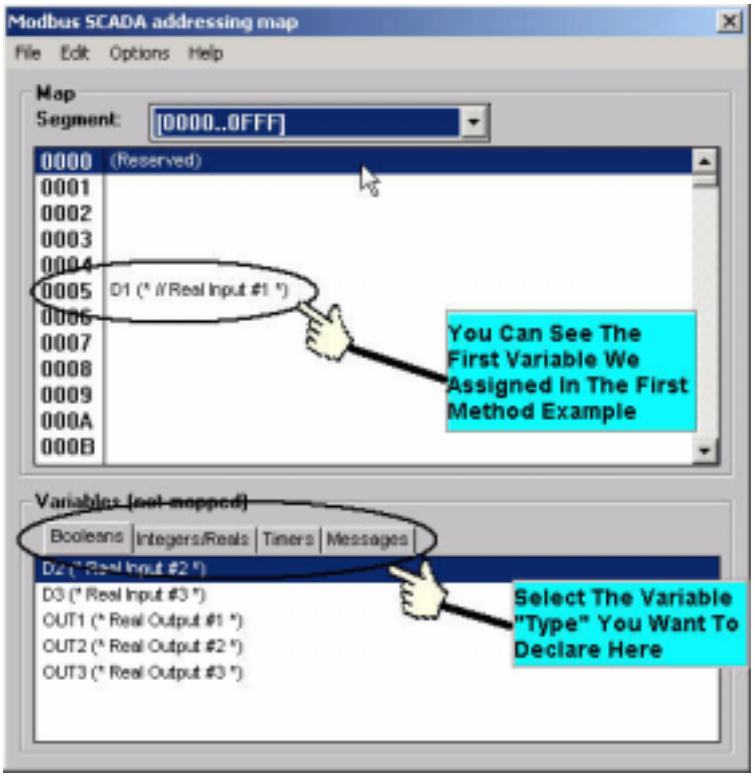


The second method for assigning network addresses to variables requires that you declare the variables BEFORE you assign them. This method allows you to assign numerous network address variables before you link them to an ISaGRAF program.

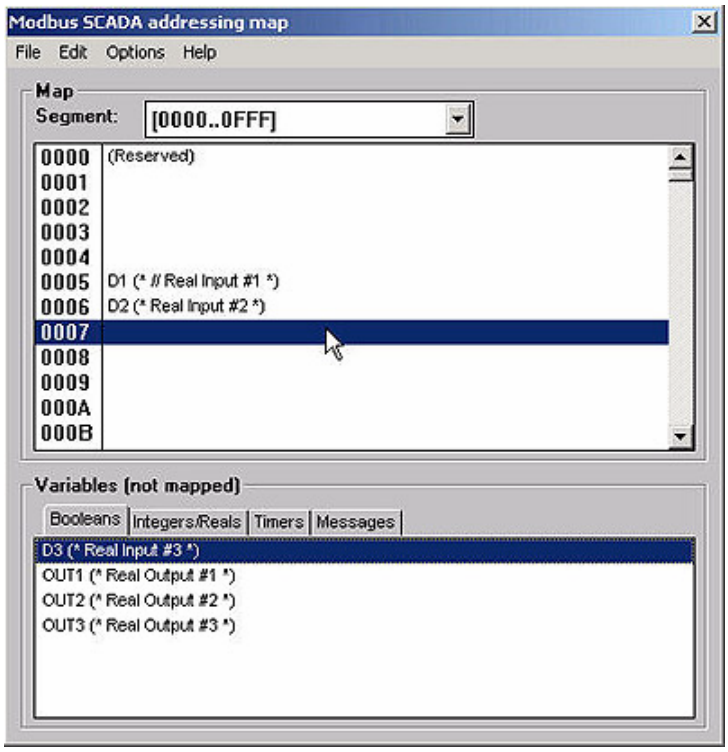


When you click on "Modbus SCADA Addressing Map" (SCADA is an industrial process control acronym that stands for "Supervisory Control And Data Acquisition") the "Modbus SCADA Addressing Map" window will open.

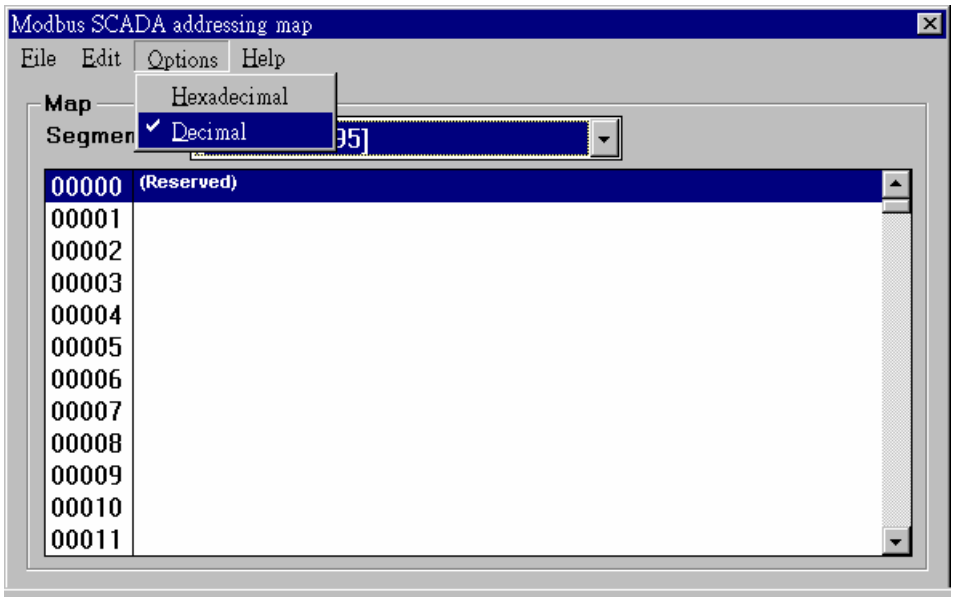
Note that one of the variables (D1) is already assigned from our previous network-addressing example. You will note that the other variables that are not yet mapped are displayed in the lower portion under the "Variables (Not Mapped)" portion of the "Modbus SCADA Addressing Map" window.



To assign the other variable address click on an unassigned "Map Segment" number, and then double click on the variable you want to assign to the address and the variable will automatically assign itself to the "Map Segment".



For human's thinking method, network address represented in hexadecimal format is inconvenient and it increases the chance to make mistake. Therefore, it's better to change it to be represented in decimal format. To do that is as below.



IMPORTANT NOTE REGARDING MODBUS NETWORK ADDRESSING

The Modbus network address definition scheme is sometimes different between HMI devices and other software programs. The difference is typically that the other programs may assign a network address number that is one (1) less than that of the I-7188XG / I-7188EG controller system.

Known addressing disparities include "LabLink" and "Hitech" HMI software programs and devices. If you are assigning a network address of "B" (hexadecimal) of these products the I-7188XG / I-7188EG network address should be set to "C". A network address of "2" should be associated with a network address of "3" in the I-7188XG / I-7188EG controller system.

HMI or devices such as Iconics, Citech, Wizcon, Kepware's OPC server, Intellution's "iFix", Wonderware's "Intouch", National Instruments "Labview", and ICP DAS's Touch 200, Touch 250, Touch 506, Touch 509 and Touch 510 do have the exact same addressing scheme as the I-7188XG / I-7188EG controller system.

ICP DAS has not been able to test every possible HMI software program or hardware device that has Modbus addressing capability. If you are trying to connect your HMI software program or hardware device with Modbus to an I-7188XG / I-7188EG controller system, **REMEMBER** that you may have to offset the Modus addressing by 1 between these products so they will properly communicate with each other.

Developers who design and write their own software interface programs using Microsoft's Visual Basic or Visual C++ programming language should refer to Chapter 5 of this manual for more information on how to interface the Modbus protocol to these programming languages.

NOTE:

While talking to the **I-7188XG**, **ONE** Modbus frame cannot request more than **255 bits**, and also cannot request more than **125 words**. It should be divided into 2 or more requests to achieve it. For **I-7188EG**, can not request more than **255 bits** and **122 words** in one modbus frame.

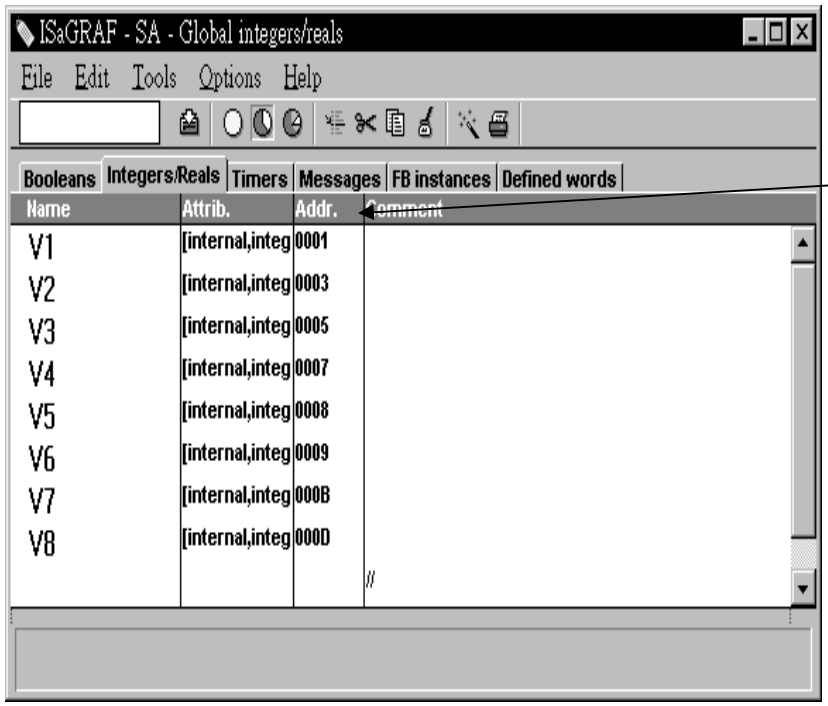
4.2:Read/Write Word, Long Word & Float through Modbus

Modbus protocol provides function 3 for reading multiple words while function 6 and 16 to write words. Please refer to Chapter 5 for more information about the protocol.

The word defined in the Modbus protocol of I-7188XG / I-7188EG controllers is like a signed short integer, which occupies 2 bytes and range from –32,768 (8000 in hexa.) to +32,767 (7FFF in hexa.). It is normally used to describe the behavior of analog I/O channels. For examples, the X304 I/O expansion board (please refer to section 1.7)

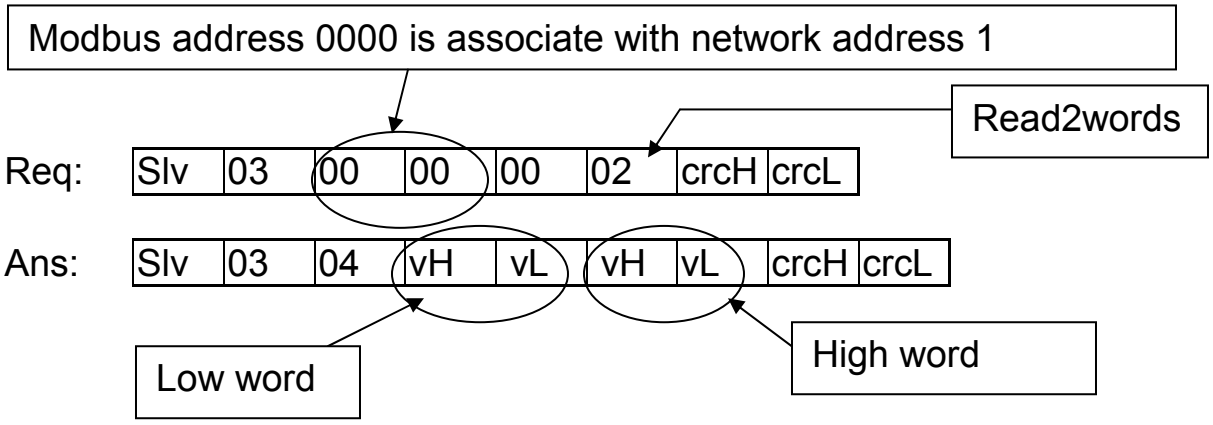
The long word defined in the Modbus protocol of the I-7188XG / I-7188EG controller is like a signed long integer, which occupies 4 bytes and range from -2,147,483,648 (8000 0000 in hexa.) to +2,147,483,647 (7FFF FFFF in hexa.). It is normally used to describe the value of internal integer variables declared on ISaGRAF workbench.

All integer variables declared on ISaGRAF are signed 32-bit format however the integer variable, which assigned with a network address will only, occupies 1 word (2 bytes) in the Modbus transportation format. Since a long word occupies 2 words (4 bytes), to Read/Write long word through Modbus, the network address assigned to the integer variable has to be followed as below.

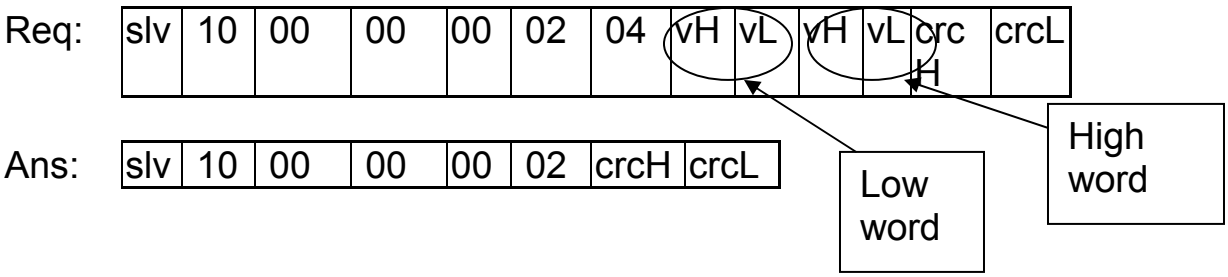


V1 is assigned to a network address “1”. If the network address “2” is not assigned to any other variable, V1 will occupy a long word (4 bytes) in the Modbus transportation formate.
However if “2” is assigned to one another variable, V1 will only occupy one word (2 bytes) in the Modbus transportation formate.
In this example, V1, V2, V3, V6, V7 and V8 will occupy 4 bytes however V4 and V5 only occupy 1 word (Lowest word) in the Modbus transportation formate.

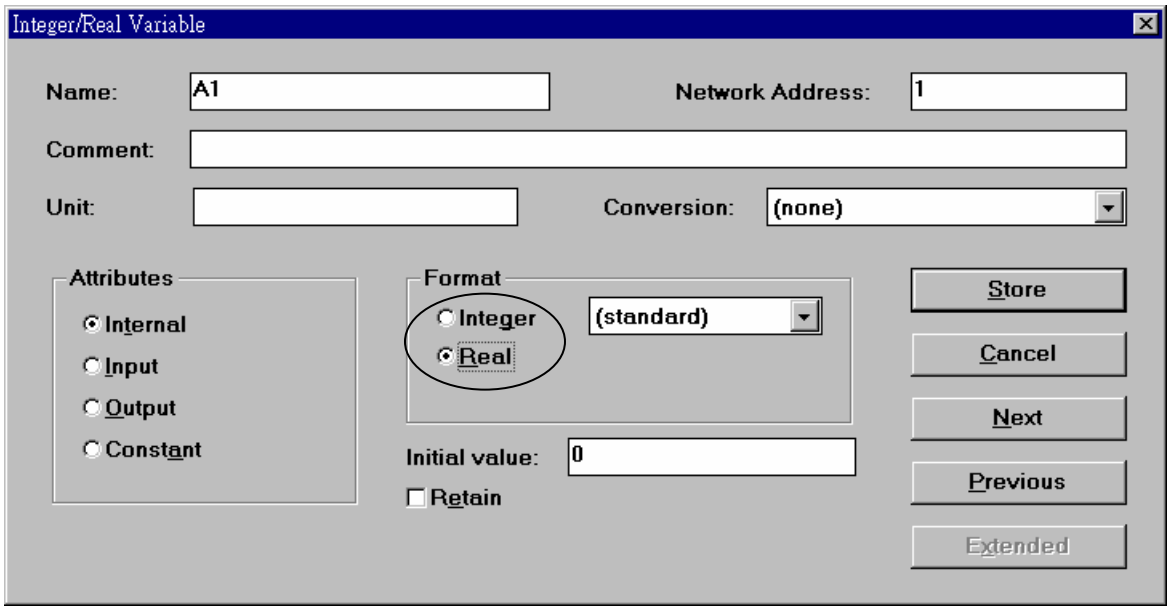
To read long word value of V1 is to read 2 words by using modbus function 3 (please refer to section 5.1).



To write long word to V1 is to write 2 words by using modbus function 16.



To read / write float (4 bytes) is very similar to read / write long word. The difference is the variable should be declared as “Real” type, and the next network address No. should not be assigned to any other variable.



Chapter 5: Modbus Protocol

The Modbus protocol is a powerful and flexible communications protocol that allows numerous software programs and hardware devices to communicate with each other. Any I-7188XG / I-7188EG variable that will be used to communicate through the Modbus protocol **MUST** have a unique network address before it can communicate through a Modbus link (please refer to section 4.1).

Please refer to Chapter 5 of “User’s Manual Of The I-8417/8817/847/8837 ISaGRAF Embedded Controllers”.

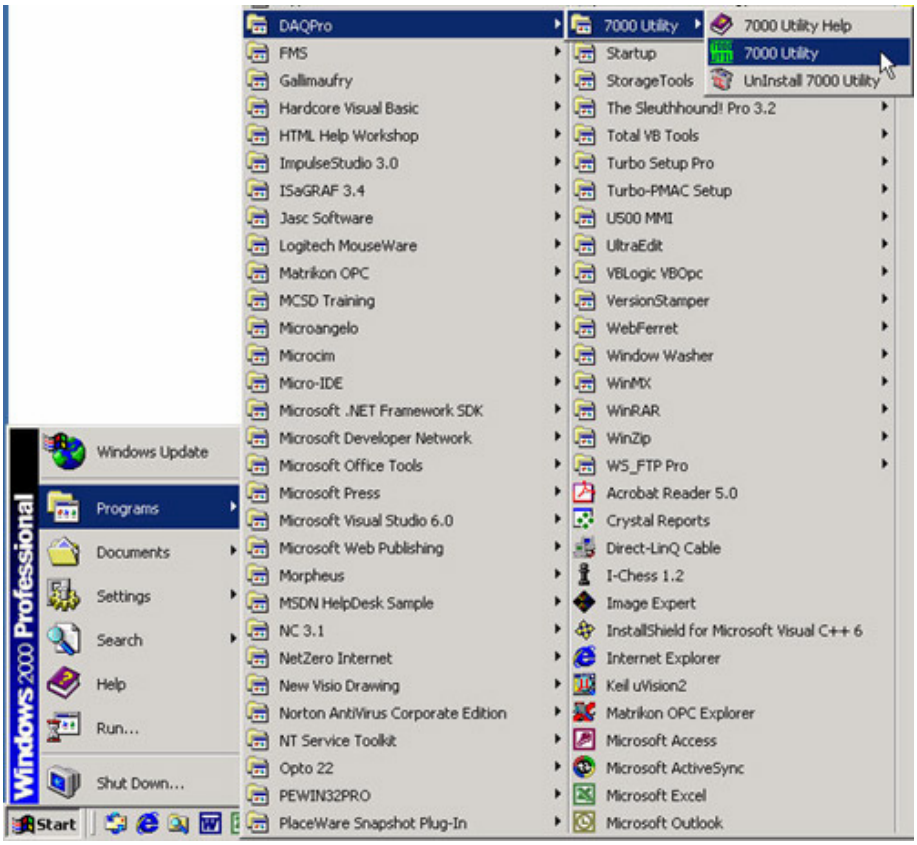
Chapter 6: Linking I-7000 & I-87xx Modules

The I-7188XG / I-7188EG controller system provides the capability to integrate with ICP DAS's I-7000 and I-87xx (87K4 / 87K5 / 87K8 / 87K9) series modules. This functionality to interface with these modules expands the capability of the I-7188XG / I-7188EG controller series products.

