User's Manual of ISaGRAF® PAC

By ICP DAS CO. , LTD. & ICP DAS-USA , January 2002, All Rights Reserved

The "User's Manual of ISaGRAF PAC" is intended for integrators, programmers, and maintenance personnel who will be installing and maintaining an ISaGRAF series controller system featuring the ISaGRAF Workbench software program.

XP-8047-CE6 / XP-8046-CE6 / XP-8347-CE6 / XP-8346-CE6 / XP-8747-CE6 / XP-8746-CE6 ,
XP-8147-Atom-CE6 / XP-8146-Atom-CE6 / XP-8347-Atom -CE6 / XP-8346-Atom -CE6 /
XP-8747-Atom -CE6 / XP-8746-Atom -CE6 ,
WP-8147 / WP-8146 / WP-8447 / WP-8446 / WP-8847 / WP-8846 ,
WP-8137 / WP-8136 / WP-8437 / WP-8436 / WP-8837 / WP-8836 ,
WP-5147/ WP-5146/ WP-5147-OD/ WP-5146-OD ,
VP-25W7 / VP-25W6 / VP-23W7 / VP-23W6 ,
μPAC-5007/5107/5207/5307/5507 , μPAC-7186EG , I-7188EG , I-7188XG , VP-2117 ,
iP-8447 / iP-8847 / iP-8417 / iP-8817 ,
I-8417 / I-8817 / I-8437-80 / I-8837-80

ICP DAS CO., LTD. would like to congratulate you own your purchase of our ISaGRAF controllers. The ease to integration of the controller system and the power of the IEC 61131-3 ISaGRAF software program combine to make a powerful, yet inexpensive industrial process control 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.

FAQ:

Please visit www.icpdas.com - "FAQ" - "Software" - "ISaGRAF" for Frequently Asked Question, or visit http://www.icpdas.com/faq/isagraf.htm

Data Sheet :

Please visit http://www.icpdas.com/products/PAC/i-8000/isagraf.htm (click "Data Sheet" icon)
or visit http://www.icpdas.com/products/PAC/i-8000/data%20sheet/data%20sheet.htm

Table of Contents

Reference Guide

English manual:

File Name: "user_manual_i_8xx7.pdf"
iP-8xx7, I-8000 & μPAC-7186, I-7188 CD: \napdos\isagraf\8000\english_manu\
WP-8xx7 CD: \napdos\isagraf\wp-8xx7\english_manu\
WP-5xx7 CD: \napdos\isagraf\wp-5xx7\english_manu\
XP-8xx7-CE6 CD: \napdos\isagraf\xp-8xx7-ce6\english_manu\
VP-2xW7 CD: \napdos\isagraf\vp-25w7-23w7\english_manu\

中文 **ISaGRAF** 進階使用手冊**:**

File Name: "chinese_user_manual_i_8xx7.pdf"
iP-8xx7, I-8000 & μPAC-7186, I-7188 CD: \napdos\isagraf\8000\chinese_manu\
WP-8xx7 CD: \napdos\isagraf\wp-8xx7\chinese_manu\
WP-5xx7 CD: \napdos\isagraf\wp-5xx7\chinese_manu\
XP-8xx7-CE6 CD: \napdos\isagraf\xp-8xx7-ce6\chinese_manu\
VP-2xW7 CD: \napdos\isagraf\vp-25w7-23w7\ chinese_manu\

Soft-GRAF HMI : The XP-8xx7-Atom-CE6, XP-8xx7-CE6, WP-8xx7, VP-25W7, VP-23W7 and WP-5xx7 support the Soft-GRAF software to create a colorful HMI application. Please refer to http://www.icpdas.com/faq/isagraf.htm > FAQ-146 and http://www.icpdas.com/products/Software/Soft-GRAF/soft-graf.htm .

All ISaGRAF Getting Started Manual (User Manual):
http://www.icpdas.com/products/PAC/i-8000/isagraf.htm (click "Manual" icon)

Resource on the Internet:
Newly updated ISaGRAF IO libraries, drivers and manuals can be found at
http://www.icpdas.com/products/PAC/i-8000/isagraf.htm (click "Driver" or "Manual" icon)

Technical Service:
Please contact local agent or email problem-report to service@icpdas.com
New information can be found at www.icpdas.com

FAQ:
Please visit www.icpdas.com - "FAQ" - "Software" - "ISaGRAF" for Frequently Asked Question,
or visit http://www.icpdas.com/products/PAC/i-8000/isagraf.htm (click "FAQ" icon)
or visit http://www.icpdas.com/faq/isagraf.htm

Performance Comparison Table 1 of ISaGRAF PACs

PACs	CPU	Compared with I-8417: Normal running Speed (Normal PLC scan-time)	Compared with I-8417: Normal Speed for floating point calculation (scan-time)	Ethernet	ISaGRAF code size limitation (bytes)	Memory for running program (bytes)
XP-8xx7-CE6	LX 800 500 MHz	About **10~50** (times) (3~15 ms)	About **10~50** (times) (3~15 ms)	**2** ports **10/100** Mbps	**2 MB**	About **200~400 MB**
WP-8xx7	PXA270, 520 MHz or compatible	About **10~30** (times) (3~15 ms)	About **10~30** (times) (3~15 ms)	**2** ports **10/100** Mbps	**1 MB**	About **20~40 MB**
VP-25W7 **VP-23W7**	PXA270, 520 MHz or compatible	About **10~30** (times) (3~15 ms)	About **10~30** (times) (3~15 ms)	1 port **10/100** Mbps	**1 MB**	About **20~40 MB**
VP-2117	80186,**80** MHz or compatible	About **4** (times) (2~25 ms)	About **0.8** (times) (10~125 ms)	1 port **10/100** Mbps	64 KB	About **768 KB**
iP-8447 **iP-8847**	80186,**80** MHz or compatible	About **4** (times) (2~25 ms)	About **0.8** (times) (10~125 ms)	**2** ports **10/100** Mbps	64 KB	About **768 KB**
iP-8417 **iP-8817**				-		About 512 KB
I-8437-80 **I-8837-80**				1 port 10 Mbps		
I-8437 **I-8837**	80188 40 MHz or compatible	About 1 (times) (5~100 ms)	About **0.2** (times) (25~500 ms)			
I-8417 **I-8817**				-		
µPAC-5xx7	80186,**80** MHz or compatible	About **4** (times) (2~5 ms)	About **0.8** (times) (10~125 ms)	1 port **10/100** Mbps	64 KB	About **768 KB**
µPAC-7186EG						About **640 KB**
I-7188EG	80188,40 MHz or compatible	About 1 (times) (5~100 ms)	About **0.2** (times) (25~500 ms)	**1 port** **10 Mbps**		About 512 KB
I-7188XG				-		

Note: W-8xx7/I-8x37 has phased out. Please select compatible WP-8x47/iP-8x47.

Performance Comparison Table 2 of ISaGRAF PACs

XPAC, WinPAC Series with iPAC:

OS	WinCE		MiniOS7
Model	XP-8xx7-CE6 *1	WP-8x37/WP-8x47 *1	iP-8447 iP-8847 *1
Modbus TCP Master (Max. Connecting)	Max. 100 devices		-
Modbus RTU/ASCII Master Function Block (Max.)	(Per port) 256		(Total) 128
Modbus RTU/ASCII Master COM Port (Max.) *2	33 ports COM1 ~ 33	10 ports 1 ~ 14	2 ports 1 ~ 5
Modbus RTU Slave COM Port (Max.) *2	9 ports COM1 ~ 33	5 ports 1 ~ 8	2 ports 1 or 2/3
Modbus TCP/IP Slave Connections *3	64	32	6
Modbus Address Range	1 ~ 8191		1 ~ 4095
VGA Resolution (Max.)	1024x768	1024x768/800x600	-
USB Port *4	2	2/1	-
Battery Backup SRAM *5	512 KB		
PAC to PAC Data Exchange	Ebus		Fbus, Ebus
Send E-mail (file attached)*6	Yes		
Redundant Ethernet Port *7	Yes		
Mbus24r & mbus24r1 Function Block	Yes		
Mbus_xr & Mbus_xr1 Function Block *8	Yes		-
Software Features (Require Optional Accessories)			
Support FRnet I/O *9	Yes		
Support CAN/CANopen *10	Yes		
Support VW Sensor	Yes		
Support New Redundant System *11	Yes		-
Remote I/O Modules (Optional Accessories)			
Support Ethernet I/O (with I-8KE4/E8-MTCP)	Yes		-
Support I-7K/87K I/O (*Only support 1 COM Port)	Max. Connecting: 255 COM 3 or 4	2 or 3	64 2 or 3 or 4

ViewPAC Series PAC:

OS	WinCE	MiniOS7
Model	**VP-25W7/VP-23W7**	**VP-2117**
Modbus TCP Master (Max. Connecting)	**Max. 100 devices**	-
Modbus RTU/ASCII Master Function Block (Max.)	(Per port) **256**	(Total) 128
Modbus RTU/ASCII Master COM Port (Max.) ***2**	10 ports COM 2, 3, 5 ~ 14	2 ports 1 ~ 3, 5
Modbus RTU Slave COM Port (Max.) ***2**	5 ports COM 2, 3, 5 ~ 8	2 ports 1 or 2/3
Modbus TCP/IP Slave Connections ***3**	32	6
Modbus Address Range	1 ~ 8191	1 ~ 4095
LCD Monitor	TFT 5.7"/3.5"	STN
Touch Panel	Yes/ -	-
VGA Resolution (Max.)	640x480/320x240	128x64
USB Port ***4**	1	-
Battery Backup SRAM ***5**	512 KB	
PAC to PAC Data Exchange	Ebus	Fbus, Ebus
Send E-mail (file attached) ***6**	Yes	
Redundant Ethernet Port ***7**	Yes	-
Mbus24r & mbus24r1 Function Block	Yes	
Mbus_xr & Mbus_xr1 Function Block ***8**	Yes	-
Software Features (Require Optional Accessories)		
Support FRnet I/O ***9**	Yes	
Support CAN/CANopen ***10**	Yes	
Support VW Sensor	Yes	
Support New Redundant System ***11**	Yes	-
Remote I/O Modules (Optional Accessories)		
Support Ethernet I/O (with I-8KE4/E8-MTCP)	Yes	-
Support I-7K/87K I/O (*Only support 1 COM Port)	Max. Connecting: **255** COM 2 or 3	64

I-8000, iPAC Series with WinPAC:

OS	MiniOS7				WinCE
Model	**I-8417/8817**	**I-8x37-80** ***1**	**iP-8447** **iP-8847** ***1**	**iP-8417** **iP-8817** ***1**	**WP-8x37/** **WP-8x47** ***1**
Modbus TCP Master (Max. Connecting)	-				**Max. 100 devices**
Modbus RTU/ASCII Master Function Block (Max.)	(Total)				(Per Port)
	64		128		**256**
Modbus RTU/ASCII Master COM Port (Max.) ***2**	2 ports				10 ports
	COM 1, 3, 4, 5		1 ~ 5		1 ~ 14
Modbus RTU Slave COM Port (Max.) ***2**	2 ports				5 ports
	COM 1, 2	1, 3	1 or 2/3		1 ~ 8
Modbus TCP/IP Slave Connections ***3**	0	4	6	0	32
Modbus Address Range	1 ~ 4095				1 ~ 8191
VGA Resolution (Max.)	-				1024x768/800x600
USB Port ***4**	-				2/1
Battery Backup SRAM ***5**	Optional		512 KB		
PAC to PAC Data Exchange	Fbus	Fbus, Ebus		Fbus	Ebus
Send E-mail (file attached) ***6**	-	Yes		-	Yes
Redundant Ethernet Port ***7**	-		Yes	-	Yes
Mbus24r & mbus24r1 Function Block	-		Yes		
Mbus_xr & Mbus_xr1 Function Block ***8**	-			-	Yes
Software Features (Require Optional Accessories)					
Support FRnet I/O ***9**	-		Yes		
Support CAN/CANopen ***10**	-		Yes		
Support VW Sensor	Yes				
Support New Redundant System ***11**	-				Yes
Remote I/O Modules (Optional Accessories)					
Support Ethernet I/O (with I-8KE4/E8-MTCP)	-				Yes
Support I-7K/87K I/O (*Only support 1 COM Port)	Max. Connecting: 64				**255**
	COM 3 or 4		2 or 3 or 4		2 or 3

I-7188, µPAC Series PAC:

OS	MiniOS7			
Model	**I-7188XG**	**I-7188EG**	**µPAC-7186EG**	**µPAC-5xx7 *1**
Modbus RTU/ASCII Master Function Block (Max.)	(Total) 64		(Total) **128**	
Modbus RTU/ASCII Master COM Port (Max.) ***2**	COM2, 3	COM1, 2, 3		
Modbus RTU Slave COM Port (Max. 2 Port) ***2**	COM1 or 2/3			
Modbus TCP/IP Slave Connections ***3**	0	**4**	**6**	
Modbus Address Range	1 ~ 4095			
Battery Backup SRAM ***5**	Optional			512K
PAC to PAC Data Exchange	Fbus	Fbus, Ebus		
Send E-mail (file attached) ***6**	-		Yes	
Mbus24r & mbus24r1 Function Block	-		Yes	
Software Features (Require Optional Accessories)				
Support FRnet I/O ***9**	-		Yes	-
Support CAN/CANopen ***10**	-		Yes	
Remote I/O Modules (Optional Accessories)				
Support I-7K/87K I/O (*Only support 1 COM Port)	Max. Connecting: 64 COM 2 or 3			

Annotations:

*1. µPAC-5xx7 represents µPAC-5007/5107/5207/5307/5507.
I-8x37/I-8x37-80 represents the products of I-8437/8837/8437-80/8837-80.
iP-8xx7 represents the products of iP-8417/8817/8447/8847.
WP-8x37 represents the products of WP-8137/8437/8837.
WP-8x47 represents the products of WP-8147/8447/8847.
XP-8xx7-CE6 represents the products of XP-8047-CE6/8347-CE6/8747-CE6

*2. I-8xx7's COM5 ~ 20 & W-8x47/ 8x37's COM5 ~ 14 resides at the I-8112/8114 /8142/8144/ 8142i expansion modules ;

iP-8xx7's COM5~20 & VP-2117's COM5~16 resides at the I-8112iW/ I-8114W/ I-8114iW/ I-8142iW/ I-8144iW expansion modules;

WP-8x47, WP-8x37 and VP-25W7/23W7's COM5 ~ 14 resides at the I-8112iW/ I-8114W/ I-8114iW/ I-8142iW/ I-8144iW expansion modules;

XP-8x47-CE6's COM6 ~ 33, resides at the I-8112iW/ I-8114W/ I-8114iW/ I-8142iW/ I-8144iW expansion modules;

I-7188/ µPAC-7186's COM3 ~ 8 resides at the X-board (X5xx) expansion boards.
µPAC-5xx7's COM3 ~ 8 resides at the XW-board (XW5xx) expansion boards.

*3. The Ethernet communication of the XP-8xx7-CE6 is more efficient than WP-8xx7 and VP-2xW7. It supports up to 64 Modbus TCP/IP connections.

The W-8x47 with driver version 4.02 or older version only supports 8 Modbus TCP/IP connections, while supports up to 32 Modbus TCP/IP connections since the version 4.03.
If the controller is W-8347/8747 (two Ethernet ports), its OS image must update to the version released on July, 1, 2008 to ensure the network communications is correct.
Please refer to www.icpdas.com > FAQ > Software > ISaGRAF > 095 for more information.

*4. The USB port for the mouse device of the XP-8xx7-CE6 is more efficient than the WP-8xx7 and VP-2xW7. The WP-8x37 supports 2 USB Port, the WP-8x47 supports 1 USB Port.

*5. I-8x17/8x37-80 equip with S256/S512, µPAC-7186EG, I-7188EG/XG equip with X607 (128K) / X608 (512K), can support up to 1024 retained variables. The data, date & time can also be stored in it.

*6. µPAC-7186EG has to use an extra X607/X608 battery backup SRAM expansion card for sending E-mail with an attached file, or it can only send E-mail without attached file.

*7. If the cable of one Ethernet port is broken or damaged, the PC/HMI can communicate with the other Ethernet port by Modbus TCP/IP protocol.
(Please plug one I-8135W in VP-25W7/23W7 to enable the 2nd Ethernet port)

*8. The Mbus_xr and Mbus_xr1 can read max. 120 words or 60 long integers or 60 real values. Please refer to www.icpdas.com > FAQ > Software > ISaGRAF > FAQ-101 for more information.

*9. To support FRnet I/O in µPAC-7186EG, please insert one FX-016 in it.
VP-2xW7 & VP-2117 support Max. **3** pcs. of I-8172W (Max. ch.768 DI & 768 DO).
iP-8xx7 support Max. **4** pcs. of I-8172W (Max. ch. 1024 DI & 1024 DO).
WP-8x47 & WP-8x37 support Max. **8** pcs. of I-8172W (Max. ch.2048 DI & 2048 DO)
XP-8xx7-CE6, W-8x47/8x37 support Max. **7** pcs. of I-8172W (Max. ch.1792 DI & 1792 DO).

*10. XP-8xx7-CE6, µPAC-5xx7, µPAC-7186EG, iP-8xx7, WP-8x47, WP-8x37, VP-25W7/23W7 and W-8xx7 supports the I-7530 (RS-232 to CAN converter) to connect to other CAN/CANopen devices.

*11. Only the XP-8xx7-CE6, WP-8x47, WP-8x37, VP-25W7/23W7 and W-8x47 supports new redundant system, the W-8x37 doesn't support it.

How to select ISaGRAF PAC

Memory considerations:

1. The I-8417, I-8817, I-8437-80, I-8837-80 , I-7188EG, μ PAC-7186EG, I-7188XG, μ PAC-5xx7, VP-2117and iP-8447 / iP-8847 has memory limitation. The ISaGRAF code size can not exceeds 64K bytes. (size of the “appli.x8m” file)

2. WP-8147, WP-8447, WP-8847, WP-5147, WP-5147-OD and VP-25W7, VP-23W7 has code size limitation of 1M bytes. The size is 16 times of the size of I-8xx7, iPAC-8447/8847, μ PAC-5xx7, μPAC-7186EG & I-7188EG/XG (XP-8xx7-CE6, XP-8xx7-Atom-CE6: 2M byte).

CPU speed considerations:

The CPU of I-8417/8817, I-7188EG and I-7188XG is 80188 (40MHz) or compatible. It is a 16-bit CPU. It is not good at doing floating point value calculation. If your application will do lots of floating point value calculation, it is better to use WP-8xx7, WP-5xx7 or VP-25W7 / VP-23W7 or XP-8xx7-CE6 or XP-8xx7-Atom-CE6 or future advanced ISaGRAF controllers. The CPU is 32-bit and its speed is about 10 to 20 times compared with the I-8xx7 & I-7188EG/XG, especially for floating point value calculation.
The speed of I-8437-80, I-8837-80, iPAC-8447/8847, μPAC-5xx7 and μPAC-7186EG are about 4-times of the I-8xx7 & I-7188EG/XG.

Redundancy considerations:

XP-8xx7-CE6, XP-8xx7-Atom-CE6 and WP-8xx7 supports redundancy solution. Two controllers to be one redundancy system. One is redundant Master, one is redndant slave. Master handles all inputs & outputs of the remote RS-485 I/O (I-7k & I-87K) at run time. If master is dead, Slave will take over the control of the remote I/O. **All Outputs** should be configured as RS-485 remote I/O. **Inputs** can locate at slot 1 through 7 or configured as RS-485 remote I/O. Redundant Change Over Time: <= 500 ms, Synchronization: <= 75ms

Ethernet considerations:

1. The WP-8xx7, WP-5xx7, XP-8xx7-CE6 and iP-8447 / iP-8847‘s ethernet is 10/100M bps (XP-8xx7-Atom -CE6 is 10/100M/1G bps) and dual ports.
   All other current ISaGRAF controllers have only one Ethernet port. μPAC-7186EG and μPAC-5xx7 ‘s Ethernet are 10/100M bps. I-7188EG , I-8437-80, I-8837-80 are 10 Mbps.
   All of above controllers support Modbus TCP/IP slave protocol.
   I-7188XG, I-8417/8817 and iP-8417 / iP-8817 don’t support Ethernet.

2. XP-8xx7-CE6, XP-8xx7-Atom -CE6, WP-8xx7, WP-5xx7 and VP-25W7 / VP-23W7, VP-2117, μPAC-5xx7, μPAC-7186EG and iP-8447/8847 support sending email with one attached file and sending / receiving user’s defined message (string) via UDP/IP or TCP/IP to PC or other devices, however I-8417/8817, I-7188EG, I-8437-80/8837-80, iP-8417 / iP-8817 no supporting them.

Windows CE Interface:
WP-8xx7, WP-5xx7, VP-25W7/23W7, XP-8xx7-Atom -CE6 and XP-8xx7-CE6 support the Windows CE Interface (The VP-25W7 has a Touch screen).
WP-8xx6, WP-5xx6, VP-25W6/23W6, XP-8xx6-Atom -CE6 and XP-8xx6-CE6 support the ISaGRAF and InduSoft software (The VP-25W6 has a Touch screen)

Modbus RTU Slave ports:
WP-8xx7, WP-5xx7, VP-25W7, VP-23W7 can support max. 5 Modbus RTU slave ports.
(Please refer to Appendix G, E and A.2 of its Getting Started Manual in the product box).

XP-8xx6-Atom -CE6, XP-8xx7-CE6 can support max. 9 Modbus RTU slave ports
I-7188EG/XG, µPAC-7186EG, µPAC-5xx7, iP-8xx7, I-8xx7 and VP-2117 can support max. 2 Modbus RTU slave ports (Please refer to its Getting Started Manual in the product box).

Chapter 1. Software & Hardware Installation

NOTE:
The I-8xx7 is the abbreviation for the I-8417, I-8437-80, I-8817 and I-8837-80 controllers.
The WP-8xx7 is the abbreviation for the WP-8147/8447/8847 and WP-8137/8437/8837 controllers.
The WP-5xx7 is the abbreviation for the WP-5147/5147-OD controllers.
The XP-8xx7-CE6 is the abbreviation for the XP-8047-CE6/ XP-8347-CE6/ XP-8747-CE6 controllers.
The XP-8xx7-Atom-CE6 is the abbreviation for the XP-8147-Atom -CE6/ XP-8347-Atom -CE6/ XP-8747-Atom CE6 controllers.
The iP-8xx7 is the abbreviation for the iP-8447/ iP-8847 controllers.

1.1: Installing The ISaGRAF Workbench Software Program

For the I-8xx7, I-7188EG/XG, µPAC-7186EG, µPAC-5xx7, iP-8xx7, VP-2117, VP-25W7 / VP-23W7, WP-8xx7, WP-5xx7, XP-8xx7-Atom -CE6 & XP-8xx7-CE6 controller system and the ISaGRAF Workbench software to operate properly, it is imperative that each is setup correctly. This chapter covers the details of how to setup the controller system and the ISaGRAF Workbench software in a minimum of time. Before you can start programming the ISaGRAF PAC system with the ISaGRAF software program, you must first install the ISaGRAF Workbench software program on a target PC.

Hardware Requirements

- A Personal Computer With At Least A Pentium, 133 MHz Or Faster Processor
- 32 MB Memory (Preferably 64 MB RAM)
- A Hard Drive With At Least 128 MB Of Storage Space (Preferably Larger)
- At Least One RS-232 Serial Port

Software Requirements

One of the following computer operating systems must be installed on the target computer system before you can install the ISaGRAF Workbench software program.

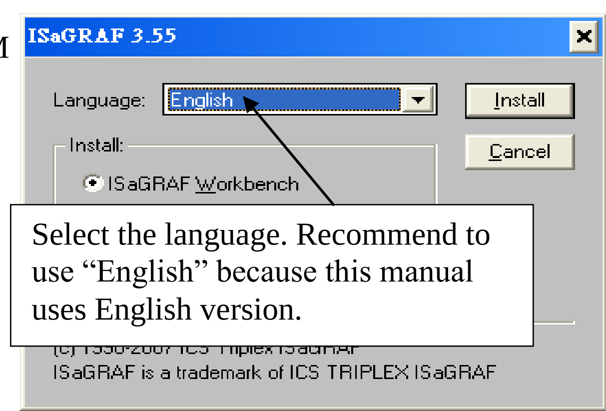
- Windows 98, Windows 2000 or Windows XP
- Windows NT Version 3.51 or Windows NT Version 4.0
- Windows Vista or Windows 7 (refer to FAQ-117)

Steps To Installing The ISaGRAF Workbench Program

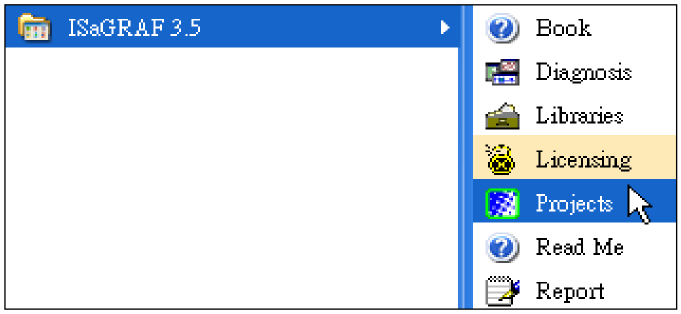
If your PC OS is Windows Vista or Windows 7 (32-bit), refer to 1.1.4.
If your PC OS is Windows 7 (64-bit), please refer to 1.1.5.

Insert the ISaGRAF Workbench CD into your CD-ROM drive. 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.



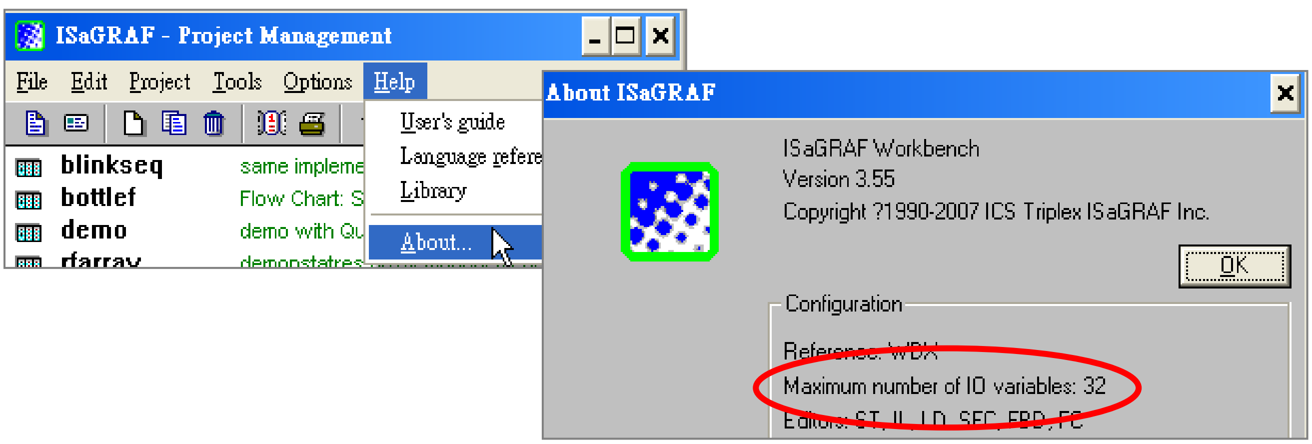
To begin the ISaGRAF 3.x software program, click on the Windows "Start" button, then on "Programs", and you should see the ISaGRAF program group as illustrated below.



1.1.1 The hardware protection device (dongle & USB Key-Pro)

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. (ISaGRAF-32-E & ISaGRAF-32-C DO NOT need dongle or USB Key-Pro.)

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 driver may not be installed well. Please do the following steps.

Dongle Protection:

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.

USB Key-Pro Protection:

1. To make your PC recognize the ISaGRAF USB protection-key, please **un-plug** the USB protection-key from your USB port first, then run “**\Sentinel\SSD5411-32bit.exe**“ in the ISaGRAF 3.55 CD-ROM (or later version) after you have installed the ISaGRAF. Then please reset your PC.
2. To run ISaGRAF Ver. 3.5x, please always plug the USB protection-key in the PC’s USB port.

1.1.2 Important Notice For Windows 2000 users

When closing my ISaGRAF window on windows 2000, it holds. Why ? This problem usually happens on the windows 2000. When you close some ISaGRAF windows by clicking on the “X”, it holds about 20 to 40 seconds (No response).

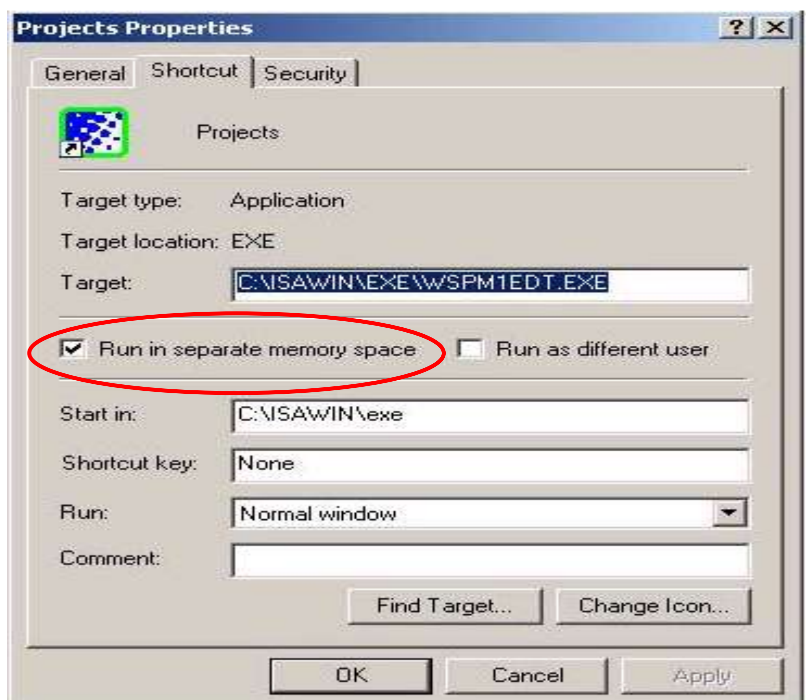
This “hold” behavior is caused by the “CTFMON.EXE” process.

We still don’t know the reason yet. You may stop this process by click on the “Ctrl” & “Alt” & “Del” at the same time to open the window Task Manager, and then stop “CTFMON.EXE” as next page.

If you want to know more about the “CTFMON.EXE”, please visit www.microsoft.com & search “CTFMON.EXE”.

One Quick way to avoid the “hold” problem on windows 2000:

You may create a short cut for the “ISaGRAF project manager. And then check on "run in separate memory space" option in the shortcut property.



1.1.3 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.
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

1.1.4 Important Notice for Windows Vista or Windows 7 (32-bit) Users

Before installing the ISaGRAF, if your operating system is Windows Vista or Windows 7 (32-bit), please change the User Account Control settings to avoid some of the setup restrictions.

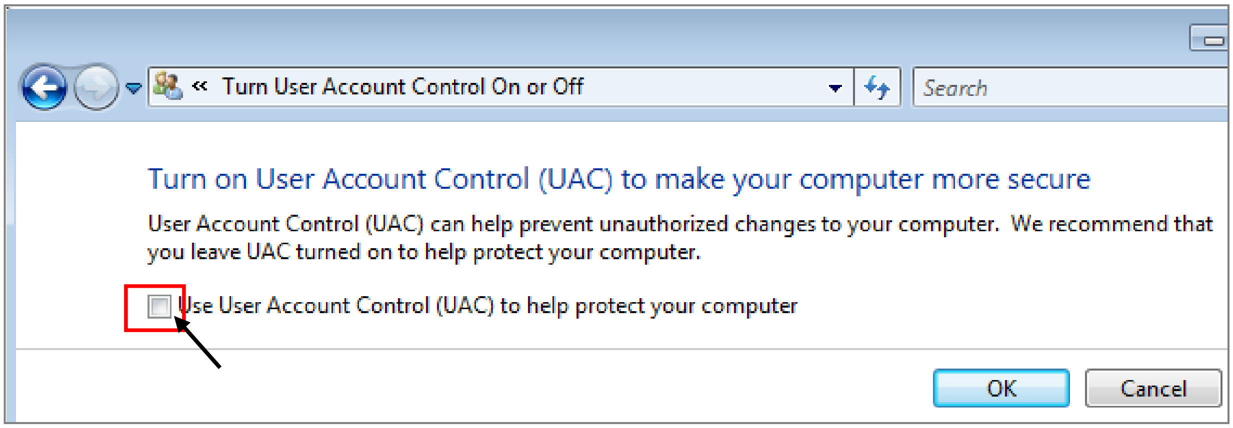
<u>How to disable "UAC" (User Account Control) ?</u>

The "UAC" (User Account Control) setting requires administrator-level permission.

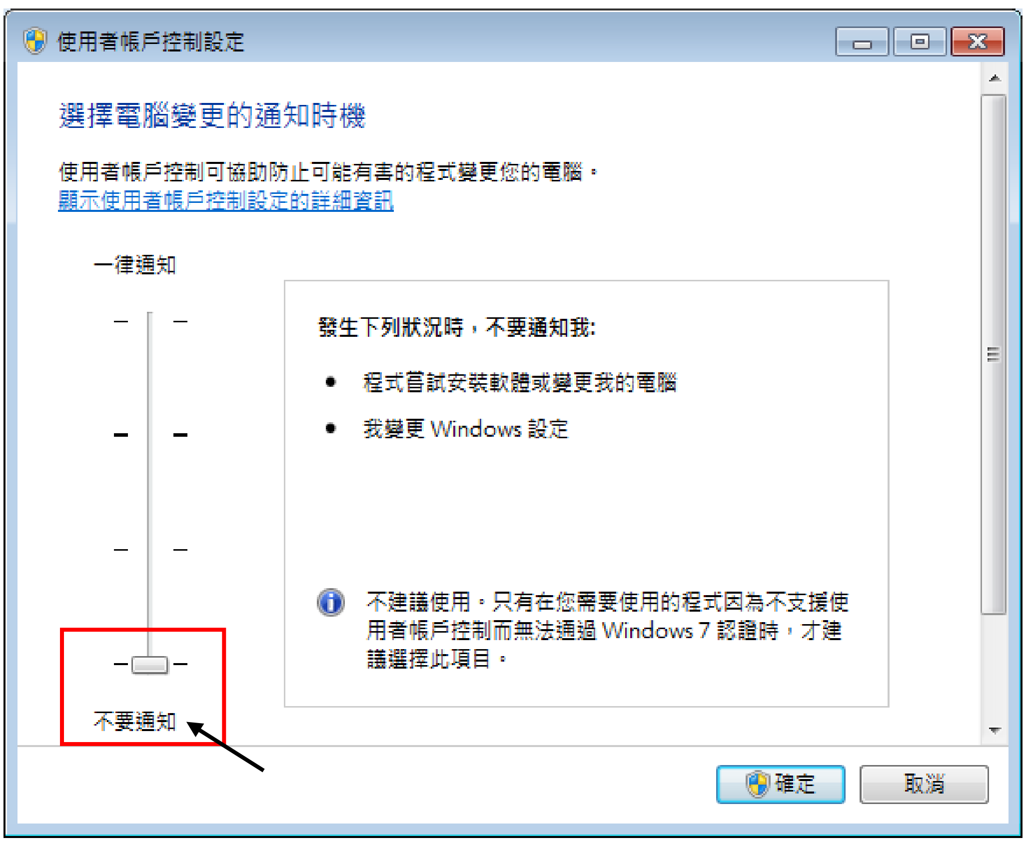
1. From the "Start" menu, choose "Control Panel > User Accounts and Family Safety > User Accounts", then click "Change User Account Control settings" or "Turn User Account Control on or off".

2. After clicking, it will show up the screen as below.

Windows Vista:
Uncheck the option – "Use User Account Control (UAC) to help you protect your computer" and then click on "OK".



Windows 7:
Move the slider down to "Never Notify" and then click on "OK".



3. Reboot your computer to apply the change.
4. After rebooting, please refer to section 1.1 Installing the ISaGRAF.

1.1.5 Important Notice for Windows 7 (64-bit) Users

If your operating system is Windows 7 (64-bit) Professional, Enterprise, or Ultimate, the ISaGRAF must be installed under the XP Mode. Please do the following steps to install Virtual PC and XP Mode.

Installing the Virtual PC and XP Mode:

1. Download Windows Virtual PC and Windows XP Mode installers from the Windows Virtual PC Web site (http://go.microsoft.com/fwlink/?LinkID=160479)
2. Double-click on "WindowsXPMode_nn-NN.exe" (where nn-NN is the locale, e.g. en-US) and follow the instructions in the wizard to install Windows XP Mode.
3. Double-click on "Windows6.1-KB958559-x64.msu" to install Windows Virtual PC.
4. Reboot your computer.
5. After rebooting, click on "Star > All Programs > Windows Virtual PC″ and then click Windows XP Mode.
6. Follow the instructions in the wizard to complete Windows XP Mode Setup and Configuration. Record the password that is provided during the Setup because it is required to log on to your virtual machine.
7. Now, go back to section 2.1 to install the ISaGRAF.

1.1.6 Important Setting for Using Variable Arrays

Important setting for using variable arrays:

Please add two lines on the top of the c:\isawin\ese\isa.ini file to enable the usage of variable arrays.

[DEBUG]
Arrays=1

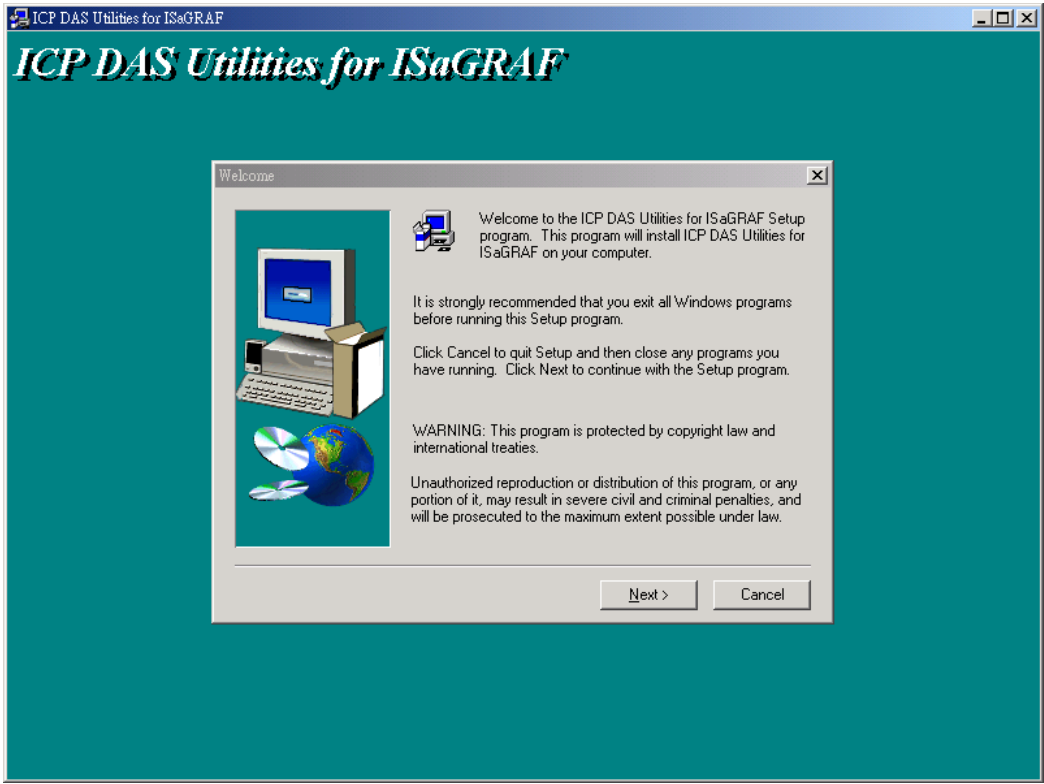
1.2: Installing The ICP DAS Utilities For ISaGRAF

The "ICP DAS Utilities For ISaGRAF" consists of 3 major items.

I/O library definition (For ISaGRAF PAC)
Modem_Link utility
Auto-scan I/O utility

The ISaGRAF Workbench software program must be installed before attempting to install the "ICP DAS Utilities for ISaGRAF". 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 ISaGRAF controllers with the "ICP DAS Utilities for ISaGRAF". Please insert the CD-ROM into your CD-ROM drive. Then run "setup.exe" in the folder of CD-ROM: \napdos\isagraf\ . Follow the steps to install it.



Note:
If "setup.exe" is not in your CD-ROM, please download "**ICP DAS Utilities For ISaGRAF**" from http://www.icpdas.com/products/PAC/i-8000/isagraf.htm

1.3: Hardware Setting on ISaGRAF PAC

All the hardware setting on the ISaGRAF PAC, such as IP address, Net-ID, Modbus RTU slave port, pin assignment of communication port, etc. Please refer to each ISaGRAF getting started manual which can be found in the accompanying CD_ROM of the shipping box.

WP-8xx7 CD:\napdos\isagraf\wp-8xx7\english_manu\getting_started_wp-8xx7.pdf
WP-5xx7 CD:\napdos\isagraf\wp-5xx7\english_manu\wp-5xx7_manual.pdf
XP-8xx7-CE6 CD: \napdos\isagraf\xp-8xx7-ce6\english_manu\getting-started-xp-8xx7-ce6-english.pdf
VP-2xW7 CD: \napdos\isagraf\vp-25w7-23w7\english_manu\getting-started-vp-2xW7.pdf
iP-8xx7 CD: \napdos\isagraf\ip8000\english_manu\ipac-8x47_getting_started_english.zip
μPAC-7186EG, I-7188EG/XG CD:
\napdos\isagraf\7188eg\english_manu\718xegxg_getting_started_english.zip
VP-2117 CD: \napdos\isagraf\vp2k\english_manu \vp-2117_getting_started_english.zip

or visit to http://www.icpdas.com/products/PAC/i-8000/isagraf.htm and click "Manual " icon to download the Manual.

Chapter 2. Getting Started

NOTE:
The I-8xx7 is the abbreviation for the I-8417, I-8437-80, I-8817 and I-8837-80 controllers.
The WP-8xx7 is the abbreviation for the WP-8147/8447/8847 and WP-8137/8437/8837 controllers.
The WP-5xx7 is the abbreviation for the WP-5147/5147-OD controllers.
The XP-8xx7-CE6 is the abbreviation for the XP-8047-CE6/ XP-8347-CE6/ XP-8747-CE6 controllers.
The XP-8xx7-Atom-CE6 is the abbreviation for the XP-8147-Atom -CE6/ XP-8347-Atom -CE6/ XP-8747-Atom CE6 controllers.
The iP-8xx7 is the abbreviation for the iP-8447/ iP-8847 controllers.

2.1: A Simple Ladder Logic (LD) Program

For more extensive information regarding all of the capabilities of the ISaGRAF programming system, please refer to **Appendix E: "Language Reference"** of this manual or 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".

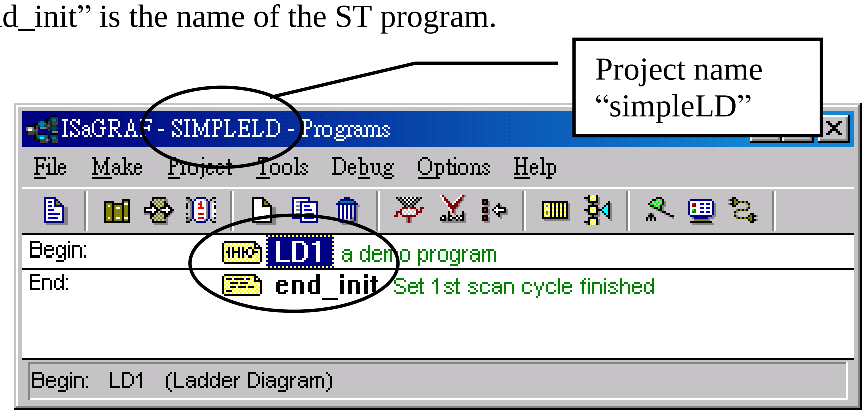
Ladder Logic Basics
"Ladder Logic" programming (LD) is a graphical representation of Boolean equations, combining **contacts** (input arguments) and **coils** (output results). Ladder Logic most closely resembles the electrical schematics that an electrician or technician may use to diagnose and troubleshoot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

We will create one another Structured Text (hence forth referred as "ST") program to indicate the first PLC scan cycle. That means in this example ISaGRAF project, we have two programs inside it. One is written in LD and the other is written in ST.

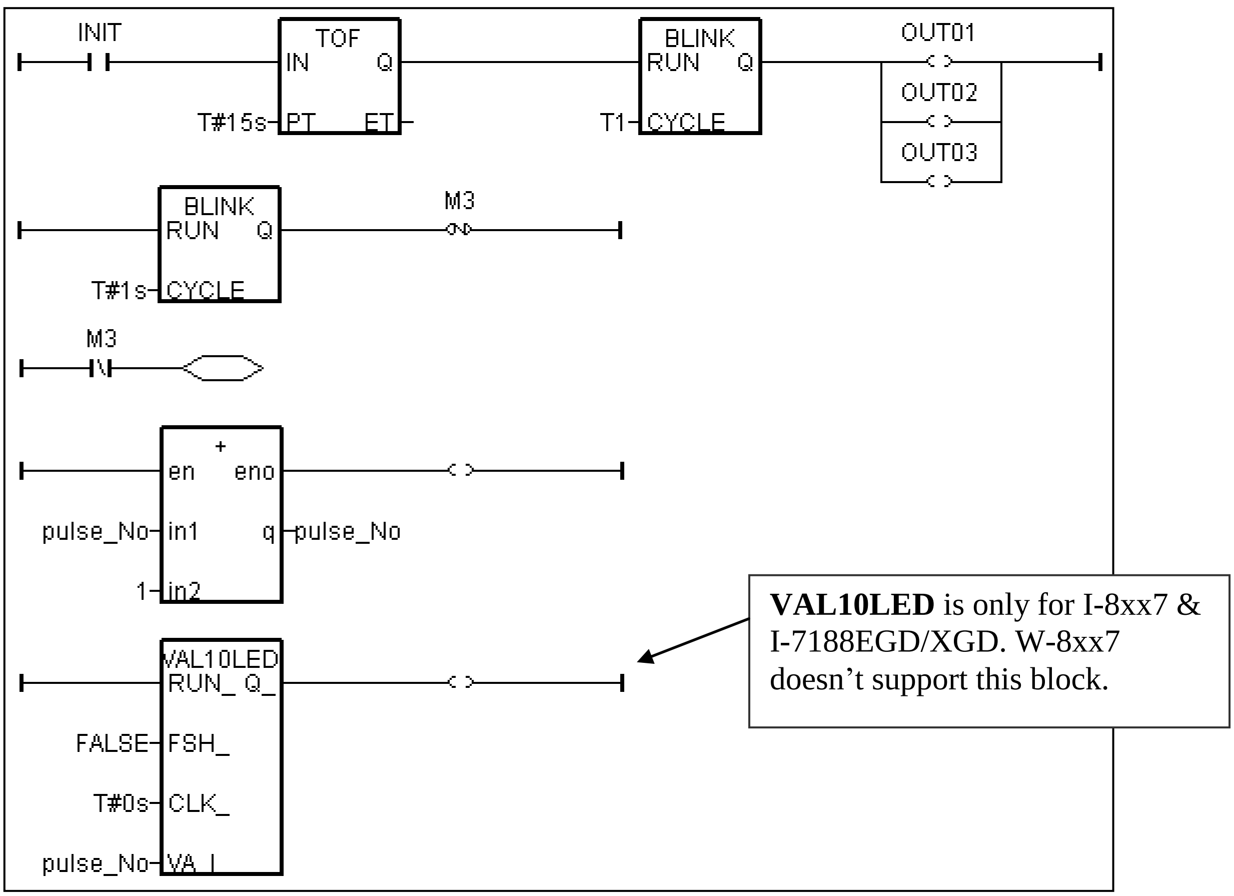
The example project name is "simpleLD". The name of the LD program of this example project is "LD1" and "end_init" is the name of the ST program.



Variables Used In The Example LD Program:

Name	Type	Attribute	Description
INIT	Boolean	**Internal**	**Initial value at “TRUE”**. TRUE means 1st scan cycle
M3	Boolean	**Internal**	Indicate a pulse is generated or not.
OUT01	Boolean	**Output**	Output 1
OUT02	Boolean	**Output**	Output 2
OUT03	Boolean	**Output**	Output 3
T1	Timer	**Internal**	Time Period of blinking, **initial value is set at "T#1s"**
Pulse_No	Integer	**Internal**	To puls one when M3 pulse is generated **initial value is set at "0"**

Ladder Logic Program “LD1” Outline:



ST program “end_init” Outline:

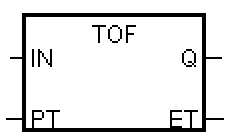
```
INIT := FALSE ;
```

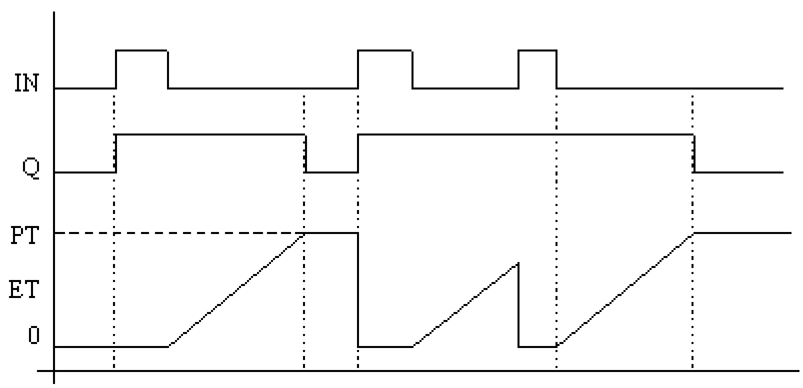
Process Operation Actions:

Ladder Logic Program "LD1" :
Blink Outputs 1, 2, & 3 with a period of "T1" in the first 15 seconds, "T1" has initial value equal to 1 second. Atfer these 15 seconds, Outputs 1, 2, & 3 will be turned OFF.
Generate a pulse output every 1 second to the internal boolean variable "M3".
Plus integer variable "pulse_No" by 1 every time when "M3" pulse is generated.
Display the value of "pulse_No" to the 7-Seg leds of the I-8xx7 or I-7188EG/XG controller.

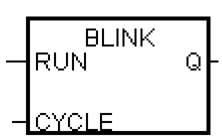
ST Program "end_init" :
Set boolean variable "INIT" to FALSE at the end of the PLC scan cycle. So that "INIT" will be TRUE only at the first scan cycle.

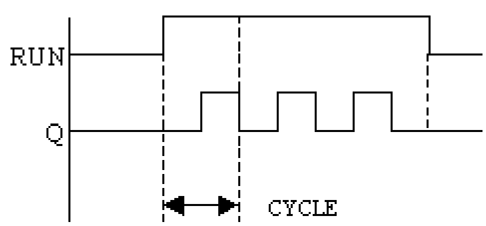
Description of block and some basic LD item:
TOF: To turn off a boolean however delay a time of "PT".
"IN" is a boolean parameter, if falling from TRUE to FALSE. The timer ticks from 0 to "PT".
"PT" is a timer parameter, it defines the delay time of output.
"Q" is the boolean output of this block. It will be turned OFF when "PT" is reached.
"ET" is the timer output of this block. (We don't use it in this example)



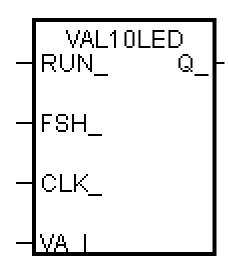


BLINK: To blink a boolean with a period of "CYCLE".
"RUN" is a boolean parameter, if it is TRUE, the boolean output "Q" will be blinking at period of the timer parameter "CYCLE".

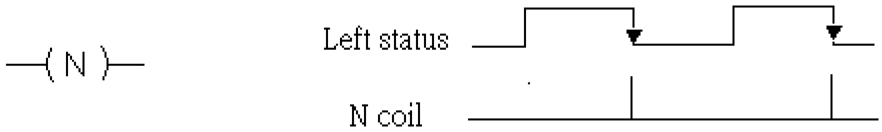




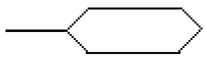
VAL10LED: Display an interger value to the 7-Seg leds of the controller.
Only for I-8xx7, I-7188EGD & I-7188XGD.
"RUN_" is a boolean parameter. TRUE to display.
"FSH_" is a boolean parameter. TRUE to blink the display.
"CLK_" is a timer parameter. It defines the blinking period.
"VA_I_" is the integer to display.



“N” coil: Coil with N type means it will be set to a pulse TRUE when the left status is just falling from TRUE to FALSE.



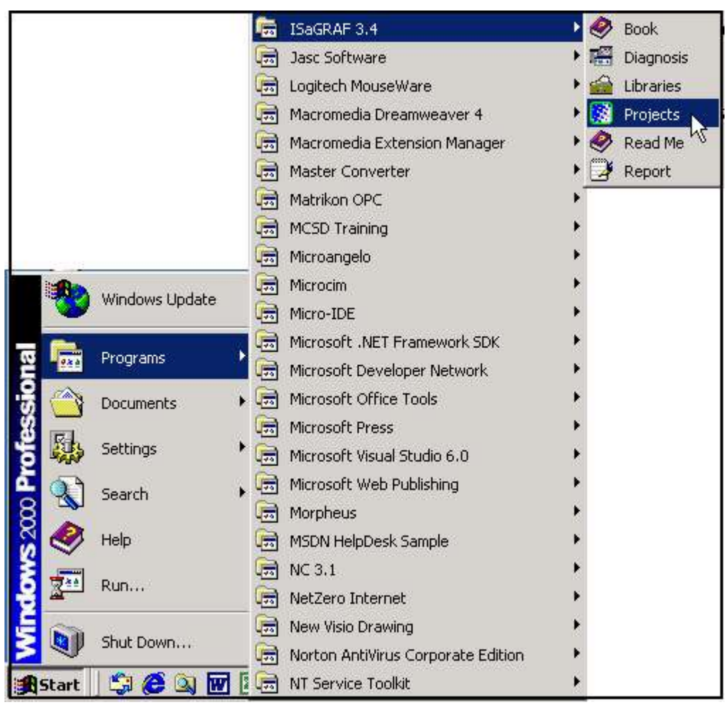
“Retrun”: To return from the excution if the left status is TRUE, that is, the reset LD rungs of the program below this “return” will not excute when the left status is TRUE.



2.1.1: Programming LD

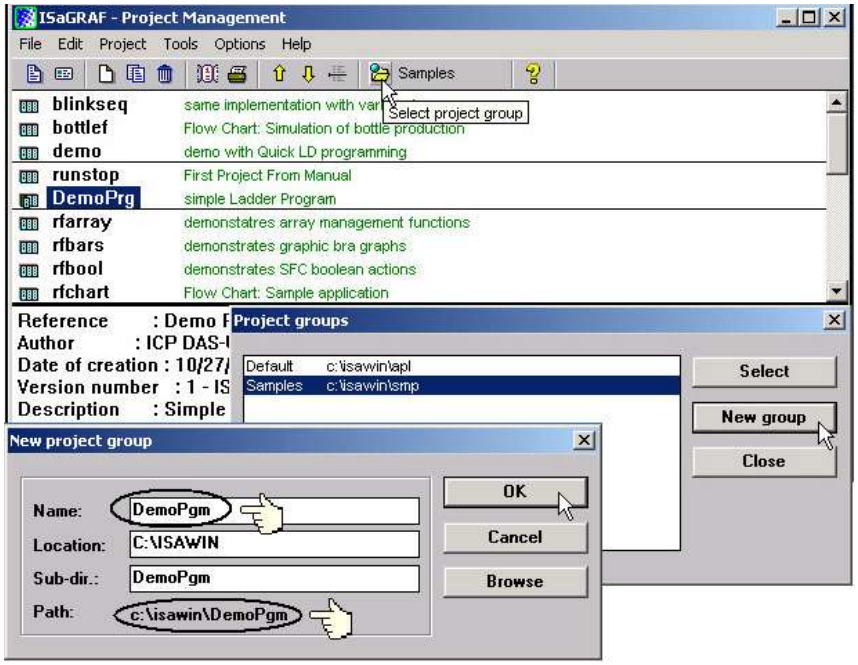
Starting & Running The ISaGRAF Workbench Program.

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 Project 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".

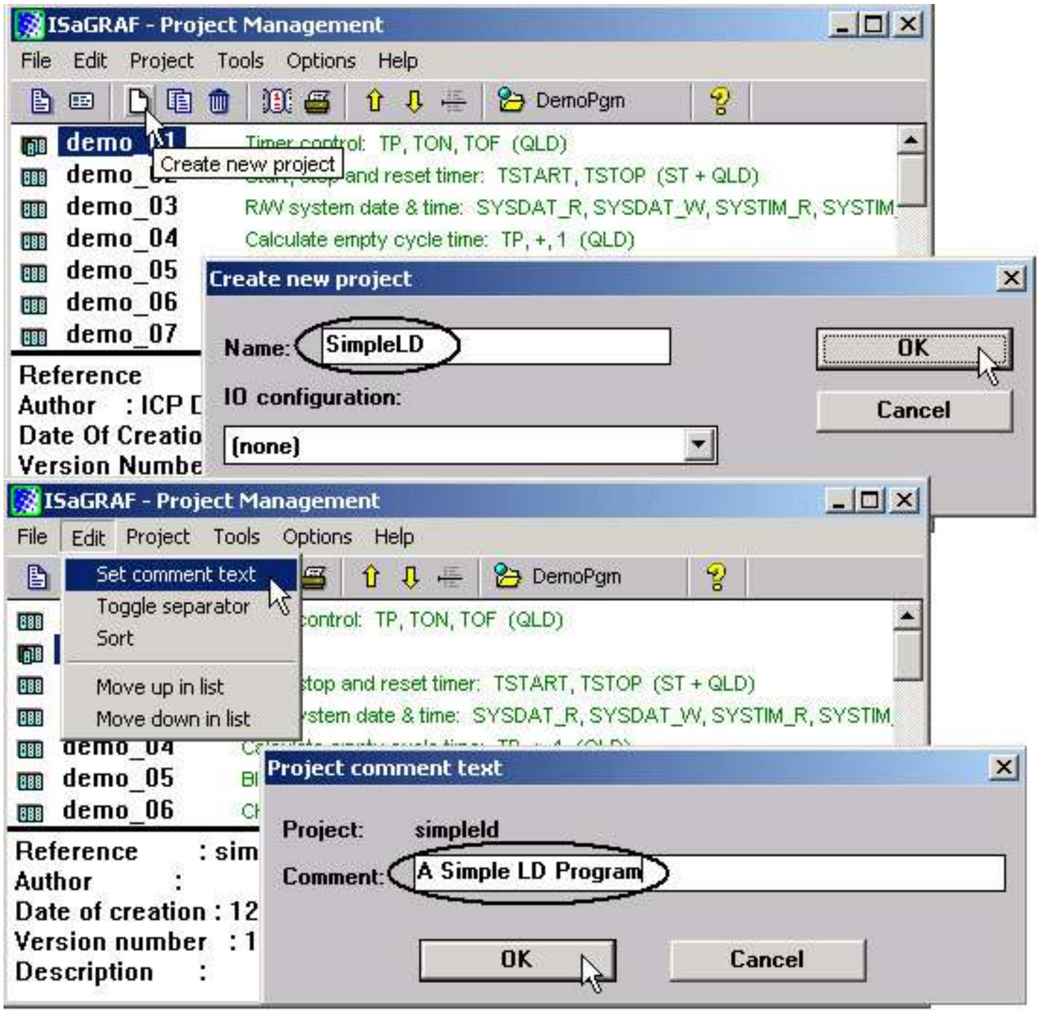


Note that the name that you give the "New Project Group" also creates a new sub-directory corresponding to the project group name in the "c:\isawin" sub-directory.

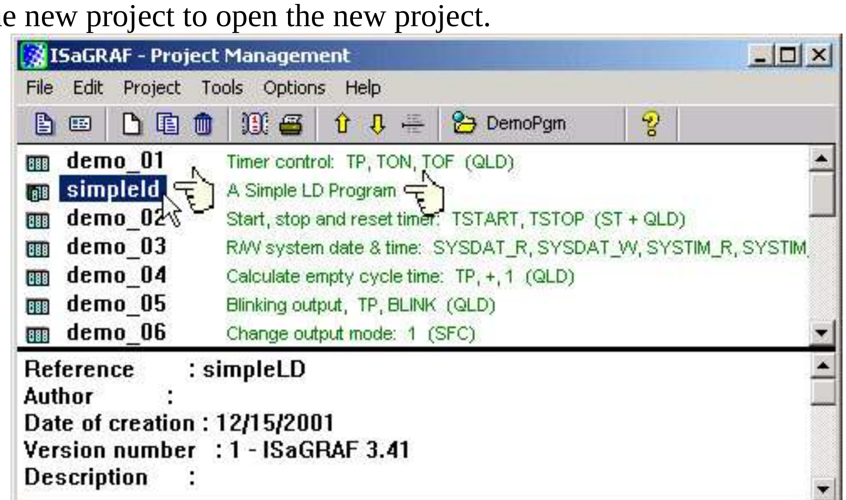
To get into the new project group, either double click on the new group name, or click on the new group name (the name will be highlighted) to select the new project group and click on the "Select" button.

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.

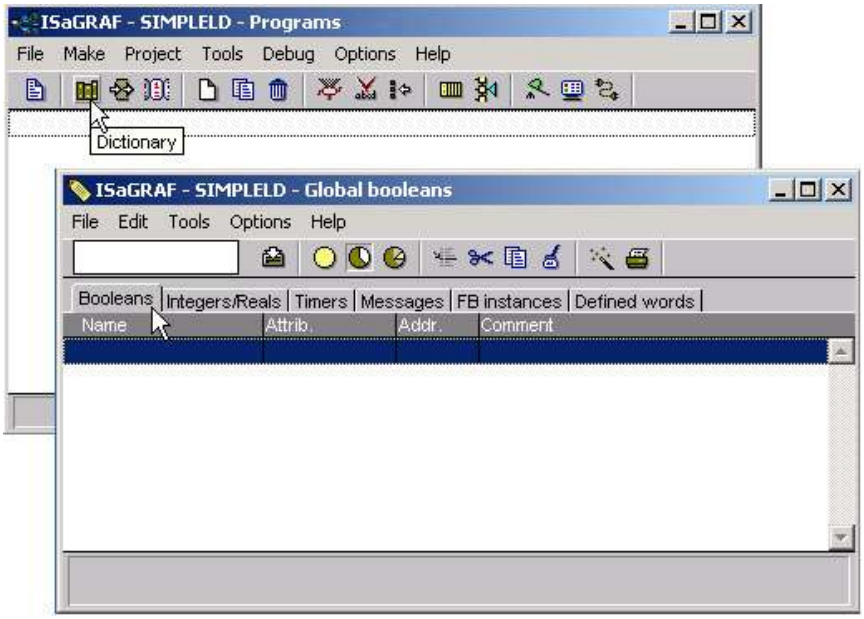


You will now see the name of the new project in the "Project Management" window. Double click on the name of the new project to open the new project.

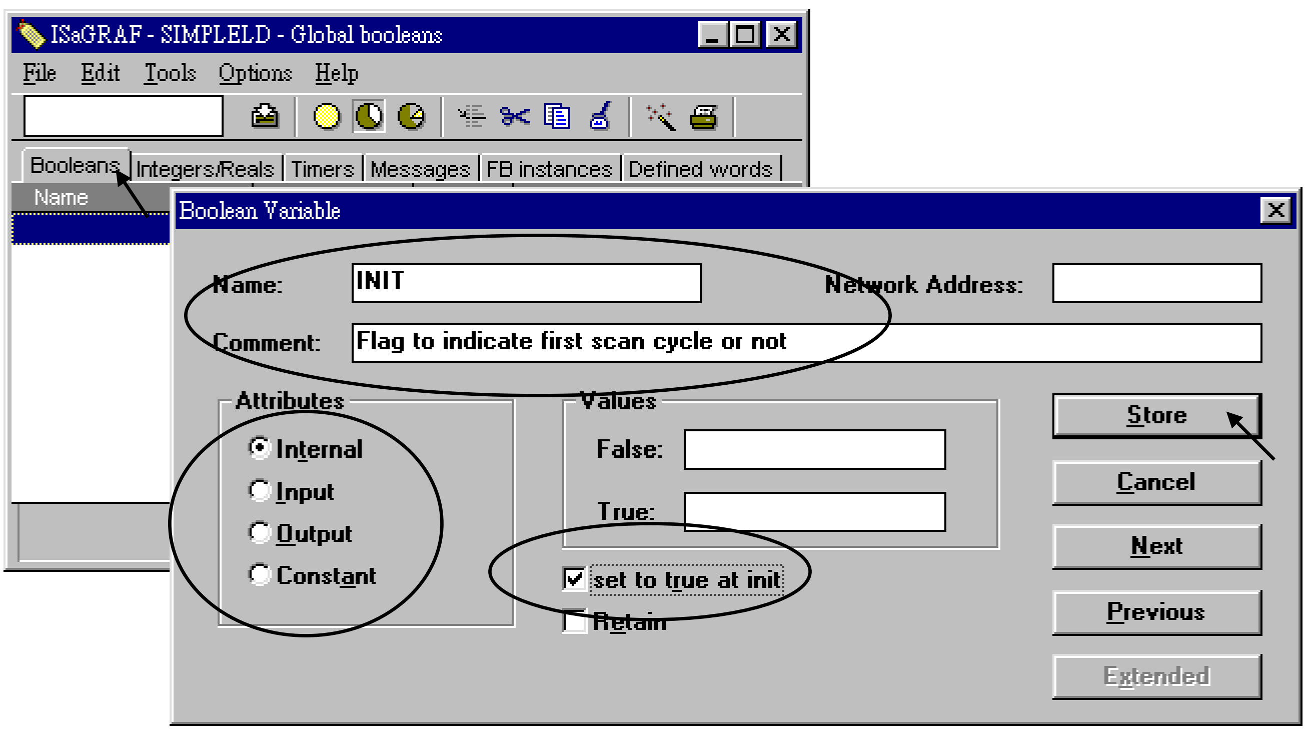


2.1.1.3: Declaring The ISaGRAF Project Variables

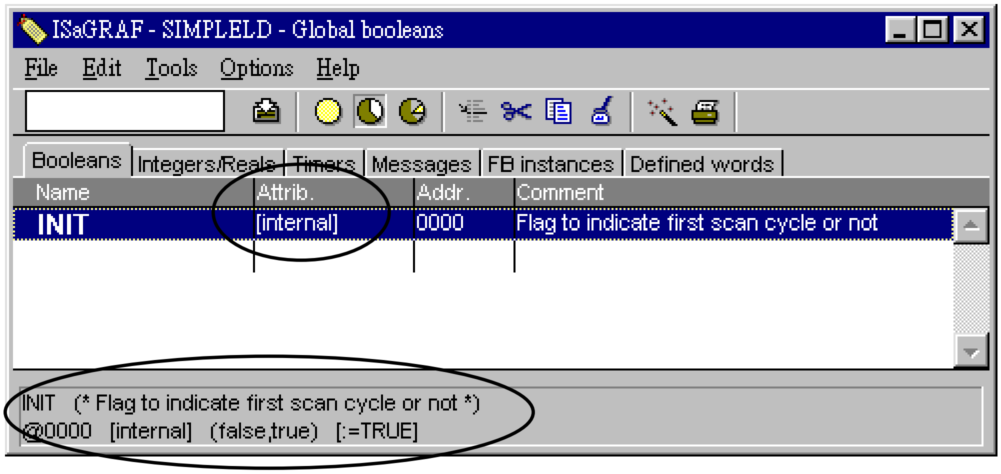
Before you can start creating an ISaGRAF program, you must first declare the variables that will be used in the ISaGRAF program. To begin this process, first click on the "Dictionary" icon and then click on the "Boolean" tab to declare the Boolean variables that will be used in our example program.



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 "INIT", and "Flag to indicate first scan cycle or not" 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, INIT's attribute will be an "Internal". Lastly, check on the "set to true at init" since we need INIT has its initial value as TRUE when the project is just power up to run. Then press the "Store" button to save the Boolean variable that has been created.

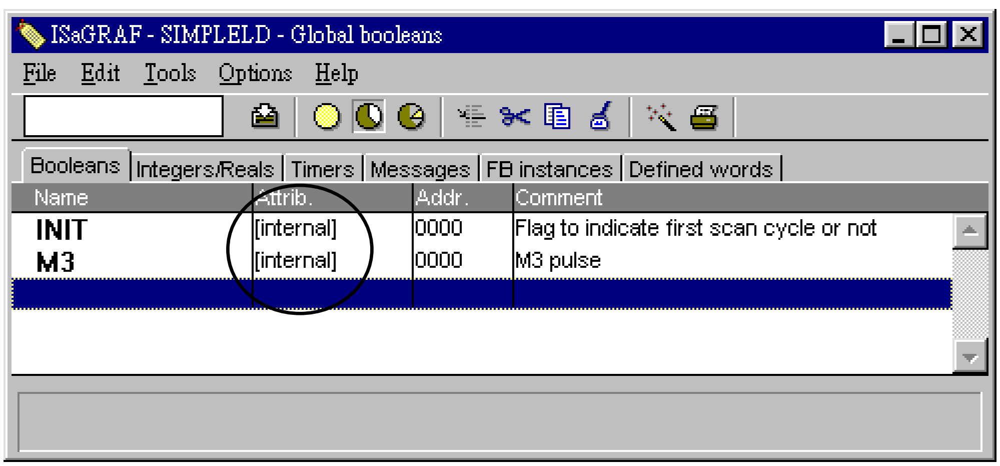


The new Boolean variable has now been declared. Note the other information areas that are provided for the programmer to fully explain how the variable will be handled.

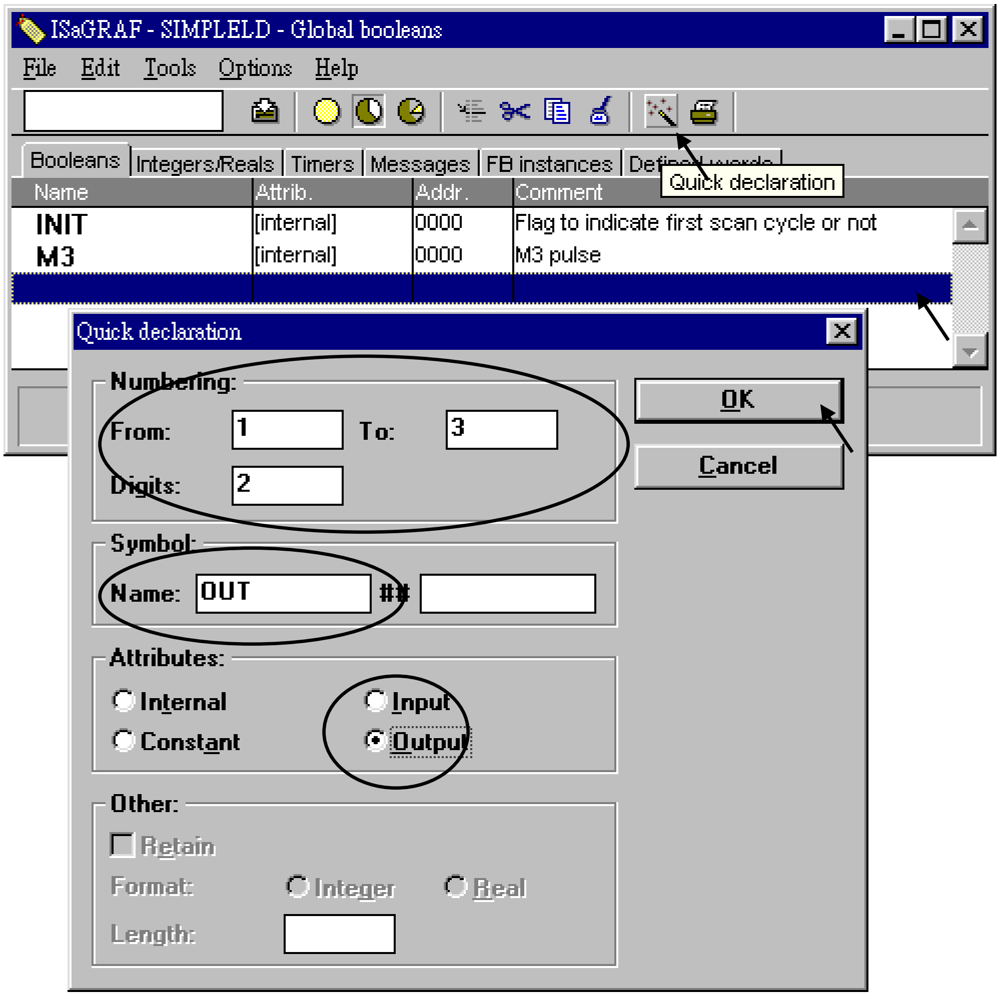


NOTE: You MUST make sure that the variable you have declared has the desired Attribute assigned. If you decide that you want to change a project variable's attribute, just double click on the variable name and you can reassign the attribute for the variable.

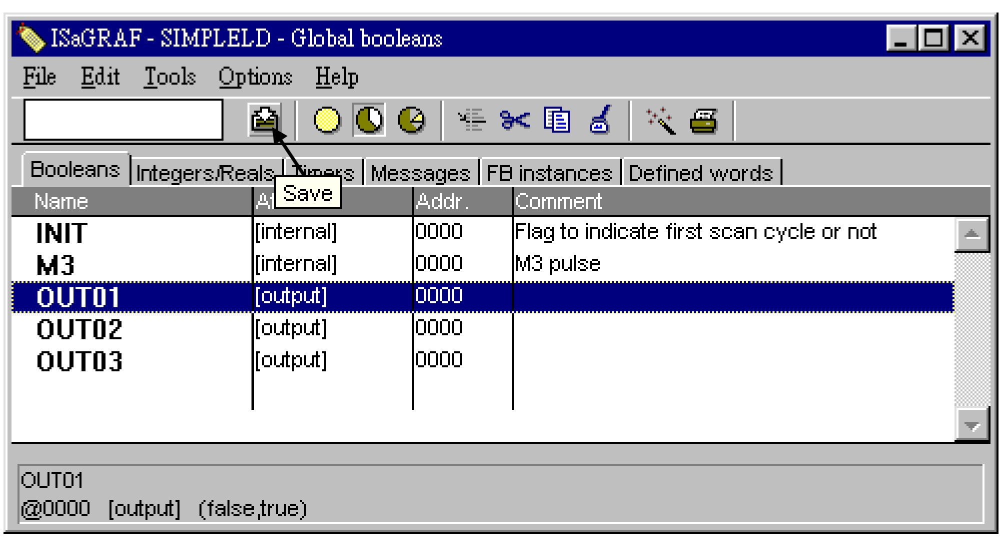
Using the same method described above, declare the additional Boolean variables for this example program, "M3". When you have completed the Boolean variable assignments, the Global Boolean window should look like the example below.



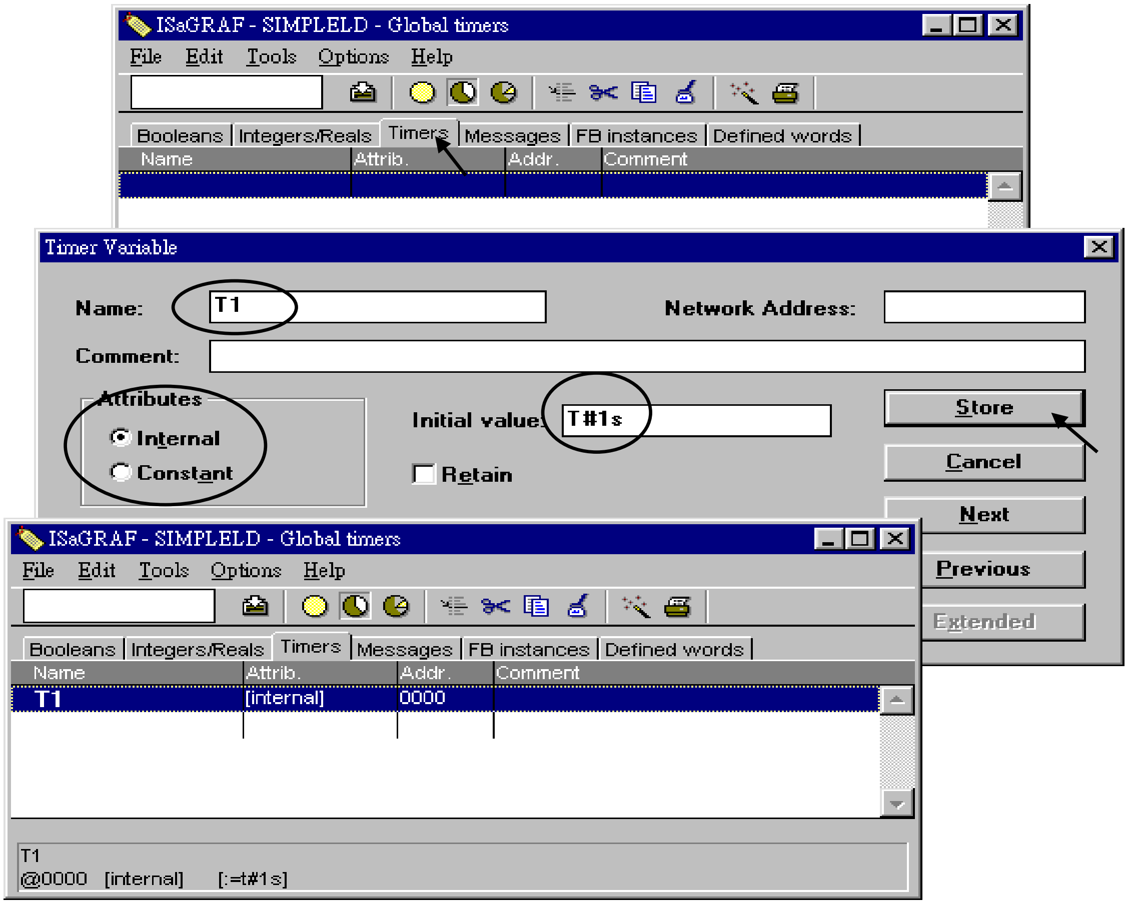
There are three outputs used in this example program named "OUT01, OUT02, and OUT03". ISaGRAF provides a quick and easy way to declare like variables that are sequentially ordered. To begin this process, click on the "Quick Declaration" icon, and enter in the output number that you will start with in 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".



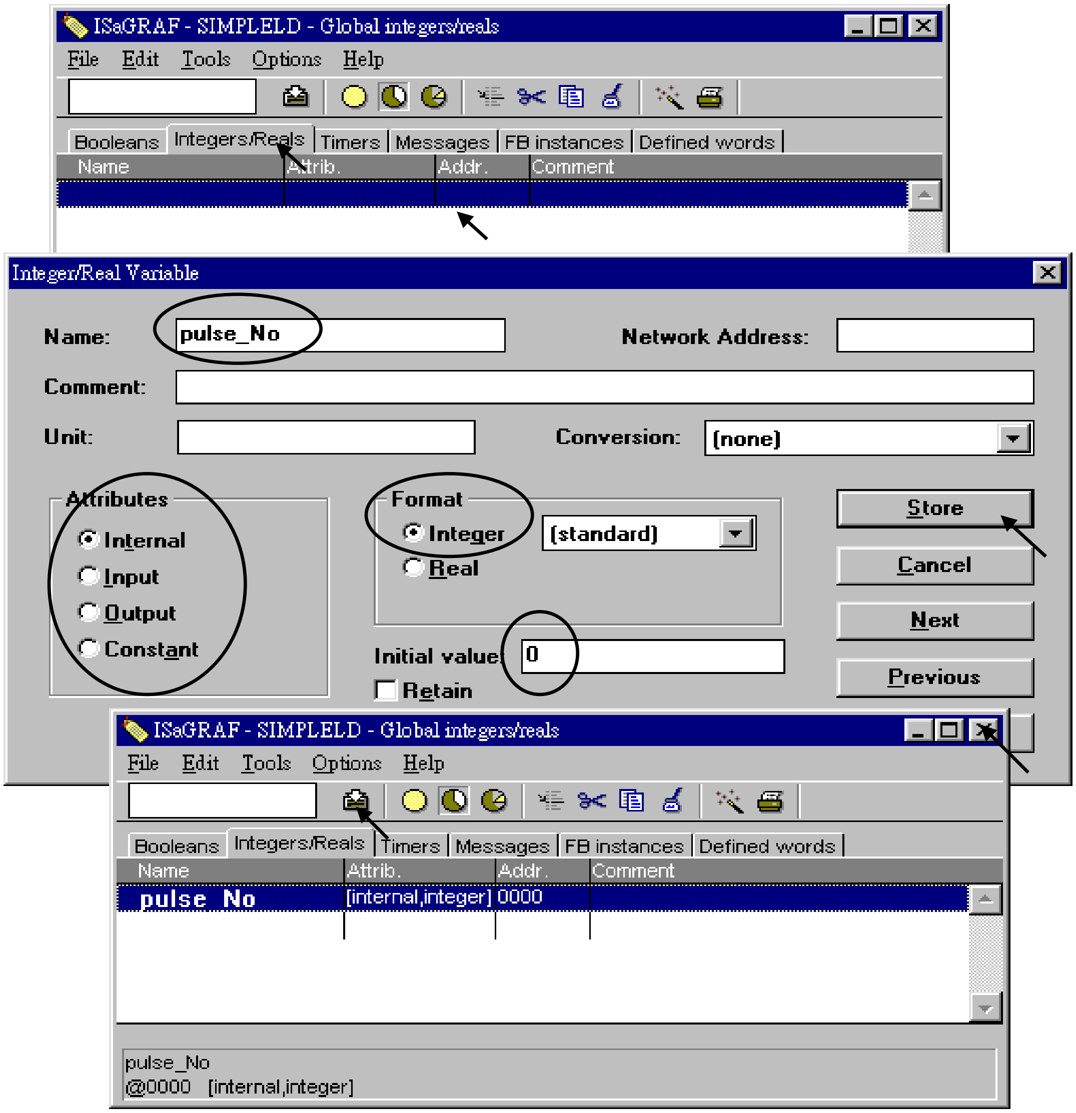
When you click on the "OK" button, all three outputs will be immediately added to the "Global Boolean" window. Lastly, click “Save” to save the settings.



To declare the timer (T1) variable used in this example program, click on the "Timers" tab in the setup screen. Double click on the colored area and enter the Name as "T1", set the "Attributes" to "Internal", the "Initial Value" to "T#1s", then click on the "Store" button.



To declare the Integer (pulse_No) variable used in this example program, click on the "Integers/Reals" tab in the setup screen. Double click on the colored area and enter the Name as "pulse_No", set the "Attributes" to "Internal", the “Format” to “Integer”, and the "Initial Value" to "0", then click on the "Store" button.

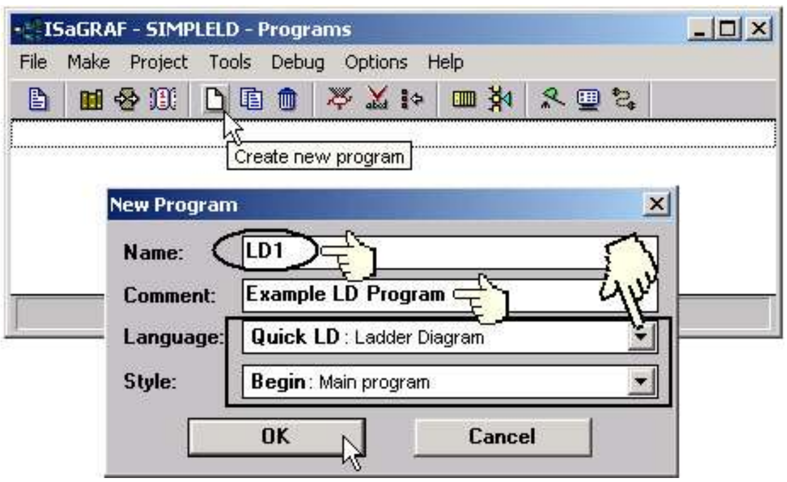


Once all of the variable characteristics have been properly setup, click on “save” and then click on "X" at the top right of the setup window to close the variable dictionary for this example project.

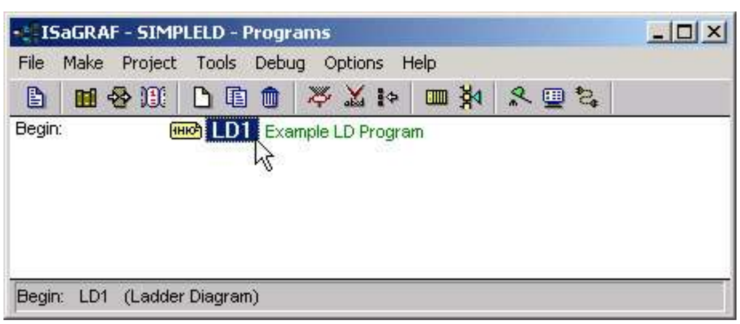
2.1.1.4: Creating The Example LD Program

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.

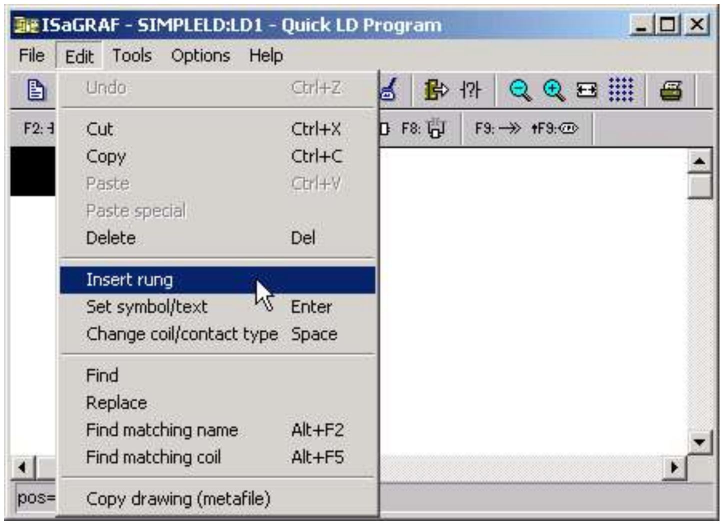


The "LD1" program has now been created. To open the "LD1" program, double click on the "LD1" name.

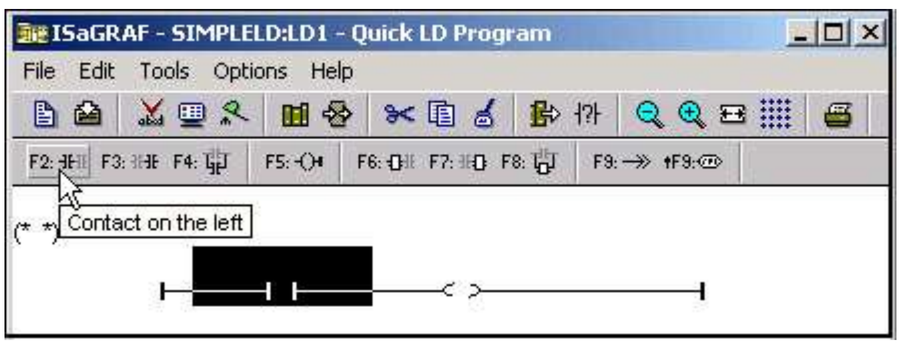


2.1.1.5: Editing The Example "LD1" Program

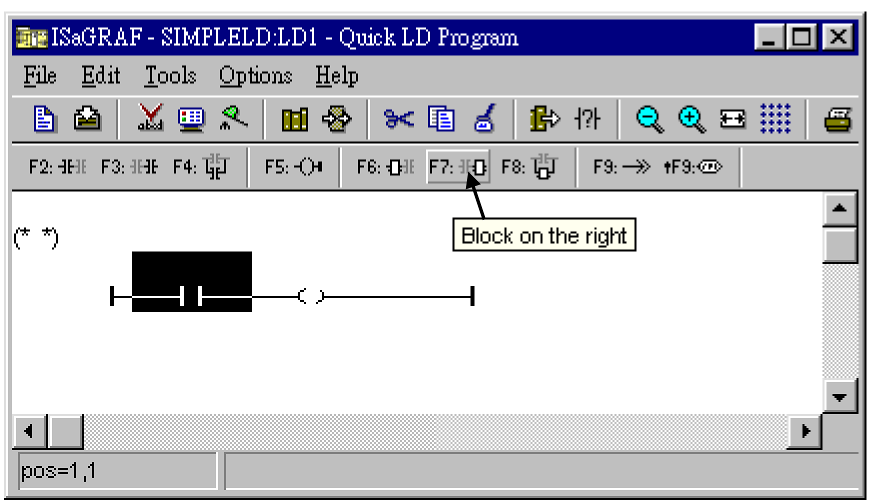
When you double click on the "LD1" name the "Quick LD Program" window will appear. Click on "Edit" from the main menu bar and then click on "Insert Rung" as shown below.



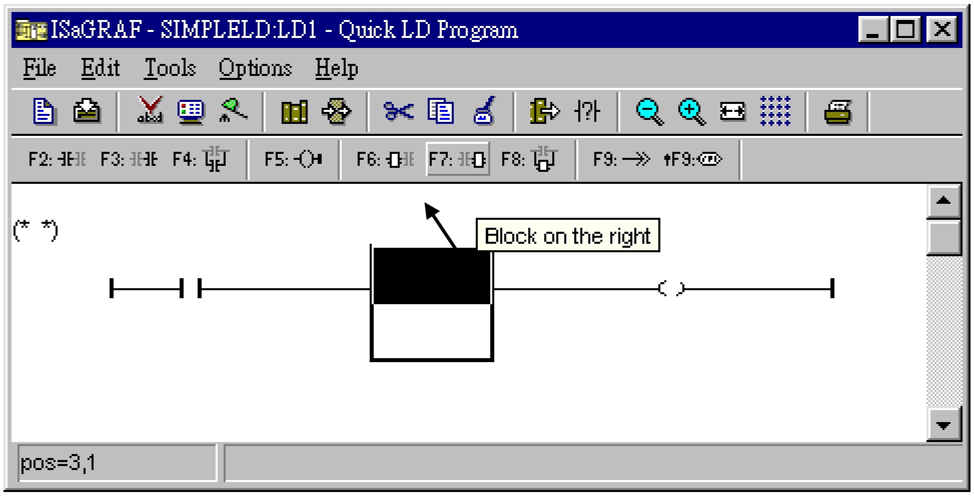
Or, you may just simply click on the "F2 (Contact On The Left)" icon, and the following will appear within the Quick LD Program window.