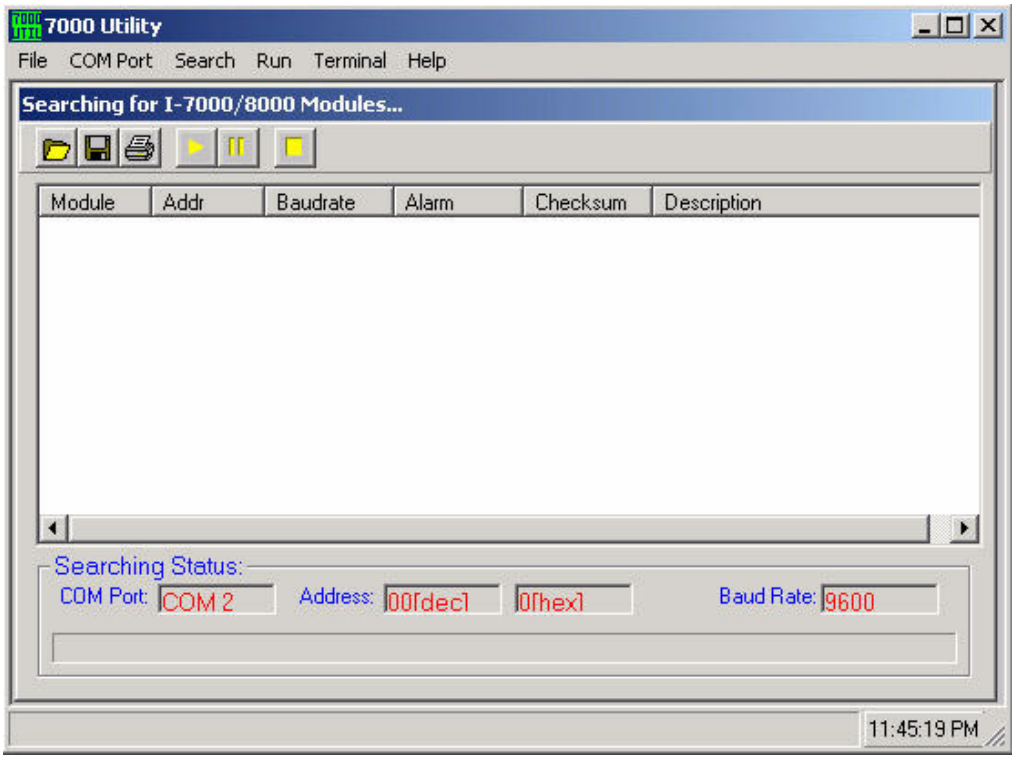
You must first make sure that the I-7188XG / I-7188EG I/O libraries have been installed, please refer to Section 1.2 for the library installation procedure, and refer to Section 1.4 for connection instructions between the I-7188XG / I-7188EG controller system to the I-7000 and I-87xx series modules.

6.1: Configuring The I-7000 & I-87xx Modules

To begin configuration of the I-7000 and I-87xx series modules to the I-7188XG / I-7188EG controller system, use the "7000 Utility" program to set up the I-7000 and I-87xx modules.



Once you have selected the "7000 Utility" program, the "7000 Utility" window will open.



The "7000 Utility" will attempt to link to any I-7000 and I-87xx modules.

IMPORTANT NOTES Regarding I-7000 & I-87xx Modules

One I-7188XG / I-7188EG controller can link up to a maximum of 64 I-7000 and I-87xx modules. It is recommended though that you do not link more than 24 modules to a single I-7188XG / I-7188EG controller system. **Each I-7000 and I-87xx module MUST have it's own unique address to properly link to an I-7188XG / I-7188EG controller system. Make sure to set the "Checksum" to disabled, and make sure that all of the I-7000 and I-87xx modules are set to the same baud rate as the I-7188XG / I-7188EG controller system**.

If you need assistance on changing the baud rate or checksum of I-7000 & I-87K modules, please refer to the "Change Baud Rate & Checksum" section in the "Getting Started With I-7000 Series Modules". You can find all of the documentation on the CD provided with your I-7000 series module from ICP DAS in a file titled "getstart.pdf".

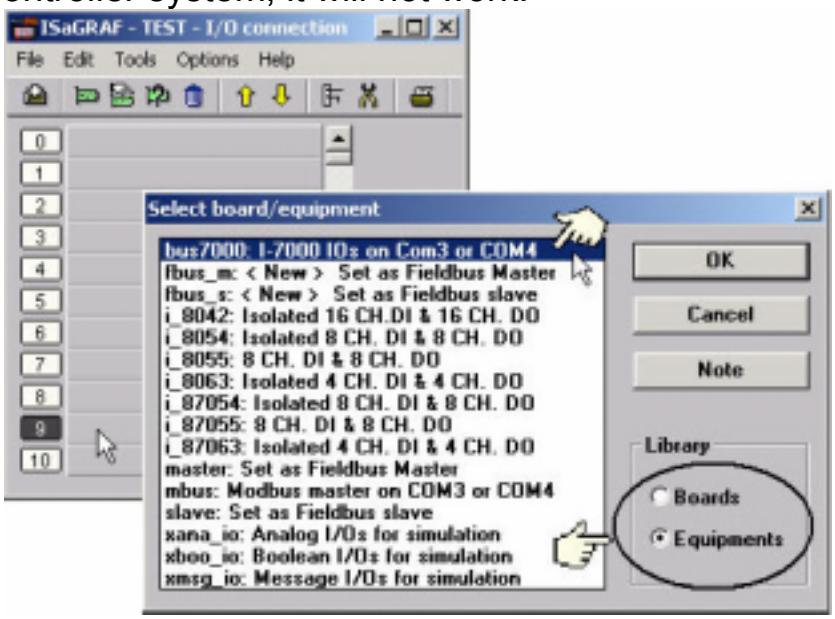
The I-7000 and I-87xx "**Analog Input**" type modules MUST have their data format set to "**2's Complement**". This includes the I-7013, I-7017, I-7018, I-7033, I-87017, and I-87018 analog input modules.

The I-7000 and I-87xx "**Analog Output**" type modules MUST have their data format set to "**Engineer Unit**". This includes the I-7021, I-7022, I-7024 and I-87024 analog output modules.

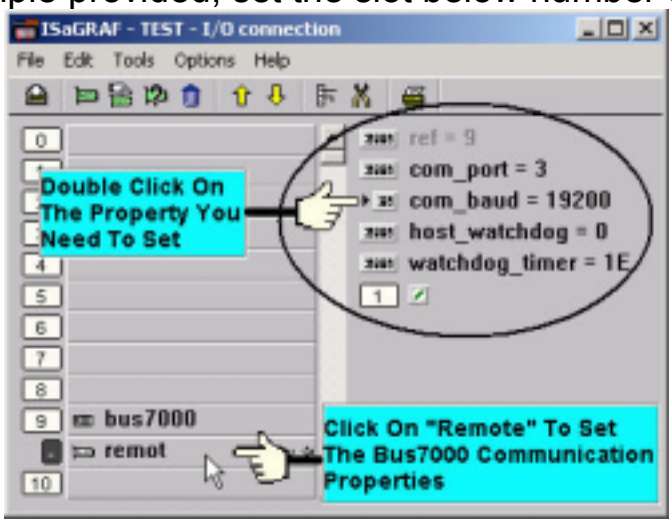
6.2: Opening The "Bus7000" Function

To create a link between the I-7188XG / I-7188EG controller system and an I-7000 and I-87xx module, you need to connect the "Bus7000" function through the "ISaGRAF I/O Connection" window. The "Bus7000" function is considered a "virtual board", and must be selected from the "Equipments" section of the "Select Board/Equipment" window.

The "Bus7000" MUST be connected to slot number 1 or higher on the "ISaGRAF I/O Connection" window (since slot 0 is used to connect to I/O expansion boards). **Only one "Bus7000" can be linked to one I-7188XG / I-7188EG controller system!** If you attempt to connect more than one "Bus7000" to an I-7188XG / I-7188EG controller system, it will not work.



In the example provided, set the slot below number 9 to "Bus7000: Remote".



Don’t need to care the "com_port" parameter. Whatever you set, the I-7188XG / I-7188EG controller system always communicate with the I-7000 / I-87xx module through Com2.

The "com_baud" parameter defines the baud rate that the I-7188XG / I-7188EG will communicate with the I-7000 / I-87xx module. The possible values are 2400, 4800, 9600, 19200, 38400, 57600, and 115200. You must make sure that the I-7188XG / I-7188EG controller system and the I-7000 / I-87xx modules are all set to the same "com_baud" value.

The "host_watchdog" parameter enables or disables the watchdog function for the I-7000 and I-87xx module. Setting the "host_watchdog" parameter to a non-zero value will enable the "host_watchdog" feature.

The "watchdog_timer" parameter defines the amount of time before a "host_watchdog" will occur. The value for the "watchdog_timer" is defined in a **hexadecimal** value with the units defined in 0.1-second increments. For example, if the "watchdog_timer" is set to a value of 1E, the "watchdog_timer" is set for 3 seconds. If the "watchdog_timer" value is set to 2A, the "watchdog_timer" is set for 4.2 seconds.

If the host watchdog feature is active and the watchdog timer is exceeded on the I-7188XG / I-7188EG controller system (it means the connection is break between the I-7188XG / I-7188EG controller and I-7000 / I-87xx modules), the I-7000 / I-87xx modules will go to a "safe" predetermined value.

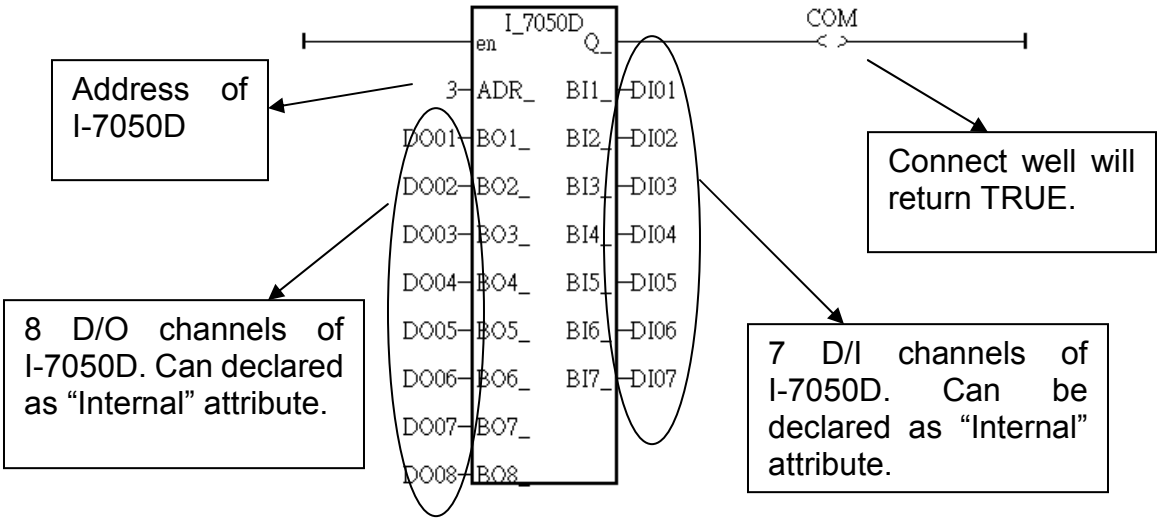
There is an analog input channel available on the "Bus7000: Remote" virtual board. This analog input channel will return a value equal to the currently set baud rate.

6.3: Programming an I-7000 Module

To link any I-7000 and I-87xx module to the I-7188XG / I-7188EG controller system, the "Bus7000" module MUST be opened first. Once the "Bus7000" is opened, the "I_7xxx" / "I-87xx" function block can now be programmed and you can access all of the I/O channels available from that function block, and that data can now be used in a LD program.

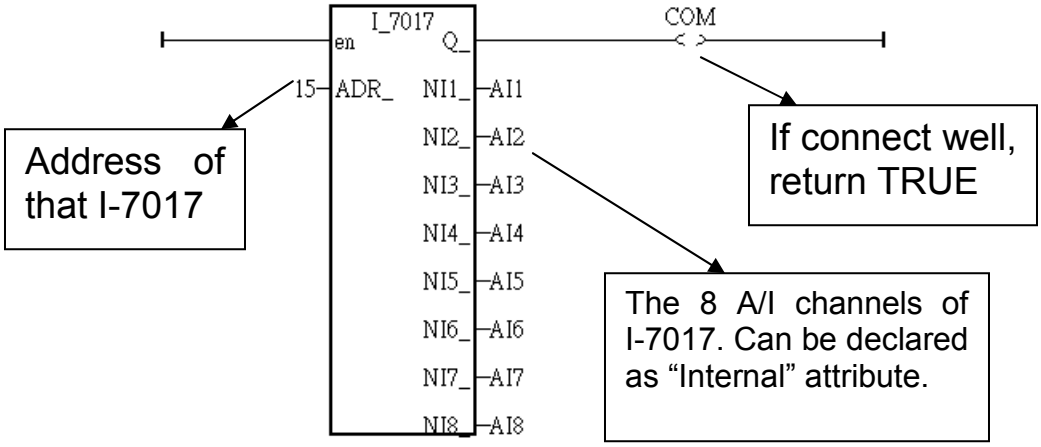
NOTE:
You can declare all variables which connect to the I-7xxx / I-87xx function block as "Internal" attribution.

Example 1: Programming An I-7050D Module

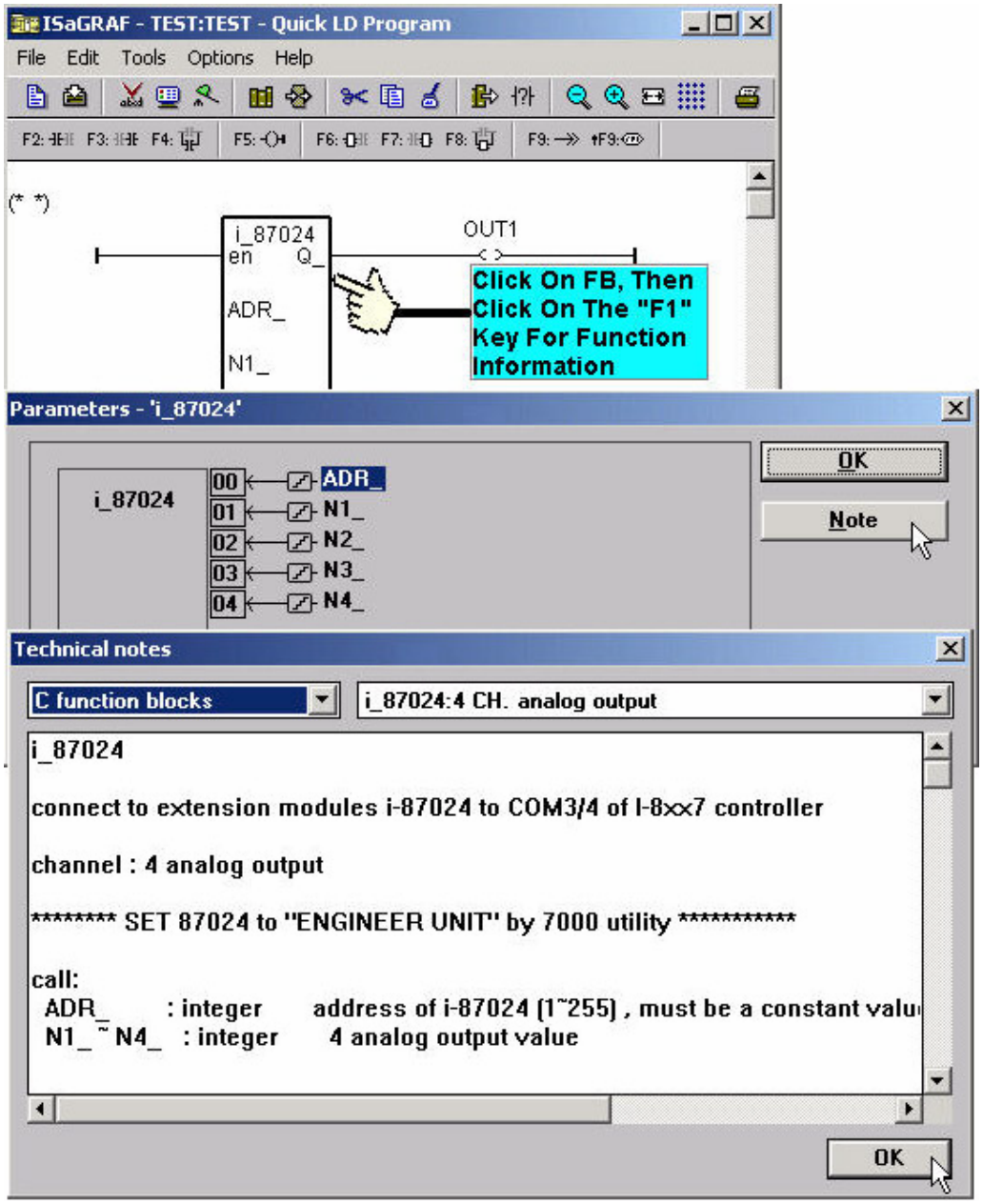


Example 3: Programming An I-7017 Module

The Data Format Used must be: 2's Complement



For additional information regarding any I-7000 and I-87xx module, click on the function block and press the "F1" key for an on-line description with "Technical Notes" for the selected function block.



Chapter 7: Controller To Controller Data Exchange

The I-7188EG, I-8437 & I-8837 controller support Ebus. Ebus is a software mechanism which allows controllers to access data to each other through the ethernet port.

This section will be available in the future. The file name will be "Ebus_I_8xx7.pdf" and will be found at
ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/8000/english_manu/

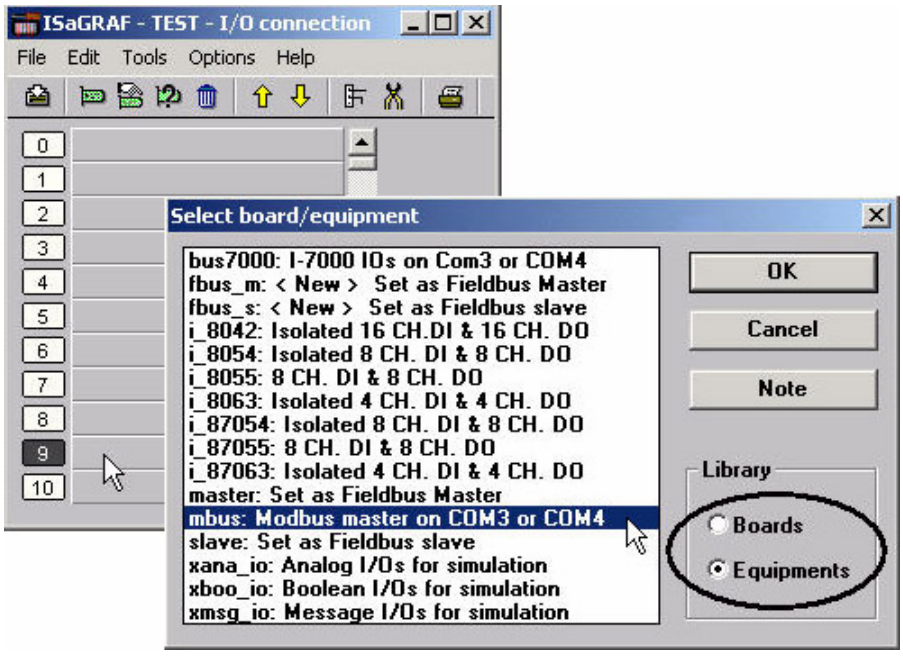
Chapter 8: Linking Modbus RTU & Other Devices

The I-7188XG / I-7188EG can interface with the Modbus RTU Serial or other Modbus devices. You must first make sure that the I-7188XG / I-7188EG I/O Libraries have been installed. Please refer to Section 1.2 for the library file installation instructions and Section 1.5 for the connection interface between the I-7188XG / I-7188EG controller system to Modbus RTU and other Modbus devices.

8.1: Configuring As A Modbus Device

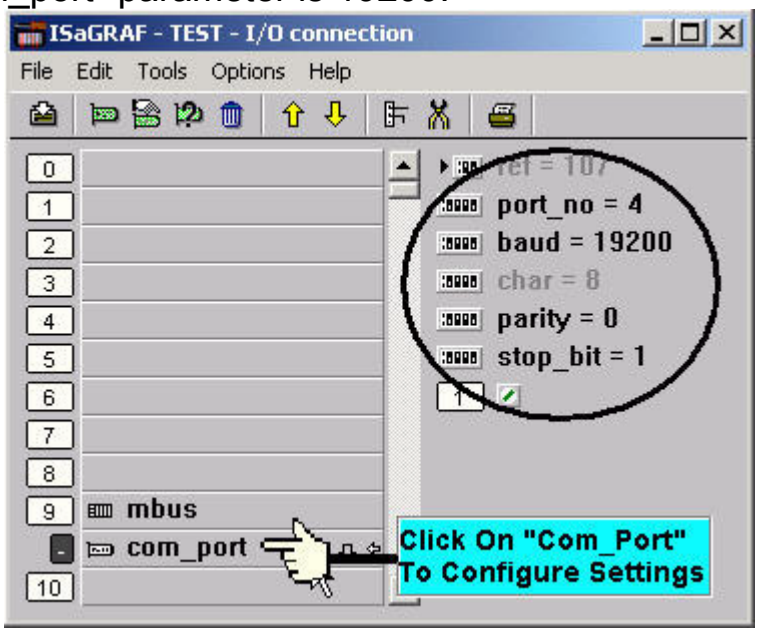
To begin configuring an I-7188XG / I-7188EG controller system to interface with a Modbus device, you must first configure the ISaGRAF program by linking the "Mbus" function to the ISaGRAF project. Open the "ISaGRAF I/O Connections" window and double click on a slot number higher than 0 and the "Select Board/Equipments" window will open. From the "Library", click on the "Equipments" choice, and then click on the "Mbus: Modbus Master On COM2 Or COM3" selection, and then click on the "OK" to complete the installation.