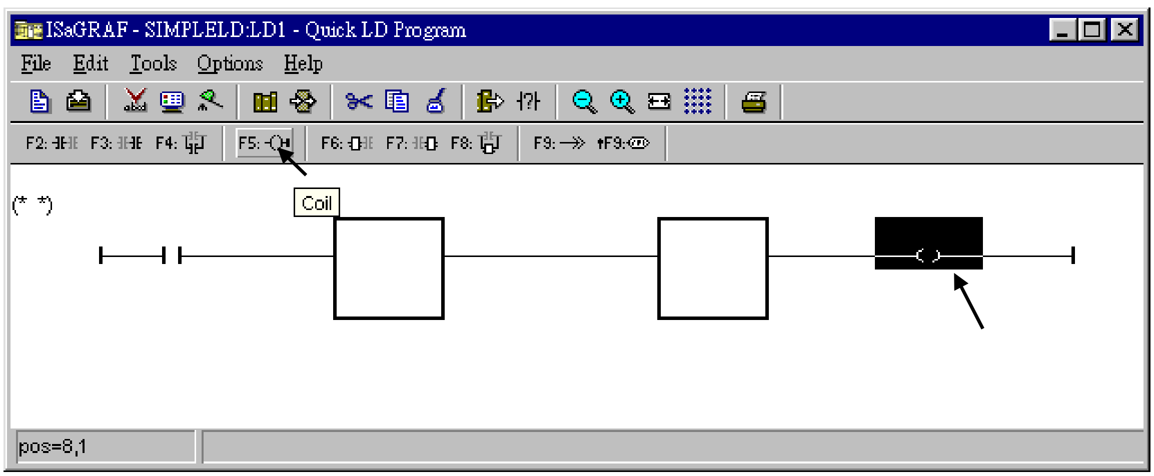
Click on the "F7 (Block on the right)" icon and you will create a block on the right of the first input contact.



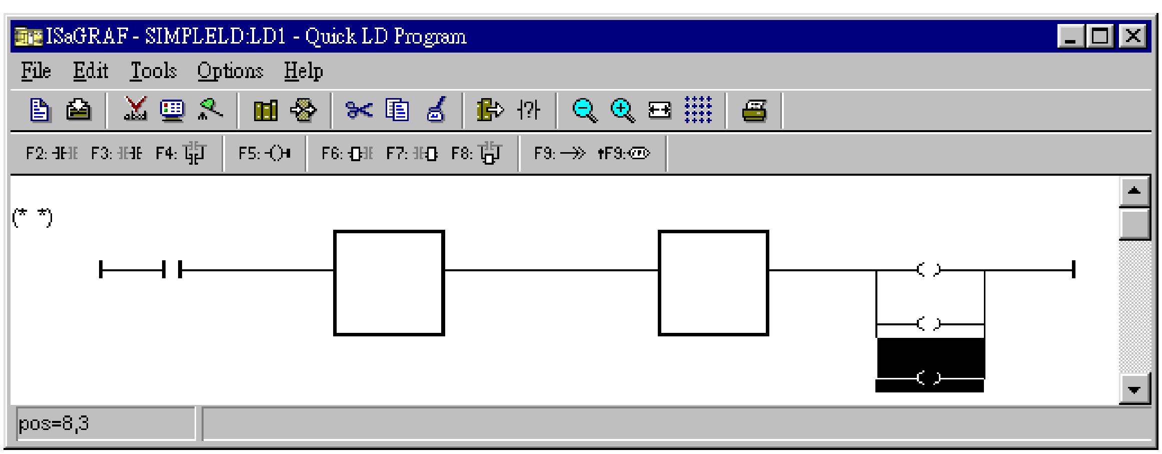
Click on "F7 (Block on the right)" icon again to create one another block on the right of the first block.



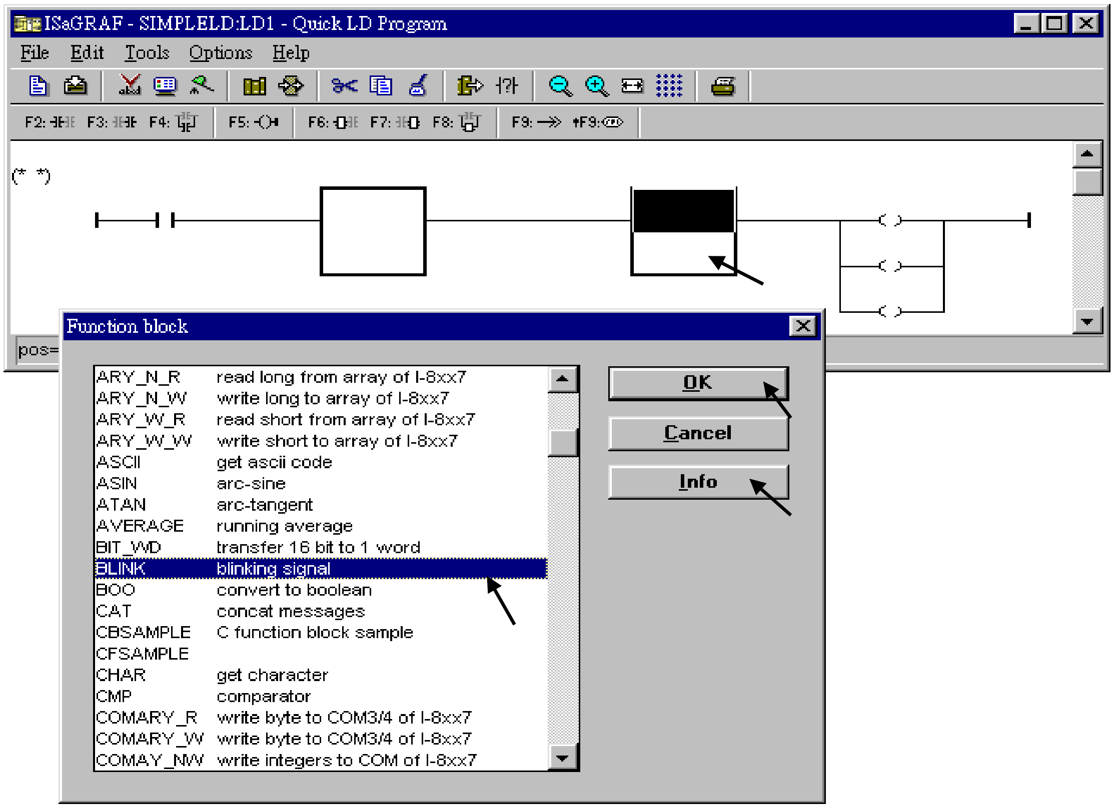
Then you will get the window as below. Move the cursor to the Coil on the right. Then click on "F5 (Coil)" to add one coil just below the first coil. And then click on "F5 (Coil)" again to add the third coil.



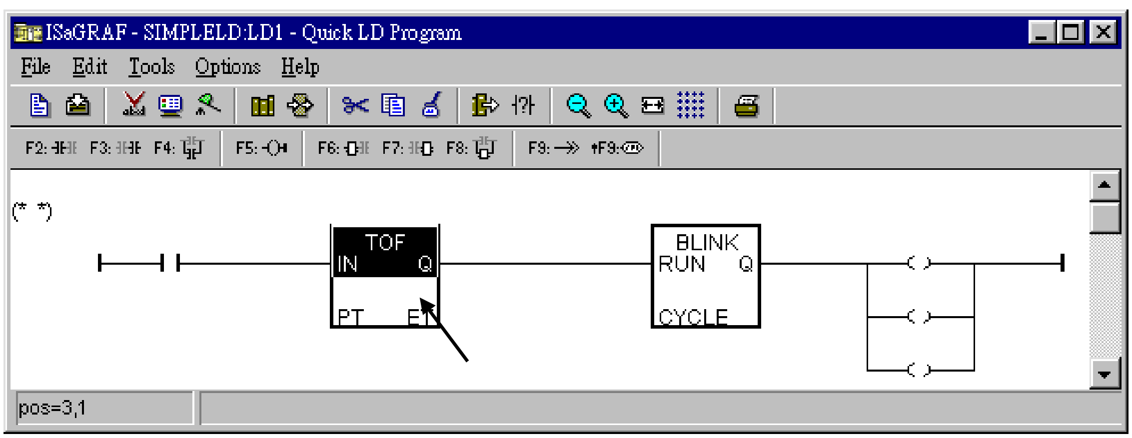
Then the window will look like below.



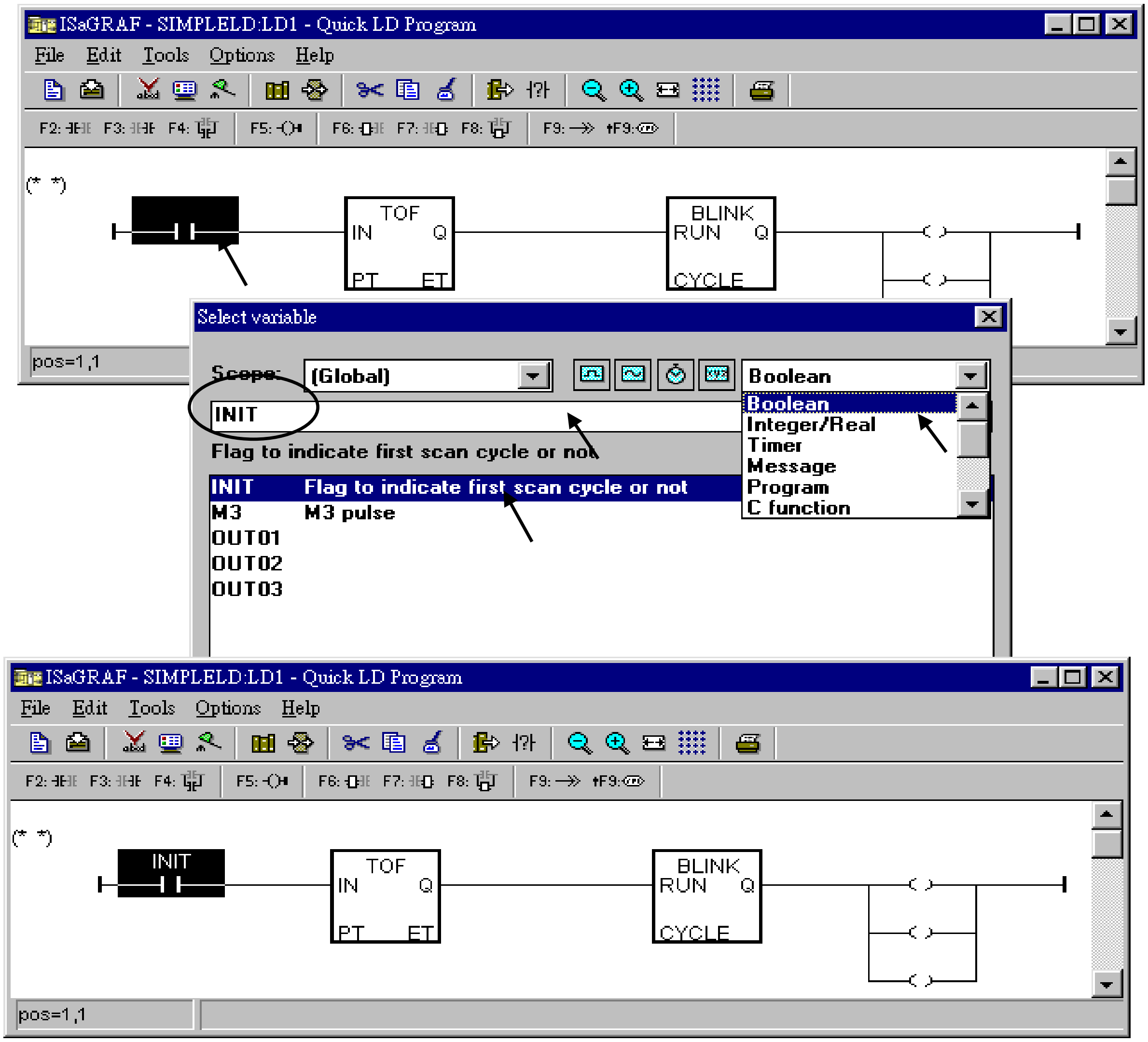
Double click anywhere inside of the second block and the "Function Block" assignment window appears. Select the "BLINK" type function block are using in our example program. To learn how the "BLINK" function operates you can click on the "Info" button for a detailed explanation of its functionality.



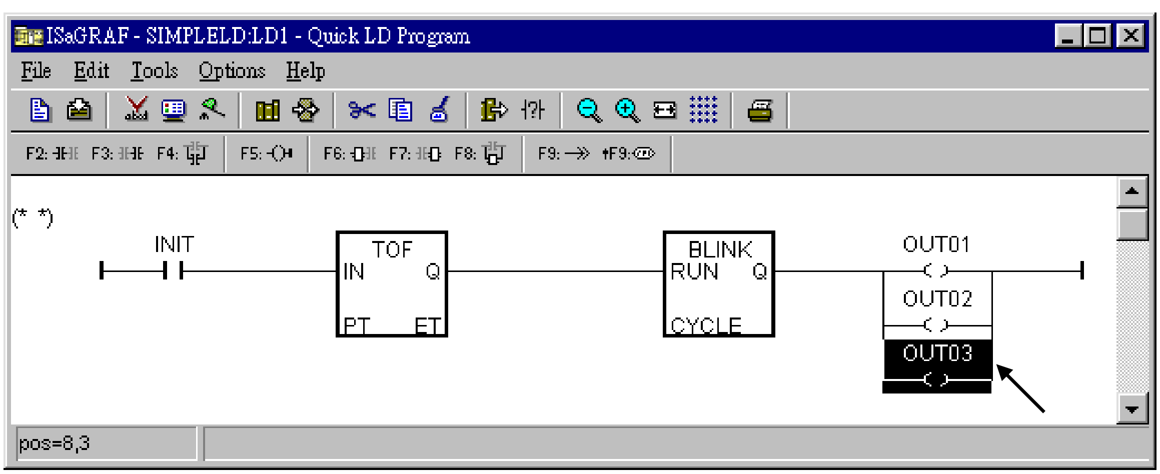
Using the same procedure to assign the first block to “TOF” as below.



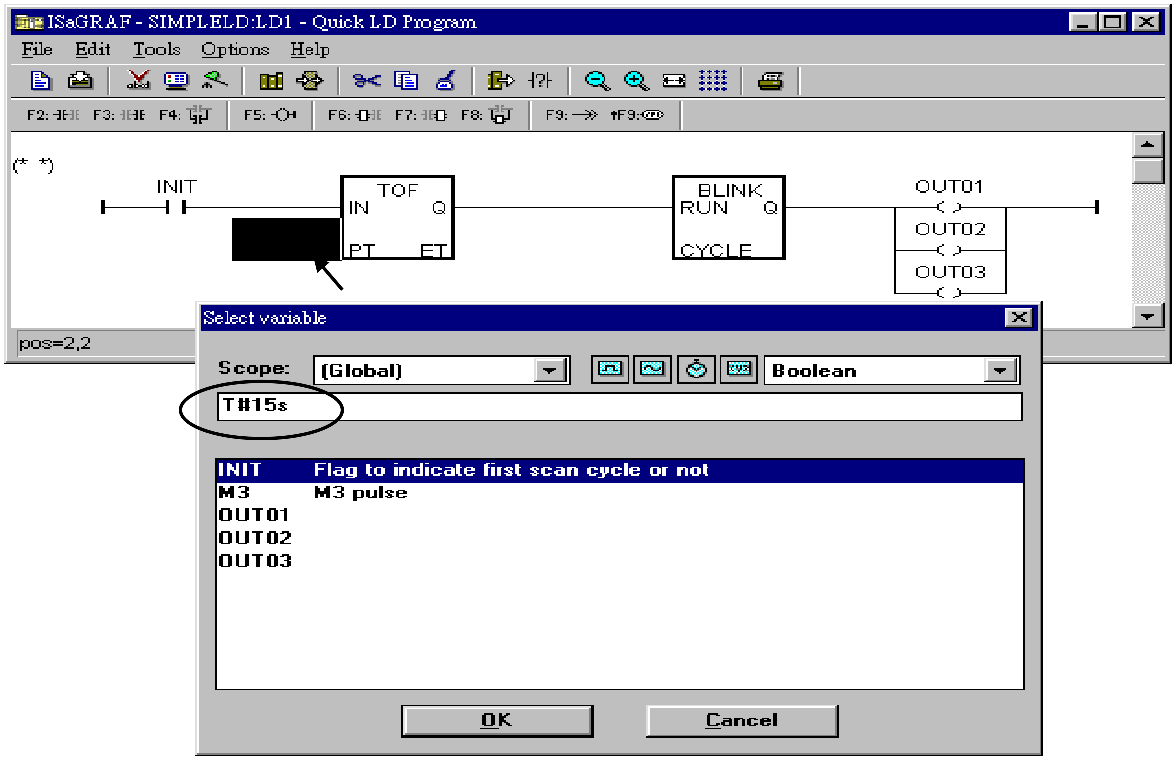
Now we are going to assign the associated variable & constant to each item. Double click on the first contact, a "Select variable" screen appeared. First select the "Scope" to "(Global)" and the proper type to "Boolean". Then double click on "INIT" or you may use the keyboard to type "INIT".



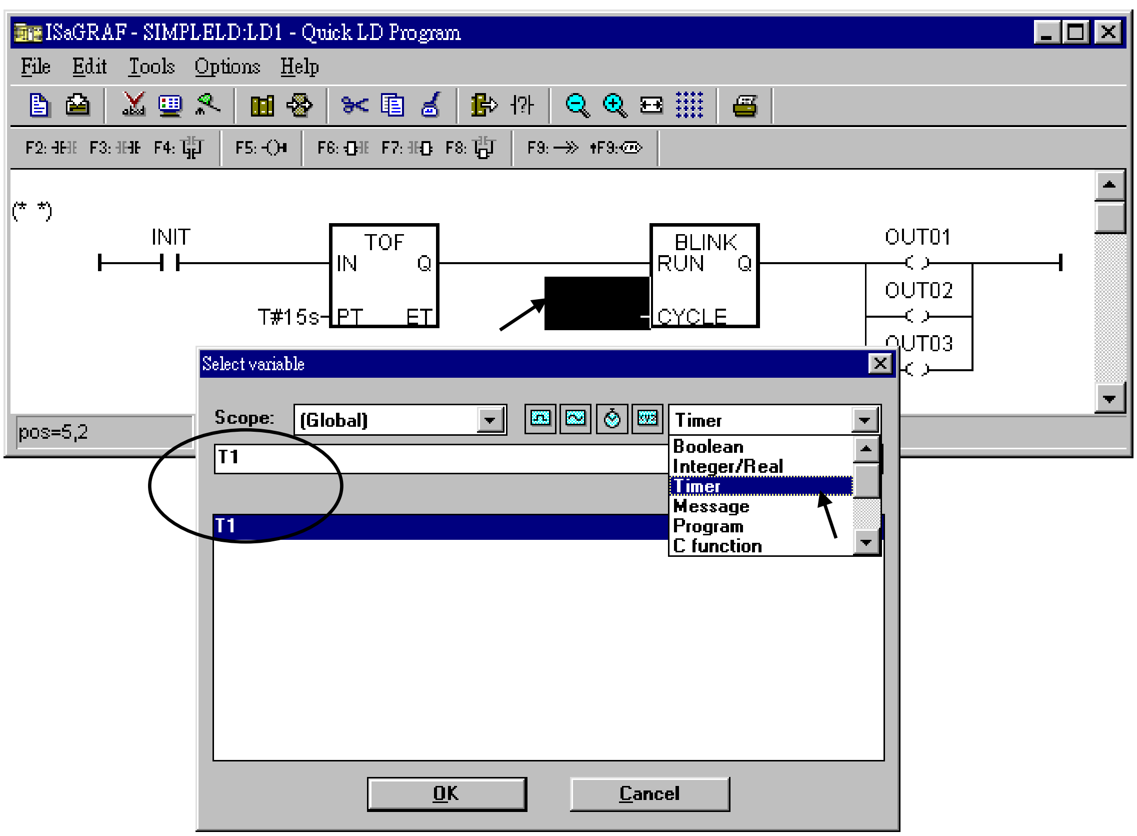
Using the same procedure to assign OUT01 thru. OUT03 to the associated coil.



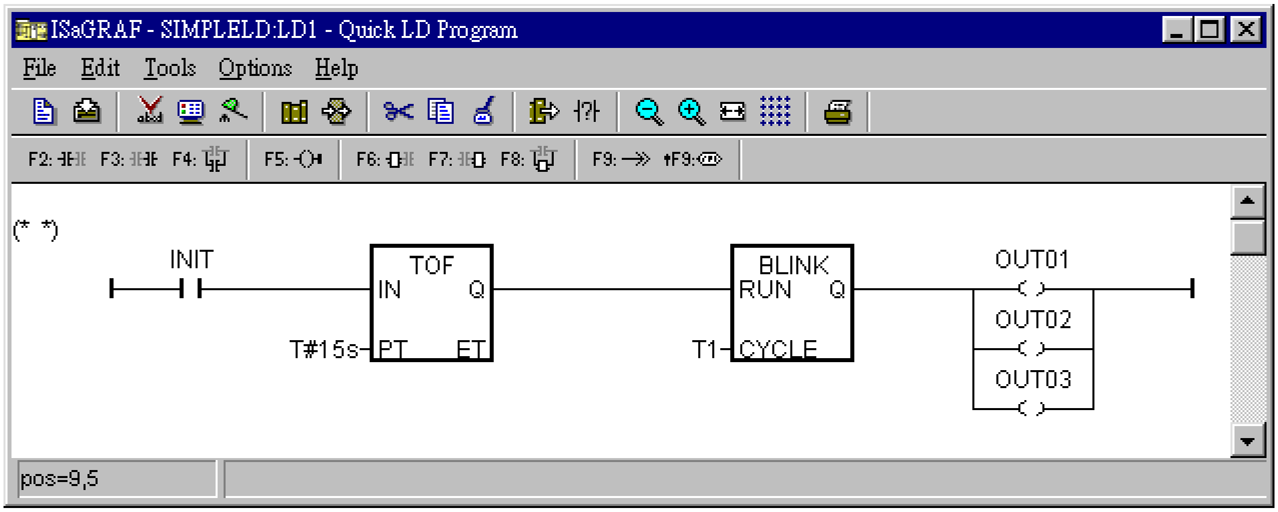
Now move your cursor to the left of the parameter “PT” of the “TOF” block. Double click on it, type “T#15s” (it means 15 second), then press “OK”.



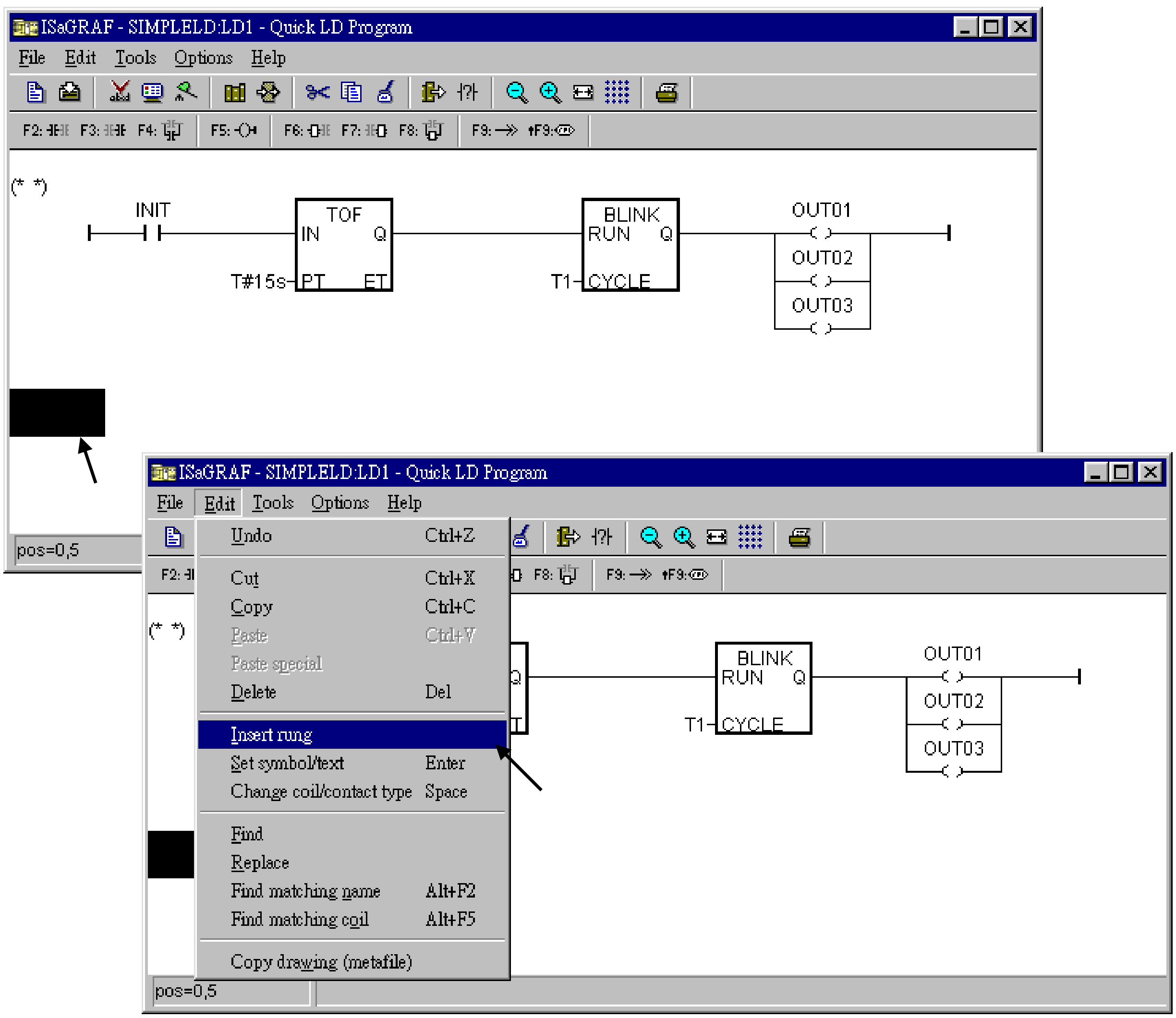
Do the same way to assign “T1” to the left of the parameter “CYCLE” of the “BLINK” block.



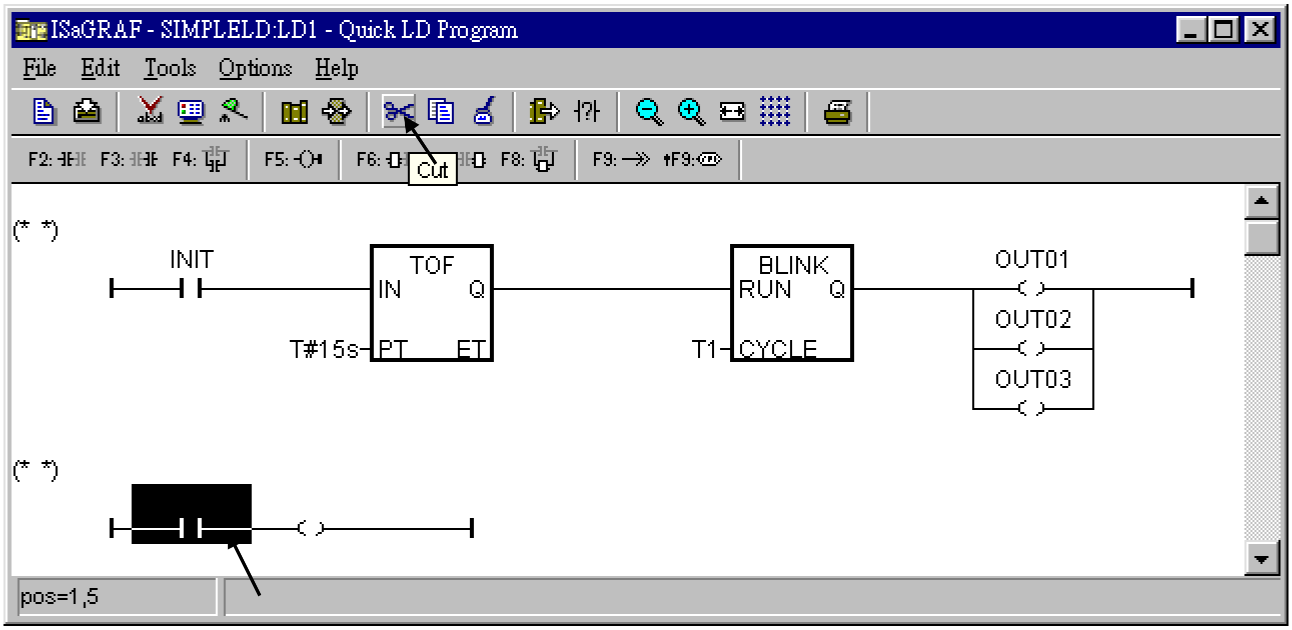
Now the window will look like below.



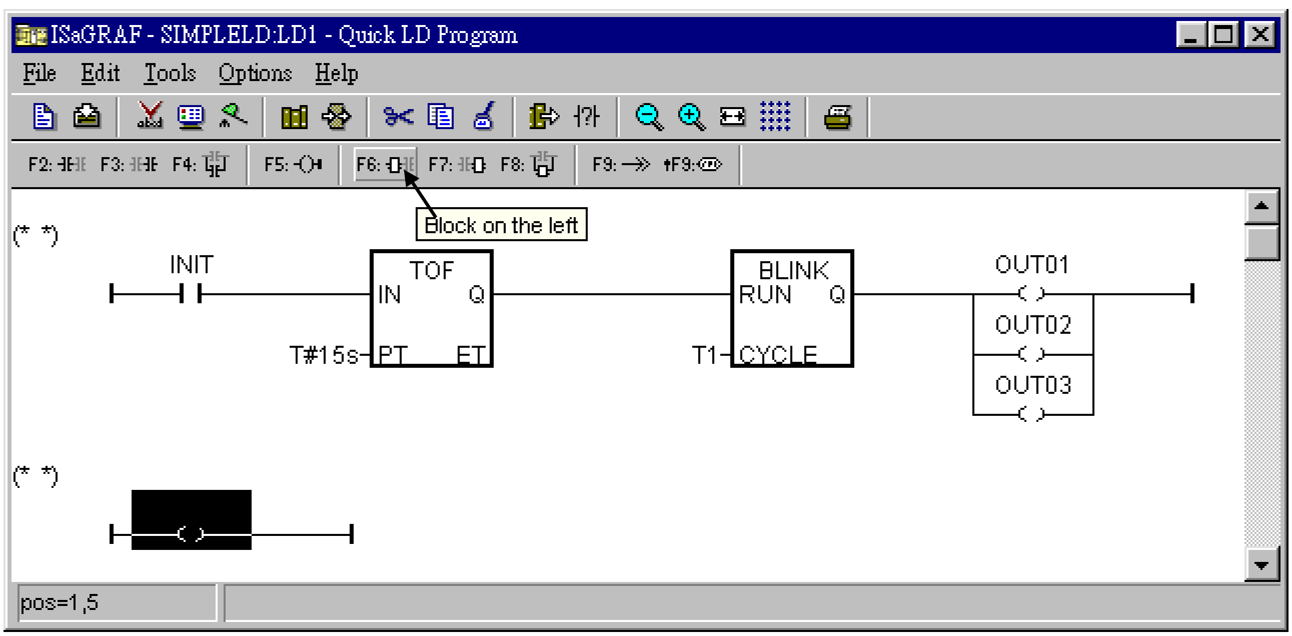
To add a new LD rung, first move the cursor to the proper position below the first rung. Then click on "Edit – Insert rung"



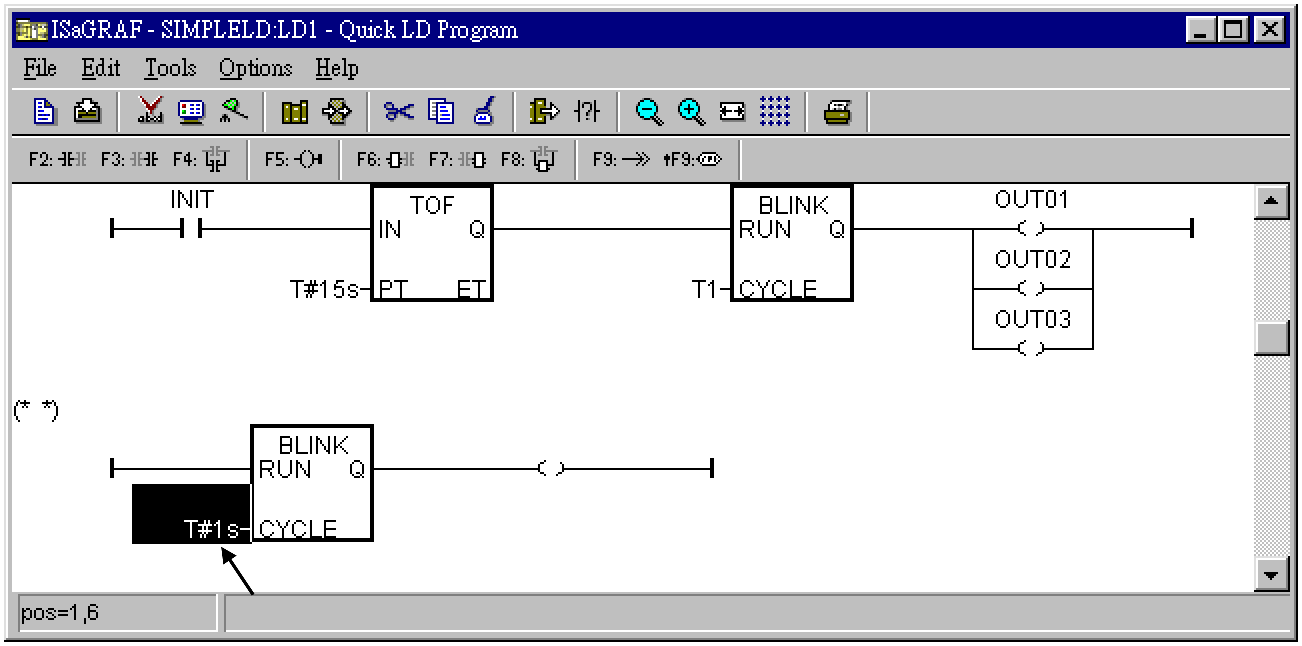
We don’t need the contact in the new rung, move cursor to it, then click on “Cut”.



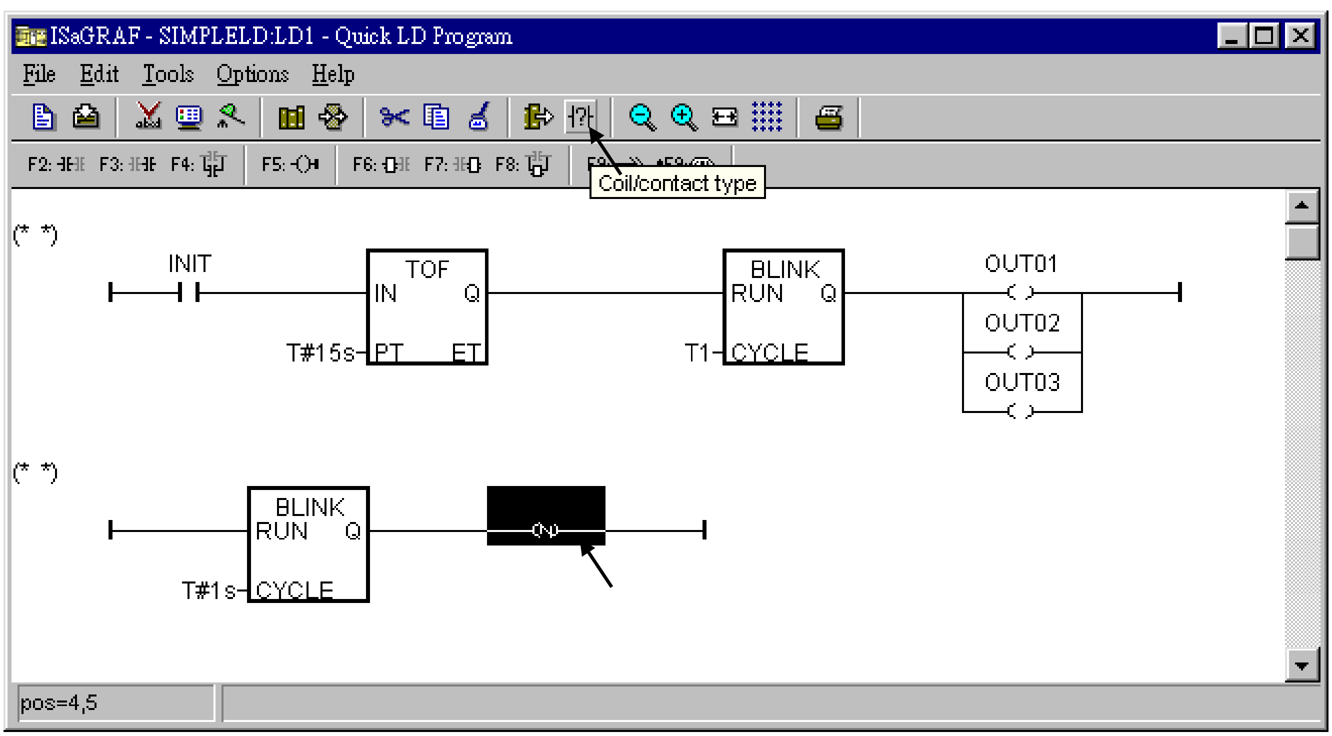
Now click on “F6 (Block on the left)” , and then double click on inside the block to create a “BLINK” block.



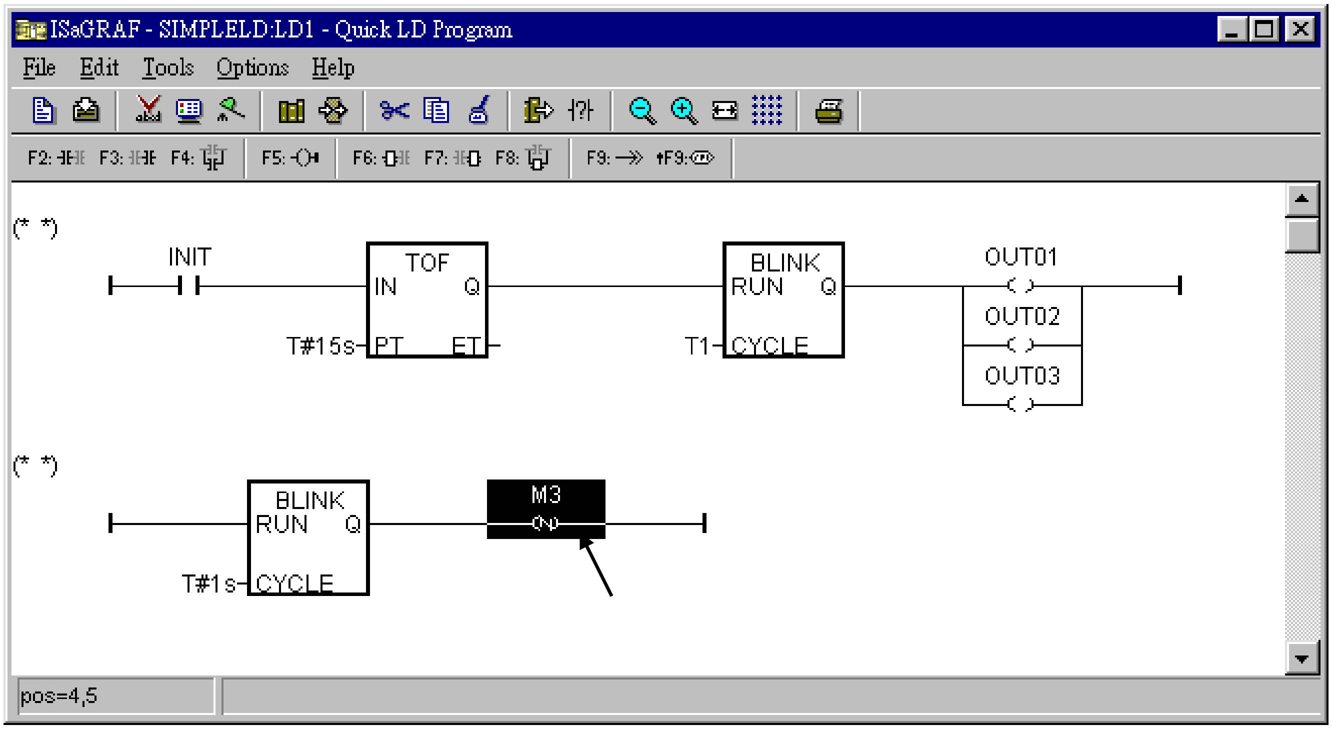
Assign “T#1s” to the parameter of “CYCLE”, then we got the below window.



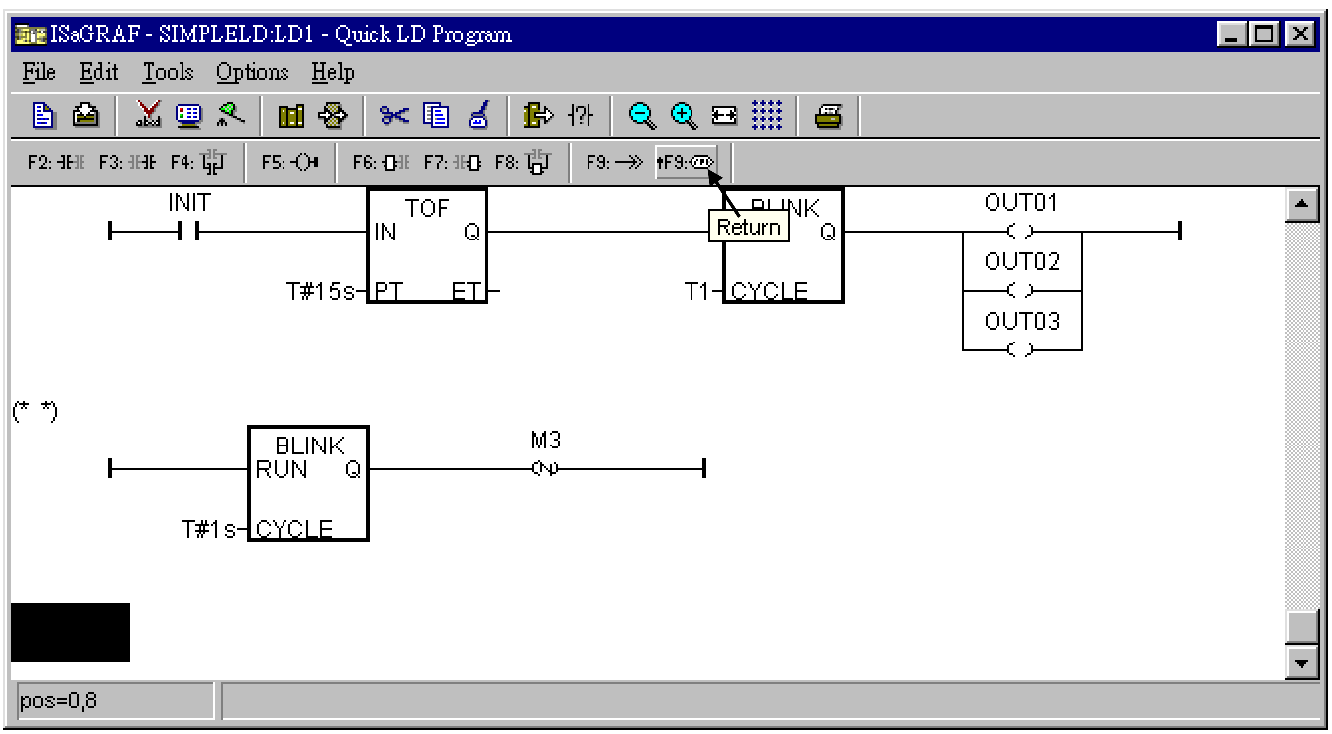
Move the cursor to the right coil, then click on “Coil/contact type” some times to assign the type to “N”.



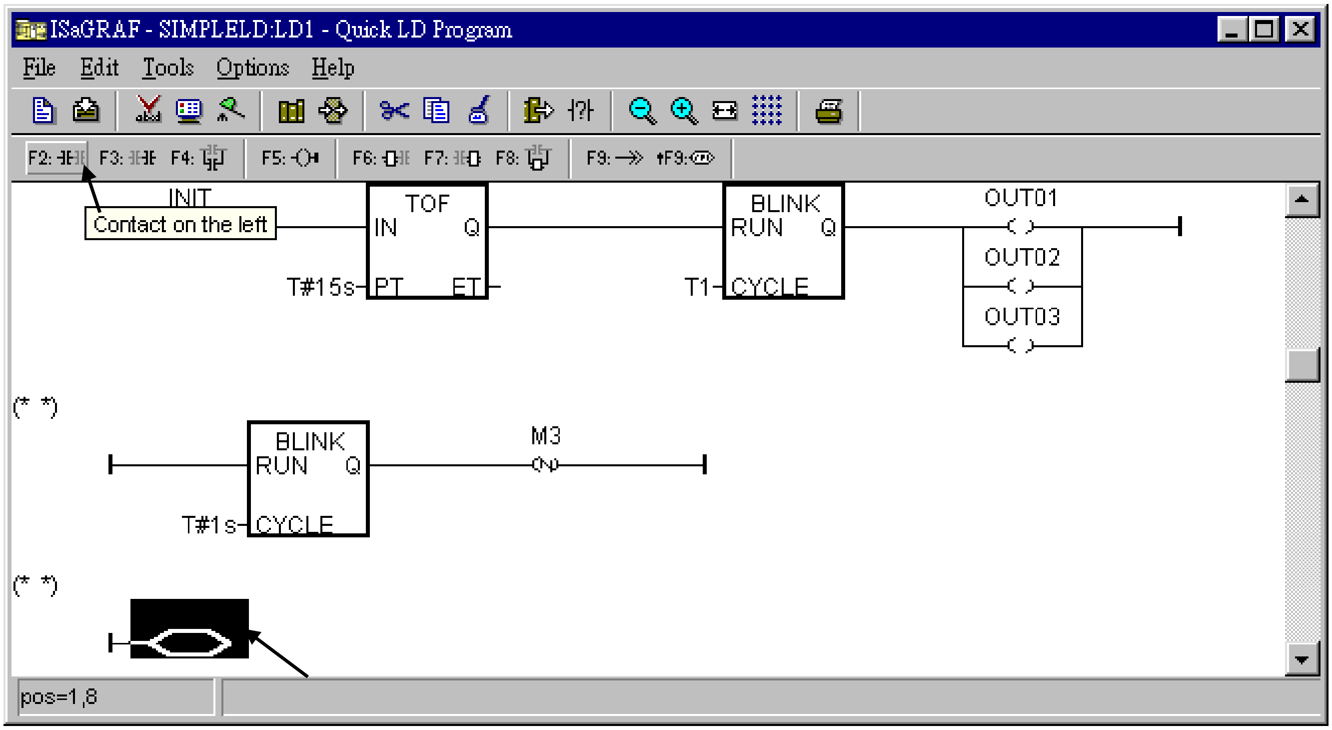
Double click on the “N” coil to assign “M3” to it.



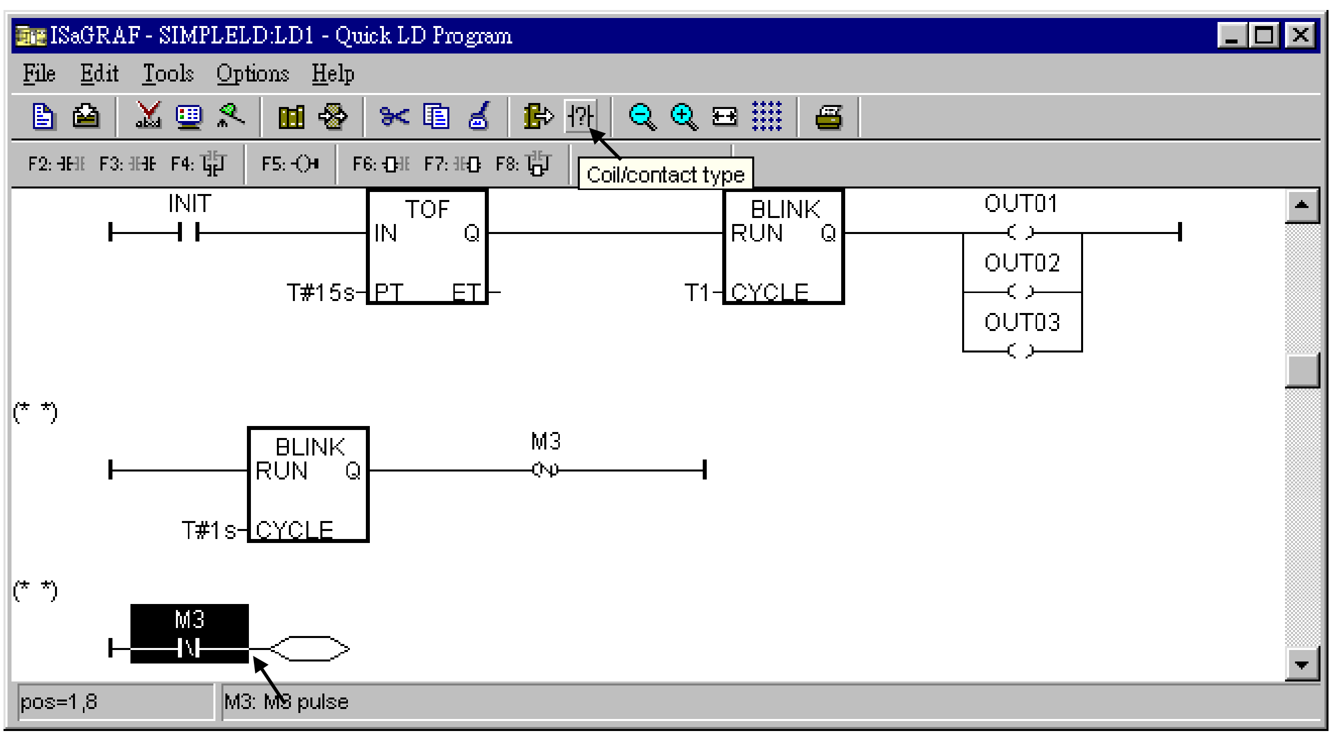
Now we are going to add another LD rung. Move the cursor to the below position of the second rung. And click on “F9 (Return)”.



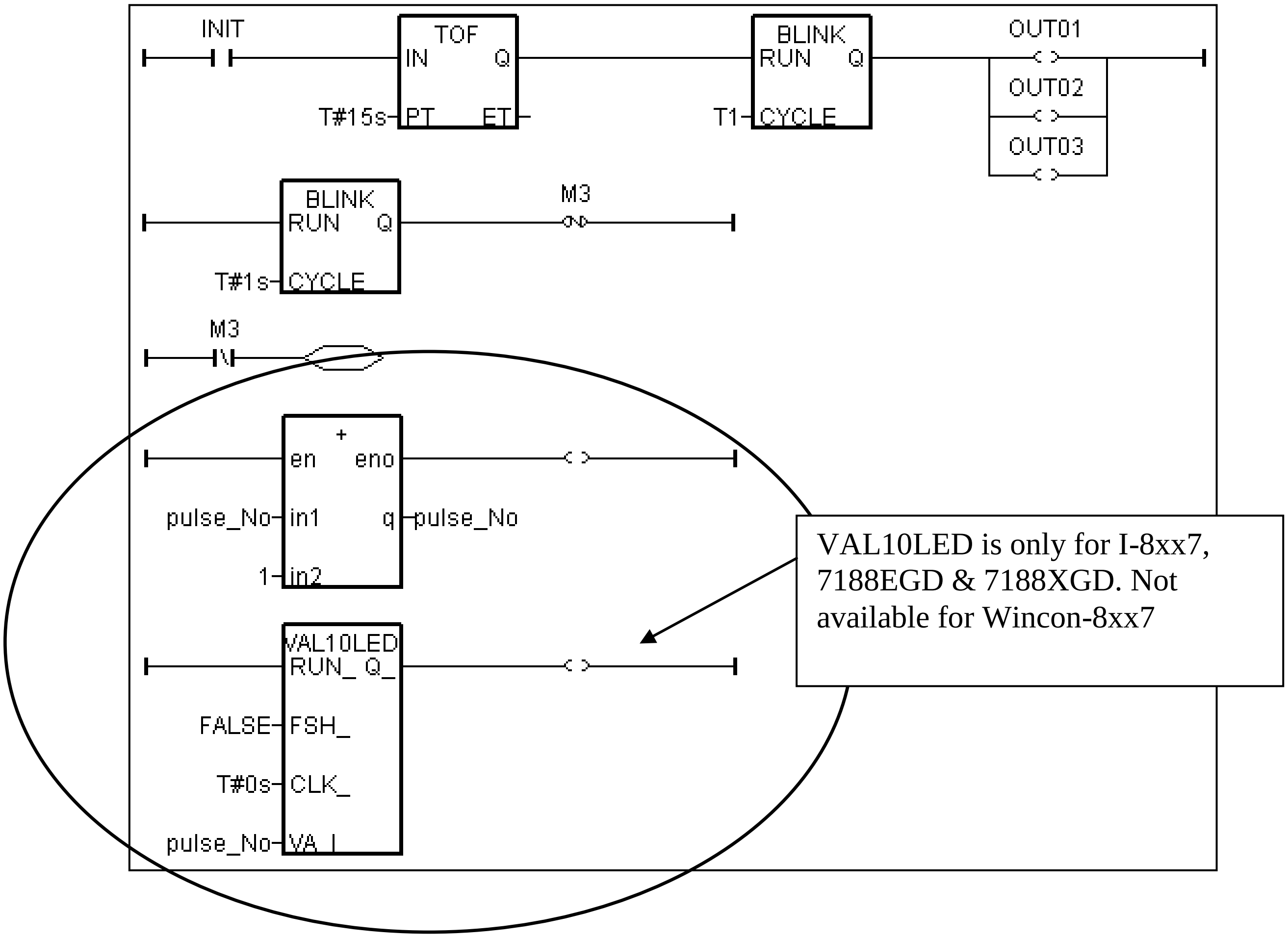
Move the cursor to “return” and then click on “F2 (Contact on the left)” to add a contact on the left.



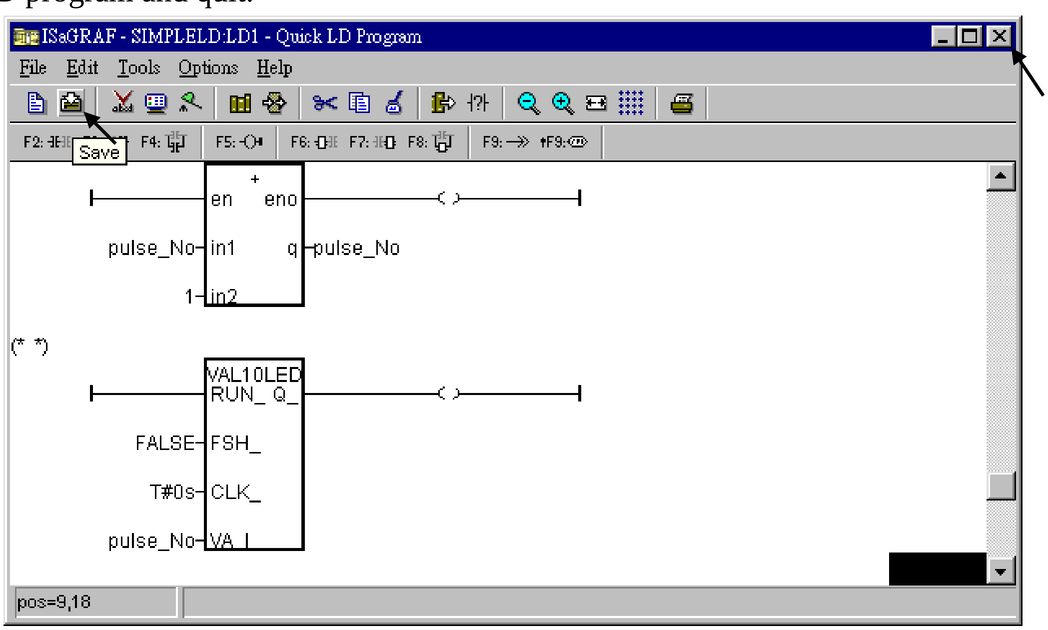
Then double click on the contact to assign “M3” to it. And change its type to “\” (inverted contact).



The procedure to create the forth & the last LD rung is similar as former steps. Please do it by yourself. The final LD program should look like the below.

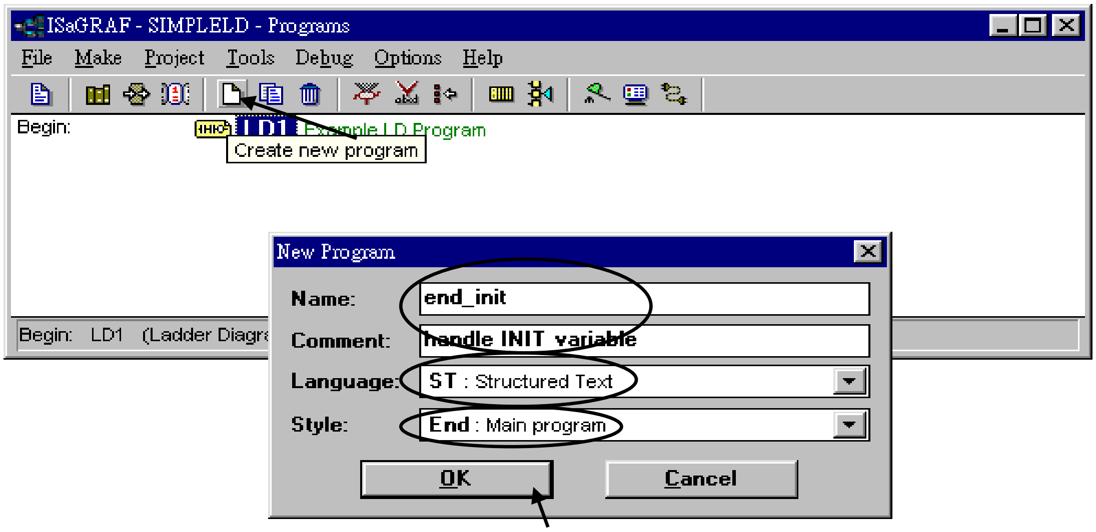


Save this LD program and quit.

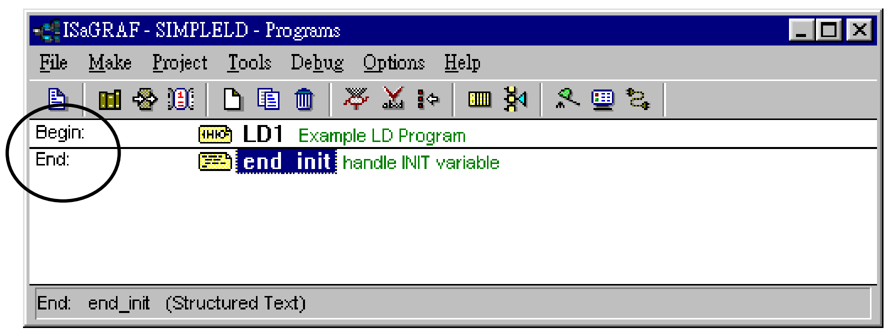


2.1.1.6: Create The ST "end_init" Program

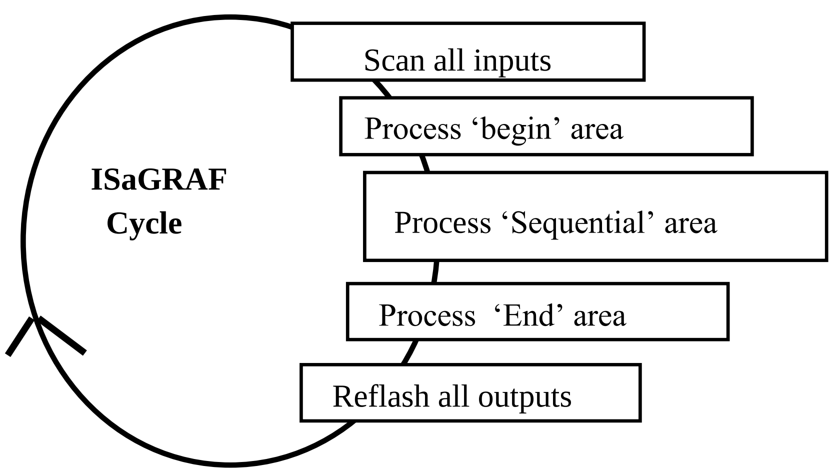
In this project we need an extra ST program to handle the “INIT” variable. Click on “Create new program” in the “ … - Programs” window to add a ST program. Given the Name as “end_init” , Comment as “Handle INIT variable”, Language as “ST: Structured Text” , & Style as “End: Main program”. Then click on “OK”.



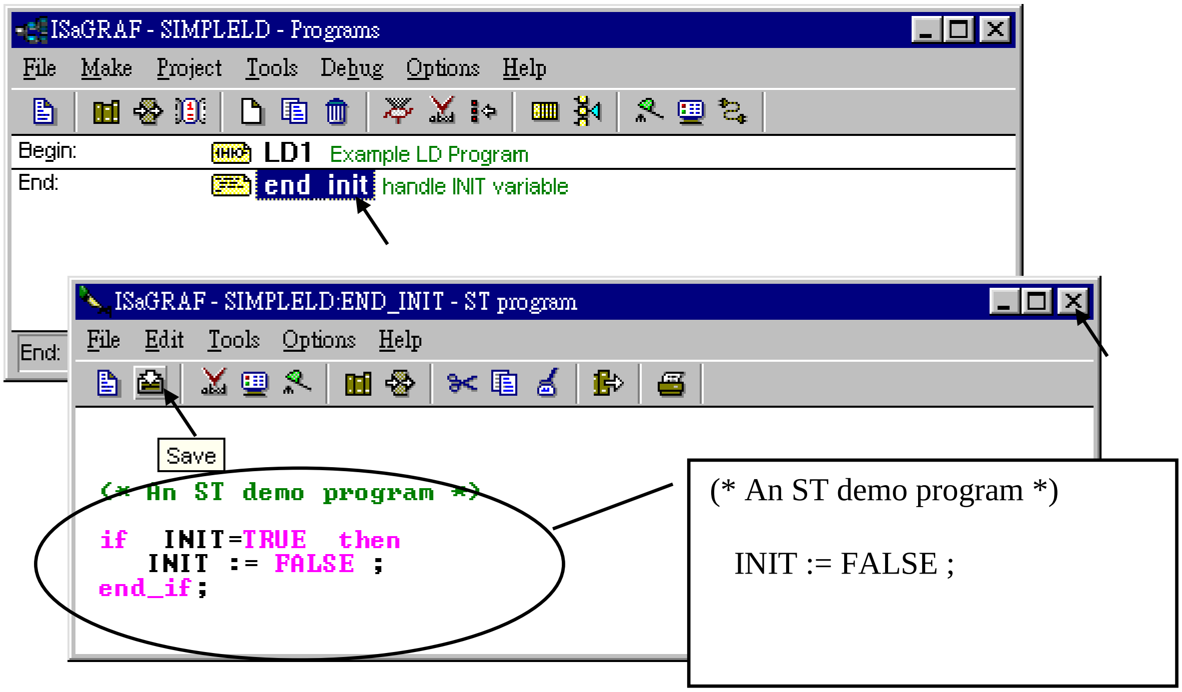
Now we have two programs inside this project.



ISaGRAF will run these two programs one time in each PLC scan cycle. Programs in the “begin” area will run first, then the “Sequential” area, and last the “End” area. An ISaGRAF cycle run in the way as the below scheme.



Double click on "end_init" program to edit it. Click on "save" and then exit when you finish it. (Any character inside between "(*" and "*)" is the comment.)

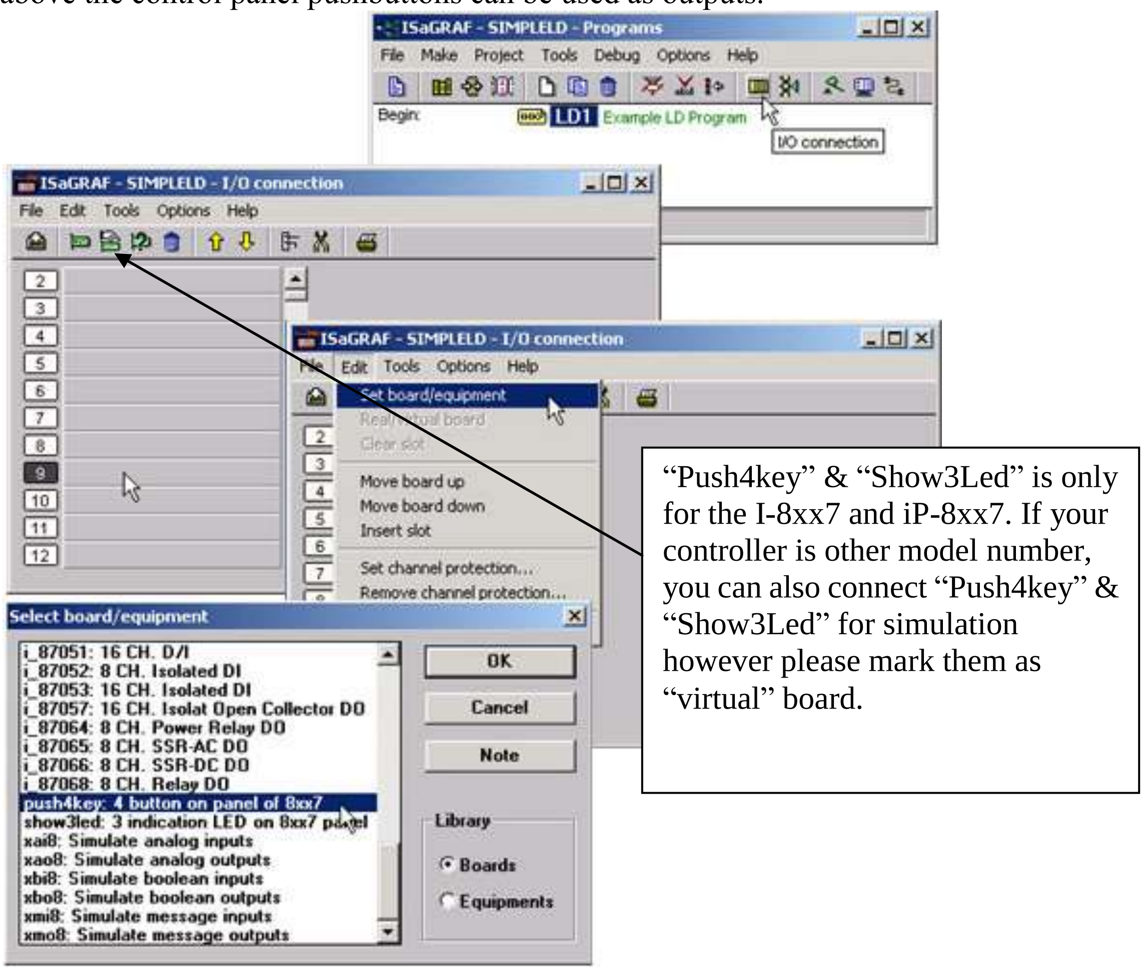


Since "INIT" is declared with an initial value "TRUE", this ST program will let "INIT" set to "FALSE" at the end of the first scan cycle. In other word, "INIT" will indicate this project is running in the first scan cycle or not (TRUE: first scan cycle, FALSE: other cycles).

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 "driver" in their respective controller for the ISaGRAF program to operate properly. The ICP DAS series ISaGRAF controllers have the ISaGRAF driver embedded, creating a powerful and flexible industrial controller system.

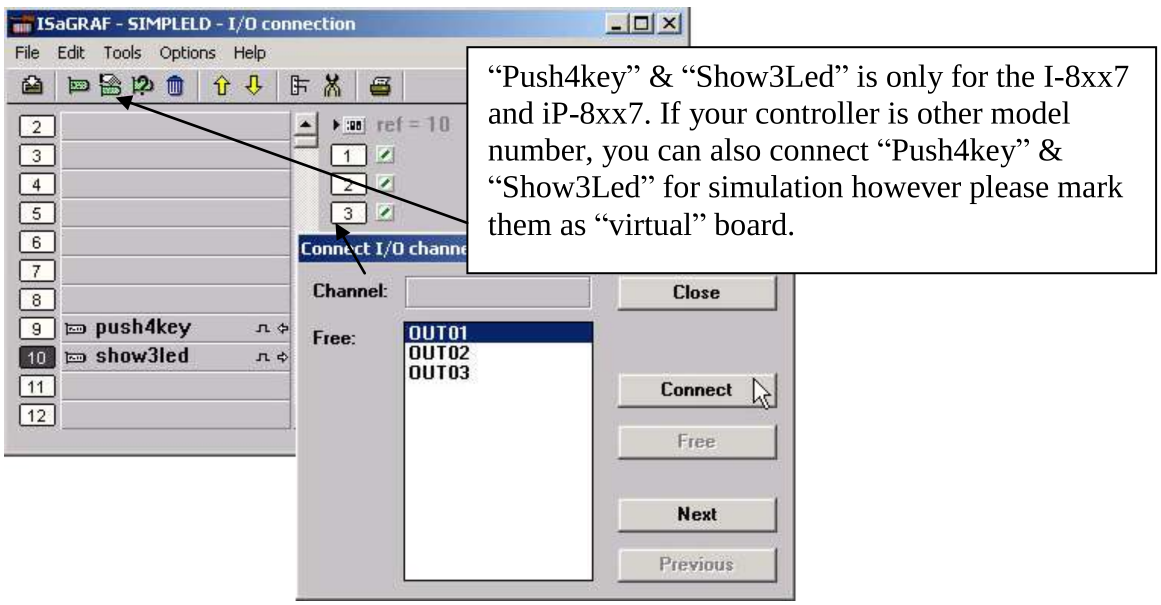
Now that you have created the ISaGRAF example program, now you must connect the I/O to the controller system. A useful feature of the ISaGRAF controller system is that you can run program we have created WITHOUT having any I/O boards plugged into the controller system. The four pushbuttons on the I-8xx7, iP-8xx7 controller system can be used as four digital inputs, and the three left LED's above the control panel pushbuttons can be used as outputs.



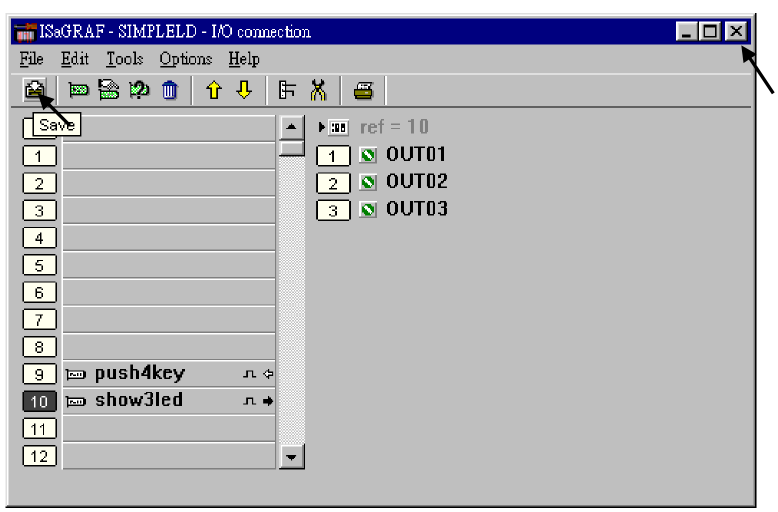
Click on the "I/O Connection" icon as shown in the top picture and the "I/O Connection" window will appear as shown in the next illustration. For the purpose of this example, you can either double click on the "9" slot, or just click on the "9" slot, then click on "Edit" and then "Set Board/Equipment" and then the "I/O Connection" window will appear. This now associates the four control panel pushbuttons - "push4key" as four digital inputs. (We don't use it in this example program since there is no boolean variable declared with "Input" attribution).

IMPORTANT NOTICE:
I/O Slots 0 through 7 are reserved for REAL I/O boards that will be used in the ISaGRAF controller. You can use slots 8 and above for additional functionality as illustrated by the example program.

To create the I/O connections for the outputs, double click on the "10" slot, then click on the "Show3led: 3 indication LED on 8xx7 panel" selection. This will now associate the three LED's above the four control panel pushbuttons as the three outputs for the example program. Your "I/O Connection" window should now look like the screen below.



Remember to click on the "SAVE" icon to save the I/O connections that have been created for the example program. And click on the "X" to exit the window.



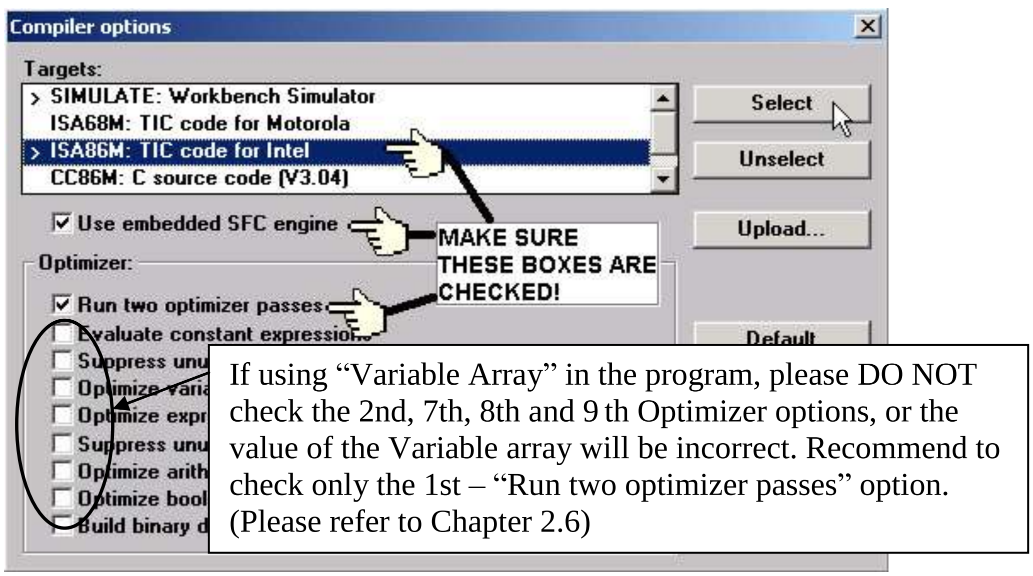
IMPORTANT NOTE:
All of the variables with Input and Output attribute MUST be connected through the I/O connection as described above for any program to be successfully compiled. Only the Input and Output attributed variables will appear in the "I/O Connections" window. In this example we have only 3 boolean output variables, they are OUT01, OUT02 & OUT03.

2.1.3: Compiling The Example LD Project

<u>For ANY AND EVERY ISaGRAF program to work properly with any of the ISaGRAF 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.</u>

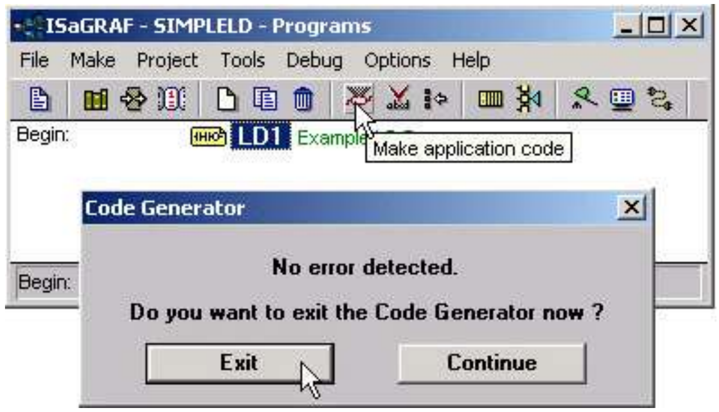
First, click on the "MAKE" option from the main menu bar, and then click on "Compiler Options" as shown below.

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.



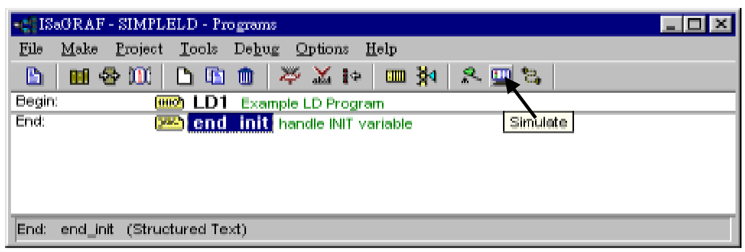
TIME TO COMPILE THE PROJECT!
Now that you have selected the proper compiler options, click on the "Make Application Code" icon to compile the example LD project. If there is no compiler errors detected during the compilation process, CONGRATULATIONS, you have successfully created our example LD program.

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.



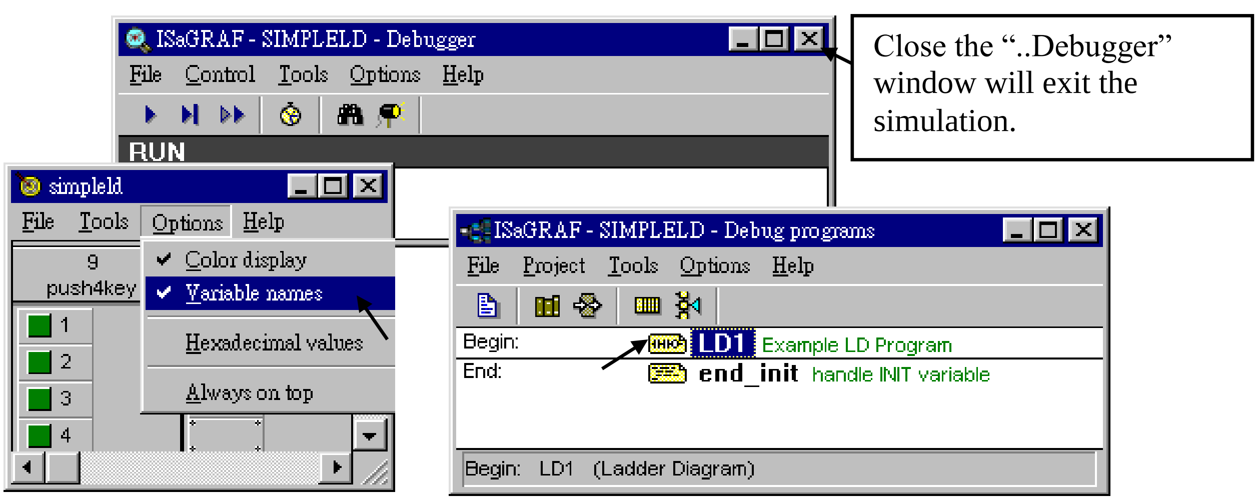
2.1.4: Simulating The LD Project

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



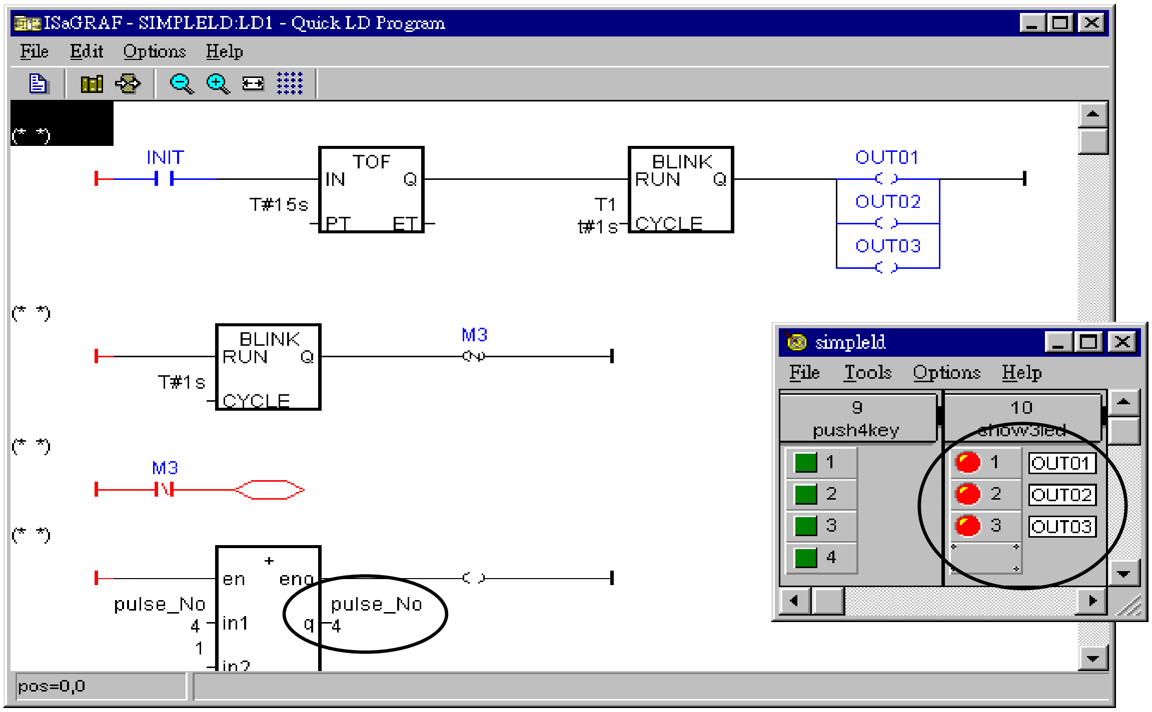
When you click on the "Simulate" icon three windows will appear. The windows are the "ISaGRAF Debugger", the "ISaGRAF Debug Programs", and the "I/O Simulator" windows. If the I/O variable names you have created DO NOT appear in the I/O simulator window, just click on the "Options" and "Variable Names"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



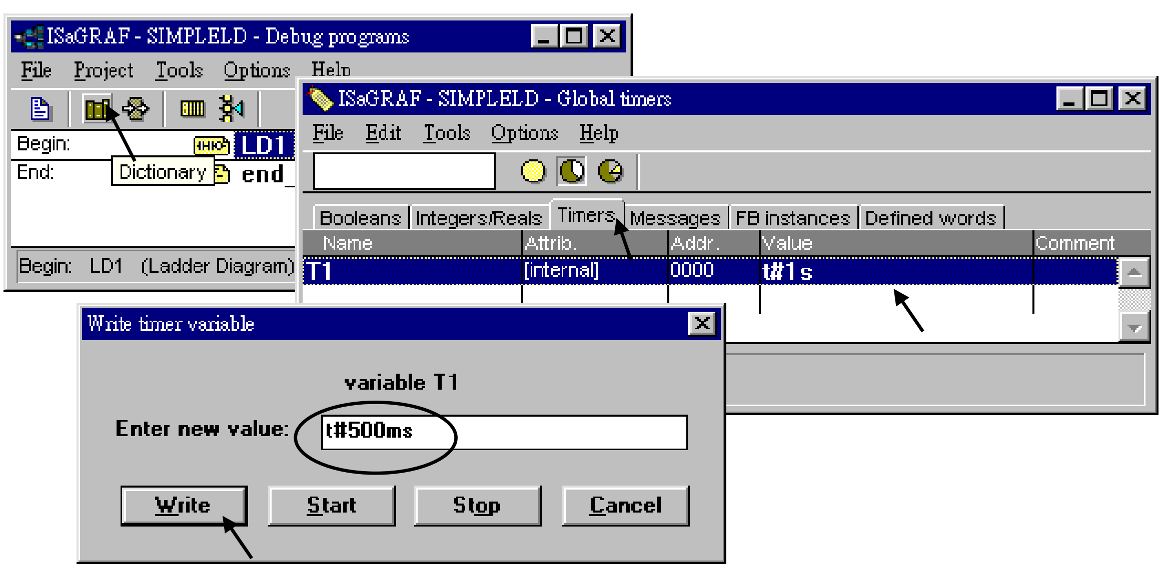
Running The Simulation Program

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



You can see outputs “OUT01” thru. “OUT03” will blink in the first 15 seconds. And the “pulse_No” continuously plus one every second.

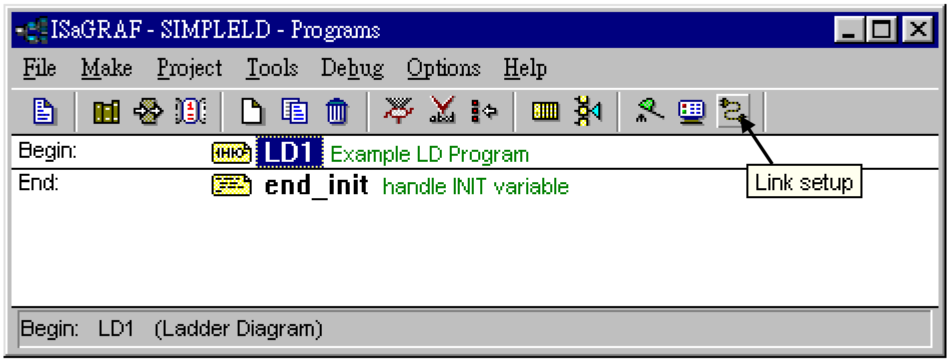
You can adjust the "T1" variable while the program is running. To accomplish this, click on the "Dictionary" icon which will open the "ISaGRAF Global Variables" window as shown in the first two pictures below. Click on “Timer” tab and then double click on “T1” to change the timer value to “T#500ms” (this means 0.5 second). Then click on “Write”.



2.1.5: Download & Debugging The Example LD Project

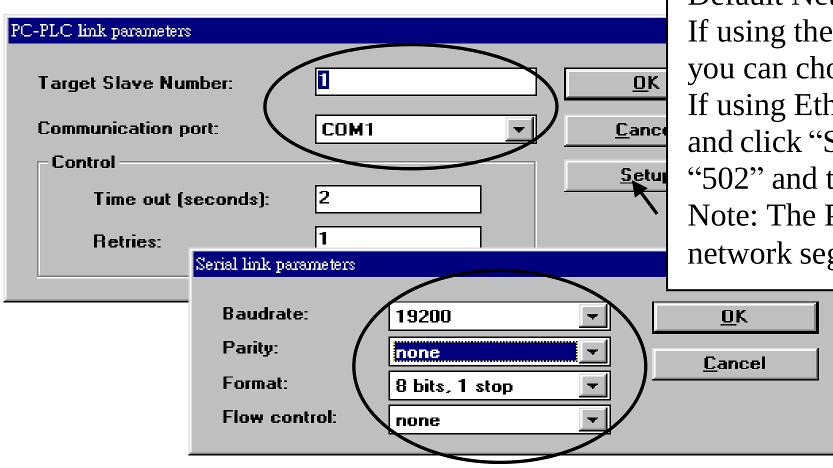
Lastly, you can download the example LD program into the ISaGRAF PAC to do some operations. Before downloading, you must establish communication between the controller and the PC.

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 Net-ID address for the I-8xx7 controller as defined by the DIP Switch settings outlined in Chapter 1, Section 1.3.1. The Net-ID DIP Switch is located in the bottom right portion of the I-8xx7 controller. If your I-8xx7 controller is the first one, the Net-ID address should be set to "1". The "Communication Port" is the serial port connection on your development PC, and this is normally either COM1 or COM2. (The Net-ID setting for the I-7188EG/XG, µPAC-7186EG, µPAC-5xx7, iP-8xx7, WP-8xx7, WP-5xx7, VP-25W7/23W7, XP-8xx7-Atom-CE6 and XP-8xx7-CE6, please refer to their getting started manual)

Default Net-ID of the ISaGRAF PAC is 1.
If using the RS-232/RS-485 port to connect, you can choose COM1 ~9.
If using Ethernet, you can choose "Ethernet" and click "Setup" to fill out the Port number as "502" and the PAC IP (ex. 192.168.255.1).
Note: The PC and PAC must in the same network segament. (ex. PC = 192.168.255.15)

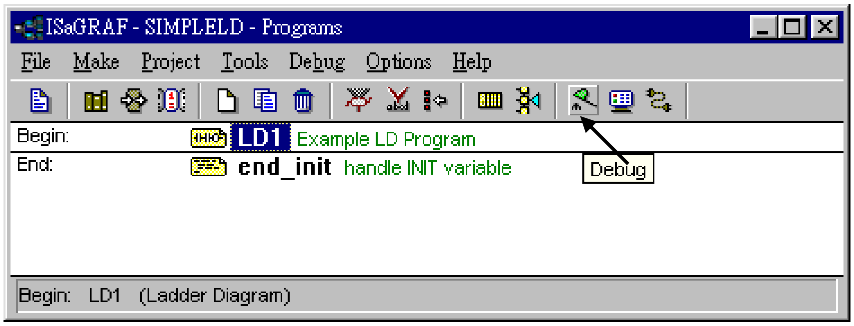


The communication parameters for the target controller MUST be set to the same serial communication parameters for the development PC. For I-8xx7, I-7188EG/XG controllers (serial port communications), and the default parameters for COM1 (RS-232) and COM2 (RS-485) ports are:

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 ISaGRAF controller system.

DOWNLOADING THE EXAMPLE PROJECT
Before you can download the project to the controller, you must first verify that your development PC and the controller are communicating with each other. To verify proper communication, click on the "Debug" icon in the "ISaGRAF Programs" window as shown below.