IMPORTANT NOTE:
Only **ONE** "Mbus" complex equipment function can be linked to **ONE** I-7188XG / I-7188EG controller system.



"Mbus: com_port" Parameter
The "Mbus: com_port" parameter sets the same baud rate that the I-7188XG / I-7188EG controller system and all Modbus devices will communicate at. ALL

devices MUST be set to the same baud rate setting. The default setting for the "Mbus: com_port" parameter is 19200.



"Mbus: port_no" Parameter
The "Mbus: port_no" parameter defines which COM port the Modbus devices will communicate with the I-7188XG / I-7188EG controller system. The "Mbus: port_no" parameter can be set to either a value of **"2" (COM2) or "3" (COM3)**.

"Mbus: baud" Parameter
The "Mbus: baud" parameter defines what the communications baud rate setting will be. The "Mbus: baud" can be set to 2400, 4800, 9600, 19200, 38400, 57600 or 115200 baud rate. The default baud rate value is 19200 for the I-7188XG / I-7188EG controller system. All controllers on the same Modbus MUST be set to the same baud rate.

"Mbus: parity" Parameter
The "Mbus: parity" parameter defines what the communications parity setting will be. Setting the "Mbus: parity" parameter to a value of "0" sets the parity to "none", a value of "1" sets the parity to "even", and a value of "2" sets the parity to odd.

"Mbus: stop_bit" Parameter
The "Mbus: stop_bit" parameter defines the number of stop bits will be used in the Modbus communications. If the "Mbus: stop_bit" parameter is set to "1", this equals 1 stop bit, and a value of "2" equals 2 stop bits.

8.2: Programming A Modbus Device

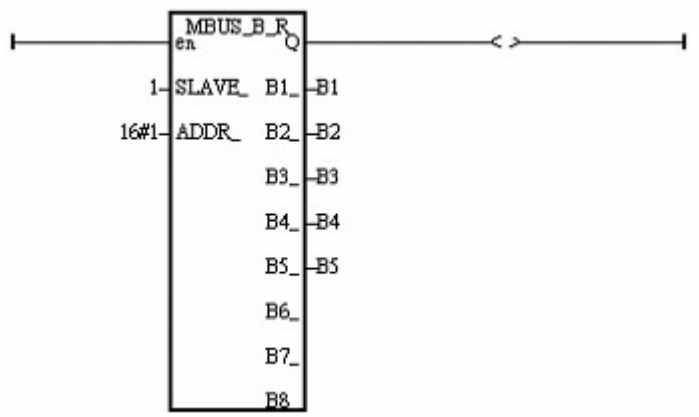
To access data from a Modbus device you must first make sure the "Mbus" library function has been installed. If you haven't installed the "Mbus" library, refer to Section 8.1 on how to install the "Mbus" library function. Once the "Mbus" library function has been installed, you can program to pass data through the Modbus protocol between I-7188XG / I-7188EG controller and Modbus devices.

The following function blocks can be used to pass data through the Modbus protocol in an LD program.

Mbus_b_r Reads 8 bits (booleans) from modbus devices. Mbus_b_w Writes 1 to 4 bits to modbus devices. Mbus_n_r Reads 8 words (short integers) from modbus devices. Mbus_n_w Writes 1 to 4 words to modbus devices.

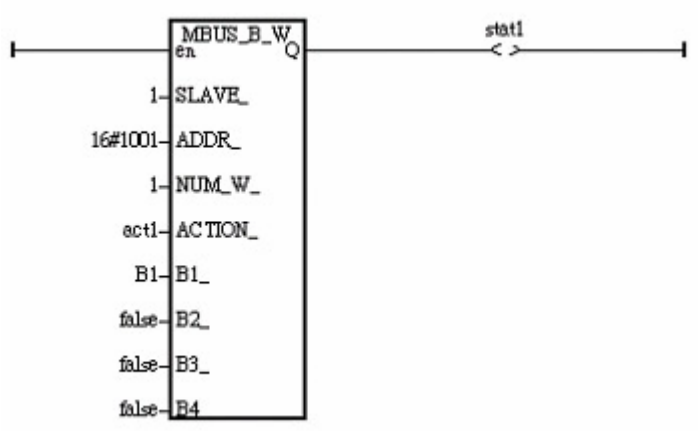
NOTE:
The maximum number of each "Mbus_x_x" function block that can be used with one I-7188XG / I-7188EG controller system is 64.

Modbus Example Function #1: "Mbus_b_r"
The following example the "Mbus_b_r" function block is reading five (5) bits from a slave Modbus device with a NET ID address of 1, with the Modbus address starting from 1. In this example the results of "B1" contains the value of the Modbus address 1, "B2" equals the value of Modbus address 2, etc. "B5" equals the value of the Modbus address 5.

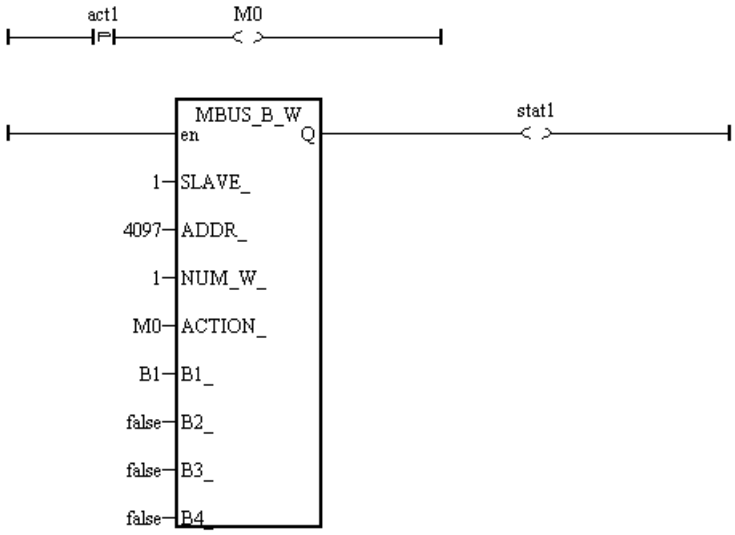


Modbus Example Function #2: "Mbus_b_w"
The following example of the "Mbus_b_w" function block is writing one (1) bit to a slave Modbus device with a NET ID address of 1. The "Mbus_b_w" function will only write this one bit when the "ACTION_" line is true. In the example below the resulting value of "B1" is written to the Modbus address 16#1001 (or 4097) of that Modbus device when the "ACTION_" line is true.

The value of "Stat1" is connected to the output coil and if the operation is successful "Stat1" will be true, otherwise the value of "Stat1" will be false.

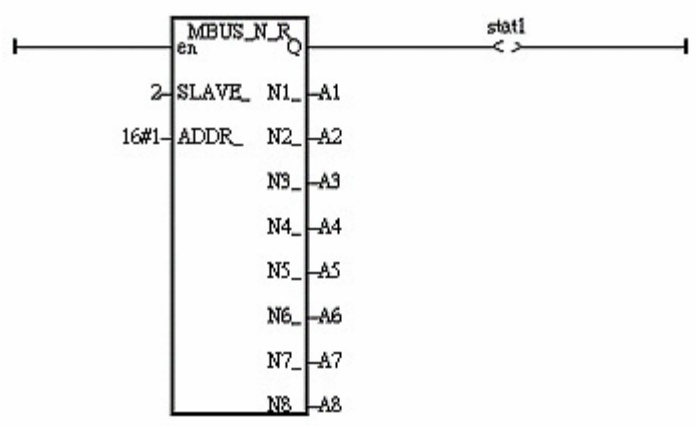


If the “ACTION_” input keeps at the status of TRUE, it will continue to write this "B1" many times to that Modbus device until it is reset to FALSE. If you just want to write one time, you can write a LD program similar as the following. The M0 is declared as an internal Boolean variable.



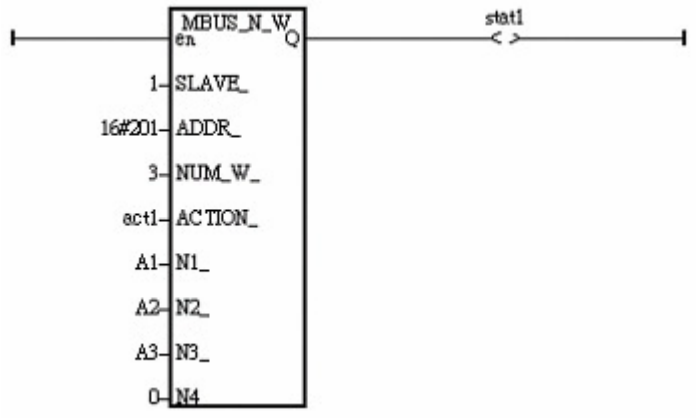
Modbus Example Function #3: "Mbus_n_r"
The following example the "Mbus_n_r" function block is reading eight (8) words from a slave Modbus device with a NET ID address of 2 (the Modbus addressing starts from 1). In this example the results of "A1" contains the value of the Modbus address 1, "A2" equals the value of Modbus address 2, etc., through "A8" which equals the value of the Modbus address 8.

The value of "Stat1" is connected to the output coil and if the operation is successful "Stat1" will be true, otherwise the value of "Stat1" will be false.

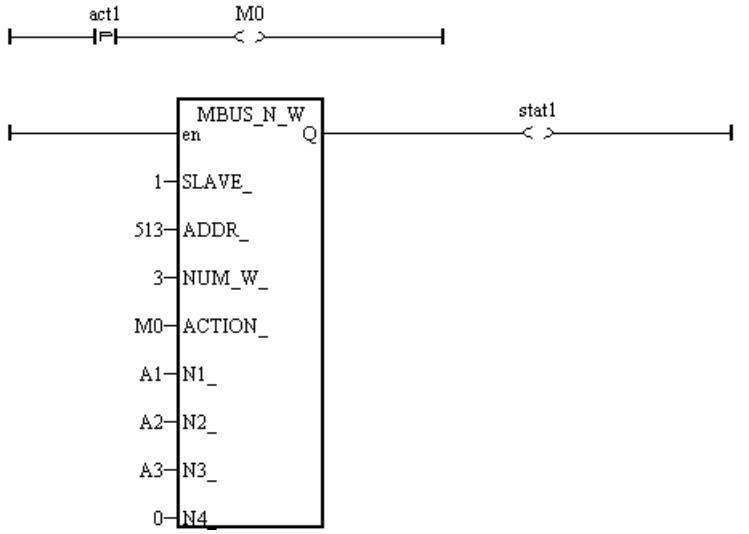


Modbus Example Function #4: "Mbus_n_w"
The following example of the "Mbus_n_w" function block is writing three (3) words to a slave Modbus device with a NET ID address of 1, and the Modbus address is starting from 16#201. The "Mbus_n_w" function will only write when the "ACTION_" line is true. In this example when the "ACT1" line is True, the value of A1 will be written to the value of Modbus address 16#201 of that Modbus device, the value of A2 will be written to the value of Modbus address 16#202, and A3 will be written to the value of Modbus address 16#203.

The value of "Stat1" is connected to the output coil and if the operation is successful "Stat1" will be true, otherwise the value of "Stat1" will be false.



If the “ACTION_” input keeps at the status of TRUE, it will continue to write these "A1" through "A3" many times to that Modbus device until it is reset to FALSE. If you just want to write one time, you can write a LD program similar as the following. The M0 is declared as an internal Boolean variable.



Chapter 9: Commonly Used ISaGRAF Utilities

This section details some useful ISaGRAF utilities.

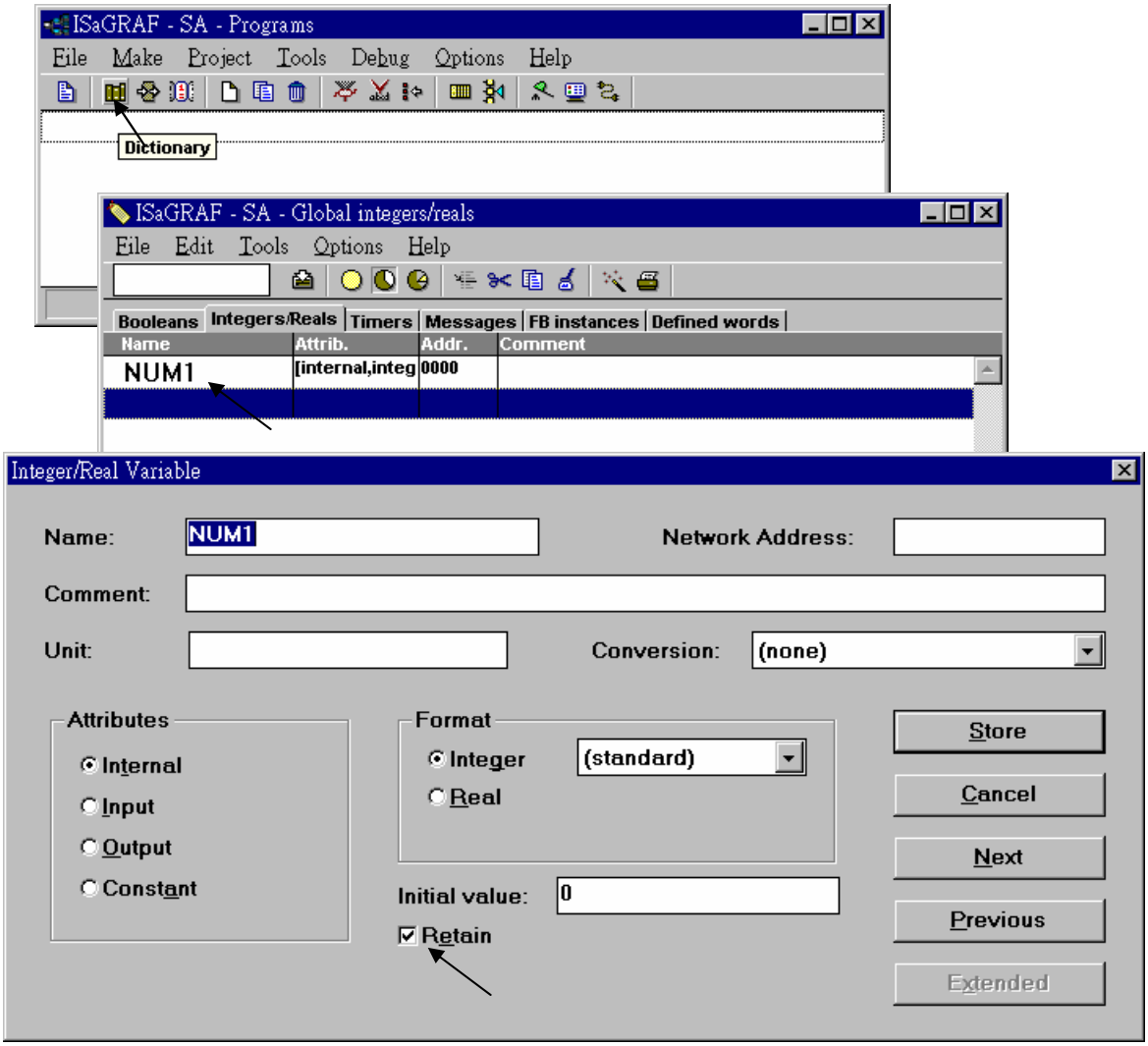
Please refer to Chapter 9 of “Use’s Manual Of The I-8417/8817/8437/8837 ISaGRAF Embedded Controllers”

Chapter 10: The Retained Variable And Data Backup

10.1: The Retained Variable

For some real applications, data has to be retained when the power is dead, and these data should be restored to its last value when the power is coming up again. I-7188XG / I-7188EG controllers provide battery backup memories to fit such kind of applications. The battery used can provide the energy to keep the retained variables alive last for some years. It also can provide the energy for the Real-Time-Clock.

A maxinum of **six integers** (signed 32-bit) and **sixteen Booleans** can be retained. To enable the retained function, click on “Retain” for each associated variable.



10.2: Data Backup To The EEPROM

Data can be stored into the EEPROM. The value will be always hold even the power is dead unless the value is updated. The EEPROM of I-7188XG / I-7188EG controller can be read freely however can be written only about to 100,000 times.

To read a value from the EEPROM, the following functions can be used.

EEP_B_R	Reads one boolean
EEP_BY_R	Reads one byte
EEP_WD_R	Reads one word (2 bytes, signed)
EEP_N_R	Reads one integer (4 bytes, signed)

To write a value to the EEPROM, should remove the protection of the EEPROM first and then write operation is possible. The following functions can be used.

EEP_EN	Removes the protection of EEPROM
EEP_PR	Set the protection of EEPROM
EEP_B_W	Writes a boolean, up to 256 booleans can be stored.
EEP_BY_W	Writes one byte, up to 1,512 bytes can be stored.
EEP_WD_W	Writes one word (2 bytes, signed), up to 756 words can be stored.
EEP_N_W	Writes one integer (4 bytes, signed),up to 378 integers can be stored.

Bytes, words and integers will be stored to the same memory area in the EEPROM. Be careful to arrange their address before using the above write functions. There are total 1,512 bytes in this EEPROM memory area. The addressing No. of bytes is range from 1 to 1,512, while words is 1 to 756, and integers is 1 to 378. The following No. will use the same memory address in the EEPROM.

Byte	4n-3, 4n-2, 4n-1, 4n	(* n = 1, 2, …378 *)
Word	2n-1, 2n	
Integer	n	

When using the write functions, the EEPROM will be damaged if the write operation is more than 100,000 times. For example, the following program is dangerous since the EEPROM will be written once per cycle (normally, the cycle is about 1 to 40 ms depends on the application) .

```
(* ST program, Val is declared as an integer, TEMP is declared as a boolean *)
TEMP := eep_n_w(1, Val);    (* dangerous *)
```

However the following program is safe if Val is not changed frequently.

```
(* ST program, Val, Old_Val declared as integers, TEMP declared as a
boolean *)
IF  Val <> Old_Val  THEN
       TEMP := eep_n_w(1, Val);
     Old_Val := Val;
END_IF;
```

Each read / write operation on the EEPROM will consume a lot of CPU time of I-7188XG / I-7188EG controller system. The following approximate time is for each function being called.

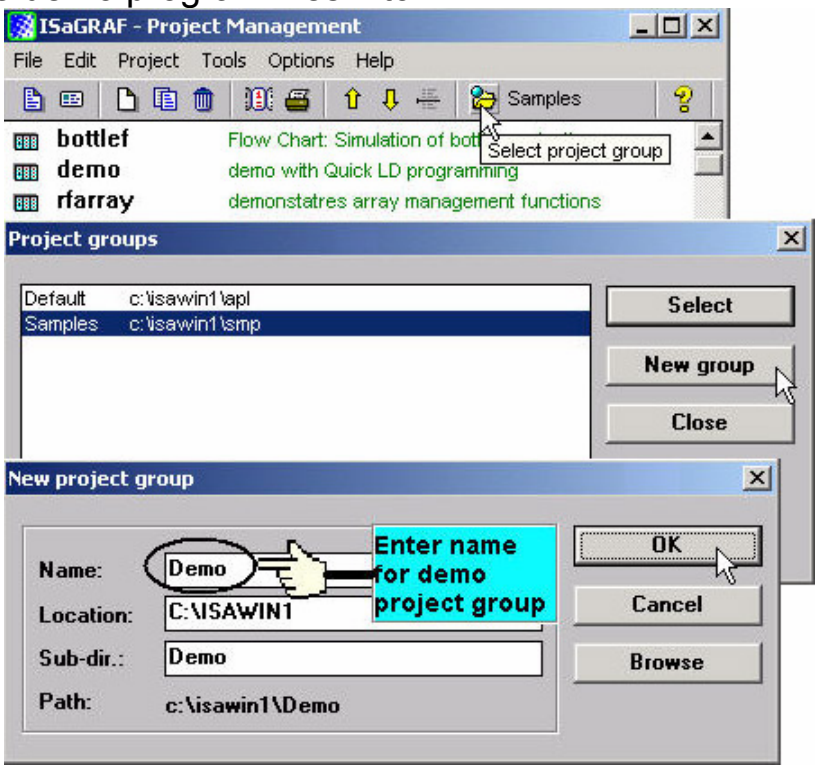
EEP_EN	~ 0.08 ms	**EEP_PR**	~ 0.08 ms
EEP_B_R	~ 0.8 ms	**EEP_B_W**	~ 6 ms
EEP_BY_R	~ 0.8 ms	**EEP_BY_W**	~ 6 ms
EEP_WD_R	~ 1.5 ms	**EEP_WD_W**	~ 12 ms
EEP_N_R	~ 2.9 ms	**EEP_N_W**	~ 23 ms

Recommend to read values from the EEPROM at one time when the I-7188XG / I-7188EG is powered up, and then updated the associated address in the EEPROM when the value is changed.