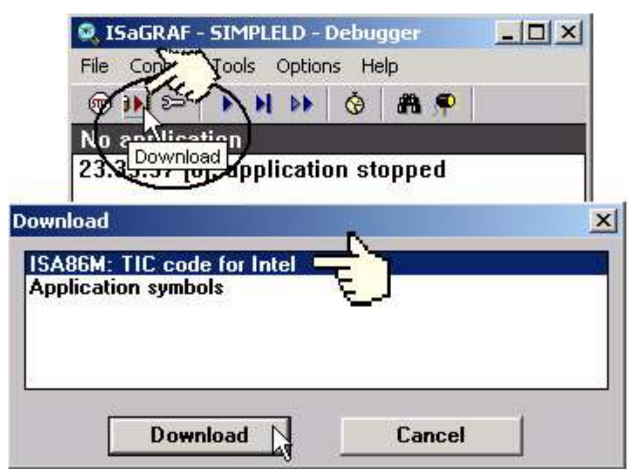
If a program is already loaded in the controller system, the name of the project will be displayed with the word "Active" following it. If the message in the "ISaGRAF Debugger" says "Disconnected", it means that the development PC and the controller system have not established communications with each 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 controller system. You may have to either change the serial port communication settings for the development PC or change the "Serial Link Parameters" in the ISaGRAF program.

If there is a project already loaded in the controller system you will need to stop that project before you can download the example project. Click on the "STOP" icon as illustrated above to halt any applications that may be running.

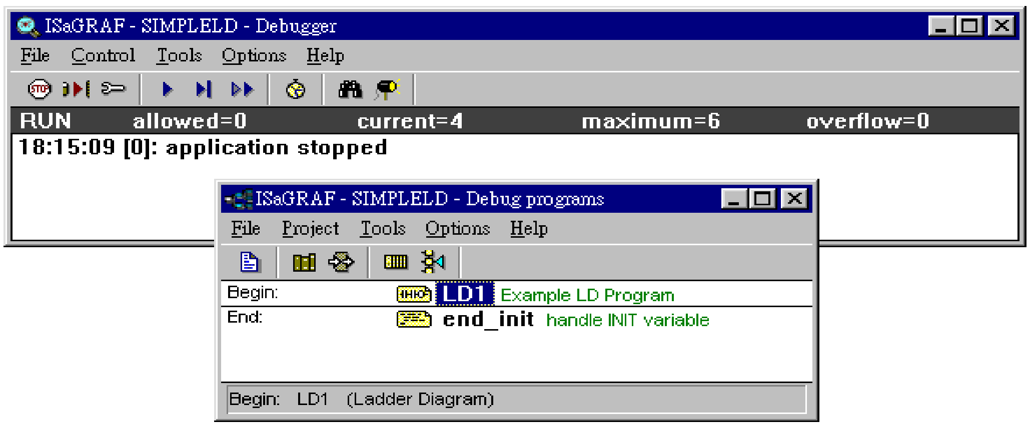
STARTING THE DOWNLOADING PROCESS

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.



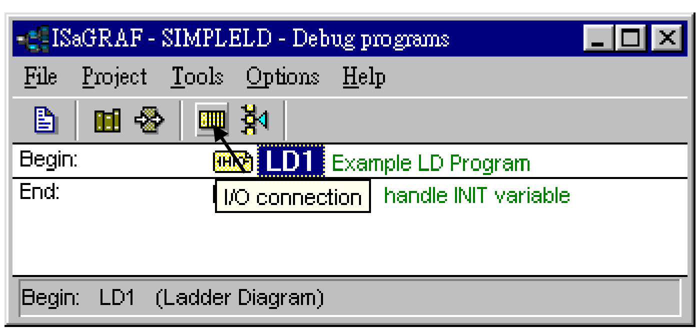
The example project will now start downloading to the ISaGRAF controller system. A progress bar will appear in the "ISaGRAF Debugger" window showing the project downloading progress.

When the example project has successfully completed the downloading process to the controller system the following two windows will appear.

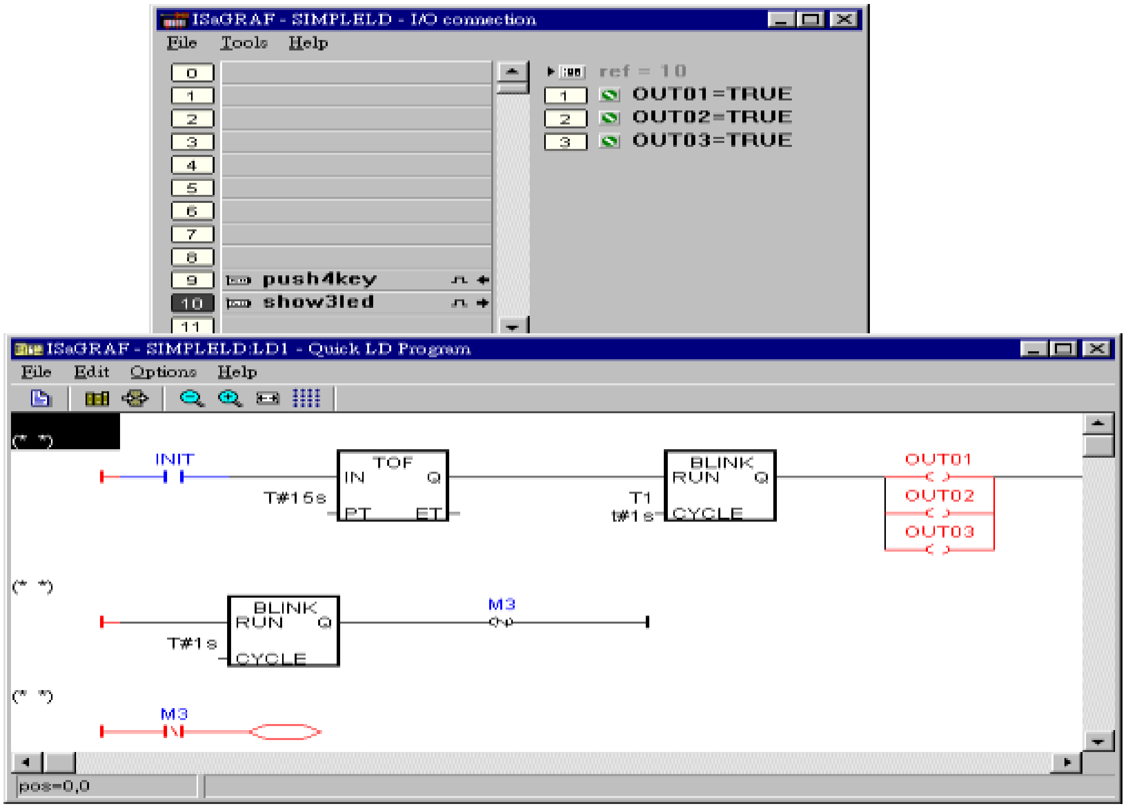


RUNNING THE EXAMPLE LD PROGRAM

You can observe the real time I/O status from several ISaGRAF windows while you are running the example project. 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 Connection" 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 OUT01 thru. OUT03 is blinking in the first 15 seconds. 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 ISaGRAF controller systems.

PRACTICE, PRACTICE, PRACTICE!

Now that you have successfully created and ran your first ISaGRAF program with the ISaGRAF 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 Structured Text (ST) Program

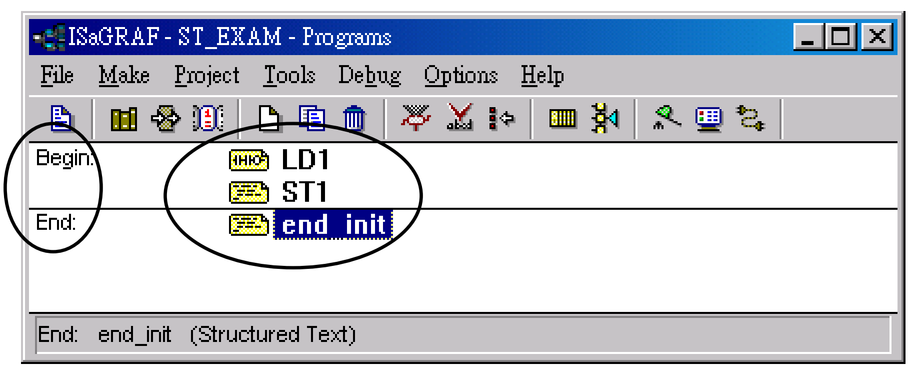
A "Structured Text" program is a high-level program language that is designed for automation process control applications. The "Structured Text (henceforth referred to as "ST") is primarily used to implement complex procedures that cannot be easily expressed by a graphical language such as LD or FBD.

An ST program is comprised by a list of "ST Statements", and each "ST Statement" MUST end with a semi-colon ";". All characters inside between "(*" and "*)" is comment.

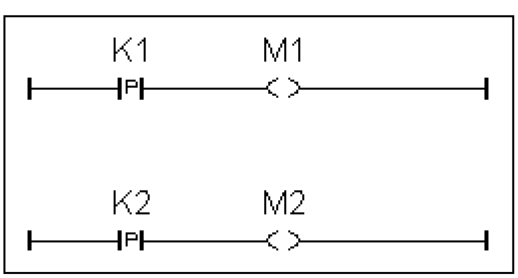
Variables Used In The Example ST Project:

Name	**Type**	**Attribute**	**Description**
INIT	Boolean	**Internal**	**initial value at "TRUE"**. TRUE means 1st scan cycle
K1	Boolean	**Input**	The first pushbutton on the front panel of the I-8xx7
K2	Boolean	**Input**	The second pushbutton on the front panel of the I-8xx7
M1	Boolean	**Internal**	Indicate pushbutton K1 is just pushed.
M2	Boolean	**Internal**	Indicate pushbutton K2 is just pushed.
TEMP	Boolean	**Internal**	A boolean variable for temporary use
COUNT	Integer	**Internal**	A integer value generated by push K1 & K2 **initial value is set at "0"**

Three programs are used in this example. One is LD program named "LD1", The other two are ST programs named respectively as "ST1" & "end_init".



LD program "LD1" Outline:



ST program “ST1” Outline:

```
(* Open Com3 with 9600 baud rate, 8 char. size, no parity, 1 stop bit at first scan cycle *)
if   INIT = TRUE   then
  TEMP  :=  comopen( 3 , 9600 , 8 , 0 , 1 ) ;
end_if  ;

(* Do something when K1 or K2 is pushed *)
if   ( M1 = TRUE )  or  ( M2 = TRUE )   then

  (* COUNT plus 1 when K1 is pushed *)
  If   M1 = TRUE   then
    COUNT  :=  COUNT + 1  ;
  end_if  ;

  (* COUNT plus 10 when K2 is pushed *)
  if   M2 = TRUE   then
    COUNT  :=  COUNT + 10  ;
  end_if  ;

  (* save COUNT value to the 5th Pos. of No.2 integer arry *)
  TEMP  :=  ARY_N_W( 2 , 5 , COUNT )  ;

  (* write one byte = 2 (hex.) to Com3 *)
  TEMP  :=  COMWRITE( 3 , 16#2 )  ;

  (* write 1 integer (1 long integer contains 4 bytes) of Pos. 5 inside No.2 array to Com3 *)
  TEMP  :=  COMAY_NW( 3 , 2 , 1 , 5 )  ;

  (* write one byte = 3 (hex.) to Com3 *)
  TEMP  :=  COMWRITE( 3 , 16#3 )  ;

end_if  ;
```

ST program “end_init” Outline:

```
INIT  :=  FALSE  ;
```

Process Operation Actions:

LD Program “LD1” :

Catch the rising edge status when pushbutton K1 is just pushed and save it into an internal boolean variable “M1”

Catch the rising edge status when pushbutton K2 is just pushed and save it into an internal boolean variable “M2”

ST Program “ST1” :

Open Com3 of the I-8xx7 controller with 9600 baud rate, 8 char. size, no parity and 1 stop bit at the first scan cycle.

Plus “COUNT” value by 1 every time when pushbutton K1 is pushed.

Plus “COUNT” value by 10 every time when pushbutton K2 is pushed.

Send “Count” value to a PC via Com3 of the I-8xx7 controller in the below frame format.

STX	Value of COUNT	ETX

Lowest byte	2nd lower byte	3rd lower byte	Highest byte

STX : Start of frame, byte value = 2

ETX : End of frame, byte value = 3

ST Program “end_init” :

Set boolean variable “INIT” to FALSE at the end of the PLC scan cycle. So that “INIT” will be TRUE only at the first scan cycle.

Function description:

“P” contact : Contact with P type means the right status will be set to a pulse TRUE when the contact is just rising from FALSE to TRUE.

Comopen(PORT, BAUD, CHAR, PARI, STOP) : To open a Com port of the I-8xx7 controller

Parameter

PORT :	Integer	3:COM3 ,4:COM4, ..., 20:COM20
BAUD :	Integer	baud rate, 2400, 4800, 9600, 19200, 38400, 57600, 115200
CHAR :	Integer	char. size, 7 or 8
PARI :	Integer	parity, 0:none, 1:even, 2:odd
STOP :	Integer	stop bit, 1 or 2

Return : boolean ok.: TRUE , fail: FALSE

Ary_N_W(NUM, ADR, DATA) : Save one long integer into an integer array.

Parameter

NUM :	Integer	save to which array (1-6)
ADR :	Integer	save to which Pos. in this array (1-256)
DATA :	Integer	the integer value to save

Return : boolean ok.: TRUE , fail: FALSE

ComWrite(PORT, DATA) : Write one byte to a Com port

Parameter

PORT :	Integer	3:COM3 ,4:COM4, ..., 20:COM20
DATA :	Integer	the byte value (0 - 255) to write

Return : boolean ok.: TRUE , fail: FALSE

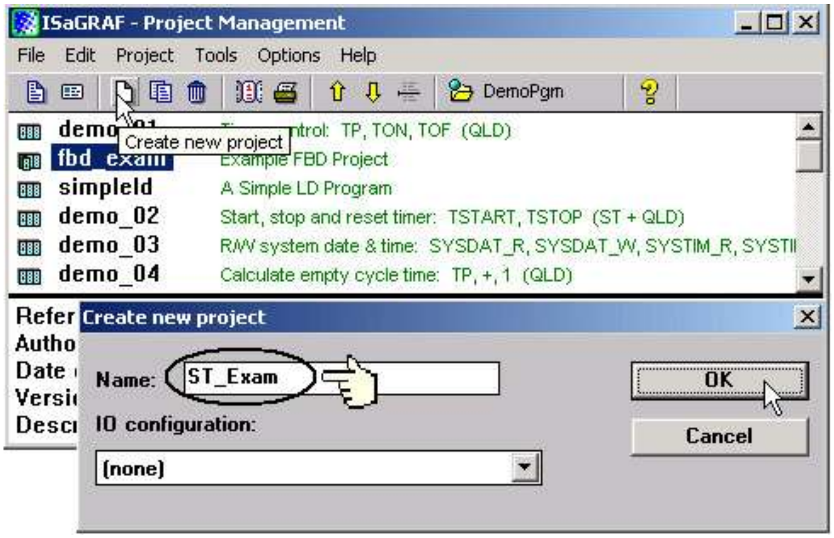
ComAy_NW(PORT, ARY_NO, NUM, POS) : Write an integer array to a Com port

Parameter

PORT :	Integer	3:COM3 ,4:COM4, ..., 20:COM20
ARY_NO :	Integer	the array No. to write (1-6)
NUM :	Integer	number of integers to write (0-256)
POS :	Integer	start position inside the array to write (1-256)

Return : boolean ok.: TRUE , fail: FALSE

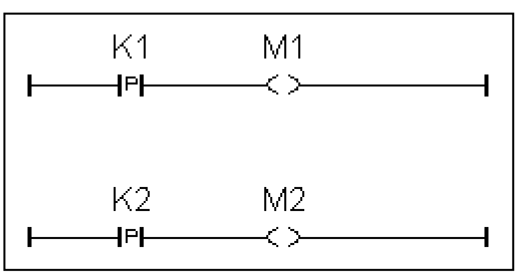
The first step is to create a new project.
From the "ISaGRAF Project Management" window click on the "Create New Project" icon and enter a project name (ex."ST_Exam").



Declaring The Example ST Variables as below content
Refer to Section 2.1.1.3. "Declaring The Variables" for assistance.

Name	**Type**	**Attribute**	**Description**
INIT	Boolean	**Internal**	**initial value at "TRUE"**. TRUE means 1st scan cycle
K1	Boolean	**Input**	The first pushbutton on the front panel of the I-8xx7
K2	Boolean	**Input**	The second pushbutton on the front panel of the I-8xx7
M1	Boolean	**Internal**	Indicate pushbutton K1 is just pushed.
M2	Boolean	**Internal**	Indicate pushbutton K2 is just pushed.
TEMP	Boolean	**Internal**	A boolean variable for temporary use.
COUNT	Integer	**Internal**	A integer value generated by push K1 & K2 **initial value is set at "0"**

Creating a LD program "LD1" with the below content.
Refer to Section 2.1.1.4. and 2.1.1.5 for assistance.



Follow the same steps as 2.1.1.6. to create a ST program “end_init” with the below content.

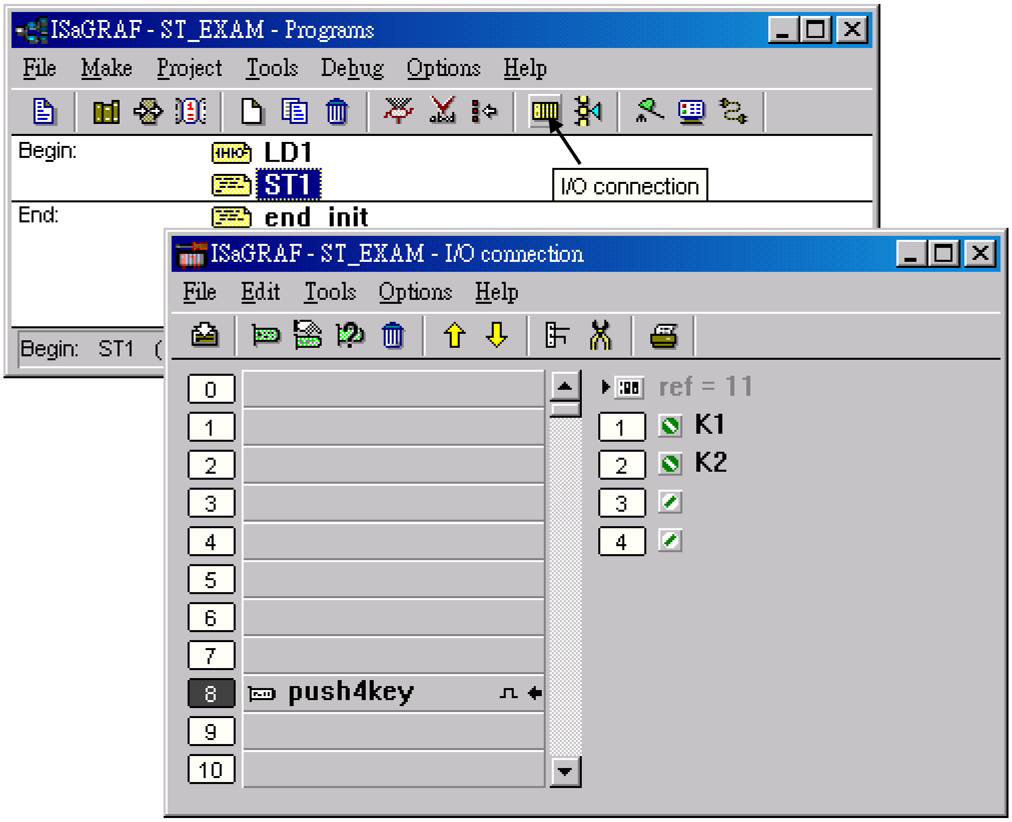
```
INIT  :=  FALSE  ;
```

Creating a ST program “ST1” with the below content.
Refer to Section 2.1.1.6. for assistance.

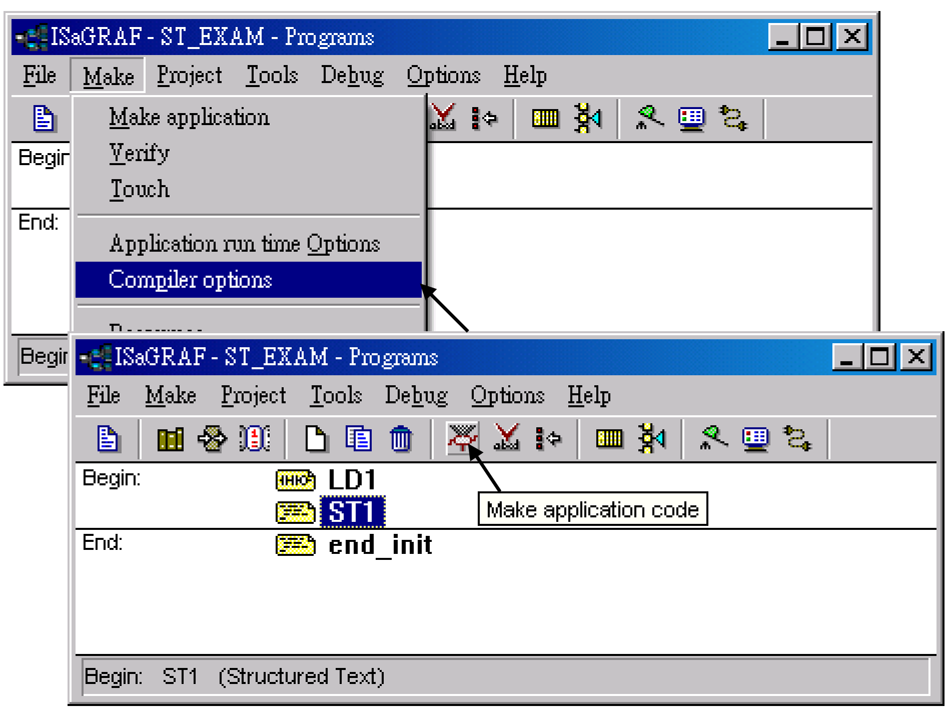
```
(* Open Com3 with 9600 baud rate, 8 char. size, no parity, 1 stop bit at first scan cycle *)
if   INIT = TRUE   then
  TEMP  :=  comopen( 3 , 9600 , 8 , 0 , 1 ) ;
end_if  ;

(* Do something when K1 or K2 is pushed *)
if   ( M1 = TRUE )  or  ( M2 = TRUE )   then

  (* COUNT plus 1 when K1 is pushed *)
  If   M1 = TRUE   then
    COUNT  :=  COUNT + 1  ;
  end_if  ;

  (* COUNT plus 10 when K2 is pushed *)
  if   M2 = TRUE   then
    COUNT  :=  COUNT + 10  ;
  end_if  ;

  (* save COUNT value to the 5th Pos. of No.2 integer arry *)
  TEMP  :=  ARY_N_W( 2 , 5 , COUNT )  ;

  (* write one byte = 2 (hex.) to Com3 *)
  TEMP  :=  COMWRITE( 3 , 16#2 )  ;

  (* write 1 integer (1 long integer contains 4 bytes) of Pos. 5 inside No.2 array to Com3 *)
  TEMP  :=  COMAY_NW( 3 , 2 , 1 , 5 )  ;

  (* write one byte = 3 (hex.) to Com3 *)
  TEMP  :=  COMWRITE( 3 , 16#3 )  ;

end_if  ;
```

IMPORTANT NOTE
Each ST statement line MUST end with a semi-colon ";" as shown above. After entering in the above example program remember to click on the "Save" icon to save the program, then click on "Exit".

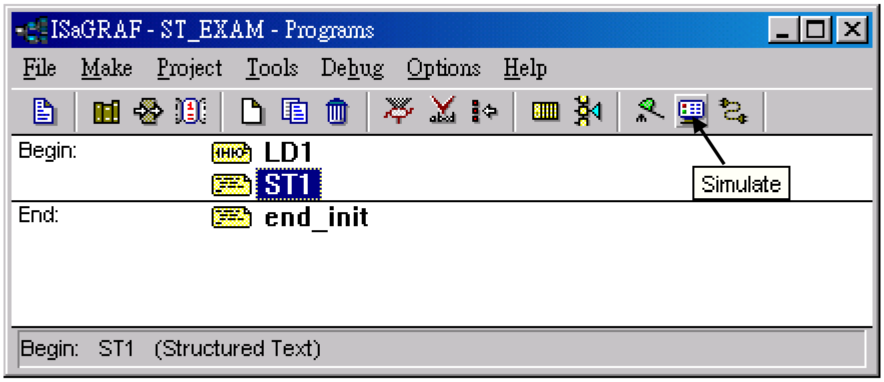
Use the similar procedure for the "Connecting I/O" as detailed in Section 2.1.2



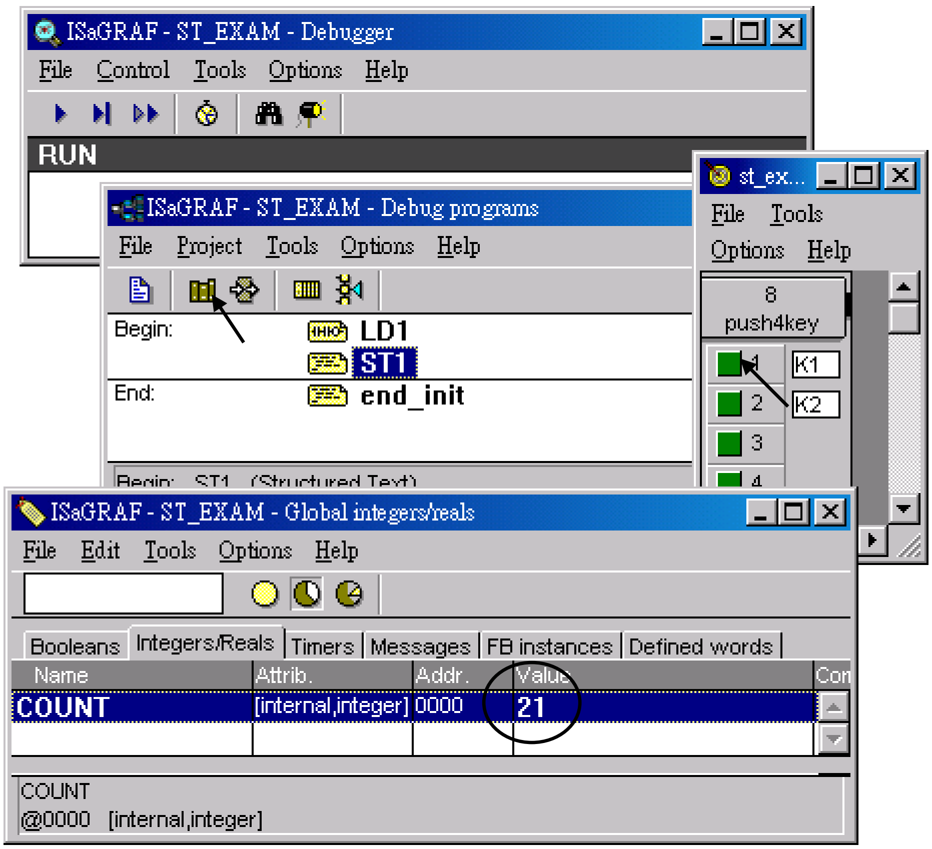
Use the similar procedure for the "Compilling the project" as detailed in Section 2.1.3



After compiling the example ST project click on the "Simulate" icon to observe the ST program running.



You may open the dictionary window to see the “COUNT” value. Click on “K1” or “K2”, you will see the “COUNT” value is changed.



You can now download this example project to the I-8xx7 controller system. Please follow the same procedure as outlined in Section 2.1.5.

After downloading to the controller, the program will send 6 bytes via Com3 of the controller whenever K1 or K2 is pushed. If you have your RS-232 monitoring program running on your PC, you can connect Com3 to your PC to see how it works.

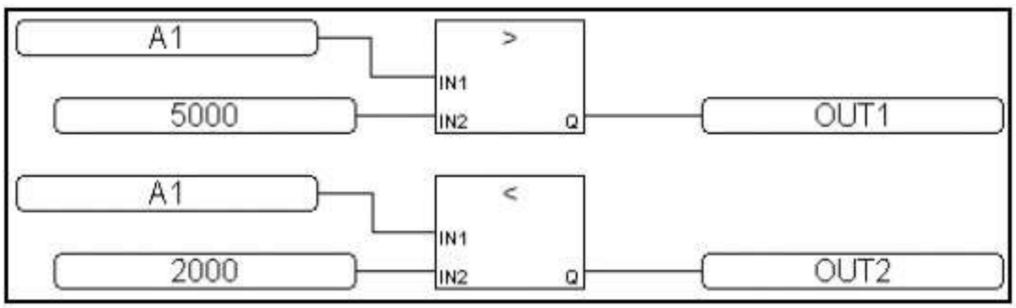
2.3: A Simple Function Block Diagram (FBD) Program

The "Function Block Diagram (FBD) is a graphical programming language that allows a programmer to build complex procedures by taking existing "Functions" from the ISaGRAF library. The following section details how to build a "Function Block Diagram" program with ISaGRAF.

Example FBD Control Specification:
The following details the variables that will be used in our example Function Block Diagram program.

Name	Type	Attribute	Description
OUT1	Boolean	Output	High alarm
OUT2	Boolean	Output	Low alarm
A1	Integer	Internal	Simulate a temperature input, **initial value is 0**

FBD Program Outline:



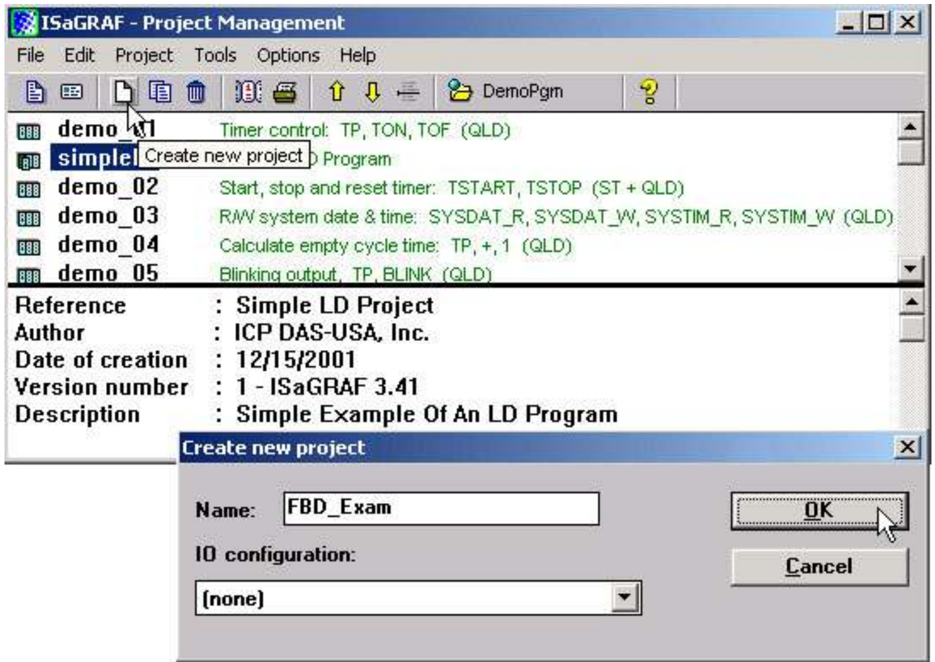
FBD Program Action:

If “A1” > 5000, output “OUT1” is "TRUE".
If “A1” < 2000, output “OUT2” is "TRUE".
Other situation, output “OUT1” and “OUT2” are "FALSE"

2.3.1: Programming The Example FBD Program

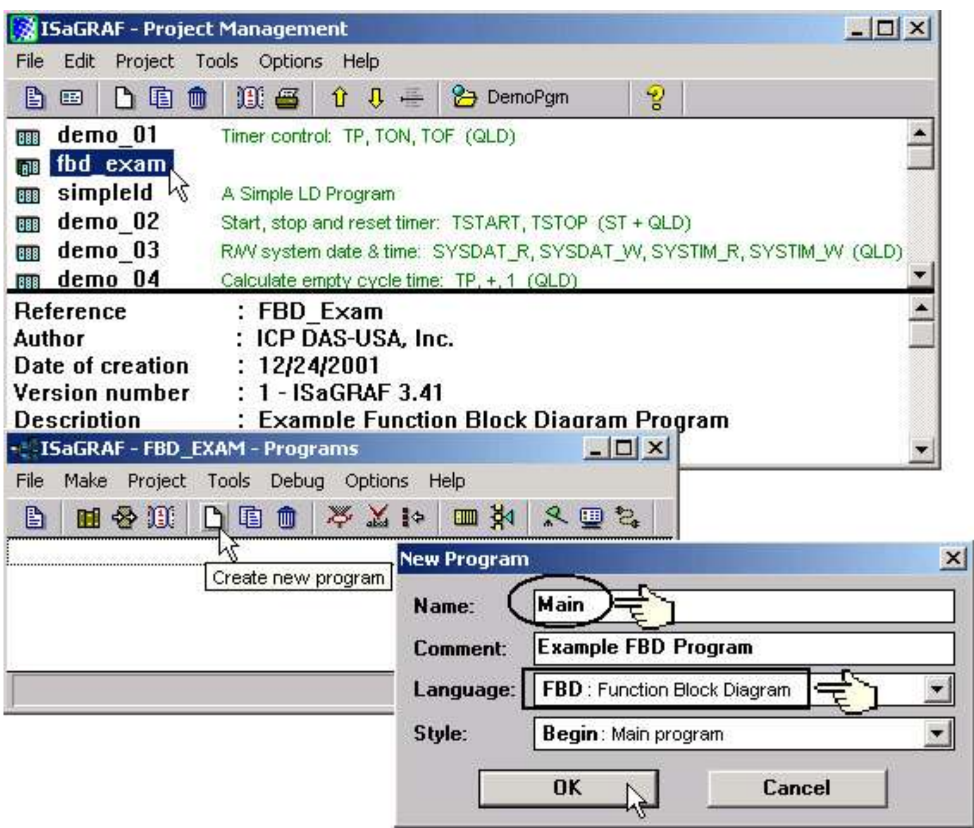
Creating a Function Block Diagram (henceforth referred to as "FBD") program is very similar to creating a LD program as outlined in Section 2.1. The following steps detail how easy it is to create a FBD program.

Creating a New FBD Project
From the "ISaGRAF Project Management" window click on the "Create New Project" icon and enter the name "FBD_Exam".



After you have created the new project, double click on the "FBD_Exam" name in the "ISaGRAF Project Management" window to open the new FBD project. Click on the "Create New Program" icon in the "ISaGRAF Programs" window, which will open the "New Programs" window.

First, enter "Main" in the name field and selecte "FBD: Function Block Diagram" in the "Language" field. You can add a comment about your program, but it is not mandatory. Then click on the "OK" button.

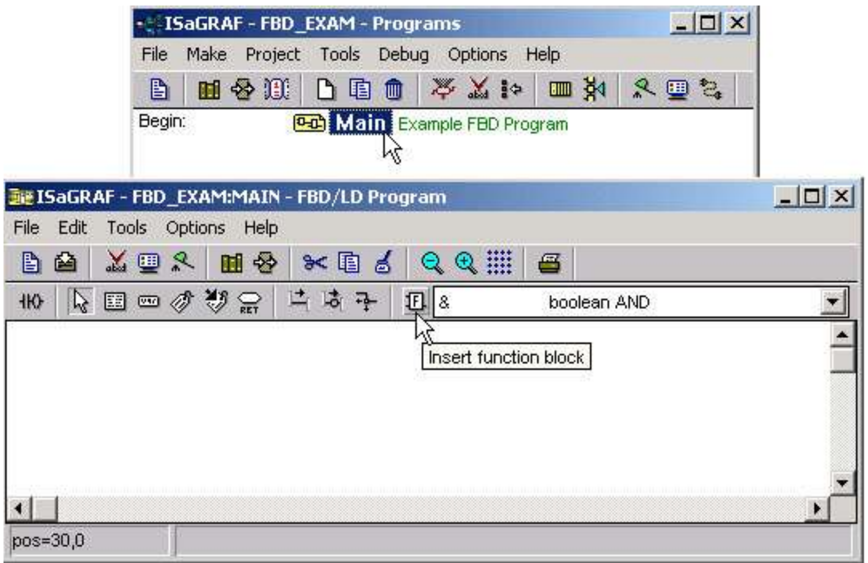


Declaring The Variables

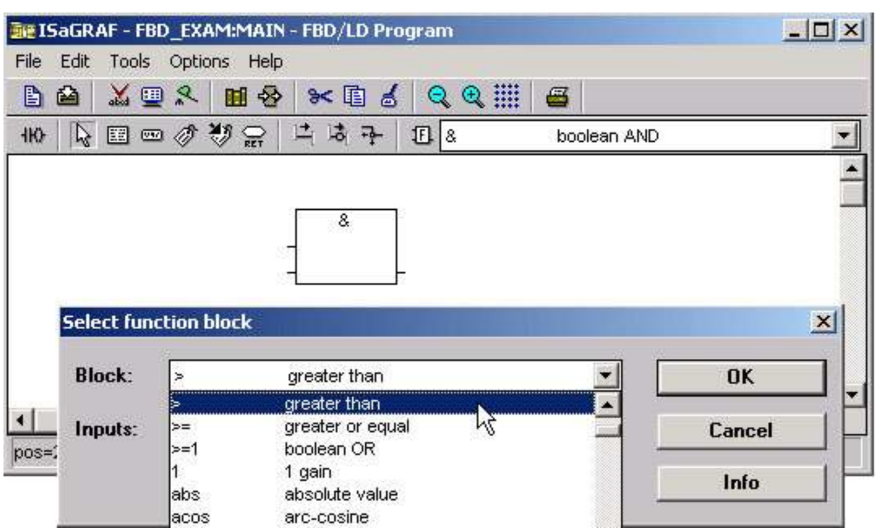
For our example FBD program we are going to declare three variables. The variables to be used are "OUT1", "OUT2", and an integer variable called "A1". Declaring variables for the FBD program is like declaring variables for the LD program. Refer to Section 2.1.1.3 – "Declaring The Variables" to review the variable declaration process.

Editing The FBD Program

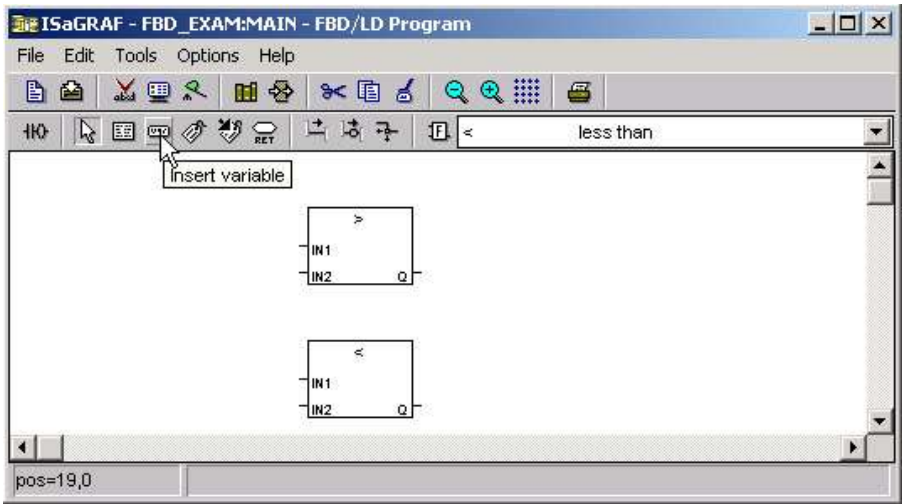
To create and edit the example FBD program, double click on word "MAIN" in the "ISaGRAF Programs" window, and then click on the "Insert Function Block" icon as shown below.



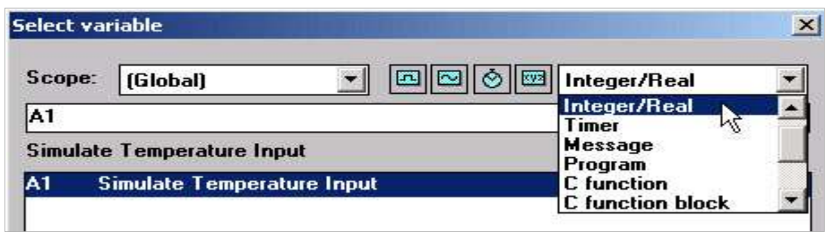
Move the cursor to approximately the middle of the "ISaGRAF FBD/LD Program" window and click the mouse one time to add the first function block. Next, double click on the block to select "> Greater Than". For more information regarding any of the function blocks available in the ISaGRAF program just click on "Info" button.



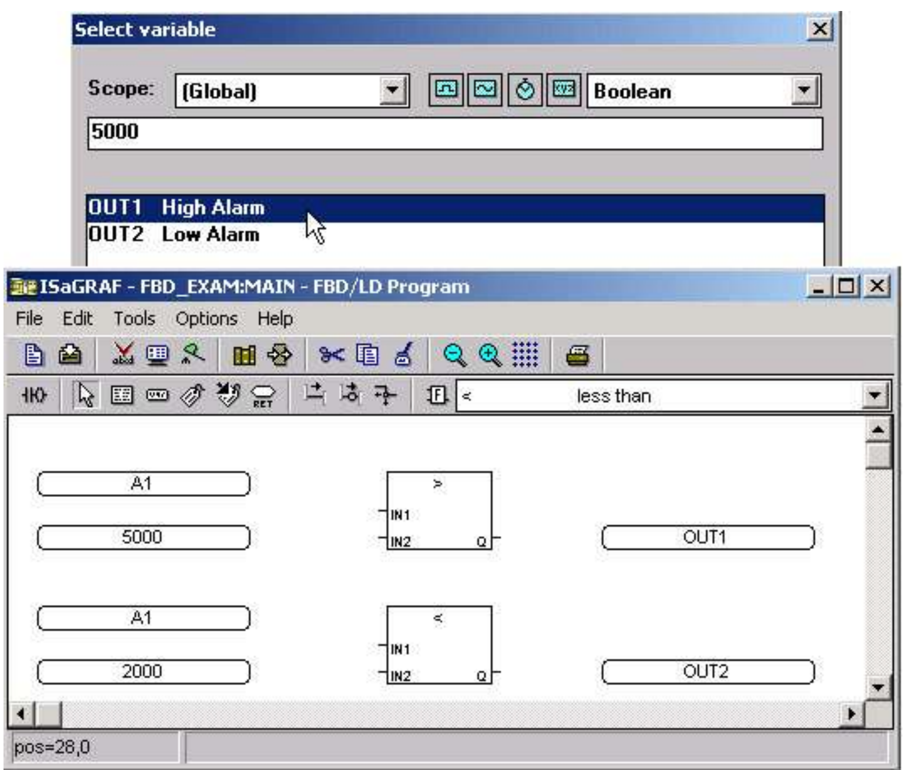
Using the same procedure as described above, add a "< Less Than" function block below the "Greater Than" function block.



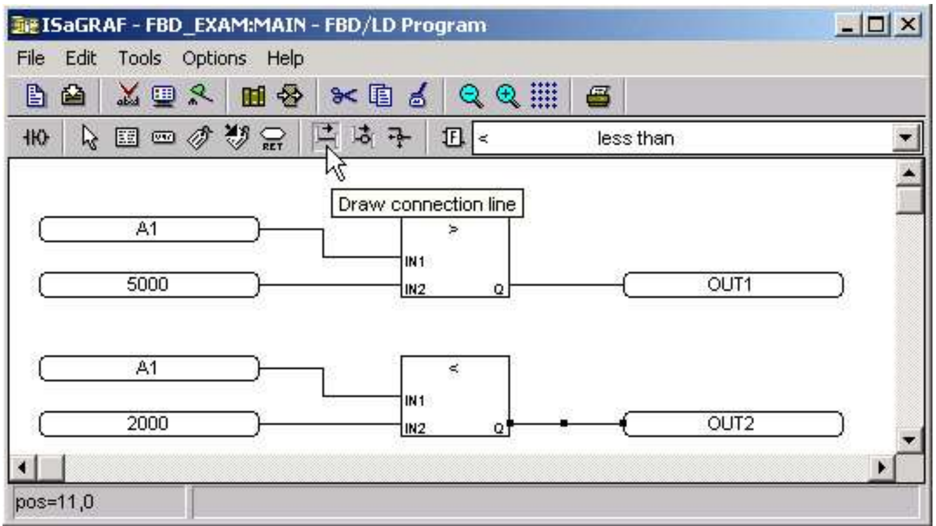
Click on the "Insert Variable" icon as shown above, and then click on "Integer/Real" from the "ISaGRAF Select Variable" window. This will cause the variable "A1" to appear in the "ISaGRAF Select Variable" to appear. Double click on the highlighted "A1 Simulate Temperature Input" which will then place the variable "A1" inside of the "ISaGRAF FBD/LD Program" window.



Repeat the same process to add a second "A1", "OUT1" and "OUT2" variables. Lastly, add two constant variables, "5000" and "2000", and place them below these two"A1" variables as shown below.



The last task to accomplish is making the connection between each of the variables (and constants) and the function blocks. Click on the "Draw Connection Line" icon and draw a line between each of the variables and function blocks as shown below.

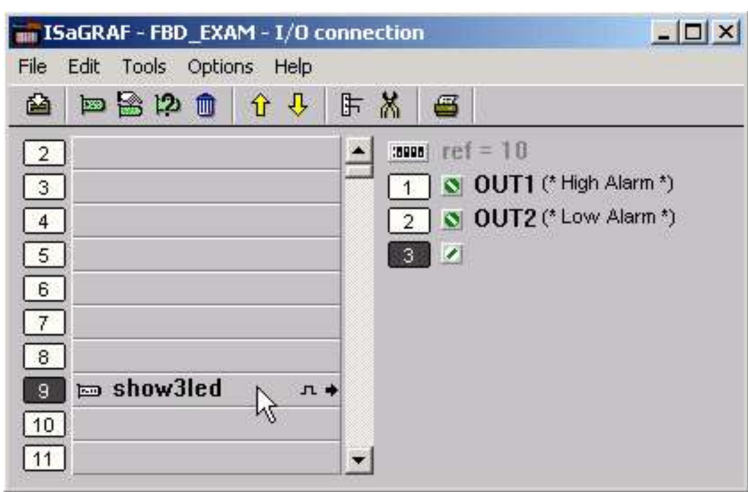


The top "A1" variable should connect to the "IN1" of the "> Greater Than" function block, the "5000" constant to the "IN2" of the "> Greater Than" function block, the bottom "A1" variable to the "IN1" of the "< Less Than" function block, and the "2000" constant to the "IN2" of the "< Less Than" function block.

Lastly, connect the "Q" of the "> Greater Than" function block to the "OUT1" variable, and the "Q" of the "< Less Than" function block to the "OUT2" variable.

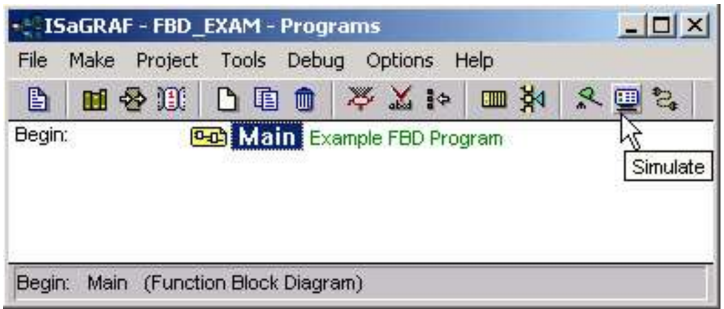
Connecting The I/O & Compiling The Project

Follow the same procedure as outlined in Section 2.1.2 and 2.1.3 for connecting the I/O and compiling the FBD example program. The "ISaGRAF I/O Connection" window should look like the example below.



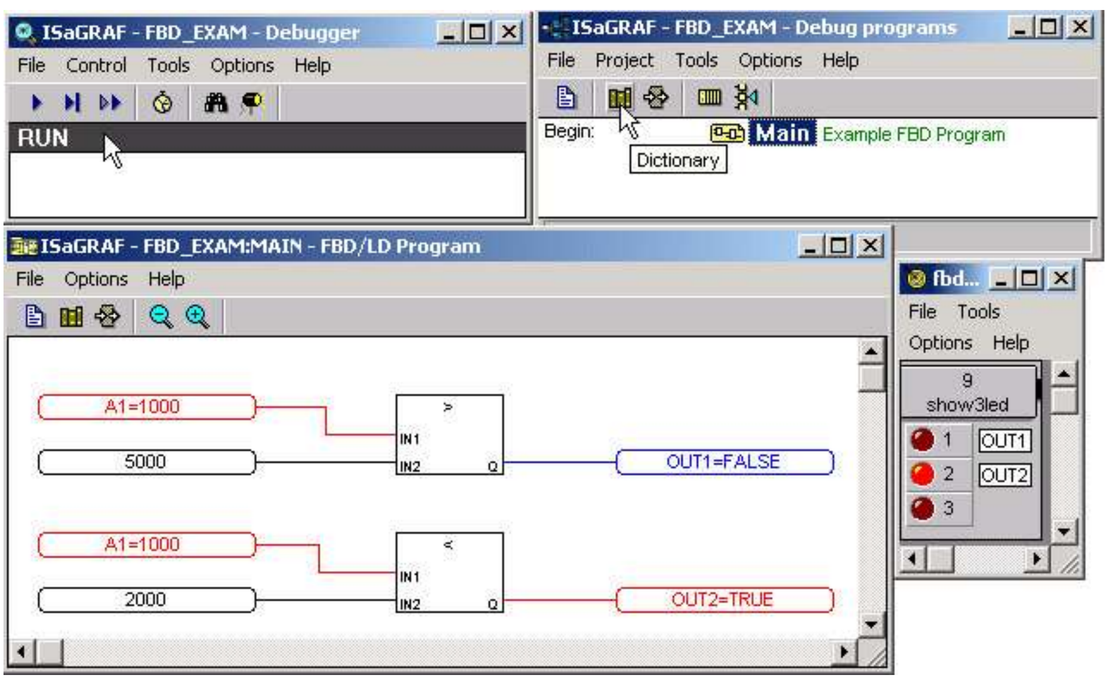
2.3.2: Simulating The FBD Program

You can now run the "Simulate" on the example FBD program by clicking on the "Simulate" icon in the "ISaGRAF Programs" window.

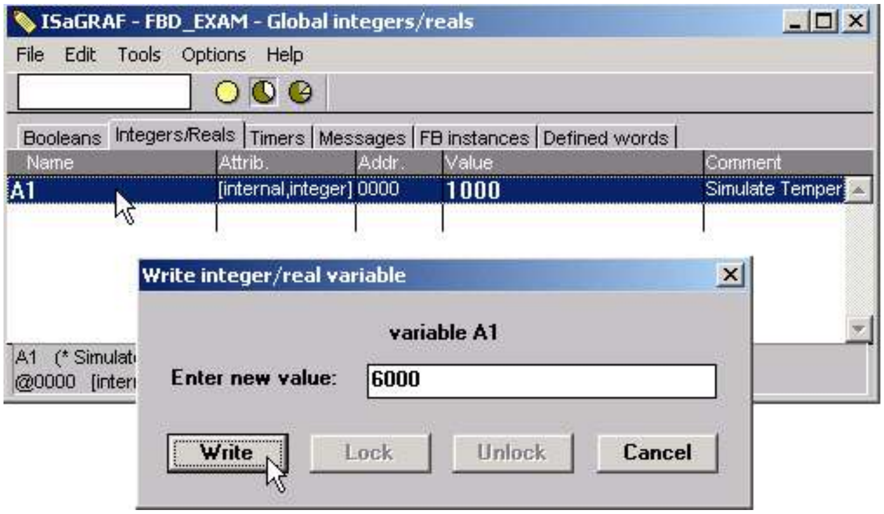


When you click on the "Simulate" icon the "ISaGRAF Debugger" window, the "ISaGRAF Debug Programs", and the "I/O Simulator" window will now open. If you double click on "MAIN" in the "ISaGRAF Debug Programs" window the "ISaGRAF FBD/LD Program" window will open showing the state of the program.

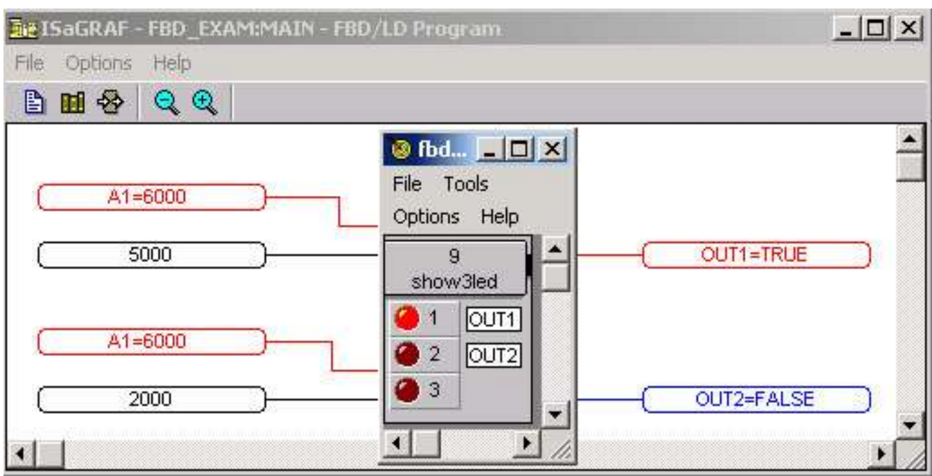
Notice that because the "A1" variable is less than 2000 (currently set to 1000 in the example below) that the "OUT2" output is currently true and the "OUT1" output is false.



To further test the example FBD program, click on the "Dictionary" icon in the "ISaGRAF Debug Programs" window to open the "Global Dictionary" window, and click on the "Integer/Real" tab. Click on the highlighted "A1" and the "Write Integer/Real Variable" will open.



Type in "6000" in the "Enter New Value" field and click on the "Write" button. Now the following changes will be observed.



You can now download the example FBD program to the ISaGRAF controller system. Follow the same procedure as outlined in Section 2.1.5 for downloading the program to the I-8xx7 controller system.

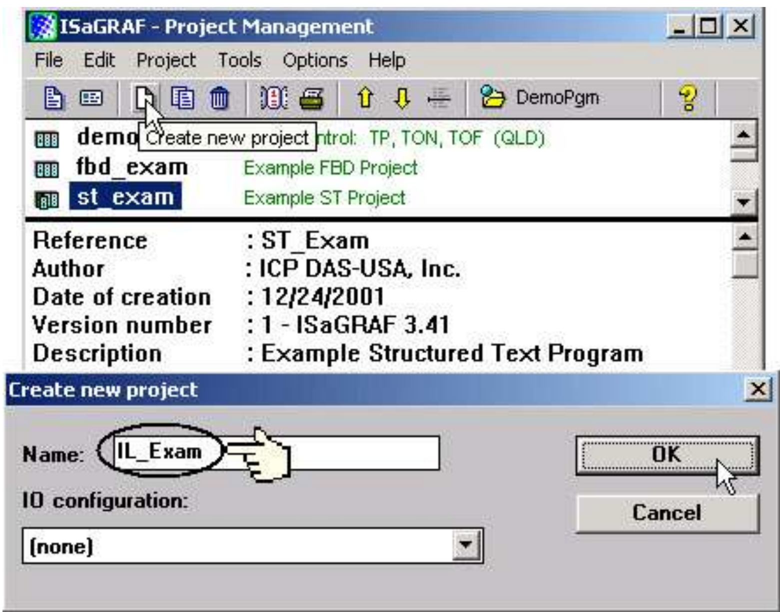
2.4: A Simple Instruction List (IL) Program

Instruction List (IL) programming is a low level programming language consisting of a list of instructions. Each instruction always relates to the **current result** (or **IL register**) and must begin on a new line and must contain an **operator.** The operator indicates the operation that must be made between the current value and the **operand**. The result of the operation is stored again in the result.

Instruction List (IL) programming requires adherence to a strict programming format that must be followed. Each instruction must begin on a new line, it must contain an **operator**, completed with optional modifiers and if necessary, for the specific operation, one or more operands, separated with commas (","). A **label** followed by a colon (":") may precede the instruction. If a comment is attached to an instruction, it must be the last component of the line. Comments must always begin with **(*** and end with ***).** The following is an example of a comment in IL; **(* place comment here *)**.

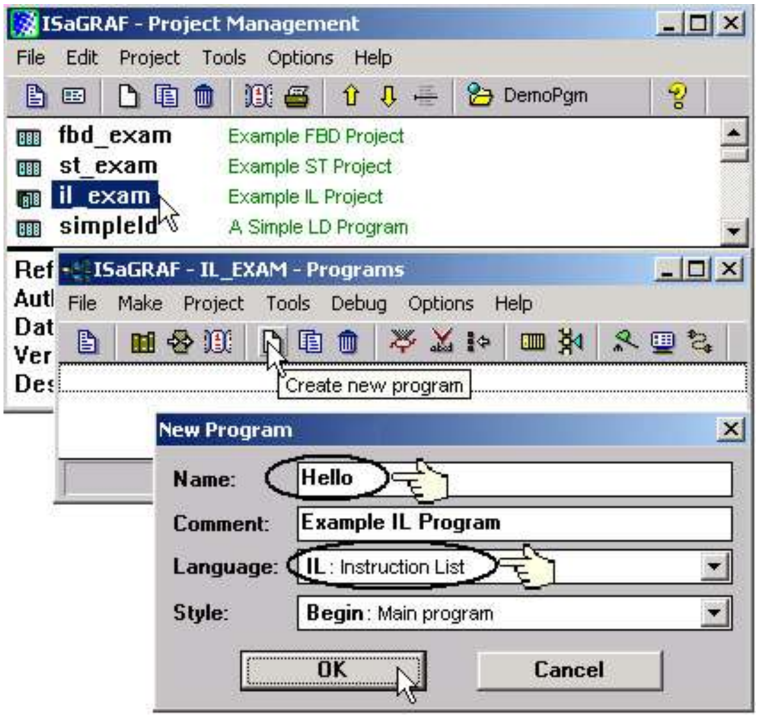
This section describes how to program an Instruction List (henceforth referred to as IL) program. This IL program has the same program specification as the ST and FBD program as outlined in Section 2.2 ans Section 2.3.

The first step is to create an IL project.

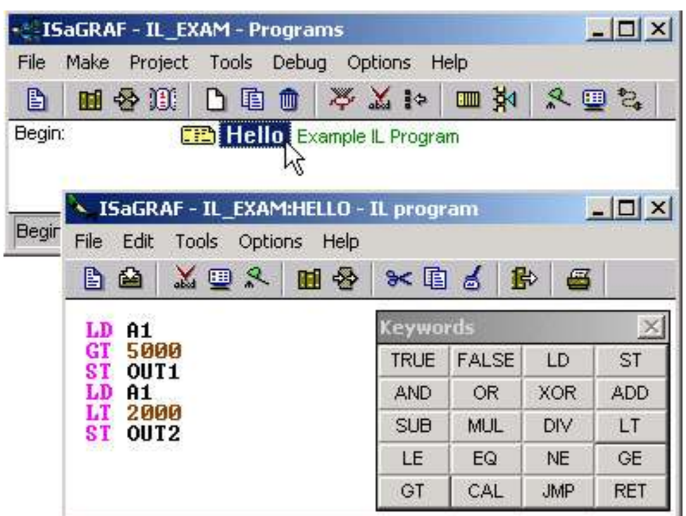


For the purpose of this example IL program I have created a new IL project name of "IL_Exam". Click on the "OK" button and the "ISaGRAF Project Management" window will appear with the new project name.

Double click on the "IL_Exam" name and the "ISaGRAF Programs" window will appear. Click on the "Create New Program" icon and the "New Program" window will appear. Enter "Hello" in the name field (and you can add a program comment if desired) and make sure to select "IL: Instruction List" from the language field, click on the "OK" button when you are done.



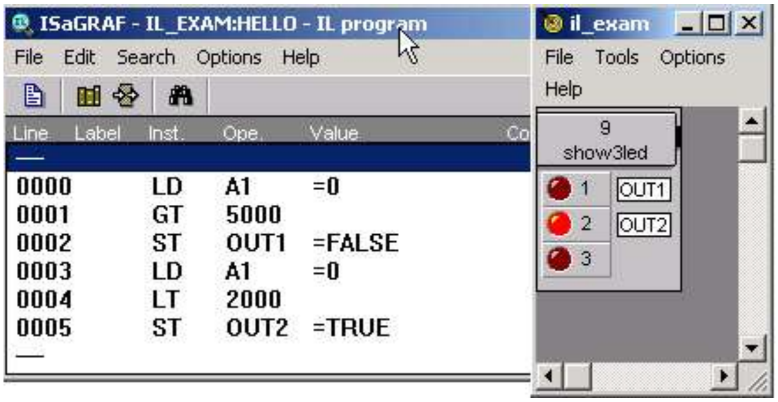
When you click on the "OK" button the "ISaGRAF Programs" window will open. Double click on "Hello" and the "ISaGRAF IL Program" window will open.



Declaring The Example IL Variables

This example IL program uses the same variables as the example FBD program, "OUT1", "OUT2" and the integer variable "A1". Refer to Section 2.1.1.3 "Declaring The Variables" for assistance. Use the same procedure for the "Connecting I/O" and "Compiling" the program as detailed in Section 2.1.2 and 2.1.3, and use the same procedure to "Simulate" the program as detailed in Section 2.3.2.

When you have connected the I/O and compiled the example IL program, click on the "Simulate" icon and the following window will appear.



Because the variable "A1" value is 0, "OUT1" is set to false and "OUT2" is set to true. Change the value of "A1" to a value greater than 5001 and you will see that "OUT1" is set to true and "OUT2" is set to false.

2.5: A Simple Sequential Function Chart (SFC) Program

A Sequential Function Chart (SFC) program is a graphical programming language used to describe **sequential operations**. The process is represented as a set of defined **steps**, linked by **transitions**. A **Boolean condition** is attached to each transition, and **actions** with the steps are detailed by using other languages such as ST, IL, LD and FDB.

An SFC program is a graphical set of **steps** and **transitions**, linked together by **oriented links**. Multiple connection links are used to represent divergences and convergences. Some parts of the complete program may be separated and represented in the main chart by a single symbol, call **macro steps**. The basic graphic rules for an SFC program are:

1. A Step CANNOT Be Followed By Another Step
2. A Transition CANNOT Be Followed By Another Transition

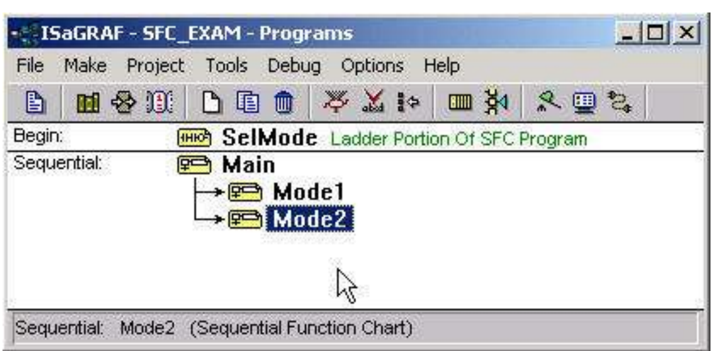
The basic components (graphical symbols) of the SFC programming language are: steps and initial steps, transitions, oriented links, and jumps to a step.

This section details how to build a Sequential Function Chart (henceforth referred to as SFC) program.

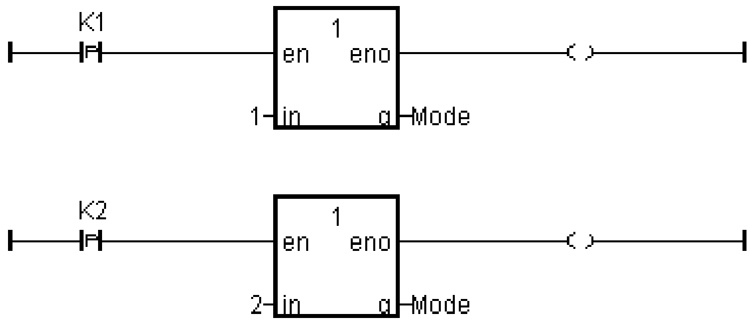
Example SFC Control Specification:
The following details the variables that will be used in our example SFC program.

Name	Type	Attribute	Description
OUT1	Boolean	Output	Output 1
OUT2	Boolean	Output	Output 2
K1	Boolean	Input	Mode 1 button input
K2	Boolean	Input	Mode 2 button input
TMR1	Timer	Internal	Switch time of output, initial value is "T#1s"
Mode	Integer	Internal	1 means mode1 , 2 means mode2, **initial value is 1**

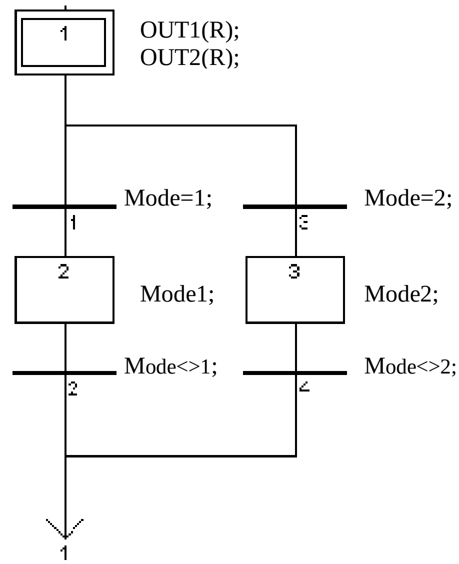
The SFC Program Outline:
When you have completed the "ISaGRAF Programs" window, it should look like the following:



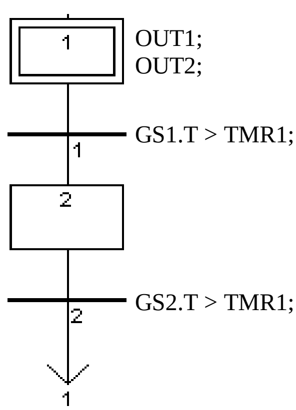
LD Program "SelMode"



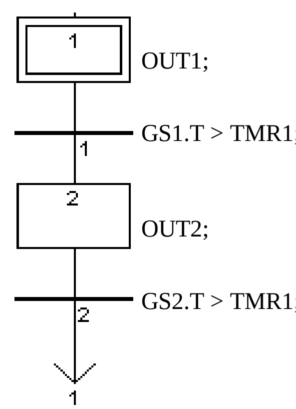
SFC Program "Main"



SFC Child Program "Mode1"



SFC Child Program "Mode2"

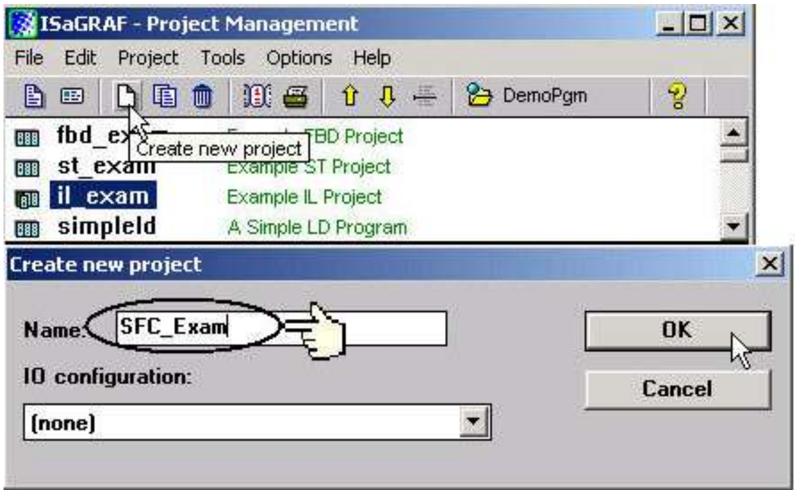


SFC Program Action:

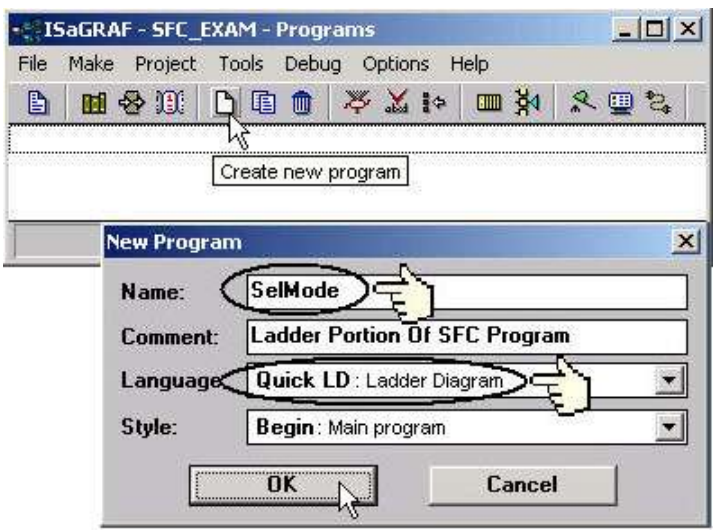
1. When "K1" is pressed, run the "Mode1" program.
2. When "K2" is pressed, run the "Mode2" program.

2.5.1: Programming The Example SFC Program

The procedure for creating the example SFC program is the same as outlined in Section 2.1. You must remember to declare the variables "K1", "K2", "OUT1", "OUT2", "TMR1" and "MODE". The following illustrates creating the new SFC project.

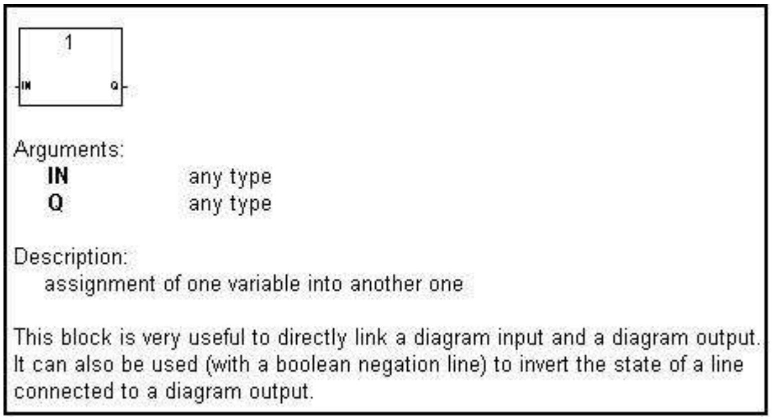


After creating the new SFC project, the next step is to create an LD program named "SelMode" as illustrated below.



IMPORTANT NOTE:

The example SFC program uses a function block that has not been used throughout the manual. We will be adding the "**1 Gain**" function block to our LD program.

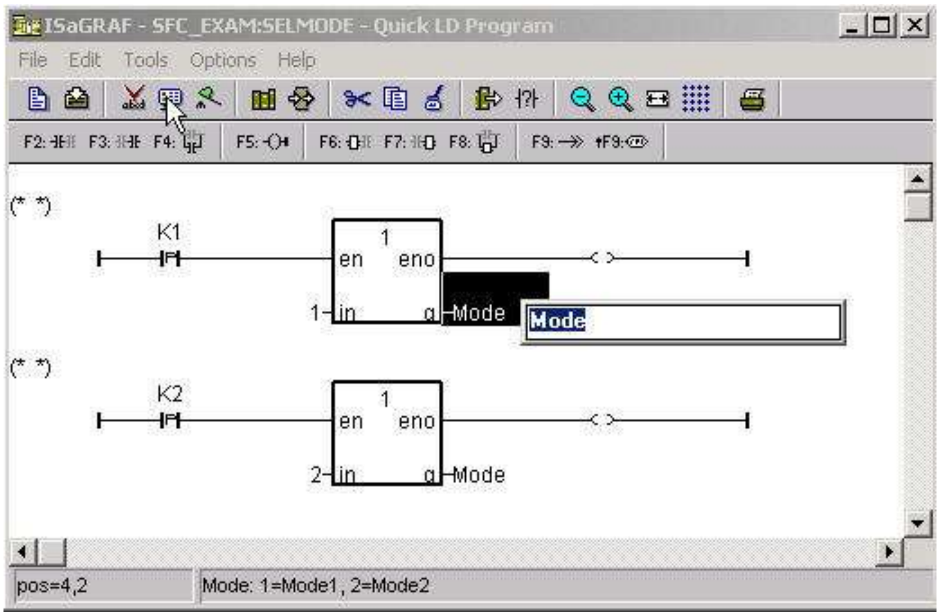


Arguments:
IN any type
Q any type

Description:
assignment of one variable into another one

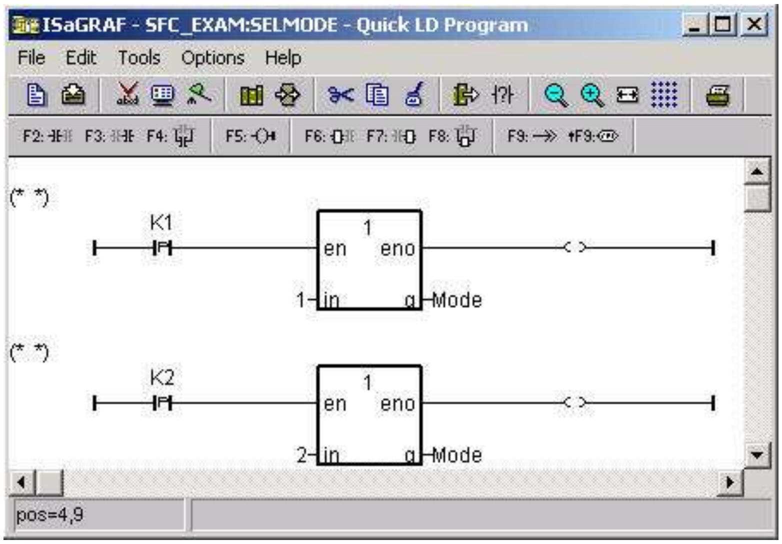
This block is very useful to directly link a diagram input and a diagram output. It can also be used (with a boolean negation line) to invert the state of a line connected to a diagram output.

Even though the "EN" (input) and "ENO" (output) arguments are not shown in the above example, they will be added when you place the "1 Gain" function block in the program.

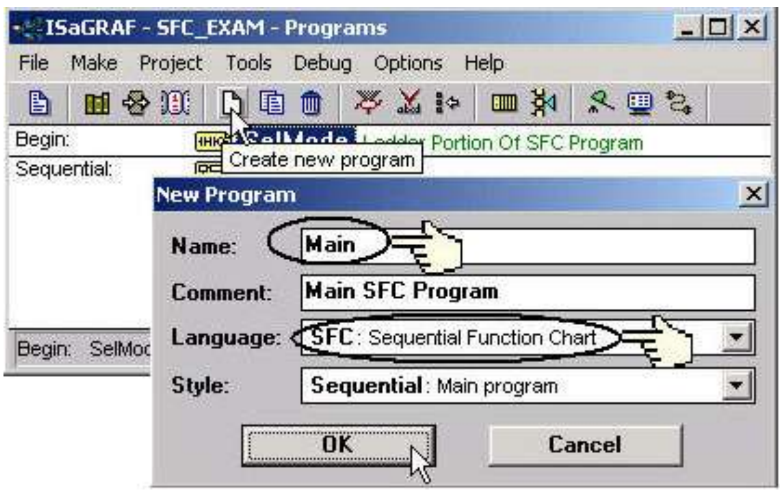


You will need to change the "K1" and "K2" contacts type to "P". The "P" contact (Positive) enables a Boolean operation between a connection line state and the rising edge of a Boolean variable. Place the cursor to the right of the "Q" and click once, then type in "Mode" for both lines of logic. Place the cursor to the left of the "IN" on the top "1 Gain" function block, click once and enter a "1". Do the same for the second LD line and enter a value of "2", then click once on the "Q" and enter in "Mode".

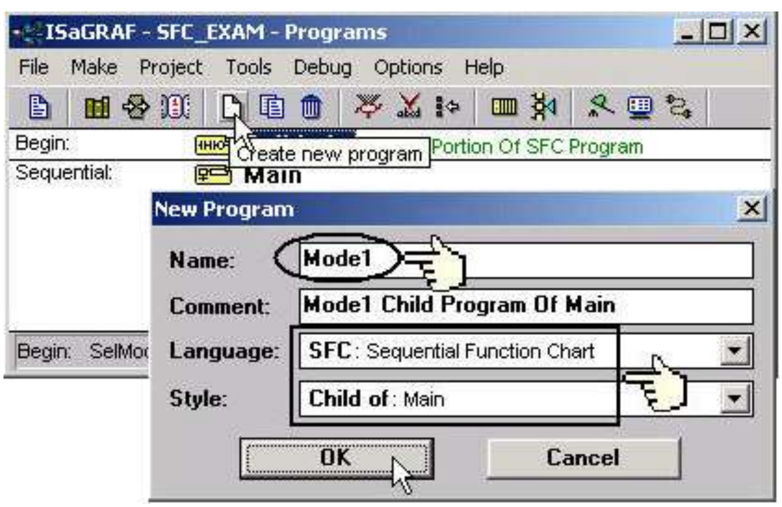
When you are finished editing the "ISaGRAF Quick LD Program" window it should look like the below example.



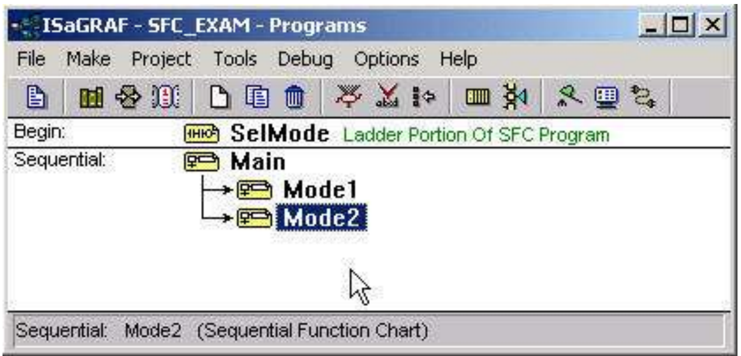
The next step is to create a new SFC program called "Main".



The next step is to create a "CHILD" program called "Mode1".

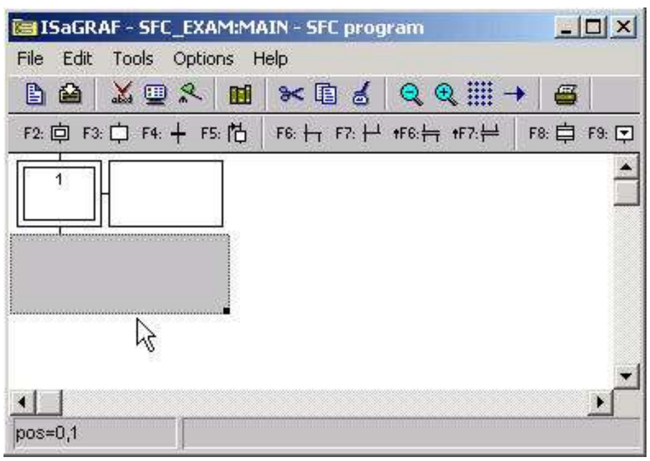


Follow the same procedure to create a second "CHILD" program named "Mode2". When you are completed the "ISaGRAF Programs" window should look as follows.

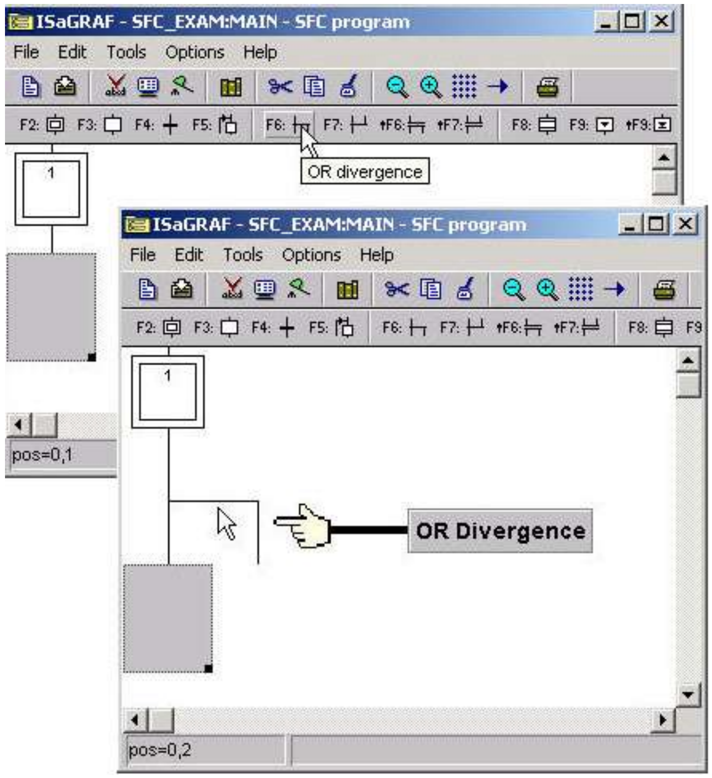


2.5.2: Editing The SFC Program

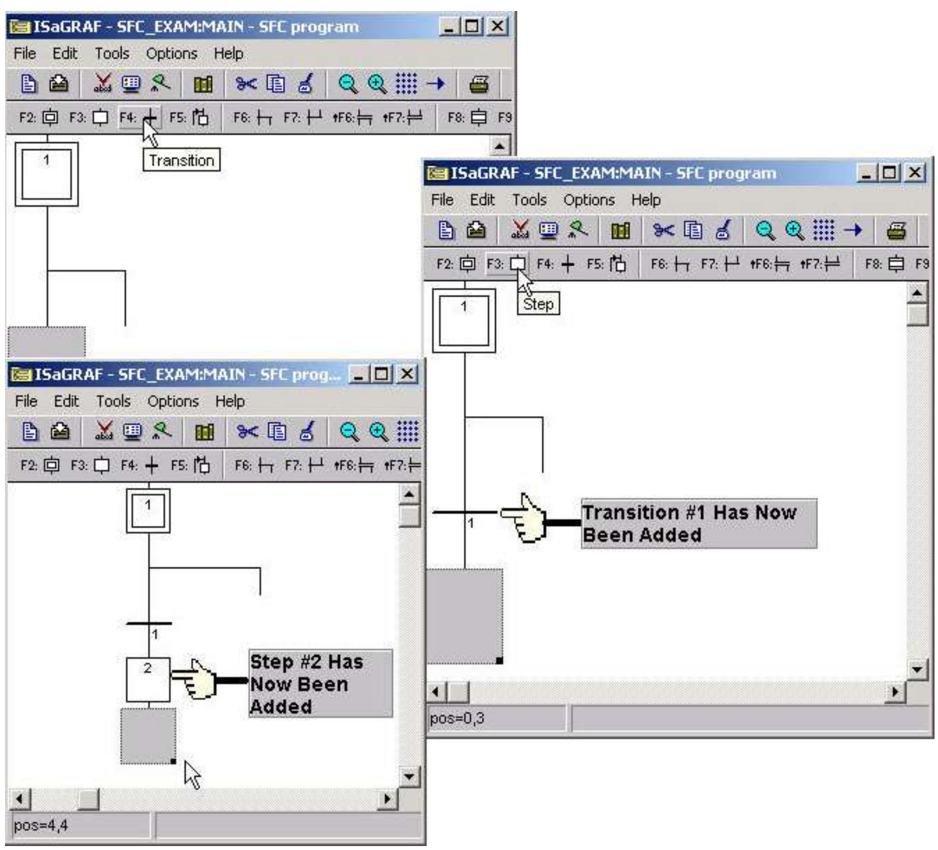
To begin editing the example SFC program double click on "Main" in the sequential portion of the "ISaGRAF Programs" window and the "ISaGRAF SFC Program" window will appear.



You will note an additional box to the right of the initial step box. This box will contain the code for each of the steps and transitions in the example SFC program. The "code box" is not required during the initial programming so you can to get rid of it temporarily by clicking on the black dot in the gray box area below the initial step and resize the window to approximately the size of the initial step box.

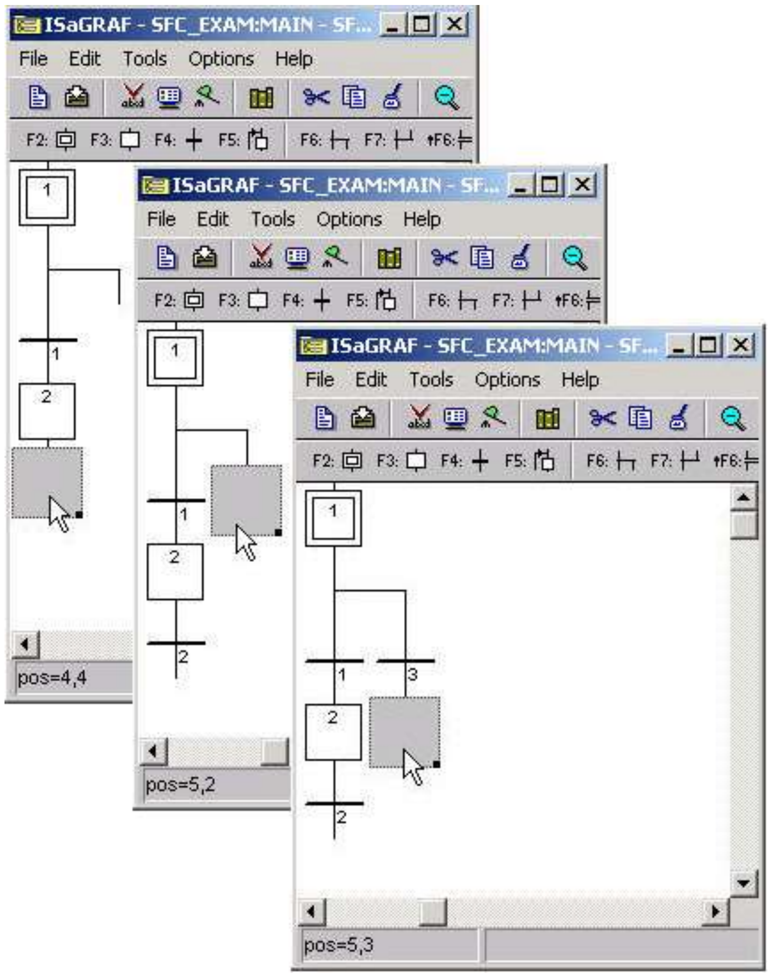


The gray box will move down automatically when you click on the "OR Divergence" icon. The next step is to click on the "Transition" icon to create "Transition 1" and then the "Step" icon to create "Step 2 as shown below.

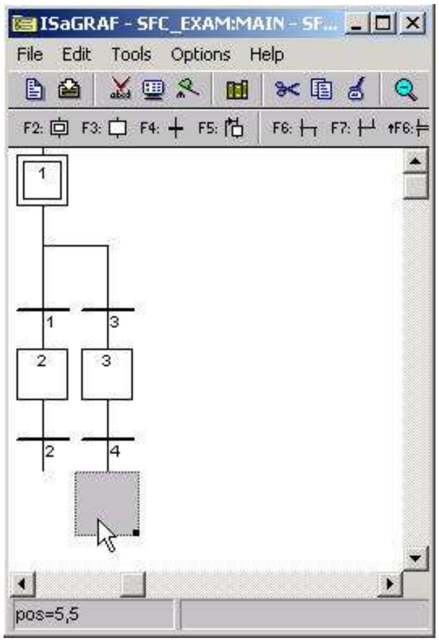


With the gray box below "Step 2" click on the transition button to add a second transition (transition #2) to the example SFC program. After adding the second transition below "Step 2", click directly below the "OR Divergence" so that the gray box is now placed there. Click on the transition icon again with the gray box below the "OR Divergence" to add a third transition (transition #3).

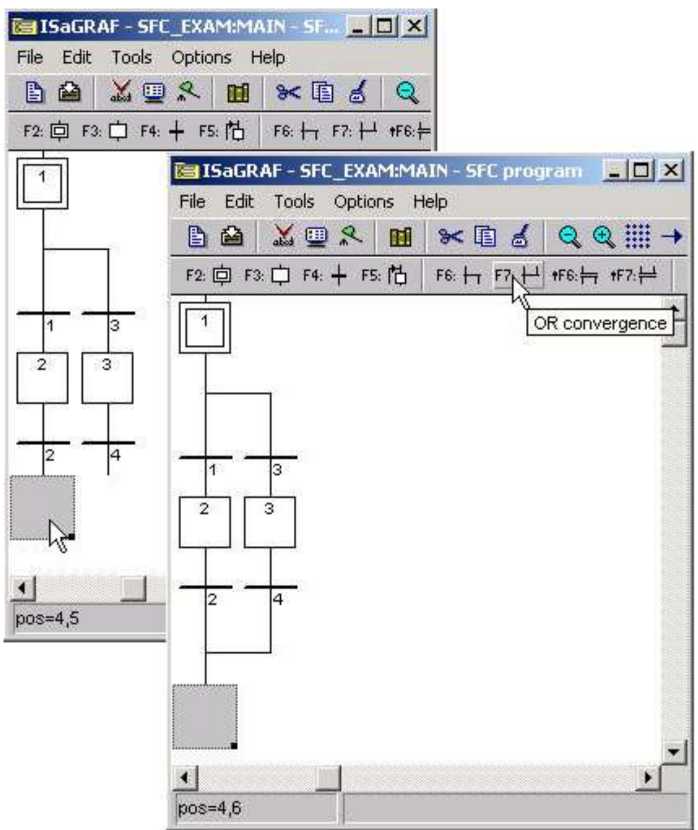
When you have completed these tasks your SFC program should now look like the third SFC picture below.