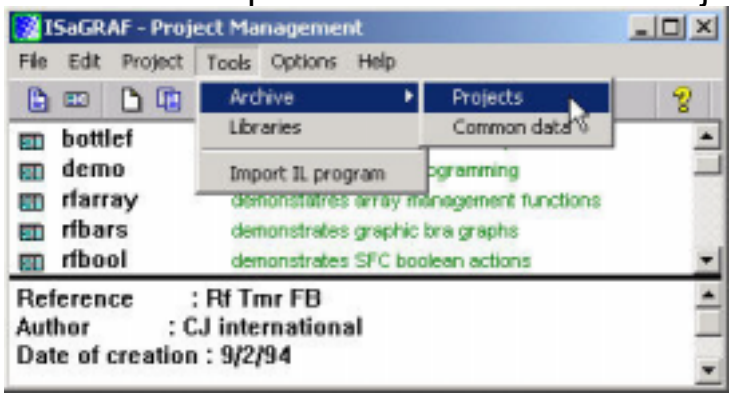
Chapter 11: ISaGRAF Programming Examples

The ISaGRAF programming examples are installed on the same CD-ROM as the I-7188X / I-7188EG I/O library that you receive with the controller. You will find the programming example files in the "**\Napdos\ISaGRAF\7188XG\Demo**" or "**\Napdos\ISaGRAF\7188EG\Demo**" sub-directory on the CD-ROM.

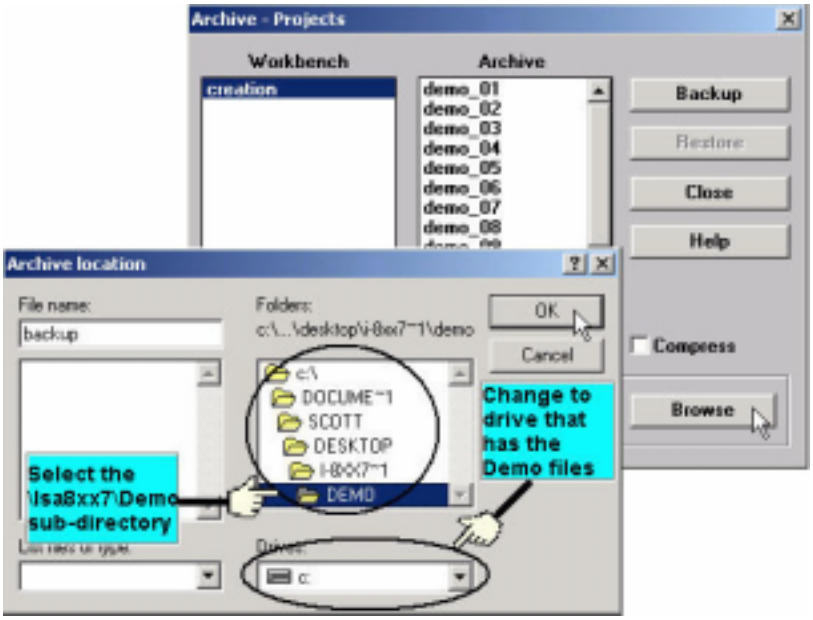
When you install the ISaGRAF programming example for the I-7188X / I-7188EG controller system it is recommended that you create an "ISaGRAF Project Group" to install the demo program files into.



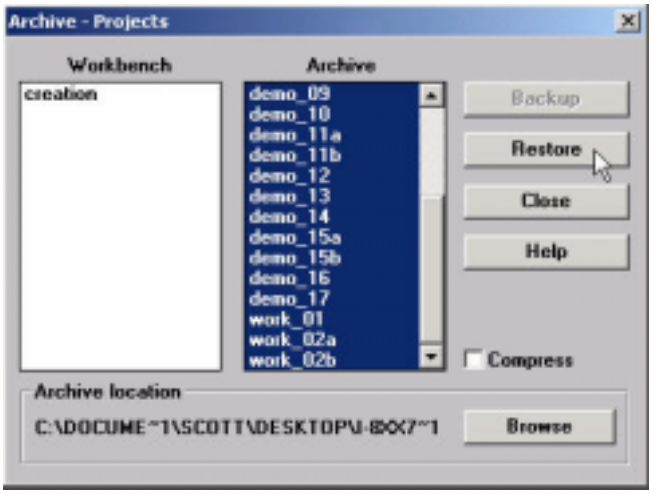
To install the demo programs into the project you have created open the "ISaGRAF Project Management" window to select "Tools" from the menu bar, then select the "Archive" option and then click on "Projects".



When you click on the "Projects" selection the "Archive Projects" window will open. Click on the "Browse" button to select the drive and the sub-directory where the demo files are located (**\Napdos\ISaGRAF\7188XG\Demo\ on the CD-ROM**) .



To install all of the Demo files, click on the "demo01" file, then press and hold down the "Shift" key, continue to hold down the "Shift" key and use your mouse to scroll down to last file in the "Archive" window. Click on the last file name from the demo file location and that will select the entire group of demo files. Lastly, click on the "Restore" button in the "Archive Projects" window and all of the demo files will be installed into the sub-directory you have created.



Appendix A: Function & Function Blocks For The I-7188XG / I-7188EG

Appendix A.1: Standard ISaGRAF Function Blocks

The following details the standard ISaGRAF function blocks that that can be programmed with the I-7188XG / I-7188EG controller system however labeled with "*" is not supported.

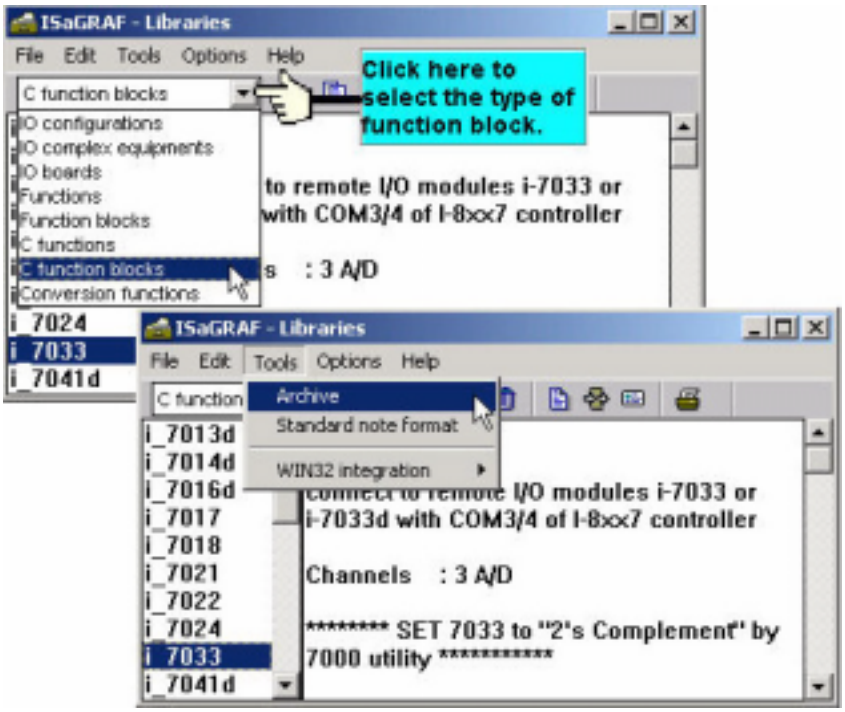
-	*ARWRITE	*F_ROPEN	MSG	SHR
& (AND)	ASCII	F_TRIG	MUX4	SIG_GEN
*	ASIN	*F_WOPEN	MUX8	SIN
/	ATAN	*FA_READ	Neg	SQRT
+	AVERAGE	*FA_WRITE	NOT_MASK	SR
<	BLINK	FIND	ODD	STACKINT
<=	BOO	*FM_READ	*OPERATE	*SYSTEM
<>	CAT	*FM_WRITE	OR_MASK	TAN
=	CHAR	HYSTER	POW	TMR
=1 (XOR)	CMP	INSERT	R_TRIG	TOF
>	COS	INTEGRAL	RAND	TON
>=	CTD	LEFT	REAL	TP
>=1 (OR)	CTU	LIM_ALRM	REPLACE	TRUNC
1 gain	CTUD	LIMIT	RIGHT	XOR_MASK
ABS	*DAY_TIME	LOG	ROL	
ACOS	DELETE	MAX	ROR	
ANA	DERIVATE	MID	RS	
AND_MASK	EXPT	MIN	SEL	
*ARCREATE	*F_CLOSE	MLEN	SEMA	
*ARREAD	*F_EOF	MOD	SHL	

Please refer to the "**ISaGRAF User's Guide**" for more details regarding the "**Standard Operators, Function Blocks & Functions**" available from the ISaGRAF Workbench program.

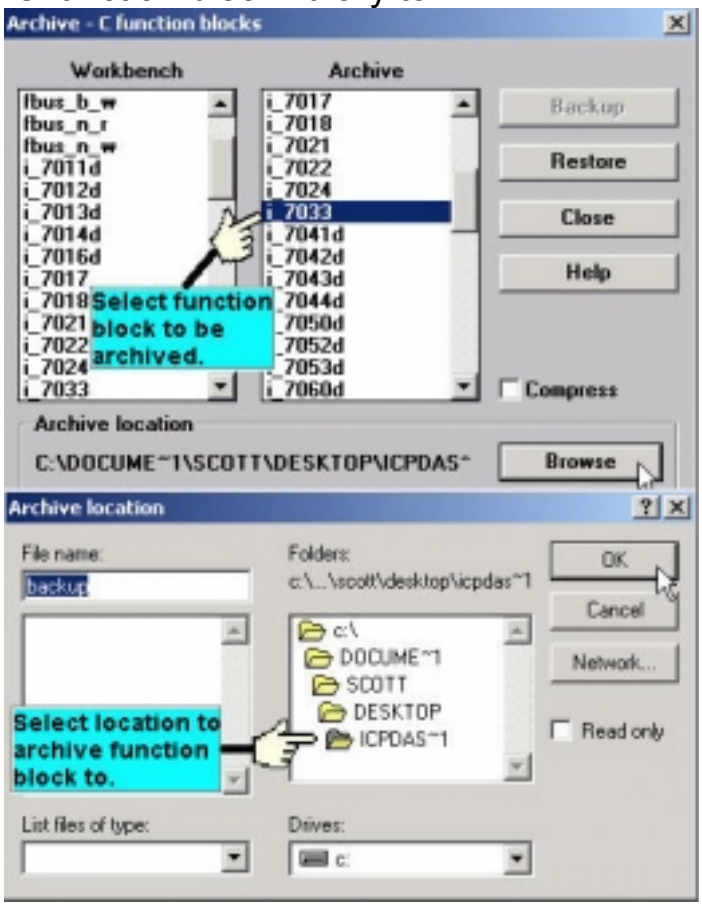
Appendix A.2: Adding New Function Blocks To ISaGRAF

To add or update functions or function blocks for the ISaGRAF Workbench program, click on the Windows "Start" menu, select "Programs", select "ISaGRAF 3.4", then click on "Libraries" to begin installing or updating ISaGRAF functions or function blocks.

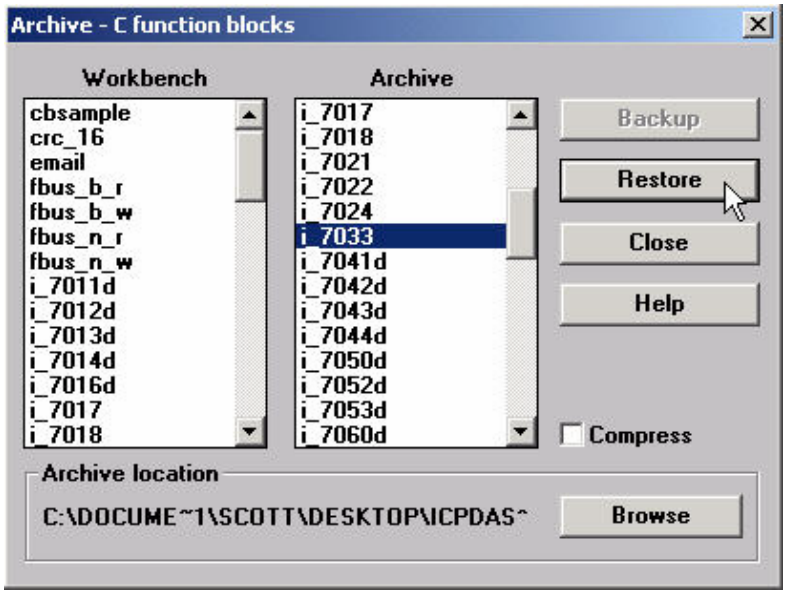
When you click on "Libraries" the "ISaGRAF Libraries" window will open. To add a new function block or function select "Tools" from the menu bar and then click on "Archive".



Click on the file name you want to "Archive" and then click "Browse" button to select the sub-directory to where (CD_ROM\Napdos\ISaGRAF\ARK\) you want to archive the function block library to.

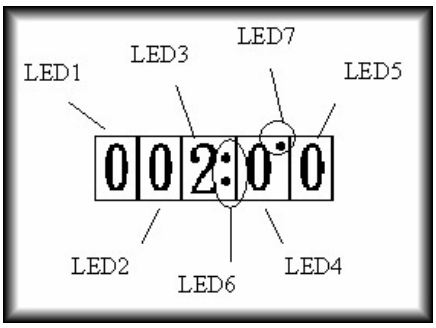


Select the new function block in the "Archive" window that you want to add, and then click on the "Restore" button. When you click on the "Restore" button the function block will be added to the ISaGRAF Workbench window.



Appendix A.3: 7-Segment LED Reference Table

The following table provides the reference definitions for programming the 7 LED indicators on the I-7188XG / I-7188EG controller system.



LED 6: Set to TRUE to display ":" (colon):
LED 7: Set to TRUE to display "." (period above LED 4)

Display Table: LED 1 Through LED 5

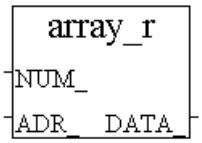
Displayed Char.	Given Value	Displayed Char.	Given Value	Displayed Char.	Given Value
0	0	4.	20	r	40
1	1	5.	21	L	41
2	2	6.	22	n	42
3	3	7.	23	y	43
4	4	8.	24	U	44
5	5	9.	25	P	45
6	6	A.	26	o	46
7	7	b.	27	r.	47
8	8	C.	28	n.	48
9	9	d.	29	y.	49
A	10	E.	30	h.	50
b	11	F.	31	L.	51
C	12	˜	32	U.	52
d	13	¯	33	P.	53
E	14	_	34	o.	54
F	15	¯_	35	_.	55
0.	16	H	36	¯.	56
1.	17	h	37	_.	57
2.	18	H.	38	¯r	Others
3.	19	.	39		

Appendix A.4: Function Blocks For The I-7188XG/7188EG

The following function blocks have been developed specifically for the I-7188XG / I-7188EG controller system.

ARRAY_R

Description:
Function **Read one byte from a byte array**

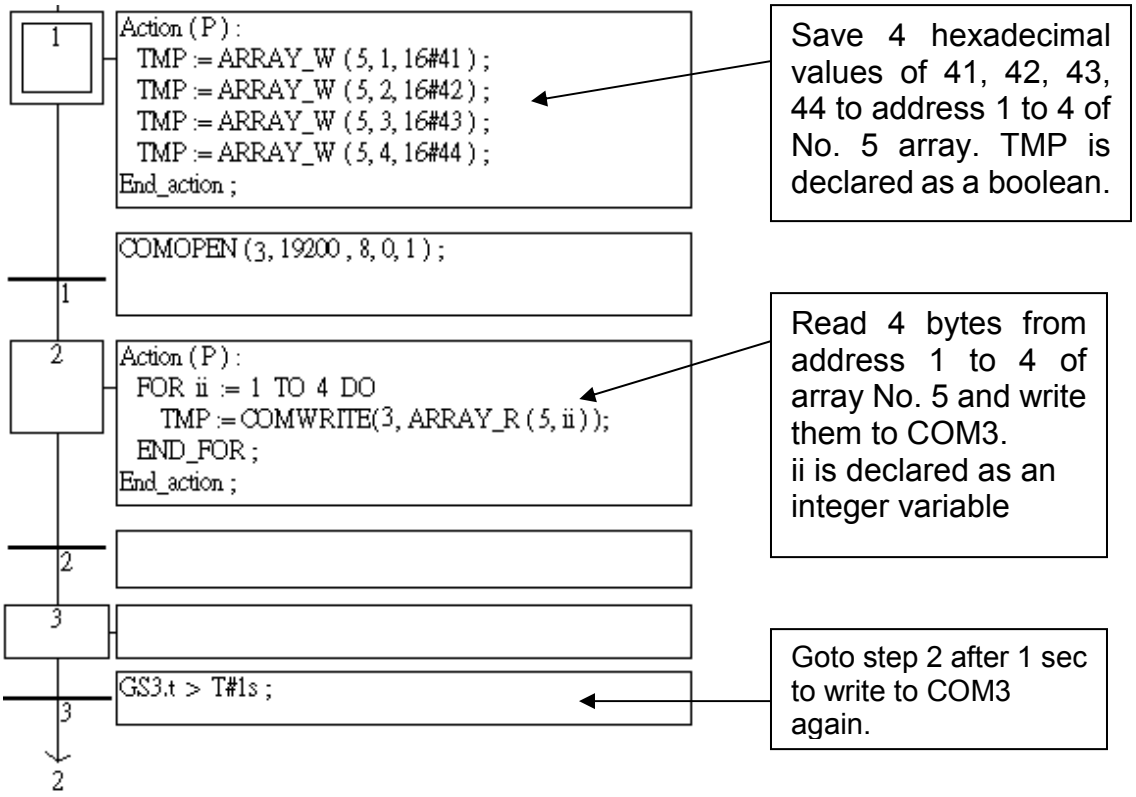


Arguments:

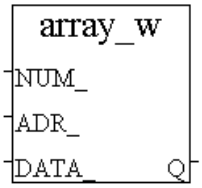
NUM_	integer	array ID to be operated, valid range values from 1 to 24
ADR_	integer	address in the array where the byte is to be stored, valid range values from 1 to 256
DATA_	integer	the byte value returned

* There are 24 byte arrays that can be used.
* Each array can store up to 256 bytes.

Example:



ARRAY_W



Description:
Function **Save one byte to a byte array**

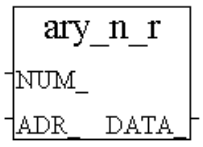
Arguments:

NUM_	integer	array ID to be operated, valid range values from 1 to 24
ADR_	integer	address in the array where the byte is to be stored, valid range values from 1 to 256
DATA_	integer	the byte value to be saved to, valid range values from 0 to 255.
Q_	boolean	if OK. return TRUE, else return FALSE

* There are 24 byte arrays that can be used.
* Each array can store up to 256 bytes.

Example: Refer to the “ARRAY_R” example.

ARY_N_R



Description:
Function **Read one integer (signed 32-bit) from an integer array**

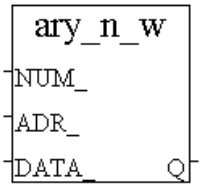
Arguments:

NUM_	integer	array ID to be operated, valid range values from 1 to 6
ADR_	integer	address in the array where the integer is to be stored, valid range values from 1 to 256
DATA_	integer	the integer value returned

* There are 6 integer arrays that can be used.
* Each array can store up to 256 integers.

Example: Refer to the “ARRAY_R” example

ARY_N_W



Description:
Function **Save one integer to an integer array**

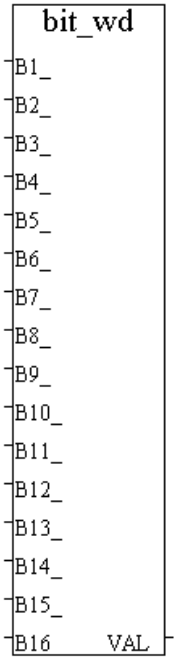
Arguments:

NUM_	integer	array ID to be operated, valid range values from 1 to 6
ADR_	integer	address in the array where the integer is to be stored, valid range values from 1 to 256
DATA_	integer	the integer value to be saved to.
Q_	boolean	if OK. return TRUE, else return FALSE

* There are 6 integer arrays that can be used.
* Each array can store up to 256 integers.

Example: Refer to the “ARRAY_R” example.