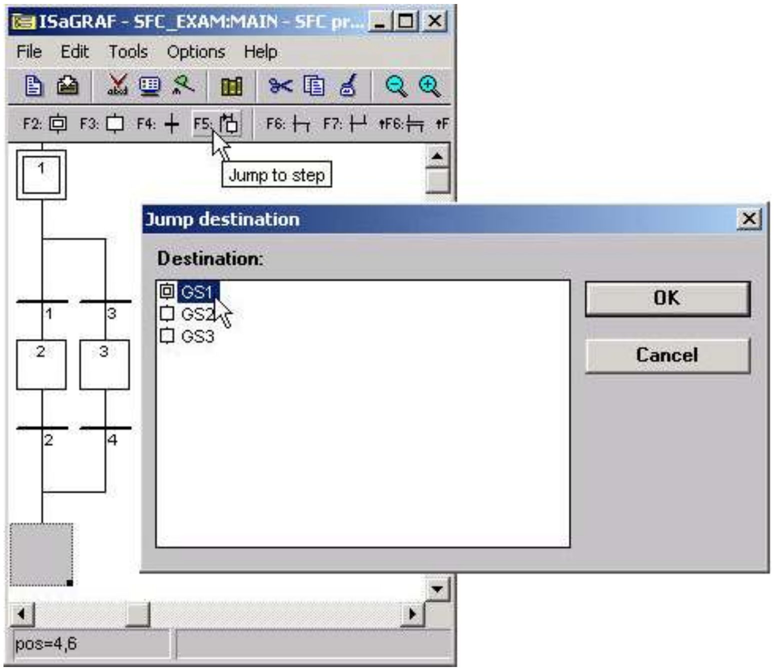
From where the gray box is currently click on the "Step" icon to add Step #3, and then with the gray box below the newly created step #3 click on the transition icon to add a fourth transition (transition #4) to the example SFC program. Your SFC program should now look like the below example.



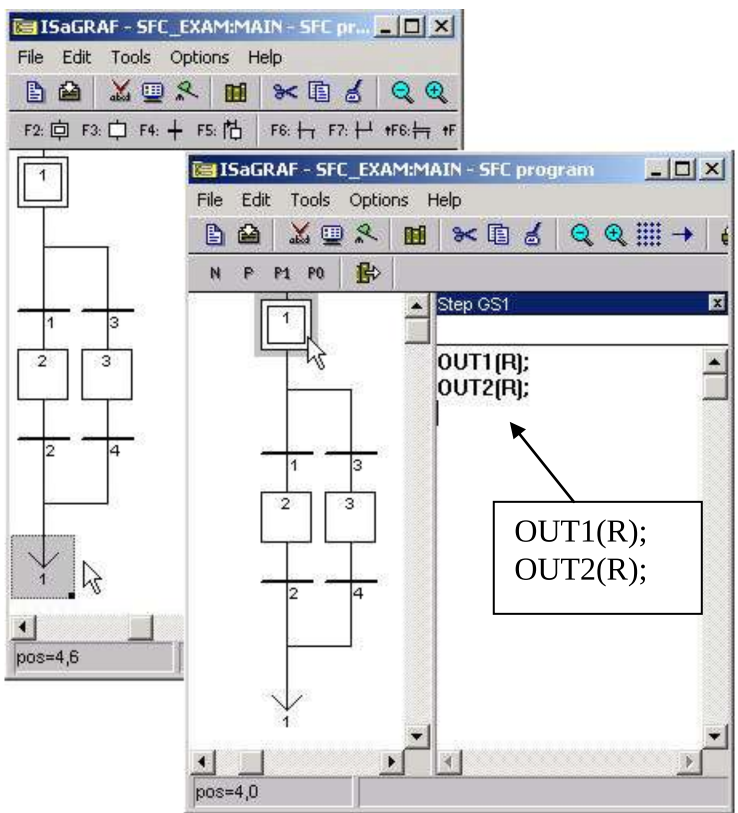
Now click the gray box below transition #2 and click on the "OR Convergence" (F7) icon.



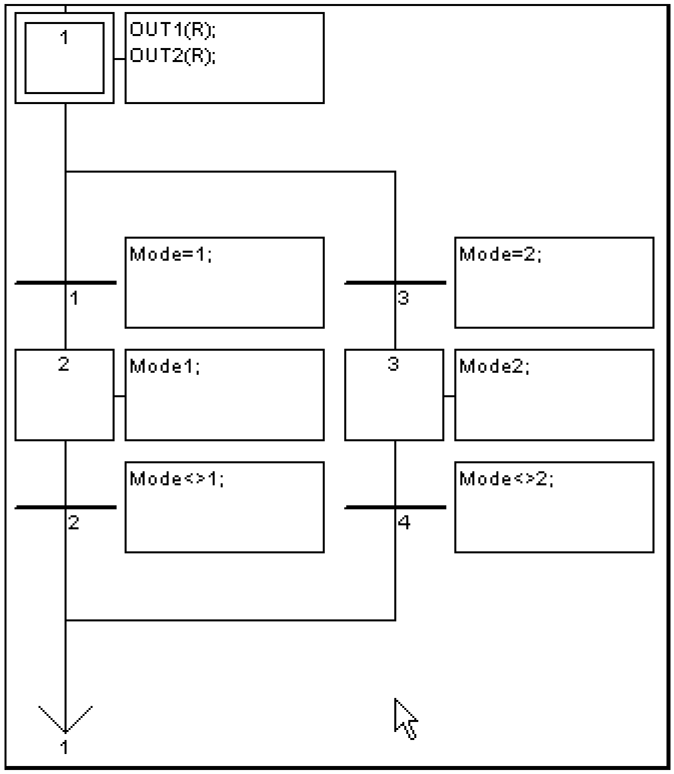
Now click on the "Jump To Step" (F5) icon, this will open the "Jump Destination" window. Double click on the "GS1" label in the "Jump Destination" window.



We have now finished programming the "Main" portion of the example SFC program. The next detail is to add the code for each of the steps and transitions. Double click on step #1 (initial step) and the "ISaGRAF SFC Program" window will open. Type the displayed text into the area shown below. This will associate the typed in code with the step #1. **REMEMBER** to type a semi-colon (":") at the end of each line of code.



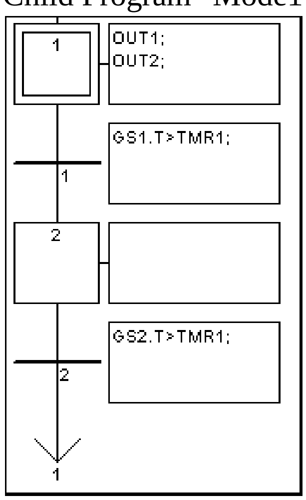
Using the same method as described above, double click on each transition and step and add the code for each item as shown below.



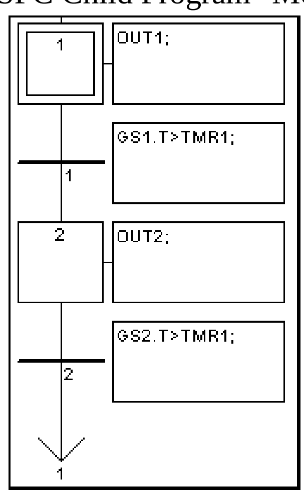
CONGRATULATIONS! You have now successfully programmed the "Main" section of the example SFC program (and the most time consuming).

The last portion of creating the example SFC program requires the creation and editing of the two "CHILD" programs. You program the "CHILD" programs using the exact same method as required for creating the "MAIN" program. When you are finished creating and editing the "CHILD" programs your two windows should look like the examples below.

SFC Child Program "Mode1"



SFC Child Program "Mode2"

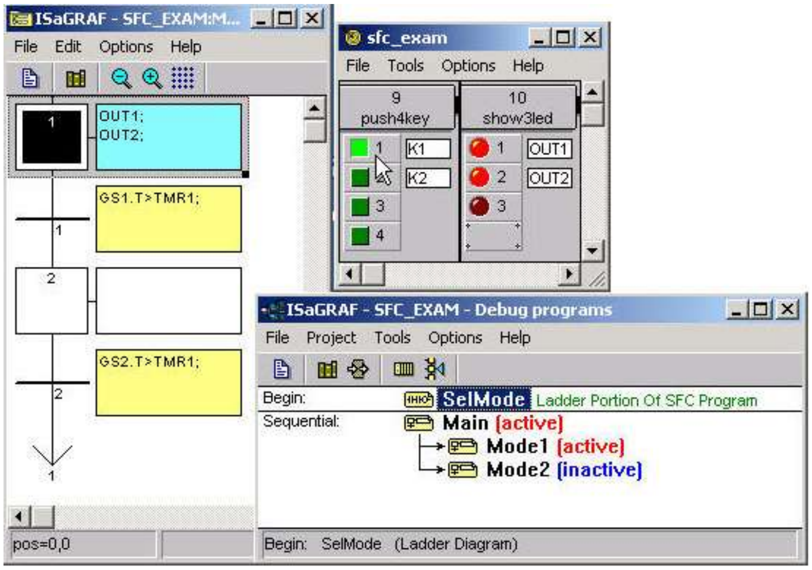


Final Details

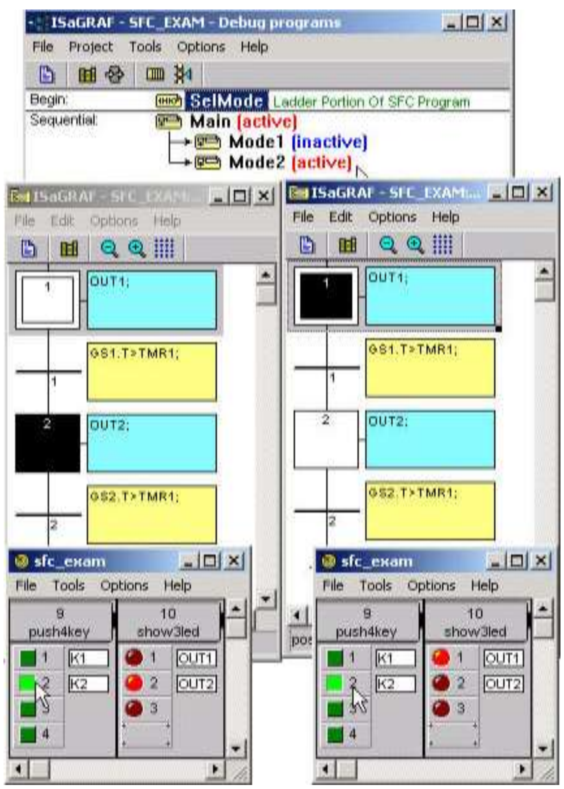
Remember that you must follow the same procedure for "Connecting I/O's" and "Compiling The Project" as detailed in Section 2.1.2 and Section 2.1.3.

2.5.3: Simulating The SFC Program

After you have successfully compiled the SFC program, you can now run the example SFC program in "Simulate" mode to observe how the two "CHILD" programs work within the "MAIN" SFC program. When "K1" is on, "Mode1" is true and both "OUT1" and "OUT2 turn on and off together, and "Mode2" is false.



When "K2" is on "Mode2" is true "OUT1" will turn on while "OUT2" is off and then they will alternate where "OUT2" will turn on and "OUT1" will be off, and "Mode1" is false.



2.6: Using Variable Array

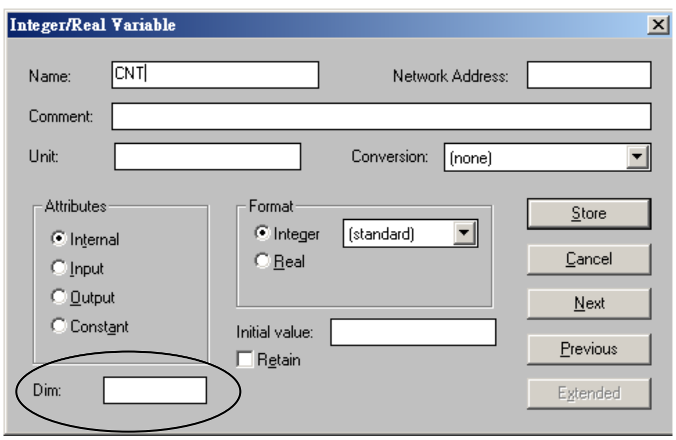
If your ISaGRAF Workbench is version of 3.4 or 3.5, you can declare variable array in the ISaGRAF dictionary, And then program them in each language (ST, LD,FBD, SFC, IL & FC).

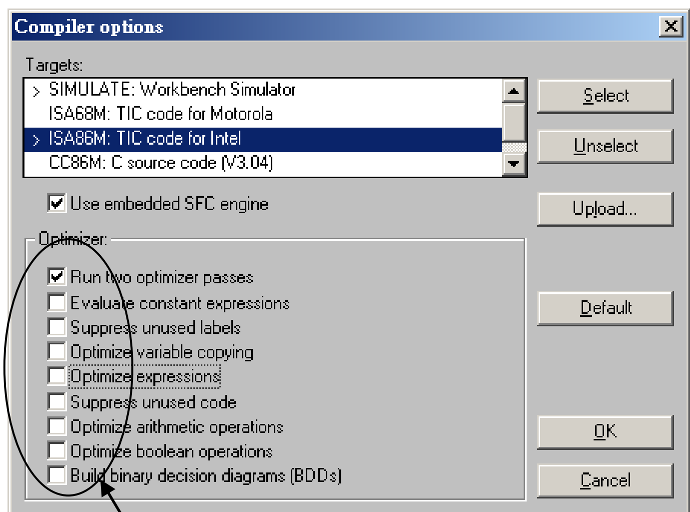
Please close all ISaGRAF windows first, and then add two extra lines in your ISaGRAF workbench root “EXE” directory, normally in the c:\isawin\exe.

In the “C:\ISAWIN\EXE\ISA.INI”, adds two extra raws on the top of this file.

[DEBUG]
arrays=1

And then re-open the ISaGRAF workbench, you will find there is one more “Dim” column in the ISaGRAF dictionary. The number entered can be 1 to 255. However **it is very important, please always declare the proper number you want. The larger “Dim” number, the larger memory is consumed**. (Such as the I-7188EG/XG, I-8xx7 memory-constrained controller)





If using “Variable Array” in the program, please DO NOT check the 2nd , 7th , 8th and 9 th Optimizer options, or the value of the Variable array will be incorrect. Recommend to check only the 1st – “Run two optimizer passes” option.

The index of the variable array is always starting from 0. For example, if you declare an integer “CNT” with “Dim” = 10 , the varable array will be CNT[0..9] , that is the item can be used is CNT[0], CNT[1], …, CNT[9].

How to program variable array ?

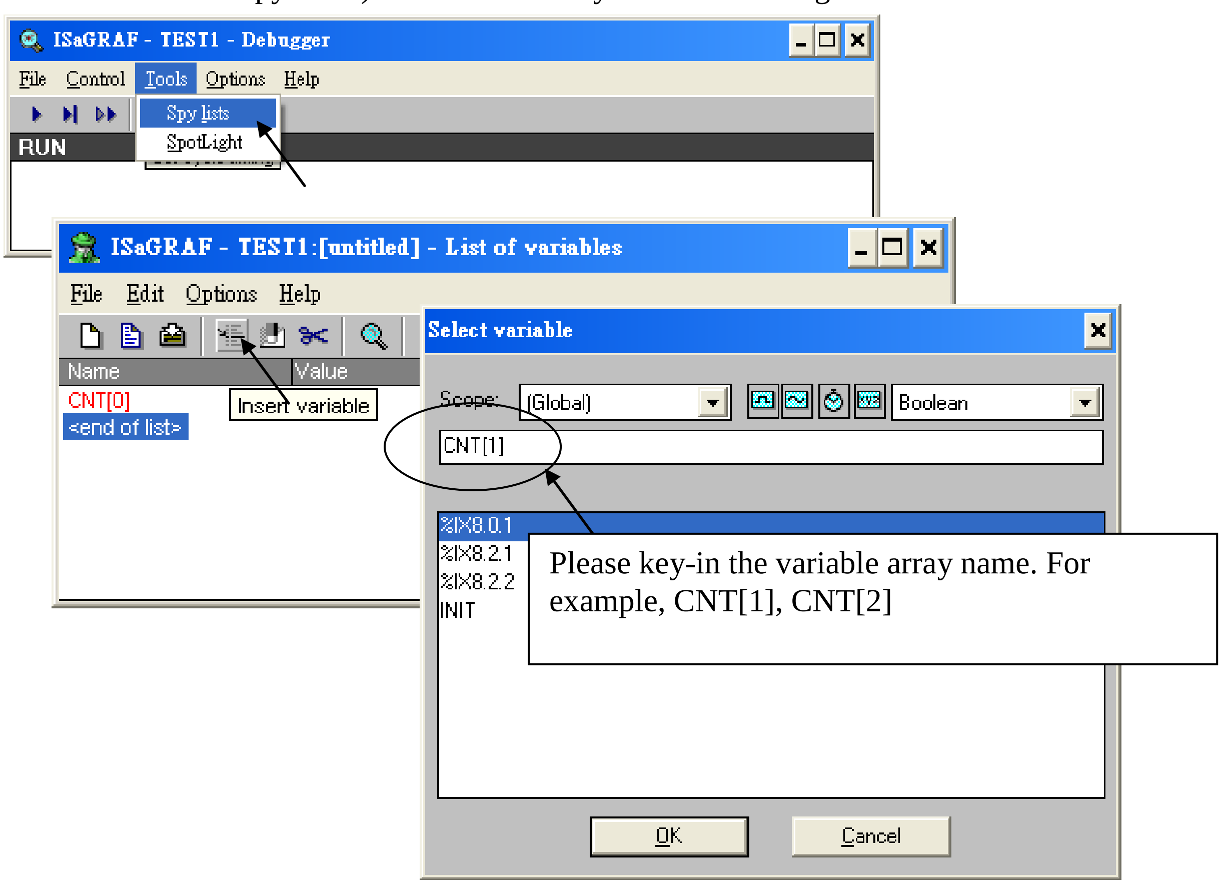
For example, the below ST code can assign an intial value of 100 to 109 to CNT[0] to CNT[9]

```
(* INIT is declared as an internal Boolean with initial value of TRUE *)
(* CNT is declared as an integer array with "Dim" = 10 *)
(* ii is declared as an internal integer *)

IF   INIT   THEN
   INIT  :=  FALSE  ;      (* only do it once at 1st PLC scan cycle *)
   For  ii  :=  0  to  9  do
     CNT[ii]  :=  100 + ii ;
   End_For  ;
END_IF  ;
```

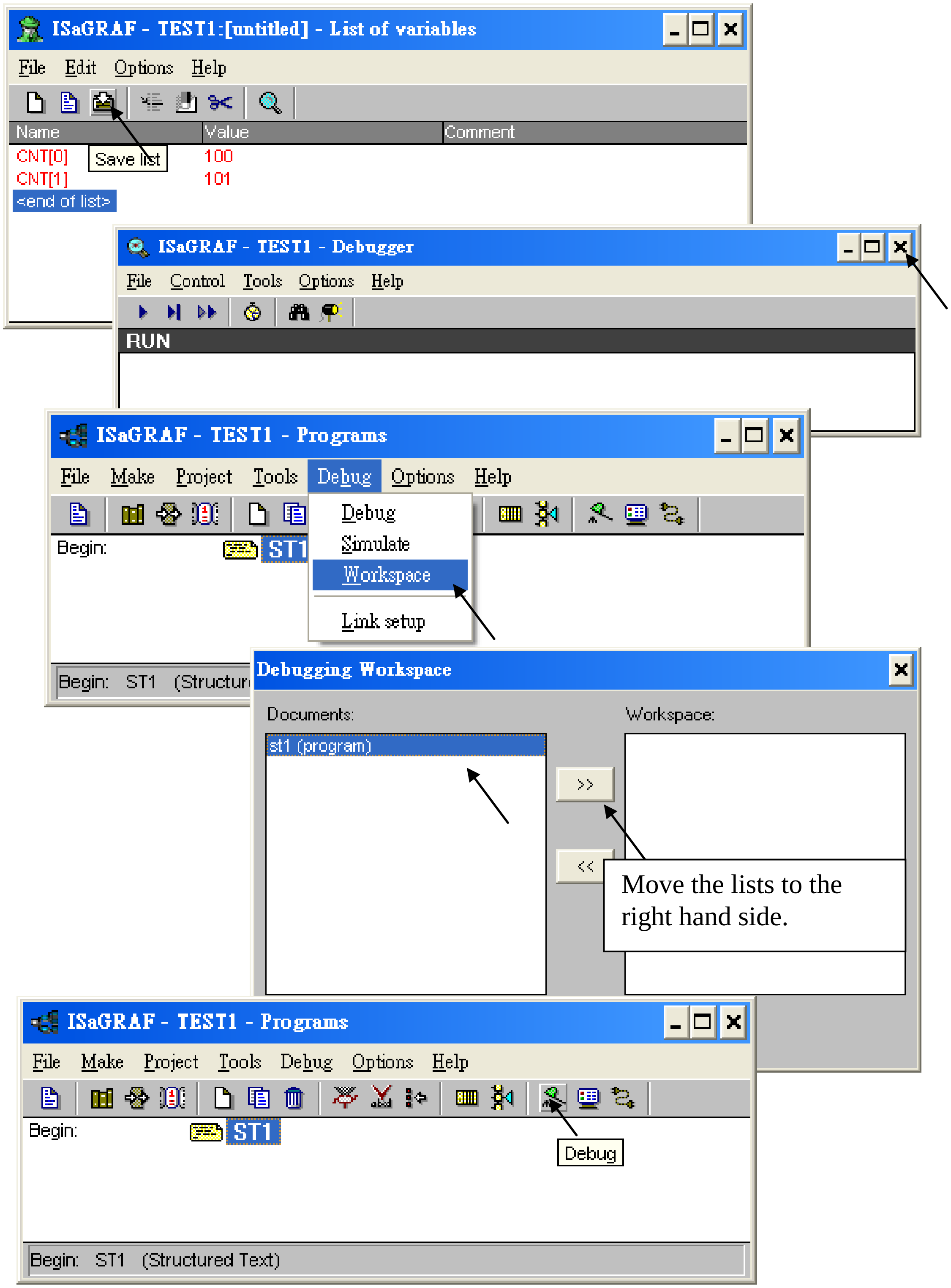
Note: Please do not exceed the "Dim" number of variable array. For example please do not program at CNT[10] or CNT[11] in the above example since the CNT 's dimension is only 10 , CNT[0], CNT[1], …, CNT[9]. There is no CNT[10], CNT[11], …

How to debug variable array ?

After you compile your ISaGRAF project, you may download the project to the controller or simulate it. Please open the "Tools" - "Spy lists" on the "Debugger" windows. (please refer to section 9.12 for more information about "Spy lists"). Insert the name you want to debug.

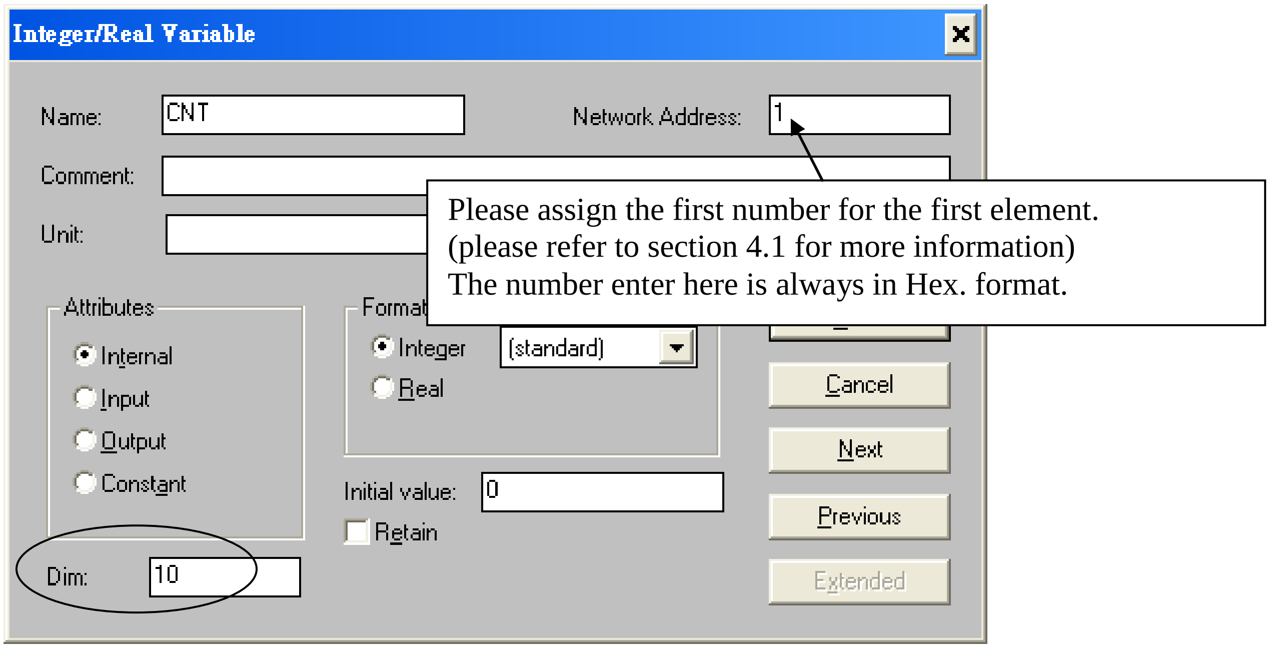


Please remember to save the “spy list” to a name, for example – “list1” and then put it into the workspace. You will find the “list1” will automatically pop-up when you open the debugger.



2.6.1 Assign Network Address No. To Variable Array

To assign Modbus Network address number (that is used to exchange the data with the SCADA or HMI, please refer to chapter 4) to the varable array. Please assign the network address number to the first element, For example, No. = 1 assigned to CNT[0]. And the using “S_MB_ADR( )” function as below.



And then using “S_MB_ADR” to assign the other network address number for each element. For example,

1. **Assign continuous Network No** = 1 ,2 ,3, …,10 to CNT[0], CNT[1], CNT[2], …, CNT[9]

(* INIT is declared as internal Boolean with initial value at TRUE *)
(* TMP is declared as internal Boolean *)

IF INIT THEN
INIT := FALSE ; (* only do it at 1st PLC scan *)
TMP := S_MB_ADR(1, 10, 0) ; (* assign 10 elements starting at No.=1, continuous No. *)
END_IF ;

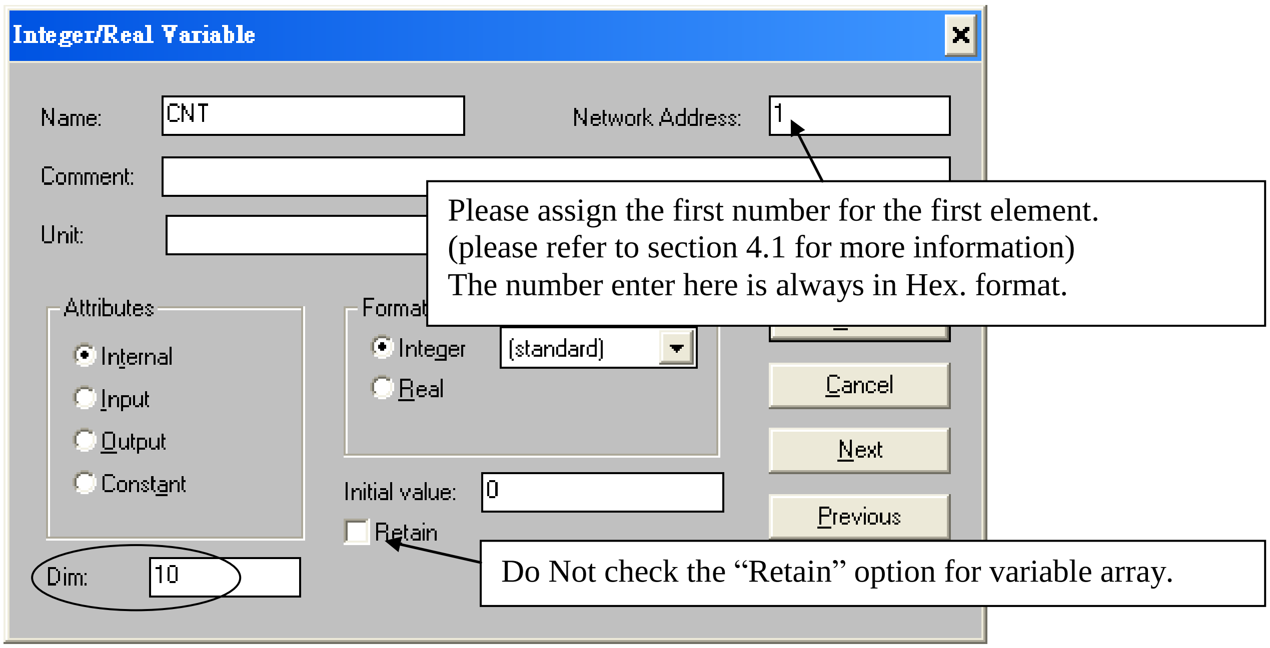
2. **Assign Jumping Network No** = 1 , 3 , 5, …,19 to CNT[0], CNT[1], CNT[2], …, CNT[9]

(* INIT is declared as internal Boolean with initial value at TRUE *)
(* TMP is declared as internal Boolean *)

IF INIT THEN
INIT := FALSE ; (* only do it at 1st PLC scan *)
TMP := S_MB_ADR(1, 10, 1) ; (* assign 10 elements starting at No.=1, jumping No. *)
END_IF ;

2.6.2 Setting Variable Array As Retained Variable

To set “varable array” as retained data, please assign the network address number to the first element, For example, No. = 1 assigned to CNT[0]. And then using “Retain_A( )” function as below. Please refer to chapter 10.1 for more information about the “New retained function”.



For example, setting integer variable array CNT[0..9] as retained data in the integer retained memory starting from 20 , 21 , … to 29.

(* INIT is declared as internal Boolean with initial value at TRUE *)
(* TMP is declared as internal Boolean *)

```
IF   INIT   THEN

   INIT   :=  FALSE  ;  (* only do it at 1st PLC scan *)

   TMP  :=  Retain_A( ‘N’ , 1 , 10 , 20 ) ;
  (*
     1st  parameter :   'B' : boolean , 'N' : Integer , 'F' : Real , 'T' : Timer
     2nd  parameter :  Network address No. for the 1st element of the "Variable Array".
     3rd parameter :  1 - 255 , number of element in the “variable array” to be assigned as
                      retained data.
     4th parameter :  starting retained address for this “variable array”.
        7188EG/XG+X607/608, I-8xx7+S256/512 : ‘B’ & ‘T’ is 1 to 256 , ‘N’ & ‘F’ is 1 to 1024 .
        WinCon-8xx7/8xx6+S256/512 : ‘B’ & ‘T’ is 1 to 1024 , ‘N’ & ‘F’ is 1 to 4096
  *)

END_IF  ;
```

Chapter 3. Establishing I/O Connections

Before you can operate an ISaGRAF program with the iP-8xx7, I-8xx7, I-7188EG/XG, µPAC-7186EG, µPAC-5xx7, WP-8xx7, WP-5xx7, VP-25W7, XP-8xx7-Atom-CE6 and XP-8xx7-CE6 controller, you must make sure that the I/O Library has been installed. If you haven't done so already, install it as outlined in Section 1.2 "Installing The ICP DAS Utilities For ISaGRAF".

Please visit below Web site to get more information about I-8K and I-87K I/O boards.
http://www.icpdas.com/products/PAC/i-8000/8000_IO_modules.htm

The following I-8K parallel I/O can be used in slot 0 ~ 7:

Note:
The I/O boards are divided into low profile and high profile I/O, and the name with "W" is high profile version. For example, the I-87057 is low profile I/O, the I-87057W is high profile I/O. The iP-8xx7, WP-8xx7, XP-8xx7-CE6, XP-8xx7-Atom-CE6, VP-2117 and VP-25W7/23W7 only support high profile I/O plug in the slot 0 ~ 7. If using low profile I/O in these PAC, it may cause communication error, retain variables exception or incorrect I/O behavior.

I-8K parallel I/O boards can be plug in PAC's slot 0 through slot 7 (and slot 1 through slot 7 of XP-8xx7-CE6, XP-8xx7-Atom-CE6 controller). The I-87K4, I-87K5, I-87K8, I-87K9, RU-87P4 and RU-87P8 expansion unit supports only I-87K serial I/O moduls (better to use high profile modules). They don't support I-8K parallel I/O modules.

Digital Input / Output	I-8037W, I-8040W, I-8040PW, I-8041W, I-8042W, I-8046W I-8050W, I-8051W, I-8052W, I-8053PW I-8054W, I-8055W, I-8056W, I-8057W, I-8058W , I-8060W, I-8063W I-8064W, I-8068W, I-8069W
Analog Input (V , mA)	I-8017HW (8-Ch. Differential / 16-Ch. Single-end) iP-8xx7: max. I-8017HW Sample rate is about 125Hz (8-Ch.), 62 Hz (16-Ch) WP-8xx7, XP-8xx7-CE6, VP-25W7: max. I-8017HW Sample rate is about 200Hz (8-Ch.), 100 Hz (16-Ch). It depends on the PLC scan time. The bigger scan time, the samller sample rate.
Analog Output (V , mA)	I-8024W (4 Ch.)
Counter / Frequency Input	I-8084W (4/8 Ch counter or Frequency) I-8088W (8-Ch. PWM output
Motion control, Encoder	I-8092F, I-8094, I-8094F: only supprot XP-8347-CE6 and XP-8747-CE6 **,** XP-8147-Atom-CE6, XP-8347-Atom-CE6 and XP-8747-Atom-CE6. Please refer to http://www.icpdas.com/faq/isagraf.htm > FAQ-132. I-8091W (2 Axes. Pulse output), I-8090W (Encoder): only support iP-8xx7, WP-8xx7, VP-25W7 and XP-8xx7-CE6. I-8093W(Encoder): only support iP-8xx7, WP-8xx7, VP-25W7 and XP-8xx7-CE6.
CANopen Master	I-8123W: only support XP-8xx7-CE6, XP-8xx7-Atom-CE6**,** WP-8xx7, VP-25W7 and VP-23W7. Please refer to http://www.icpdas.com/faq/isagraf.htm > FAQ-145.
FRnet Master card	I-8172W (2-port FRnet), it has to be used with FRnet I/O. Please visit

	http://www.icpdas.com/products/Remote_IO/frnet/frnet_list.htm or http://www.icpdas.com/faq/isagraf.htm FAQ-082
Expansion serial communication ports	I-8112iW (2-Ch. RS-232) , I-8114iW (4-Ch. RS-232) I-8142iW (2-Ch. RS-485 / 422) , I-8144iW (4-Ch. RS-485 / 422) (I-8xx7, iP-8xx7 can expand max. 16 ports at slot 0 through slot 3) (WP-8xx7, VP-25W7 can expand max. 10 ports at slot 0 through slot 2) (XP-8xx7-CE6 and XP-8xx7-Aton-CE6 can expand max. 28 ports at slot 1 through slot 7)

The following I-87K parallel I/O can be used in slot 0 ~ 7:

Digital Input / Output	I-97037W, I-87040W, I-87041W, I-87046W, I-87051W, I-87052W, I-87053W, I-87053PW, I-87053W-A5, I-87053W-AC1, I-87053W-E5 I-87054W, I-87055W, I-87057W, I-87057PW, I-87058W, I-87059W, I-87061W, I-87063W, I-87064W, I-87065W, I-87066W, I-87068W, I-87069W, I-87069PW
Anlog Input (V , mA)	I-87016W, I-87017W, I-87017W-A5, I-87017RW, I-87017RCW, I-87017DW, I-87017ZW, I-87019PW, I-87019RW, I-87019ZW
Analog Output (V , mA)	I-87024W, I-87024DW, I-87024RW, I-87028CW
Multifunction	I-87026W
Temperature input	I-87005W, I-87013W, I-87015W, I-87015PW, I-87018W, I-87018RW, I-87018PW, I-87018ZW, I-87019RW, I-87019ZW
Counter / Frequency, PWM	I-87082W, I-87084W, I-87088W (PWM)
HART Master	I-87H17W: only support XP-8xx7-CE6, XP-8xx7-Atom-CE6, WP-8xx7, VP-25W7 and VP-23W7. Please refer to http://www.icpdas.com/faq/isagraf.htm > FAQ-136.
Vibrating Wire Input	I-87089W (Please refer to http://www.icpdas.com/faq/isagraf.htm > FAQ-091).

Important:
The I/O boards are divided into low profile and high profile I/O, and the name with “W” is high profile version. For example, the I-87057 is low profile I/O, the I-87057W is high profile I/O. The iP-8xx7, WP-8xx7, XP-8xx7-CE6, XP-8xx7-Atom-CE6, VP-2117 and VP-25W7/23W7 only support high profile I/O plug in the slot 0 ~ 7. If using low profile I/O in these PAC, it may cause communication error, retain variables exception or incorrect I/O behavior.

I-8K parallel I/O boards can be plug in PAC‘s slot 0 through slot 7 (and slot 1 through slot 7 of XP-8xx7-CE6 controller). The I-87K4, I-87K5, I-87K8, I-87K9, RU-87P4 and RU-87P8 expansion unit supports only I-87K serial I/O moduls (better to use high profile modules). They don't support I-8K parallel I/O modules.

Please refer to “Getting Started Manual” of I-7188EG/XG or µPAC-7186EG for information about X-xxx board. or
http://www.icpdas.com/products/PAC/i-8000/getting_started_manual.htm
http://www.icpdas.com/products/PAC/i-o_expansion/x_list.htm

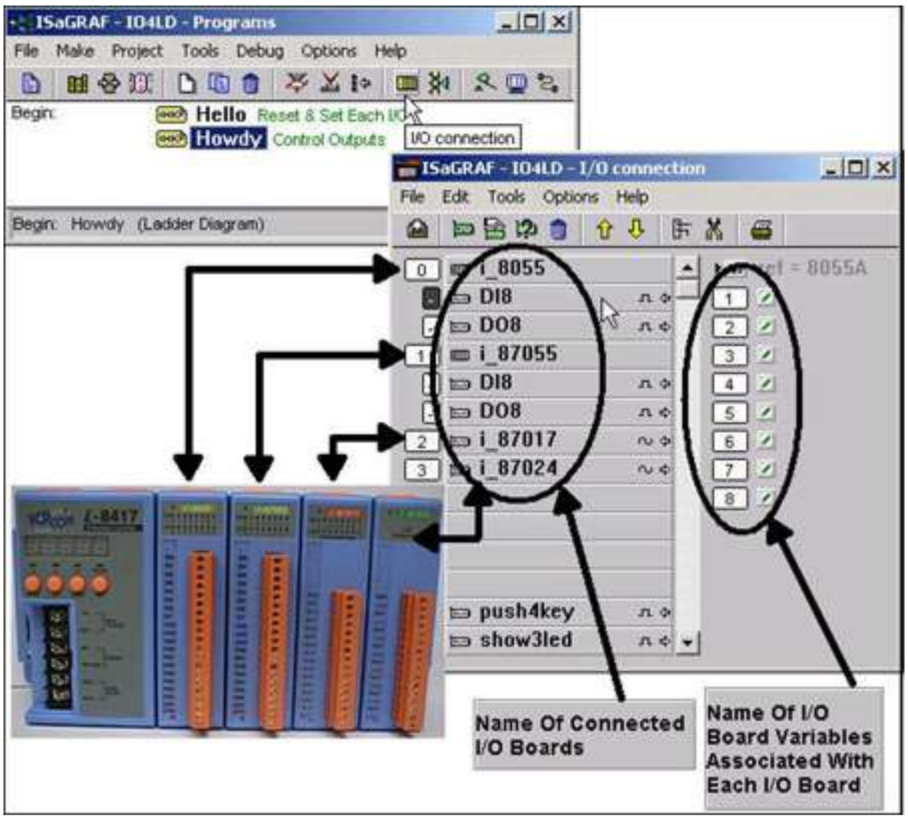
Please visit below Web site to get more information about I-8K and I-87K I/O boards.
http://www.icpdas.com/products/PAC/i-8000/8000_IO_modules.htm

3.1: Linking I/O Boards To An ISaGRAF Project

To begin connecting I/O boards to an ISaGRAF project you must first link the I/O boards to the ISaGRAF program. The numbers on the left of the "I/O Connections" window indicate the slot number. Slots 0 through 7 are used ONLY for **real** ISaGRAF PAC series I/O boards (**Note: Slot 1 through 7 for XP-8xx7-CE6, XP-8xx7-Atom-CE6, W-8xx7**). Slots 8 and above can be used for "virtual" I/O boards such as the "Push4Key" and "Show3Led" functions. For I-7188EG/XG, µPAC-7186EG, slot 0 is for X-xxx serial I/O boards (such as X-107), slot 1 & above are for others.

In this example I/O connection we are using the I-8417 controller system that has the following boards installed:

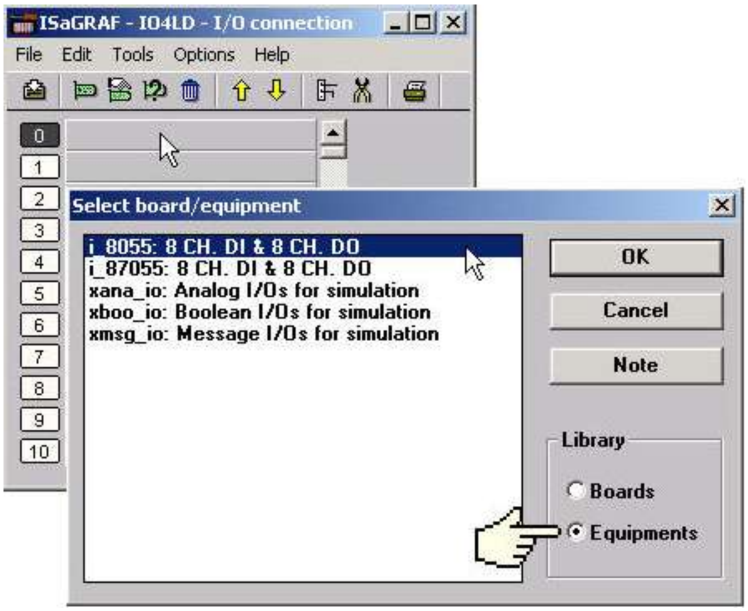
Slot 0: I-8055 Board (8 digital inputs & 8 digital outputs)
Slot 1: I-87055 Board (8 serial inputs & 8 serial outputs)
Slot 2: I-87017 Board (8 channel analog input)
Slot 3: I-87024 Board (4 channel analog output)
Slot 8: "Push4Key"
Slot 9: "Show3Led"



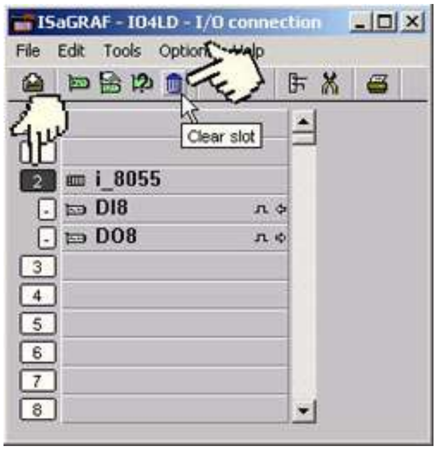
3.1.1: Linking I/O Boards

With the "I/O Connection" window open double click on the slot that you want to connect an I/O board to. The "Select Board/Equipment" window will open, scroll to the name of the I/O board that you want to associate with the particular slot.

The ISaGRAF 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 I-8055 digital I/O board that has 8 digital inputs and 8 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.

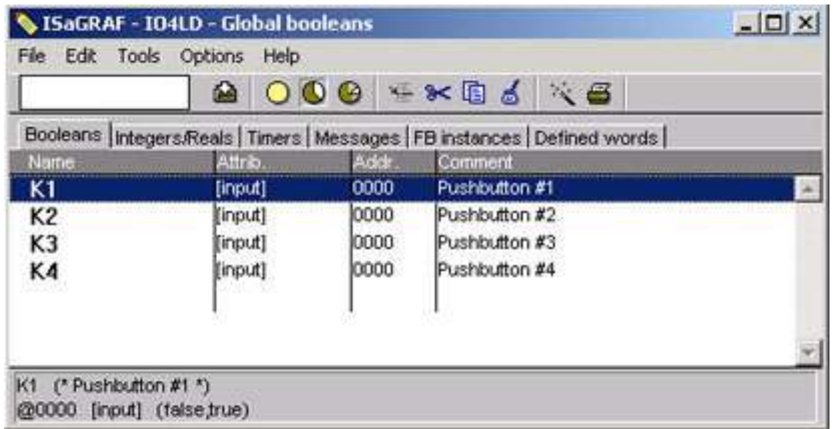


3.1.2: Linking Input & Output Board Variables

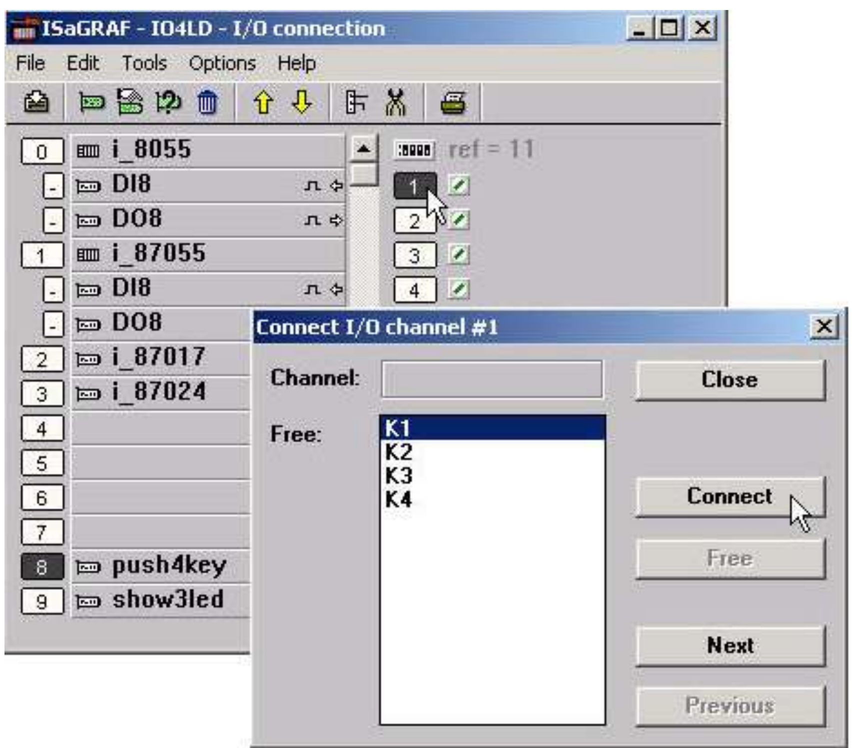
All of the input and output board "variables (or names)" must be linked (connected) in the "I/O Connection" window. Click on the slot you wish to link the attribute to, then double click on the channel (or I/O point name) number on the right hand portion of the "I/O Connection" window. Lastly, choose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

IMPORTANT NOTE

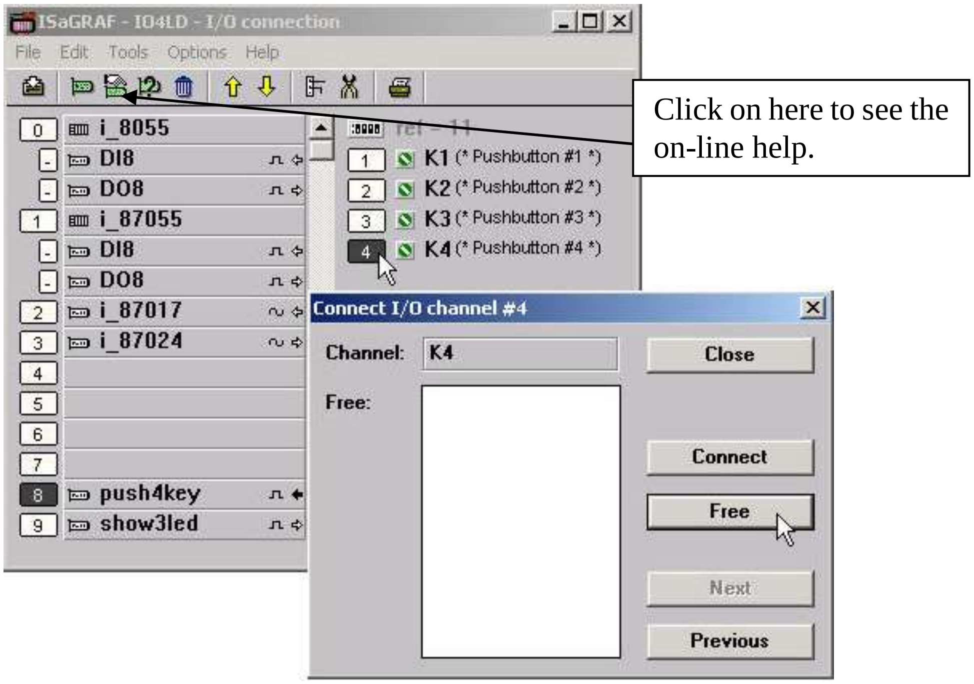
Remember that before you can assign any input or output, you must FIRST declare the variable in the "ISaGRAF Global Variables" window as shown below.



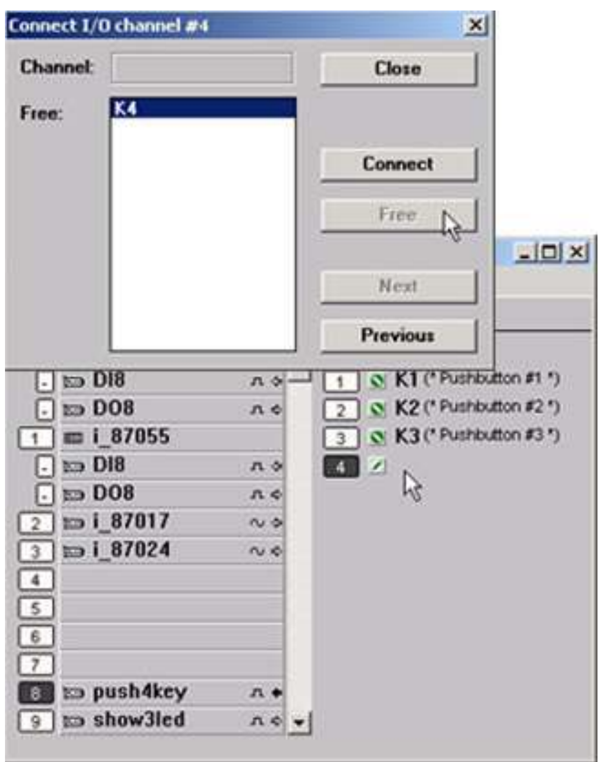
Click once on slot 8, then double click on "1" on the right hand side of the "ISaGRAF I/O Connection" window. With the "Connect I/O Channel #1" window now open, click on the "Connect" button to create the link between the variable "K1" and channel number 1 of the "Push4Key" input.



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

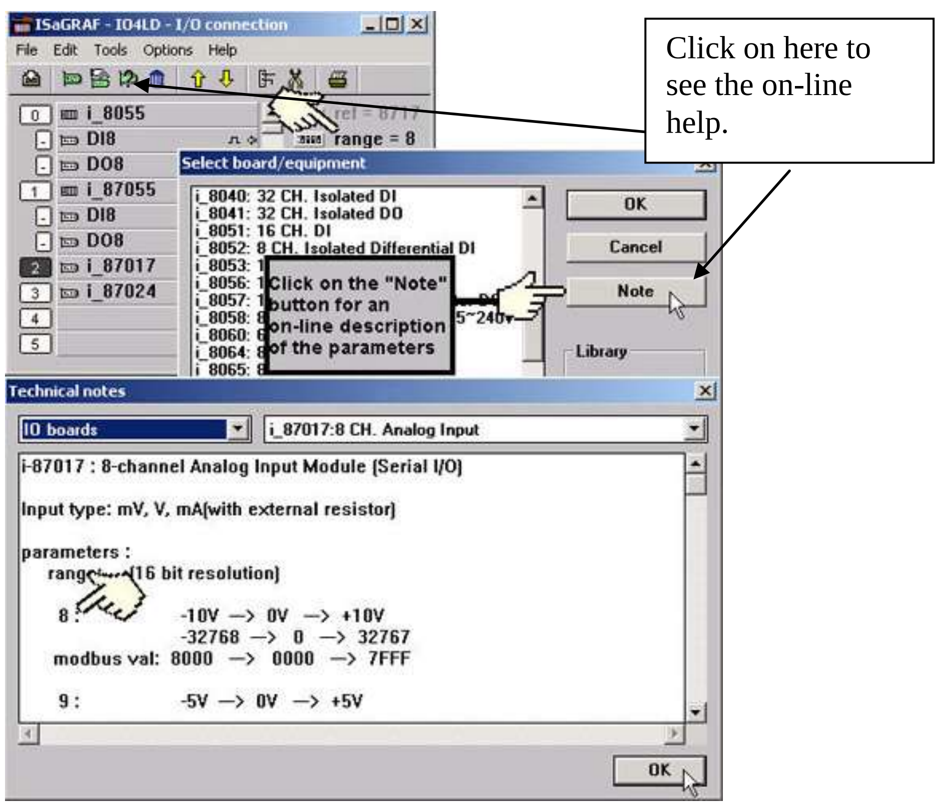
Note:

1. The I-87017ZW is better than I-87017 and I-87017C in industrial application. The I-87017ZW can be set as 10-Ch. Differential input or 20-Ch. Single-end input by a jumper. For better usage, you can set the proper “Range” setting for each channel. If it set as 10-Ch., you can measure 0 ~ 20 mA, 4 ~ 20 mA, +/- 20 mA without an external resistor of 125 ohms, just switch the “Jumper”. (When using I-87017W, I-87019ZW, I-87017DW to measure the Current, it requires an external resistor of 125 ohms.)
2. The I-87018ZW or I-87019ZW is better than I-87018 and I-87018W in industrial application and more precise measured values.

3.2.1: Setting “range” parameter and conversion functions for analog IO board

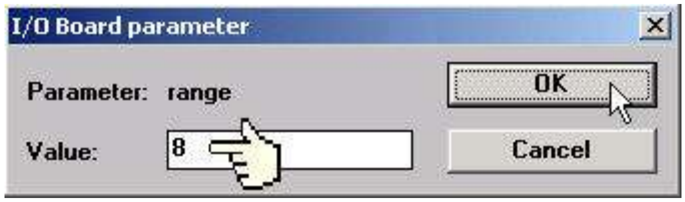
The method to connect analog type I/O boards to the controller system is very similar to that of connecting digital I/O boards.

The ONE main difference is that you MUST define one parameter that defines the “range” for the analog board so it will operate as expected.



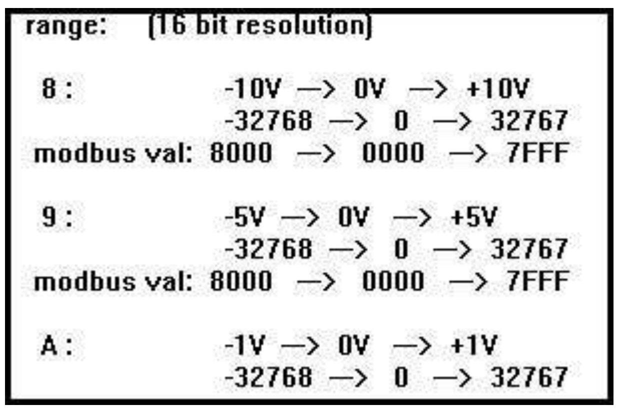
To modify the analog board "Range" parameter, click on the word "Range" in the "ISaGRAF I/O Connection" window and the "I/O Board Parameter" window will open.

Enter in the correct "Range" parameter for your particular analog board application.



The below table provides information on several of the possible options for the "Range" parameter. Note that the default value is set to "8", which means you can interface to a –10v to +10v signal with a range value of –32768 to 32767. Changing the value of "Range" parameter to "9" means you can interface to a –5v to +5v signal with a range value of –32768 to 32767.

Note that if you set the "Range" parameter to "A" you will be interface to a –1v to +1v signal with a range value of –32768 to 32767. This range value can be very helpful in analog applications that require a great deal of resolution over a very small range (typically temperature) control.



```
range:    (16 bit resolution)

 8 :            -10V  --> 0V   --> +10V
                -32768 --> 0   --> 32767
modbus val: 8000  -->  0000  --> 7FFF

 9 :            -5V --> 0V  --> +5V
                -32768 --> 0   --> 32767
modbus val: 8000  -->  0000  --> 7FFF

 A :            -1V --> 0V  --> +1V
                -32768 --> 0   --> 32767
```

Please refer to **Appendix D - "Table of The Analog IO Value"** for more information for several different types of analog boards and their respective ranges.

Note: Analog conversion functions:

The below functions are useful if user want to convert analog I/O value to application engineering value. For example, to convert 4 to 20 mA to become 0 to 1000 Psi, user can use function - "A4_20_To" to do it. Please refer Appendix A.4 for more description.

1. **A4_20_To** : to convert 4 to 20 mA analog value to become engineering value (32-bit Real)
2. **V0_10_To** : to convert 0 to 10 V analog value to become engineering value (32-bit Real)
3. **To_A4_20** : to convert engineering to become 4 to 20 mA analog output Value (Integer).
4. **To_V0_10** : to convert engineering to become 0 to 10 V analog output Value (Integer).
5. **BIN2ENG** : to convert Analog value (Integr, value should be in –32768 to +32767) to become engineering value (Integer, value should be in –32767 to +32768)

3.2.2: Setting special “range” parameter of temperature input board to get clear “Degree Celsius” or “Degree Fahrenheit” input value

ICP DAS provides many temperature input modules as below.
(The I-7000 series I/O is only for RS-485 remote I/O, CAN’T be use in slot 0 ~7)

With “broken-line detection” or called “wire opening detection”:

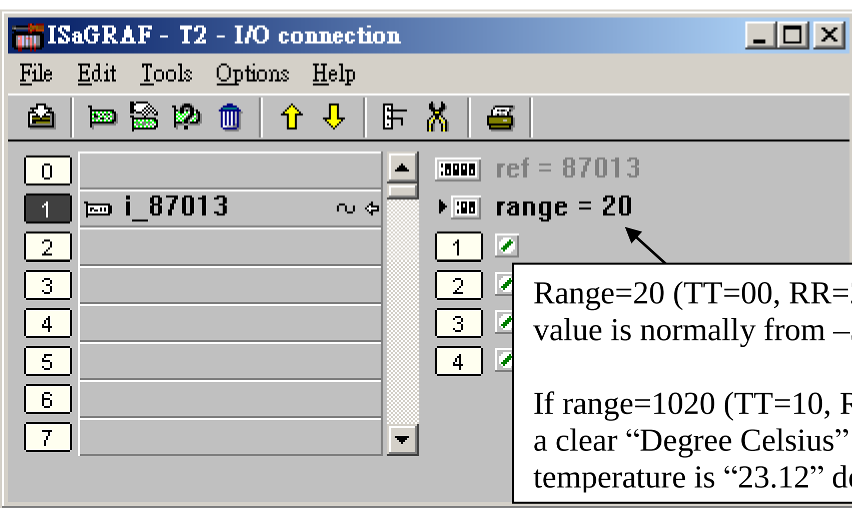
Thermocouple type:	I-87018RW, 87018PW, 87018ZW, 87019RW, 87019PW, 87019ZW, I-7018R, 7018BL, 7019, 7019R, 7018Z
RTD type:	I-87013W, 87015W, 87015PW, I-7013, 7015, 7033
Thermister type:	I-87005W, I-7005

Without “broken-line detection:

Thermocouple type:	I-87018, 7018, 7018P

The “range” parameter of temperature IO board can be “standard setting” or “special setting”.

For example, I-87013: 4 channels RTD input module. Its range can be
20 : Platinum 100, a=0.00385, -100 ~ +100 degree Celsius
…
2F : Platinum 100, a=0.003916, -200 ~ +200 degree Celsius



Range=20 (TT=00, RR=20), standard setting. Input value is normally from –32768 to +32767

If range=1020 (TT=10, RR=20), the input value will be a clear “Degree Celsius” value. For ex, 2312 means temperature is “23.12” degree celsius.

If seting range as 20 (or 21 to 2F), then it is “standard setting”. The temperature input value is 2’s complement value from –32768 to +32767 depends on the “range” value. For example, setting range as 20, value of –32767 means temperature is about –100 Degree, +32766 is about +100 Degree. Value of 16383 means +50 Degree (**Note**: Normally value of –32768 or +32767 means wire “broken-line”)

If user want to get a clear temperature input value, for example, value of 2312 means “23.12” Degree Celsius. Then please set “range” to a special value defined as below.

Important:
Special “range” is supported since driver version of I-8xx7: 3.11, W-8xx7: 3.24. The iP-8xx7, WP-8xx7, VP-25W7/23W7, XP-8xx7-CE6, XP-8xx7-Atom-CE6 are also support right now.

Format: TTRR (Hex. Value)

TT=10 (Convert to "Degree Celsius")
TT=20 (Convert to "Degree Fahrenheit")
TT=00 (Default value, -32768 to +32767, this is “standard setting”)
RR: original "range" setting

For example, setting I-87013’s "range" as

A. 1020 : (TT=10, RR=20) the input value will be "Degree Celsius", unit is 0.01 degree, range= "20 : Platinum 100, a=0.00385, degree Celsius". That results input value of "2356" = 23.56 Degree Celsius, "-489" = -4.89 Degree Celsius, "999990" = sensor broken line.

B. 202A : (TT=20, RR=2A) the input value will be "Degree Fahrenheit", unit is 0.01 degree, range= "2A: Platinum 1000, a=0.00385, degree Celsius". That results input value of "4512" = 45.12 Degree Fahrenheit, "500" = 5.00 Degree Fahrenheit, "999990" = sensor broken line.

C. 21 : (TT=00, RR=21) the input value will be Default value (standard “range” setting), -32768 to +32767, range = "21 : Platinum 100, a=0.00385, degree Celsius"

3.2.3: Using the I-87017ZW

I-87017ZW is a voltage and current measuring board in industrial application, it can be set as a 10 input channels (Differential Mode) or 20 input channels (Single-ended Mode) by the jumper on the board. It must be set as 10 input channels when measure the current on any channel (For voltage measuring, can set as 10 or 20 input channels). It doesn't need to plug an external resistor of 125 ohms to measure current input, just switch the "Jumper" on the board.

Please refer to http://www.icpdas.com/products/PAC/i-8000/8000_IO_modules.htm > I-87017ZW. When using ISaGRAF to measure it, the sample rate is about 10/10 = 1 Hz (if it set as 10 input channels). It means to sample all 10 channels once per second (if it set as 20 input channels, the sample rate is about 10/20 = 0.5 Hz). (If your application needs faster sample rate, please use the I-8017HW)

The analog input value of I-87017ZW is an integer between –32768 to +32767 as below table.

Voltage:

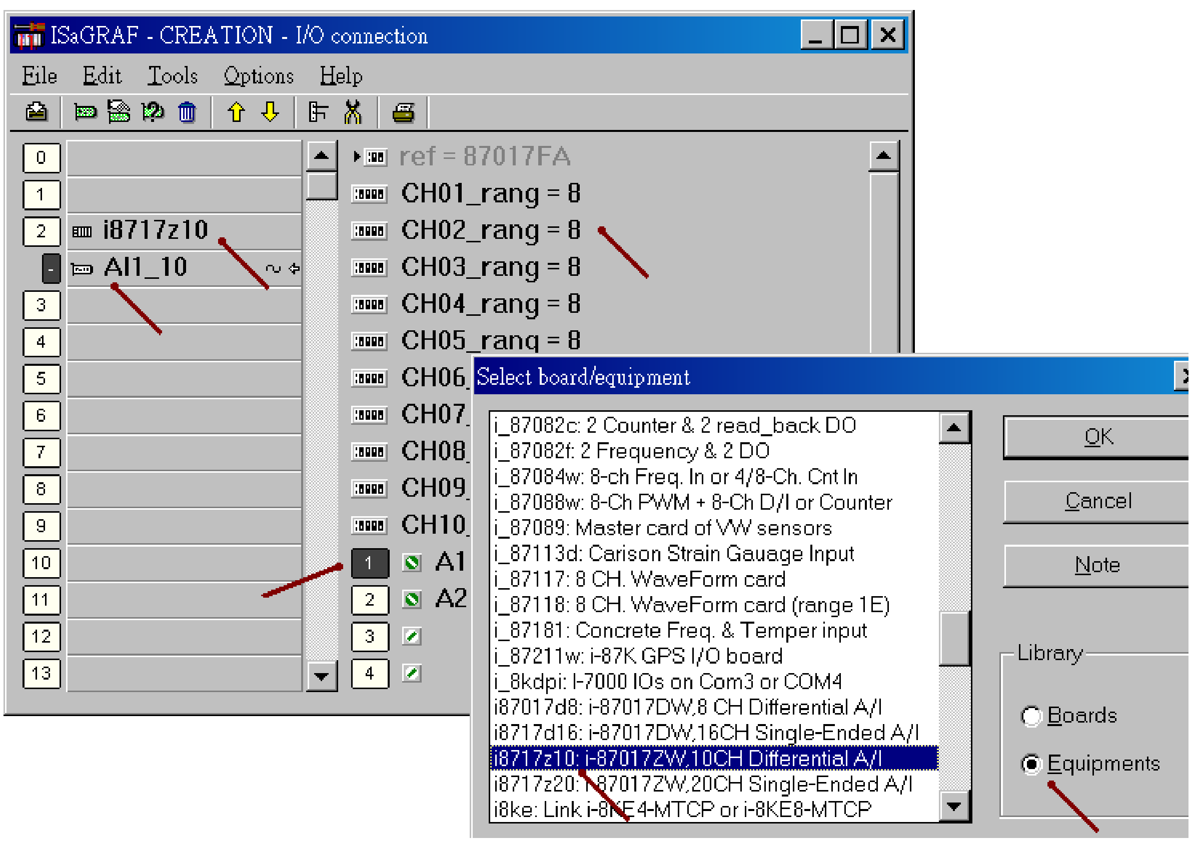
Range Type Code (Hex)	Data Format	Max value	Min value
08 (Default)	Input Range	+10.0 V	-10.0 V
	Decimal Value	+32767	-32768
	2's Complement HEX	7FFF	8000
09	Input Range	+5.0 V	-5.0 V
	Decimal Value	+32767	-32768
	2's Complement HEX	7FFF	8000
0A	Input Range	+1.0 V	-1.0 V
	Decimal Value	+32767	-32768
	2's Complement HEX	7FFF	8000
0B	Input Range	+500.0 mV	-500.0 mV
	Decimal Value	+32767	-32768
	2's Complement HEX	7FFF	8000
0C	Input Range	+150.0 mV	-150.0 mV
	Decimal Value	+32767	-32768

Current:

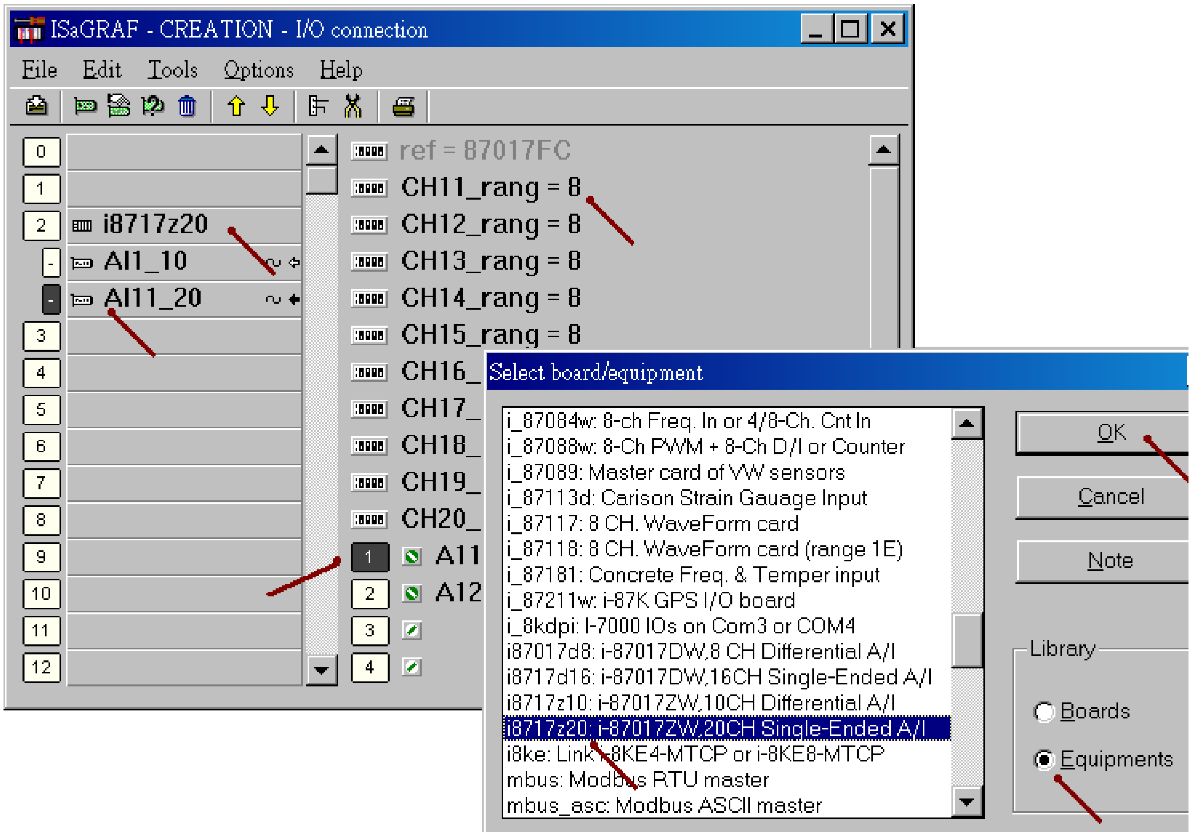
Range Code (Hex)	Item	Max.	Minimum
7	Physical	+20.0 mA	+4.0 mA
	Analog Input value (Decimal)	+32767	0
	Analog Input value (Hex.)	7FFF	0
D	Physical	+20.0 mA	-20.0 mA
	Analog Input value (Decimal)	+32767	-32768
	Analog Input value (Hex.)	7FFF	8000
1A	Physical	+20.0 mA	0 mA
	Analog Input value (Decimal)	+32767	0
	Analog Input value (Hex.)	7FFF	0

If using I-87017ZW and set it as 10 input channels, please connect “i8717z10” in related slot No. in the IO connection window, then set the proper “Range” setting for each channel (double-click “Ch0x_rang” to set it), and then connecting related integer input variables in channel 1 to 10. (Double-click the channel No. to set it).

Note: If the jumper on the I-87107ZW is set to 10-Ch. means that it can measure voltages or currents. It is also noteworthy that you must set the proper “Voltage input” or “Current Input” way for each channel by adjusting the jumper (JP2 – JP11) on the board.
Please refer to http://www.icpdas.com.tw/product/solutions/datasheet/i-8k&i-87k/I-87017ZW.pdf



If using I-87017ZW and set it as 20 input channels, please connect “i8717z20” in related slot No. in the IO connection window. (Notice: it only for voltage measuring when the I-87107ZW’s jumper is set to 20 channels, so the “range” can not be set as “Current” type – 07 , 0D and 1A. Moreover, the jumper (JP2 – JP11) on the board must be set as “Voltage input”.)



Note:
If the current input sensor is 4 to 20 mA, user may better set I-87017ZW ‘s range type to “[D] : +/- 20 mA” , or “[1A] : 0 ~ 20 mA”. (set as " [7] : 4 to 20 mA" is not good)

The reason is :
If setting I-87017ZW‘s range type as “[7] : 4 to 20 mA”, analog Input value of 0 or close to 0 could mean the Sensor input is 4 mA , and also possible the Sensor is broken-line. So it is not easy to distinguish these two situations by software.

Howevr, if setting I-87017ZW‘s range type as “[D] : +/- 20 mA” or “[1A] : 0 ~ 20 mA”, analog input value of 0 or close to 0 only means the Sensor is broken-line. If the Sensor input is 4 to 20mA, the analog value should be 6553 to 32767, not close to 0.

If you want to distinguish whether the current sensor (4 to 20 mA) is normal? It’s better to set the range type as [D] : +/- 20 mA or [1A] : 0 ~ 20 mA. So, when the input values - A1, A2, are less than 5000 or 4000 in your program, you can determine the sensor is broken-line or abnormal. (If the range type is set as [7] : 4 ~ 20 mA, you can’t judge the sensor is 4 mA or abnormal)

3.2.4: Using the I-8017HW

I-8017HW is an I-8K parallel analog input board and provides 8-Ch. (Differential input) or 16-Ch. (Single-end input). If using in the iP-8xx7 controller, the I-8017HW's max. Sample rate is about 125Hz (8-Ch.) or 62 Hz (16-Ch.). If using in WP-8xx7, VP-25W7, XP-8xx7-Atom-CE6, XP-8xx7-CE6 controllers, the I-8017HW 's max. Sample rate is about 200Hz (8-Ch.) or 100 Hz (16-Ch.). However, the sample rate depends on the PLC scan time, the bigger PLC scan time, the smaller sample rate will be. For ex, if the PLC scan time is 50 ms that will make the max. I-8017HW sample rate only about 1000 / 50 = 20 Hz.

Please refer to http://www.icpdas.com/products/PAC/i-8000/8000_IO_modules.htm > I-8017HW

Range setting	Minimum Physical (analog val.)	Physical (analog val.)	Maximum Physical (analog val.)
5	-2.5 V (-32768)	0 V (0)	+ 2.5 V (+32767)
6 (need an external resistor of 125 ohms)	-20 mA (-32768)	0 mA (0)	+ 20 mA (+32767)
7	-1.25 V (-32768)	0 V (0)	+ 1.25 V (+32767)
8	-10 V (-32768)	0 V (0)	+ 10 V (+32767)
9	-5 V (-32768)	0 V (0)	+ 5 V (+32767)

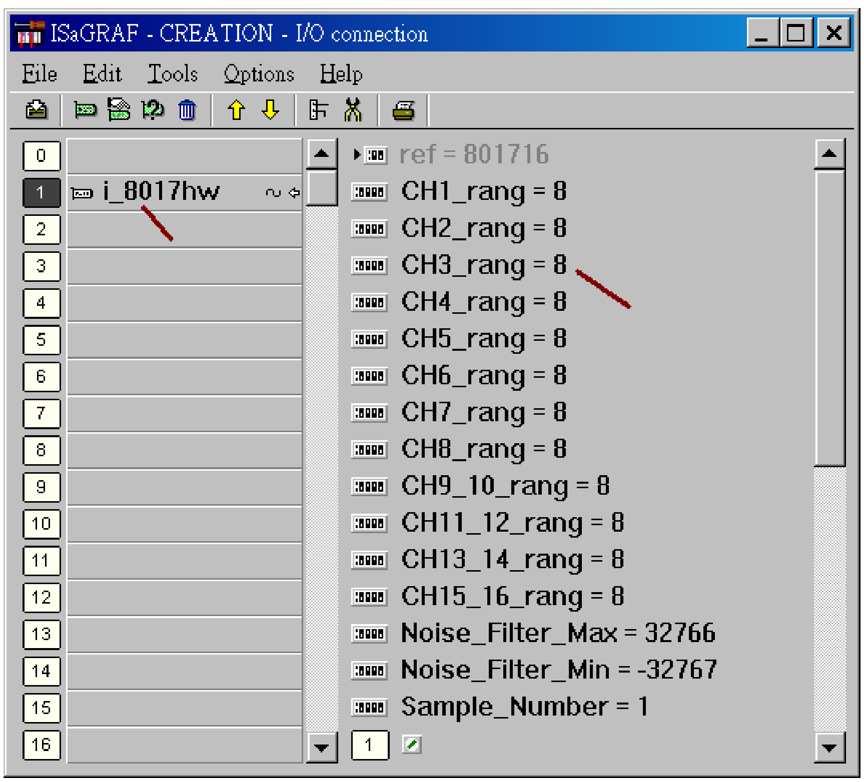
Parameters:

Noise_Filter_Max: The maximum allowed analog input value. -32468 to +32767. If the analog input value is larger than this value, it will be modified to become this value.

Noise_Filter_Min: The minimum allowed analog input value. -32768 to +32467. If the analog input value is smaller than this value, it will be modified to become this value.

Sample_Number:
The sampling number to be averaged as an Analog input value. Default is 1. It can be modified to become 1 to 500. Setting as 1 using the max. sampling speed. However setting as 500 means every 500 samples to be averaged as one analog input value.

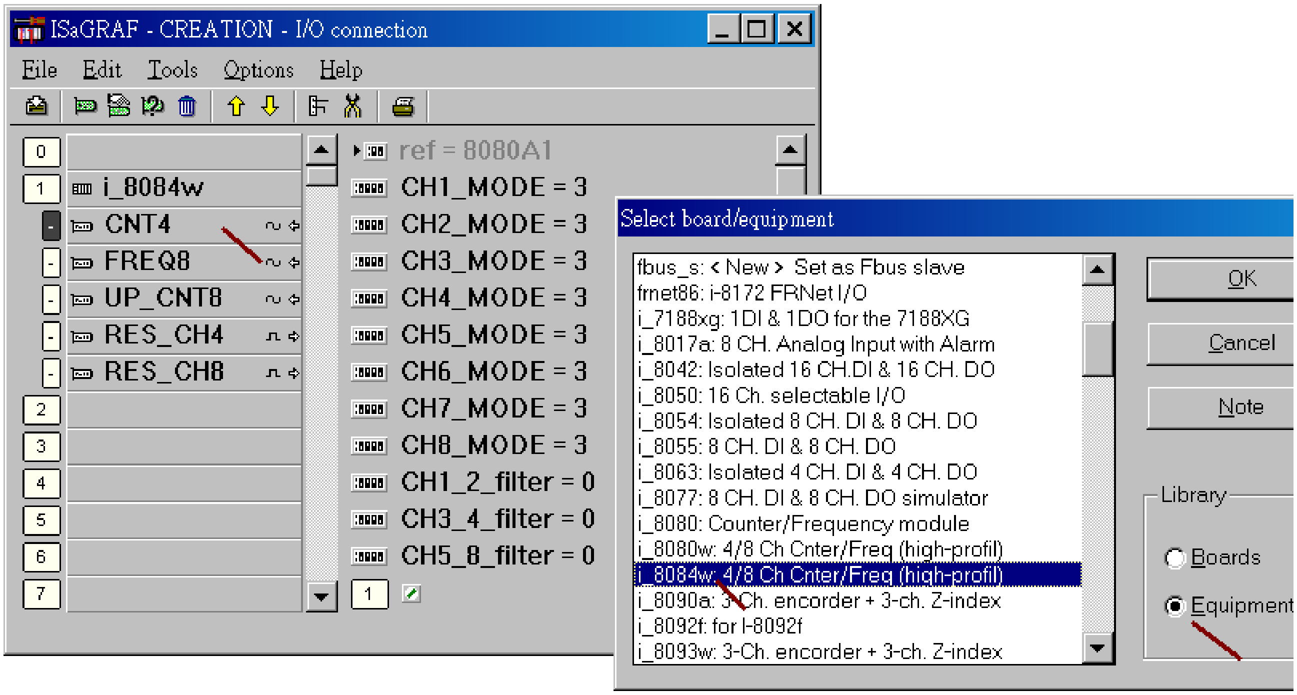
The higher this value the lower the sample rate is. And the I-8017HW's input waveform will become smoother.



3.2.5: Using the I-8084W

The iP-8xx7, WP-8xx7, XP-8xx7-CE6 , XP-8xx7-Atom-CE6 and VP-25W7 / 23W7 support the I-8084W board. The I-8084W can be used as “8-Ch. Up Counter” or “4-Ch. Dir/Pulse Counter” or “4-Ch. Up/Down Counter” or “4-Ch. A/B phase Counter (Quard. mode)”, and it can also be used as “8-Ch. Frequency input”.

Please refer to http://www.icpdas.com/products/PAC/i-8000/8000_IO_modules.htm > I-8084W



The default "CHx_x_filter" setting is 0 (“0” means do not enable this function). If the value is not “0”, it is for filtering, the input signal with smaller time-width (that is, larger input frequency) will be filtered out.

CH1_2_filter: for Ch.1 and Ch.2 of “8-Ch. Up Counter” and “8-Ch. Frequency” or
Ch.1 of “4-Ch. Dir/Pulse Counter” and “ 4-Ch. Up/Down Counter“

CH3_4_filter: for Ch.3 and Ch.4 of “8-Ch. Up Counter” or “8-Ch. Frequency” or
Ch.2 of “4-Ch. Dir/Pulse Counter” and “ 4-Ch. Up/Down Counter“

CH5_8_filter: for Ch.5, 6, 7, 8 of “8-Ch. Up Counter” and “8-Ch. Frequency” or
Ch.3 and Ch.4 of “4-Ch. Dir/Pulse Counter” and “ 4-Ch. Up/Down Counter“

Please set a proper filter value according to the physical input signal.

Max. allowed input signal (Hz)	CHx_x_filter value
0 ~ 1K	200
0 ~ 2K	100
2K ~ 5K	40
5K ~ 10K	20
10K ~ 20K	10
20K ~ 100K	2
100K ~ 450K	0 (disabled)

"CHx_MODE" setting is to set the signal input type of each channel as below.

"CH1_MODE" to "CH8_MODE" is for Ch.1 to Ch.8 of "8-Ch Up Counter" and "8-Ch Frequency" .

If setting as "4-Ch. DIR / Pulse Counter" or "4-Ch. Up / Down Counter" mode,
CH1_MODE and CH2_MODE must set as the same value. It is for Ch1.
CH3_MODE and CH4_MODE must set as the same value. It is for Ch2.
CH5_MODE and CH6_MODE must set as the same value. It is for Ch3.
CH7_MODE and CH8_MODE must set as the same value. It is for Ch4.

For example,
1. if setting CH1_MODE as 0 : "Dir / Pulse" (4-Ch), then CH2_MODE should be also set as 0.
2. if setting CH1_MODE as 3 : "Up Count" (8-Ch), then CH2_MODE can be set as 83 , 2, 6, A, 82, 86 or 8A

Below value is for Counter input type.

0 : Dir / Pulse (4-Ch.)
1 : Up / Down (4-Ch.)
3 : Up Count (8-Ch)
80 : Dir / Pulse (4-Ch. Inverse input signal)
81 : Up / Down (4-Ch. Inverse input signal)
83 : Up Count (8-Ch. Inverse input signal)

Below valus is for Frequency input type.

2 : Frequency (apply the "AutoTT" setting)
6 : Frequency (apply the "LowTT" setting)
A : Frequency (apply the "HighTT" setting)
82 : Frequency (apply the "AutoTT" setting, Inverse input signal)
86 : Frequency (apply the "LowTT" setting, Inverse input signal)
8A : Frequency (apply the "HighTT" setting, Inverse input signal)

Note:

1. "DIR / Pulse" mode and "Up / Down Counter" mode are similar as Encorder Input. The Counter value should be controlled in between -2,147,483,648 to 2,147,483,647. Or it will be overflow.

2. The input value of "Up Counter" mode is a 32-bit integer. It starts at 0, then increasing by the signal input, 1, 2, … to max. value of +2,147,483,647, then if one more signal input, the value will suddenly drop to -2,147,483,648. Then increasing … to -2 , -1 , 0, 1, 2, … to +2,147,483,647.

   The ISaGRAF integer value is a signed 32-bit integer, it can not get a positive value larger than +2,147,483,647.

   If user apply SCADA software which can handle unsigned 32-bit integer, then the value displayed in the SCADA software can be 0, 1, …, +2147483647 , +2147483648 , +2147483649 , … , +4294967295 , then back to 0, 1, …

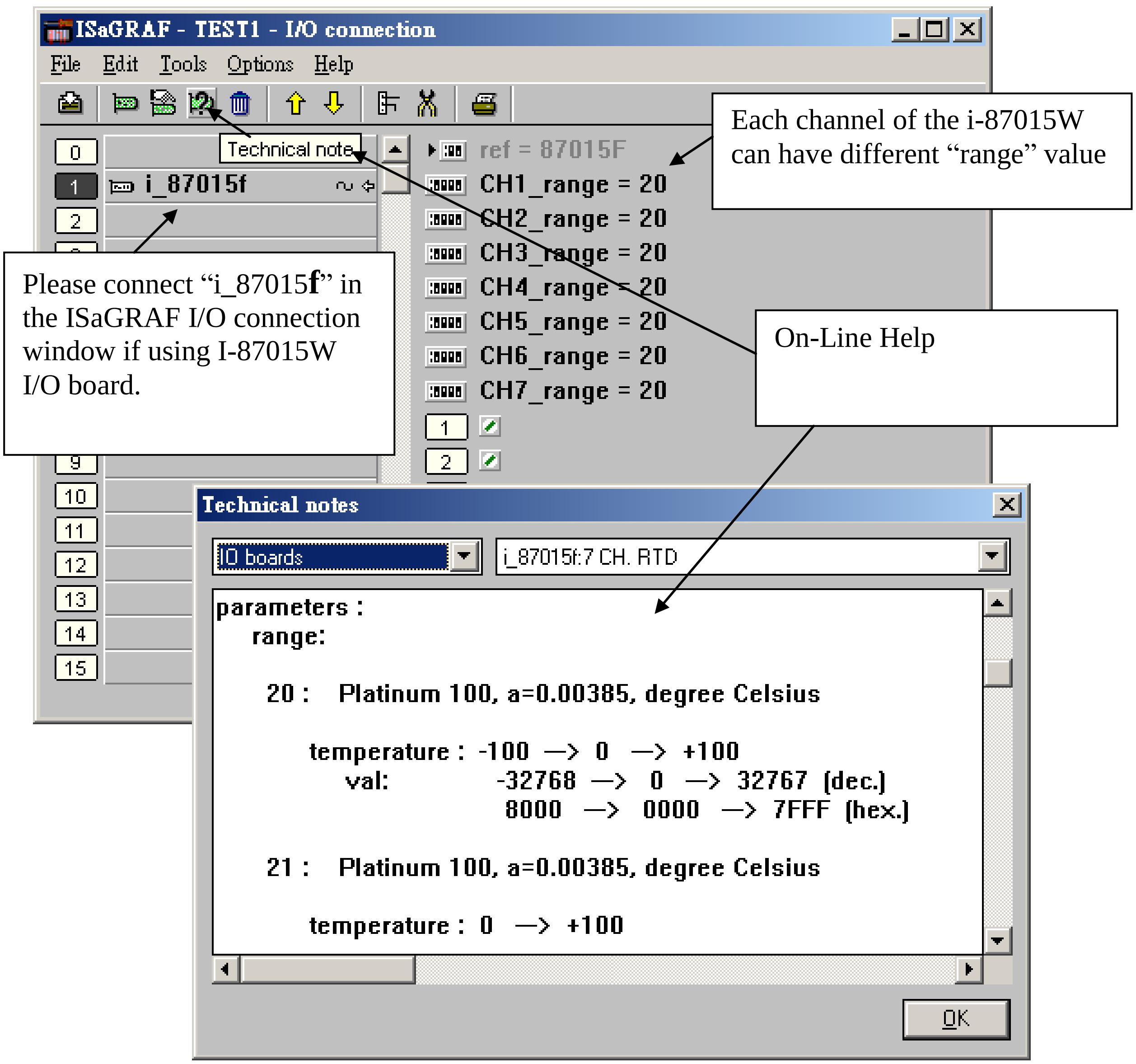
3.2.6: Using the I-87015W and I-87015PW

If the position of RTD sensor is far away from the "I-87015W" board, purchasing "I-87015PW" is a better choice.

The I-87K I/O board (like I-87015W, I-87015PW) can plug in the slot 0 through slot 7 of the ISaGRAF controller. It can also be used as RS-485 remote I/O module. (That is plugged in the RS-485 I-87K4/5/8/9 or RU-87P4/8 expansion unit. Please refer to Chapter 6).

Please refer to http://www.icpdas.com/products/PAC/i-8000/8000_IO_modules.htm > I-87015W
I-87015 's Sample Rate is about 12/7 = 1.7 Hz that means 1.7 samples per second on all 7 channels.

The below figure is using the I-87015 in slot 1.
The I-87015W / 87015PW has 7-channel of RTD temperature input. Please connect "i_87015**F**" in the ISaGRAF I/O connection window if using I-87015W I/O board. Each channel can have different "Range" setting. Please refer to the On-Line Help as below for more information. (If users want to convert to degree Celsius or Degree Fahrenheit, please refer to Chapter 3.2.2 to set proper "Range" value)



3.2.7: Using the I-87019ZW

I-87019ZW is a 10-channel of Thermo-Couple temperature input board for industrial use or can be set as normal analog input board, like "±10V" or "±20mA". Each channel can have different "range" type setting.

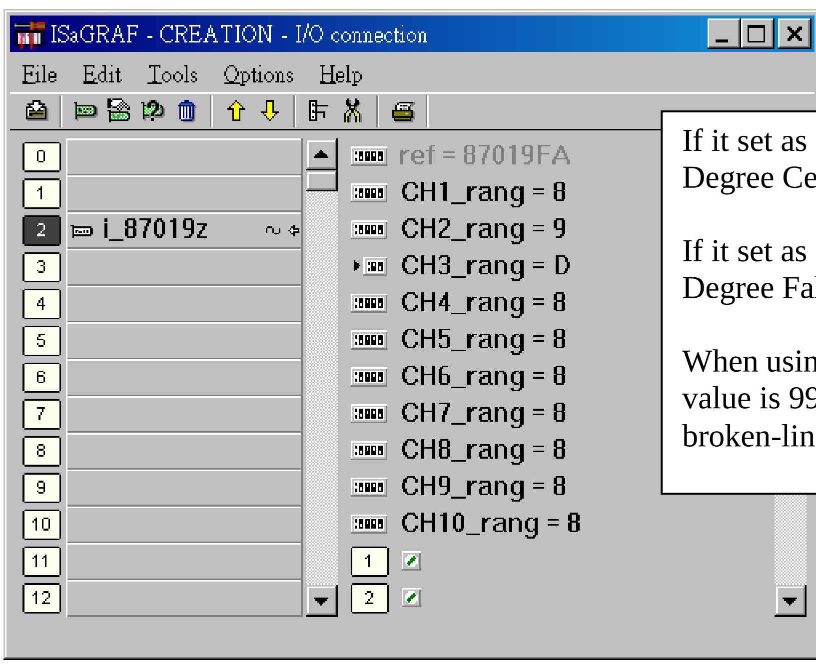
Please refer to http://www.icpdas.com/products/PAC/i-8000/8000_IO_modules.htm > I-87019ZW

The analog input value of I-87019ZW is between –32768 to +32767 (refer to Appendix D).
When applying as temperature input, please refer to Chapter 3.2.2 to get direct temperature value.