BIT_WD

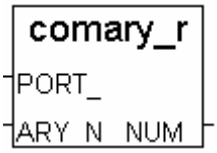


Description:
Function **Convert 16 boolean values to a word value**

Arguments:

B1_ ~ B16_	boolean	the 16 boolean values to be converted
VAL_	integer	the word value after the conversion For ex. If B1_ and B2_ are TRUE and others are all FALSE, VAL_ will be 3. If only B4_ is TRUE and others are all FALSE, VAL_ will be 8

COMARY_R

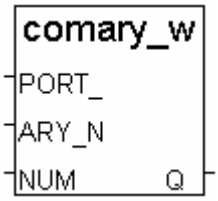


Description:
Function **Read all of the ready data of a COM PORT to a byte array**

Argument:

PORT_	integer	port ID, 2:COM2, 3:COM3, …, 8:COM8
ARY_NO_	integer	Byte array ID (1-24) which is used to store the read bytes
NUM_	integer	return the number of bytes been read

COMARY_W

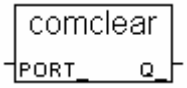


Description:
Function **Write a byte array to a COM PORT**

Argument:

PORT_	integer	port ID, 2:COM2, 3:COM3, …, 8:COM8
ARY_NO_	integer	Byte array ID (1-24) which is to write
NUM_	integer	the number of bytes starting from the first address in the byte array to write
Q_	boolean	OK. return TRUE

COMCLEAR

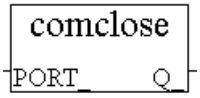


Description:
Function **Clear receiving buffer of a COM PORT**

Argument:

PORT_	integer	port ID, 2:COM2, 3:COM3, …, 8:COM8
Q_	boolean	OK. return TRUE

COMCLOSE

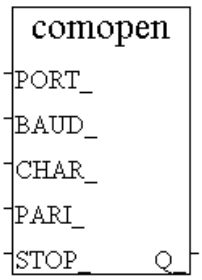


Description:
Function **Close COM PORT**

Argument:

PORT_	integer	port ID, 2:COM2, 3:COM3, …, 8:COM8
Q_	boolean	OK. return TRUE

COMOPEN



Description:
Function **Open COM port**

Argument:

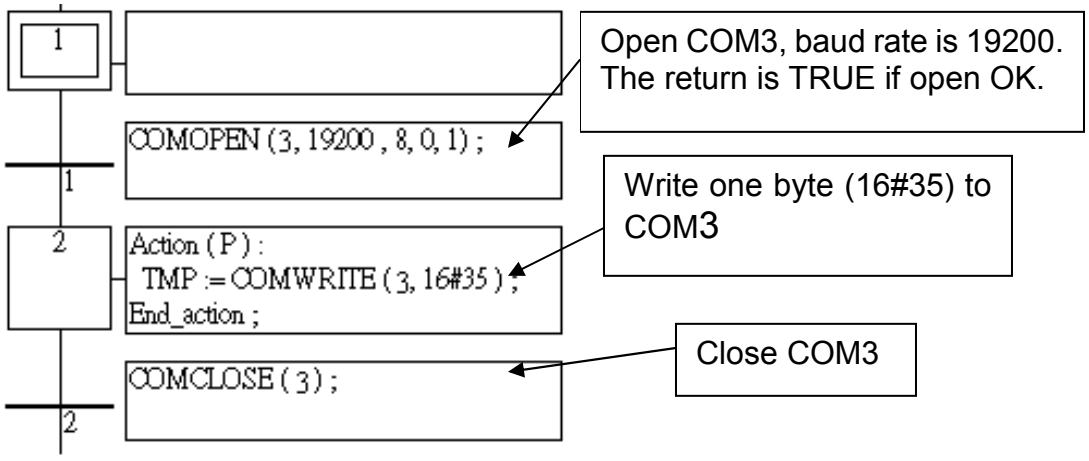
PORT_	integer	port ID, 2:COM2, 3:COM3, …, 8:COM8
BAUD_	integer	baud rate, can be 2400,4800, 9600, 19200, 38400, 57600, 115200
CHAR_	integer	character size, can be 7 or 8
PARI_	integer	parity, can be 0: none, 1: even, 2: odd
STOP_	integer	stop bit, can be 1 or 2
Q_	boolean	OK. return TRUE

Note:

* After COM port is opeded, function "COMREAD" , "COMWRITE ", "COMSTR_W" , "COMCLEAR" , "COMREADY", can be called to read, write, and test data values.
* Recommended for use in SFC program.

Example:

Please refer to Chapter11: Demo01, Demo02 & Demo03.



COMREAD

Description:
Function **Read one byte from a COM port**

Argument:

PORT_	integer	port ID, 2:COM2, 3:COM3, …, 8:COM8
Q_	integer	the data returned

Note:

* Call COMREADY to test data coming or not . If there is data, COMREAD can be used to read the data. If no data comimg, do not call COMREAD or COM port will block.

COMREADY

Description:
Function **Test a COM port for data**

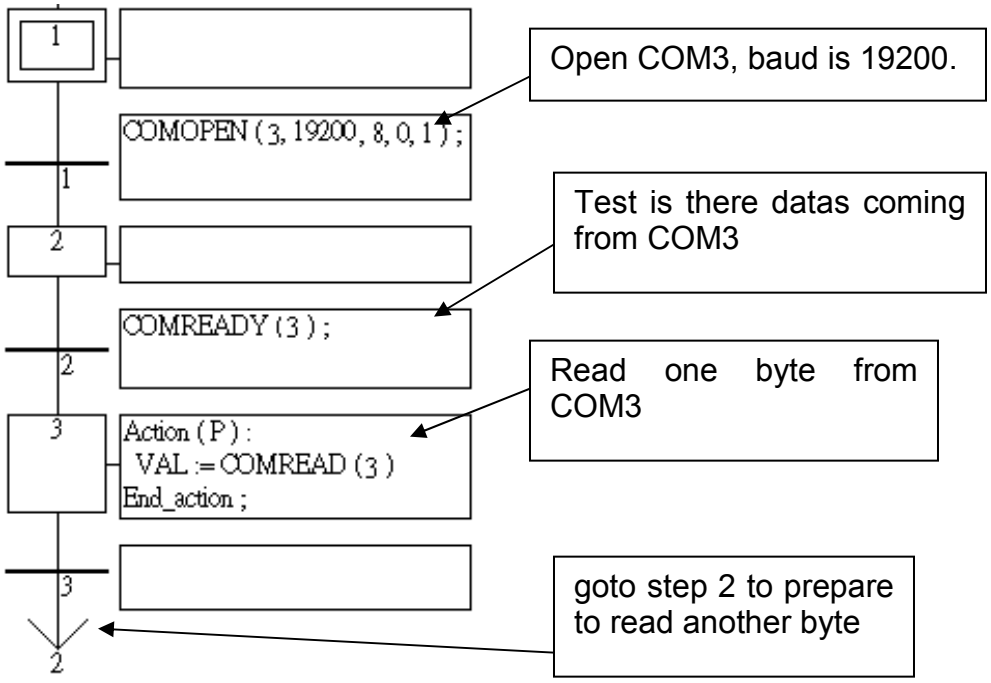
Argument:

PORT_	integer	port ID, 2:COM2, 3:COM3, …, 8:COM8
Q_	boolean	If there is data coming, return TRUE. Else, return FALSE.

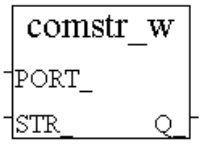
Note:

* This function should be called to test data coming or not . If there is data, COMREAD can be used to read the data. If no data comimg, do not call COMREAD, or COM port will block.

Example:



COMSTR_W

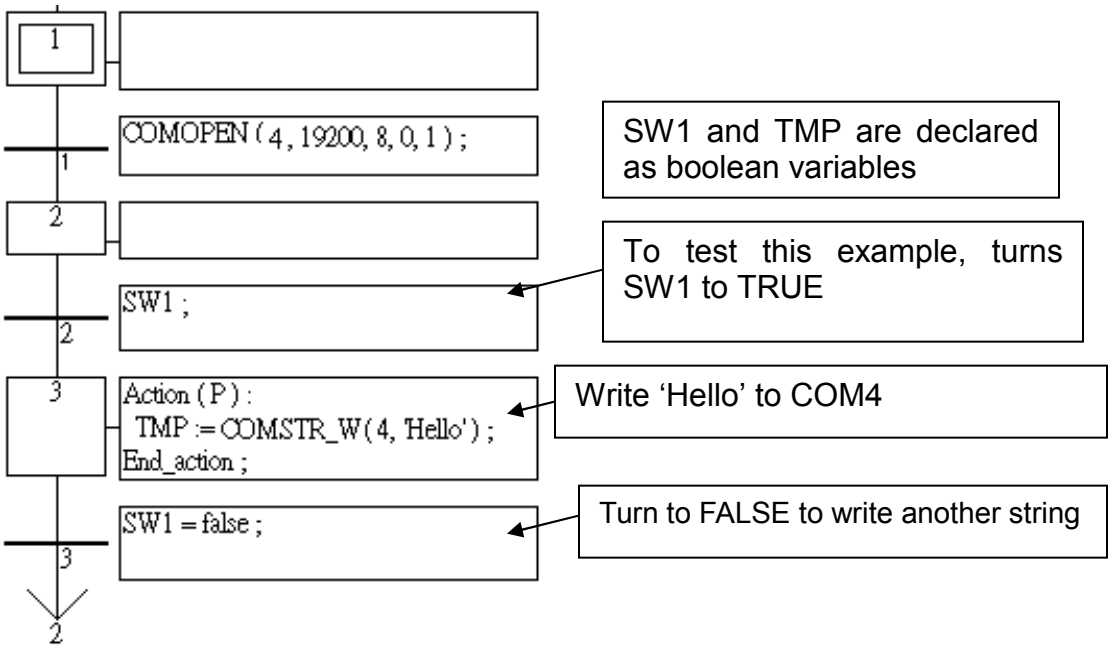


Description:
Function **Write one string to a COM port**

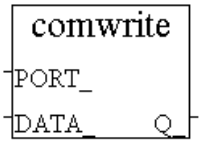
Argument:

PORT_	integer	port ID, 2:COM2, 3:COM3, …, 8:COM8
STR_	Message	the string to be written (max length is 255).
Q_	boolean	Ok. return TRUE, else return FALSE.

Example:



COMWRITE

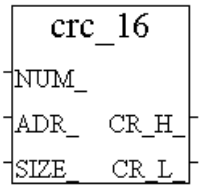


Description:
Function **Write one byte to a COM port**

Argument:

PORT_	integer	port ID, 2:COM2, 3:COM3, …, 8:COM8
DATA_	integer	the byte to be written, valid range values from 0 ~ 255.
Q_	boolean	Ok. return TRUE, else return FALSE.

CRC_16



Description:
Function Block **Calculate checksum - CRC-16**

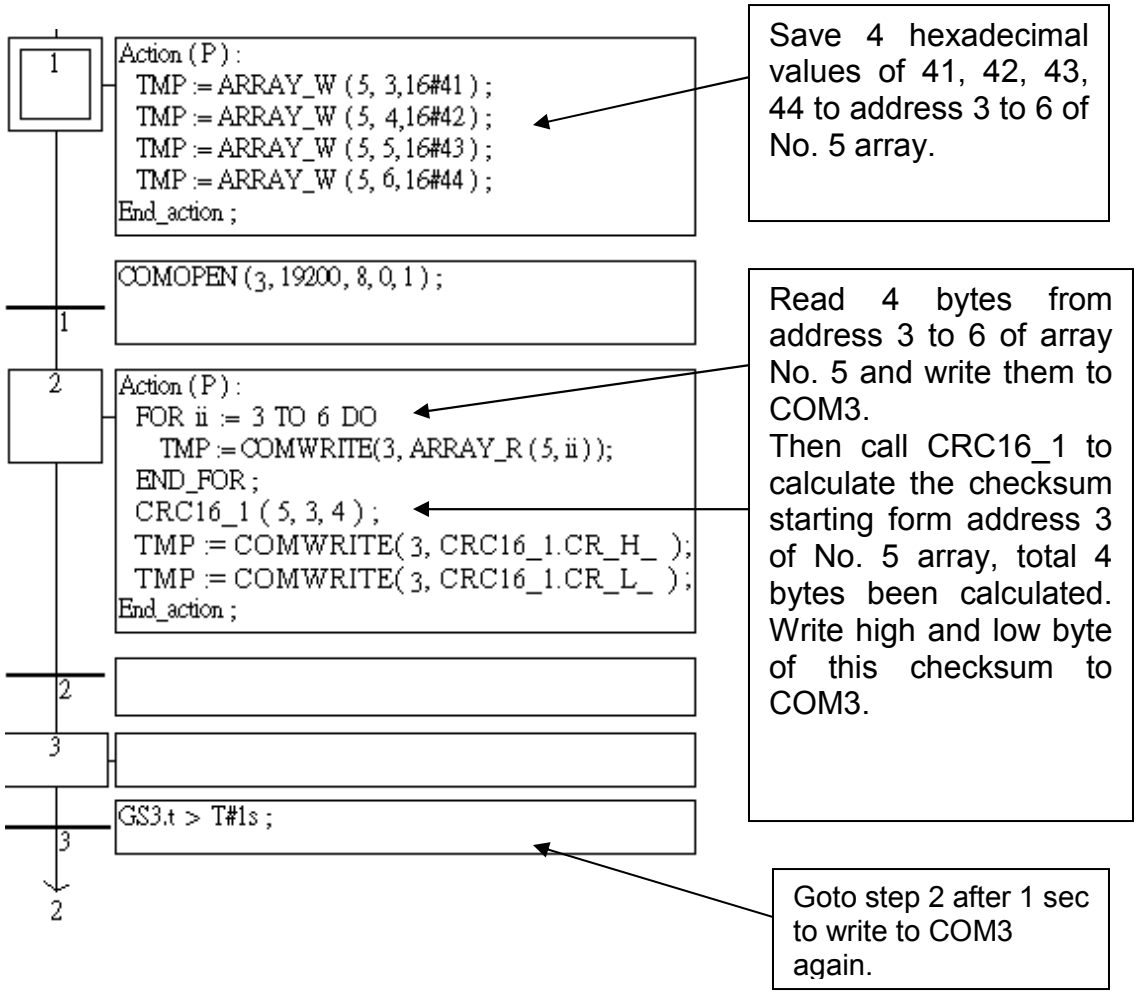
Argument:

NUM_	integer	byte array ID to be operated, valid range values from 1 to 24
ADR_	integer	starting address in the array which is to be calculated
SIZE_	integer	the number of bytes to be calculated
CR_H_	integer	the returned high byte of the CRC-16 after calculation.
CR_L_	integer	the returned low byte of the CRC-16 after calculation.

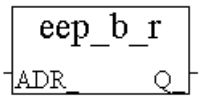
* There are 24 byte arrays that can be used.
* Each array can store up to 256 bytes.

Example:

TMP is declared as a boolean. ii, CR_H_ and CR_L_ as integers, CRC16_1 is declared as FB instance of type – CRC_16.



EEP_B_R



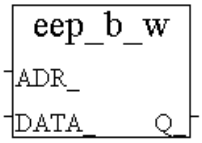
Description:
Function **read a boolean value from the EEPROM**

Argument:

ADR_	integer	address in the EEPROM where the boolean value is stored, valid range values from 1 to 256
Q_	boolean	the boolean value returned

* Read operation of the EEPROM can be used freely without to remove the protection.
* Be careful to use EEP_B_W, EEP_BY_W, EEP_WD_W and EEP_N_W, the EEPROM can only to be written up to 100,000 times.

EEP_B_W



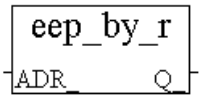
Description:
Function **write a boolean value to the EEPROM**

Arguments:

ADRES_	integer	address in the EEPROM where the boolean value is to be written to, valid range values from 1 to 256
DATA_	Boolean	the boolean value to be written to
Q_	Boolean	Ok. return TRUE.

* To write to the EEPROM, the protection must be removed in advance
* Be careful to use EEP_B_W, EEP_BY_W, EEP_WD_W and EEP_N_W, EEPROM can only to be written up to 100,000 times.

EEP_BY_R



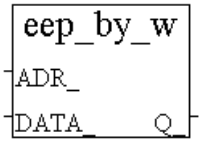
Description:
Function **read a byte (8-bit integer) value from the EEPROM**

Argument:

ADR_	integer	address in the EEPROM where the byte value is stored, valid range values from 1 to 1512
Q_	integer	the byte value returned (0~255)

* If you are using this function with the EEP_WD_R, EEP_WD_W, EEP_N_R, and EEP_N_W functions simultaneously, you must be careful to arrange the ADR_ because they all occupy the same memory area. For example, ADR_2 of EEP_N_R occupies 4 bytes, and it uses the same memory area as ADR_3 and ADR_4 of EEP_WD_R and the same address of ADR_5, 6, 7, and 8 of EEP_BY_R.
* Read operation of the EEPROM will work without removing the EEPROM protection.
* The EEP_B_W, EEP_BY_W, EEP_WD_W and EEP_N_W functions should not be used to write to the EEPROM more than 100,000 times.

EEP_BY_W



Description:
Function **write a byte (8-bit integer) value to the EEPROM**

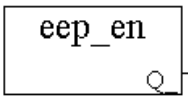
Arguments:

ADR_	integer	address in the EEPROM where the byte value is to be written to, valid range values from 1 to 1512
DATA_	integer	the byte value to be written to, valid range values from 0 to 255.
Q_	Boolean	Ok. return TRUE.

* If you are using this function with the EEP_WD_R, EEP_WD_W, EEP_N_R, and EEP_N_W functions simultaneously, you must be careful to arrange the ADR_ because they all occupy the same memory area. For example, ADR_2 of EEP_N_R occupies 4 bytes, and it uses the same memory area as ADR_3 and ADR_4 of EEP_WD_R and the same address of ADR_5, 6, 7, and 8 of EEP_BY_R.
* Read operation of the EEPROM will work without removing the EEPROM protection.
* The EEP_B_W, EEP_BY_W, EEP_WD_W and EEP_N_W functions should not be used to write to the EEPROM more than 100,000 times.

EEP_EN

Description:
Function **Remove the EEPROM write protection**



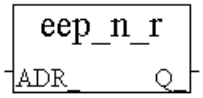
Argument:

Q_	Boolean	Ok: return TRUE, Fail: return FALSE

* BEFORE writing to the EEPROM, the EEPROM write protection must be turned off.
* The EEP_B_W, EEP_BY_W, EEP_WD_W and EEP_N_W functions should not be used to write to the EEPROM more than 100,000 times.

EEP_N_R

Description:
Function **read an 32-bit integer value from the EEPROM**



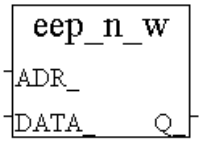
Argument:

ADR_	integer	address in the EEPROM where the 32-bit integer value is stored, valid range values from 1 to 378
Q_	integer	the 32-bit integer value returned

* If you are using this function with the EEP_WD_R, EEP_WD_W, EEP_BY_R, and EEP_BY_W functions simultaneously, you must be careful to arrange the ADR_ because they all occupy the same memory area. For example, ADR_2 of EEP_N_R occupies 4 bytes, and it uses the same memory area as ADR_3 and ADR_4 of EEP_WD_R and the same address of ADR_5, 6, 7, and 8 of EEP_BY_R.
* Read operation of the EEPROM will work without removing the EEPROM protection.
* The EEP_B_W, EEP_BY_W, EEP_WD_W and EEP_N_W functions should not be used to write to the EEPROM more than 100,000 times.

EEP_N_W

Description:
Function **write a 32-bit integer value to the EEPROM**



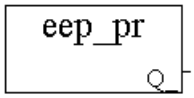
Arguments:

ADR_	integer	address in the EEPROM where the 32-bit integer value is to be written to, valid range values from 1 to 378
DATA_	integer	the 32-bit integer value to be written to
Q_	Boolean	Ok. return TRUE.

* If you are using this function with the EEP_WD_R, EEP_WD_W, EEP_BY_R, and EEP_BY_W functions simultaneously, you must be careful to arrange the ADR_ because they all occupy the same memory area. For example, ADR_2 of EEP_N_R occupies 4 bytes, and it uses the same memory area as ADR_3 and ADR_4 of EEP_WD_R and the same address of ADR_5, 6, 7, and 8 of EEP_BY_R.
* Read operation of the EEPROM will work without removing the EEPROM protection.
* The EEP_B_W, EEP_BY_W, EEP_WD_W and EEP_N_W functions should not be used to write to the EEPROM more than 100,000 times.

EEP_PR

Description:
Function **Set the EEPROM write protection**



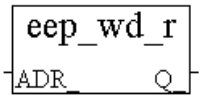
Argument:

Q_	Boolean	Ok: return TRUE, Fail: return FALSE

* After writing to an EEPROM, it is better to turned off the write protection.
* The EEP_B_W, EEP_BY_W, EEP_WD_W and EEP_N_W functions should not be used to write to the EEPROM more than 100,000 times.