If applying I-87019ZW to measure current input ("range" setting is 06, 07 or 0D), it doesn't need to plug external resistor of 125 ohms. (There is "Jumper" to be switched on the board).

I-87019ZW's Sample Rate is about 10/10 = 1 Hz. that means 1 sample per second on all 10 channels. (If your application is to measure faster signal of "±10V" or "±20mA", please use I-8017HW)

To program I-87019ZW, please connect "i_87019z" in the related slot in the ISaGRAF I/O connection window. Then set proper "range" setting. Then connecting related Integer Input variable in Ch.1 to Ch. 10 as below.



If it set as 100F: T/C K-Type, convert to Degree Celsius, unit is 0.01degree.

If it set as 200F: T/C K-Type, convert to Degree Fahrenheit, unit is 0.01degree.

When using special setting and the return value is 999990, it means Sensor broken-line (Refer to Section 3.2.2)

3.2.8: Using the I-8024W

The I-8024W is a 4-channel analog output board which can output signal of "±10V" and "0 to 20mA" . Each channel can have different "range" setting.
Please refer to http://www.icpdas.com/products/PAC/i-8000/8000_IO_modules.htm > I-8024W

The ouput speed of the I-8024W depends on the PLC Scan Time. For example, if PLC scan time is 10 ms, then the ouput value will be ouput in about 10 ms. However if PLC scan time is 100ms, then the ouput value will be ouput in about 100 ms.

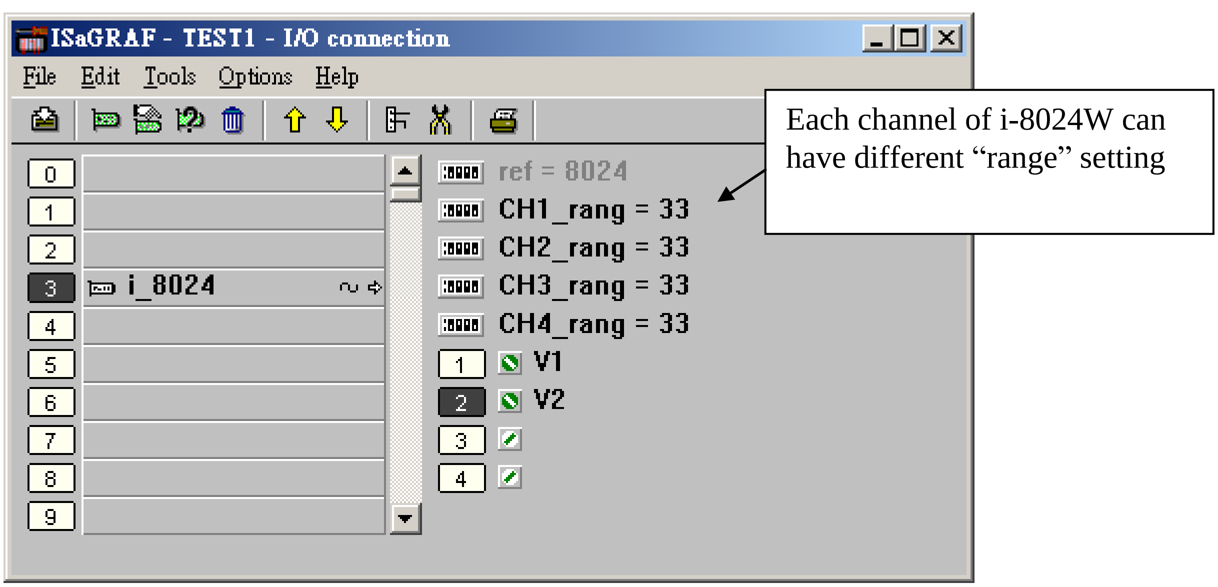
The speed of the I-8024W is faster than I-87024W. Besides, unlike the I-87024W, the I-8024W can have different "range" setting in each channle. (All channels of I-87024W use the same "range" setting)

However I-87024W can use in slot 0 to 7, and also in the RS-485 remote I-87K4/5/8/9 or RU-87P4/8 expansion unit. (I-8024W can only plug in slot 0 to 7 of the controller)

The analg output value of the I-8024W is :
Current : (0 to 32767) means (0 to 20 mA) or
Voltage : (-32768 to 32767) means (-10 V to 10 V)

Range Code	Item	Max.	Minimun
30	Physical signal	+20.0 mA	+0.0 mA
	Analog output value	+32767	+0
33	Physical signal	+10.0 V	-10.0 V
	Analog output value	+32767	-32768

To program I-8024W, please connect "i_8024" in the related slot No. on the I/O connection window. Then set proper "range" setting. Then connect related Integer Output variable to Ch.1 to Ch. 4.



3.2.9: Using the I-87018ZW

I-87018ZW is a 10-channel of Thermo-Couple temperature input board for industrial use or can be set as normal analog input board, like ±20mA, 0 ~ 20 mA, 4 ~ 20 mA, ±2.5V , ±1V , ±500mV, ±100mV, ±50mV or ±15mV. The thermocouple measurement for each channel of the I-87018ZW is more accurate than I-87018W and I-87018RW and each channel can configure to be different Input type and range. For example, using Ch.1 to 4 to measure 4 to 20 mA , using Ch.5 to 8 as Thermo-Couple K-Type, using Ch.9 to measure +/- 2.5 V, and using Ch.10 as Thermo-Couple R-Type.

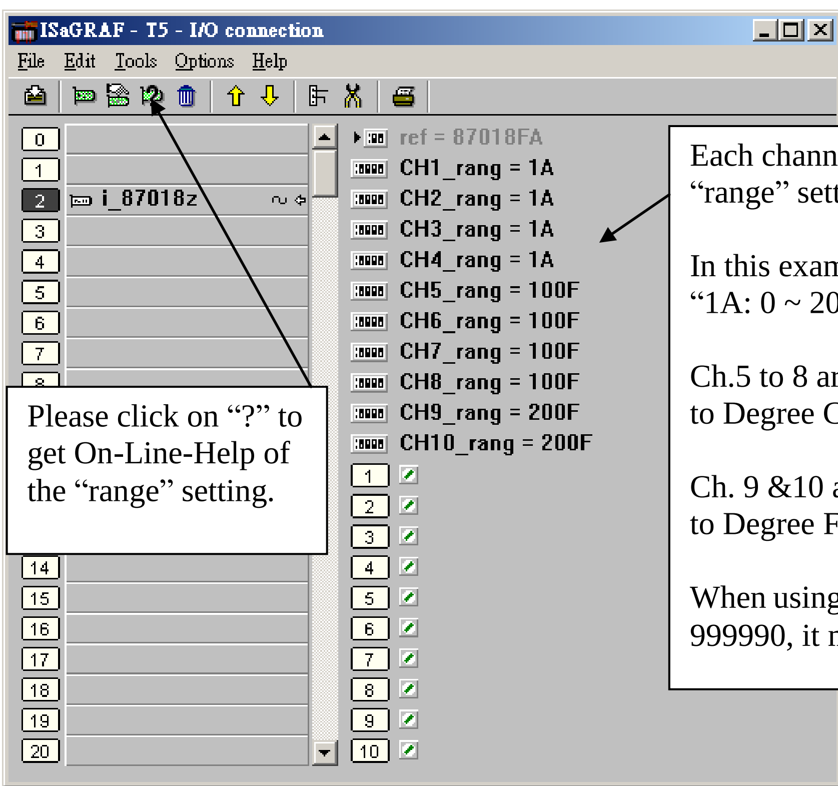
Please refer to http://www.icpdas.com/products/Remote_IO/i-87k/i-87018z.htm > I-87018ZW

The analog input value of I-87018ZW is between –32768 to +32767 . (refer to Appendix D)
When applying as temperature input , please refer to Chapter 3.2.2 to get direct temperature value.

Please connect an external resistor of 125 ohms if using I-87018ZW to measure current input (“range” setting is 06: ±20mA or 07: 4 ~ 20 mA or 1A: 0 ~ 20 mA) .

I-87018ZW ‘s Sample Rate is about 10/10 = 1 Hz. Means 1 sample per second on all 10 channels (If your application is to measure faster signal of “±10V” or “±20mA” , please use I-8017HW)

To program I-87018ZW, please connect “i_87018z” in the related slot No. on the I/O connection window. Then set proper “range” setting. Then connect related Integer Input variable to Ch.1 to Ch. 10 .



Each channel of i-87018ZW can have different “range” setting.

In this example, Ch. 1 to 4 are using “1A: 0 ~ 20mA “.

Ch.5 to 8 are using “100F: T/C K-Type, convert to Degree Celsius, unit is 0.01degree” .

Ch. 9 &10 are using “200F T/C K-Type, cobvert to Degree Fahrenheit, unit is 0.01degree.

When using special setting and the return value is 999990, it means Sensor broken-line.

3.3: Linking Some Special Virtual board

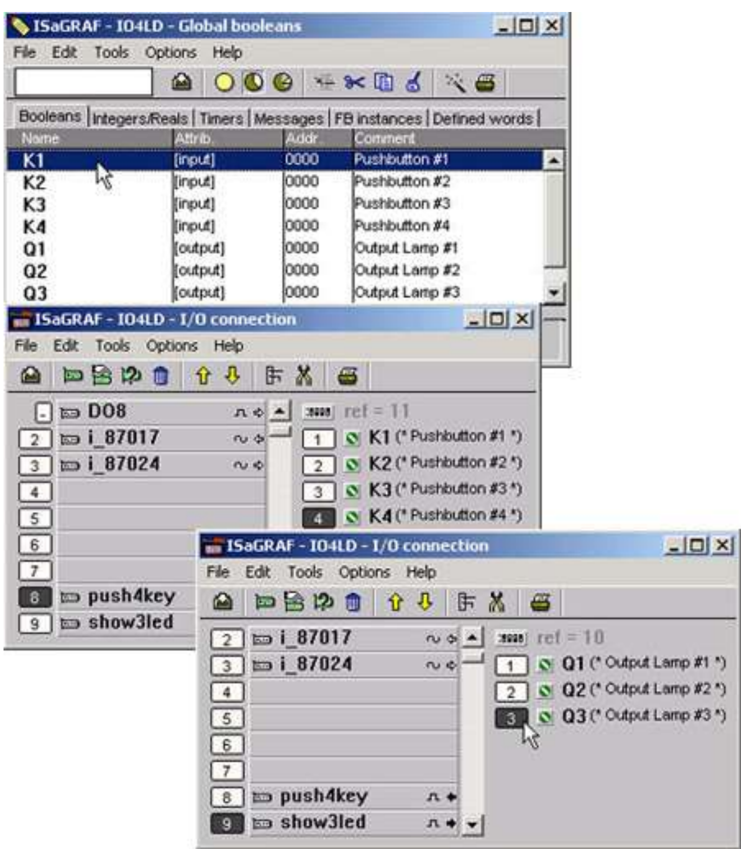
3.3.1: Using “Push4Key” and “Show3Led”

The I-8xx7 and iP-8xx7 controller has some helpful tools for testing and debug, that are "Push4Key" and "Show3Led" functions on the front panel of the controller. The I-8xx7, iP-8xx7, XP-8xx7-Atom-CE6, WP-8xx7 and WP-5xx7 supports the "Show3Led" to control Leds on the front panel of these PAC. (Some can control 3 Leds. However some can only control 2 Leds or one Led). Only the I-8xx7 and iP-8xx7 support the "Push4Key" .

The "Push4Key" are the four pushbuttons on the I-8xx7 control front panel and they are handled as digital inputs. The "Show3Led" are three of the four LED’s on the I-8xx7 control front panel (the first three from left to right, the fourth LED is strictly to show if the power is turned on the I-8xx7 controller system) and they are handled as digital outputs.

Both of these can be linked to an ISaGRAF program through the "I/O Connection" window and can be used to interface with Man Machine Interface (MMI) programs or for program debugging. It is recommended that you assign these functions to slot 8 or higher (remember, slots 0 through 7 are reserved for real I/O boards.

IMPORTANT NOTE:
As with any real digital input or real digital output, you MUST declare a variable name for each of the "Push4Button" inputs and "Show3Led" outputs in the "ISaGRAF Global Variables" window BEFORE they can be assigned to an ISaGRAF program.



3.3.2: Using “io_state” to test the operation state of real I/O boards

The “io_state” function can be used to test the operation stste of real I/O board in slot 0 through slot 7. The failure states are as below.

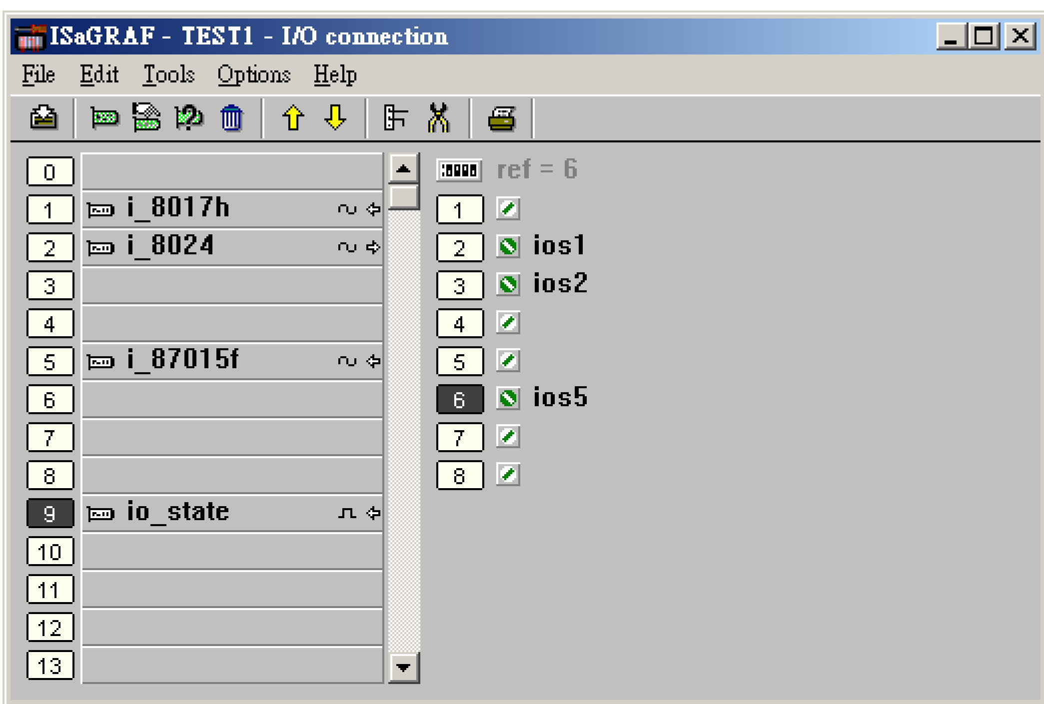
1. Wrong I/O board plugged in the slot. (It can not detect I-8112iW/8114iW/8142iW/8144iW and I-8212W / I-8213W, I-8072/I-8073 I/O boards)
2. I/O board is absent.
3. I/O board is damaged which cause the ID byte can not be read by the controller.

Note:
Please DO NOT plug the I-8xxx and I-8xxxW parallel I/O boards on slot 0 ~ 7 when controller power is ON. This action may make the IO board and backplane damaged. Only the I-87xxxW (high profile) I/O boards can support “Hot-Swap” function on the iP-8xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6, WP-8xx7, VP-25W7, VP-23W7 and VP-2117 controllers.

“io_state” can only detect the real I/O boards in slot 0 through 7. If the slot No. in the ISaGRAF I/O connection window is empty, it is not detectable.

To use “io_state”, please connect it at slot 8 or larger slot No in the I/O connection window.

The below example will detect if slot 1, slot 2 and slot 5 working normal or failure. The channel 1 ~ 8 in the “ io_state” will show the operation state of real I/O boards on slot 0 ~ 7. It means working normal if the related channel is TRUE, False means failure or can not find the related I/O board (The XP-8xx7-CE6, XP-8xx7-Atom-CE6 and WinCon doesn’t have channel 1 – “Slot 0”)



3.3.3: Using "echo_tim" to delay some milli-seconds before the Modbus RTU Slave port to replying

I-8xx7, iP-8xx7 and W-8xx7 / 8xx6 controllers support "echo_tim" to delay some milli-seconds before the Modbus RTU slave port to replying. Each I-8xx7 and W-8xx7/8xx6 can control only one Modbus RTU slave port to delay the reply, for other Modbus RTU slave port, it will quickly reply when PC/ HMI or other devices received the Modbus Request.

The "echo_tim" can control the below Modbus RTU Slave port.

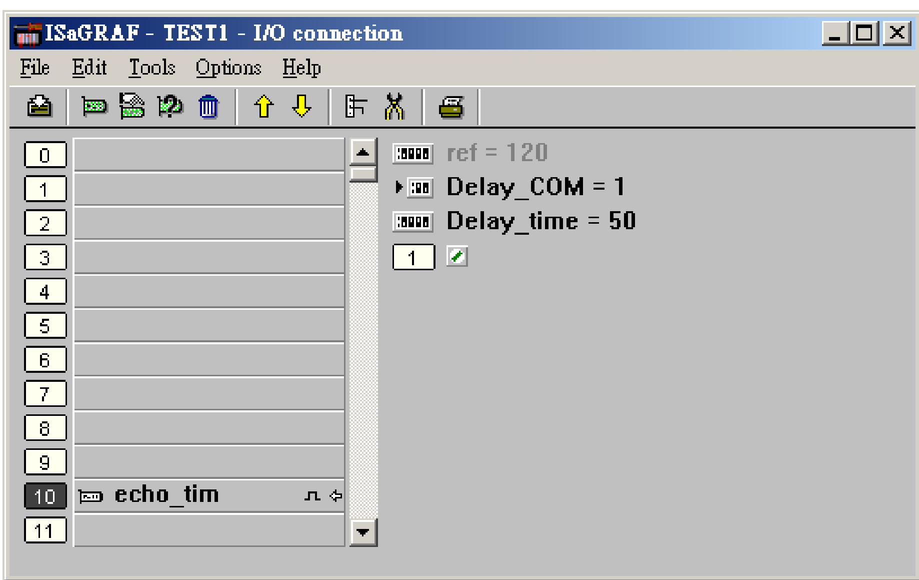
I-8xx7, iP-8xx7 :	One of COM1 and COM2 and COM3
W-8xx7 / 8xx6 :	One of COM2 and COM3.

To use "echo_tim", please connect it at slot 8 or larger slot No.

The first "Delay_COM" setting is the COM port No. to be delay. I-8xx7, iP-8xx7 can be 1, 2 or 3. W-8xx7 can be 2 or 3. The second "Delay_time" is the milli-seconds to delay. It can be 1 to 10000, unit is ms (0.001 second).

It means Ok if Channel 1 returns FALSE. If it returns TRUE, something wrong, for example, setting wrong COM port No.

The below example set I-8xx7's COM1 Modbus RTU Slave port to delay 50 ms (= 0.05 second) before responding (WinCon just can delay COM2 or COM3).

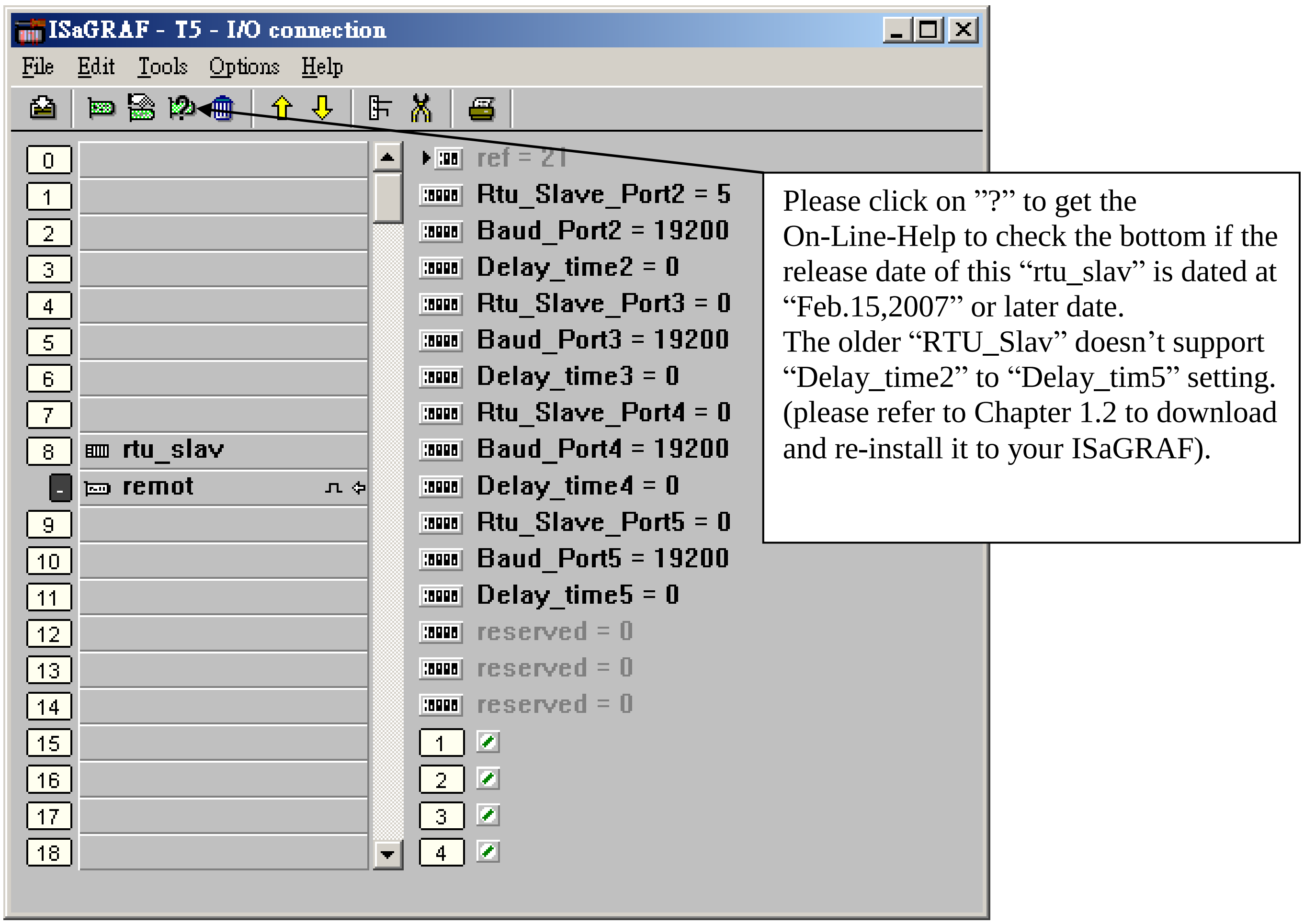


Why Modbus RTU Slave port need to delay ?
For example, there is some application applying some communication equipment (like radio modem) which needs some time to switch the sending / receiving state. When PC / HMI send Modbus RTU request to I-8xx7, iP-8xx7 and W-8xx7 via this "radio modem" , the I-8xx7, iP-8xx7 and W-8xx7 should not reply immediately. They should delay some milli-seconds to wait this "radio modem" to complete its switching to send / receive state, then reply.

3.3.4: Using “RTU_Slav” to expand more Modbus RTU Slave ports in WP-8xx7, WP-5xx7, VP-25W7, XP-8xx7-Atom-CE6 and XP-8xx7-CE6

WP-8xx7, WP-5xx7, VP-25W7 can configure one of its COM2:RS-232 and COM3:RS-485 to become a Modbus RTU slave port. In addiction, it can expand max. four Modbus RTU slave ports via plug I-8112iW / 8114iW / 8142iW / 8144iW serial communication board (COM5 to COM8). Then please refer to Appendix E and G of the W-8xx7 “Getting Started Manual” to setup COM5 to other COM No. first .

Please connect “RTU_Slav” at slot 8 or larger slot No as below.



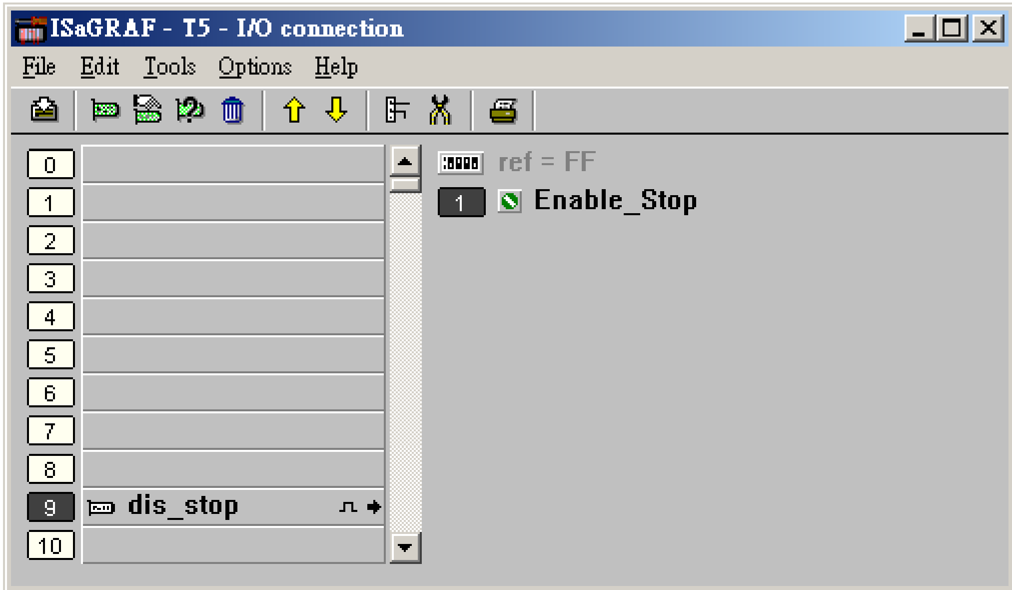
“Rtu_Slave_Port2” to “Rtu_Slave_Port5” setting can be 0, 5, 6 ,7 or 8 . Setting as 0 means not enable the Modbus RTU Slave port. “Baud_Port2” to “Baud_Port5” setting can be 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600 or 115200. “Delay_time2” to “Delay_time5” setting can be 0 or 1 to 10000. unit is 0.001 second. It specifies the delaying milli-seconds before replying to the PC / HMI / SCADA software. Why dealy ? Please refer to the description of Chapter 3.3.3.

Note:

1. The XP-8xx7-Atom-CE6 and XP-8xx7-CE6 support "RTU-slav" and "RTU_slv2" to expand max. 8 another Modbus RTU slave ports.
2. The WP-8xx7, VP-25W7, VP-23W7 and WP-5xx7 support only the "RTU-slav" to expand max. 4 another Modbus RTU slave ports.

3.3.5: Using “dis_stop” to disable / enable the ISaGRAF Download function

For some reason, to prevent someone to use ISaGRAF software to stop or to download a different controller project already running in the ISaGRAF PAC, the “Dis_stop” can be applied . Please connect “dis_stop” at a slot No. larger than 8 and init the channel value to become TRUE. Then stop / download command is not allowed in this controller.



To disable “Dis_stop” to accept stop / download command, please run the original ISaGRAF project to link to this controller and set the channel value to become False.

Please refer Chapter 19 for more information about the Internet security.

3.4: Directly Represented Variables

If you have an ISaGRAF-256 or ISaGRAF-L workbench, you don't need to use the skill described in this section.

A very useful feature of the ISaGRAF Workbench program is the ability to create "directly represented (or internal)" variables. Internal variables are program variables that can be used in an ISaGRAF program, but they are not physically connected to any of the input or output variables. There are four versions of the ISaGRAF Workbench program available with the ISaGRAF PAC: ISaGRAF-32, ISaGRAF-80, ISaGRAF-256, and ISaGRAF-L. The number after "ISaGRAF" represents the number of I/O variables that are allowed with that particular ISaGRAF Workbench program.

The ISaGRAF Workbench program comes with a hardware protection device (dongle) that plugs directly into your development computers parallel port. Every time you compile a program in ISaGRAF the hardware protection device is read to make sure that you are not trying to connect to more program variables than are allowed with your particular copy of the ISaGRAF Workbench program that you purchased with your ISaGRAF PAC.

These "directly represented (henceforth called "internal") variables can be used in lieu of your real world inputs and outputs so you can create additional program variables that do not count against the amount of ISaGRAF program variables. The only "caveat emptor" to these internal variables is that you must follow a strict programming scheme to program and access these internal variables, and they are more complicated to create than the regular input and output variables. **For a professional programmer, recommend to purchase an ISaGRAF-256 workbench.**

Single Type (Board) Internal Variable Programming Scheme:

For single-typed board: "**s**" is the slot No, "**c**" is the channel No.

%IX**s.c**	free channel of a **boolean input** board,	ex. %IX2.3
%QX**s.c**	free channel of a **boolean output** board,	ex. %QX0.2
%ID**s.c**	free channel of an **integer input** board,	ex. %ID3.1
%QD**s.c**	free channel of an **integer output** board,	ex. %QD2.4
%IS**s.c**	free channel of a **message input** board,	ex. %IS3.1
%QS**s.c**	free channel of a **message output** board,	ex. %QS2.4

Complex Type (Equipment) Internal Variable Programming Scheme:

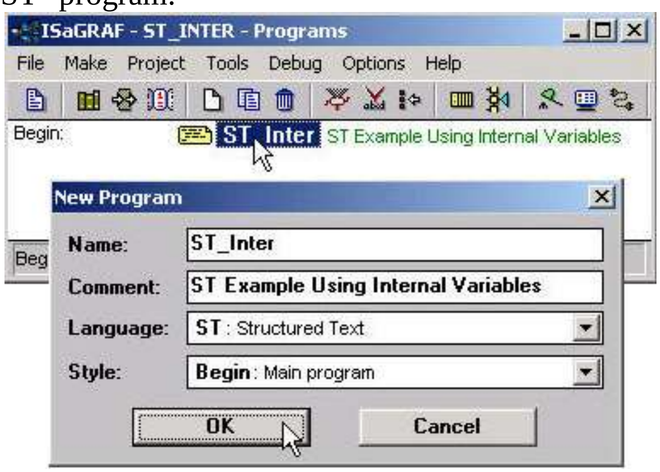
For complex board: "**s**" is the slot No, "**b**" is the index of the single board within the complex equipment. "**c**" is the channel No.

%IX**s.b.c**	free channel of a **boolean input** board,	ex. %IX2.3.2
%QX**s.b.c**	free channel of a **boolean output** board,	ex. %QX0.2.1
%ID**s.b.c**	free channel of an **integer input** board,	ex. %ID3.1.3
%QD**s.b.c**	free channel of an **integer output** board,	ex. %QD2.4.3
%IS**s.b.c**	free channel of a **message input** board,	ex. %IS3.3.1
%QSs.**b**.**c**	free channel of a **message output** board,	ex. %QS2.1.4

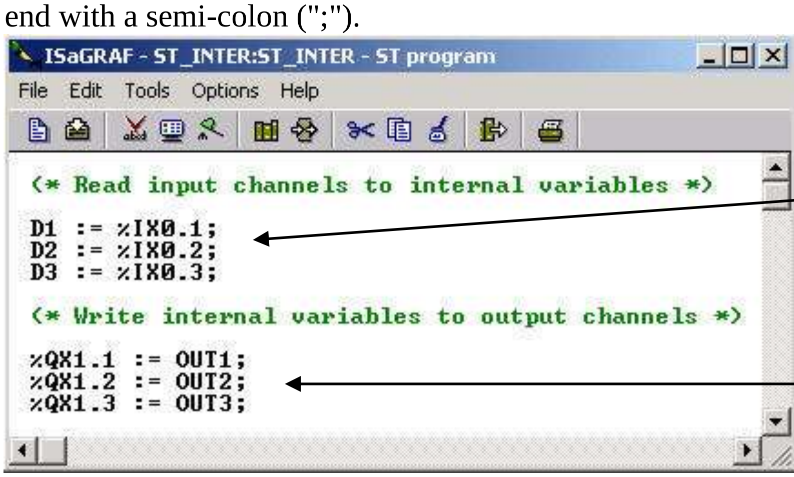
An Internal Variable Program Example

Create a new project for an ISaGRAF ST program, and then create a link to the I/O boards that are specified in the window below. Declare three input variables called "D1", "D2", & "D3" for the I-8051 board located at slot 0, and then create three output variables called "OUT1", "OUT2", & OUT3" for an I-8056 board located at slot 1. This time set each of their respective attributes to "internal" instead of input or output (this means they are not connected to any real physical I/O).

Create a new "ST" program.



Double click on the "ST_Inter" that is highlighted and the "ISaGRAF ST Program" window will open. Type in the program code displayed in the window below EXACTLY as shown. Remember, each line MUST end with a semi-colon (";").

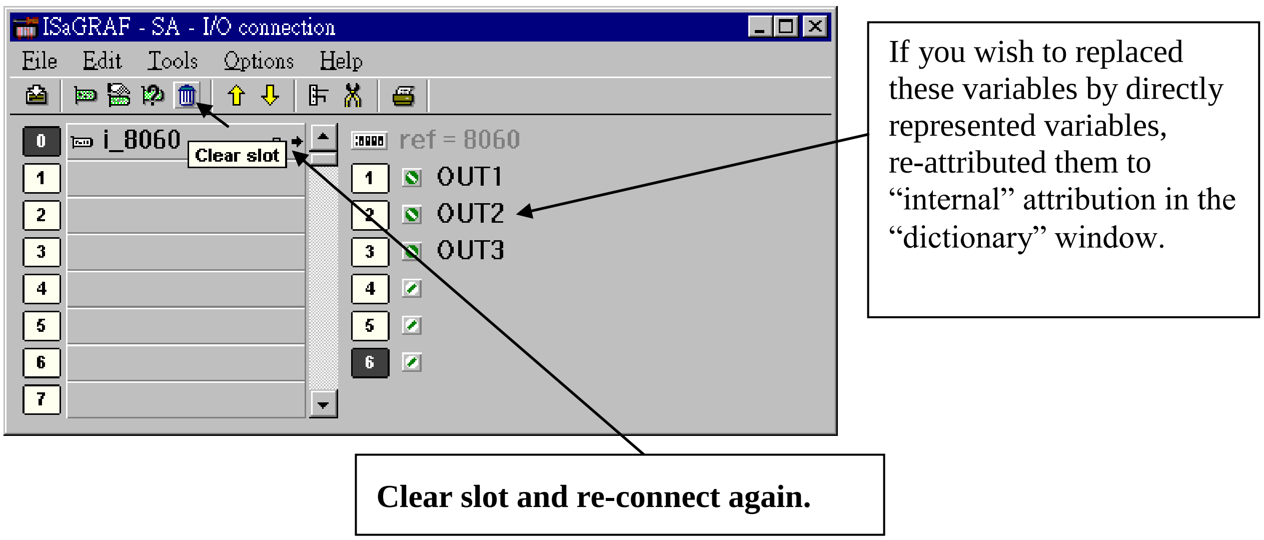



```
D1 := %IX0.1 ;
D2 := %IX0.2 ;
D3 := %IX0.3 ;
```

```
%QX1.1 := OUT1 ;
%QX1.2 := OUT2 ;
%QX1.3 := OUT3 ;
```

Now we can use the internal variables D1 through D3 and OUT1 through OUT3 that have been created in other programs in the same project. The newly created internal variables will generate input and output actions to the associated channels in this ST program.

IMPORTANT NOTE:
If once the input or output attributed variables have been connected to an connected IO board or complex equipment, and if they would like to be replaced by Directly represented variables, these input or output attributed variables have to be re-attributed to “internal” and the board or equipment **must be re-connected to the slot**.

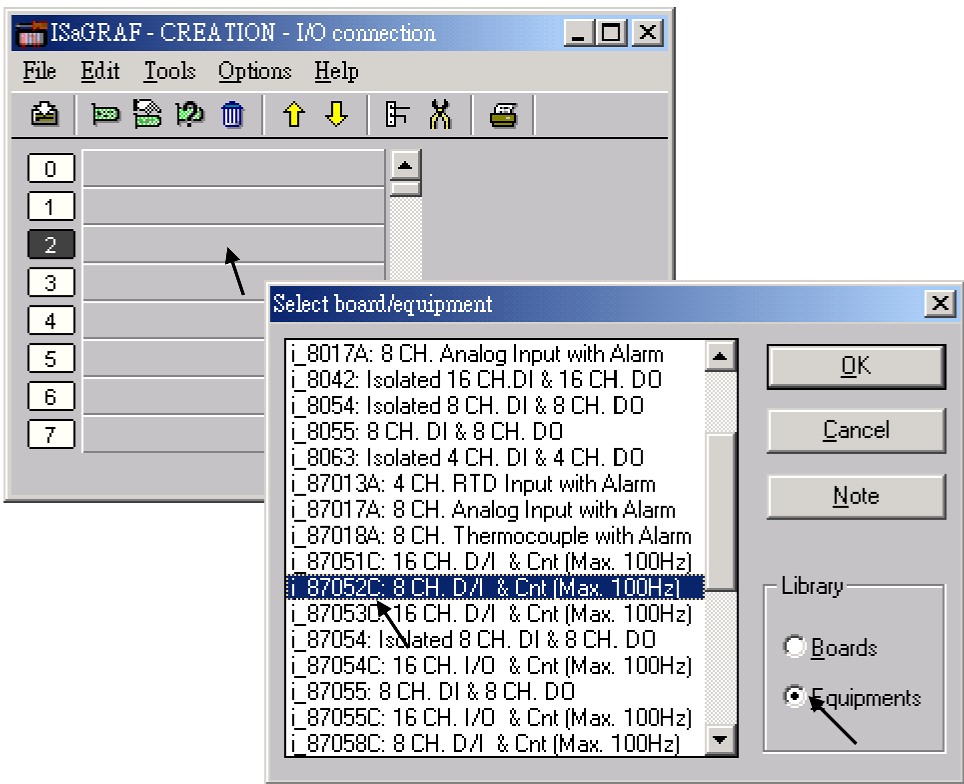


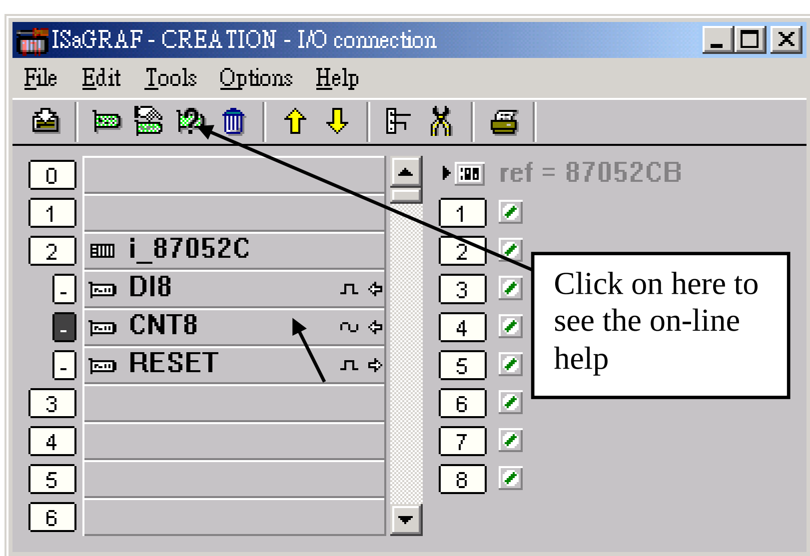
IMPORTANT NOTE
If you enable the compiler option of upload, option “**Comments for not connected I/O channels**” must be choosed if “Directly represented variables” is used in this project (refer to section 9.2).

3.5: D/I Counters Built in The I-87xxx & I-7000 D/I Modules

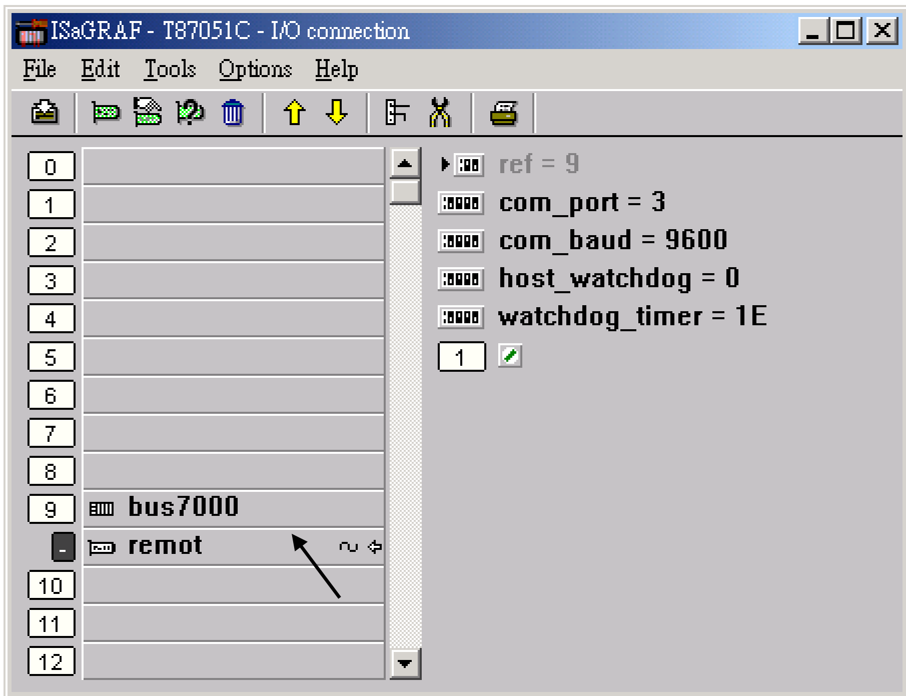
87051W, 87052W, 87053W, 87054W, 87055W, 87058W, 87063W & I-7050, 7052, 7053, 7041, 7044, 7060, 7063, 7065 have built-in low speed D/I counters associated with each D/I channel. The max counter speed of these modules is 100 Hz. The counter value is ranging from 0 to 65535 and can be reset to 0.

To use these I-87xxx D/I counters on slot 0 ~ 7 of the controller, connect these I/O modules with a last character – "C" in the "I/O connection" window. For ex. "i_87052C" .

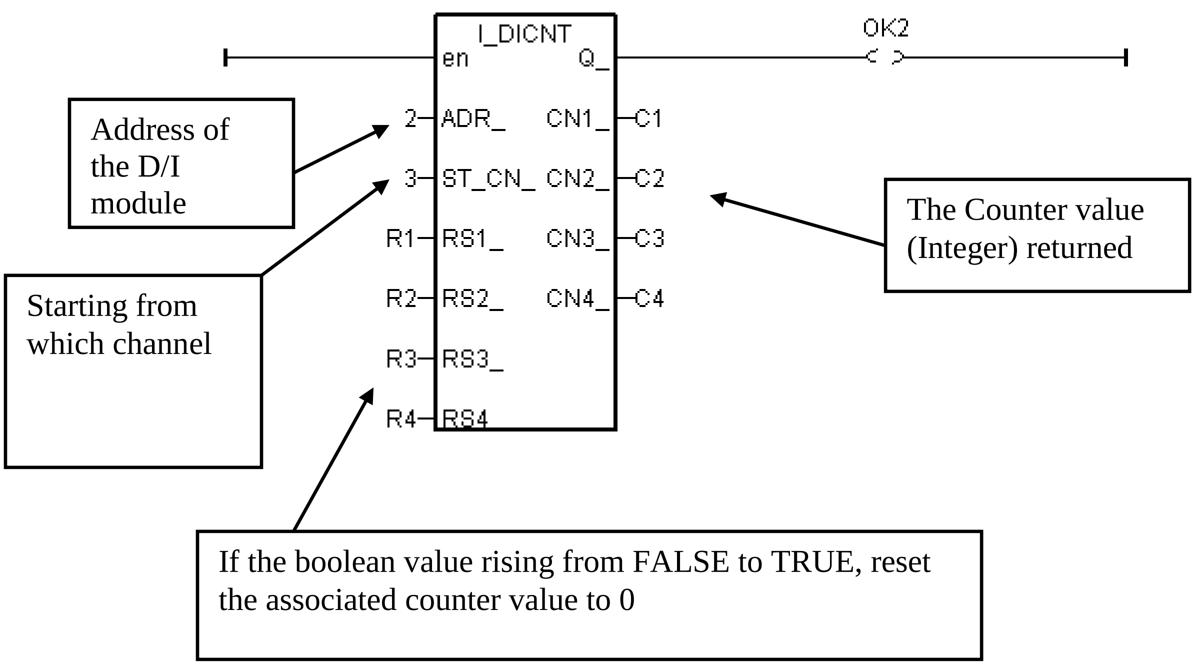




If the I-87xxx D/I Module is plugged in the 87K4, 87K5, 87K8 & 87K9 extension base module, or the I-7000 D/I module is used, Please refer to Chapter 6 to use "DCON Utility" to set the appropriate address, baud rate , then connect "Bus7000" on the "I/O connection" window.



Then using "I_DiCnt" block to get the "D/I Counter" value in the LD program. Each "I_DiCnt" can get 4 counters.

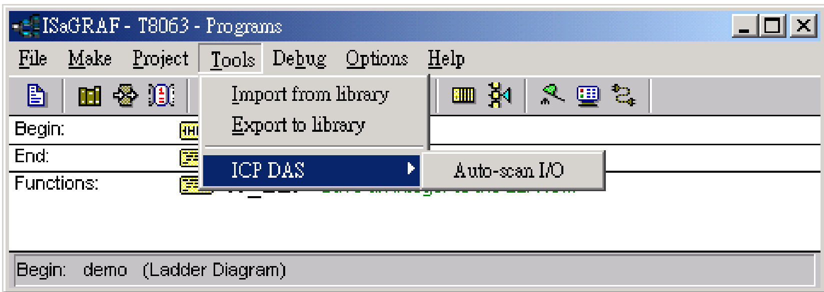


3.6: Auto-Scan I/O

Before you can use Auto-scan I/O utility, please refer to section 1.2 to make sure the "ICP DAS Utilities For ISaGRAF" has been installed. (Note: Not all the I/O boards are support Auto-scan)

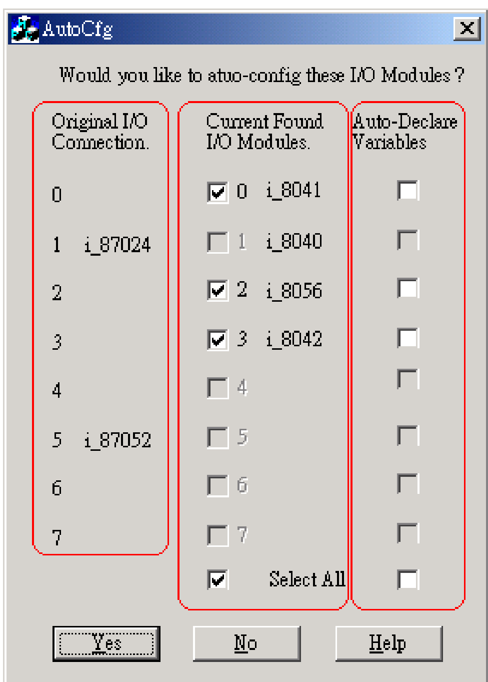
What is Auto-scan I/O :
It's a tool for ISaGRAF to easily configure your I/O connection and automatically declare variables for each I/O channel in ISaGRAF controllers.

How to use ?
A. Open your ISaGRAF program.
B. Click on "Tools/ICP DAS/Auto-scan I/O" to run Auto-scan.



C. The Auto-scan I/O is divided into three area.

Original I/O Connection shows the modules that already exist in your I/O connection at the first eight slots of your ISaGRAF project.
Current Found I/O Modules shows the I/O modules that detected in your controller (By RS-232 or TCP/IP).
Auto-Declare Variables shows what modules that you want Auto-scan to automatically declare variables for you also.



D. In the “Current Found I/O Modules.” area:

The check box will be enabled only when an I/O module is detected in the controller and the slot is **not used** by original I/O connection.

E. In the “Auto-Declare Variables”:
The check box can be enabled only when one I/O module **is checked** in the current found area.

F. You can check the “Select All” to check all available boxes in the respective area.

What is necessary for Auto-scan I/O ?

A. Make sure the “Link setup” parameter is correct (COM1, COM2, Ethernet, etc).

B. Plug in I/O boards first before your ISaGRAF can detect them.

Naming rules of automatically declared variables

Name format : Type_Slot_Channel

Type:
Digital Input : DI
Digital Output : DO
Analog Input : AI
Analog Output : AO

Slot : one digital slot number.
Channel : two digital channel numbers.

For ex. :
DI_0_02 , Digital Input channel at channel No.2 of slot 0.
AI_5_06 , Analog Input channel at channel No.6 of slot 5.
DO_2_12, Digital Output channel at channel No.12 of slot 2.
AO_1_03, Analog Output channel No. 3 of slot 1.

3.7: PWM Output

The scan time of the ISaGRAF controller depends on the ISaGRAF program and the hardware driver. For normal usage, the scan time is about 5 to 40 ms. It may go up to 100 ms sometime when the user's ISaGRAF program is very complicated. It is not easy to generate a precise periodic pulse output because the scan time of ISaGRAF is always varying, for example, a square curve of 2 ms OFF & then 1 ms ON. To achieve this kind of application, ISaGRAF provide PWM output functions.

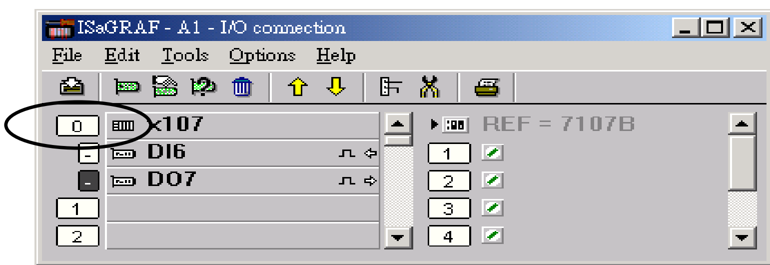
Note: please refer to http://www.icpdas.com/faq/isagraf.htm > FAQ-105 for more about the high speed PWM, such as I-8088W, I-87088W or I-7088W.

Only parallel Output boards are supported the PWM function. As follows:
I-8037W, 8041W, 8042W, 8054W, 8055W, 8056W, 8057W, 8060W, 8063W, 8064W, 8065W, 8066W, 8068W, 8069W (The PWM function is not fit for Relay output, because the response time is too slow.)

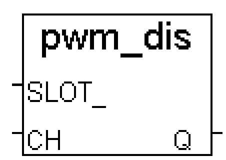
For **I-7188EG/XG, µPAC-7186EG**, only the X-board with digital output channels are available with PWM function.

For **µPAC-5xx7, WP-5xx7,** only the XW-board with digital output channels are available with PWM function.

Note:
1. Max. 8 PWM outputs can be used at the same time.
2. I-7188EG/XG, µPAC-7186EG must connect the Xxxx board at slot 0 (the WP-5xx7 requires the XW-board), or the PWM function will not work.



PWM_dis Disable PWM output



Parameters:

SLOT_	integer	Which slot ? 0 ~ 7
CH_	integer	Which channel ? 1 ~ 32.

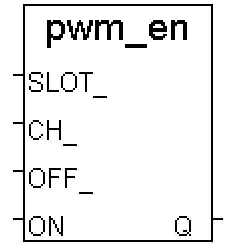
Return:

Q_	boolean	TRUE: Ok . FALSE: wrong input parameters, too many PWM outputs been enabled, or the associate output channel is not found.

Note:
1. After calling PWM_dis, the associate output will then be controlled by the ISaGRAF cycle engine.
2. Max 8 output channels can call PWM_en, PWM_en2, pwm_ON, pwm_OFF at one controller.

Example: I-8xx7, iP-8xx7: demo_63 , WinCon: Wdemo_22

PWM_en Enable PWM to output until PWM_dis is called



Parameters:

SLOT_	integer	Which slot ? 0 ~ 7
CH_	integer	Which channel ? 1 ~ 32.
OFF_	integer	Off time, scale is 1 ms.
ON_	integer	On time, scale is 1 ms.

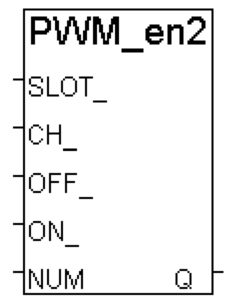
I-8xx7, I-7188EG/XG, µPAC-7186EG, µPAC-5xx7 , iP-8xx7: 1 ~ 32,767
WP-8xx7, WP-5xx7, XP-8xx7-CE6, XP-8xx7-Atom-CE6,VP-25W7:
2 ~ 32766 (it must be set as multiples of 2)

Return:

Q_	boolean	TRUE: Ok . FALSE: wrong input parameters, too many PWM outputs been enabled, or the associate output channel is not found.

Example: I-8xx7, iP-8xx7: demo_63 , W-8xx7: Wdemo_22

PWM_en2 Enable PWM to output a given number of pulse



Parameters:

SLOT_	integer	Which slot ? 0 ~ 7
CH_	integer	Which channel ? 1 ~ 32.
OFF_	integer	Off time, scale is 1ms.
ON_	integer	On time, scale is 1 ms.

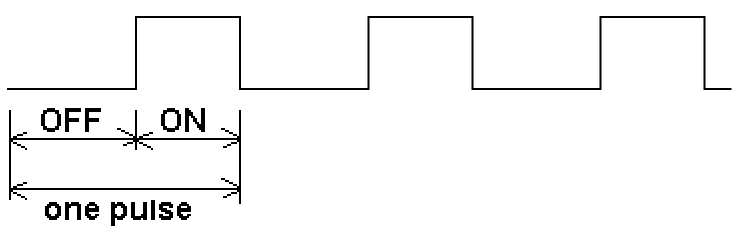
I-8xx7, I-7188EG/XG, µPAC-7186EG, µPAC-5xx7 , iP-8xx7: 1 ~ 32,767
WP-8xx7, WP-5xx7, XP-8xx7-CE6, XP-8xx7-Atom-CE6,VP-25W7:
2 ~ 32766 (it must be set as multiples of 2)

NUM_	integer	number of pulse to output, 1 - 2,147,483,647. If gives a negative value to NUM_, for ex. –1, it will ouput indefinitely until pwm_dis been called.

Return:

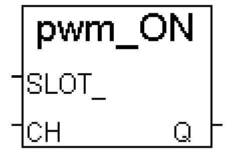
Q_	boolean	TRUE: Ok . FALSE: wrong parameters, too many PWM outputs been enable, or the associate output channel is not found.

PWM output curve:



Note:
1. Every time the PWM_en or PWM_en2 is called, it will reset its internal tick to 0, and re-start ticking to OFF, ON, OFF, ON, ...
2. If the given number of pulse of pwm_en2 is reached, it will stop & disable PWM auomatically (Calling PWM_dis for pwm_en2 is not necessary).
3. PWM_sts can be used to test if pwm_en2 reaches its given number of pulse or not.
4. Max 8 output channels can call PWM_en, PWM_en2, pwm_ON, pwm_OFF at one controller.
5. Do not enable the channel that is already enabled.

pwm_ON Set parallel D/O to TRUE immediately



Parameters:

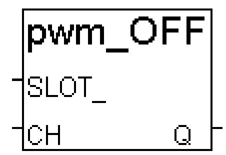
SLOT_	integer	Which slot ? 0 ~ 7
CH_	integer	Which channel ? 1 ~ 32.

Return:

Q_	boolean	TRUE: Ok . FALSE: wrong input parameters, too many PWM outputs been enabled, or the associate output channel is not found.

Example: I-8xx7, iP-8xx7: demo_63 , W-8xx7: Wdemo_22

pwm_OFF Set parallel D/O to FALSE immediately



Parameters:

SLOT_	integer	Which slot ? I-8xx7: 0 ~ 7, I-7188EG/XG: 0, W-8xx7: 1 ~ 7
CH_	integer	Which channel ? 1 ~ 32.

Return:

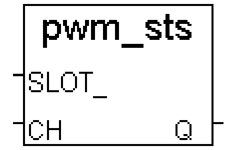
Q_	boolean	TRUE: Ok . FALSE: wrong input parameters, too many PWM outputs been enabled, or the associate output channel is not found.

Example: I-8xx7, iP-8xx7: demo_63 , W-8xx7: Wdemo_22

Note:

1. Max 8 output channels can call PWM_en, PWM_en2, pwm_ON, pwm_OFF at one controller.
2. pwm_ON will set the associate parallel D/O to TRUE immediately.
3. pwm_OFF will set the associate parallel D/O to FALSE immediately.
4. If users wish to enable one D/O as PWM output by PWM_en or PWM_en2 after pwm_ON & pwm_OFF has been called, please disable it first by PWM_dis, then call PWM_en or PWM_en2.

PWM_sts Get PWM status



Parameters:

SLOT_ integer Which slot ? 0 ~ 7

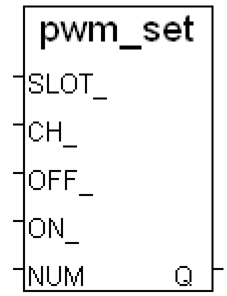
CH_ integer Which channel ? 1 ~ 32.

Return:

Q_ boolean TRUE: this channel has been enabled and still in an operation.
FALSE: disable (for pwm_en2 been called, it means the given pulse number is reached).

Note:

1. Max 8 output channels can call PWM_en, PWM_en2, pwm_ON, pwm_OFF at one controller.
2. This function can be used to test if "PWM_en2" reachs its given pulse number or not.



PWM_set Dynamically change the ON_, OFF_ & NUM_ setting

Parameters:

SLOT_ integer Which slot ?

CH_ integer Which channel ? 1 ~ 32.

OFF_ integer Off time, scale is 1ms.

ON_ integer On time, scale is 1 ms.

I-8xx7, I-7188EG/XG, µPAC-7186EG, µPAC-5xx7 , iP-8xx7: 1 ~ 32,767

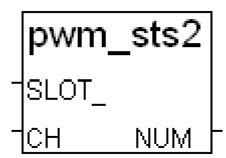
WP-8xx7, WP-5xx7, XP-8xx7-CE6, XP-8xx7-Atom-CE6,VP-25W7: 2 ~ 32766 (it must be set as multiples of 2)

NUM_ integer number of pulse to output, 1 - 2,147,483,647. If gives a negative value to NUM_, for ex. –1, it will ouput indefinitely until pwm_dis been called.

Return:

Q_ boolean TRUE: Ok . FALSE: An error occurred.

PWM_sts2 Get pulse number been output by pwm_en2 and pwm_en



Parameters:

SLOT_ integer Which slot ? 0 ~ 7

CH_ integer Which channel ? 1 ~ 32.

Return:

NUM_ integer the pulse number already been output by pwm_en & pwm_en2.

Note:

1. This function only works when "pwm_en” & “pwm_en2" have been enabled.
2. The returned pulse number may less than the given number in "pwm_en2" when it reaches the destination. For example, gives 20000 pulse in "pwm_en2", however when reach the end, the "pwm_sts2" may return only 19998.
3. If the ouput number given in the "pwm_en2" is a negative value, the pulse output will never stop unless the "pwm_dis" command is given. Then the returned number of "pwm_sts2" will become 0 , 1, 2, ... to 2,147,483,647 and then go back to 0, 1, 2, ...

Example: I-8xx7, iP-8xx7: demo_63 , W-8xx7: Wdemo_22

3.8: Counters Built in Parallel D/I Boards

Only parallel input boards plug **at slot 0** are supported D/I counters (XP-8xx7-Atom-CE6, XP-8xx7-CE6: slot 0). As follows, I-8040W, 8042W, 8051W, 8052W, 8053W, 8054W, 8055W, 8058W, 8063W

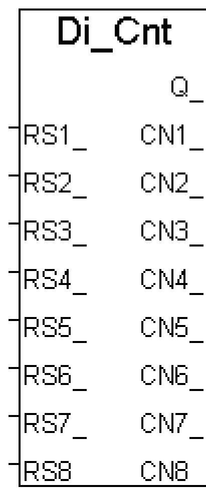
For I-7188EG/XG, µPAC-5xx7 and µPAC-7186EG, only the X-xxx boards with digital input channels are available with D/I counter.

For **µPAC-5xx7, WP-5xx7,** only the XW-board with digital input channels are available with D/I counter.

Note: please refer to http://www.icpdas.com/faq/isagraf.htm > FAQ-100 for more about the high speed Counter, Frequency, such as I-8084W or I-87084W.

The max channel amount of parallel D/I counter available in one controller is up to 8. And the max frequency of counter input is up to 500 Hz with minimum NO and OFF width > 1 ms.

The below c function block is for getting/reset D/I counters at slot 0.



Parameters:

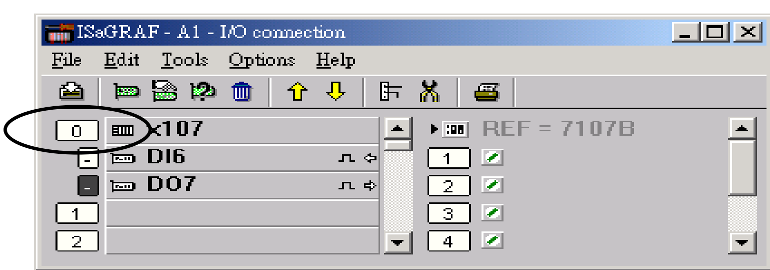
RS1_ ~ RS8_	boolean	Reset the associated D/I counter when rising from False to True

Return:

Q_	boolean	work ok. : TRUE. If Q_ is FALSE , it means “No parallel D/I module found at slot 0 "
CN1_ ~ CN8_	integer	DI Counter value of channel No. 1 to 8. Valid value is ranging from 0 to 2,147,483,647. If value is over 2,147,483,647, it will restarts at 0.

Note:

Only Parallel D/I board plug in slot 0 supports “Di_Cnt” (Do not support other slot).
Only the first 8 D/I channel support “Di_Cnt”.
I-7188EG/XG, µPAC-7186EG must connect the Xxxx board at slot 0 (the WP-5xx7 requires the XW-board), or the “Di_Cnt” will not work.



Demo:

Please refer to I-8xx7 and iP-8xx7’s demo_52 & demo_53.

Chapter 4. Linking Controllers To An HMI Program

This chapter details how to make data from the I-8xx7, I-7188EG/XG, µPAC-7186EG, µPAC-5xx7, iP-8xx7, VP-2117, WP-8xx7, WP-5xx7, XP-8xx7-CE6, XP-8xx7-Atom-CE6, VP-25W7 and VP-23W7 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 controller system.

After you realize the material described in section 4.1, if you would like to use the ISaGRAF controller as a **Modbus RTU or Modbus TCP/IP I/O**, you may refer to section 4.3. 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 controller . Section 4.4 illustrates how to link a "Touch 510" monitor to an ISaGRAF controller system.

Note:
1. Please refer to "Getting Started Manual" for each PAC to know how to enable the Modbus RTU Slave port.
2. All the ISaGRAF controllers support Modbus TCP/IP Slave protocol at its Ethernet port.
3. I-8417 / 8817 and iP-8xx7's COM1: RS-232 and COM2: RS-485 default supports Modbus RTU Slave.
4. I-8437-80/8837-80's COM1: RS-232 default supports Modbus RTU Slave protocol. I-7188EG/XG & µPAC-5xx7, µPAC-7186EG's COM1 default supports Modbus RTU Slave protocol.
5. WP-8xx7, WP-5xx7, XP-8xx7-CE6, XP-8xx7-Atom-CE6,VP-25W7 and VP-23W7 default no support Modbus RTU Slave port. But, their Ethernet port has enabled the Modbus TCP/IP Slave.

4.1: Declaring Variable Addresses For Network Access

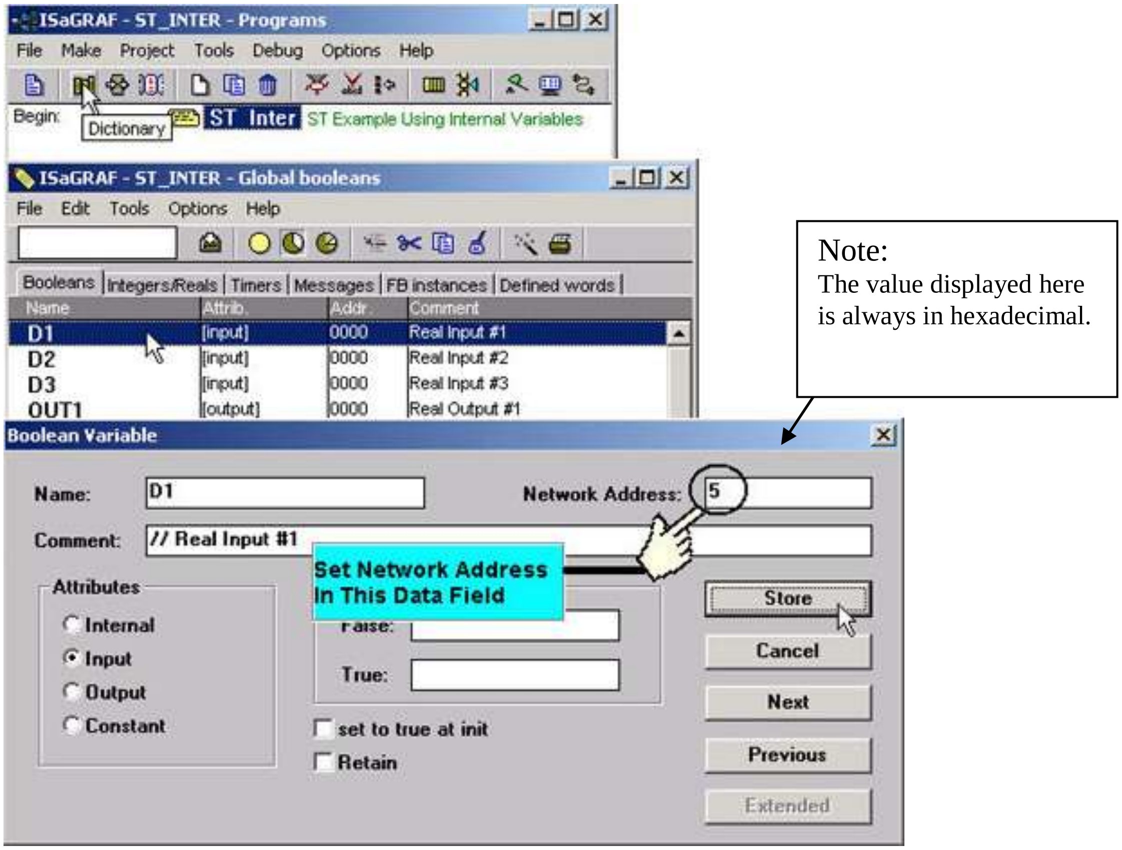
To make data from an ISaGRAF controller system available to other software programs or HMI devices, you must first declare the variable with a "Network Address". The variable must be declared with a network address number that is in the "Modbus" format. Other software programs or HMI devices will access the controller information through these network addresses.

Note:
1. The valid network addresses for an **I-8xx7, I-8437-80, I-8837-80, I-7188EG/XG, VP-2117, µPAC-5xx7, µPAC-7186EG and iP-8xx7** controller system is from 1 to FFF in hexadecimal (**1 ~ 4095**). Network address **5001 to 8072** is for word and integer arrays (please refer to Section 4.5).

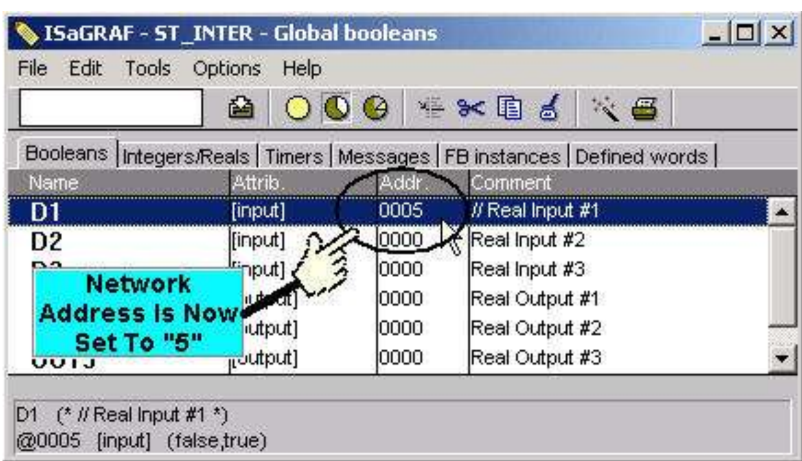
2. The valid network addresses for the WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6, VP-25W7 and VP-23W7 controller system is from 1 to 1FFF in hexadecimal (1 ~ 8191). Network address 10,001 to 19,216 is for word and integer arrays (please refer to Section 4.5).

There are two ways to assign a Modbus network address No. to a variable. One is as below figure. (To assign many Modbus Network address No. to the "Variable Array", please refer to Chapter 2.6)

Open an "ISaGRAF Programs" windows and click on the "Dictionary" icon, then double click on the variable to assign a network address number.

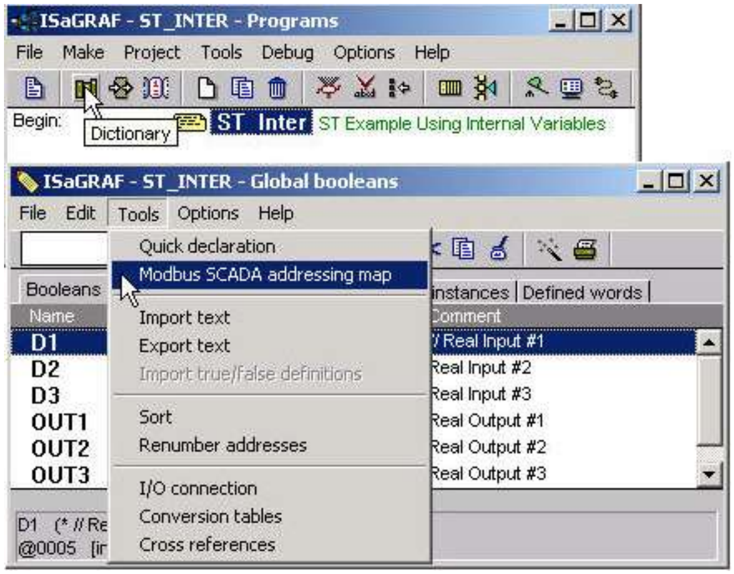


As above figure, When you click on the "Store" button you will see that "ISaGRAF Global Variables" window will now be updated with the new network address for the variable.

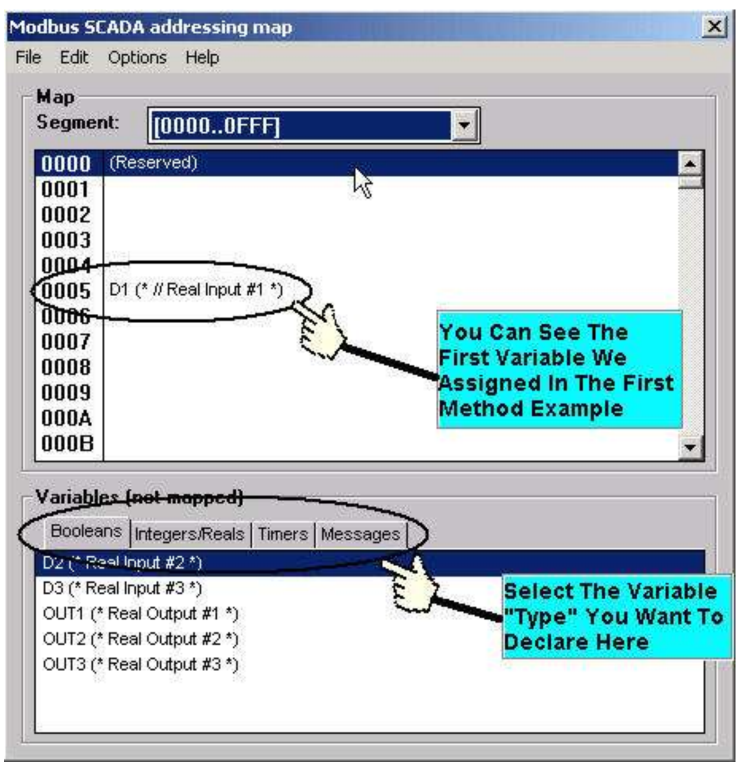


The second method for assigning network addresses to variables requires that you declare the variables BEFORE you assign them. This method allows you to assign numerous network address variables before you link them to an ISaGRAF program.

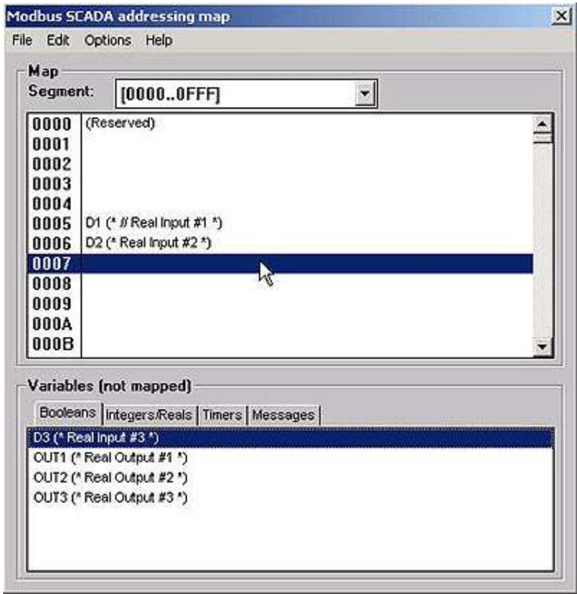
When you click on "Modbus SCADA Addressing Map" (SCADA is an industrial process control acronym that stands for "Supervisory Control And Data Acquisition") the "Modbus SCADA Addressing Map" window will open.



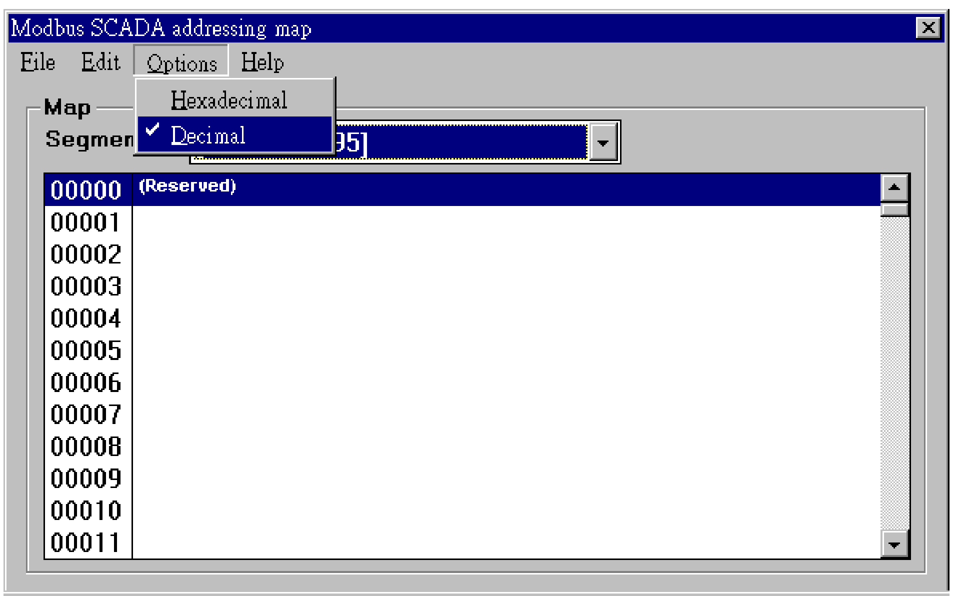
Note that one of the variables (D1) is already assigned from our previous example. You will note that the other variables that are not yet mapped are displayed in the lower portion under the "Variables (Not Mapped)" portion of the "Modbus SCADA Addressing Map" window.



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 way, 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 following.



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 ISaGRAF controller system.

HMI or devices such as Indusoft, Iconics, Citech, Wizcon, Kepware's OPC server, Intellution's iFix, Wonderware's "Intouch", National Instruments "Labview", and ICP DAS's Touch 506L, Touch 506T and Touch 510T do have the exact same addressing scheme as the ISaGRAF controller system.

The network address definition scheme for some HMI and ISaGRAF PAC are different. If you are assigning a network address of "B" (hexadecimal) of these products, the PAC's network address should be set to "C". A network address of "2" should be associated with a network address of "3" in the ISaGRAF controller system.

Another things mistaked very often is the first digit of the network address of many SCADA and HMI softwares resprent the data type and Read/Write authority not one part of the network address. (The max. address number for the I-8xx7, I-7188EG/XG, µPAC-7186EG, µPAC-5xx7 and iP-8xx7 is 4095. The max. address number for the WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6, VP-25W7 and VP-23W7 is 8191.)

For example, the network address relation between "iFix" and ISaGRAF is as below.

iFix(Decimal)	ISaGRAF PAC (Decimal)
00001 (R/W Boolean)	1
00002 (R/W Boolean)	2
10010 (Read Boolean)	10
10011 (Read Boolean)	11
31000(Read Word)	1000
31001(Read Word)	1001
40101(R/W Word)	101
42001(R/W Word)	2001

ICP DAS has not been able to test every possible SCADA or 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 ISaGRAF 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 communicating with the I-8xx7, µPAC-7186EG, µPAC-5xx7, I-7188EG/XG , iP-8xx7 and VP-2117 controller system, **One single Modbus frame** cannot request more than **255 bits (or Boolean)** and cannot request more than **122 words** in one single modbus frame. It should be divided into 2 or more reading frames to achieve it. For the WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6, VP-25W7 and VP-23W7, **One single Modbus frame** can request up to **1968 bits (or Boolean)** and also cannot request more than **122 words** in one single modbus frame.

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

Modbus protocol provides function 3 and 4 for reading multiple words while function 6 and 16 to write words. Please refer to Chapter 5 for more information about the protocol.

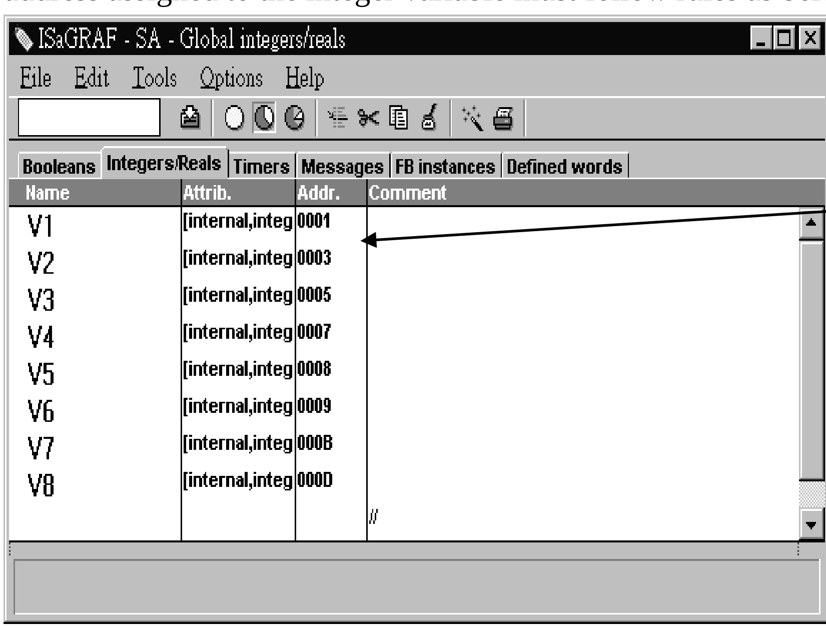
The **word** defined in the Modbus protocol of ISaGRAF controllers is like a signed short integer, which occupies 2 bytes and range from –32,768 (8000 in hexa.) to +32,767 (7FFF in hexa.). It is normally used to describe the behavior of analog I/O channels. For examples, the I-87017W I/O board (please refer to section 3.2)

I-87017W :

Range ID (hexadecimal)	Electrical Range	Values on the channel (decimal)		
		-32768	0	+32767
8 (default)	± 10V	- 10V	0V	+ 10V
9	± 5V	- 5V	0V	+ 5V
A	± 1V	- 1V	0V	+ 1V
B	± 500mV	- 500mV	0mV	+ 500mV
C	± 150mV	- 150mV	0mV	+ 150mV
D	± 20mA	- 20mA	0mA	+ 20mA

The **long word** defined in the Modbus protocol of ISaGRAF controllers 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 in 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 Mudbus transportation format. Since a long word occupies 2 words (4 bytes), to R / W long word through Modbus, the network address assigned to the integer variable must follow rules as below.

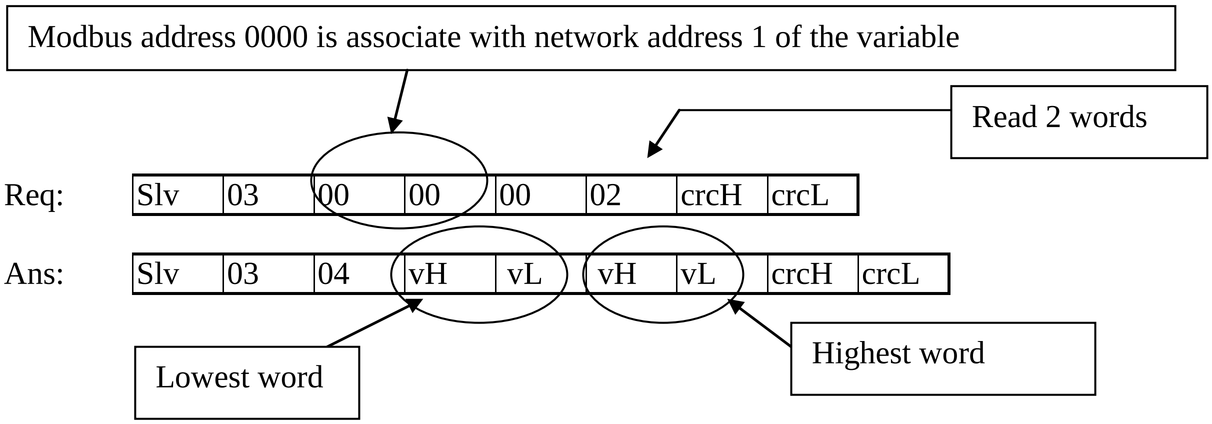


V1 is assigned to a network address “1”. If the network address “2” is not assigned to any other variable, V1 will occupy a long word (4 bytes) in the Modbus transportation formate.

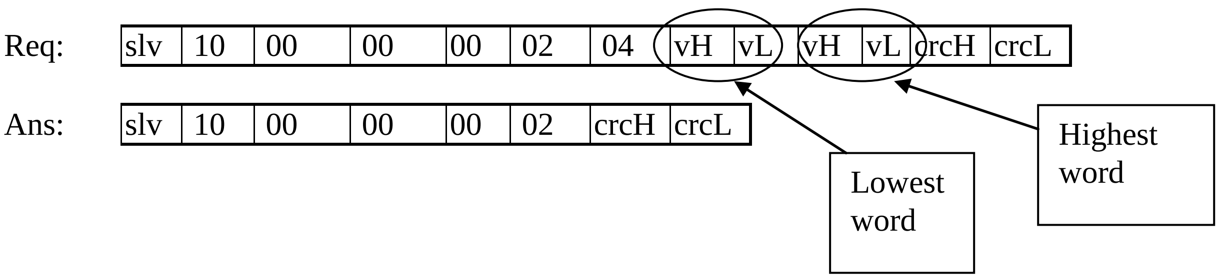
However if “2” is assigned to one another variable, V1 will only occupy one word (2 bytes) in the Modbus transportation format.

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.

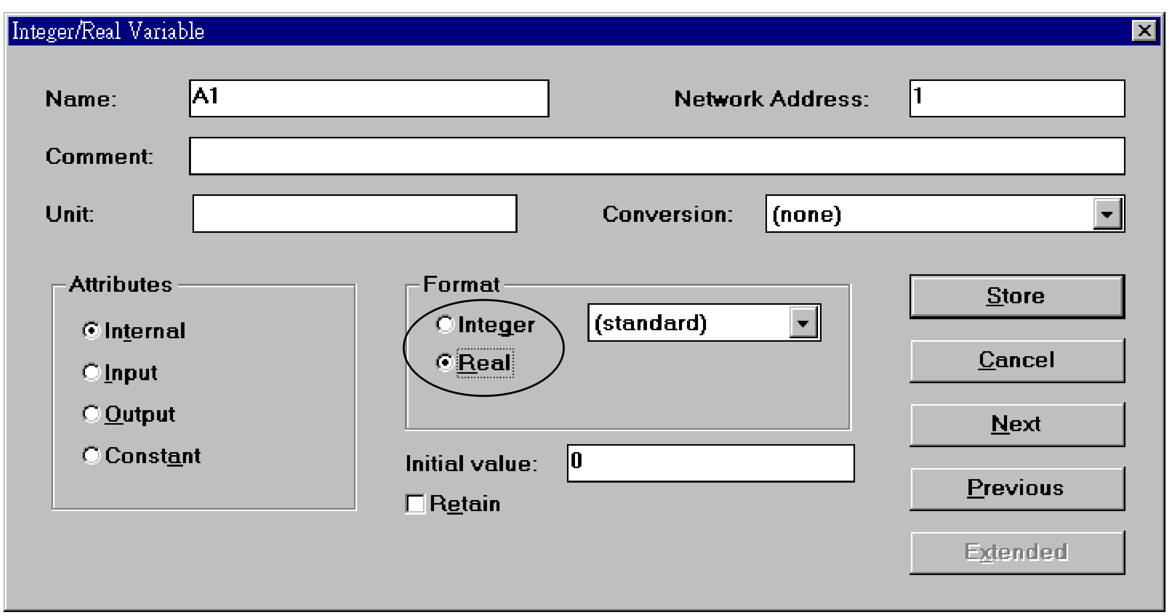
To read **long word** value of V1 is to read **2 words** by using modbus function 3 or 4 (please refer to section 5.1).



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.



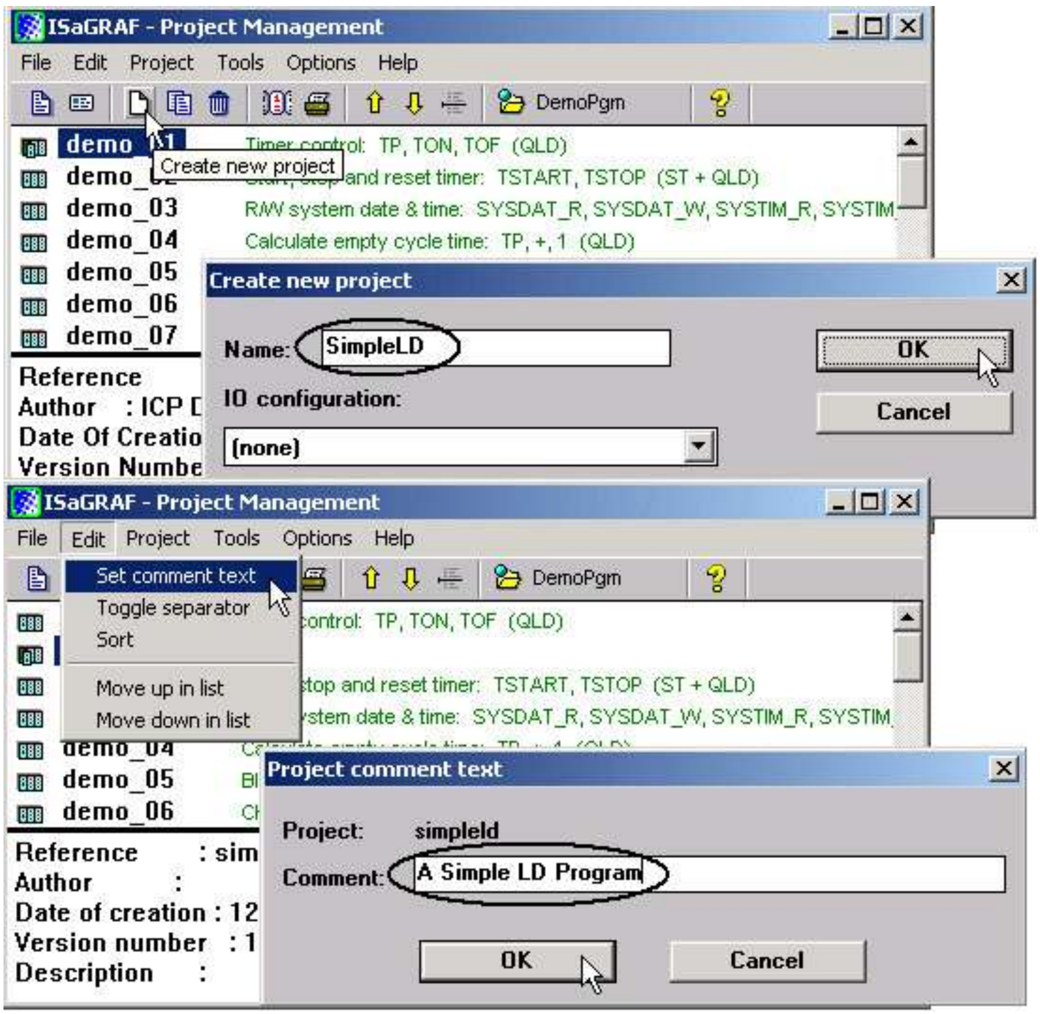
There is much available HMI software on the market. You must be noted whether it supports the Modbus protocol. Just be careful to assign the correct network address on ISaGRAF.

4.3: Using I-8xx7 As A Modbus I/O Or A Modbus TCP/IP I/O

There are some configurations that the HMI software gathers the I/O data from some called Modbus I/O modules. These I/O modules scan each input channel and refresh the output channels when need. Most of time there are no control logic inside these I/O modules, they are controlled by the HMI. To fit such kind of usage, the I-8xx7, iP-8xx7, I-8437-80, I-8837-80 can be treated as a Modbus I/O module; additionally the I-8437-80, I-8837-80 and iP-8xx7 can be treated as a Modbus TCP/IP I/O module. To do that, follow the following procedures (If you are not familiar with the ISaGRAF programming, recommended to review Chapter 2).

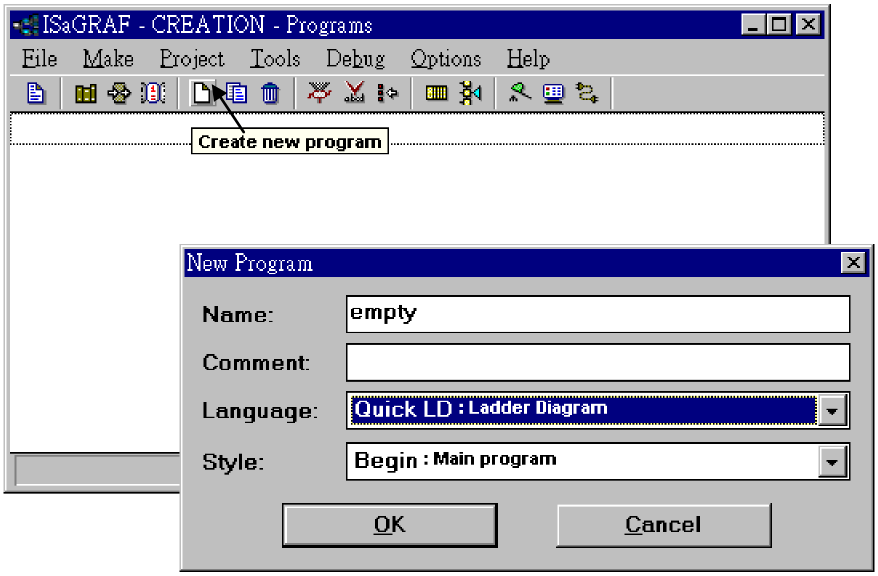
Create a new project
You may refer to section 2.1.1.2

Example:



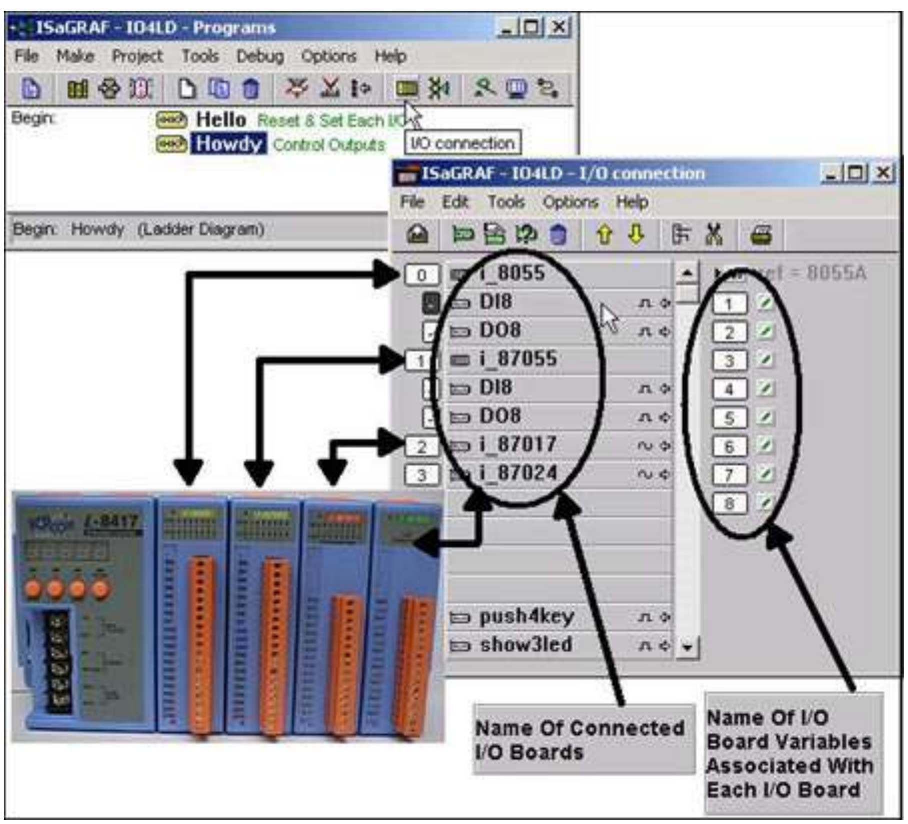
Create an empty program
No control logic in this program.

Example:



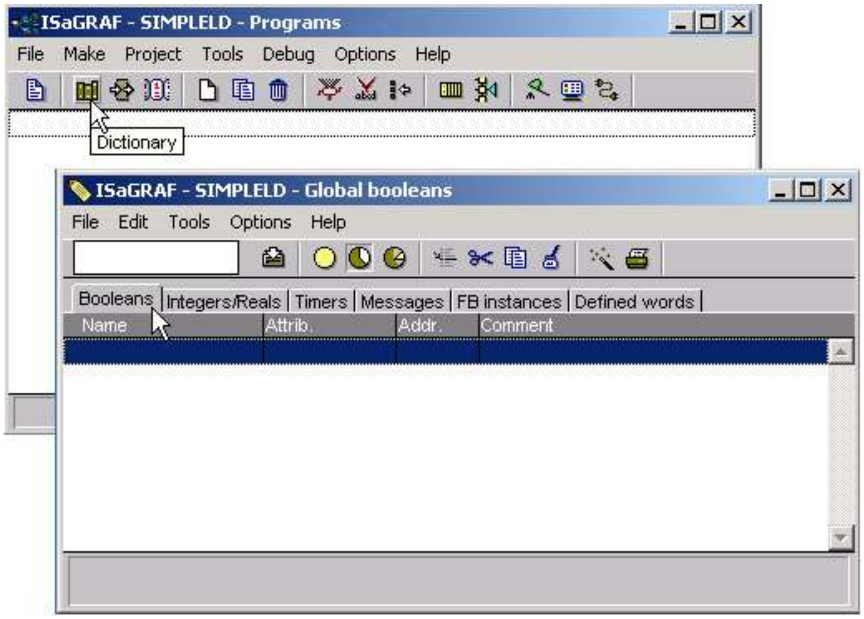
Connect I/O modules
You may refer to section 3.1

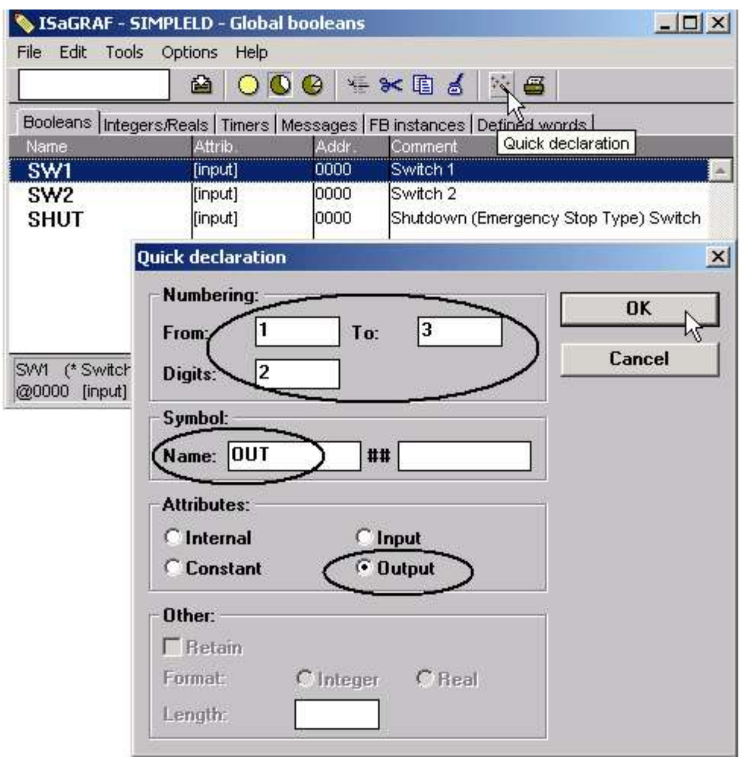
Example:



Declare Variables associated with the channels of connected I/O modules.
You may refer to section 2.1.1.3

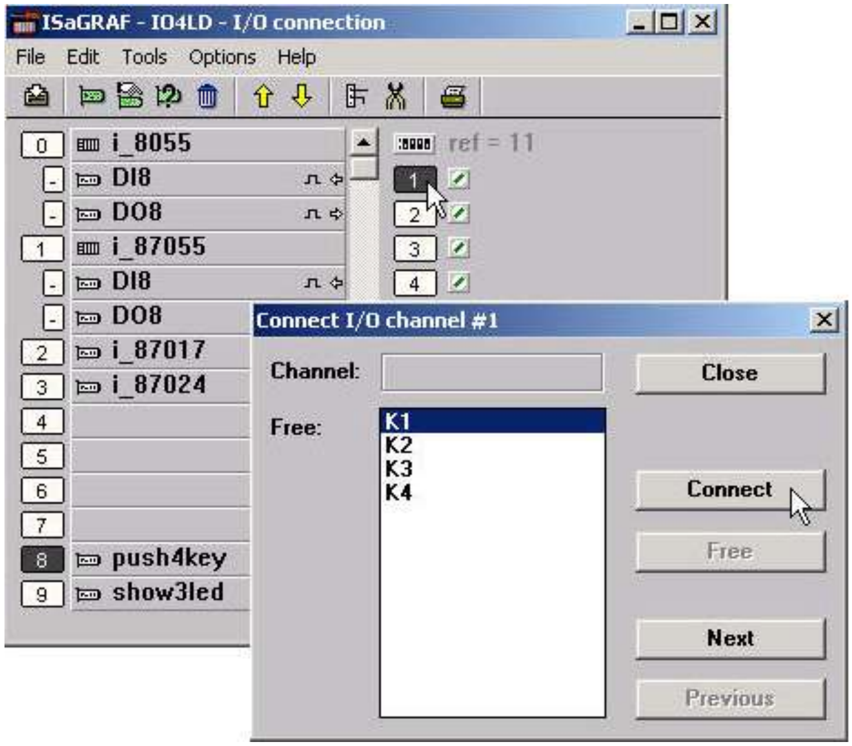
Example:





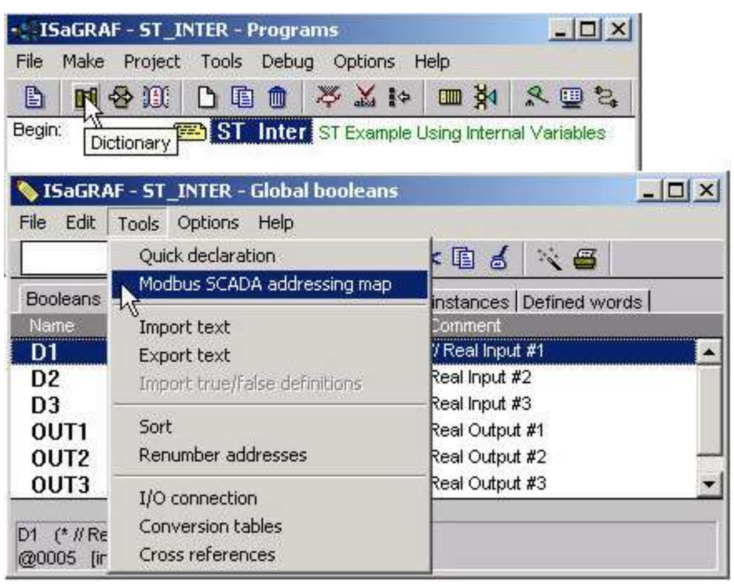
Link Variables to the associated channels of connected I/O modules.
You may refer to section 3.1.2

Example:



Assign the linked Variable a network address No.
You may refer to section 4.1

Example:



Compile & download the project

You may refer to section 2.1.3 & 2.1.5

Note:

If using Modbus TCP/IP protocol, make sure the Net ID (section 1.3.1), IP and Mask address (appendix B) for the I-8437-80, I-8837-80 and iP-8xx7 is set up correctly.

The HMI can access to I/O channels through the associated network address now!

4.4: Linking ISaGRAF PAC To Touch 500

Touch500 series HMI support below protocols to link to ICP DAS ISaGRAF controllers.

Item	Protocols
Touch-506L	Modbus RTU RS-232 , Modbus RTU RS-485
Touch-506TE	Modbus RTU RS-232 , Modbus RTU RS-485, Modbus TCP/IP
Touch-510T	Modbus RTU RS-232 , Modbus RTU RS-485

Please install "EasyBuilder 500" software (Ver. 2.7.1 or later version) first before you can program the Touch 506L, 506T, 510T HMI. You may download the new released software and manual from below web site

http://www.icpdas.com/download/others/touch/touch.htm "**setup.zip**"

or run "setup.exe" at I-8000 CD-ROM:\napdos\others\touch\500series\setup\

RS-232 Cable Pin assignment of PC to Touch 500 series (For PC to download HMI screen).

PC 9-Pin DSUB Female (RS232)	Touch 510 (PC-232) 9-Pin DSUB Female
RXD 2	8 RXD
TXD 3	7 TXD
GND 5	5 GND

RS-232 Cable Pin assignment between controllers and Touch 500 series.

I-8000 COM1 & I-7188 COM1 (RS-232)	**Touch 506TE/506L/510T (PLC 232)**
9-Pin Dsub Male	**9-Pin Dsub Male**
2 TXD	2 TXD
3 RXD	3 RXD
5 GND	5 GND
	7 CTS
	8 RTS

WinCon COM2 (RS-232)	**Touch 506TE/506L/510T (PLC 232)**
9-Pin Dsub Female	**9-Pin Dsub Male**
2 RXD	2 TXD
3 TXD	3 RXD
5 GND	5 GND
	7 CTS
	8 RTS

RS-485 Cable Pin assignment between controllers and Touch 500 series

I-8417/8817 COM2 (RS-485)	**Touch 506TE/506L/510T (PLC 485)**
DATA +	2 R+
DATA -	1 R-

WinCon COM3 (RS-485)	**Touch 506TE/506L/510T (PLC 485)**
D +	2 R+
D -	1 R-

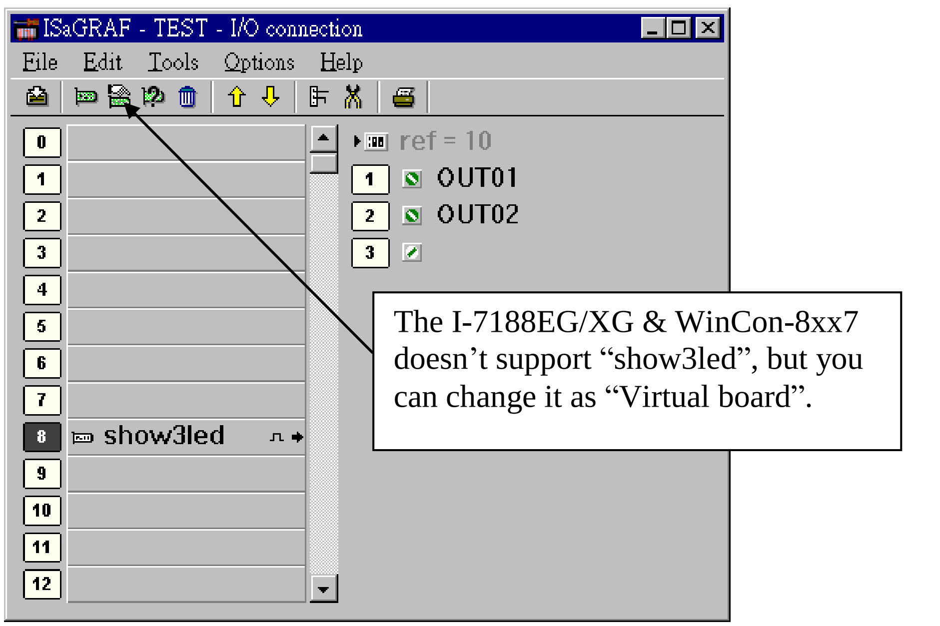
4.4.1: Program the ISaGRAF PAC

To make data of the ISaGRAF controller to be accessible to the Touch 510T, variables in the controller should be assigned a network address. Please refer to section 4.1, 4.2. If you are not familiar with the ISaGRAF programming, recommended to review Chapter 2.

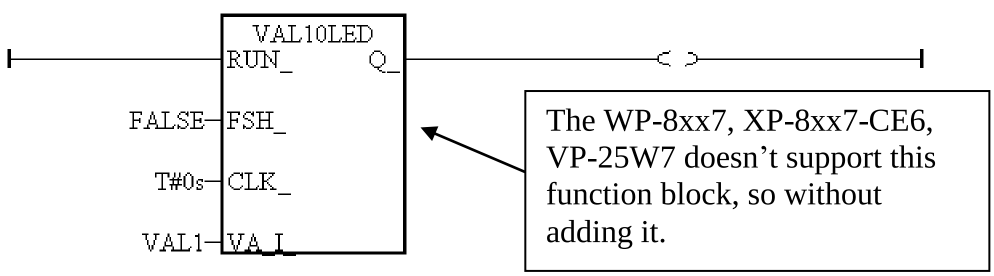
Variables used in this example.

Name	Type	Attribute	Network address	Others
OUT01	Boolean	Output	**0001**	-
OUT02	Boolean	Output	**0002**	-
VAL1	Integer	Internal	000A (10)	-

IO connection:



A simple LD program to show the "VAL1" to 7-segment LED:

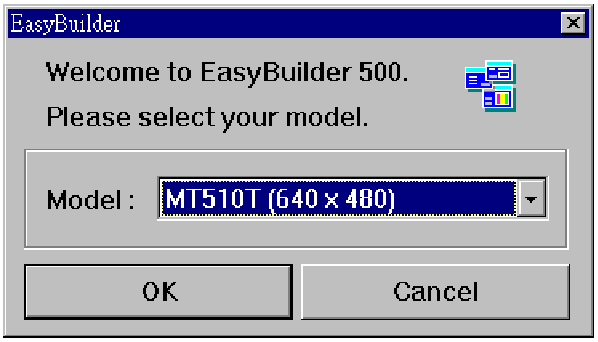


After you finish this project, compile and download it to the ISaGRAF controller.

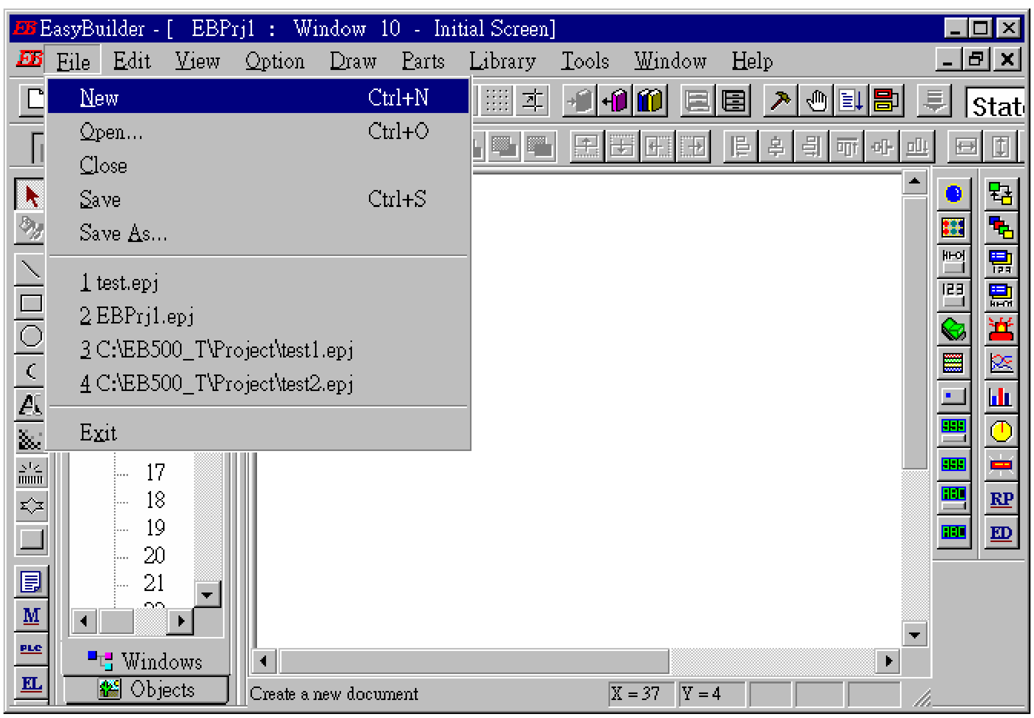
4.4.2: Program the Touch 500

The “EasyBuilder 500” software can be used to designe many useful pictures for Touch 500 series. This section illustrates a simple example to program a Touch 510T. For more information about programming on the Touch series, please refer to the user manual which is provided with the “Touch” series hardware.

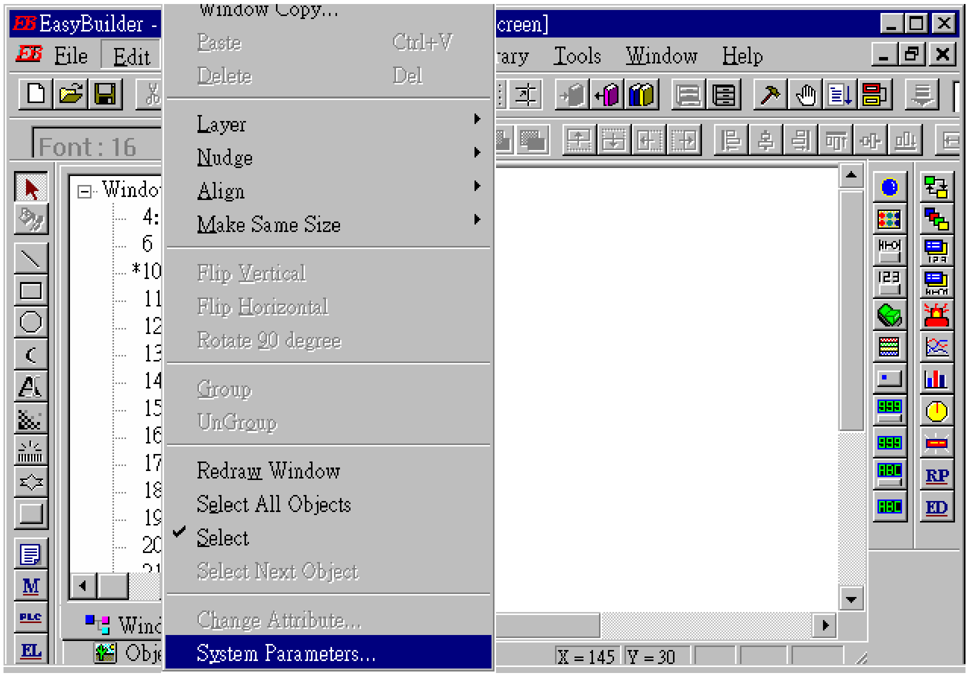
Click on the Windows "Start" button, then click on the "Program" button, then click on the "EasyBuilder" – “EasyBuilder 500” button. The following window will be displayed. Select the proper model for your application.



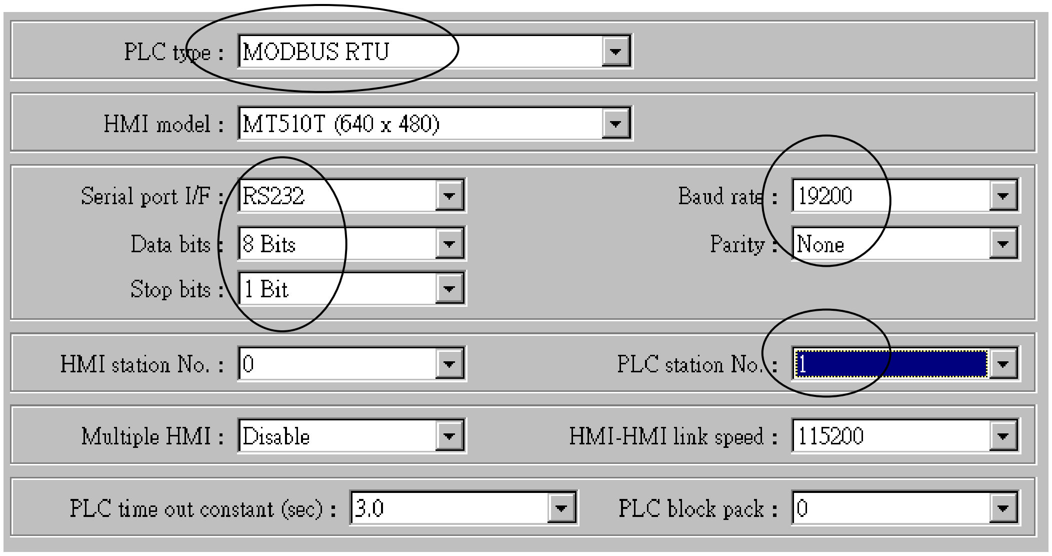
Click “File” – “New” to create a new project.



Click “Edit” – “System Parameters” to set the communication parameter between the Touch 510 and the ISaGRAF controller.

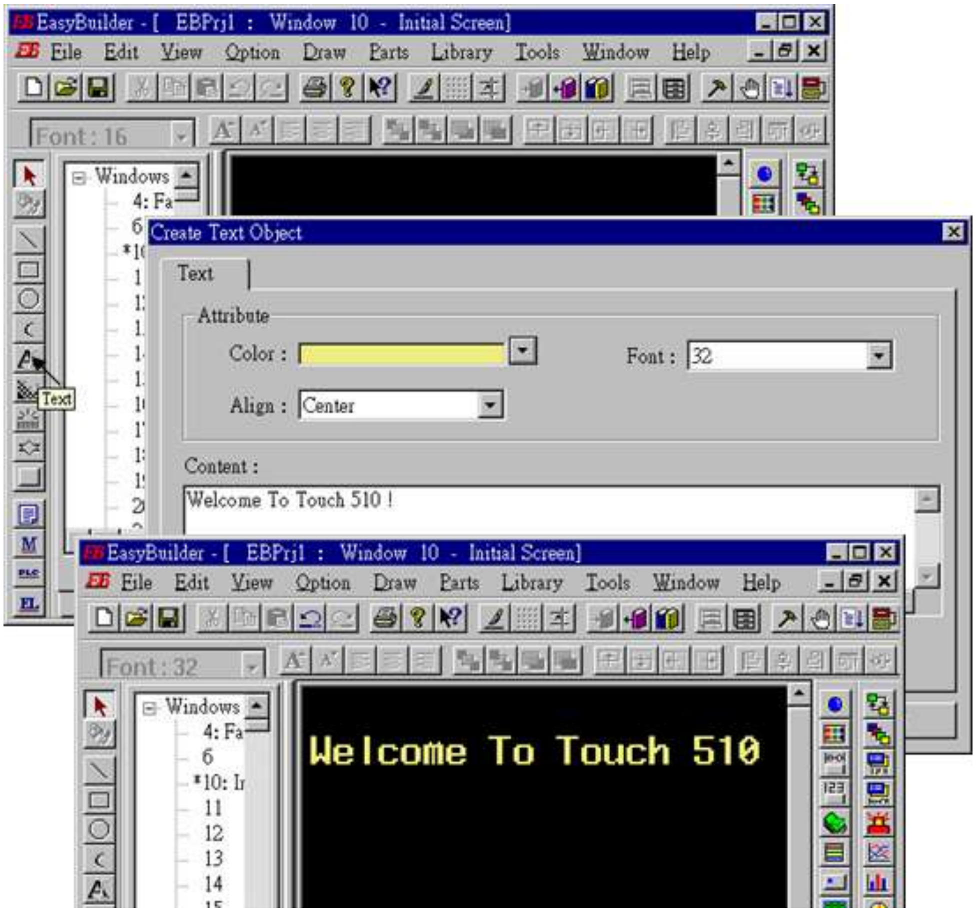


PLC type should be set to “**MODBUS RTU**”, Serial port set to “RS-232”, Data bits set to “8 Bits”, Stop bits set to “1 Bit”, Baud rate set to “19200”, Parity set to “None”, PLC station No. set to be equal to the Net-ID of the ISaGRAF controller (set to 1 in this example).

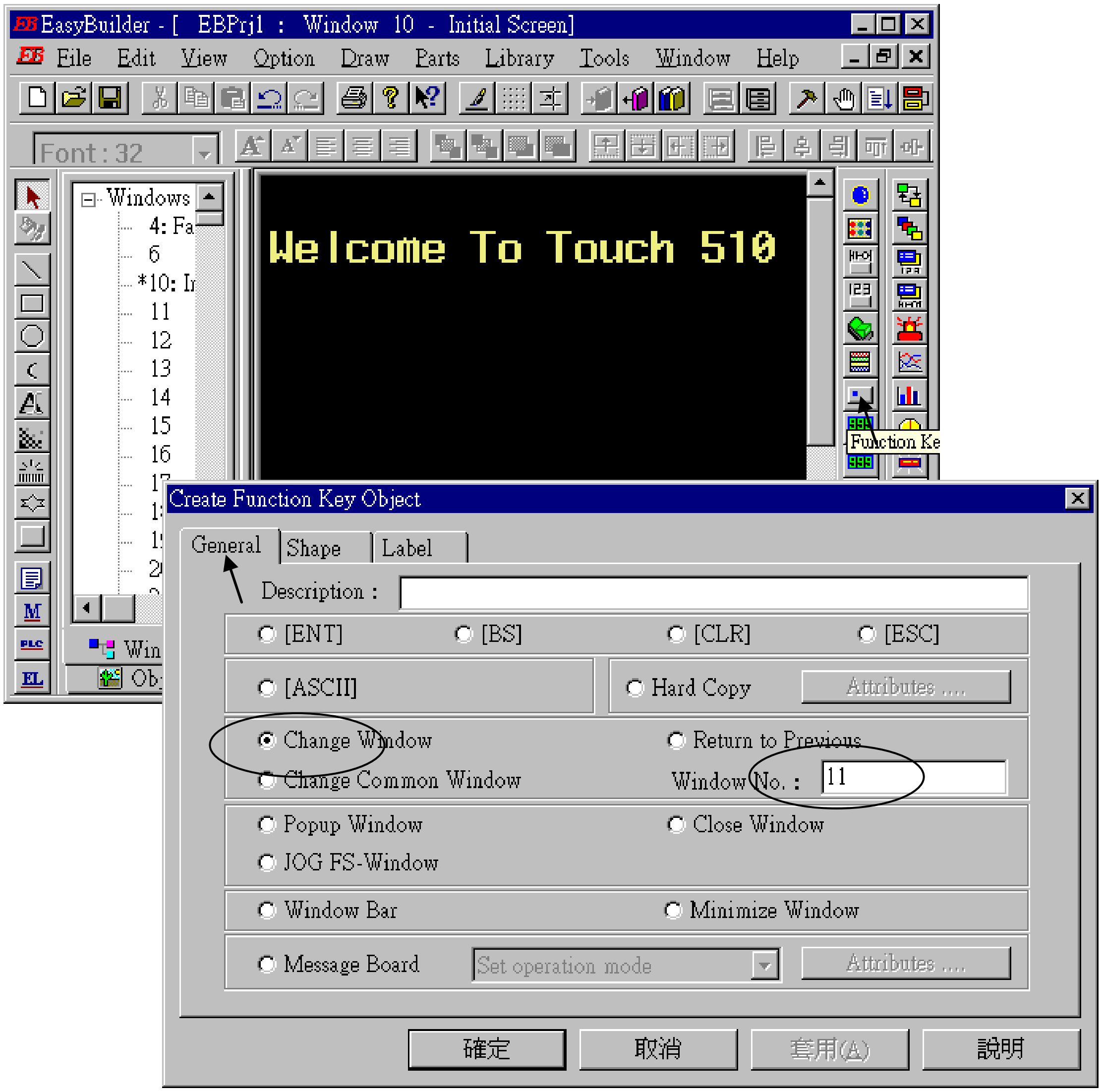


Note:
1. If using Touch506TE ‘s Ethernet to link to controller, please set PLC type as “MODBUS RTU TCP/IP” , PLC I/F port as “Ethernet” , Local IP address as Touch506TE ’s IP, Server IP address as controller ’s IP, PLC station No. as the same Net-ID No. of the controller (default is 1)
2. If the cable between the Touch 500 series and the controller is 2-wire RS-485, please set PLC type as “MODBUS RTU (RS-485 2W)”, PLC I/F port as “RS-485 2W”. Other setting is the same as RS-232.

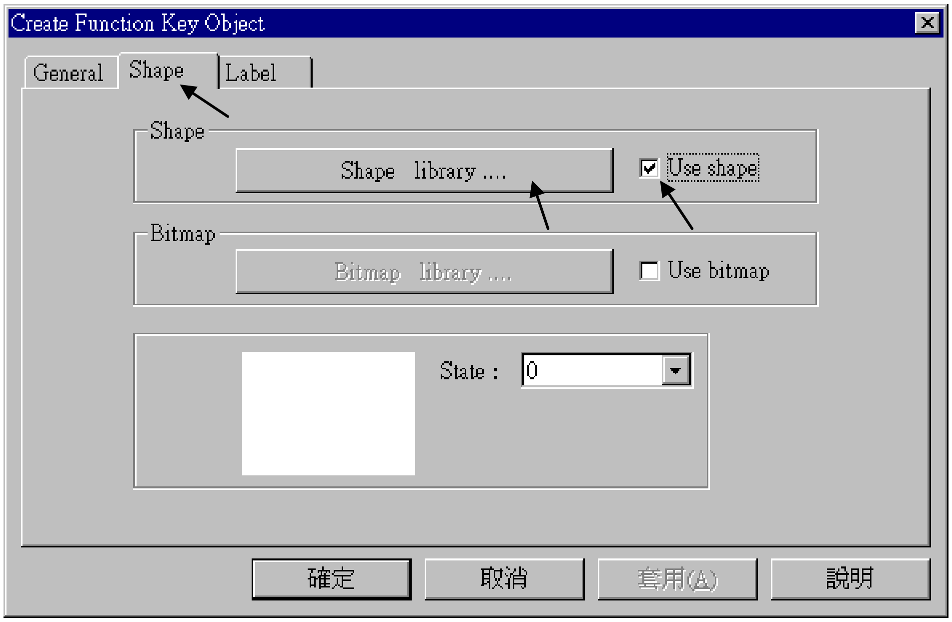
Click on “Text” to add a text. Select the prefered “Color”, “Font”, “Align” for the text and then enter the “Content”. And then place it to the proper position.



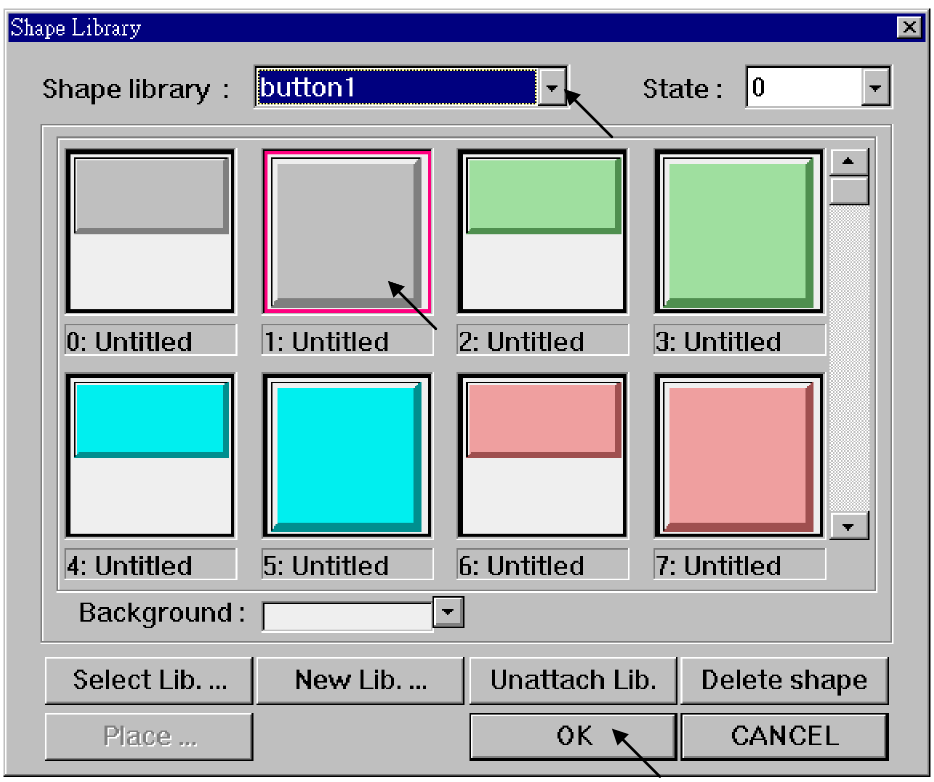
Click on “Function Key” to add a change-window button. Click on “General”, then select “Change Window” and set “Window No.” to 11.



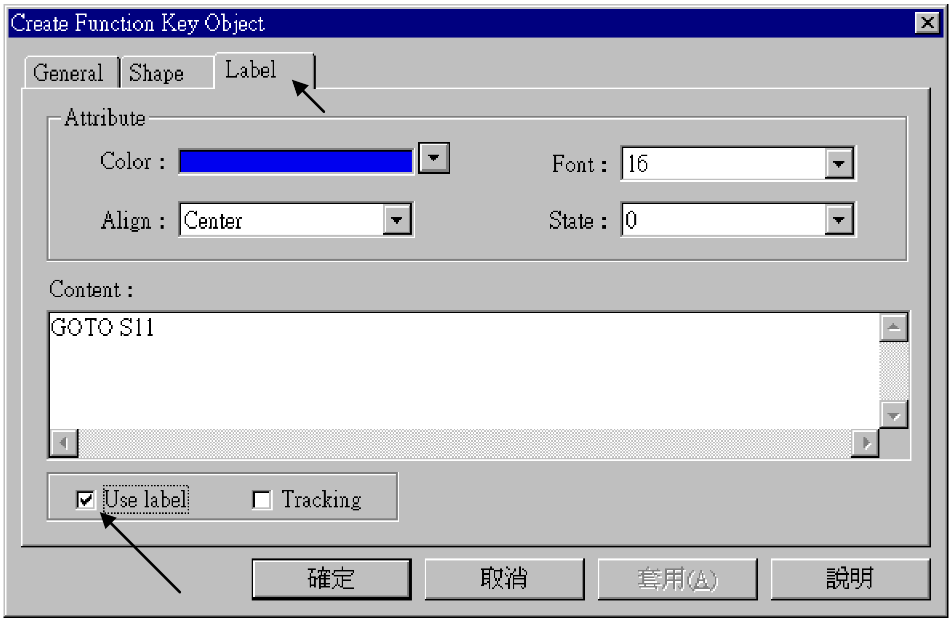
Click on “Shape”, then select “Use shape” and the click on “Shape library …”



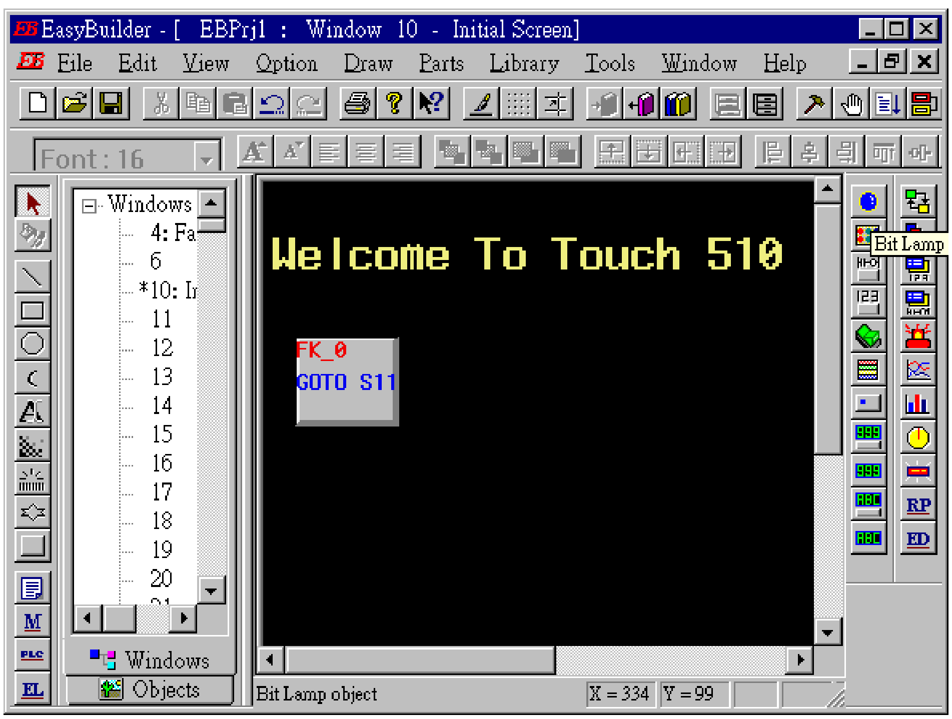
Select the prefered “Shape library” and then select one item and click on “OK”.



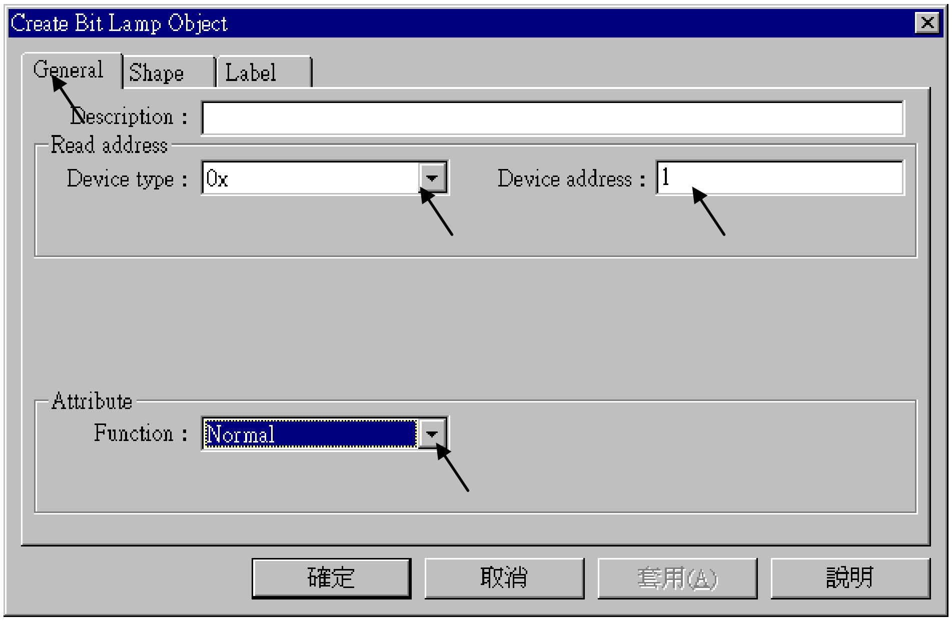
Click on “Label”, then select the prefered “Color”, “Font”, “Align” and set “Content” to “GOTO S11”, and **make sure “Use label” is selected**.



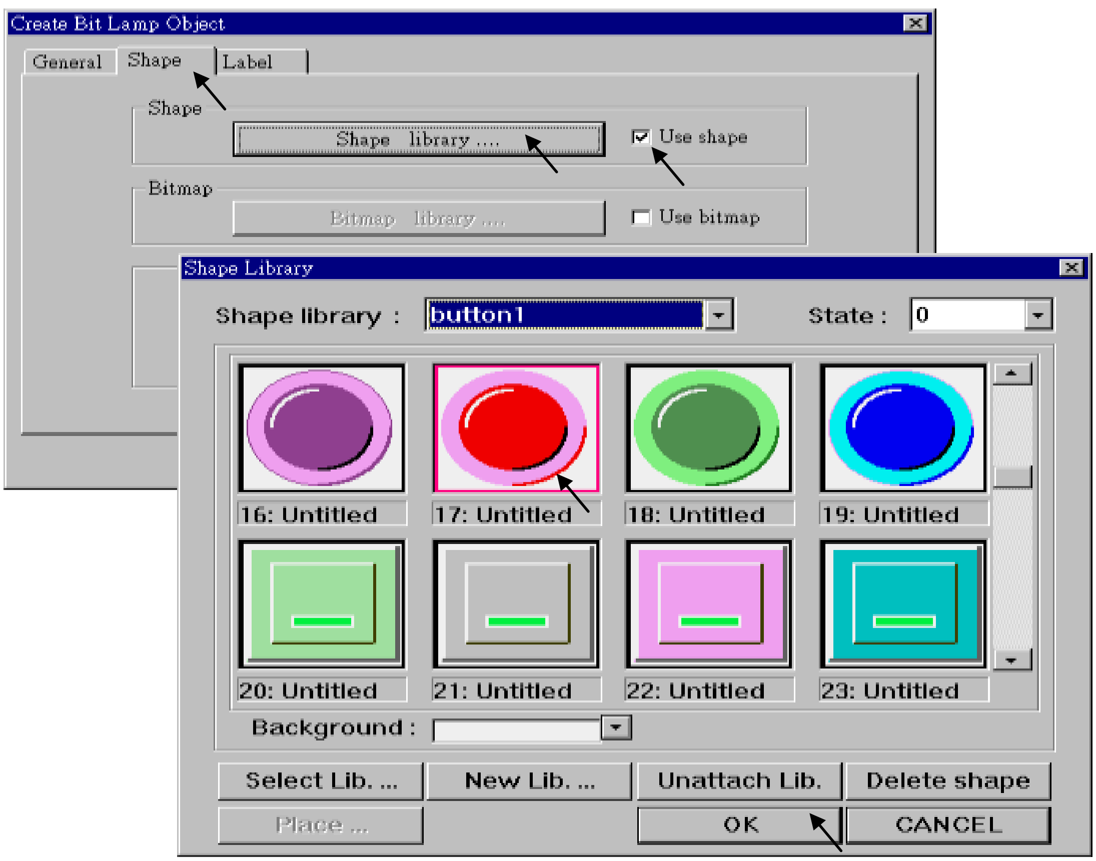
Click on “Bit Lamp”



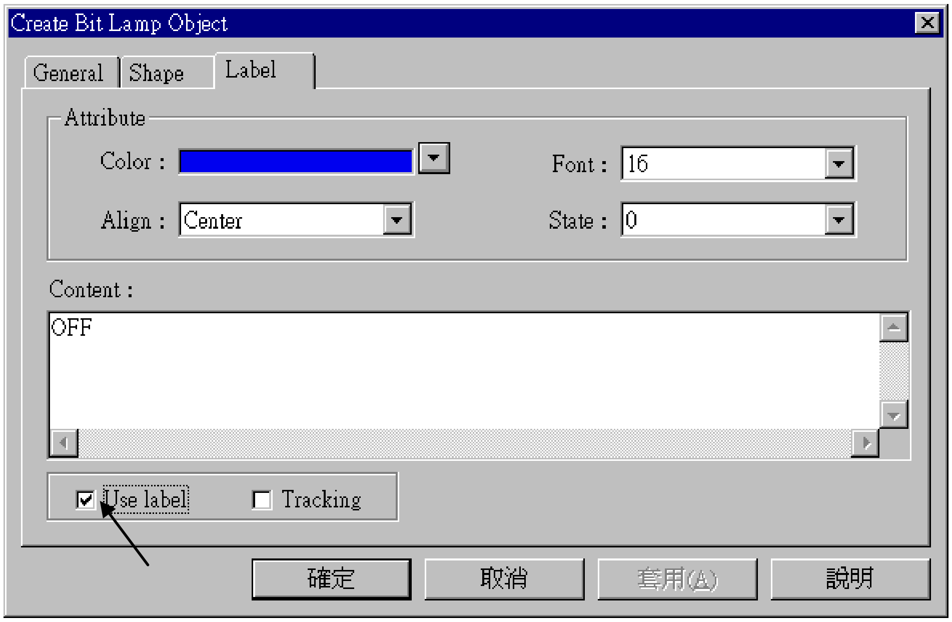
Click on "General", then select "Device type" to "**0x**" (**0x is for boolean variables**), then set "Device address" to 1 (this value is associated with the network address value of the variable in the I-8xx7). And then set "Function" to "Normal".



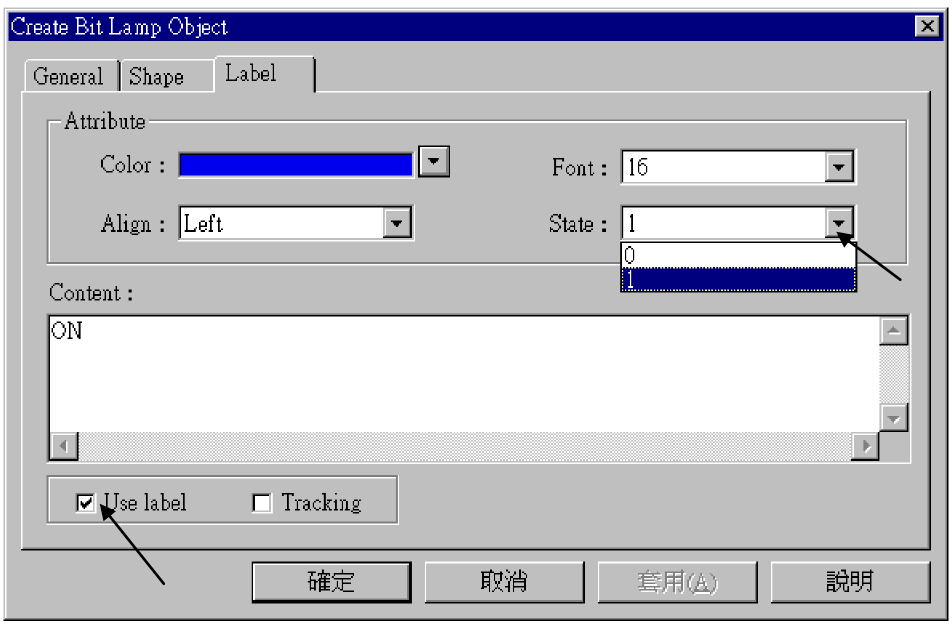
By the same way as former, select prefered "Shap library".



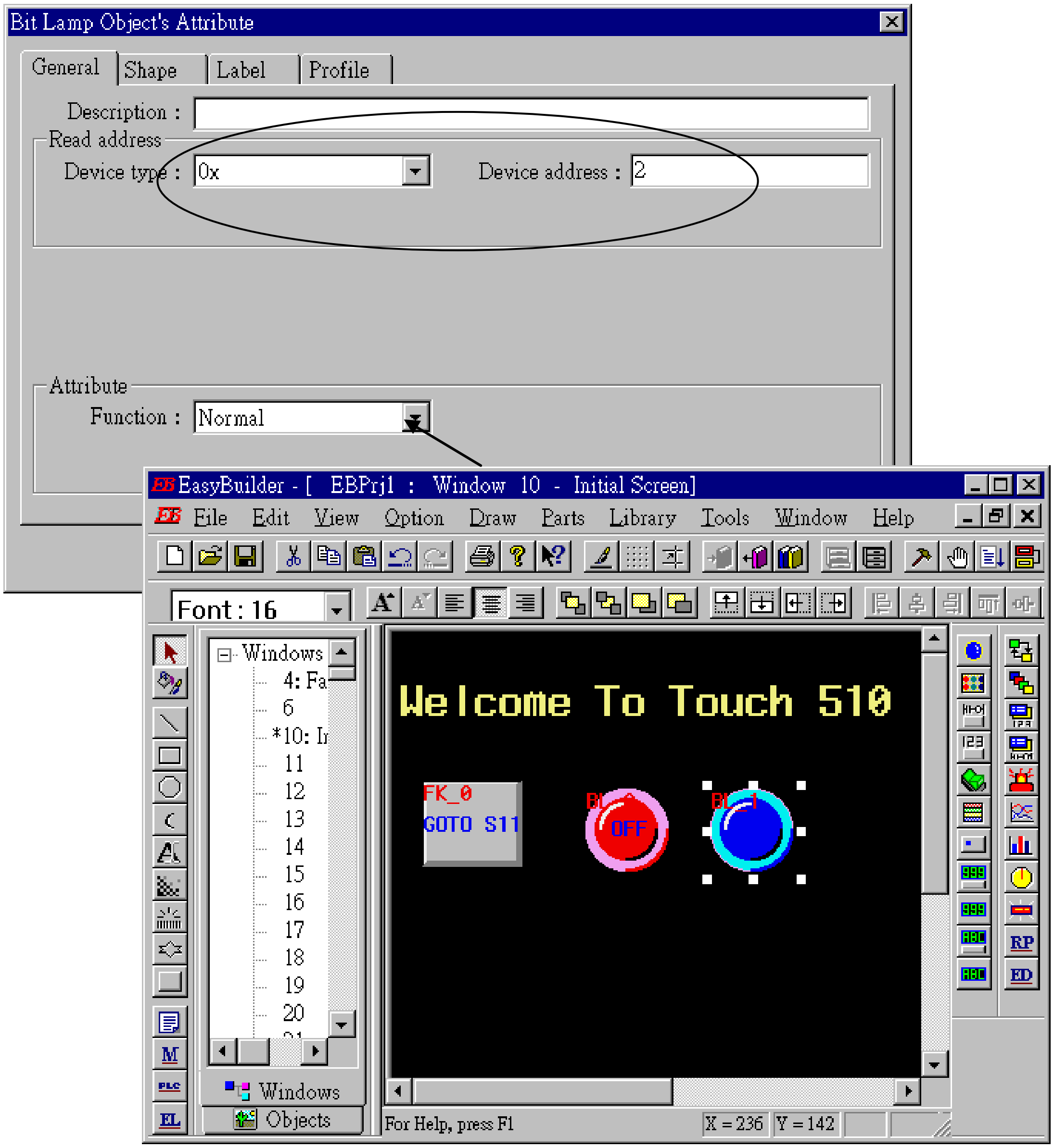
And then select “Label”, given a “OFF” to “Content” for “State : 0”. **Make sure “Use label” is choosed**.



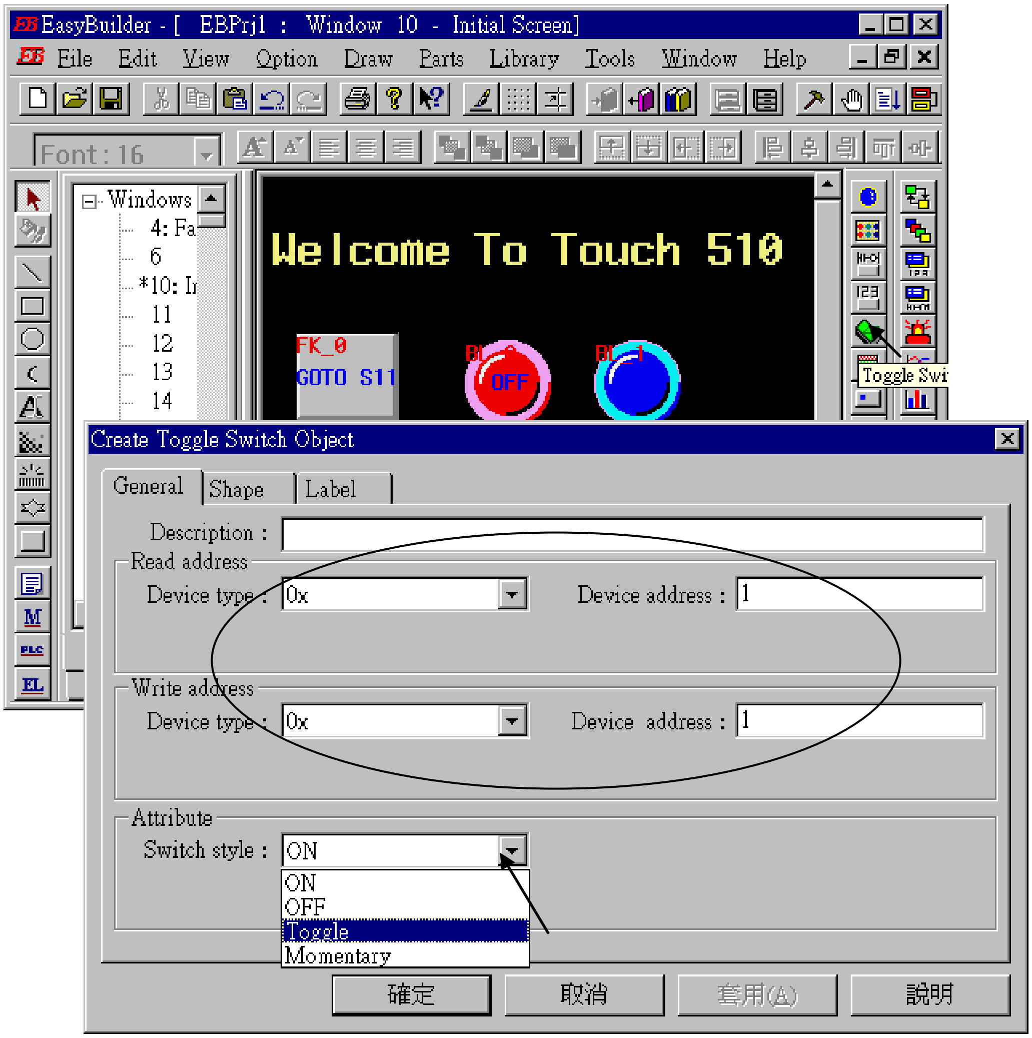
And then change “State” to 1, and given a “ON” to “Content”. **Make sure “Use label” is choosed.**



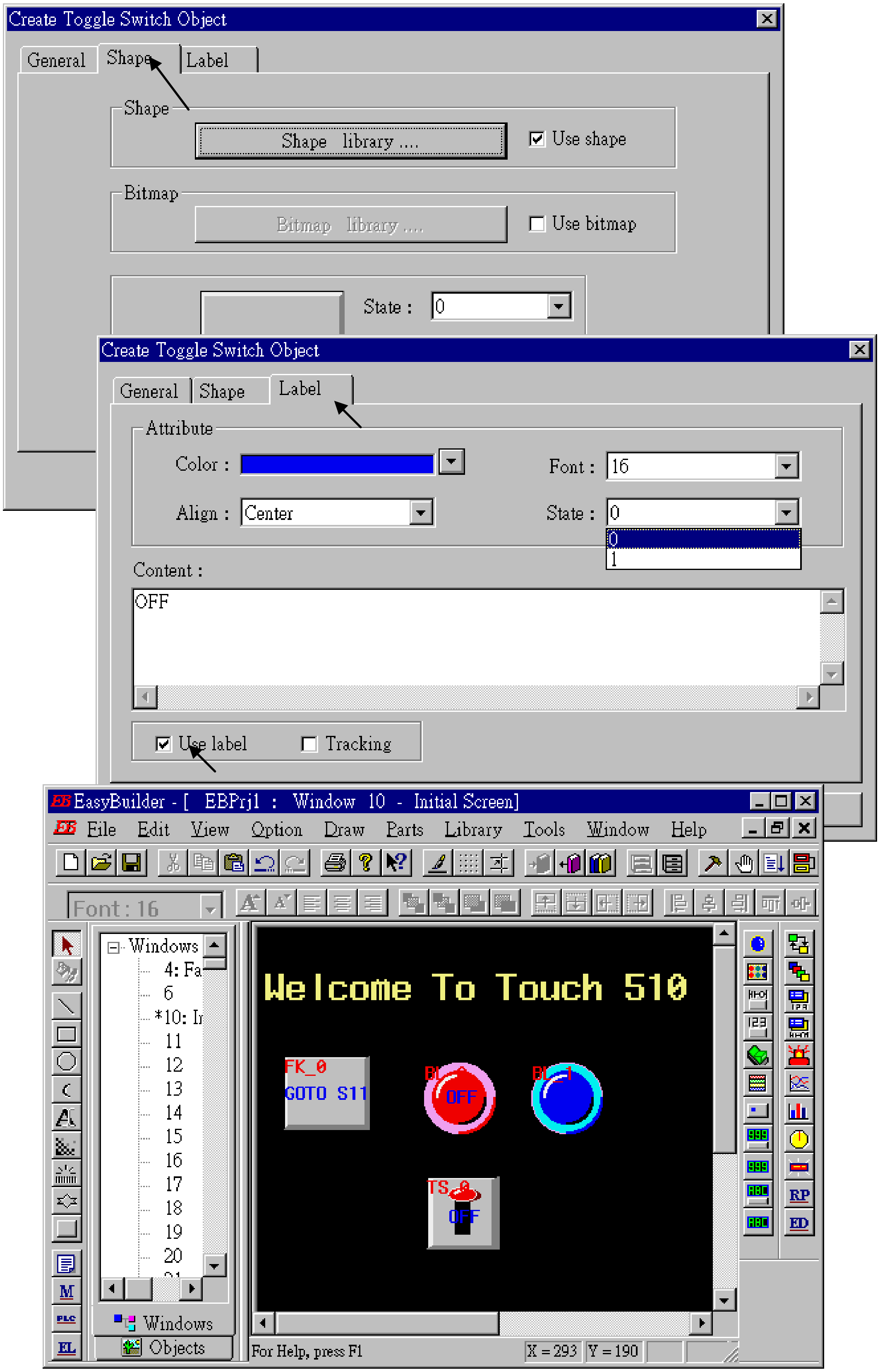
By the same way as former, create one another Bit Lamp with a “Device address” = 2.



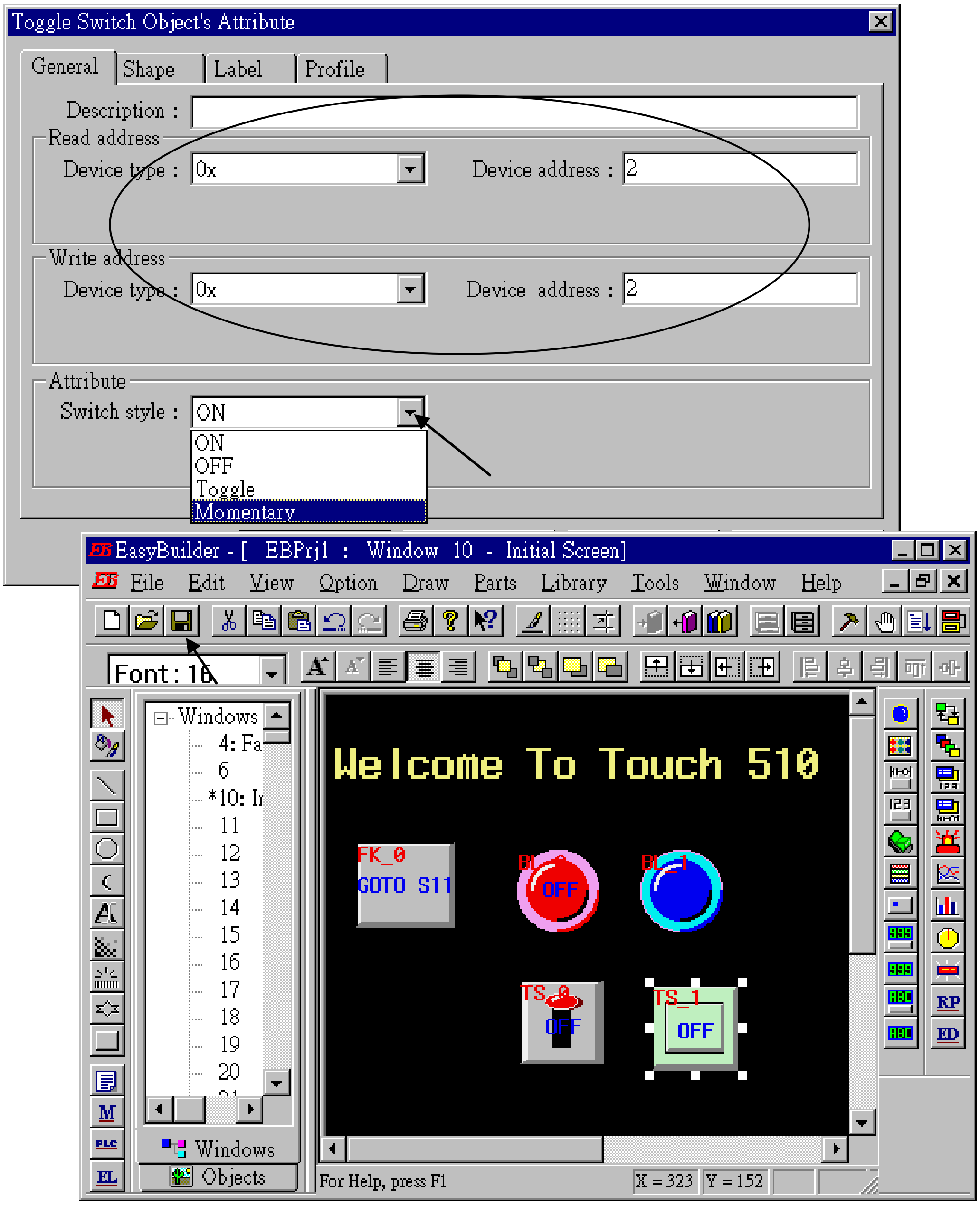
Click on “Toggle Switch”, then set all “Device Type” to “**0x**”, all “Device address” to 1 and select “Switch Type ” to “Toggle”.



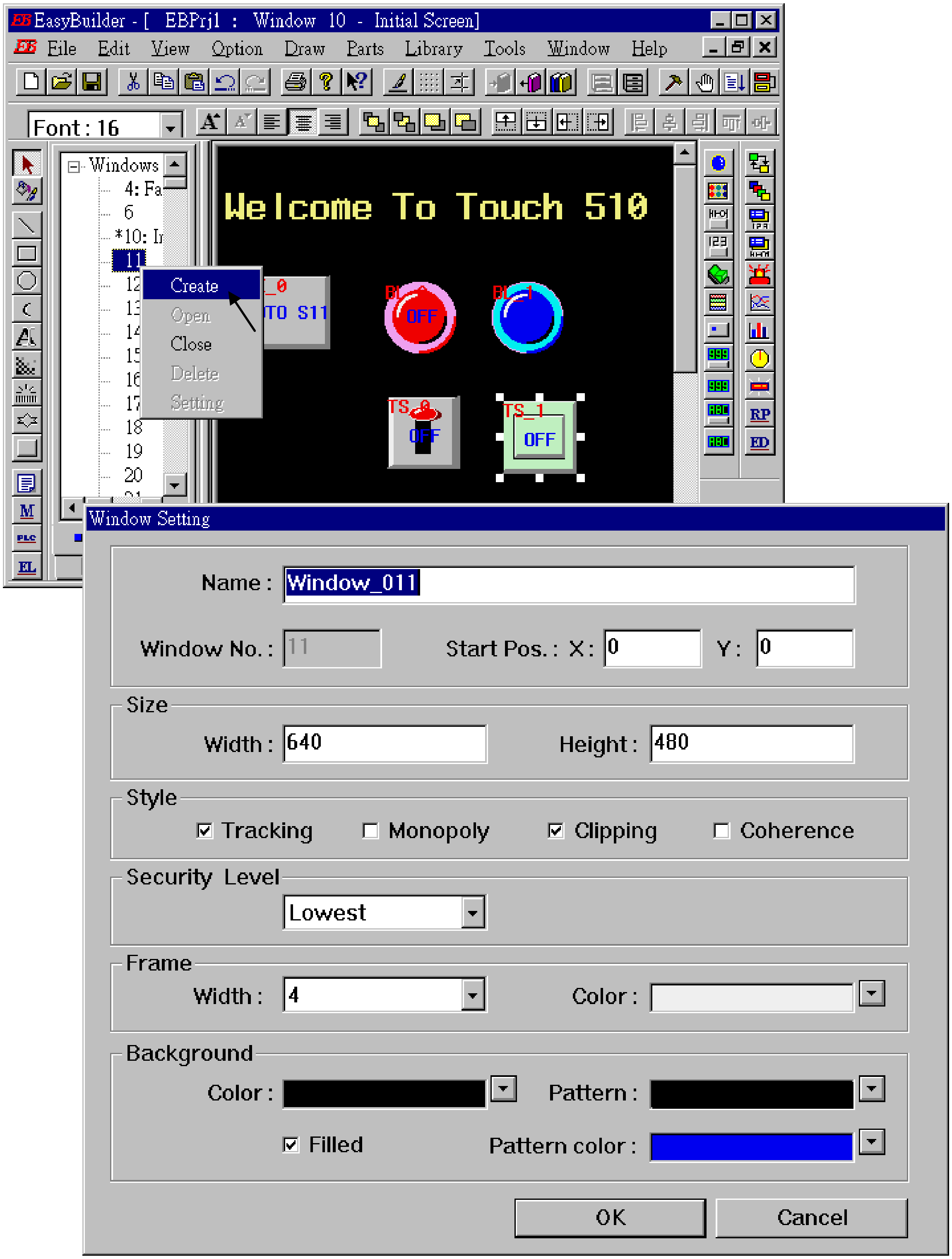
By the same way as former to choose a prefered “shape” and “label”.



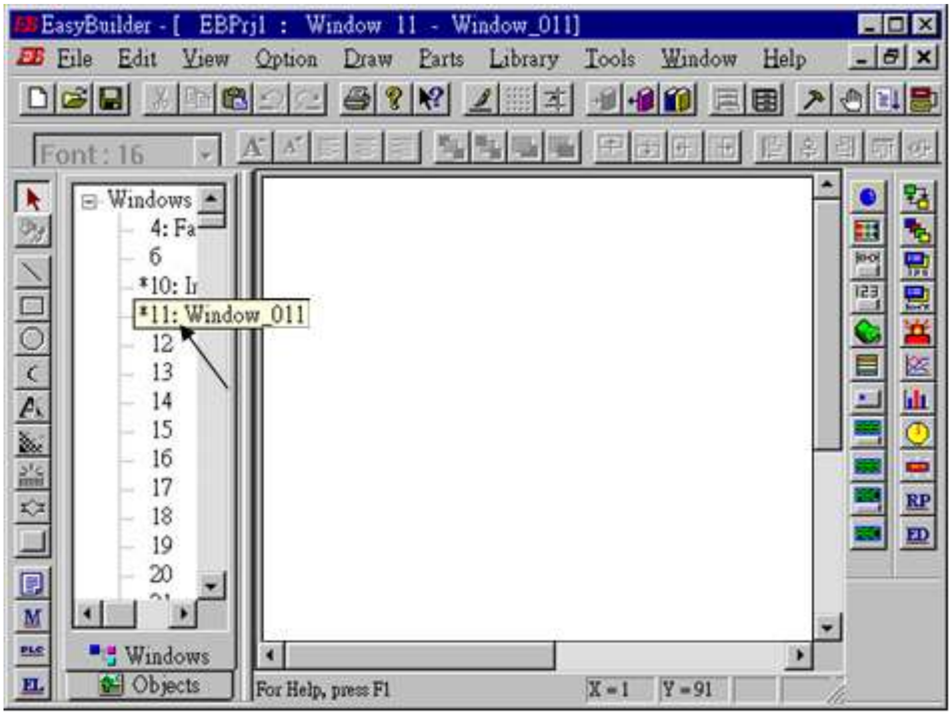
By the same way as former, create one another “Toggle Switch” however set all “Device address” to 2 and “Switch style” to “Momentary”. Click on “save” to save the project.



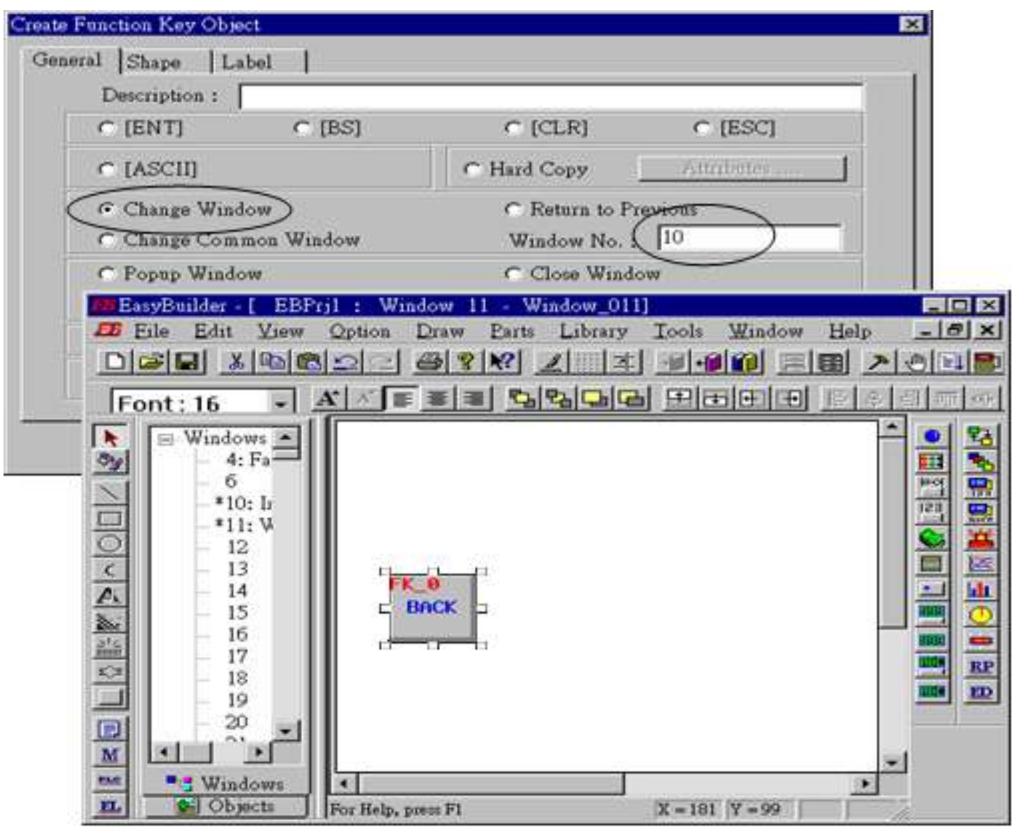
We are going to design another window. Click on “Windows” – “11”, then click the right button of the mouse and select the “Create”.



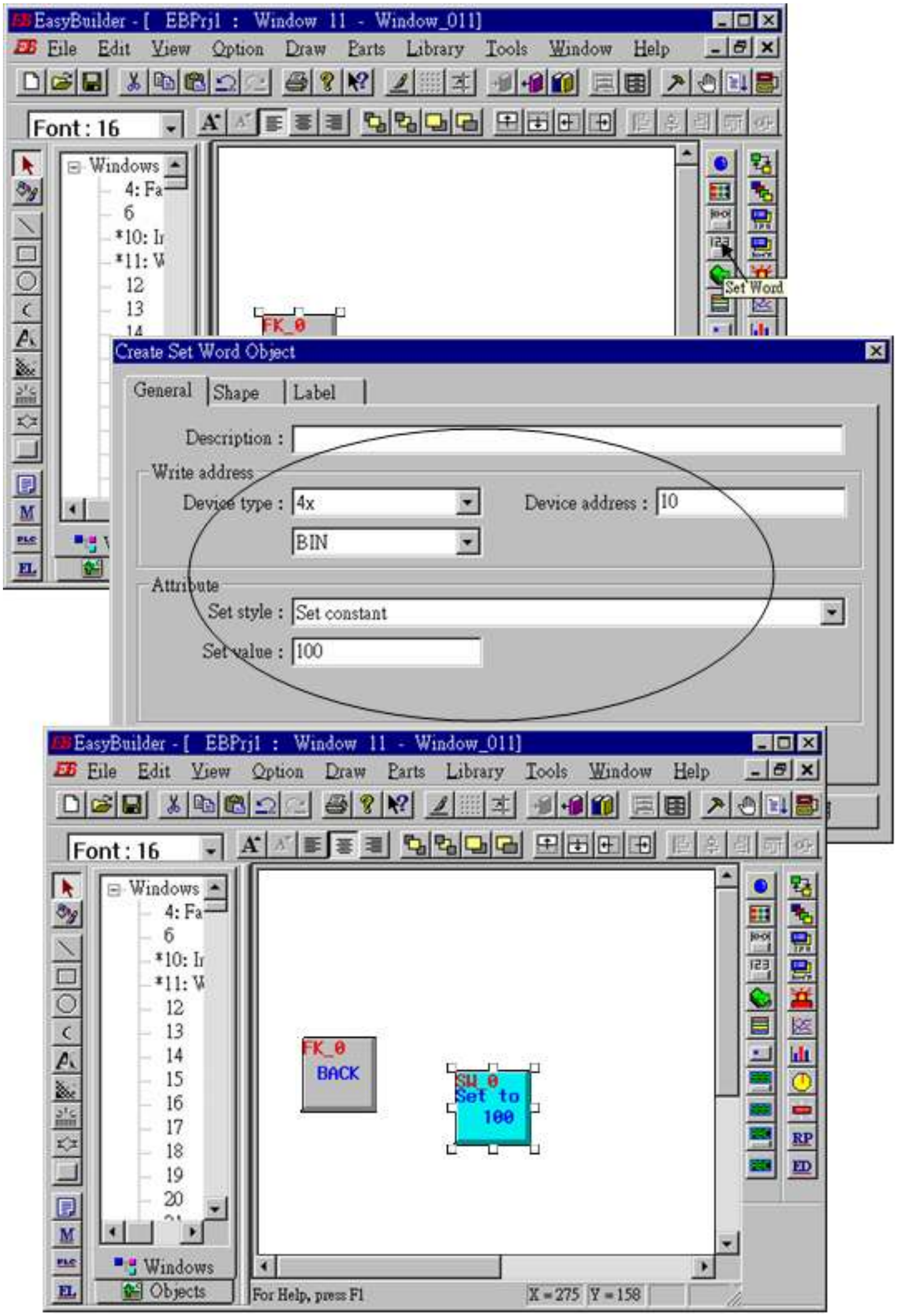
Double click on “Window_011”.



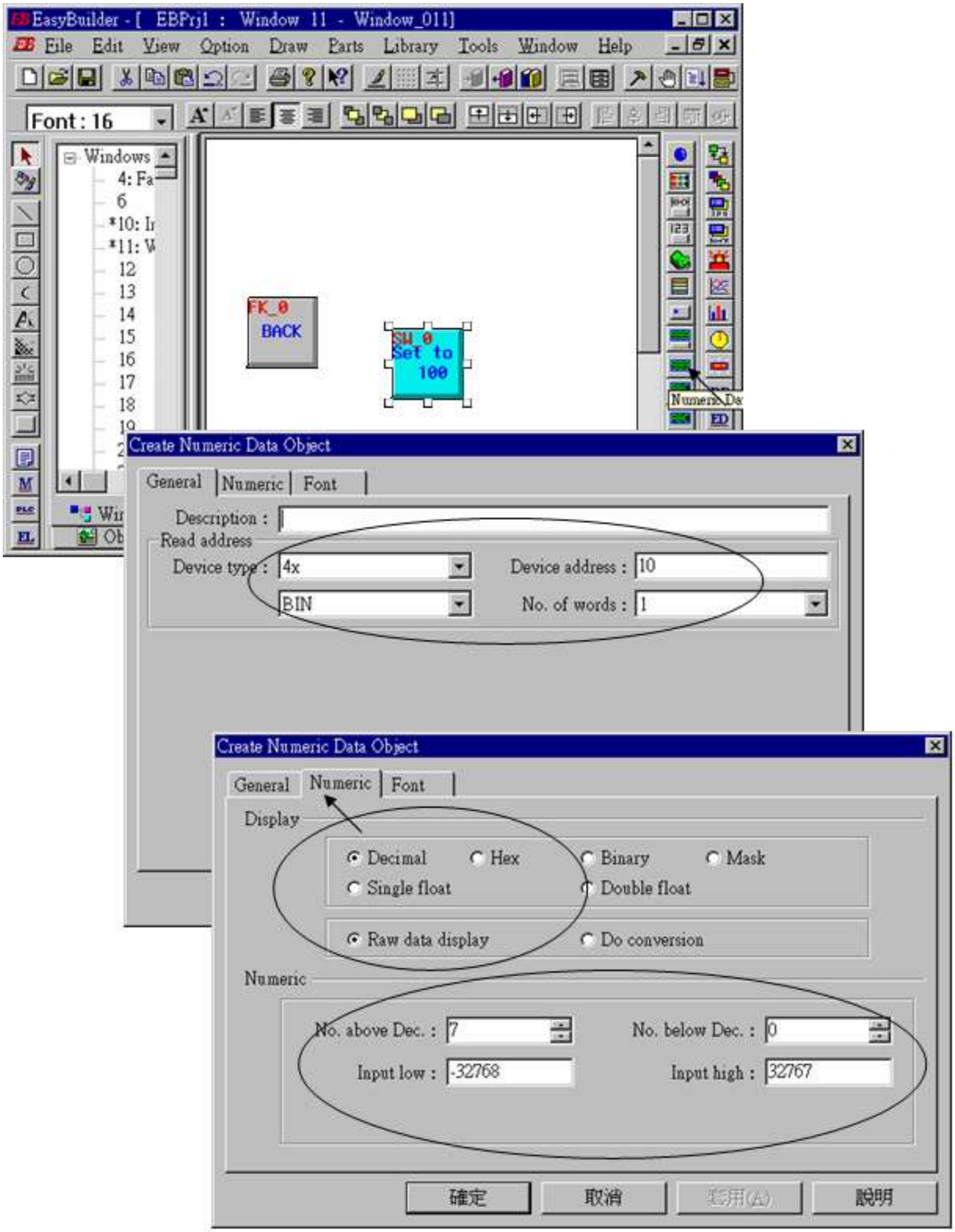
Create a change-window “Function Key” as former method to change to “Window No.” = 10, and Labeled as “BACK”.



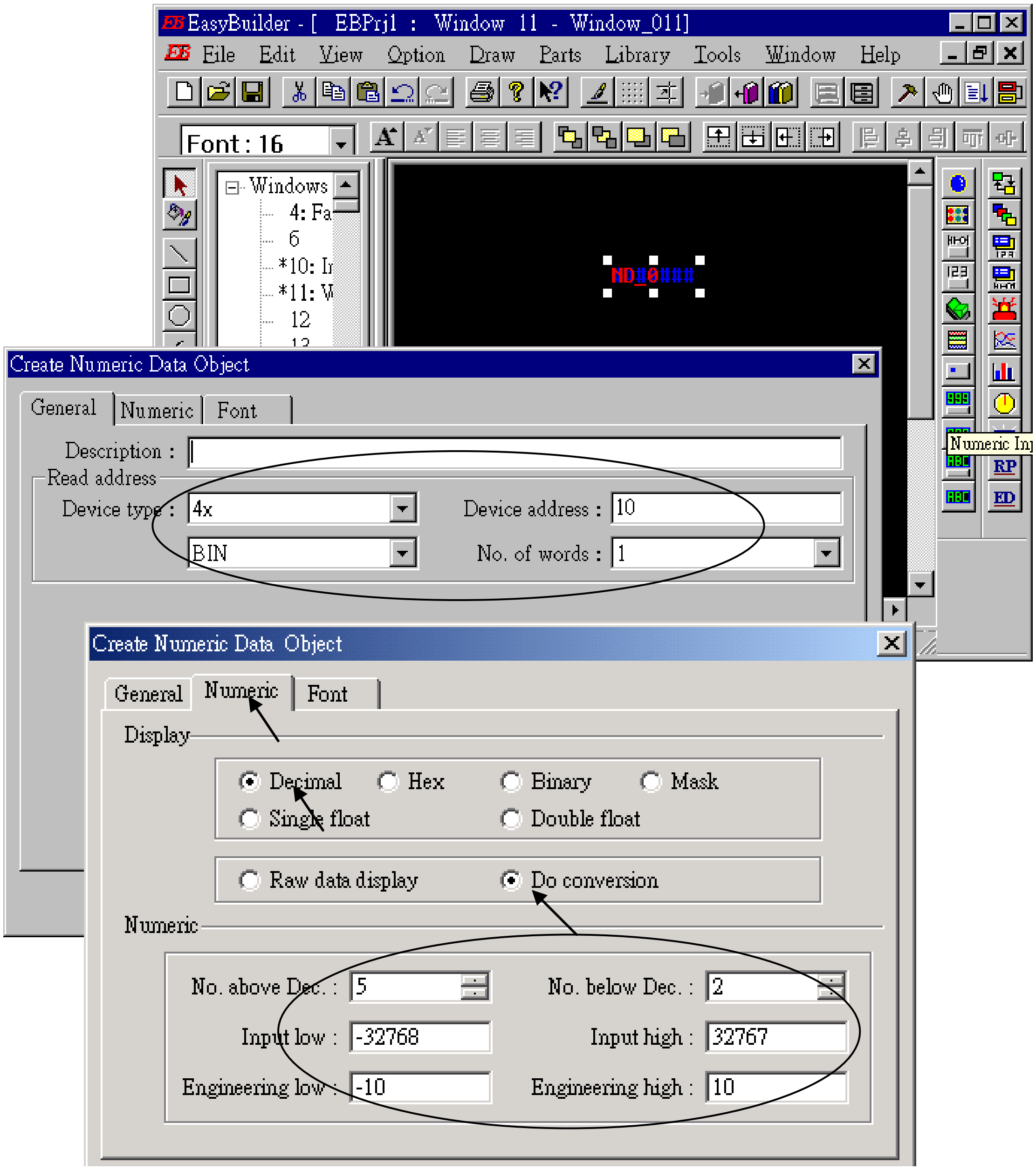
Click on “Set Word”, then set “Device Type” as “**4x**” (**4x is for short integer, 4L is for long integer**), set “Device address” to 10, “BIN”, and “Set style” to “Set Constant”, and “Set value” = 100. And then select the prefered “shape”, and set “label” to “Set to 100”.



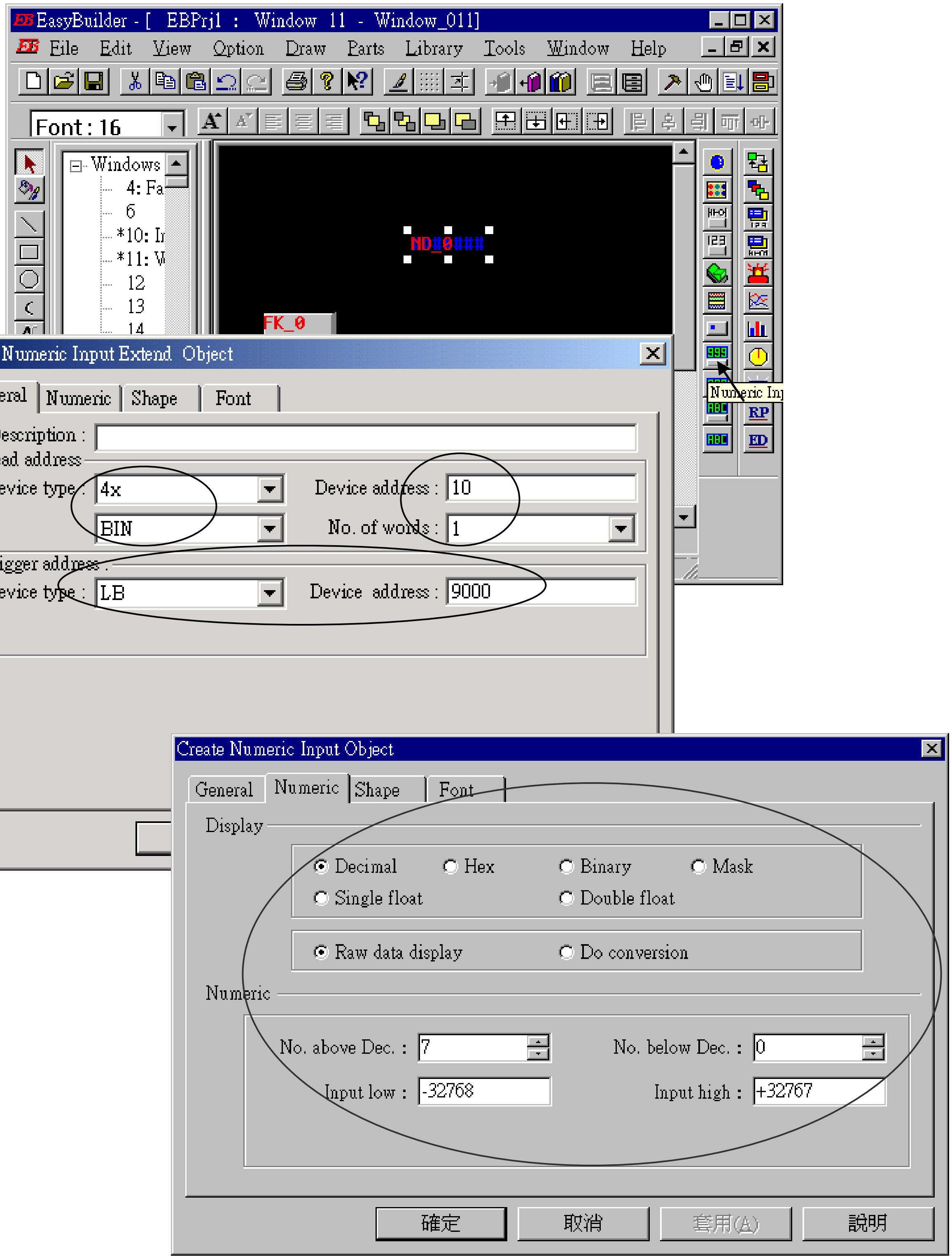
Click on “Numerical Data”, set “Device Type” to “4x” (**4x is for short integer, 4L is for long integer**), “Device address” to 10, “BIN”, “Number of words” to 1, “No. above Dec” to 7, “No. below Decimal” to 0, “Input low” to –32768, “Input high” to +32767. And then select the prefered Font.