EEP_WD_R

Description:
Function **read a word (16-bit integer) value from the EEPROM**

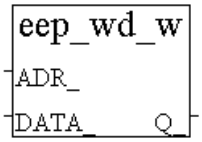


Argument:

ADR_	integer	address in the EEPROM where the word value is stored, valid range value from 1 to 756
Q_	integer	the word value returned (-32768 ~ +32767)

* If you are using this function with the EEP_N_R, EEP_N_W, EEP_BY_R, and EEP_BY_W functions simultaneously, you must be careful to arrange the ADR_ because they all occupy the same memory area. For example, ADR_2 of EEP_N_R occupies 4 bytes, and it uses the same memory area as ADR_3 and ADR_4 of EEP_WD_R and the same address of ADR_5, 6, 7, and 8 of EEP_BY_R.
* Read operation of the EEPROM will work without removing the EEPROM protection.

EEP_WD_W



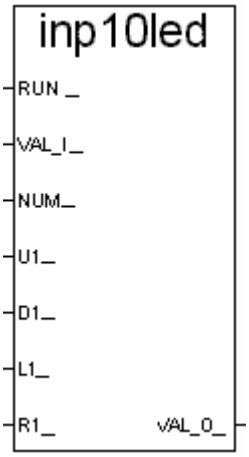
Description:
Function **write a word (16-bit integer) value to the EEPROM**

Arguments:

ADR_	integer	address in the EEPROM where the word value is to be written to, valid range values from 1 to 756
DATA_	integer	the word value to be written to, range from -32768 to +32767.
Q_	Boolean	Ok. return TRUE.

* If you are using this function with the EEP_N_R, EEP_N_W, EEP_BY_R, and EEP_BY_W functions simultaneously, you must be careful to arrange the ADR_ because they all occupy the same memory area. For example, ADR_2 of EEP_N_R occupies 4 bytes, and it uses the same memory area as ADR_3 and ADR_4 of EEP_WD_R and the same address of ADR_5, 6, 7, and 8 of EEP_BY_R.
* Read operation of the EEPROM will work without removing the EEPROM protection.
* The EEP_B_W, EEP_BY_W, EEP_WD_W and EEP_N_W functions should not be used to write to the EEPROM more than 100,000 times.

INP10LED

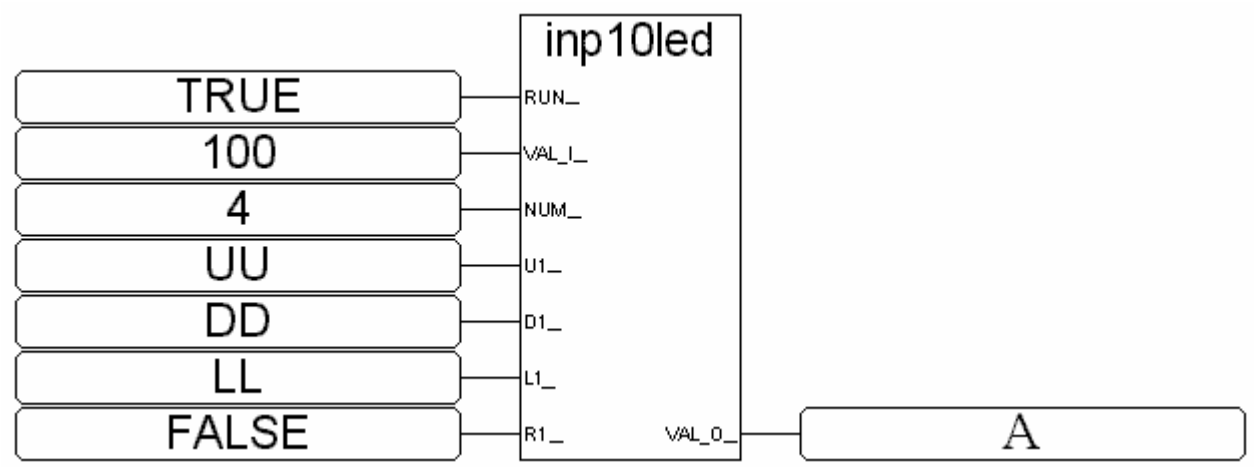


Description:
Function **input an decimal integer from the S_MMI**

Arguments:

RUN_	Boolean	When "TRUE", Process & Display Value To S-MMI
VAL_I_	Integer	Initial Value Displayed On S-MMI, Minimum Value Is "0", maximum is 99999
NUM_	Integer	Number Of Digits To Display, Valid Range From 1 To 5
U1_	Boolean	When Rising From "FALSE" To "TRUE", Add 1 To The Currently Displayed Digit
D1_	Boolean	When Rising From "FALSE" To "TRUE", Subtract 1 From The Currently Displayed Digit
L1_	Boolean	When Rising From "FALSE" To "TRUE", Shift Left 1 Position From Currently Displayed Digit
R1_	Boolean	When Rising From "FALSE" To "TRUE", Shift Right 1 Position From Currently Displayed Digit
VAL_O_	integer	The Displayed Integer Value After Operation

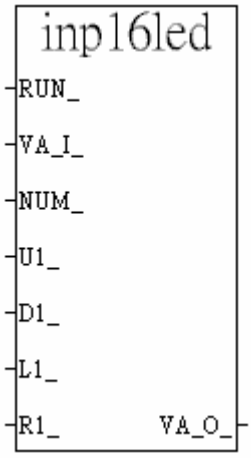
Example:



ST equivalence:

```
A := INP10LED(TRUE,100,4,UU,DD,LL,FALSE);
(* A is declared as an integer variable *)
(* UU,DD,LL are declared as boolean variables, can be linked to
                "push4key" board *)
```

INP16LED

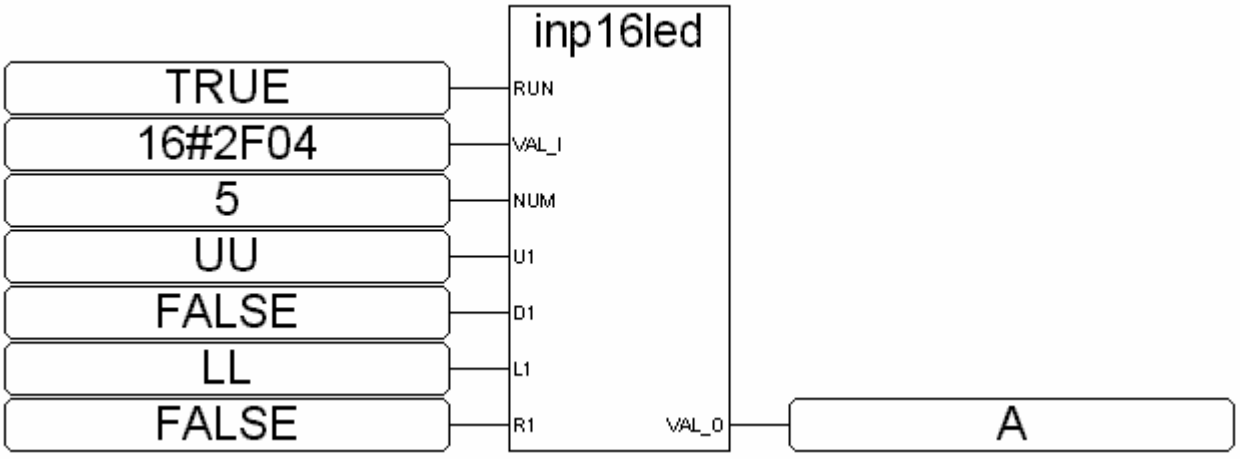


Description:
Function **input an hexadecimal integer from the S_MMI**

Arguments:

RUN_	Boolean	When "TRUE", Process & Display Value To S-MMI
VAL_I_	Integer	Initial Value Displayed On S-MMI, Minimum Value Is "0", maximum is 16#FFFF
NUM_	Integer	Number Of Digits To Display, Valid Range From 1 To 5
U1_	Boolean	When Rising From "FALSE" To "TRUE", Add 1 To The Currently Displayed Digit
D1_	Boolean	When Rising From "FALSE" To "TRUE", Subtract 1 From The Currently Displayed Digit
L1_	Boolean	When Rising From "FALSE" To "TRUE", Shift Left 1 Position From Currently Displayed Digit
R1_	Boolean	When Rising From "FALSE" To "TRUE", Shift Right 1 Position From Currently Displayed Digit
VAL_O_	integer	The Displayed Integer Value After Operation

Example:

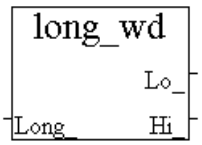


ST equivalence:

```
A := INP16LED(TRUE,16#2F04,4,UU,FALSE,LL,FALSE);
(* A is declared as an integer variable *)
(* UU,LL are declared as boolean variables, can be linked to
        “push4key” board *)
```

LONG_WD

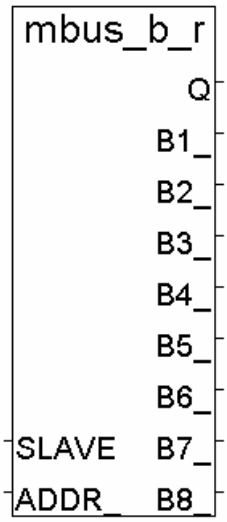
Description:
Function block **Convert one integer to two words**



Arguments:

LONG_	integer	the 32-bit integer to be converted
LO_	integer	the low word value after the coversion, valid from -32768 to +32767
HI_	integer	the high word value after the conversion, valid from -32768 to +32767

MBUS_B_R



Description:
Function block **Read 8 bits (booleans) from the Mdobus device**

Use Modbus function code ---- 1

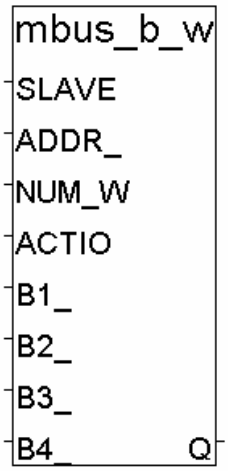
Arguments:

SLAVE_	integer	slave No. of the Modbus device, valid range from 1 to 255
ADDR_	integer	the starting Modbus address to read
Q_	boolean	Ok. return TRUE, else return FALSE
B1_ ~ B8_	boolean	the 8 boolean values that have been read

Note: The total number of "MBUS_B_R" blocks that can be used in one ISaGRAF project is up to 64.

Example: Refer to Chapter 8.

MBUS_B_W



Description:
Function block **write 1 to 4 bits (booleans) to the Mdobus device**

Use Modbus function code 5 when NUM_W = 1
Use Modbus function code 15 when NUM_W = 2 to 4

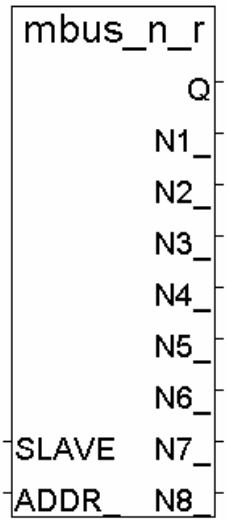
Arguments:

SLAVE_	integer	slave No. of the Modbus device, valid range from 1 to 255
ADDR_	integer	the starting Modbus address to write
NUM_W_	integer	the number of bits to write, valid range from 1 to 4
ACTION_	boolean	Set true to write, set FALSE to do nothing
B1_ ~ B4_	boolean	bits to write
Q_	boolean	Ok. return TRUE, else return FALSE

Note: The total number of "MBUS_B_W" blocks that can be used in one ISaGRAF project is up to 64.

Example: Refer to Chapter 8.

MBUS_N_R



Description:
Function block **Read 8 words (16-bit integer) from the Mdobus device**

Use Modbus function code ---- 3

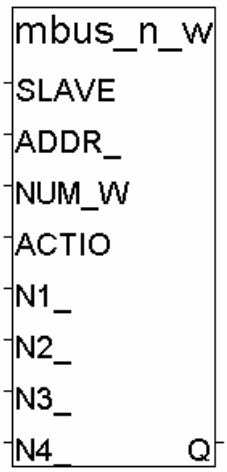
Arguments:

SLAVE_	integer	slave No. of the Modbus device, valid range from 1 to 255
ADDR_	integer	the starting Modbus address to read
Q_		boolean Ok. return TRUE, else return FALSE
N1_ ~ N8_	integer	the 8 word values that have been read, valid range values from 0 to 65535

Note: The total number of "MBUS_N_R" blocks that can be used in one ISaGRAF project is up to 64.

Example: Refer to Chapter 8.

MBUS_N_W



Description:
Function block **write 1 to 4 words (booleans) to the Mdobus device**

Use Modbus function code 6 when NUM_W = 1
Use Modbus function code 16 when NUM_W = 2 to 4

Arguments:

SLAVE_	integer	slave No. of the Modbus device, valid range from 1 to 255
ADDR_	integer	the starting Modbus address to write
NUM_W_	integer	the number of words to write, valid range values from 1 to 4
ACTION_	boolean	Set true to write, set FALSE to do nothing
N1_ ~ N4_	integer	words to write
Q_	boolean	Ok. return TRUE, else return FALSE

Note: The total number of "MBUS_N_W" blocks that can be used in one ISaGRAF project is up to 64.

Example:

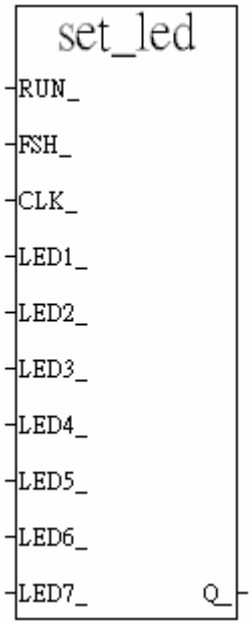
Refer to Chapter 8.

PID_AL

Example:

Please refer to Chapter 11: Demo18 &
"PID_AL.Complex PID algorithm implementation.htm" at
CD_ROM\napdos\isagraf\8000\english_manu\

SET_LED



Description:

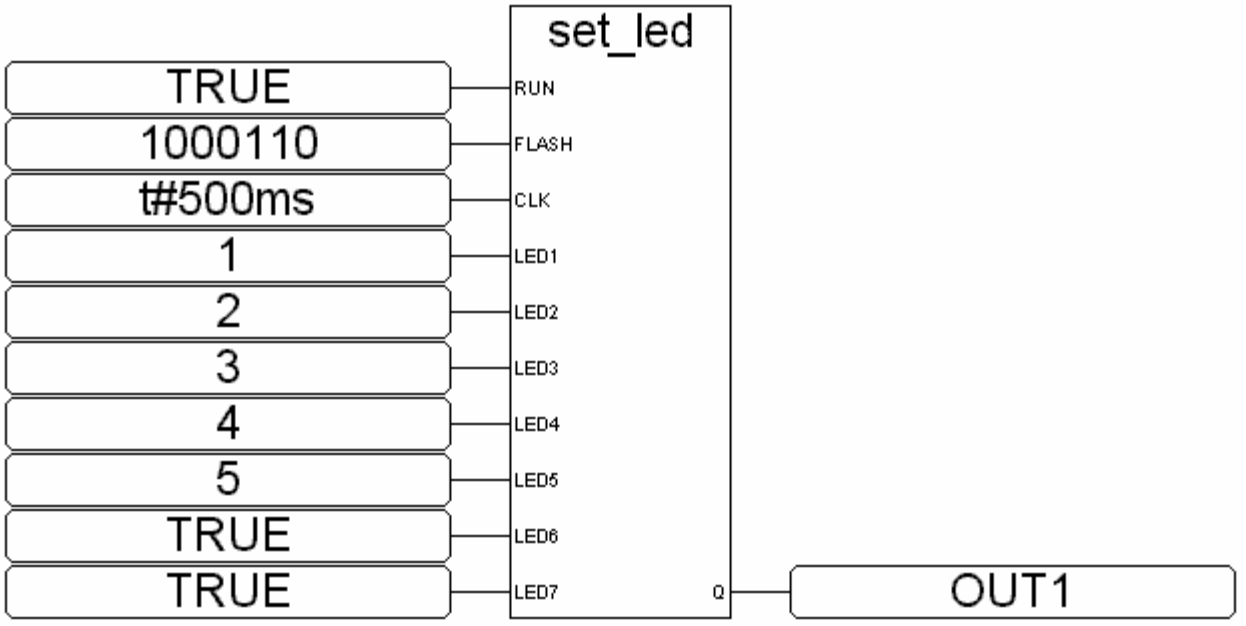
Function Displays A Message To The S-MMI

Arguments:

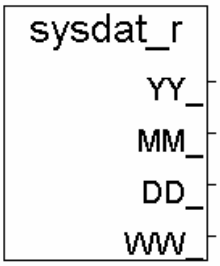
RUN_	Boolean	Set To "TRUE" To Display Message
FLASH_	Integer each	Set each digit To "1" To Flash Message. Example: Set To 11 (0000011) Means The 6th & 7th Display Positions Will Flash. Set To 100001 (0100001) Means The 2nd & 7th Display Positions Will Flash
CLK_	Timer	Amount Of Time For Display To Flash
LED1_	Integer	Value Of Position Display #1
LED2_	Integer	Value Of Position Display #2
LED3_	Integer	Value Of Position Display #3
LED4_	Integer	Value Of Position Display #4
LED5_	Integer	Value Of Position Display #5
LED6_	Integer	Value Of Position Display #6
LED7_	Integer	Value Of Position Display #7

* Refer to section A.3 to see the display char. of LED1 ~ LED5, LED6, LED7.

Example:



SYSDAT_R



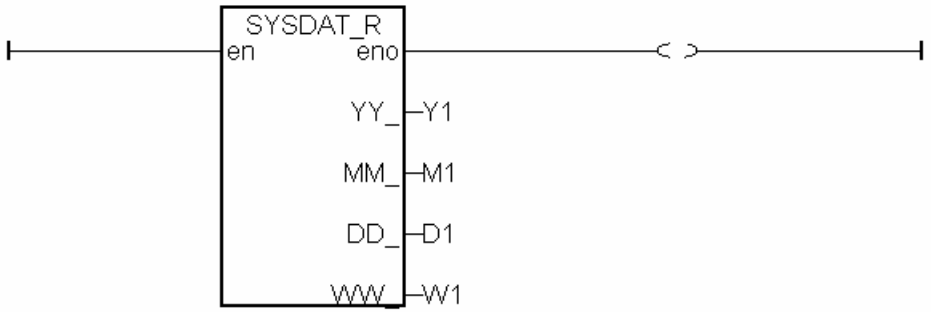
Description:
Function block **Read system year, month, day and date in**

Arguments:

YY_	Integer	Year Returned (Example: 2002, 2003, 2010, Etc.)
MM_	Integer	Month Returned (1 = Jan., 3 =March, 10 =October, Etc.)
DD_	Integer	Day Returned, Valid Range From 1 To 31
WW_	Integer	Date Returned (1 = Monday, 4 = Thursday, 7 = Sunday, Etc.)

Example:

Y1, M1, D1 and W1 are declared as integer variables.

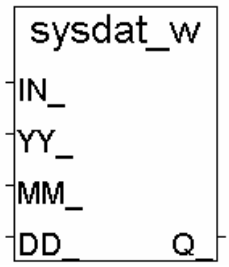


ST equivalence:

```
DAT_R1( );              (* call DAT_R1 *)
Y1 := DAT_R1.YY_ ;      (* get year *)
M1 := DAT_R1.MM_ ;          (* get month *)
D1 := DAT_R1.DD_ ;          (* get day *)
W1 := DAT_R1.WW_ ;          (* get date *)
(* DAT_R1 is declared as FB instance with typed - SYSDAT_R *)
```

SYSDAT_W

Description:
Function block **Set system year, month and day**

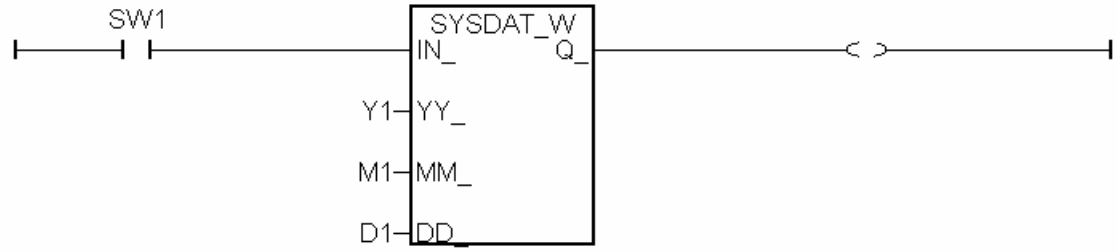


Arguments:

IN_	Boolean	Set System Date When Rising From "FALSE" To "TRUE"
YY_	Integer	Year To Write (Example: 2002, 2003, 2010, Etc.)
MM_	Integer	Month To Write (1=Jan.,3=March,10=October, Etc.)
DD_	Integer	Day Returned, Valid Range From 1 To 31
Q_	Boolean	If "OK", Returns "TRUE"

Example:

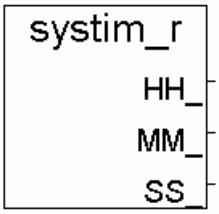
SW1 is declared as a boolean variable. Y1, M1, D1 are declared as integer variables.