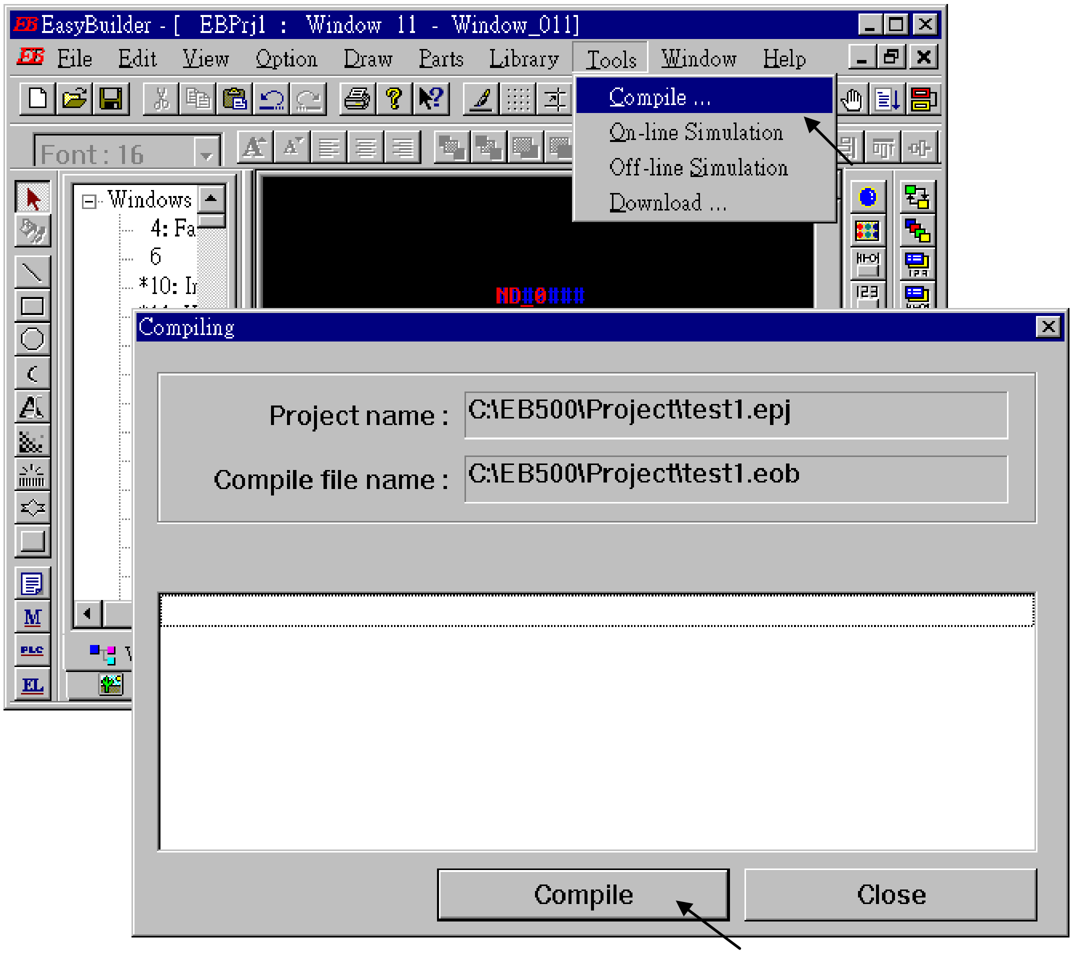
Now we are going to add one another “Numerical Data” with **conversion**.
Click on “Numerical Data”, set “Device Type” to “4x” , “Device address” to 10, “BIN”, “Number of words” to 1, “No. above Dec” to 5, “No. below Decimal” to 0, “Input low” to –32768, “Input high” to +32767, check “Do conversion”, set “engineering low” to –10, “engineering high” to +10 (**Convert [-32768,+32767] to [-10,+10]**). And then select the prefered font.



Click on “Numerical Input”, set “Device Type” to “4x”, “Device address” to 10, “BIN”, “Number of words” to 1, **“Trigger Device Type” to “LB”, “Trigger Device address” to “9000”,** “No. above Dec” to 7, “No. below Decimal” to 0, “Input low” to –32768, “Input high” to +32767. And then select the prefered shape. (Remember to save the project.)

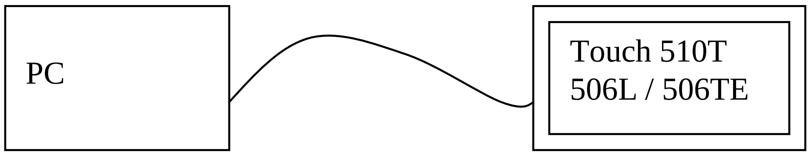


Click “Tools” – “Compile …” to compile this project.

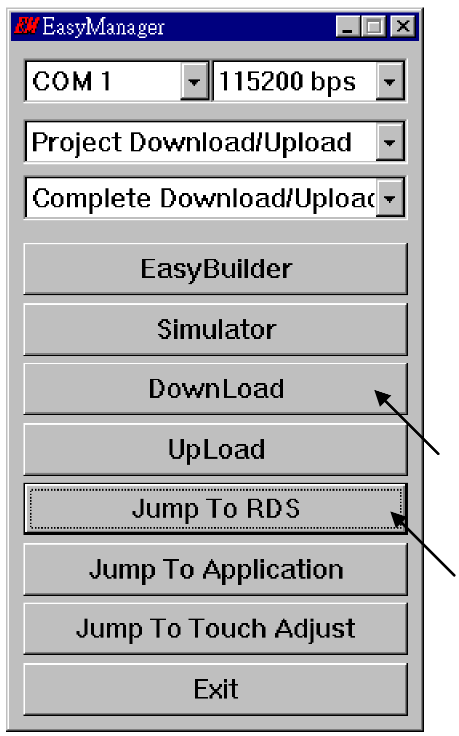


To download the project to the Touch 510, click on the Windows "Start" button, then click on the "Program" button, then click on the "EasyBuilder" – “EasyManager” button. The following window will be displayed. Choose the correct COM No. on your PC (Normally is COM1), “115200 bps”.

Connect the RS-232 download cable (refer to section 4.4) between PC and Touch 510.

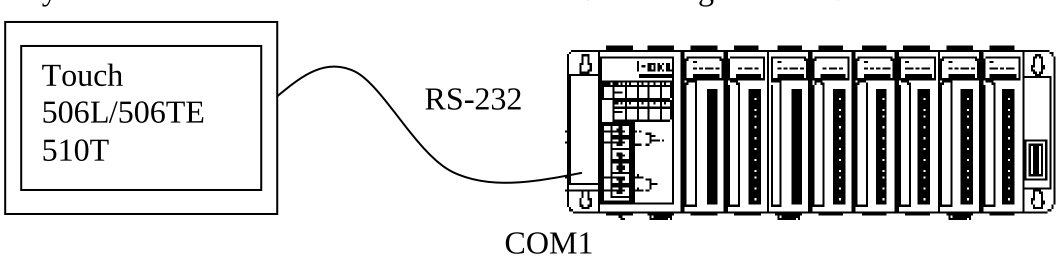


Click on “Jump To RDS” first, if OK., you can see the screen of the Touch 510 will change and wait for project download. Click on “Download” to start to download the MMI picture to the Touch 510.



If downloading is OK, You may click on “Jump To Application” or reset the Touch 510T , and then connect another RS-232 cable between Touch 510 and the I-8xx7 (refer to section 4.4).

Now, you may touch each icon on the Touch 510 to test. Have a good luck !



4.5: Access To Word & Integer Array Via Modbus

User can use the below functions to read/write word & integer arrays inside the ISaGRAF project. For more information about these functions, please refer to Appendix A.4.

ARY_N_R	Read one integer (4 byte, signed) from an integer array
ARY_N_W	Write one integer (4 byte, signed) to an integer array
ARY_W_R	Read one word (2 byte, signed) from an word array
ARY_W_W	Write one word (2 byte, signed) to an word array

Word and integer arrays built in the I-8xx7, I-7188EG/XG, µPAC-7186EG, iP-8xx7 & VP-2117 controller occupy the same memory area, please use them carefully. Other softwares (HMI, OPC server, …) running on the PC can access to these word and integer arrays via **Modbus** protocol. The valid **network address** for these arrays is from **5001 to 8072 for I-8xx7, I-7188EG/XG**, **µPAC-5xx7, µPAC-7186EG, iP-8xx7 & VP-2117** while **10,001 to 19,216 for the WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6, VP-25W7/23W7** and their relation is listed in below table.

For the I-8xx7, I-7188EG/XG, µPAC-7186EG, µPAC-5xx7, iP-8xx7, VP-2117:

Network Address (Decimal)	Word Array	Integer Array
5001	(1,1)	(1,1)
5002	(1,2)	
5003	(1,3)	(1,2)
5004	(1,4)	
…	…	…
…	…	
8071	(12,255)	(6,256)
8072	(12,256)	

For the WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6 and VP-25W7/23W7:

Network Address (Decimal)	Word Array	Integer Array
10001	(1,1)	(1,1)
10002	(1,2)	
10003	(1,3)	(1,2)
10004	(1,4)	
…	…	…
…	…	
19215	(36,255)	(18,256)
19216	(36,256)	

Note:

1. **Network address 1 to 4095 for I-8xx7, I-7188EG/XG**, **µPAC-7186EG, µPAC-5xx7, iP-8xx7, VP-2117, while 1 to 8191 for WP-8xx7, WP-5xx7, XP-8xx7-CE6, XP-8xx7-Atom-CE6, VP-25W7/23W7,** can be defined by users, please refer to Section 4.1.
2. Modbus address in the physical transmission format is equal to Network address minus one (please refer to Chapter 5). So the valid Modbus address for word & integer arrays is from 5000 to 8071 for I-8xx7, iP-8xx7, etc. and 10000 to 19215 for WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6 and VP-25W7/23W7.

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 ISaGRAF controller 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).

5.1: Modbus Protocol Format: RTU Serial

PC software programs and HMI hardware devices can access data from the variables in the ISaGRAF controller system **ONLY** after that variable is assigned a unique network address (please refer to Chapter 4). For more information regarding connecting a PC to an ISaGRAF controller system, please refer to "Getting started Manual" of each controller for details on how to properly connect these devices.

User require programing the Modbus communication program or using commercially available SCADA software to communicate with I-8xx7, I-7188EG/XG, µPAC-7186EG, µPAC-5xx7, iP-8xx7, VP-2117, WP-8xx7 , WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6, VP-25W7 and VP-23W7 controllers and they support the following Modbus functions.

Modbus function	Action
1	Read N bits (booleans)
2	Read N bits (booleans)
3	Read N words (signed short integers)
4	Read N words (signed short integers)
5	Write 1 bit (boolean)
6	Write 1 word (signed short integer)
15	Write N bits (booleans)
16	Write N words (signed short integers)

To read boolean variables, both of function 1 or 3 may be used. If using function 3, values are stored in a word field, variable TRUE means 0xFFFF.

To write boolean variables, both of function 5, 15 could be used. If using function 5, writing bit 0 of byte-vH to 1 will set the Boolean variable to TRUE. For ex, writing vH=1 or 3, or 255 will set Boolean variable to TRUE.

To read analog variables, function 3 should be used.

To write analog variables, both of function 6, 16 could be used.

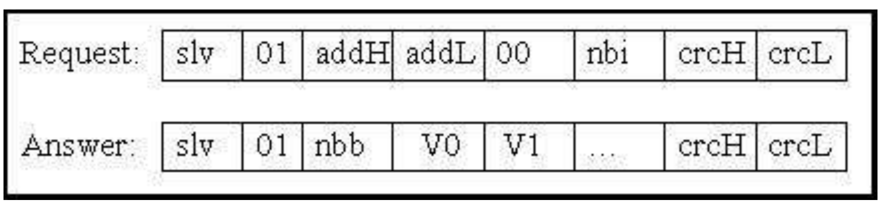
To read long words (signed long integers and float), function 3 should be used. To write long words, function 16 should be used. Please refer to section 4.2 for the definition of network address of long words.

To assist you with the naming conventions used throughout the Modbus protocol-addressing chapter, the following table describes the notations used in this chapter.

Slv	Slave number (Net ID address of the controller)
Nbw	Number of words
Nbb	Number of bytes
Nbi	Number of bits
AddH	**Modbus address**, high byte , 0 ~ 0F
AddL	**Modbus address**, low byte , 0 ~ FE
VH	Word value, high byte
VL	Word Value, low byte
V	Byte value
CrcH	Checksum, high byte , CRC-16
CrcL	Checksum, low byte , CRC-16

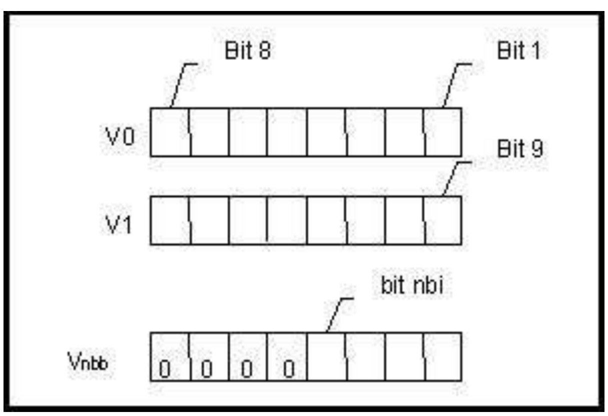
IMPORTANT NOTE
All of the values used in the request and answer frames are **hexadecimal** values.
Modbus address described in this chapter is equal to Network address of the ISaGRAF variable minus one. For ex., Modbus address 0 is associate with ISaGRAF Network address 1. Modbus address FFE (4094) is associate with ISaGRAF Network address FFF (4095).

Function 1: Read "N" Bits
Function 1 reads "n" number of bits (nbi) in Boolean starting from Modbus address addH/addL.



Request:	slv	01	addH	addL	00	nbi	crcH	crcL
Answer:	slv	01	nbb	V0	V1	…	crcH	crcL

V0, V1 … are the bit fields of number of bytes (nbb) using the following format.



Bit 1 corresponds to the Boolean value of the variables with the Modbus address addH/addL. Bit nbi corresponds to the Boolean value of the variable with the Modbus address addH/addL + nbi – 1. If the value of the Boolean variable is "True", then the corresponding bit will be set to a "1". If the value is "False", the corresponding bit will be set to a "0".

Function 2: Read N Bits
Function 2 has the same exact same format as function 1.

Function 3: Read N Words
Function 3 reads the number of words (nbw), in signed 16-bit integer format, starting from the Modbus address addH/addL.

Request:	slv	03	addH	addL	00	nbw	crcH	crcL
Answer:	slv	03	nbb	vH	vL	...	crcH	crcL

The number of bytes (nbb) is the total number of bytes from word value high byte (vH) to word value low byte (vL) inclusive.

IMPORTANT NOTE About Function 3
Integer values can be read by function 3. A word in the modbus protocol is a 16-bit value (signed short integer), and an ISaGRAF integer variable is a 32-bit value, so only the lower 16 bits of the integer variable are returned. If users would like to read a 32-bit integer (signed long integer) of I-8xx7 controller, the proper network address of the variable should be set as described in section 4.2.

Function 4: Read N Words
Function 4 has the same exact format as function 3.

Function 5: Write 1 Bit
Function 5 writes one (1) bit to the Boolean variable with the Modbus address addH/addL.

Request:	slv	05	addH	addL	V	0	crcH	crcL
Answer:	slv	05	addH	addL	V	0	crcH	crcL

Writing a 0xFF value to the byte value (V) will set the Boolean variable to "True". Writing a zero to the byte value (V) is set the Boolean variable to "False".

Function 6: Write 1 Word
Function 6 writes one (1) word (16 bits) to the integer variable with the Modbus address addH/addL.

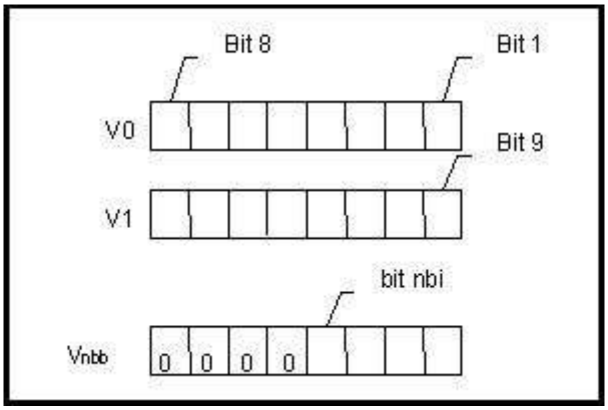
Request:	slv	06	addH	addL	vH	vL	crcH	crcL
Answer:	slv	06	addH	addL	vH	vL	crcH	crcL

Function 15: Write N Bits

Function 15 writes a number of bits (nbi) to the Boolean variables starting from the Modbus address addH/addL to addH/addL + nbi – 1. The total number of bytes (nbb) is the total amount of bytes occupied by nbi bits, that means nbb = (nbi+7)/8. For ex. nbi=1~8, nbb=1; nbi=9~16, nbb=2.

Request:	slv	0F	addH	addL	00	nbi	nbb	V0	V1	…	crcH	crcL
Answer:	slv	0F	addH	addL	00	nbi	crcH	crcL				

V0, V1 … are the bit fields of number of bytes (nbb) using the following format.



Bit 1 corresponds to the Boolean value of the variables with the Modbus address addH/addL. Bit nbi corresponds to the Boolean value of the variable with the Modbus address addH/addL + nbi – 1. Writing a 1 to a bit will set the value of the corresponding Boolean variable to "True", and writing a 0 to a bit will set the corresponding Boolean variable to "False".

Function 16: Write N Words

Function 16 writes a number of words (nbw) to the integer variables starting from the Modbus address addH/AddL to addH/addL + nbw – 1. The number of bytes (nbb) is the total amount of bytes occupied by number of words (nbw), that is nbb = 2 * nbw.

Request:	slv	10	addH	addL	00	nbw	nbb	vH	vL	…	crcH	crcL
Answer:	slv	10	addH	addL	00	nbw	crcH	crcL				

Examples Of Modbus Function Formats

Function 1: Read 15 bits starting from **Modbus address 0**x1020. The NET ID address is 1.

Request:	01	01	10	20	00	0F	79	04
Answer:	01	01	02	00	12	39	F1	

In this example function 1 returns 2 bytes, the value is 0x0012. This means variables with a **network address** of 0x102A and 0x102D are "True" (**Modbus address** is 0x1029 and 0x102C), the rest of the variables are set to "False".

Function 5: Write 1 bit to the Boolean variable with the **Modbus address** 0x0006. The NET ID address is 1. The value to write to is 0xFF.

Request:	01	05	00	06	FF	00	6C	3B
Answer:	01	05	00	06	FF	00	6C	3B

In this example of function 5 the Boolean variable is set to "True".

Function 16: Write 2 words (4 bytes) to the integer variables with the **Modbus address** starting from 0x2100. The first word value to write to is 0x1234. The second word value to write to is 0x5678. The NET ID address is 1.

Request:	01	10	21	00	00	02	04	12	34	56	78	1C	CA
Answer:	01	10	21	00	00	02	4B	F4					

5.2: Modbus Protocol Format: TCP/IP

The Ethernet port of I-8437-80, I-8837-80, I-7188EG, µPAC-7186EG, µPAC-5xx7, iP-8x47, VP-2117, WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6 controller systems supports the Modbus TCP slave communications protocol.

ALL requests are sent via TCP on port number **502**.

The Modbus TCP/IP protocol adds 6 extra bytes before the Modbus RTU serial protocol, and these 6 extra bytes and the Modbus RTU serial protocol are all packed inside the TCP/IP protocol.

TCP/IP	Extra 6 Bytes	Modbus RTU serial	TCP/IP

The request and responses are prefixed by the six bytes as follows:

Byte 0: transaction identifier - copied by server
Byte 1: transaction identifier - copied by server
Byte 2: protocol identifier = 0
Byte 3: protocol identifier = 0
Byte 4: length field (upper byte) = 0 (since all messages are smaller than 256)
Byte 5: length field (lower byte) = number of following bytes

The rest of the Modbus TCP/IP protocol is the same as the Modbus RTU Serial protocol after byte No. of 6 except that the CRC-16 is not need for the Modbus TCP/IP protocol.

Example TCP/IP Transactions

The first example of a TCP/IP transaction is reading one (1) word at Modbus address 4 from slave number 9 (NET-ID) returning a value of 8; the transaction would be as follows:

Request:	01	02	00	00	00	06	09	03	00	04	00	01
Response:	01	02	00	00	00	05	09	03	02	00	08	

The second example of a TCP/IP transaction is reading 8 bits starting from Modbus address 2 from slave number 7 (NET-ID), returning a value of 0x49 (bit field: 01001001) would be as follows:

Request:	03	29	00	00	00	06	07	01	00	02	00	08
Response:	03	29	00	00	00	04	07	01	01	49		

5.3: Algorithm For CRC-16 Check

The following C language algorithm is for Modbus RTU Serial ONLY!! This CRC (Cyclic Redundancy Check) program provides a checksum that can be used to validate information being passed through Modbus RTU Serial protocol.

This CRC-16 check program first calls "crc_init()" one time at the beginning of the communication to initialize the checksum table. Then you can call "crc_make()" to calculate a checksum whenever you want to.

```
#define POLY_CRC16 0xA001
static BYTE TABLE1[256];
static BYTE TABLE2[256];

void crc_init(void)   /* set crc table */
{
  WORD mask,bit,crc,mem;
  for(mask=0;mask<0x100;mask++)
  {
    crc=mask;
    for(bit=0;bit<8;bit++)
    {
      mem=crc & 0x0001;
      crc/=2;
      if(mem!=0) crc ^= POLY_CRC16;
    }
    TABLE2[mask]=crc & 0xff;
    TABLE1[mask]=crc >> 8;
  }
}

void crc_make(WORD size, BYTE *buff, BYTE *hi, BYTE *lo)   /* calculate crc */
{
  BYTE car,i;
  BYTE crc[2];
  crc[0]=0xff;
  crc[1]=0xff;
  for(i=0;i<size;i++)
  {
    car = buff[i];
    car ^= crc[0];
    crc[0]=crc[1] ^ TABLE2[car];
    crc[1]=TABLE1[car];
  }
  *hi=crc[0];
  *lo=crc[1];
}
```

Chapter 6. Linking I-7000 & I-87K Remote I/O Modules

Note:
1. The I-87017R and I-87017RC is better than I-87017 and I-87017C in industrial application.
2. The I-87018Z is better than I-87018 in industrial application. (I-87018Z has 10-channels. The precision is better than I-87018, I-87018R and I-87019R. And each channel can configure to be different Input type. For example, using Ch.1 to 4 to measure 4 to 20 mA, using Ch.5 to 8 as Thermo-Couple K-Type, using Ch.9 to measure +/- 2.5 V, and using Ch.10 as Thermo-Couple R-Type.)
3. The I-7018Z is better than I-7018. (The reason is the same as I-87018Z)
   I-7018z: http://www.icpdas.com/products/Remote_IO/i-7000/i-7018z.htm
   I-87018z: http://www.icpdas.com/products/Remote_IO/i-87k/i-87018z.htm

For more description about using I-7018Z, please refer to Chapter 11.3.9.

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

Note:

A. If the I-7000 and I-87xxxW I/O module's type is Analog Input, please configure the format as "2's complement". Like these AI modules: I-7005, I-7013, I-7015, I-7016, I-7017, I-7017R, I-7018, I-7018R, I-7019, I-7019R, I-7033, I-87005W, I-87013W, I-87015W, I-87015PW, I-87016W, I-87017W, I-87017RCW, I-87017ZW, I-87017DW, I-87018W, I-87018RW, I-87018ZW, I-87019RW and I-87019ZW, etc.

B. If the I-7000 and I-87xxxW I/O module's type is Analog Output, please configure the format as "Engineer Unit". Like these AO modules: I-7021, I-7022, I-7024, I-87022W, I-87024W and I-87026W.

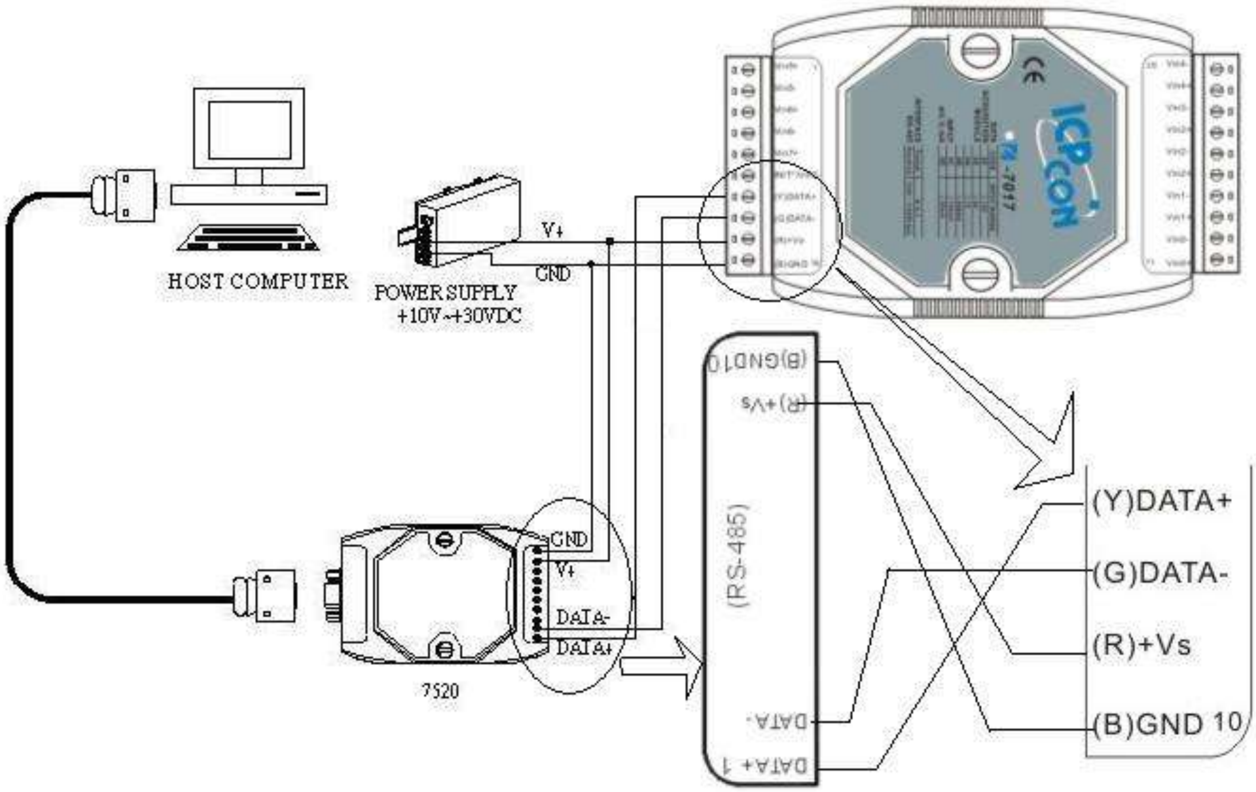
Before connecting the I-7000 and I-87K remote I/O modules to the controller system, it needs to set up the NET-ID (Must be unique ID) for each I/O modules and the same baud rate with the controller by using the "DCON Utility". "DCON Utility" is a useful software tool used to network search, configure or test the I/O modules. Please visit the website to get "DCON Utility" software and its user manual.
ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/driver/dcon_utility/

Notes: 1. Make sure the hardware connection is correct.
2. Search and configure the modules one by one.
3. Connect the module's INIT* to GND (or switch the junper to INIT) and Power on the module.

Very Important: Please wire an terminal resistor around 110 to 330 ohms (you can try 125 ohms first, then try others) at ISaGRAF controller's RS-485 port , between the D+ and the D- pin.This will ensure the host watchdog of I-7000 and I-87K output modules to work correctly. (For example, if you don't wire any terminal resistor and enable the host watchdog function at "bus7000b" (Section 6.2 , the "host_watchdog" parameter set as 1) , when you just unplug the I-7000's "DATA+" pin (keep "Data-" pin connected with the controller) , you will see the watchdog doesn't work in this I-7000. If you wire a resistor about 125 ohms between the controller's RS-485 D+ and D- pin, if you unplug any one of I-7000's "Data+" or "Data-" pin, the watchdog will work correctly.

Step 1: Hardware connection:

Please connect the PC and I-7000 module to configure it.



Note: For linking I-7000 and I-87xxx I/O module, you have to prepare an I-7520R (or I-7520) RS-232/RS-485 converter (or USB to RS-485 converter, such as tM-7561, I-7561). The ICP DAS I-7520, I-7520R, tM-7561, I-7561 high functionality converter with a Self-turner which can auto change the baud rate.http://www.icpdas.com/products/Industrial/communication_module/communication_list.htm. For other brands, the RS-232/RS-485 converter will use one specific baud rate. (Without a Self-turner, every time the baud rate changed, you must manually configure it and it's really inconvenient.)

Step 2: Set I/O module to initial state

If the module is a new one, factory have set a default settings for user's convenient. If you don't know the configuration of the module, please set the I/O module to initial state.

*** To set I-7000 module to initial state is to wire connect the INIT* to GND and Power on the module. Then the module will become initial state. (**Some new designed I-7000 modules have a Dip-switch at its back. Please switch it to the "INIT" position, then power up the module**)

*** Most of I-87xxxW module has an internal Jumper; you can set it as "Normal" or "INIT" state.

*** For old designed I-87K module's initial state is set by I-87K4/5/8/9's dip switch. For example, setting dip-2 to "ON", and then re-cycle the power, it means the second slot is in initial state. (If using **I-87K5** and **I-87K9**, please do not plug I-87xxx board in its left-most slot for initial configuration. Please plug at 2nd to 9th slot for initial configuration. The dip-1 is for 2nd slot of **I-87K5** and **I-87K9**, …, dip-4 is for 5th slot, Dip-8 is for 9th slot of I-87K9).

Some new designed I-87K High Profile I/O modules, like I-87019w, have Jumper built-in. Their "INIT / Normal" state is controlled by its own Jimper not by the dip-switch of I-87K4/5/8/9. After completed the setting, please remember to set it as "Normal" state.

<u>The default state from factory:</u>

I/O Module	I-7000	M-7000	87K series
Address	1	1	1
Baud rate	9600	9600	115200
Checksum	Disabled	Not defined	Disabled
Protocol	DCON Protocol	Modbus Protocol	DCON Protocol

<u>The initial state after initiation:</u>

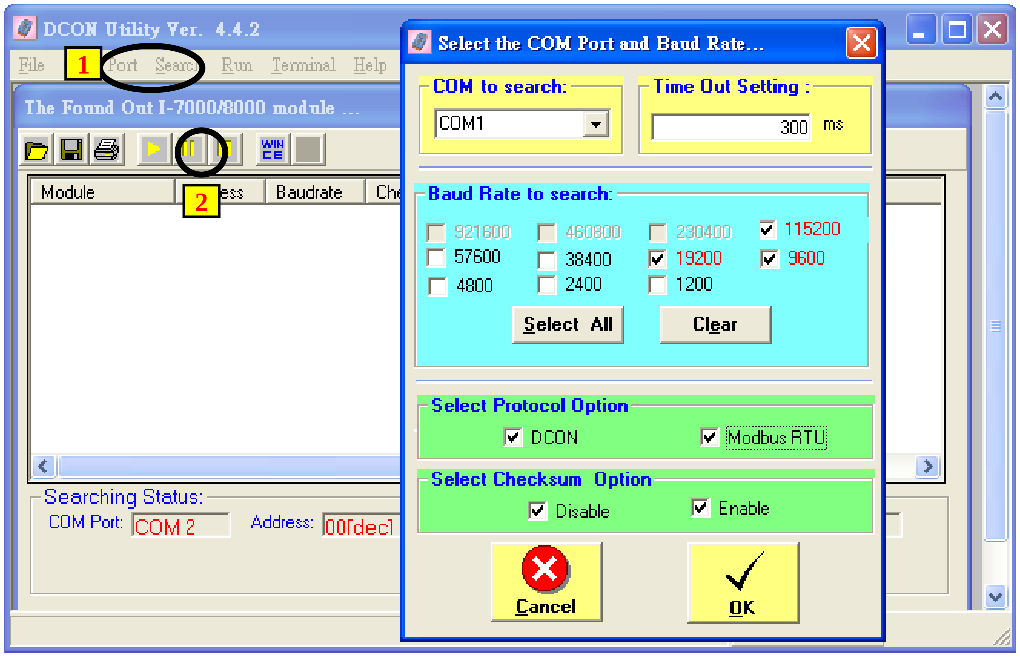
I/O Module	7000 series (I-7000 and M-7000)	87K series
Address	0	0
Baud rate	9600	115200
Checksum	Disabled	Disabled
Protocol	DCON Protocol	DCON Protocol

<u>Step 3: Select COM port and baud rate to search</u>

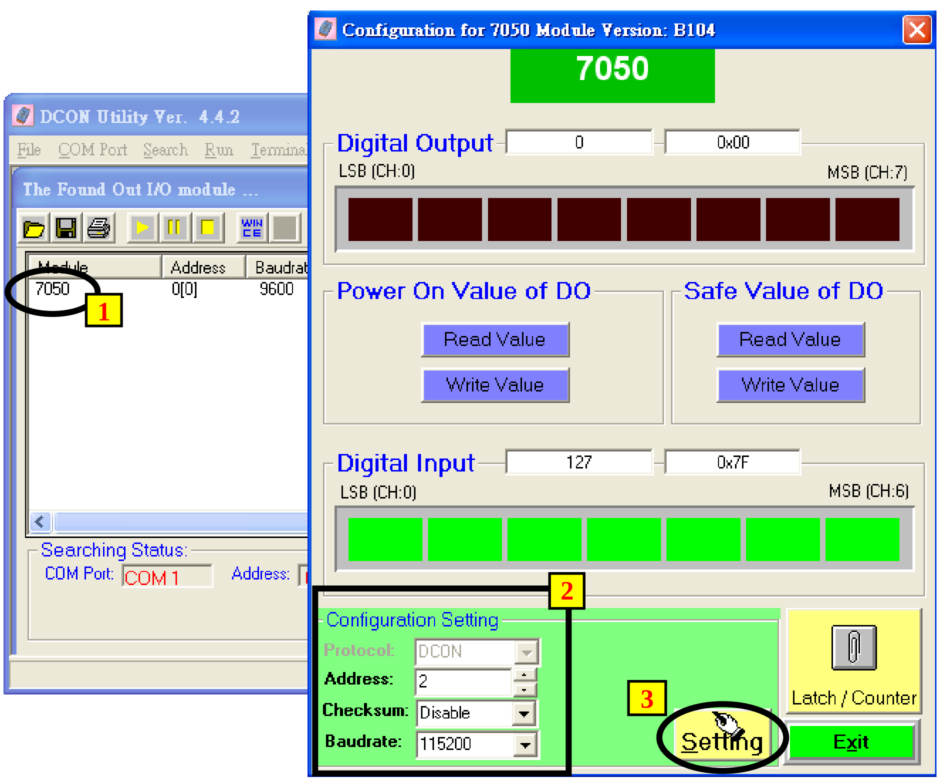
Execute the DCON Utility from "Start/programs/DAQPro/DCON Utility/".

1. Click "COM Port" menu to select the COM port and baud rate to search. You can select multi-baud rate, protocol or checksum conditions if you do not know the module's setting, but it will spend more time to scan the network. After selection, click "OK".

2. Click “Start Search” icon to begin search module. Click when it is found.



Step 4: Click Searched module ID and give the new configuration



Then follow the steps to check the new setting.

Note: Remember to remove the connection of I-7000's INIT* and GND after the setting is well configured. Then recycle its power. For I-87K I/O modules, remember to switch the related Dip to "OFF", then recycle its power. The I/O modules cannot be used under the INIT state.

IMPORTANT NOTES regarding remote I-7000 & I-87xxx Modules:

One I-8xx7, I-7188EG/XG, µPAC-7186EG, µPAC-5xx7, iP-8xx7 and VP-2117 controller system can link up to a maximum of 64 pcs. of I-7000 and I-87xxx modules (**However 255 pcs for WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6, VP-25W7/23W7**). It recommends on maximum 40 linked modules for one controller system. **Each I-7000 and I-87xxx module MUST have it's own unique address to properly link to an ISaGRAF controller system. In the "Dcon Utility", the default "Checksum" setting is "disabled" and each I-7000 and I-87xxx modules must set to the same baud rate as the controller system.**

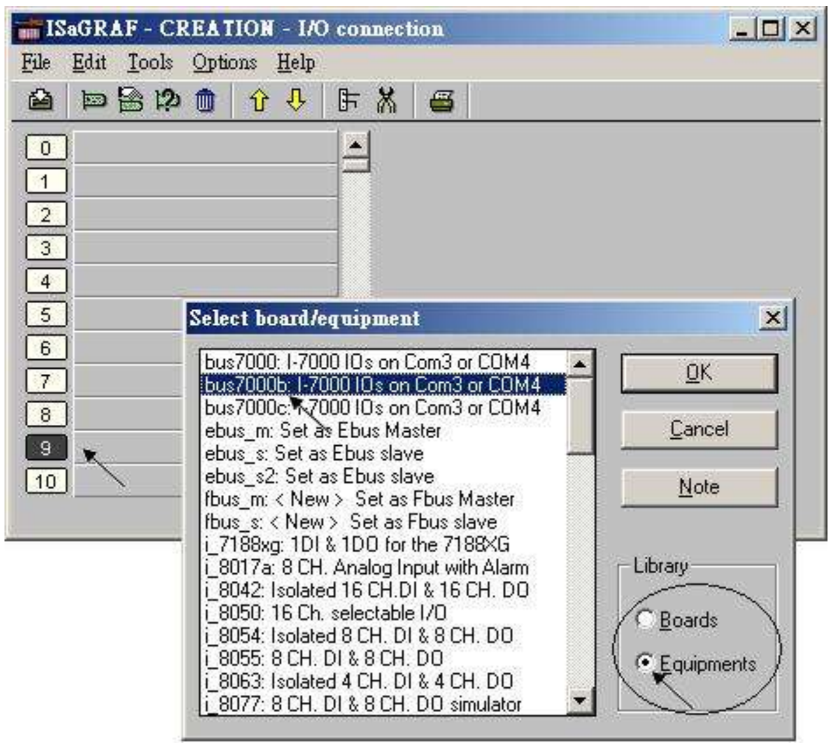
If the type for I-7000 and I-87xxx I/O module is Analog Input, please configure the format as "2's complement" by DCON utility.
Like these AI modules : I-7005, I-7013,I-7015, I-7016, I-7017, I-7017R, I-7018, I-7018R, I-7019, I-7019R, I-7033, I-87005W, I-87013W, I-87015W, I-87015PW, I-87016W, I-87017W, I-87017RCW, I-87017ZW, I-87017DW, I-87018W, I-87018RW, I-87018ZW, I-87019RW and I-87019ZW, etc.

If the type for I-7000 and I-87xxx I/O module is Analog Output, please configure the format as "Engineer Unit" by DCON utility.
Like these AO modules : I-7021, I-7022, I-7024, I-87022W, I-87024W and I-87026W.

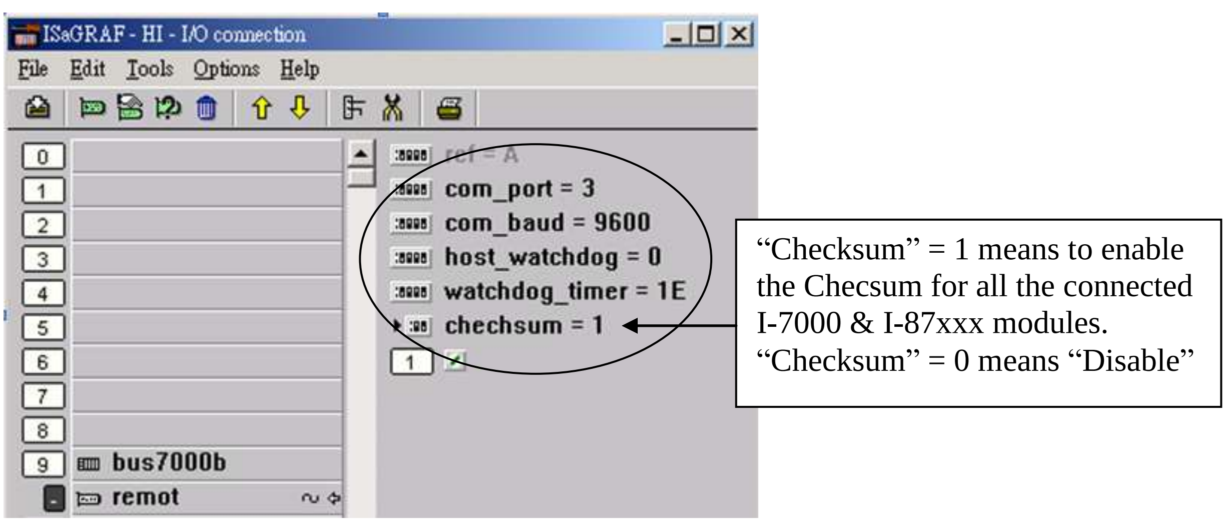
6.2: Opening The "Bus7000b" Function

To create a link between the ISaGRAF controller system and an I-7000 and I-87xxx module, you need to connect the "Bus7000" function (or “Bus7000b”, the “Checksum” can be set as “Enable” or “Disable”, but the "Bus7000" can only be used when the “Checksum” is “Disable”) through the "ISaGRAF I/O Connection" window. The "Bus7000b" function is considered a "virtual board", and must be selected from the "Equipments" section of the "Select Board/Equipment" window.

The "Bus7000b" MUST be connected to slot number 8 or higher on the "ISaGRAF I/O Connection" window (since slot 0 ~ 7 are used to connect to real I-8xxxW and I-87xxxW I/O boards). **Only one "Bus7000b" can be linked to one ISaGRAF controller system!** If you attempt to connect more "Bus7000b" to an ISaGRAF controller, it will not work.



The following figure shows a "Bus7000b” is linked to slot 9.



The "**com_port**" parameter can have a value of 3 (for COM3) or 4 (for COM4) for the I-8xx7, iP-8xx7 controller, while 2 (COM2) or 3(COM3) for the I-7188EG/XG, µPAC-5xx7 & µPAC-7186EG, while 2 (COM2) for the WP-8xx7, WP-5xx7, VP-25W7, VP-23W7 and 3 (COM3) for the XP-8xx7-Atom-CE6, XP-8xx7-CE6. This parameter defines which COM port ID the controller system will communicate with the I-7000 / I-87xxx module.

The "**com_baud**" parameter defines the baud rate that the controller will communicate with the I-7000 / I-87xxx module. The possible values are 2400, 4800, 9600, 19200, 38400, 57600, and 115200. In order to have a smooth communication, you must make sure that the controller system and the I-7000 / I-87xxx modules are all set to the same "com_baud" value.

The "**host_watchdog**" parameter defines to enables or disables the watchdog function for the I-7000 and I-87xxx module. Setting the "host_watchdog" parameter to “1” will enable the "host_watchdog" feature, set it to “0” will disable this 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 controller system (it means the connection is break between the controller and I-7000 / I-87xxx modules), the I-7000 / I-87xxx modules will go to a "safe" predetermined value by DCON utility. (Normally for Digital Output channel, the “safe” state is D/O=False.)

There is an analog input channel available on the "Bus7000b: Remote" virtual board. This analog input channel will return a value equal to the currently set baud rate. If the value is “0” means that it fails to open the communication port.

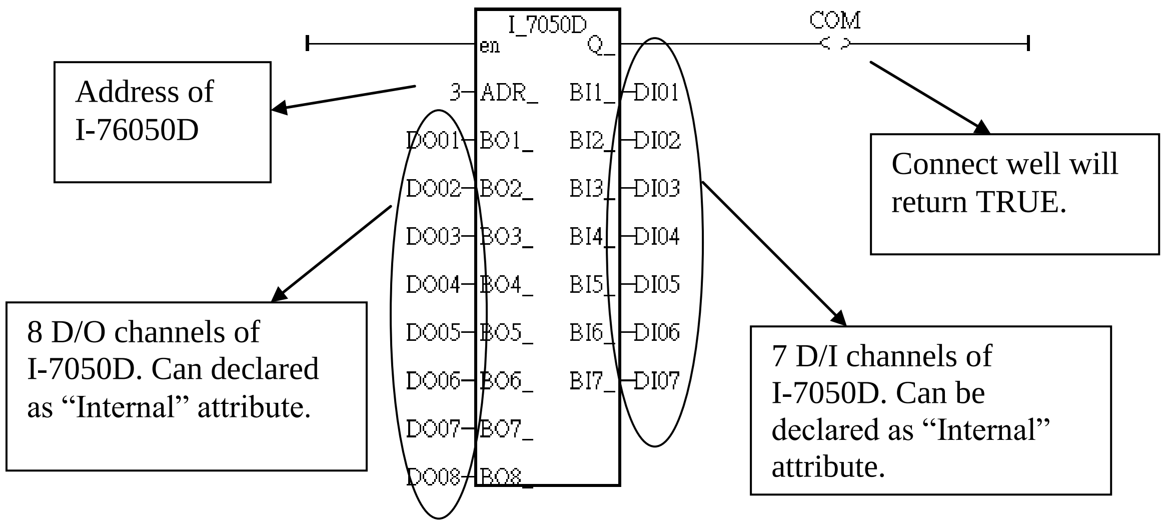
6.3: Programming an I-7000 & I-87xxx Module

6.3.1: Program I-7xxx or I-87xxx remote IO function blocks

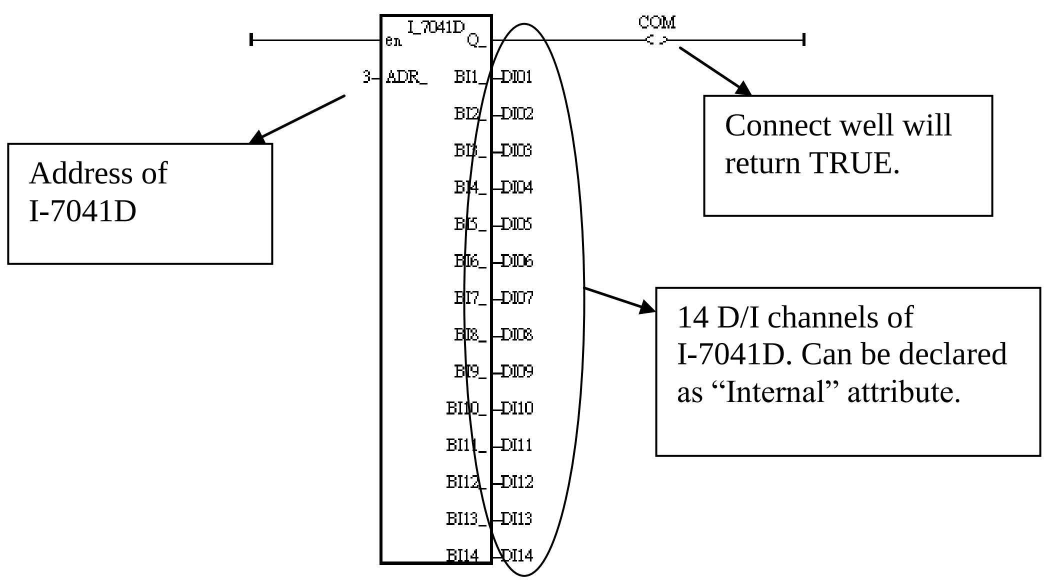
To link any I-7000 and I-87xxx module to the ISaGRAF controller system, the "Bus7000b" module MUST be opened first. Once the "Bus7000b" is opened, the "I_7xxx" / “I-87xx” function block can now be programmed and you can access all of the I/O channels available from that function block, and that data can now be used in a LD program.

NOTE:
Please declare all variables which connect to the I-7xxx / I-87xxx block as “**Internal**” attribution.

Example 1: Programming an I-7050D Module



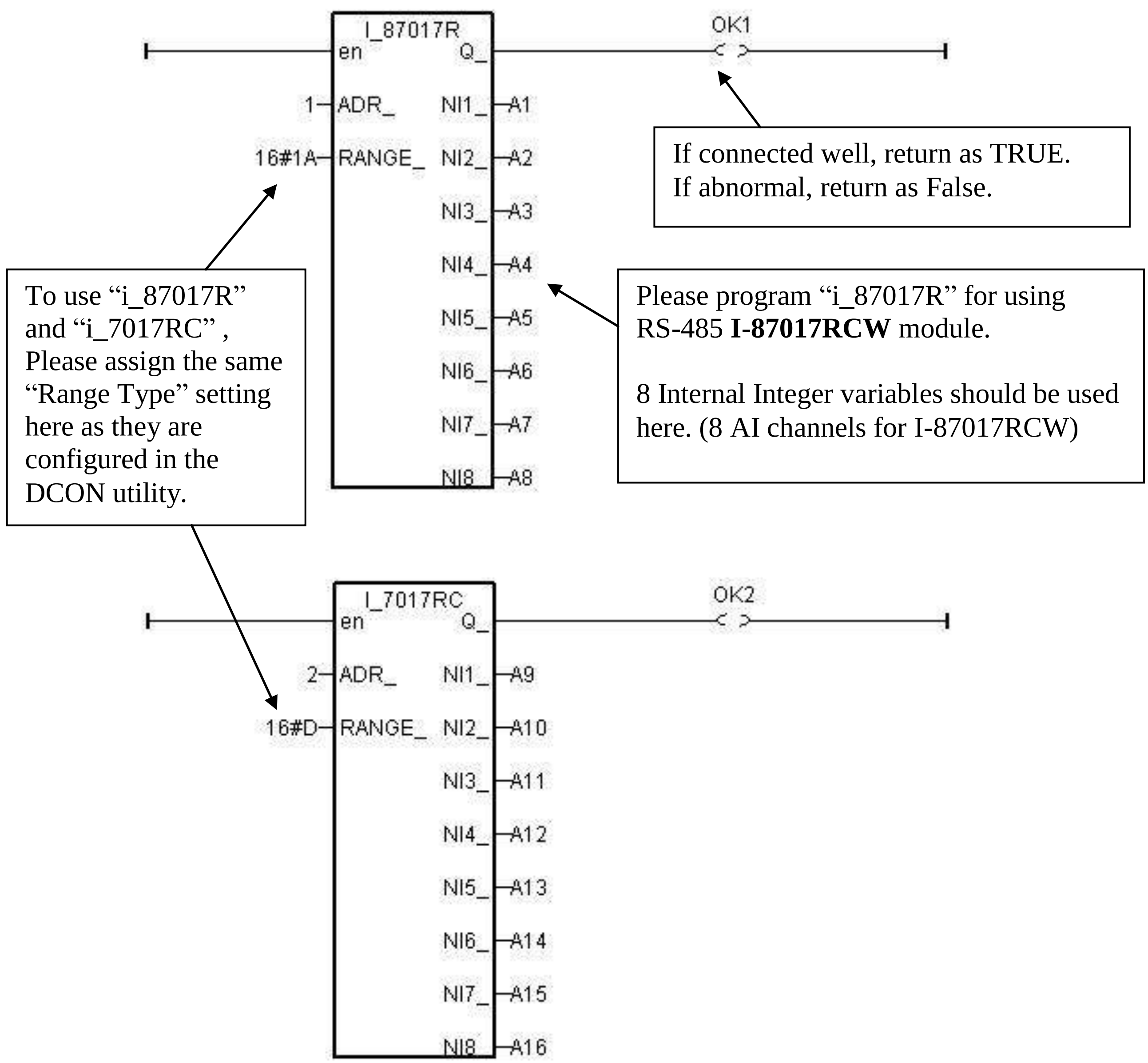
Example 2: Programming an I-7041D Module



Example 3: Programming a I-87017R or I-7017RC function block (Used when the hardware is I-**87**017RCW or I-**7**017RC)

I-87017RCW and I-7017RC can measure current input of ± 20 mA, 0 ~ 20mA and 4 ~ 20mA without an external 125 ohm resistor. Please configure their format as "2's complement" by DCON utility. (The "A4_20_to" function can be used to convert the analog input value to user's engineering value, please refer to Appendix A.4)

Range type (by "DCON Utility")	Physical value	I-7017RC /87017RCW Analog Input value (Decimal)		
		- 32768	0	+32767
7	4 ~ 20 mA		4 mA	20 mA
D	± 20mA	- 20mA	0 mA	20mA
1A	0 ~ 20 mA		0 mA	20 mA



IMPORTANT NOTES Note for Using RS-485 Remote I/O to Measure the 4 ~ 20 mA Current:

If the current input sensor is 4 to 20 mA, user may be better set the range type of analog input module to "[D] : +/- 20 mA" , or "[1A] : 0 ~ 20 mA" . (set as " [7] : 4 to 20 mA" is not good)

The reason is:

If setting the range type as "[7] : 4 to 20 mA", analog Input value of 0 or close to 0 could mean the Sensor input is 4 mA , and also possible the Sensor is broken-line. So it is not easy to distinguish these two situations by this AI value.

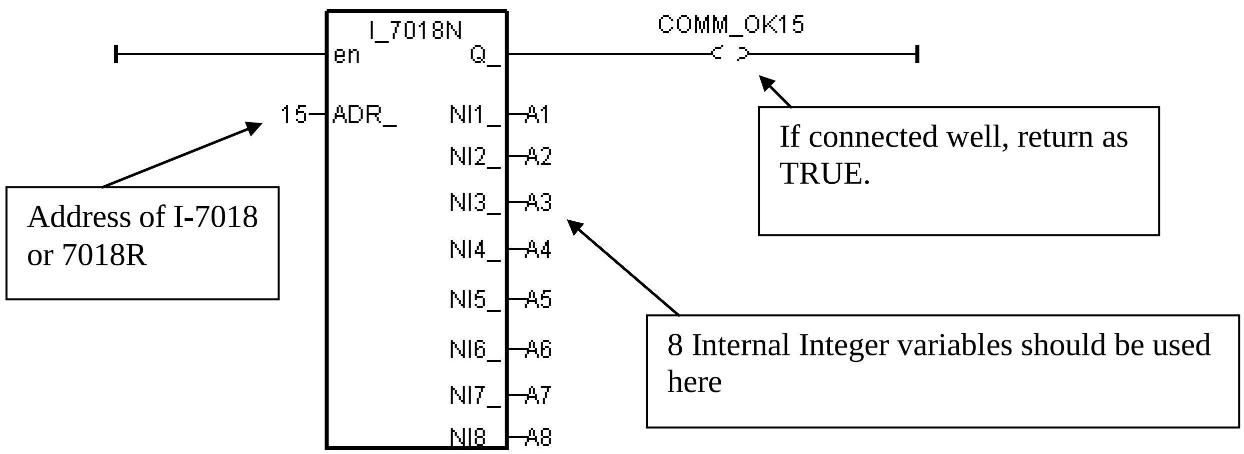
Howevr, if setting the range type as "[D] : +/- 20 mA" or "[1A] : 0 ~ 20 mA", analog input value of 0 or close to 0 only means the Sensor is broken-line . If the Sensor input is 4 to 20mA, the analog value should be 6553 to 32767. When it input 4 mA, the value is 6553 not close to 0.

(Of course, the RS-485 communication state between the AI module and controller must be Ok. The "Ok1" and "OK2" variable in the above example 3 can indicate the communication is Ok or not. If the communication is False, it means the controller can not link to the RS-485 I/O well. You need to handle this situation in your ISaGRAF program).

So, if you want to distinguish whether the Sensor (4 ~ 20 mA) is OK? It would be better to set the type as "[D] : +/- 20 mA" or "[1A] : 0 ~ 20 mA". Then, you can set the condition in your ISaGRAF program. For example, when the A1 ~ A16 input value is less than 5000 or 4000, it can be regarded as Sensor broken-line or abnormal.

Example 4: Program I-7018 block (Please use new “I_7018n” block)
(**I-7018z is a better hardware choice. Please refer to Chapter 11.3.9 for demo example**)

Please configure I-7018 and I-7018R‘s format as “2’s complement” by DCON utility. Then please program a “I_7018**n**” block (The “I_7018**n**” block request all 8-channels by one single command, however the “I_7018” block need to send 8 commands for 8-channels)



The other RS-485 I-7000 and I-87K I/O all use the similar way.
Note:
If RS-485 remote I-7000 and I-87xxx I/O module’s type is Analog Input, please configure the format as “2’s complement” by DCON utility. Like :
I-7005, I-7013,I-7015, I-7016, I-7017, I-7017R,I-7018, I-7018R,I-7018Z, I-7019, I-7019R, I-7033, I-87005W, I-87013W, I-87015W, I-87015PW, I-87016W, I-87017W, I-87017RCW, I-87017ZW, I-87017DW, I-87018W, I-87018RW, I-87018Z, I-87019RW, I-87019ZW and so on.

If RS-485 remote I-7000 and I-87xxx I/O module’s type is Analog Output, please configure the format as “Engineer Unit” by DCON utility. Like :
I-7021, I-7022, I-7024, I-87022W, I-87024W and I-87026W .

Below table is for the I-7017, 7017R, 87017W, 87017RW. (These modules need an external 125 ohm resistor when using “D: ± 20mA”. If you don’t want to use it, please choose I-7017RC or I-87017RCW or I-87017Z or I-87019ZW)

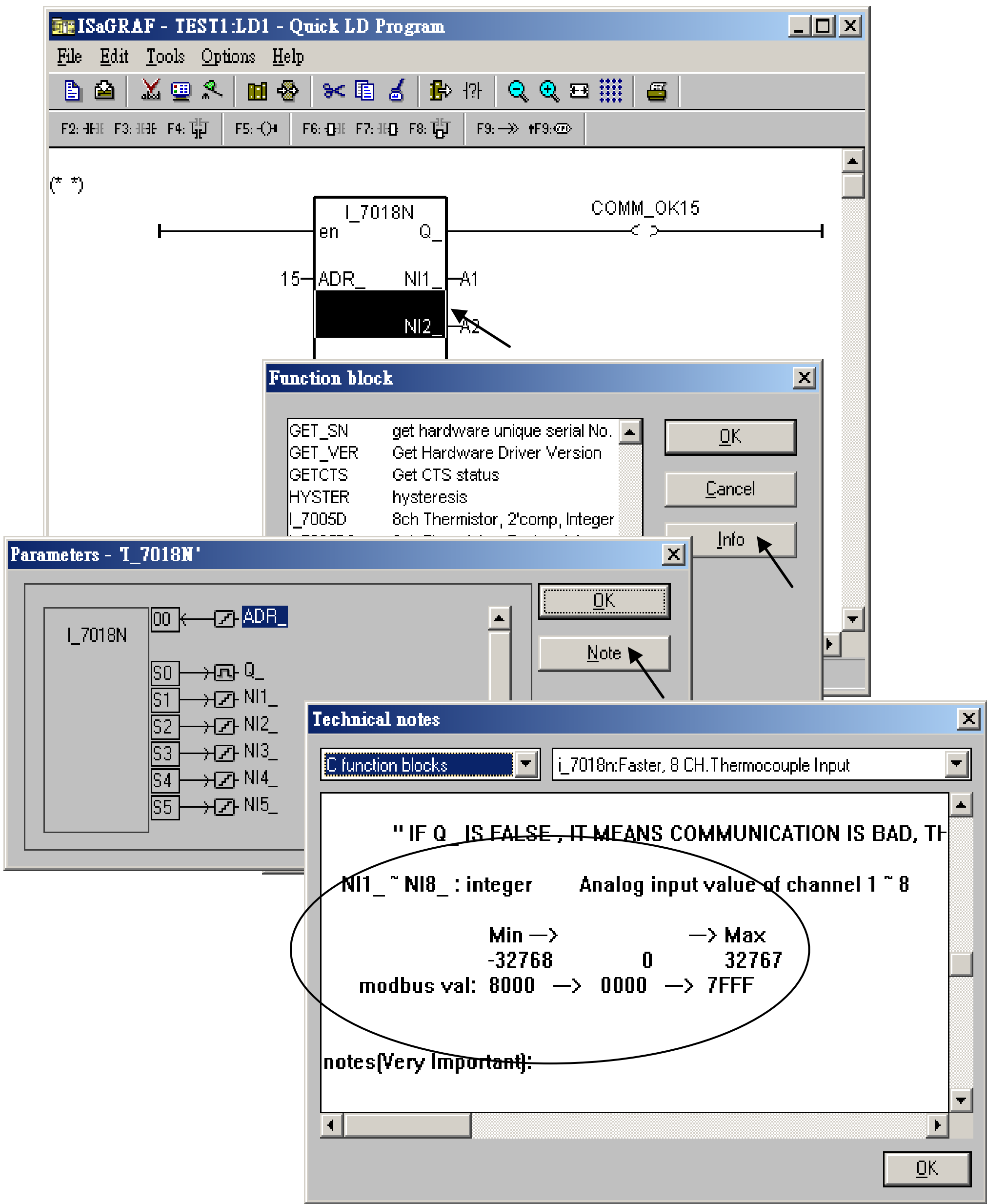
Range tyep (by DCON Utility)	Physical value	I-7017 / 87017 Analog Input value (Decimal)		
		- 32768	0	+32767
8	± 10V	- 10V	0V	+ 10V
9	± 5V	- 5V	0V	+ 5V
A	± 1V	- 1V	0V	+ 1V
B	± 500mV	- 500mV	0mV	+ 500mV
C	± 150mV	- 150mV	0mV	+ 150mV
D	± 20mA	- 20mA	0mA	+ 20mA

Below table is for the I-7017RC and I-87017RCW (no external 125 ohm resistor required)

Range type (by DCON Utility)	Physical value	I-7017RC / 87017RC W Analog Input value (Decimal)		
		- 32768	0	+32767
7	4 ~ 20 mA		4 mA	20 mA
D	± 20mA	- 20mA	0 mA	20mA
1A	0 ~ 20 mA		0 mA	20 mA

Please refer to the on-line help for each I/O module's table or refer to the Appendix D.

Please mouse double clik on the function block and click on "Info", then click on "Note".



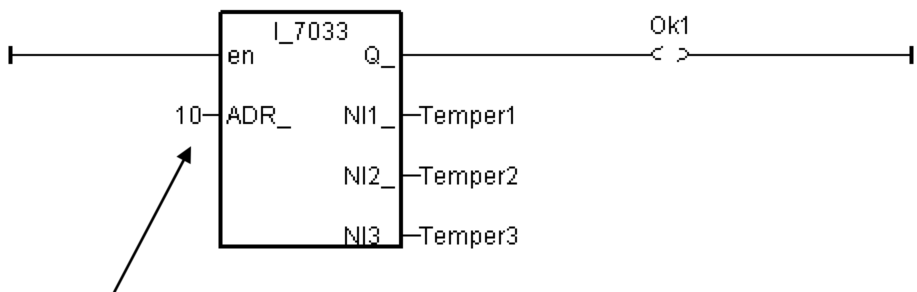
6.3.2: Setting a special "ADR_" parameter of remote temperature input module to get clear "Degree Celsius" or "Degree Fahrenheit" input value

ICP DAS provides many temperature input modules as below.

With "broken-line detection" or called "wire opening detection"
Thermocouple type:
I-87018ZW, 87018RW, 87019RW, 87019zW, 7018R, 7018BL, 7018Z, 7019, 7019R
RTD type: I-87013W, 87015W, I-87015PW, 7013, 7015, 7033
Thermister type: I-87005W, 7005

Without "broken-line detection"
Thermocouple type: I-87018, 7018, 7018P

The "ADR_" parameter of temperature IO function block can be "standard setting" or "special setting". For example setting "ARD_" of the "I_7033" function block to 1 to 255 (Dec. value) means "standard setting", the value of 1 to 255 indicates the address of the remote I-7033. The temperature input value is normally –32768 to + 32767 in the case. It depends on the IO module's "Type code" setting (Set by DCON utility). (normally value of –32768 & +32767 means wire "broken-line")



ADR_ = 10 (TT=00, RR=00, AA=0A, Hex.) means "standard setting", address=10, the temperature input value is normally –32768 to + 32767

If ADR_= 16#10201A (TT=10, RR=20, AA=1A, Hex) means "special setting", "Degree Celsius", "type code=20 of this I-7033 module set by DCON utility", address=26, the temperature input value is a clear "Degree Celsius" value, for example, value of 4556 mans "45.56" degree. "-500" means "-5.00" degree.

If user want to get a clear temperature input value, for example, value of 2312 means "23.12" Degree Celsius. Then please set "ADR_" to a special value defined as below.

Important: Special "ADR_" setting is supported since driver version of
I-8xx7: 3.11, I-7188EG: 2.09 , I-7188XG: 2.07, μPAC-7186EG: 1.01, μPAC-5xx7: 1.01, iP-8xx7: 1.01, WP-8xx7: 1.01, WP-5xx7: 1.01, XP-8xx7-Atom-CE6: 1.01, XP-8xx7-CE6: 1.01, VP-25W7/23W7: 1.01

Format: TTRRAA (Hex.)

TT=10 (Convert to "Degree Celsius") , Unit is 0.01 degrees.
TT=20 (Convert to "Degree Fahrenheit") , Unit is 0.01 degrees.
TT=00 (standard setting, -32768 to +32767. RR should be set as 00 if TT=00)
RR: "type code" setting of the related temperature input module
(The Initial value is configured by DCON Utility)
AA: address of the related temperature input module (01 ~ FF)

For example, setting "ADR_" as

A. 16#102011 : (TT=10, RR=20, AA=11, Hex) the input value will be "Degree Celsius", unit is 0.01 degree, range= "20 : Platinum 100, a=0.00385, degree Celsius", address=17(Dec.). That results input value of "2356" = 23.56 Degree Celsius, "-489" = -4.89 Degree Celsius, "999990" = sensor broken-line.

B. 16#202A03 : (TT=20, RR=2A, AA=03, Hex) the input value will be "Degree Fahrenheit", unit is 0.01 degree, range= "2A : Platinum 1000, a=0.00385, degree Celsius", address=3(Dec.). That results input value of "4512" = 45.12 Degree Fahrenheit, "500" = 5.00 Degree Fahrenheit, "999990" = sensor broken line.

C. 16#01 : (TT=00, RR=00, AA=1) standard setting, the input value will be , -32768 to +32767, address=1

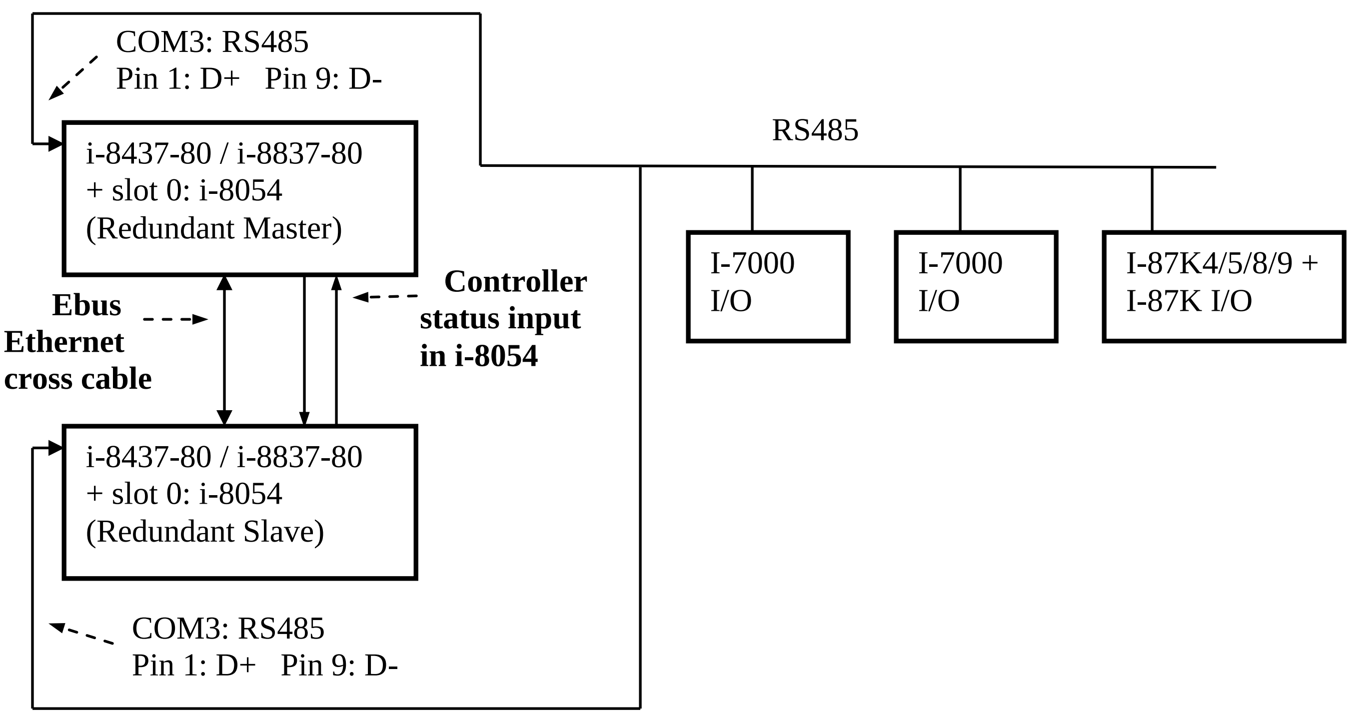
6.4: Redundant Bus7000

Note:

1. XP-8xx7-CE6, XP-8xx7-Atom-CE6 is a better redundant system; please refer to FAQ-125 or FAQ-138. (http://www.icpdas.com/faq/isagraf.htm)
2. The I-8437-80, I-8837-80 or iP-8x47 can setup a Bus7000b redundancy system as the figure below. Their CPUs are 80MHz. The CPU speed is about 2 to 4 times of the I-8417/8817/8437/8837's CPU (40MHz).
3. The 40 MHz I-8417/8817/8437/8837 and I-7188EG and I-7188XG are not good for Bus7000b redundancy system. Please use the best solution of Item (1) or the solution of Item (2)

I-8437-80/I-8837-80 (Driver since v3.20 or later) and iP-8x47 supports Redundant Bus7000b. The Ebus are for exchanging data between the "Redundant Master" & "Redundant Slave". Please wire Ch. 1 output of the redundant master's I-8054 to Ch.1 input of the redundant slave's I-8054. And also wire Ch. 1 output of the redundant slave's I-8054 to Ch.1 input of the redundant master's I-8054. These two Status inputs are to indicate the other controller – "I am still alive".

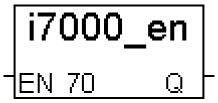
I-8437-80 : Bus7000 redundancy system



Operations Principle:

1. When the system is powered up, the control of Bus7000b belong to "Redundant Master".
2. If "Redundant Master" is damaged (or Power off), "Redundant Slave" takes the control of Bus7000b.
3. If "Redundant Master" is alive from damaged (or power up again), it takes the control of Bus7000b again.
4. Control data is exchanging via Ebus (if using a cross cable, no need any ethernet switch).

The “i7000_en” can be used to Enable/Disable the control right of Bus7000. The system’s default status is “Enable”.



Parameter:

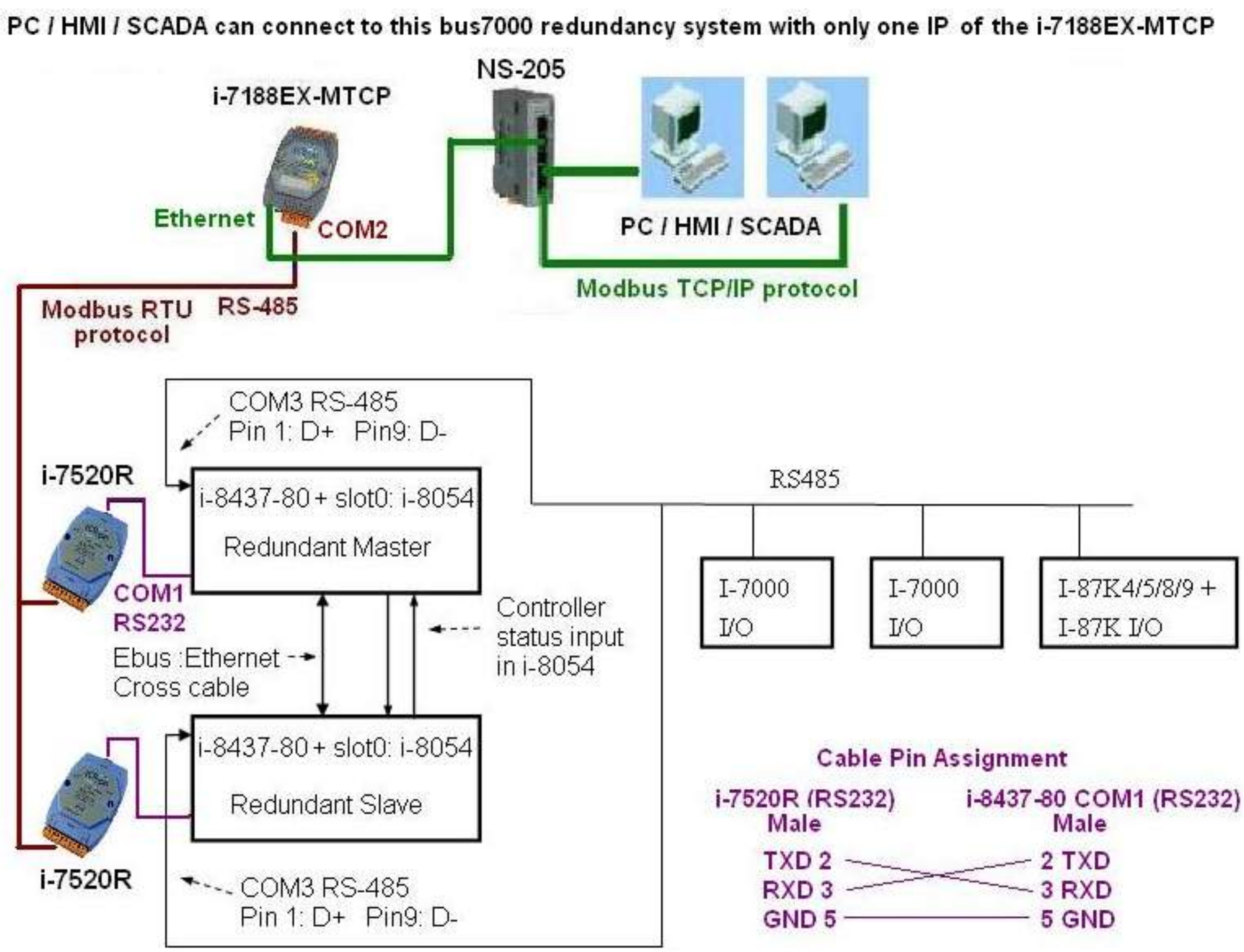
EN_7000_	integer	True: Enable, False: Disable

Return:

Q_	Boolean	Always return True.

User can use the “COM_MRTU” function to disable the I-8437-80‘s COM1 port if it is NOT redundant active (then its COM1 will never answer any question to the PC / HMI / SCADA). And also enable its COM1 by “COM_MRTU” function if it is redundancy active. Then at any time only the redundancy active controller will reply to the PC / HMI / SCADA as below configuration. (Please refer to demo_49a & demo_49b). For the use of I-7188EX-MTCP (Modbus TCP/IP to Modbus RTU gateway), please refer to Chapter 20.5 or www.icpdas.com – FAQ – Software – ISaGRAF – 062.
(**Important: Please set these two I-8437-80‘s Net-ID to the same No. for ex. , setting as No. 1. And the IP should be different but in the same domain. For ex. , setting as 192.168.1.8 and 192.168.1.9. Mask should all set to 255.255.255.0**)

Demo program: “demo_49a” and “demo_49b”.
ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/7188eg/demo/



Chapter 7. Controller To Controller Data Exchange

Important Note:
The max. boolean & integer package No. of Fbus & Ebus reduce from 256 to 128 since driver version of I-8xx7: 2.42, I-7188EG: 1.32, I-7188XG: 1.29, iP-8xx7: 1.01, μPAC-7186EG: 1.01, VP-2117: 1.01

7.1: Basic Fbus Rules

Any I-8xx7, I-7188EG/XG & μPAC-7186EG, iP-8xx7, VP-2117 controller system can access data from another I-8xx7, I-7188EG/XG through the Fbus data exchange system. **While the XP-8xx7-CE6, XP-8xx7-Atom-CE6, WP-8xx7, WP-5xx7, VP-25W7/23W7 doesn't support Fbus, it supports Ebus only. Please refer to section 7.5**. There are 2 types of data that can be exchanged through the Fbus protocol; they are "Boolean" and "integer". If you want to exchange "Real" data, please refer to appendix A.4 to use "Int_Real" and "Real_Int" block.

The Fbus driver first creates a packet of eight Boolean values to form a "Boolean package", and then creates a packet of eight 32-bit integers to form an "integer package". Both of the "Boolean packages" and "integer packages" can be distributed on the Fbus to allow the data to be exchanged from one controller system to another or more controller system.

The Following Fbus Rules MUST Be Observed:

RULE #1: Each "Boolean package" must have an attached identification number ranging from 1 to 128. This means that there is a maximum of 128 "Boolean packages" that can be exchanged across an Fbus connection.

Each "Boolean package" contains 8 Boolean values, and these Boolean values can only have the value of either "True" or "False". The Boolean values in the "Boolean package" can be assigned and exchanged with either "Internal", "Input", or "Output" Boolean variables or Boolean constants.

RULE #2: Each "integer package" must have an attached identification number ranging from 1 to 128. This means that there is a maximum of 128 "integer packages" that can be exchanged across an Fbus connection.

Each "integer package" contains eight 32-bit integer values. The integer values can range from –2147483648 to 2147483647. The integer values in the "integer package" can be assigned and exchanged with either "Internal", "Input", or "Output" integer variables or integer constants.

Rule #3: Each identification number assigned to a "Boolean package" or an "integer package" can only be written to by one controller system across the Fbus.

Each controller system CANNOT **write** the same identification number for either a "Boolean package" or an "integer package" across the Fbus. WRITTING A PACKAGE IS NOT SHARED with the other controller systems across the Fbus network.

In this example, there are five I-8xx7, I-7188EG/XG controller systems communicating through an Fbus network, and the controller systems are named S1, S2, S3, S4, and S5 respectively. If the S1 controller system attempts to write a "Boolean package" with an ID of "1" and an "integer package" with an ID of "1" across the Fbus, the other four controllers CANNOT write either a "Boolean package" or an "integer package" with the same number. However, the other controller systems could write a "Boolean package" with an ID of "3" and an "integer package with an ID of "3".

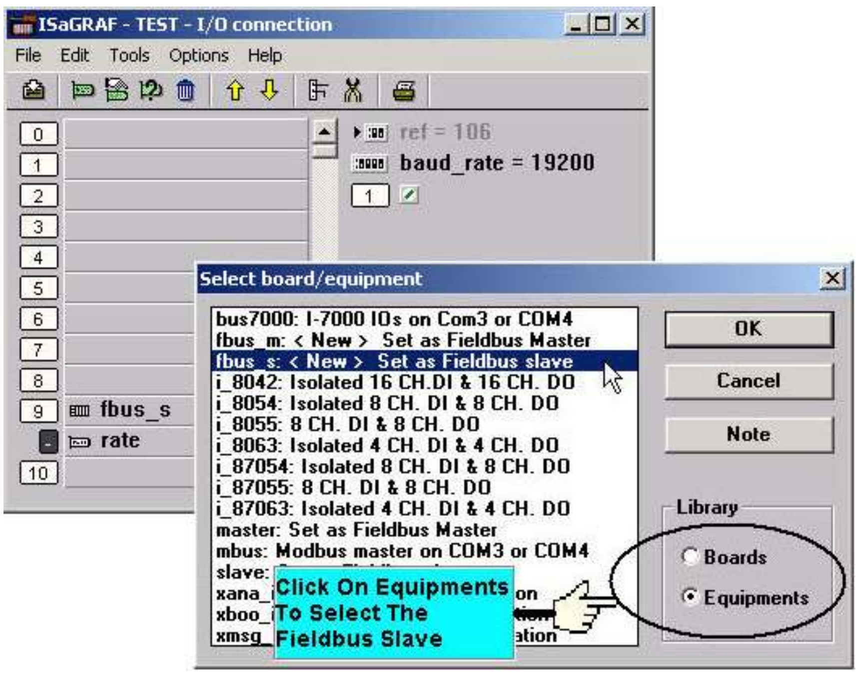
There is no limitation on how many controllers can read the same number package across the Fbus network. Any of the S2, S3, S4 and S5 controller systems can read the "Boolean package" with an ID of "1" and the "integer package" with an ID of "1" if desired.

Rule #4: ONLY ONE controller system can be configured as a Fbus "Master", all the others controller systems MUST be configured as a Fbus "Slave".

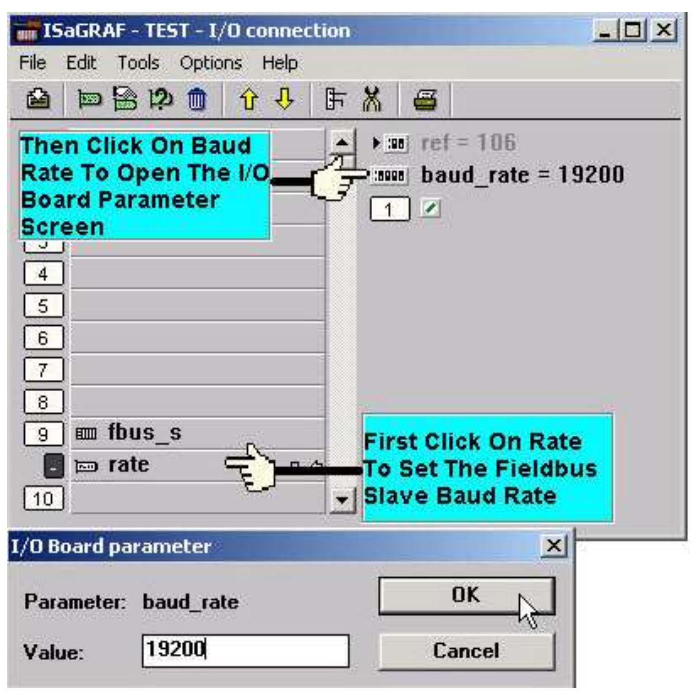
The "master" controller sends commands for how data is to be exchanged across the Fbus network. If you configure more than one controller system as a "master", or configure none of the controller systems as a "master" on the Fbus, NO DATA CAN BE EXCHANGED across the Fbus network.

7.2: Configuring The ISaGRAF PAC To Be A Fbus "Master" Or "Slave"

To begin configuring a controller system as either a Fbus master or slave **(while the XP-8xx7-CE6, XP-8xx7-Atom-CE6, WP-8xx7, WP-5xx7, VP-25W7/23W7 doesn't support Fbus, it supports Ebus only. Please refer to section 7.5**.), first open up the "ISaGRAF I/O Connections" window and double click on a slot number higher than 7. The "Select Board/Equipments" window will now open, click on "Equipments", and then double click on the "fbus_s" selection to configure an Fbus slave, or double click on "fbus_m" to configure an Fbus master. Remember, **ONLY ONE** controller can be the Fbus master, and you **CANNOT** configure a controller system to be both a Fbus master and a Fbus slave.

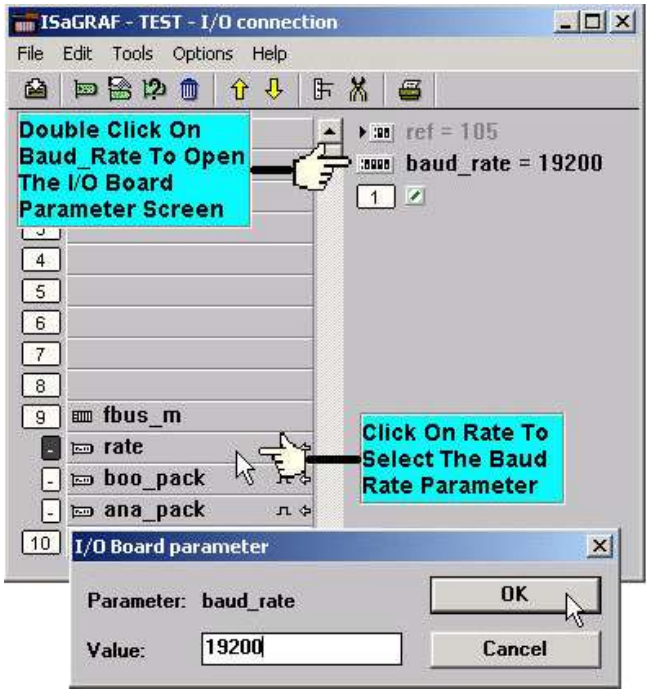


If you configure a controller system as an Fbus slave, only one parameter needs to be set, and that is the "baud_rate" parameter. The baud rate parameter can be set to 2400, 4800, 9600, 19200, 38400, 57600 or 115200 baud rate. The default baud rate value is 19200 for the controller system. All controllers on the same Fbus network MUST be set to the same baud rate.



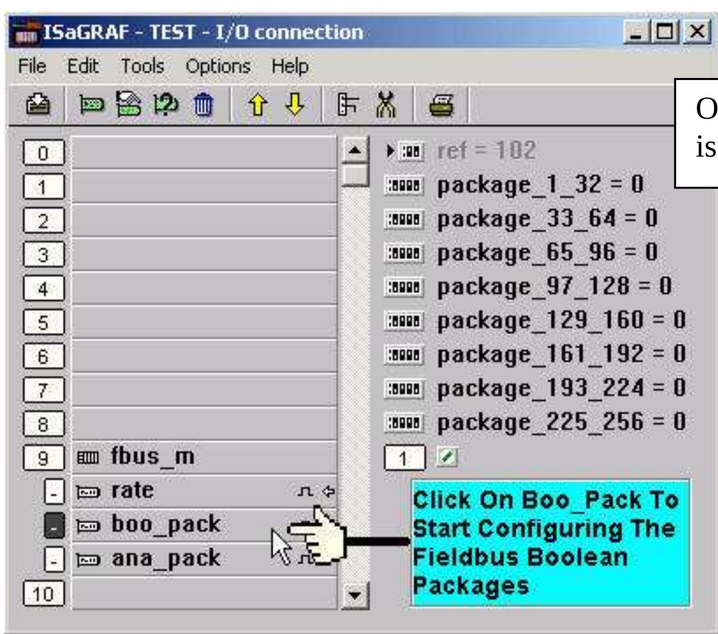
There is a digital input channel associated with the "fbus_s: rate" equipment. If the Fbus connection has been established, the digital input channel will return a "TRUE" value. If the Fbus connection failed to establish, the digital input channel will return a "FALSE" value.

If you configure an controller as Fbus master, the parameter "baud_rate" and "fbus_m: rate" can be set to 2400, 4800, 9600, 19200, 38400, 57600 or 115200. The default value is 19200 for the controller. All controllers on the same Fbus MUST be set to the same baud rate.



There is a digital input channel associated with the "fbus_m: rate" equipment. If the Fbus connection has been established, the digital input channel will return "TRUE" value, if the Fbus connection failed to establish, the digital input channel would return a value of "FALSE".

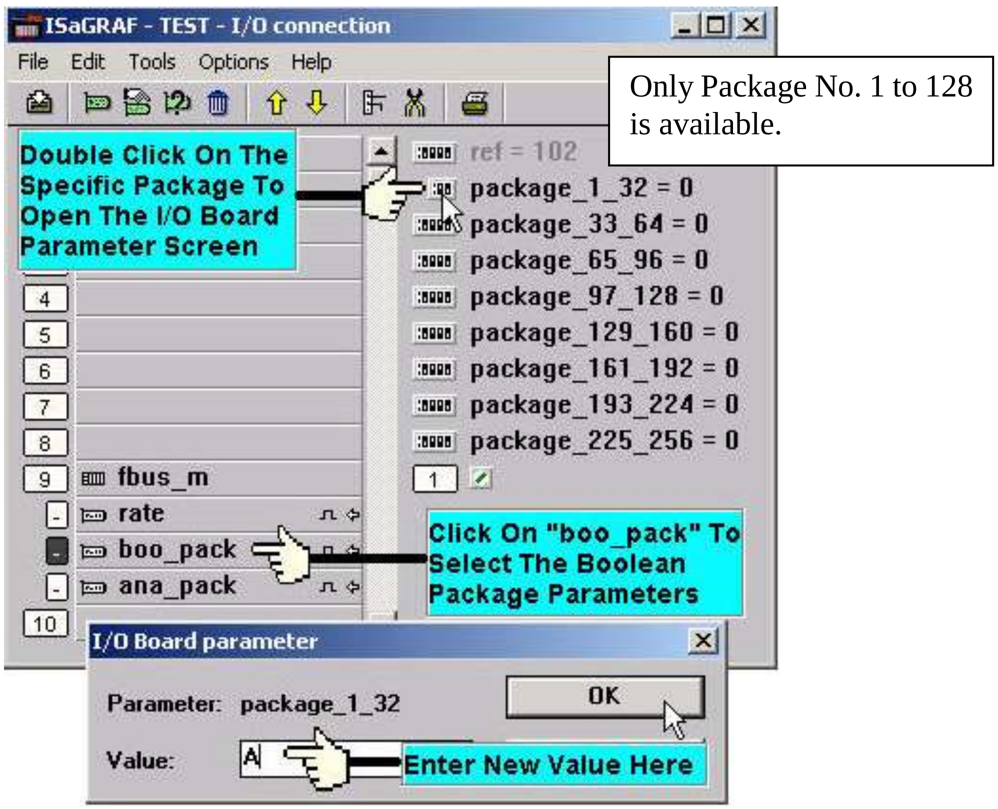
To begin configuring the Fbus Master Boolean Packages, click on the "boo_pack" selection from the "fbus_m" I/O connection.

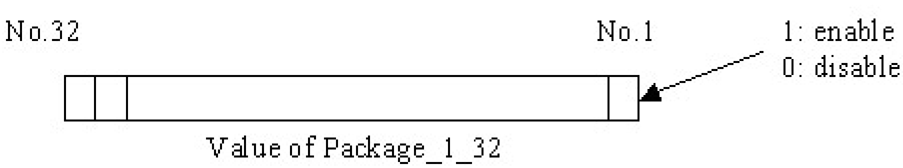


Only Package No. 1 to 128 is available.