St equivalence:

```
DAT_W1( SW1, Y1, M1, D1);     (* call DAT_W1 *)
OUT1 := DAT_W1.Q_ ;                    (* get return value *)
(* DAT_W1 is declared as a FB instance with type - SYSDAT_W *)
(* OUT1 as a boolean variable *)
```

SYSTIM_R



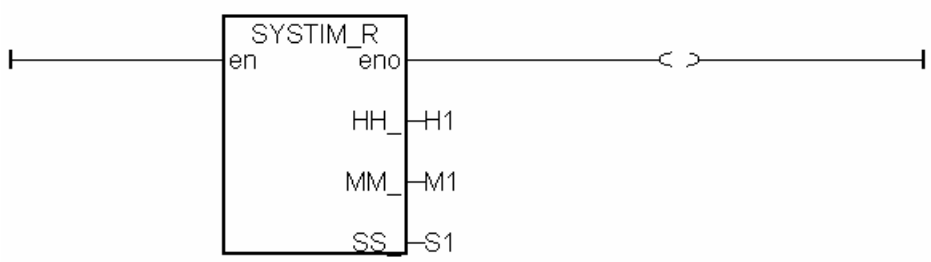
Description:
Function block **Read system hour, minute and second**

Arguments:

HH_	Integer	Hour Returned (Valid Range From 0 To 23)
MM_	Integer	Minute Returned (Valid Range From 0 To 59)
SS_	Integer	Second Returned (Valid Range From 0 To 59)

Example:

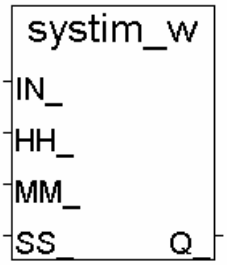
H1, M1 and S1 are declared as integer variables.



ST equivalence:

```
(* TIM_R1 is declared as FB instance with type - SYSTIM_R *)
TIM_R1( );              (* Call TIM_R1 *)
H1 := TIM_R1.HH_ ;      (* get hour *)
M1 := TIM_R1.MM_ ;      (* get minute *)
S1 := TIM_R1.SS_ ;      (* get second *)
```

SYSTIM_W

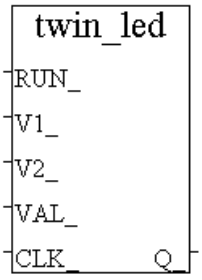


Description:
Function block **Set system hour, minute and second**

Arguments:

IN_	Boolean	Set System Date When Rising From "FALSE" To "TRUE"
YY_	Integer	Year To Write (Example: 2002, 2003, 2010, Etc.)
MM_	Integer	Month To Write (1=Jan.,3=March,10=October, Etc.)
DD_	Integer	Day Returned, Valid Range From 1 To 31
Q_	Boolean	If "OK", Returns "TRUE"

TWIN_LED

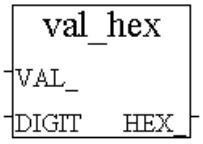


Description:
Function **show a 2 screen values to the S-MMI**

Arguments:

RUN_	boolean	to show if TRUE
V1_	integer	value displayed on the 2 digits on left of 1st screen, 0 ~ 99
V2_	integer	value displayed on the 2 digits on right of 1st screen, 0 ~ 99
VAL_	integer	value displayed on the 2nd screen, -99999 ~ 99999
CLK_	timer	the blinking period of these 2 screens
Q_	boolean	always TRUE

VAL_HEX



Description:
Function **Convert an integer to a fixed-length hexa-message**

Arguments:

VAL_	integer	the value to be converted
DIGIT_	integer	number of digits of HEX_ , valid values are 1 ~ 8. Given others will do no conversion and force HEX_ to ' ' (empty message)
HEX_	message	the hex-message after conversion

Example:

val_hex(100,3) ---> '064'

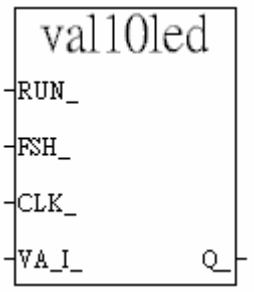
val_hex(192,4) ---> '00C0'

val_hex(4589,2) ---> 'ED' ('11ED', DIGIT_ is 2, force '11' trucated)

val_hex(4589,9) ---> ' ' (DIGIT_ > 8, output ' ')

val_hex(-2,8) ---> 'FFFFFFFE'

VAL10LED



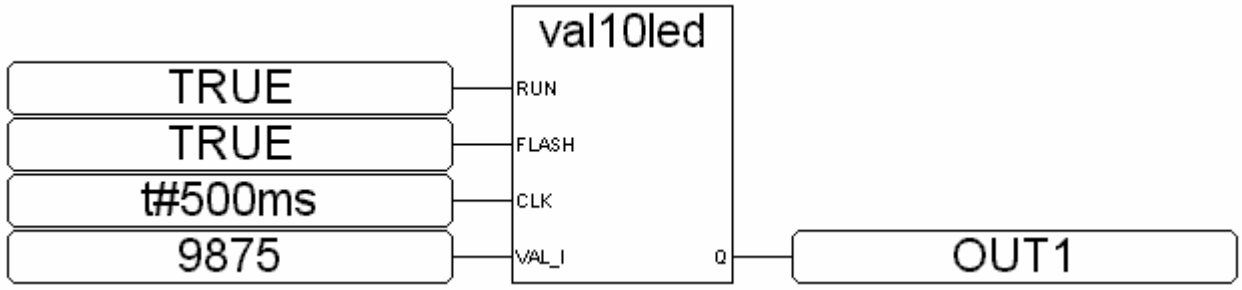
Description:

Function disply an decimal integer on the S-MMI

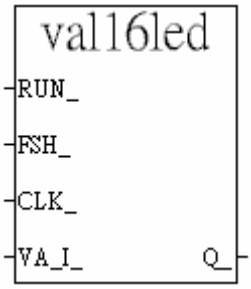
Arguments:

RUN_	Boolean	if TRUE, display it
FLASH_	Boolean	if TRUE, flash it
CLK_	Timer	the flashing period
VAL_I_	Integer	the integer to be displayed Range from -9999 to +99999
Q_	Boolean	always returns TRUE。

Example:



VAL16LED



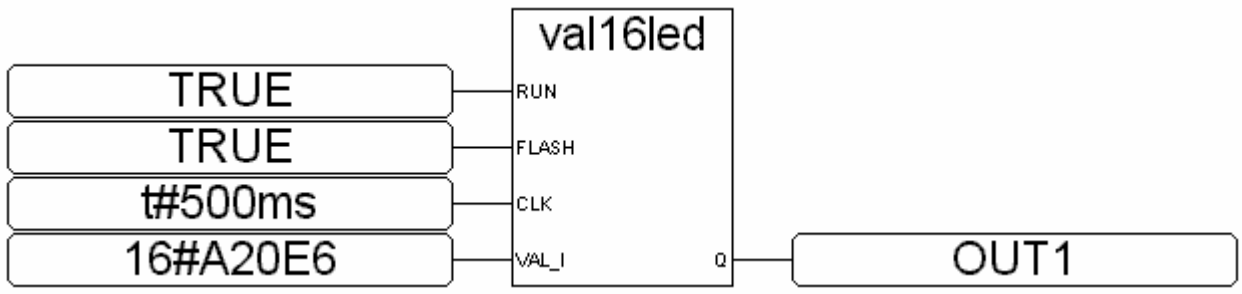
Description:

Function **display an hexadecimal integer on S-MMI**

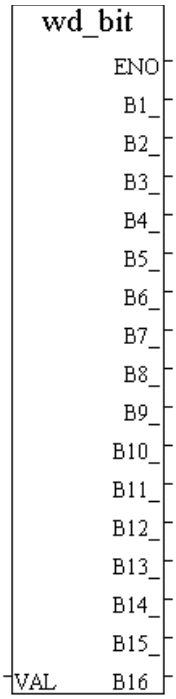
Arguments:

RUN_	Boolean	if TRUE, display it
FLASH_	Boolean	if TRUE, flash it
CLK_	Timer	the flashing period
VAL_I_	integer	the value to be displayed Valid range from 16#0 to 16#FFFFF
Q_	Boolean	always return TRUE

Example:



WD_BIT

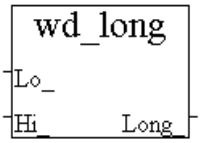


Description:
Function block **Convert a word value to 16 boolean values**

Arguments:

VAL_	integer	the word to be converted.
ENO	boolean	no usage, don't care about it.
B1_ ~ B16_	boolean	the 16 boolean values after converted For ex. If VAL_ is 4, B3_ will be TRUE and others will be FALSE. If VAL_ is 3, B1_ and B2_ will be TRUE and others will be FALSE.

WD_LONG



Description:
Function **Convert two words to one long integer**

Arguments:

Lo_	integer	Low word (only the lowest 16-bit is used)
Hi	integer	High word (only the lowest 16-bit is used)
Long_	integer	the 32-bit integer composed by Lo_ and Hi_ word

Example:

Lo_	Hi_	--->	Long_
-32768 (8000)	-1 (FFFF)	--->	-32768 (FFFF 8000)
-1 (FFFF)	-1 (FFFF)	--->	-1 (FFFF FFFF)
-32768 (8000)	0 (0000)	--->	+32768 (0000 8000)
100 (0064)	4103 (1007)	--->	+ 268 894 308 (1007 0064)

Appendix B: Setting The IP, Mask & Gateway Address of The I-7188EG Controller

This document describe the proper way to set the IP address, address mask and gateway address of the I-7188EG controller.

****
EACH I-7188EG USES TCP/IP PORT NO. 502 TO TALK TO THE HMI AND ISAGRAF WORKBENCH. A MAX. NUMBER OF 5 PCS CAN TALK TO THE I-7188EG THROUGH MODBUS TCP/IP PROTOCOL.
****

1. Create a file folder named "7188" in your hard drive.
   For example, "c:\7188".

For Dos, Windows 95 & Windows 98 Users:
2. Copy \Napdos\ISaGRAF\7188EG\Driver\7188x.exe, 7188x.ini from the CD_ROM into your "7188" folder.
3. Run "\7188\7188x.exe" in your hard drive. A "7188x" screen will appear.

For Windows NT, Windows 2000 & Windows XP Users:
2. Copy \Napdos\ISaGRAF\7188EG\Driver\7188xw.exe, 7188xw.ini from the CD_ROM into your "7188" folder.
3. Run "\7188\7188xw.exe" in your hard drive. A "7188xw" screen will appear.

4. Link from COM1 or COM2 of your PC to COM1 of the I-7188EG controller by a RS232 cable.

5. Power off the I-7188EG controller, connect pin "INIT*" to "GND", and then power it up.

6. If the connection is Ok, messages will appear on the 7188x screen.

```
7188ex>
```

7. Type "ip" to see the current IP address of the I-7188EG.

```
7188ex> ip
IP=192.168.255.255
7188ex>
```

8. Type "setip xxx.xxx.xxx.xxx" to set to a new IP address.

```
7188ex> setip 192.168.1.200
```

Set IP=192.168.1.200
[ReadBack]IP=192.168.1.200
7188ex>

9. Type "mask" to see the current address mask of the I-7188EG.

7188ex> mask
MASK=255.255.0.0
7188ex>

10.Type "setmask xxx.xxx.xxx.xxx" to set to a new address mask.

7188ex> setmask 255.255.255.0
Set MASK=255.255.255.0
[ReadBack]MASK=255.255.255.0
7188ex>

11.Type "gateway" to see the current gateway address.

7188ex> gateway
Gateway=192.168.0.1
7188ex>

12.Type "setgateway xxx.xxx.xxx.xxx" to set to a new gateway address.

7188ex> setgateway 192.168.1.1
Set GATEWAY=192.168.1.1
[ReadBack]Gateway=192.168.1.1
7188ex>

13.Press ALT_X to exit "7188x" and close the DOS SHELL, or
COM1/COM2 of the PC will be occupied.

14. Remove the connection between "INIT*" - "GND", reset the
I-7188EG controller.

Appendix C: Update to New Hardware Driver

The ISaGRAF embedded driver is firmware burned into the flash memory of the I-7188XG / I-7188EG. It can be easily upgraded by the user.

Our newly released driver can also be obtained from the following website.

ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/7188xg/driver/
ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/7188eg/driver/

Warning:
The copyright of the firmware and ISaGRAF embedded driver belongs to ICP DAS CO., LTD.
Only the I-7188XG, I-7188EG, I-8417, 8817, 8437 and 8837 have registered a legal ISaGRAF Target license. To burn an ISaGRAF embedded driver into other controllers is absolutely illegal and may be punished by law.

Make sure of your current driver version before you upgrade it.

1. Create a file folder named "7188" in your hard drive.
   For example, "c:\7188".

For Dos, Windows 95 & Windows 98 Users:
2. Copy \Napdos\ISaGRAF\7188EG\Driver\7188x.exe, 7188x.ini from the CD_ROM into your "7188" folder.
3. Run "\7188\7188x.exe" in your hard drive. A "7188x" screen will appear.

For Windows NT, Windows 2000 & Windows XP Users:
2. Copy \Napdos\ISaGRAF\7188EG\Driver\7188xw.exe, 7188xw.ini from the CD_ROM into your "7188" folder.
3. Run "\7188\7188xw.exe" in your hard drive. A "7188xw" screen will appear.

4. Link COM1 or COM2 of your PC to COM1 of the I-7188XG / I-7188EG controller through a RS232 cable.

5. Power off the I-7188XG / I-7188EG controller, connect pin "INIT*" to "GND" and then power it up.

6. If the connection is Ok, messages will appear on the 7188x screen.

```
7188EX>
```

7. Type "isa7188e *p=" for I-7188EG while type "isa7188 *p=" for I-7188XG. the version No. and copyright message will be displayed.

```
7188> isa7188e  *p=              for I-7188EG
7188> isa7188   *p=              for I-7188XG

i-7188XG / i-7188EG Driver: V1.03 , Mar.01,2002
(C)Copyright:ICP DAS  CO. , LTD.      Taiwan Id:84517297
```

To burn an ISaGRAF embedded driver, follow the following steps.

8. Copy the driver of the correct version into your "7188" folder.
   For example, version 1.03,

   I-7188EG:
   copy \Napdos\ISaGRAF\7188eg\Driver\1.03\isa7188e.exe to \7188\isa7188e.exe
   copy \Napdos\ISaGRAF\7188eg\Driver\1.03\autoexec.bat to \7188\autoexec.bat

   I-7188XG:
   copy \Napdos\ISaGRAF\7188xg\Driver\1.03\isa7188.exe to \7188\isa7188.exe
   copy \Napdos\ISaGRAF\7188xg\Driver\1.03\autoexec.bat to \7188\autoexec.bat

9. Power off the I-7188XG / I-7188EG controller, connect pin "INIT*" to "GND"
   and then power it up.

10. Type "del" and reply "y" to delete the current driver.

```
7188> del
Total File number is 2, do you really want to delete(y/n)?
```

11. Type "load", then press ALT_E and then type "autoexec.bat" .

```
7188> load
File will save to 8000:0000
StartAddr-->7000:FFFF
Press ALT_E to download file!
Input filename:autoexec.bat
Send file info. total 1 blocks
Block 1
Transfer time is: 0.329670 seconds

Back to Terminal mode
```

12. Type "load" again, then press ALT_E and then type "isa7188eg.exe" for I-7188EG while "isa7188.exe" for I-7188XG. It will take about 55 seconds to finish.

```
8000> load
File will save to 8003:0002
StartAddr-->8000:0031
Press ALT_E to download file!
Input filename:isa7188e.exe              or  isa7188.exe
```

Send file info. total 1070 blocks
Block 1070
Transfer time is: 54.505495 seconds

Back to Terminal mode

13. Type "dir" to make sure "autoexec.bat" and "isa7188e.exe" or “isa7188.exe” are well burned.

    7188> dir

14. Press ALT_X to exit "7188x".

15. Remove the connection between "INIT*" - "GND", reset the I-7188XG / I-7188EG controller.

Appendix D: Table of The Analog IO Value

I-87013, I-7013, I-7033

Range Code (Hex)	RTD Type	Data Format	Max Value	Min Value
20 (Default)	Platinum 100 a = 0.00385	Input Range (Celsius)	+100.0	-100.0
		Engineer Unit	+32767	-32768
		2's complement HEX	7FFF	8000
21	Platinum 100 a = 0.00385	Input Range (Celsius)	+100.0	+0.0
		Engineer Unit	+32767	+0
		2's complement HEX	7FFF	0000
22	Platinum 100 a = 0.00385	Input Range (Celsius)	+200.0	+0.0
		Engineer Unit	+32767	+0
		2's complement HEX	7FFF	0000
23	Platinum 100 a = 0.00385	Input Range (Celsius)	+600.0	+0.0
		Engineer Unit	+32767	+0
		2's complement HEX	7FFF	0000
24	Platinum 100 a = 0.003916	Input Range (Celsius)	+100.0	-100.0
		Engineer Unit	+32767	-32768
		2's complement HEX	7FFF	8000
25	Platinum 100 a = 0.003916	Input Range (Celsius)	+100.0	+0.0
		Engineer Unit	+32767	+0
		2's complement HEX	7FFF	0000
26	Platinum 100 a = 0.003916	Input Range (Celsius)	+200.0	+0.0
		Engineer Unit	+32767	+0
		2's complement HEX	7FFF	0000
27	Platinum 100 a = 0.003916	Input Range (Celsius)	+600.0	+0.0
		Engineer Unit	+32767	+0
		2's complement HEX	7FFF	0000
28	Nickel 120	Input Range (Celsius)	+100.0	-80.0
		Engineer Unit	+32767	-262140
		2's complement HEX	7FFF	999A
29	Nickel 120	Input Range (Celsius)	+100.0	+0.0
		Engineer Unit	+32767	+0
		2's complement HEX	7FFF	0000
2A	Platinum 1000 a = 0.00385	Input Range (Celsius)	+600.0	-200.0
		Engineer Unit	+32767	-10922
		2's complement HEX	7FFF	D556

I-8017H

* Each channel can be configured to different range ID

Range Code (Hex)	Data Format	Max value	Min value
05	Input Range	+2.5 V	-2.5 V
	Engineer Unit	+32767	-32768
	2's Complement HEX	7FFF	8000
06	Input Range	+20.0 mA	-20.0 mA
	Engineer Unit	+32767	-32768
	2's Complement HEX	7FFF	8000
07	Input Range	+1.25 V	-1.25 V
	Engineer Unit	+32767	-32768
	2's Complement HEX	7FFF	8000
08 (Default)	Input Range	+10.0 V	-10.0 V
	Engineer Unit	+32767	-32768
	2's Complement HEX	7FFF	8000
09	Input Range	+5.0 V	-5.0 V
	Engineer Unit	+32767	-32768
	2's Complement HEX	7FFF	8000

Range Code (Hex)	Data Format	Max value	Min value
08 (Default)	Input Range	+10.0 V	-10.0 V
	Engineer Unit	+32767	-32768
	2's Complement HEX	7FFF	8000
09	Input Range	+5.0 V	-5.0 V
	Engineer Unit	+32767	-32768
	2's Complement HEX	7FFF	8000
0A	Input Range	+1.0 V	-1.0 V
	Engineer Unit	+32767	-32768
	2's Complement HEX	7FFF	8000
0B	Input Range	+500.0 mV	-500.0 mV
	Engineer Unit	+32767	-32768
	2's Complement HEX	7FFF	8000
0C	Input Range	+150.0 mV	-150.0 mV
	Engineer Unit	+32767	-32768
	2's Complement HEX	7FFF	8000
0D	Input Range (with 125 ohms resistor)	+20.0 mA	-20.0 mA
	Engineer Unit	+32767	-32768
	2's Complement HEX	7FFF	8000

Range Code (Hex)	Data Format	Max value	Min value
00	Input Range	-15.0 mV	-15.0 mV
	Engineer Unit	+32767	-32768
	2's Complement HEX	7FFF	8000
01	Input Range	+50.0 mV	-50.0 mV
	Engineer Unit	+32767	-32768
	2's Complement HEX	7FFF	8000
02	Input Range	+100.0 mV	-100.0 mV
	Engineer Unit	+32767	-32768
	2's Complement HEX	7FFF	8000
03	Input Range	+500.0 mV	-500.0 mV
	Engineer Unit	+32767	-32768
	2's Complement HEX	7FFF	8000
04	Input Range	+1.0 V	-1.0 V
	Engineer Unit	+32767	-32768
	2's Complement HEX	7FFF	8000
05 (Default)	Input Range	+2.5V	-2.5V
	Engineer Unit	+100.00	-100.00
	2's Complement HEX	7FFF	8000
06	Input Range	+20.0 mA	-20.0 mA
	Engineer Unit	+32767	-32768
	2's Complement HEX	7FFF	8000

Range Code (Hex)	Thermocouple Type	Data Format	Max Value	Min Value
0E	J Type	Input Range (Celsius)	+760.0	-210.0
		Engineer Unit	+32767	-9054
		2's Complement HEX	7FFF	DCA2
0F	K Type	Input Range (Celsius)	+1372.0	-270.0
		Engineer Unit	+32767	-6448
		2's Complement HEX	7FFF	E6D0
10	T Type	Input Range (Celsius)	+400.0	-270.0
		Engineer Unit	+32767	-22118
		2's Complement HEX	7FFF	A99A

11	E Type	Input Range (Celsius)	+1000.0	-270.0
		Engineer Unit	+32767	-8847
		2's Complement HEX	7FFF	DD71
12	R Type	Input Range (Celsius)	+1768.0	+0.0
		Engineer Unit	+32767	+0
		2's Complement HEX	7FFF	0000
13	S Type	Input Range (Celsius)	+1768.0	+0.0
		Engineer Unit	+32767	+0
		2's Complement HEX	7FFF	0000
14	B Type	Input Range (Celsius)	+1820.0	+0.0
		Engineer Unit	+32767	+0
		2's Complement HEX	7FFF	0000
15	N Type	Input Range (Celsius)	+1300.0	-270.0
		Engineer Unit	+32767	-6805
		2's Complement HEX	7FFF	E56B
16	C Type	Input Range (Celsius)	+2320.0	+0.0
		Engineer Unit	+32767	+0
		2's Complement HEX	7FFF	0000
17	L Type	Input Range (Celsius)	+800.0	-200.0
		Engineer Unit	+32767	-8192
		2's Complement HEX	7FFF	E000
18	M Type	Input Range (Celsius)	+100.0	-200.0
		Engineer Unit	+16384	-32768
		2's Complement HEX	4000	8000
19	L Type DIN43710	Input Range (Celsius)	+900.0	-200.0
		Engineer Unit	+32767	-7281
		2's Complement HEX	7FFF	E38F

I-7021

Range Code (Hex)	Data Format	Max Value	Min Value
30	Output Range	+20.0 mA	+0.0 mA
	Engineer Unit	+32767	+0
	2's complement HEX	7FFF	0000
31	Output Range	+20.0 mA	+4.0 mA
	Engineer Unit	+32767	+0
	2's complement HEX	7FFF	0000
32 (Default)	Output Range	+10.0 V	+0.0 V
	Engineer Unit	+32767	+0
	2's complement HEX	7FFF	0000

I-7022

Range Type (Hex)	Data Format	Max Value	Min Value
0	Output Range	+20.0 mA	+0.0 mA
	Engineer Unit	+32767	+0
	2's complement HEX	7FFF	0000
1	Output Range	+20.0 mA	+4.0 mA
	Engineer Unit	+32767	+0
	2's complement HEX	7FFF	0000
2 (Default)	Output Range	+10.0 V	+0.0 V
	Engineer Unit	+32767	+0
	2's complement HEX	7FFF	0000

I-8024

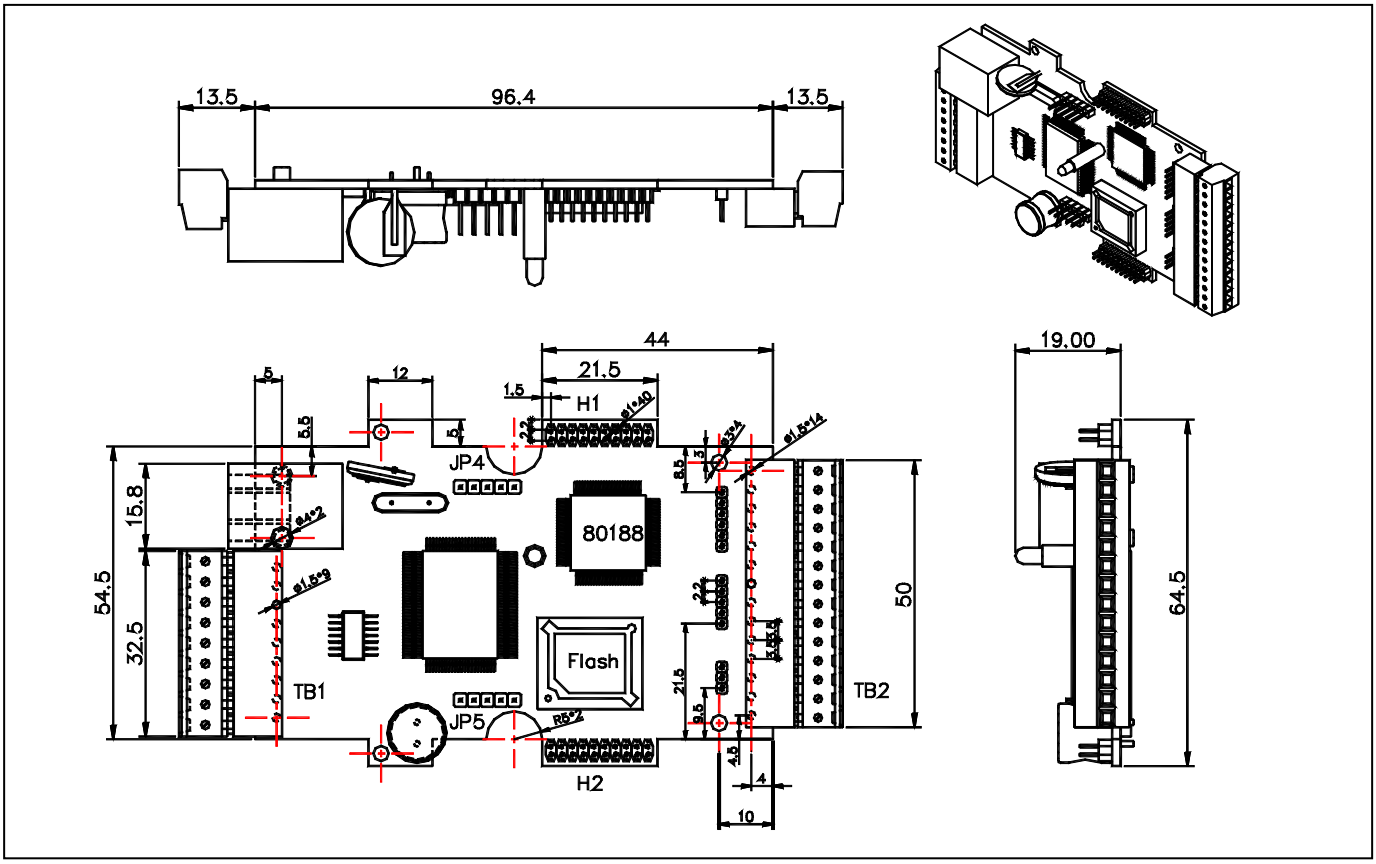
* Each channel can be configured to different range ID

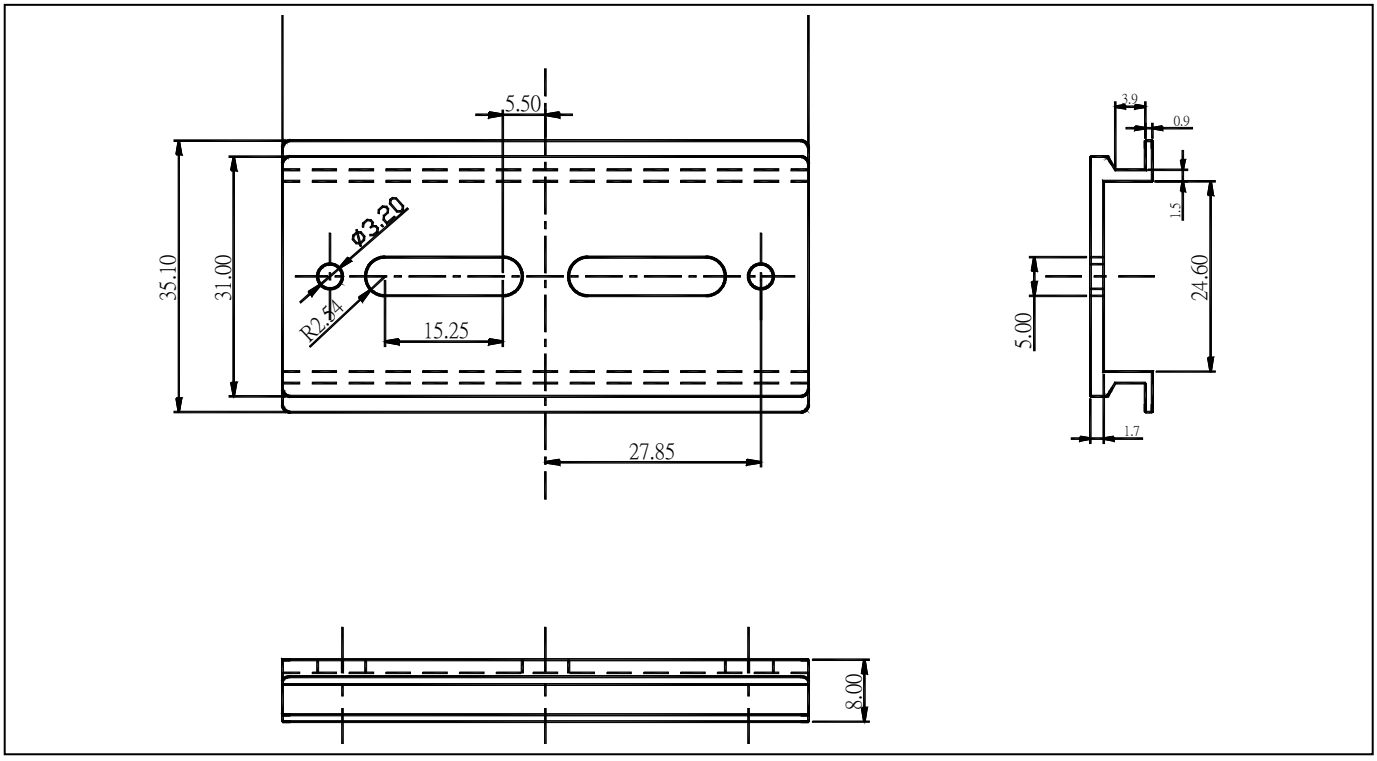
Range Code (Hex)	Data Format	Max Value	Min Value
30	Output Range	+20.0 mA	+0.0 mA
	Engineer Unit	+32767	+0
33	Output Range	+10.0 V	-10.0 V
	Engineer Unit	+32767	-32768

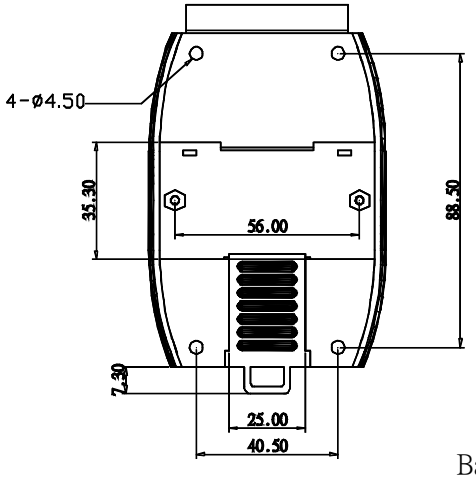
I-87024, I-7024

Range Code (Hex)	Data Format	Max Value	Min Value
30	Output Range	+20.0 mA	+0.0 mA
	Engineer Unit	+32767	+0
31	Output Range	+20.0 mA	+4.0 mA
	Engineer Unit	+32767	+0
32	Output Range	+10.0 V	+0.0 V
	Engineer Unit	+32767	+0
33 (Default)	Output Range	+10.0 V	-10.0 V
	Engineer Unit	+32767	-32768
34	Output Range	+5.0 V	+0.0 V
	Engineer Unit	+32767	+0
35	Output Range	+5.0 V	-5.0 V
	Engineer Unit	+32767	-32768

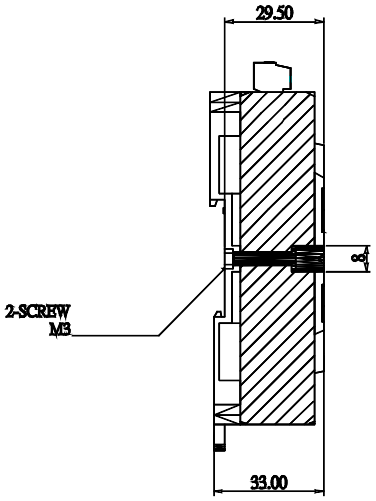
Appendix E: Dimension and Mounting





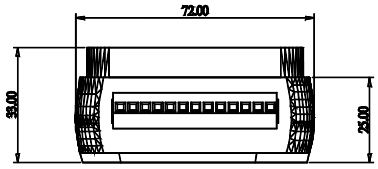


Back View

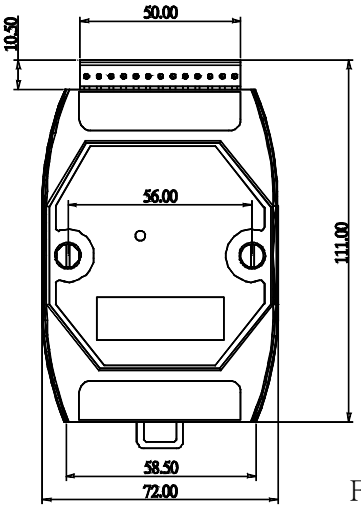


Side View

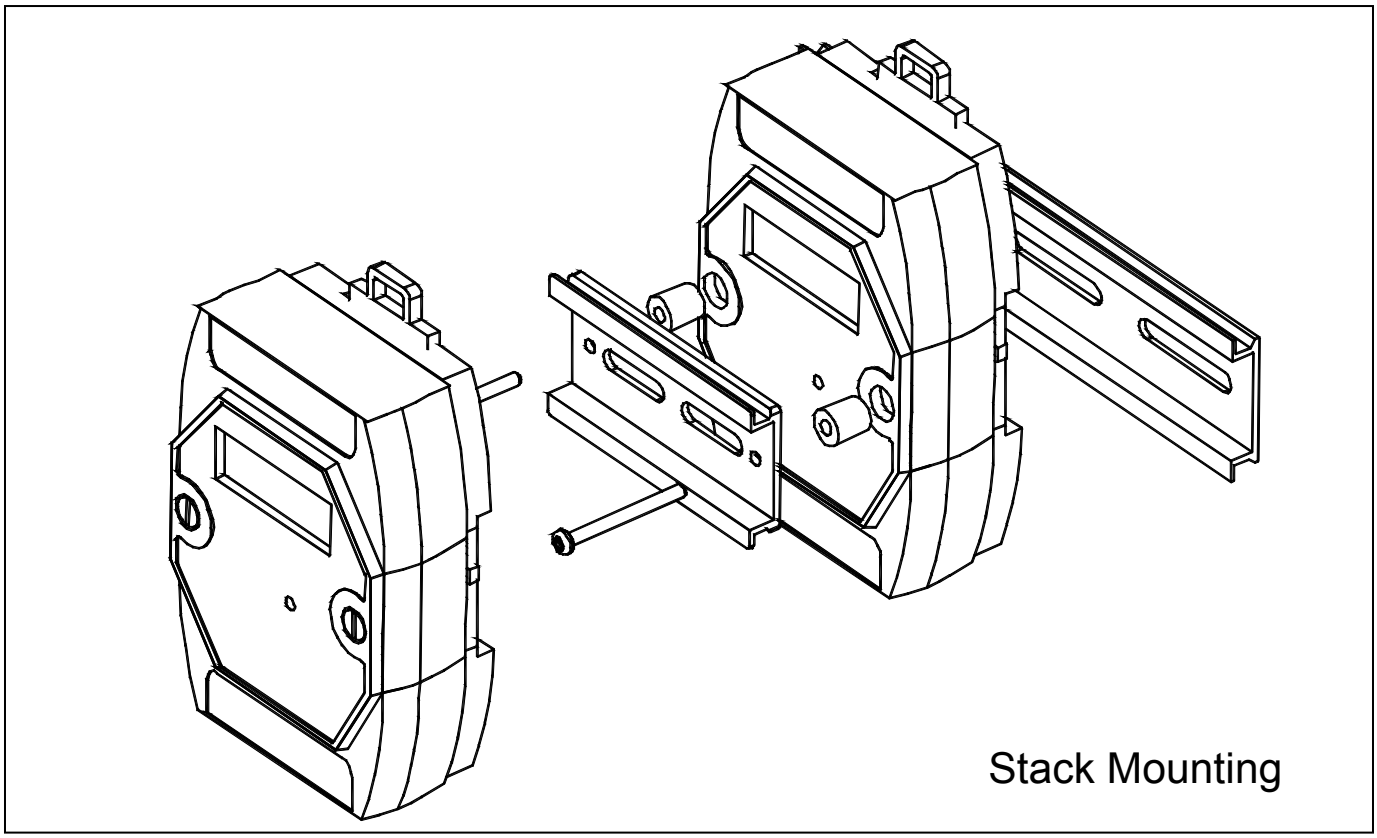
Unit : mm



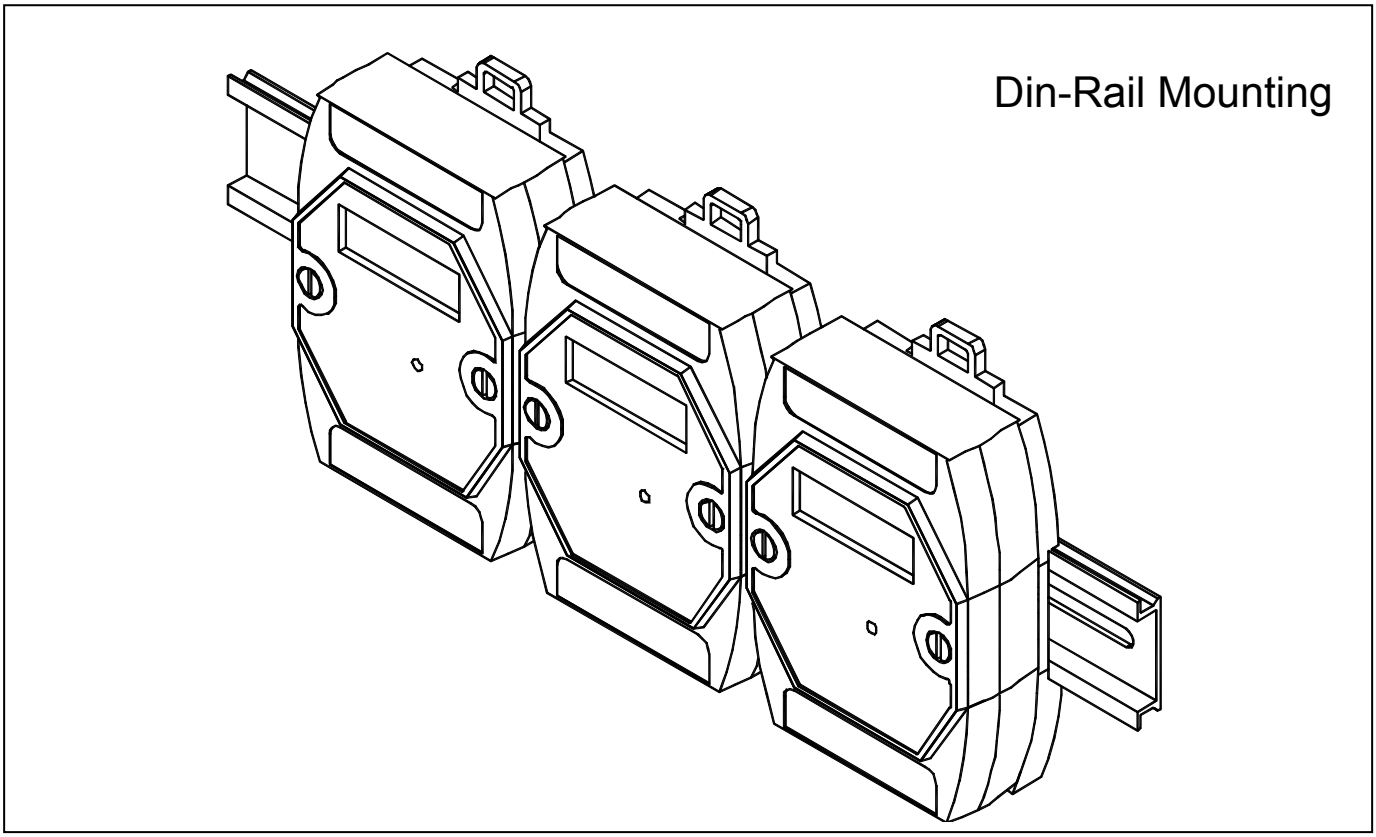
Top View



From View



Stack Mounting



Din-Rail Mounting