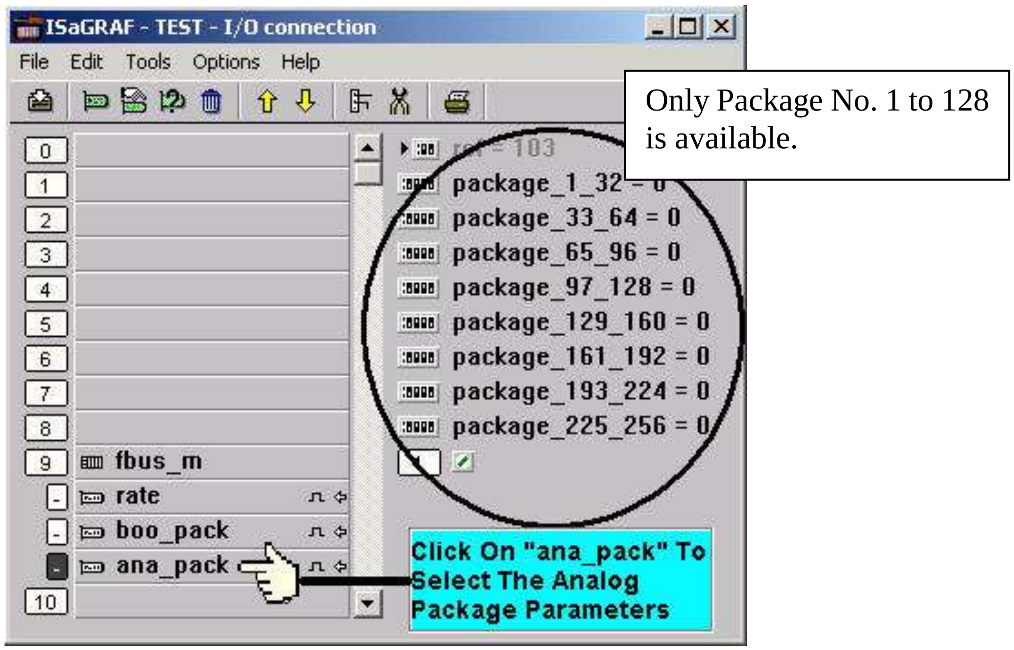
The parameter "package_xxx_xxx" at "fbus_m: boo_pack" indicates the "Boolean package" number which is allowed to be written to or read from across the Fbus network. The parameter value is given as a 32-bit integer in **hexadecimal**.

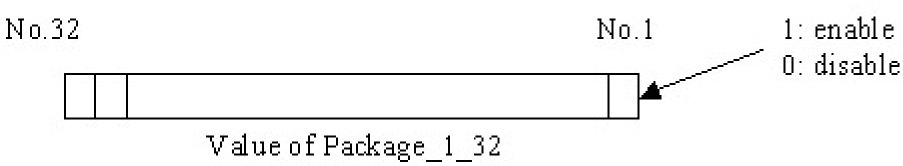
As an example, if the "package_1_32" is set to "FFFFFFFF" this will enable all the packages from number 1 to number 32 to be written to or read from across the Fbus network. If the "package _1_32" is set to a value of "A", this will only enable the number 2 and number 4 Boolean packages to be written to or read from across the Fbus network. The more packages that are enabled on a Fbus network the slower the communication efficiency will be. **With this in mind, always remember to enable only the required number of packages that you need for your application so you will have greater communication efficiency across the Fbus network**.





The parameter "package_xxx_xxx" at "fbus_m: ana_pack" indicates the "integer package" number which will be written to and read from on the Fbus network. The "fbus_m: ana_pack" is used to read and write 32-bit integer values across the Fbus network. Each of the parameter values is expressed as 32-bit integer values in **hexadecimal**, and the same configuration rules apply as those for the "Boolean package".





7.3: Programming Fbus Packages

Before you can exchange any data across a Fbus network, you must make sure that each controller is configured as either a Fbus master "fbus_m" or Fbus slave "fbus_s" (and remember, only ONE controller can be the master). Refer to Section 7.2. (The WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6, VP-25W7/23W7 doesn't support Fbus, it only support Ebus, please refer to Section 7.5)

The following Fbus function blocks can be used in a LD program to exchange data across an Fbus network.

Fbus_b_r	read one boolean package.
Fbus_b_w	write one boolean package.
Fbus_n_r	read one integer package.
Fbus_n_w	write one integer package.
Fbus_f_r	read one REAL package
Fbus_f_w	write one REAL package

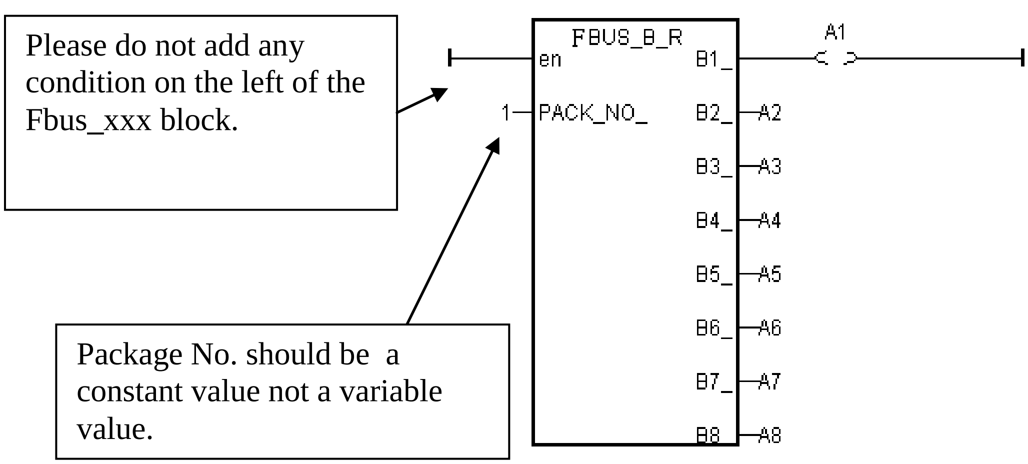
(The Integer package and REAL package use the same memory. Please DO NOT use the same package No. as Integer package and also as REAL package at the same time. Or the local fault No. 116 may happen. Please refer to Chapter 10.6)

The below block is to get the communication ststus of each Boolean & Integer Package.

Fbus_sts	Get ststus of each Package.

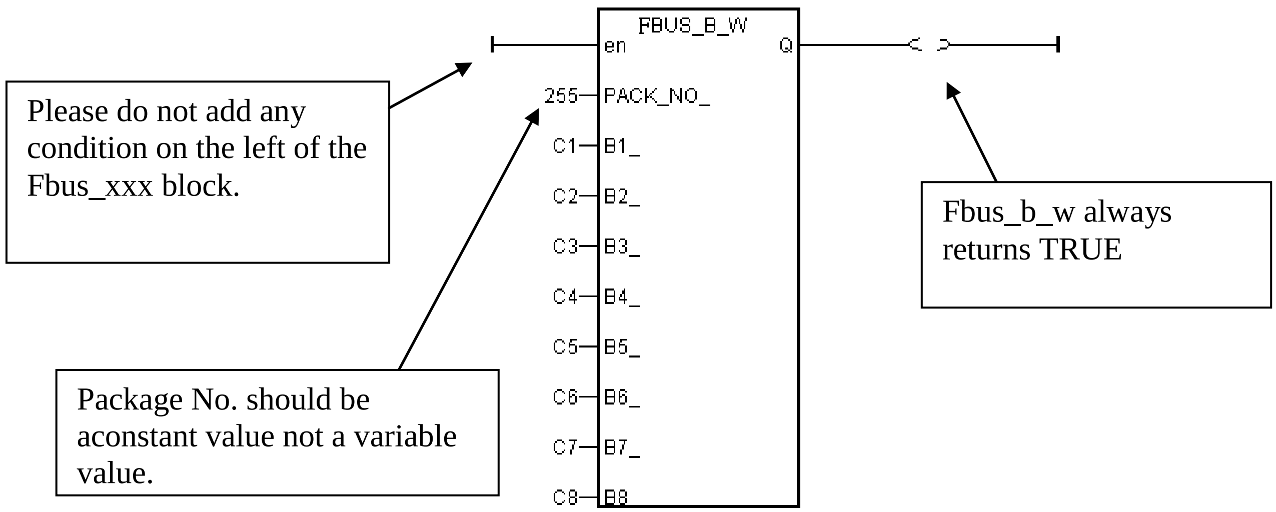
Fbus Function #1: "Fbus_b_r"

The "Fbus_b_r" function reads one Boolean package from the Fbus network. In the example below the "Fbus_b_r" function has a Boolean package ID address of "1". The "A1" output contains the value of the first Boolean of the package No. of 1, the "A2" output contains the value of the second Boolean of the package No. of 1, and the "A3" output contains the value of the third Boolean of the package No. of 1. The other outputs follow the same format to where the "A8" output contains the value of the eighth Boolean of the package No. of 1.



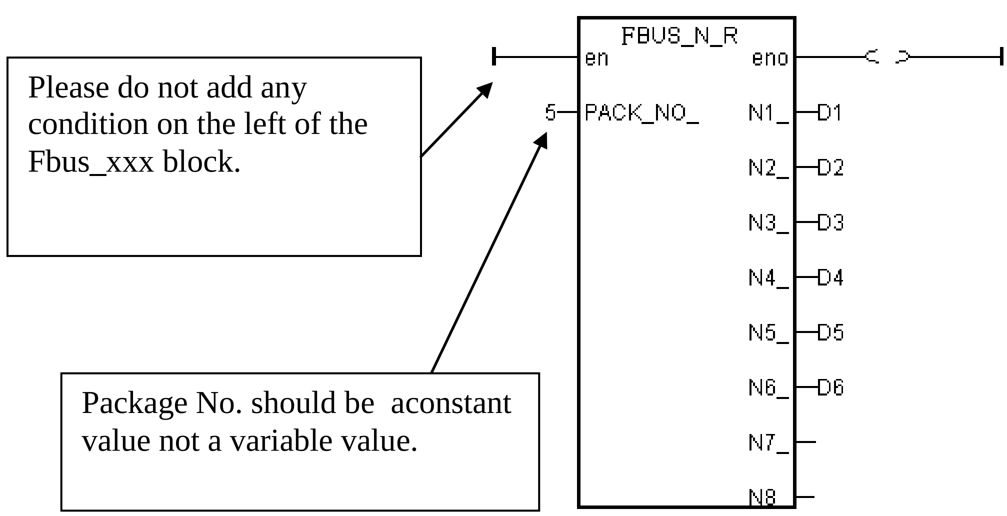
Fbus Function #2: "Fbus_b_w"

The "Fbus_b_w" function writes one Boolean package on the Fbus network. In the example below the "Fbus_b_w" function has a Boolean package ID address of "255", the "C1" input writes a value to the first Boolean of the package No. of 255, the "C2" input writes a value of the second Boolean of the package No. of 255, and the "C3" input writes a value of the third Boolean of the package No. of 255. The other inputs follow the same format to where the "C8" input writes a value of the eighth Boolean of the package No. of 255.



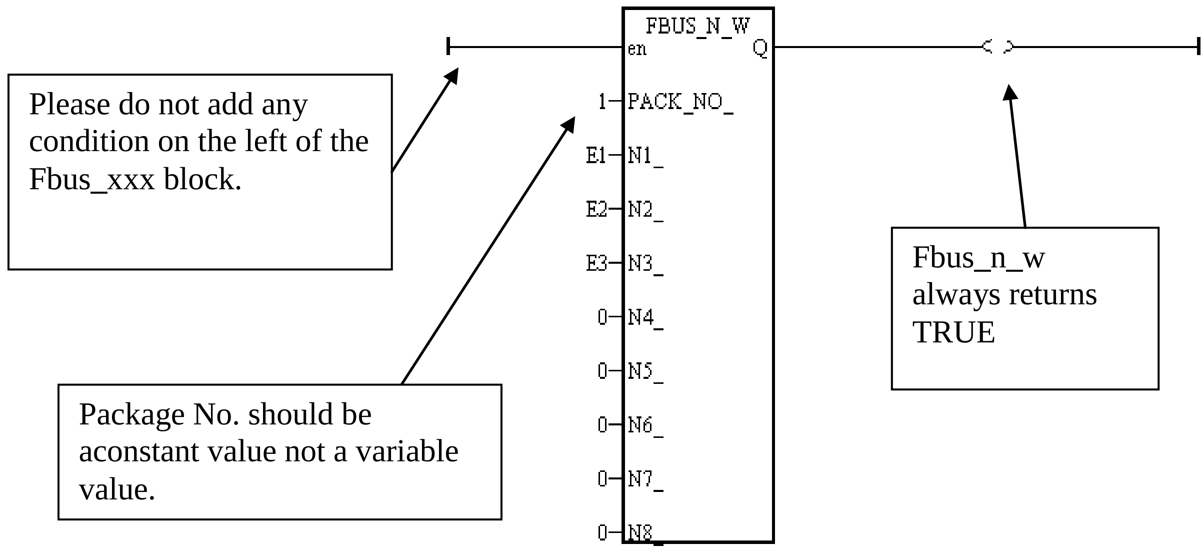
Fbus Function #3: "Fbus_n_r"

The "Fbus_n_r" function reads one integer package from the Fbus network. In the example below the "Fbus_n_r" function has an Integer package ID address of "5". The "D1" output contains the value of the first integer of the package No. of 5, the "D2" output contains the value of the second integer of the package No. of 5, and the "D3" output contains the value of the third integer of the package No. of 5. The other outputs follow the same format to where the "D6" output contains the value of the sixth integer of the package No. of 5.



Fbus Function #4: "Fbus_n_w"

The "Fbus_n_w" function writes one integer package to the Fbus network. In the below example the "Fbus_n_w" function write variables “E1” to the first integer of the package of No. 1. “E2” to the second integer of the package of No. 1. “E3” to the third integer.



7.4: An Fbus Data Exchange Example

Example Description:
In this Fbus data exchange example there are three I-8xx7, I-7188EG/XG controller systems linked together in an Fbus network. These controller systems are named "SA (master controller system #1)", "SB (slave controller system #2), and "SC (slave controller system #3).

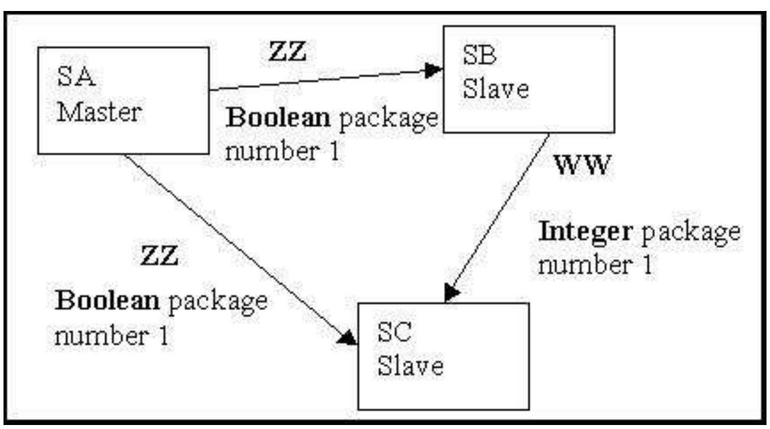
One of the digital input values from the SA controller (master system) needs to be shared with the SB and SC (the slave systems) controllers across the Fbus network, and the name for this digital input value will be called "ZZ".

The first task of this example is to create an **Input** variable named ZZ on the SA controller system. Use the "ISaGRAF Project" window to declare ZZ as an "input" variable, and then link the ZZ input variable using the "ISaGRAF I/O Connections" window for the SA controller system.

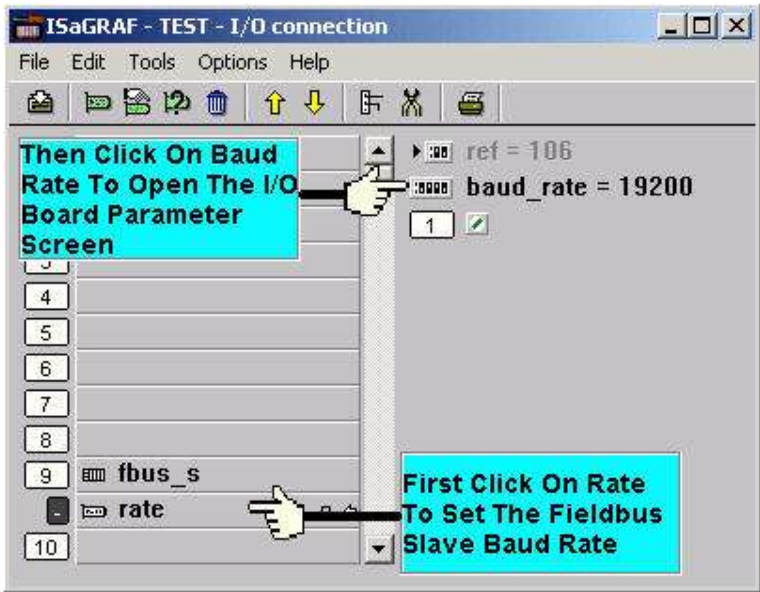
Next, you will need to declare a Boolean **Internal** variable named ZZ for both the SB and SC controllers (so they can exchange the ZZ value with the SA controller system). You must declare the ZZ variable as an internal variable for the SB and SC controllers because there is only one real input variable (from the SA controller) that is being exchanged, and either the SB or SC has a real input variable named ZZ.

An additional requirement for this example is that an internal integer value named "WW" that comes from the SB controller system needs to be shared with the SC controller system. To accomplish this declare an **Internal** integer variable named WW on both the SB and SC controller systems.

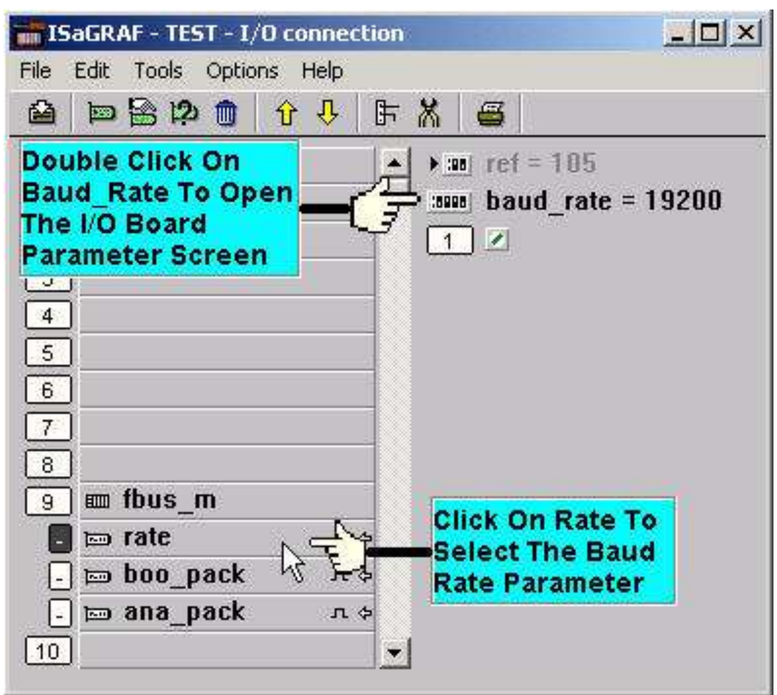
Example Prerequisites:
The SA controller system is the Fbus master controller and the SB and SC controllers are Fbus slave controllers. Each of the controllers has their baud rates set to 19200.



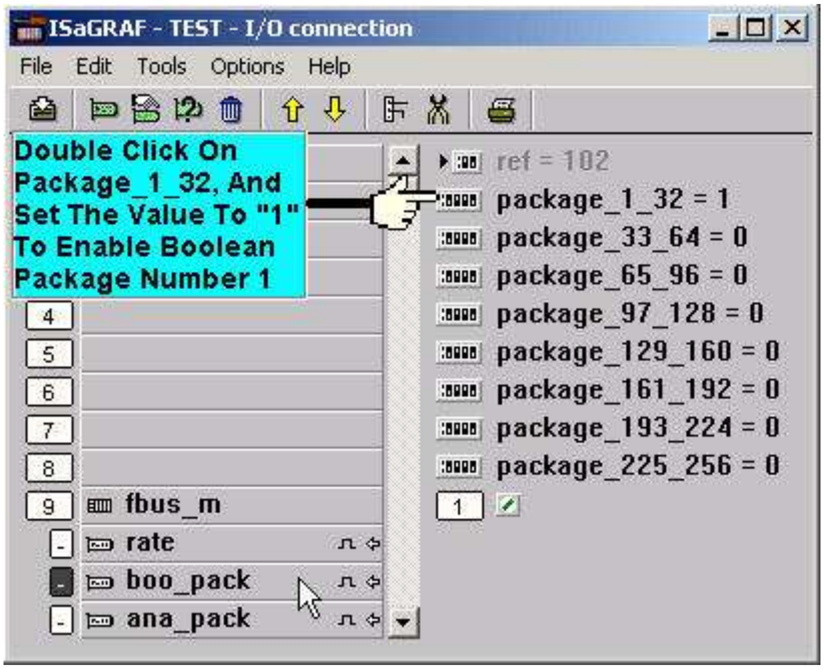
Setting The SB and SC Controllers As Fbus Slaves:
You should use the "ISaGRAF I/O Connections" window to declare the SB and SC controller systems as Fbus slaves.



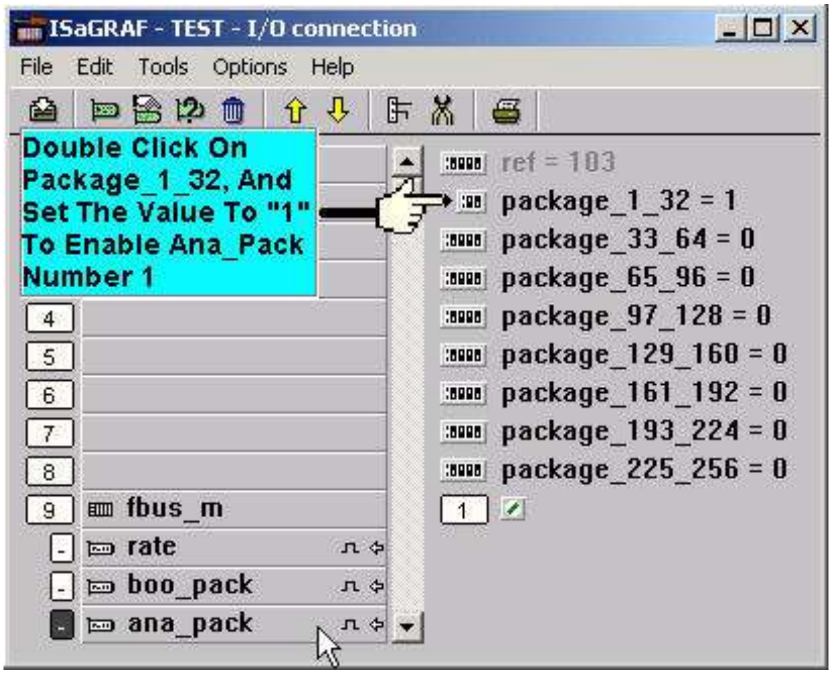
Setting The SA Controllers As Fbus Master:
Use the "ISaGRAF I/O Connections window to declare the SA controller system as the Fbus master controller.



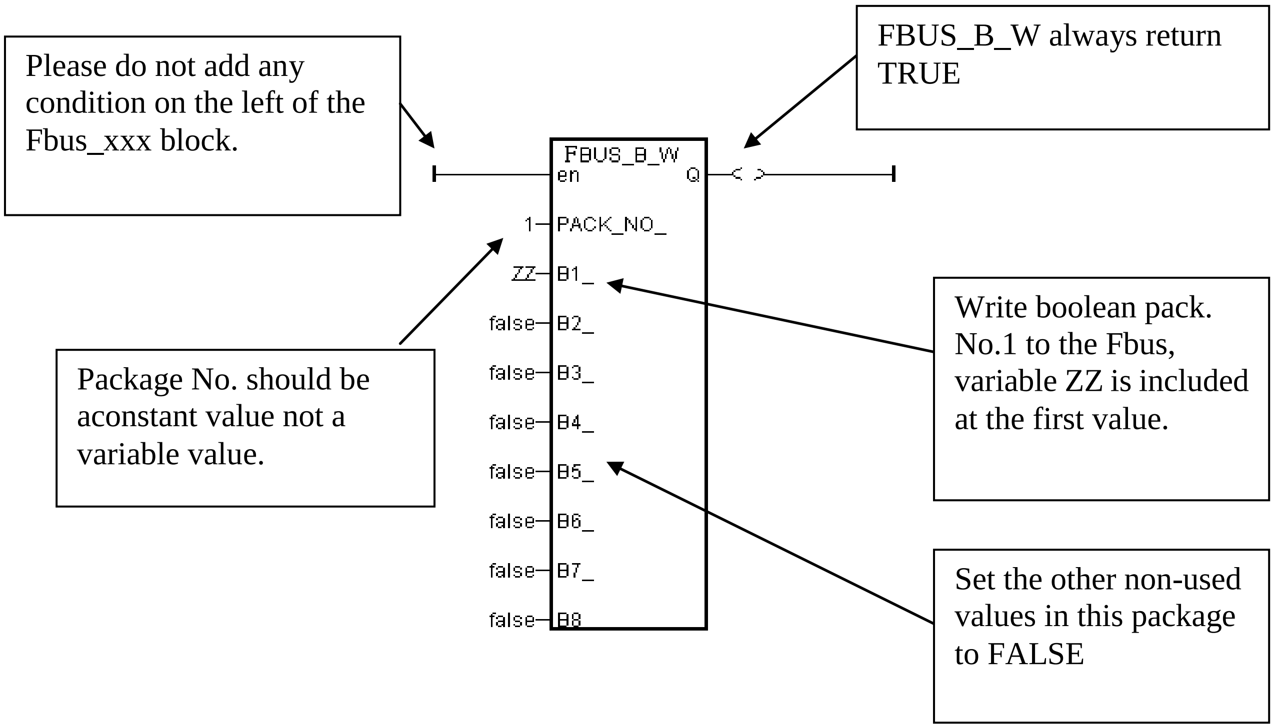
Additionally, enable the Boolean package for the SA controller. The Boolean package can be send/received through the Fbus network, only when the Boolean package number is “1”.



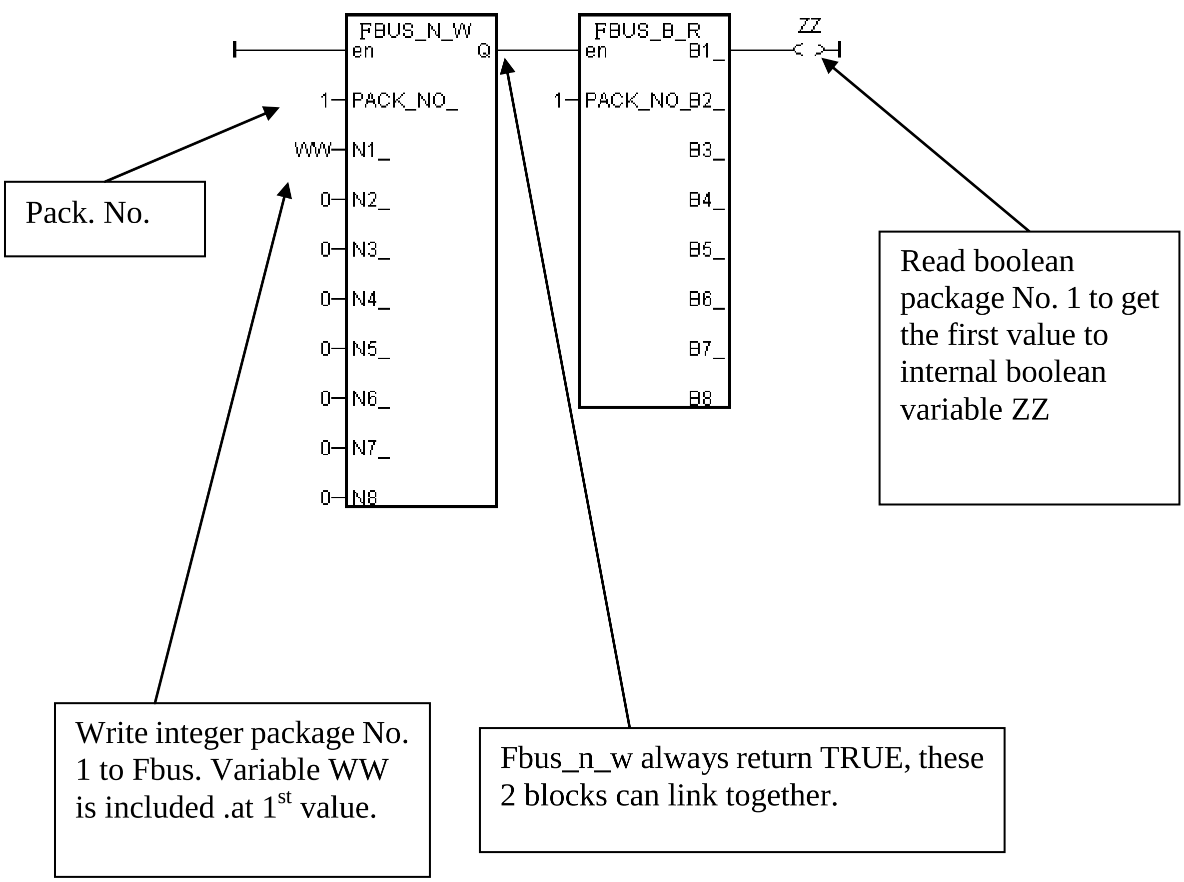
Also enable the integer package for the SA controller system. The Integer package can be send/received through the Fbus network, only when the Integer package number is “1”.



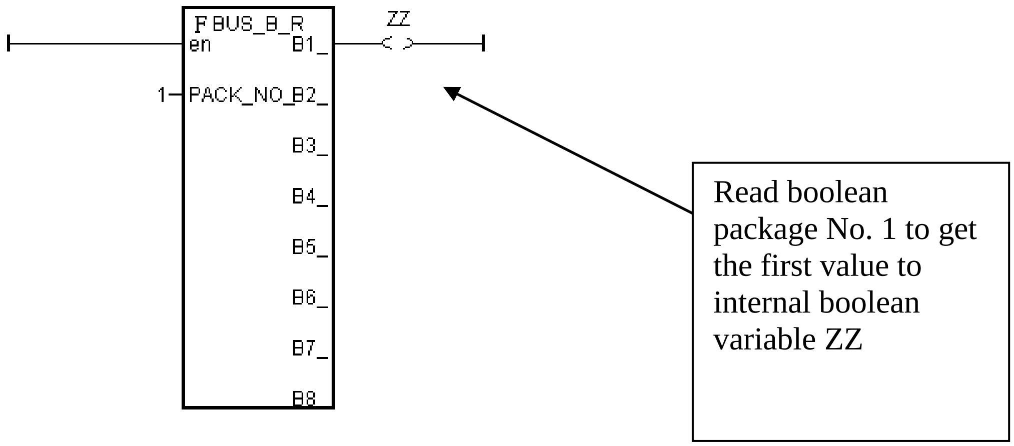
The ISaGRAF LD Project For The SA Controller:

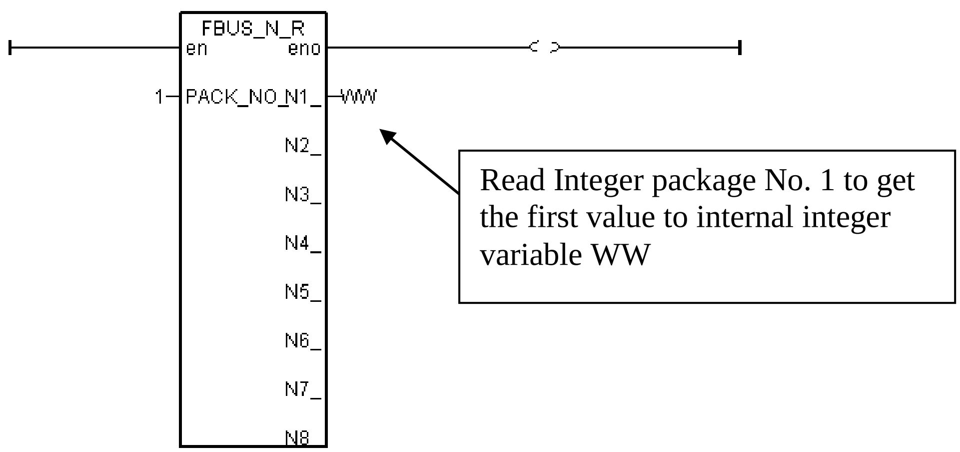


The ISaGRAF LD Project For The SB Controller:



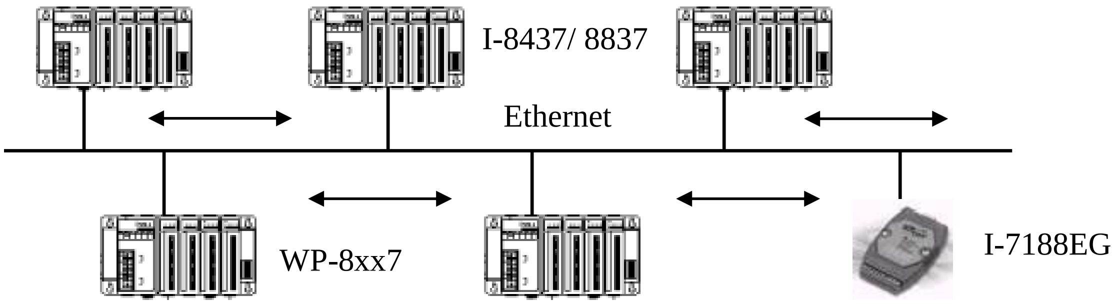
The ISaGRAF LD Project For The SC Controller:





7.5: Programming The Ebus

Ebus is a software mechanism which allows controllers to access data to each other through the ethernet port. Ebus is only working on the local area. That means exchanging data through a gateway is no possible.



The I-8437-80, I-8837-80 controllers support Ebus since its driver version of 2.15 and the I-7188EG support Ebus since its driver version of 1.08. And the µPAC-7186EG, µPAC-5xx7, iP-8x47, WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6, and VP-25W7/23W7 support Ebus.

To obtain the new released driver and update the I/O library from:
http://www.icpdas.com/products/PAC/i-8000/isagraf-link.htm
Please refer to Section 1.2 for the software installation.

Important Note:
1. **The max. boolean & integer package No. of Fbus & Ebus reduce from 256 to 128 since the driver version as below:** I-8xx7: 2.42 , I-7188EG: 1.32 , I-7188XG: 1.29 , iP-8xx7: 1.01 , µPAC-7186EG: 1.01, µPAC-5xx7: 1.01, VP-2117: 1.01
2. **If the controller is W-8x47/8x46, WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6 (Dual network port), please conntect Ebus at their "LAN2" port.**
3. **All Ebus Controllers's IP should be set in the same domain and their mask IP should be 255.255.255.0 . For example, (192.168.1.3), (192.168.1.5), (192.168.1.20).**

7.5.1: Basic Ebus Rules

The Ebus driver first creates a packet of eight Boolean values to form a "Boolean package", and then creates a packet of eight 32-bit integers to form an "integer package". Both of the "Boolean packages" and "integer packages" can be distributed on the Ebus to allow the data to be exchanged from one controller to another controller. You can exchanged the real data with "Int_Real" & "Real_Int" function (please refer to appendix A.4)

The basic Ebus rules are similiar as Fbus (refer to 7.1) as below.

RULE #1: Each Ebus network is identified with a "Group_No" ranging from 1 to 10. Data is only exchangable with controllers that are assigned with the same "Group No".

For example, there are 5 controllers located at the same local ethernet area, named A1, A2, A3, A4, and A5 respectively. A1, A2 & A3 are assigned with Ebus: Group_No = 1 while A4 & A5 are assigned with Ebus: Group_No = 2. Therefore, A1 can access data from A2 & A3 however can not access data from A4 & A5.

RULE #2: Each "Boolean package" in the same Ebus:Group_No must have an attached identification number ranging from 1 to 128. This means that there is a maximum of 128 "Boolean packages" that can be exchanged across an Ebus:Group_No connection.

Each "Boolean package" contains 8 Boolean values, and these Boolean values can only have the value of either "True" or "False". The Boolean values in the "Boolean package" can be assigned and exchanged with either "Internal", "Input", or "Output" Boolean variables or Boolean constants.

RULE #3: Each "integer package" in the same Ebus:Group_No must have an attached identification number ranging from 1 to 128. This means that there is a maximum of 128 "integer packages" that can be exchanged across an Ebus:Group_No connection.

Each "integer package" contains eight 32-bit integer values. The integer values can range from –2147483648 to 2147483647. The integer values in the "integer package" can be assigned and exchanged with either "Internal", "Input", or "Output" integer variables or integer constants.

Rule #4: Each number assigned to a "Boolean package" or an "integer package" can only be written to by one controller system across the same Ebus:Group_No network.

Each controller CANNOT **write** the same identification number for either a "Boolean package" or an "integer package" across the same Ebus:Group_No. WRITTING A PACKAGE IS NOT SHARED with the other controller across the same Ebus:Group_No network.

In this example, there are five controllers communicating through an Ebus:Group_No network, and the controllers are named S1, S2, S3, S4 and S5 respectively. If the S1 controller attempts to write a "Boolean package" with an ID of "1" and an "integer package" with an ID of "1" across the Ebus:Group_No, the other four controllers CANNOT write either a "Boolean package" or an "integer package" with the same number. However, the other controllers could write a "Boolean package" with an ID of "3" and an "integer package with an ID of "2".

There is no limitation on how many controllers can read the same number package across the same Ebus:Group_No network. Any of the S2, S3, S4 and S5 controllers can read the "Boolean package" with an ID of "1" and the "integer package" with an ID of "1" if desired.

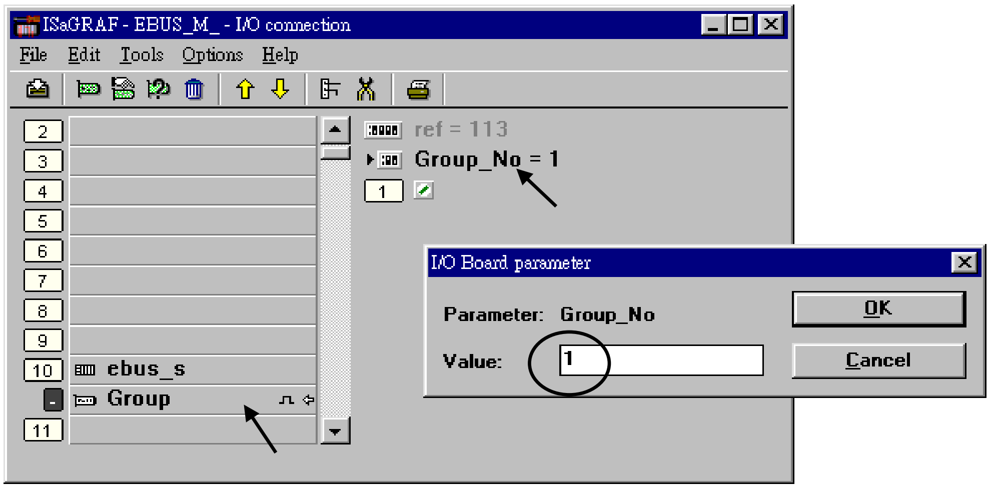
Rule #5: ONLY ONE controller in the same Group_No can be configured as a Ebus "Master", all the others controller in the same Group_No MUST be configured as a Ebus "Slave".

The "master" controller sends commands for how data is to be exchanged across the same Ebus:Group_No network. If you configure more than one controller as a "master", or configure none of the controllers as a "master", NO DATA CAN BE EXCHANGED across the Ebus:Group_No network.

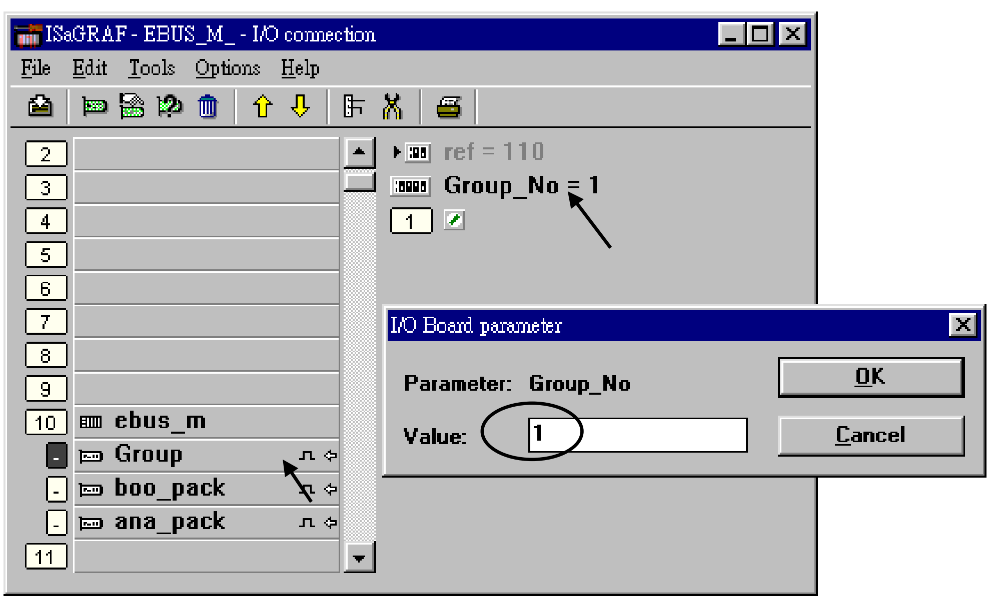
7.5.2: Configuring the ISaGRAF PAC To Be A Ebus "Master" Or "Slave"

To begin configuring the controller system as either a Ebus master or a slave, first open up the "ISaGRAF I/O Connections" window and double click on a slot number higher than 7. The "Select Board/Equipments" window will now open, click on "Equipments", and then double click on the "Ebus_s" selection to configure an Ebus slave, or double click on "Ebus_m" to configure an Ebus master. Remember, **ONLY ONE** controller system can be the Ebus master, and you **CANNOT** configure an controller to be both a master and a slave.

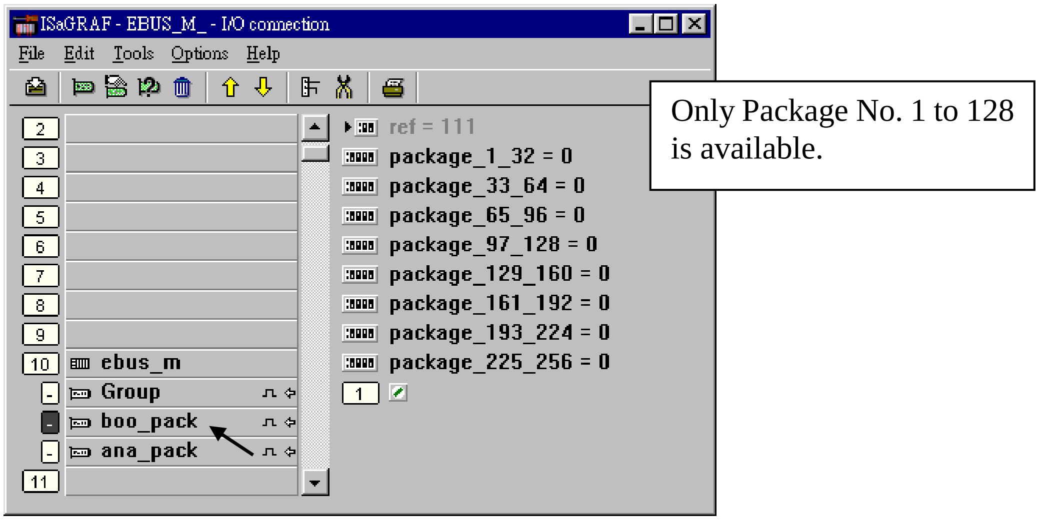
If you config a controller as an Ebus slave, only one parameter needs to be set, the “Group_No”. The valid value is ranging from 1 to 10. Set to other value will become a default value , 1.



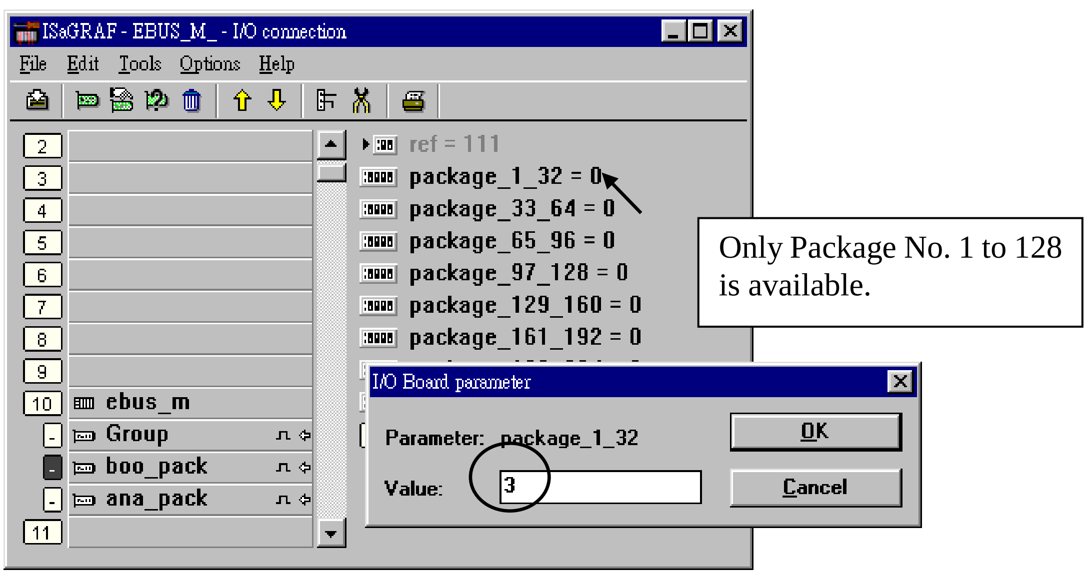
If you config a controller as an Ebus master, the parameter “Group_No” should be set to the same as the salve. The valid value is ranging from 1 to 10. Set to other value will become a default value , 1.

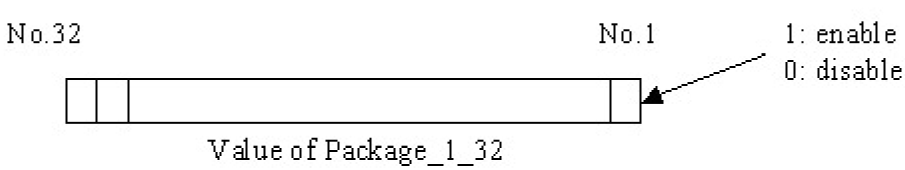


Configuring The Ebus Master Boolean Packages:
To begin configuring the Ebus Master Boolean Packages, click on the "boo_pack" selection from the "Ebus_m" I/O connection.

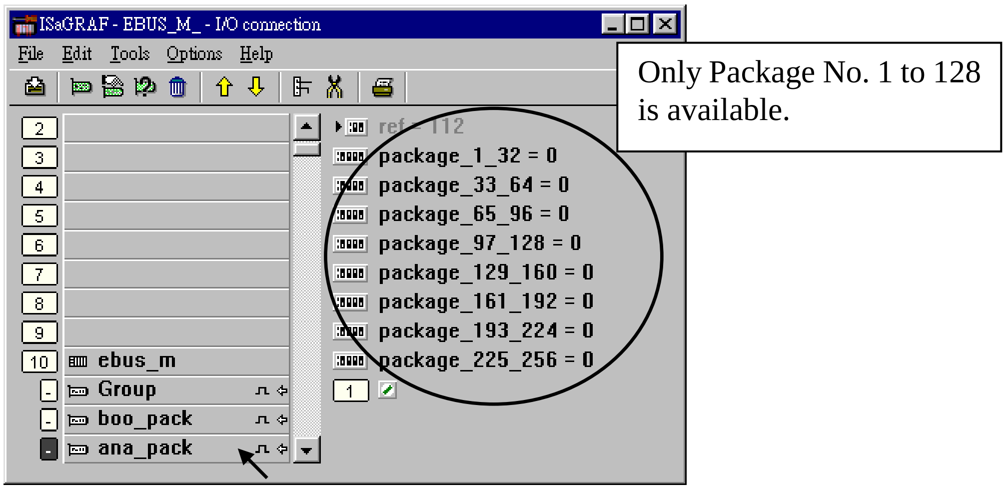


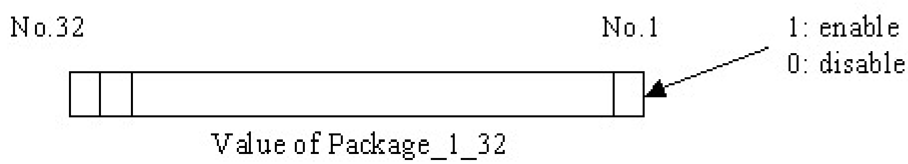
The parameter "package_xxx_xxx" at "Ebus_m: boo_pack" indicates the "Boolean package" number which is allowed to be written to or read from across the Ebus network. The parameter value is given as a 32-bit integer in **hexadecimal**. As an example, if the "package_1_32" is set to "FFFFFFFF" this will enable all the packages from number 1 to number 32 to be written to or read from across the Ebus network. If the "package _1_32" is set to a value of "A", this will only enable the number 2 and number 4 Boolean packages to be written to or read from across the Ebus network. The more packages that are enabled on a Ebus network the slower the communication efficiency will be. **With this in mind, always remember to enable only the required number of packages that you need for your application so you will have greater communication efficiency across the Ebus network**.





The parameter "package_xxx_xxx" at "Ebus_m: ana_pack" indicates the "integer package" number which will be written to and read from on the Ebus network. The "Ebus_m: ana_pack" is used to read and write 32-bit integer values across the Ebus network. Each of the parameter values is expressed as 32-bit integer values in **hexadecimal**, and the same configuration rules apply as those for the "Boolean package".





7.5.3: Programming Ebus Packages

Before you can exchange any data across a Ebus network, you must make sure that each controller is configured as either a Ebus master "ebus_m" or Ebus slave "ebus_s" (and remember, only ONE controller can be the master in the same Ebus “Group_No”). Refer to Section 7.5.2. The following Ebus function blocks can be used in a LD program to exchange data across an Ebus network.

Ebus_b_r	read one boolean package.
Ebus_b_w	write one boolean package.
Ebus_n_r	read one integer package.
Ebus_n_w	write one integer package.
Ebus_f_r	read one REAL package
Ebus_f_w	write one REAL package

(**The Integer package and REAL package use the same memory. Please DO NOT use the same package No. as Integer package and also as REAL package at the same time**. Or the local fault No. 115 may happen. Please refer to Chapter 10.6)

The below block is to get the communication ststus of each Boolean & Integer Package.

Ebus_sts	Get ststus of each Package.

Chapter 8. Linking The Modbus RTU / ASCII Devices

Note: ICP DAS ISaGRAF controllers support Modbus Master ports in different port No. as below table.

(To use COM5 to COM14 of the WP-8xx7, VP-25W7/23W7 and WinCon, please refer to Appendix E of the "Getting Started Manual" or visit below web site to download it.
http://www.icpdas.com/products/PAC/i-8000/getting_started_manual.htm)

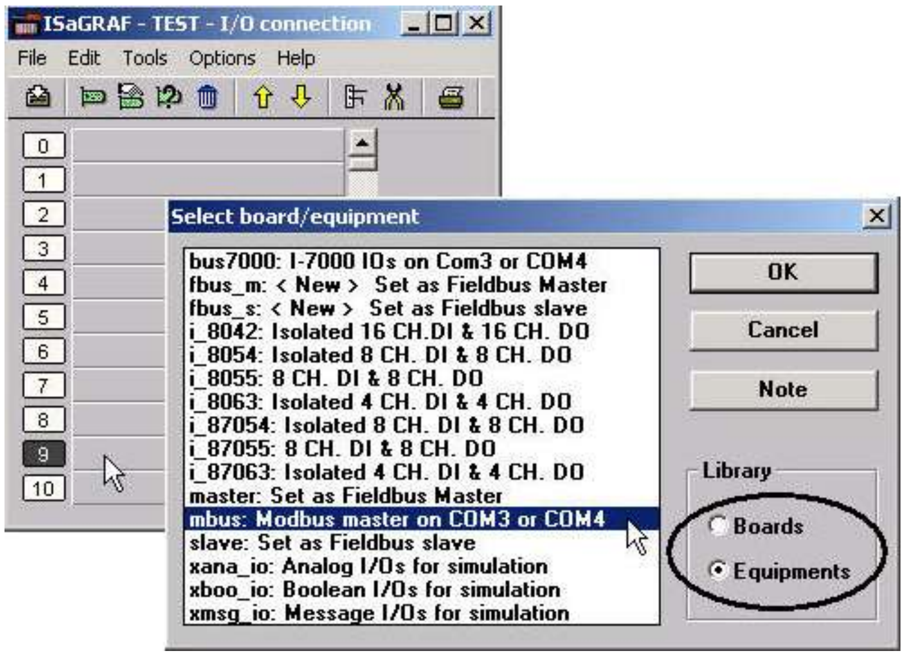
The Ethernet port of the WP-8xx7, WP-5xx7, VP-25W7, VP-23W7, XP-8xx7-Atom-CE6 and XP-8xx7-CE6 can support Modbus TCP Master, please refer to http://www.icpdas.com/faq/isagraf.htm > FAQ-113

If you can't connect the Modbus device properly, please refer to
http://www.icpdas.com/faq/isagraf.htm > FAQ-075 for troubleshooting (or specify the "Delat_Time)

(Table 1)	**µPAC-7186EG I-7188EG/XG**	**iP-8xx7 I-8xx7 I-8437/8837-80**	**WP-8xx7 VP-25W7/23W7**	**XP-8xx7-CE6 XP-8xx7-Atom-CE6**
Max. Modbus Master port	Max. 2 ports	Max. 2 ports	Max. 10 ports	Max.32 ports
Possible port No.	COM1 COM2 COM3 ("3" in X5xx or expansion board)	COM1 , COM3 , COM4 , COM5 ("5" in I-8142iW/8144iW expansion board)	COM2 , COM3 , COM5 ~ COM14 ("5 ~ 14" in I-8142iW/8144iW expansion board)	COM2 ~ COM33 ("6 ~ 33" in I-8142iW/8144iW expansion board)

8.1: Configuring The Controller To Be A Modbus Master

To begin configuring an ISaGRAF controller system to interface with a Modbus device, you must first configure the ISaGRAF program by linking the “Mbus” or “Mbus_asc” function to the ISaGRAF project. **"Mbus" means set it as Modbus RTU Master port; “Mbus_asc” means set it as Modbus ASCII Master port .** First, open the "ISaGRAF I/O Connections" window and double click on a slot number higher than 7 and the "Select Board/Equipments" window will open. From the "Library", click on the "Equipments" choice, and then click on the "Mbus: Modbus Master …" selection (choose “Mbus_asc” for ASCII), and then click on the "OK" to complete the installation. **For using multi-ports of Modbus Master needs to link more “Mbus” or “Mbus_asc” function and the “port_no” parameter for each function must be different. (It depends on model number, like table 1 in the former page)**



The description of "Mbus" and “Mbus_asc” Parameter:

"**port_no**": It defines which COM port the Modbus devices will communicate with the controller. The "port_no" parameter can be set as 1 to 33 (please refer to “Table 1” in the former page). If using multiple Modbus Master port at the same time, it needs to link more “Mbus” or “Mbus_asc” and the “port_no” parameter for each function must be different.

Delay time = port_no / 100;

Delay-time for sending the next Modbus frame, unit is ms, min. value is 1, max value is 1000 ms.
The Delay-time can be explained as "the time gap between two modbus commands" .
Some devices need larger delay-time. However setting a larger delay-time will slow down the scan rate of the Modbus communication.
Setting a value larger than 1000 will use the value of 1000 ms
Default "Delay Time" is 100 ms if setting "port_no" < 100 .
For example, if setting "port_no" as 3, it uses default delay-time value as 100 ms in COM3.
If setting "port_no" as 3206, it uses delay-time 32 ms in COM6.
If setting "port_no" as 40005, it uses delay-time 400 ms in COM5.

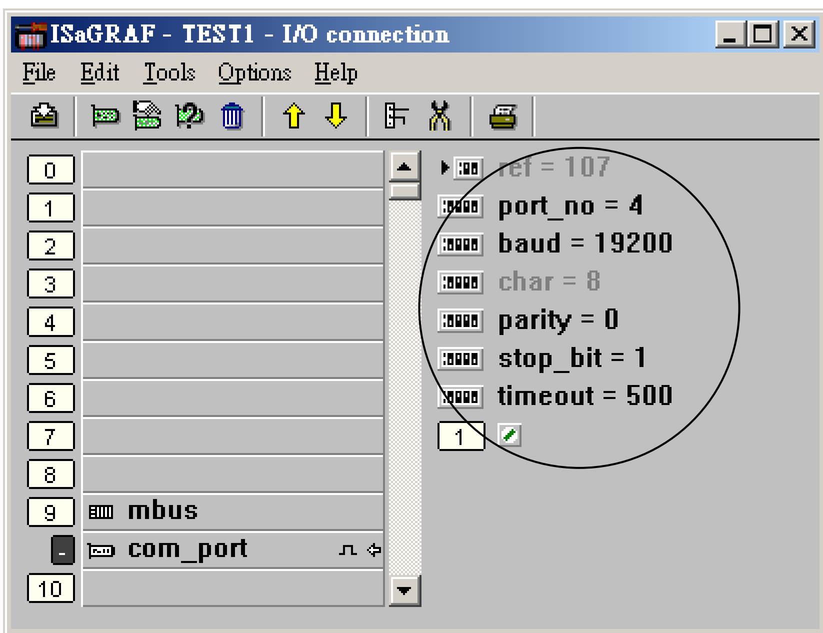
"**baud**": It defines what the communications baud rate setting will be. The "baud" can be set to 2400, 4800, 9600, 19200, 38400, 57600 or 115200 baud rate. All controllers on the same Modbus MUST be set to the same baud rate.

"**char**": The default value is 8 (it can't be changed when using "Mbus"). If using "Mbus_asc", you can set it as 7 or 8, and this value must be the same with the connected Modbus Slave device.

"**parity**": It defines what the communications parity setting will be. Setting the "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.

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

"**timeout**": It defines the allowed time to wait for response from remote device, unit is 0.001s (**The parameter is very important. The default value is 500 which means the Modbus device must reply within 0.5 second. If the response time of the connected Modbus RTU device is slowly, please set a larger "timeout" value.)** Set it as large value, it can communicate with the device which has a slower response time. But, when communication fails, the controller will wait a long time to send the next Modbus Master command. Recommend to set this value between 200 to 4000.



Note:
When setting COM1 of the I-8xx7, I-7188EG, µPAC-7186EG, iP-8xx7 to be a Modbus master port, please refer to each "Getting Started" Manual to "Setting COM1 As None-Modbus Port" to disable COM1:Modbus RTU port.

8.2: Programming A Modbus RTU Master

Before access the data from the Modbus device, you must link the “Mbus” or “Mbus_asc” function. Please refer to Section 8.1 for this procedure. Then, you can acess the data between the ISaGRAF PAC and other Modbus devices via Modbus protocol. The following function blocks can be used to pass data through the Modbus protocol in an LD or FBD program.

NOTE: Only the WP-8xx7, WP-5xx7, VP-25W7/23W7, XP-8xx7-Atom-CE6, XP-8xx7-CE6 supprt the Mbus24R, Mbus24R1, Mbus_XR, Mbus_XR1
(Refer to http://www.icpdas.com/faq/isagraf.htm > FAQ-101)

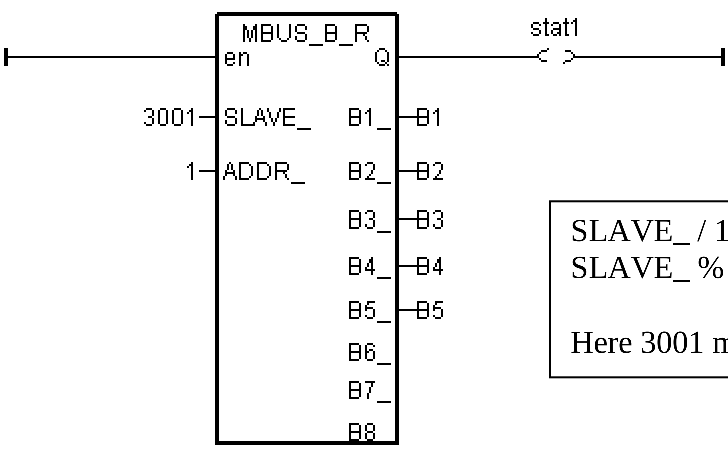
Mbus_R Mbus24R	When the “CODE_” is Modbus function 3 or 4 1. Read max. 12 word-value (Mbus24R means 24 word) and the value of each word is -32768 ~ +32767 2. Read max. six 32-bit integer-value (Mbus24R means 12 32-bit integer), -2,147,483, 648 ~ + 2,147,483,647, using “WD_LONG” to convert two word to one 32-bit integer. 3. Read max. 6 REAL-value (32-bit float), using “WD_LONG” to convert two word to one 32-bit integer, then use “INT_REAL” to convert it as 32-bit floating point value (Mbus24R means twelve 32-bit floating point value) if the “CODE_” is Modbus function 1 or 2. 4. Read max. 192 Boolean (Bit) value, using “WD_Bit” to convert one word to 16 Boolean value (Mbus24R means 384 Boolean)
Mbus_R1 Mbus24R1	Same as Mbus_R but with one extra setting – Period. Unit is second, can be set as 1~600. Read words or bits with a specified period time.
Mbus_XR Mbus_XR1	Read max. 120 word-value or 60 integers (32-bit) or 60 reals (32-bit). Refer to http://www.icpdas.com/faq/isagraf.htm > FAQ-101
Mbus_N_R	Read 8 word-value (-32768 ~ +32767) using Modbus function code 3. (It will ask eight Word for each Modbus command. If the device can’t support so many “word” or it just support Modbus function code 4, please instead to use “Mbus_R”.)
Mbus_NR1	Same as Mbus_N_R but with one extra setting – Period. Unit is second, can be set as 1~600. Read 8 words with a specified period time.
MBUS_B_R	Read 8 bit-value using Modbus function code 1. (It will ask eight bit for each Modbus command. If the device can’t support so many “bit” or it just support Modbus function code 2, please instead to use “Mbus_R”.)
MBUS_BR1	Same as Mbus_B_R but with one extra setting – Period. Unit is second, can be set as 1~600. Read 8 bits with a specified period time.
Mbus12W	Write 1 ~ 12 word (-32768 ~ +32767) to Modbus device. Refer to http://www.icpdas.com/faq/isagraf.htm > FAQ-144
MBUS_N_W	1. Write max. 4 word-value (-32768 ~ +32767) using Modbus function code 6 or 16 When “NUM_W” =1, using Modbus function code 6. When “NUM_W” = -1, using Modbus function code 16 to write one word. When “NUM_W” = 2 ~ 4, using Modbus function code 16. 2. Write 1 ~ 2 integers (32-bit) to Modbus device. Using “LONG_WD” block to convert one 32-bit integer to two word, and then put it into “MBUS_N_W”. Now, the “MBUS_N_W” must be set as 2 or 4.

	3. Write 1 ~ 2 floating point value (32-bit) to Modbus device. First, using “REAL_INT” to convert one 32-bit floating point value to one 32-bit integer, and use the “LONG_WD” block to convert it to two word , then put it into “MBUS_N_W”. Now, the “MBUS_N_W” must be set as 2 or 4.
MBUS_B_W	Write max. 4 bit-value using Modbus function code 5 or 15. When “NUM_W” =1, using Modbus function code 5. When “NUM_W” = 2 ~ 4, using Modbus function code 15.
MBUS_WB	Write max. 16 bit-value using Modbus function code 15

NOTE:
1. **The maximum number of each "Mbus_x_x" function block that with the same type can be used with one I-8xx7, I-8xx7-80, I-7188EG/XG, µPAC-7186EG, µPAC-5xx7, iP-8xx7, VP-2117 controller system is 64.**
2. **The maximum number of each "Mbus_x_x" function block that with the same type can be used for one port of WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6, VP-25W7 is 256.**
3. **“MBUS_R”, “MBUS_R1”, “MBUS24R”, “MBUS24R1”, “MBUS_N_R”and “MBUS_NR1” are with the same type.**
   “MBUS_B_R” and “MBUS_B_R1” are with the same type.
   “MBUS_B_W” and “MBUS_ WB” are with the same type.
   “MBUS_N_W” and “Mbus12W” are with the same type.
4. **Max. 128 “Mbus_XR” or “Mbus_XR1” can be used in one WP-8xx7, WP-5xx7, XP-8xx7-CE6, XP-8xx7-Atom-CE6 and VP-25W7. (It means that the amount of “Mbus_XR” plus “Mbus_XR1” can’t exceed 128)**

Modbus Example Function # 1: "**Mbus_b_r**"

The following example the "Mbus_b_r" function block is reading 8 bits from a slave Modbus device with a NET ID address of 1, with the Modbus address starting from 1 and using controller’s COM3 port. 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. If device is connected Ok, “stst1” will be TRUE.



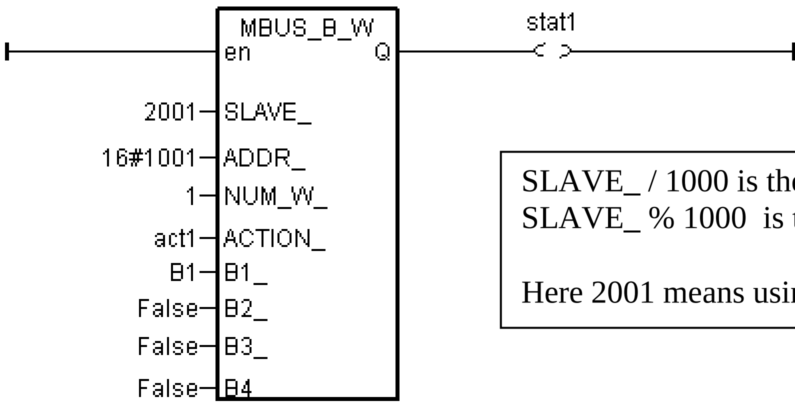
SLAVE_ / 1000 is the “COM Port No.”
SLAVE_ % 1000 is the “Net-ID of device”

Here 3001 means using COM3, device ID=1

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.

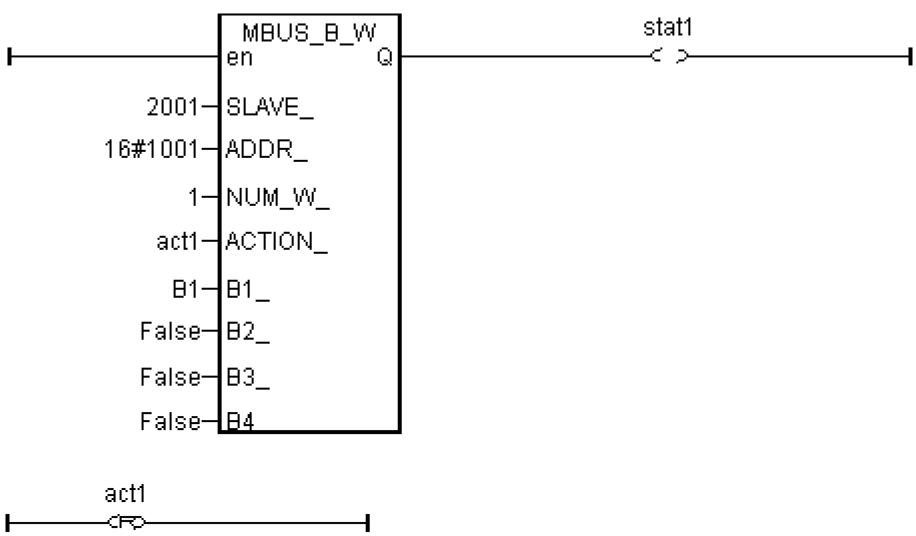
The value of "Stat1" is connected to the output coil and if the operation is successful "Stat1" will be true, otherwise the value of "Stat1" will be false.



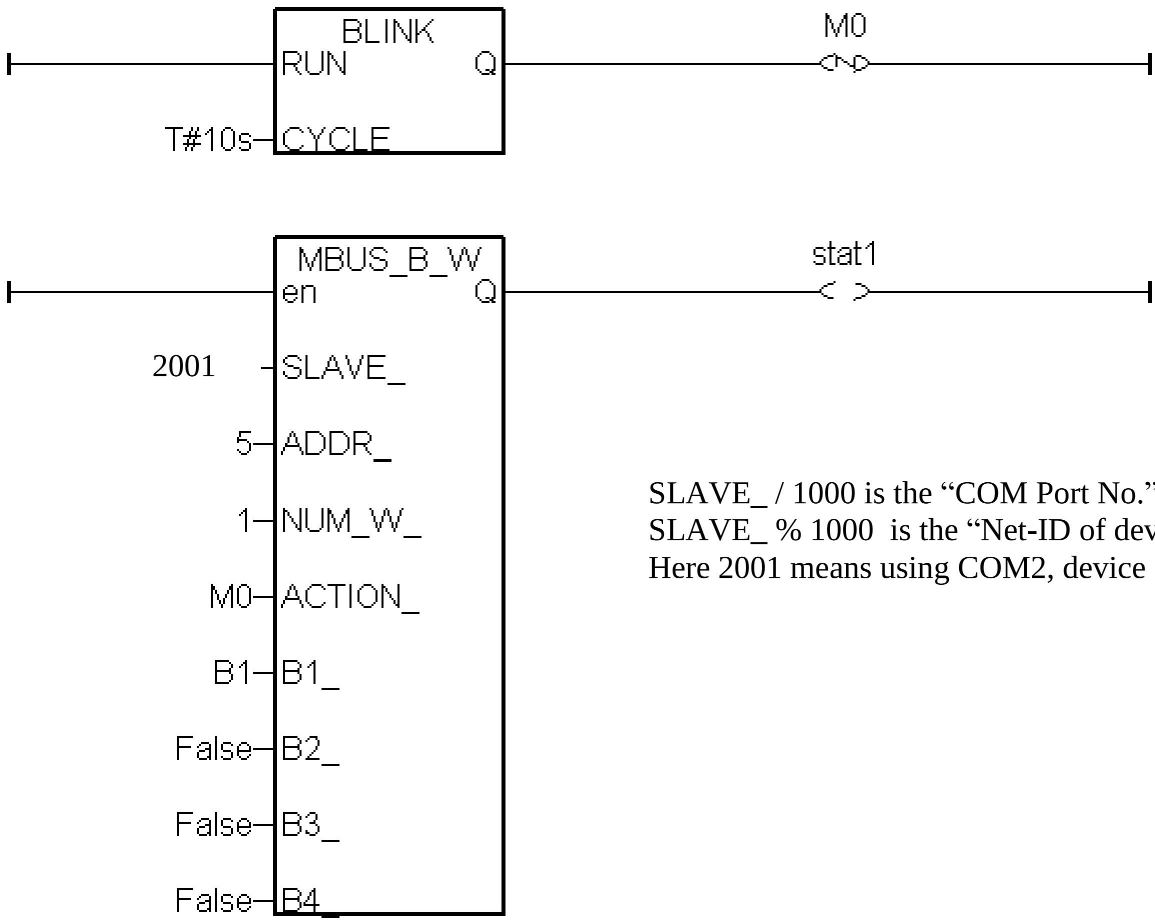
SLAVE_ / 1000 is the “COM Port No.”
SLAVE_ % 1000 is the “Net-ID of device”

Here 2001 means using COM2, device ID=1

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 “act1” is declared as an internal Boolean variable.
If set “act1” to TRUE, the below “MBUS_B_W” will write once and immediately reset “act1” to become False.



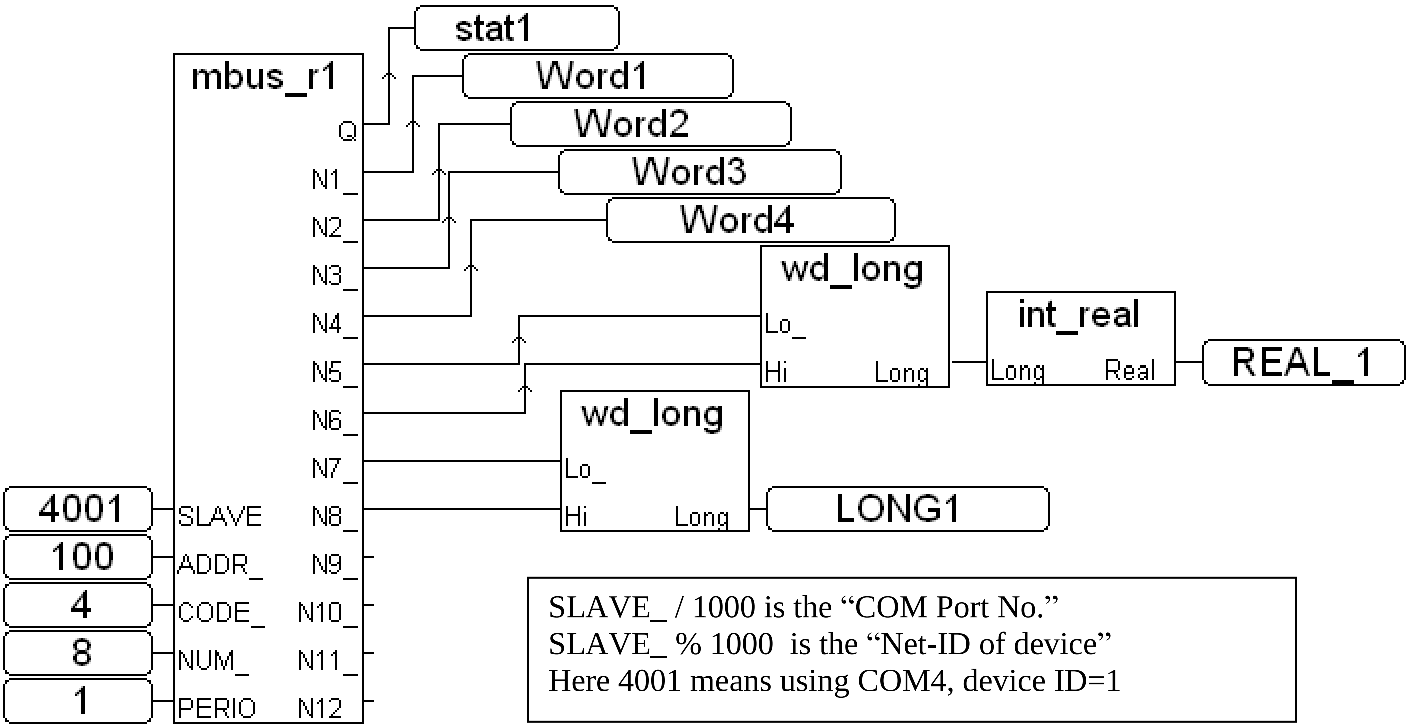
To write Modbus command periodic, user can use below similar code. (Here write once every 10 second)



SLAVE_ / 1000 is the “COM Port No.”
SLAVE_ % 1000 is the “Net-ID of device”
Here 2001 means using COM2, device ID=1

Modbus Example Function # 3: "Mbus_r1"

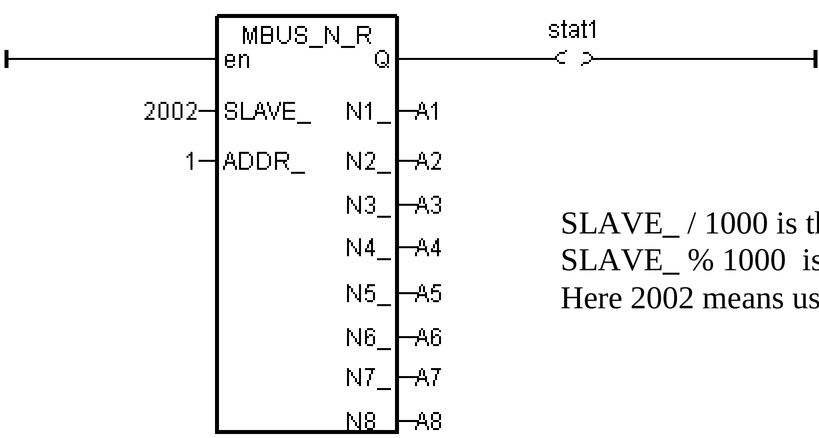
Below example use Modbus function code=4, device NET-ID=1 to read 8 word values starting at device's Modbus address No.=100 every second via controller 's COM4. The first four Word values are stored in "Word1" to "Word4" variables. The fifth and sixth word values are converted to become a REAL value stored in "REAL_1" variable. The 7th and 8th word values are converted to become a long integer value (32-bit signed integer) stored in "LONG1" variable.



Modbus Example Function # 4: "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 address 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.

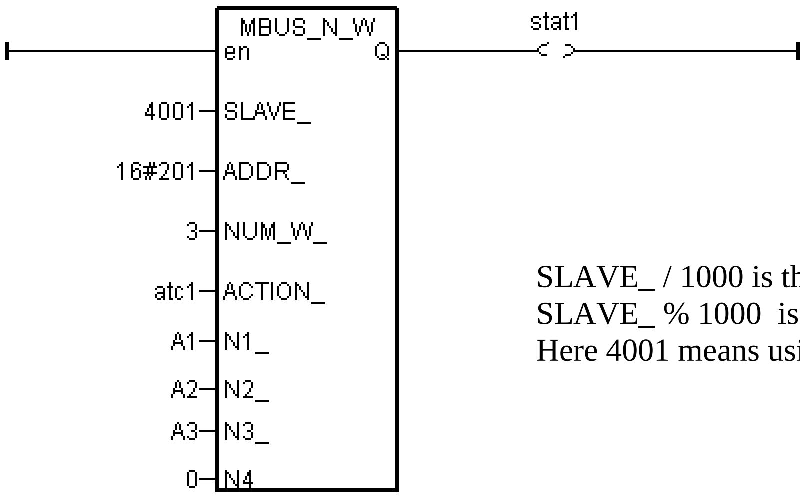
The value of "Stat1" is connected to the output coil and if the operation is successful "Stat1" will be true, otherwise the value of "Stat1" will be false.



SLAVE_ / 1000 is the "COM Port No."
SLAVE_ % 1000 is the "Net-ID of device"
Here 2002 means using COM2, device ID=2

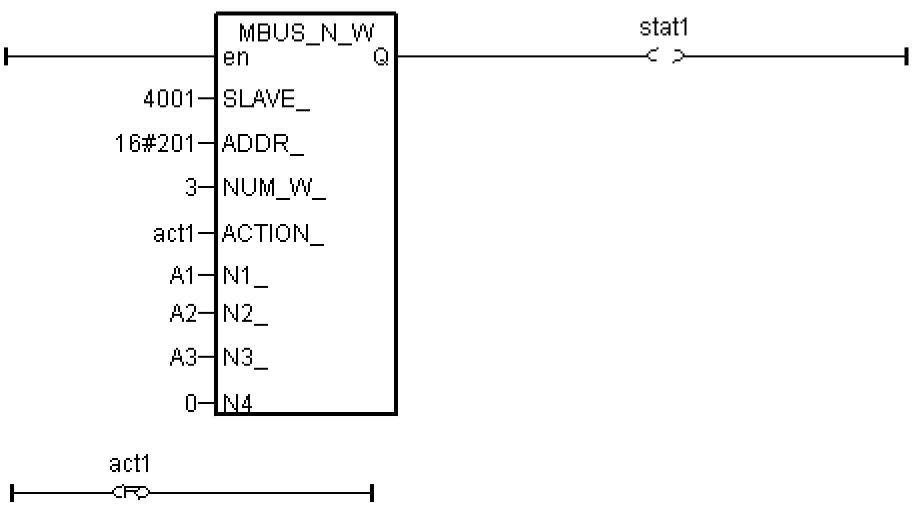
Modbus Example Function # 5: "Mbus_n_w"
Below example of the "Mbus_n_w" function block is writing three (3) words to a slave Modbus device with a NET ID address of 1, and the Modbus address is starting from 16#201 (513). The "Mbus_n_w" function will only write when the "ACTION_" line is true. In this 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.



SLAVE_ / 1000 is the "COM Port No."
SLAVE_ % 1000 is the "Net-ID of device"
Here 4001 means using COM4, device ID=1

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 "act1" is declared as an internal Boolean variable. If setting "act1" to TRUE, it writes only once.



More information about Modbus Master is at
www.icpdas.com – FAQ – Software –ISaGRAF – FAQ 144, 113, 101, 096, 075, 047, 027 , 028 , 045.

8.3: Linking The M-7000 I/O Modules

ICP DAS M-7000 series I/O modules support Modbus RS-485 RTU protocol. The ISaGRAF PAC can be configured as Modbus RTU Master to connect them.

For more information and demo program, please refer to Chapter 21 and
http://www.icpdas.com/products/Remote_IO/m-7000/m-7000_list.htm
(or http://www.icpdas.com/faq/isagraf_c.htm > FAQ-050)

Note: Each RS-485 port for ISaGRAF PAC can connect up to 32 M-7000 Modules. For more connection, it requires a RS-485 repeater (I-7510).

8.4: Linking The EKAN-Modview LED Display

The RS-485 port of EKAN-Modview LED display support Modbus RTU protocol. The ISaGRAF PAC can be configured as Modbus RTU Master to connect with them and to control the data you want to display.

For more information amd demo program, please refer to
www.icpdas.com – FAQ – Software – ISaGRAF – FAQ045 or
http://www.icpdas.com/products/HMI/led/ekan.htm

Chapter 9. Commonly Used ISaGRAF Utilities

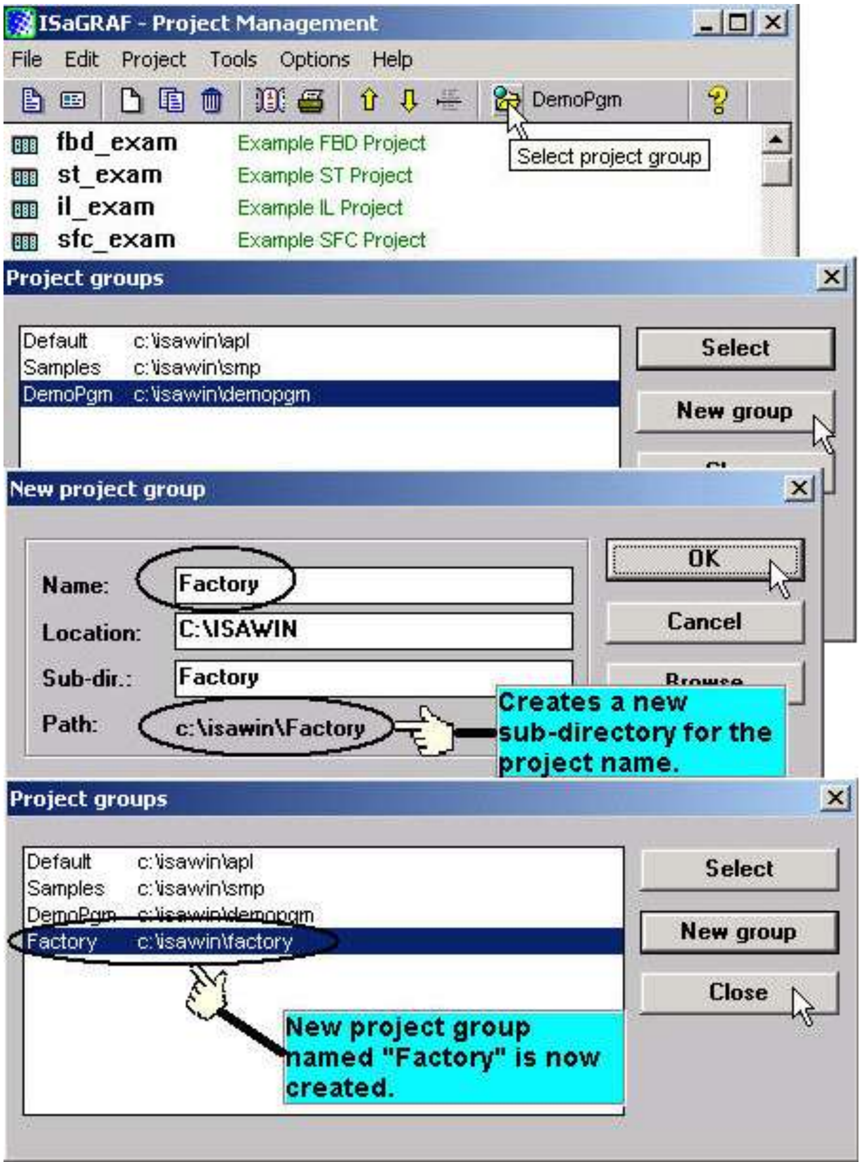
NOTE:
The I-8xx7 is the abbreviation for the I-8417, I-8437-80, I-8817 and I-8837-80 controllers.
The WP-8xx7 is the abbreviation for the WP-8147/8447/8847 and WP-8137/8437/8837 controllers.
The WP-5xx7 is the abbreviation for the WP-5147/5147-OD controllers.
The XP-8xx7-CE6 is the abbreviation for the XP-8047-CE6/ XP-8347-CE6/ XP-8747-CE6 controllers.
The XP-8xx7-Atom-CE6 is the abbreviation for the XP-8147-Atom -CE6/ XP-8347-Atom -CE6/ XP-8747-Atom CE6 controllers.
The iP-8xx7 is the abbreviation for the iP-8447/ iP-8847 controllers.

The following chapter describes many useful features and utilities of the ISaGRAF Workbench programming environment. These features and utilities make programming an ISaGRAF project quick and easy.

This chapter in no way contains all of the features and utilities available with the ISaGRAF Workbench program. For more details and information about all the features the ISaGRAF Workbench program has to offer consult 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".

9.1: Creating An ISaGRAF Project Groups

A very useful feature of the ISaGRAF program is the ability to organize numerous programs into "projects". The "Creating Projects" feature assists an ISaGRAF programmer who must create and maintain many different ISaGRAF programs for different application projects.

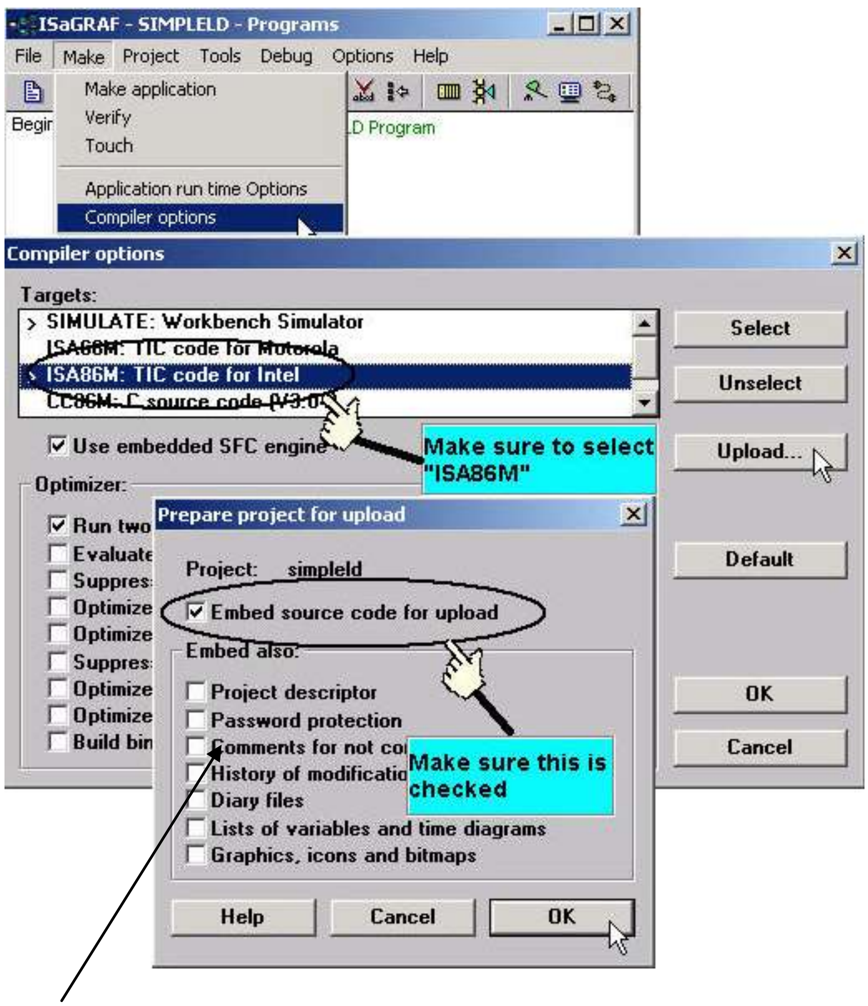


If you want to delete an existing project group, simply use the Windows Explorer to locate the ISaGRAF sub-directory you want to delete. An example of this is that if you wanted to delete the project just created, use the Windows Explorer and go to the C:\isawin\factory directory, and then just delete the "factory" sub-directory.

9.2: Uploading An ISaGRAF Project

The “Upload” functionality provided by ISaGRAF (Due to enable this function will make the original code **ONE and A HALF TO THREE** times larger, it isn’t recommended to use in a small-capacity controller, such as I-7188EG/XG and I-8xx7 controller system) can be used to upload the program that already running on your controller. Before uploading, you need to do some setup.

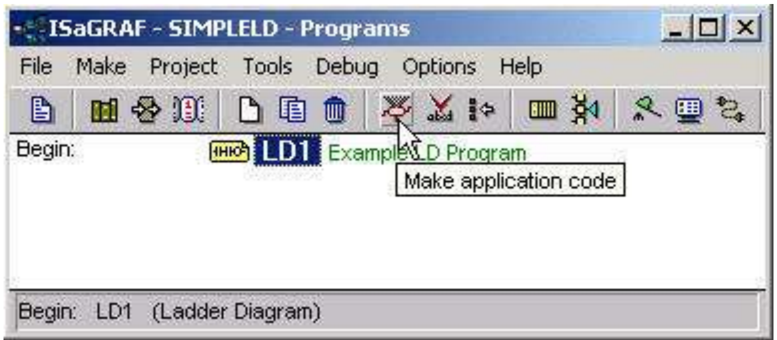
To turn the upload function on from the "Compiler Option", open the "ISaGRAF Programs" window, select "Make" from the menu bar, and then click on "Compiler Options". The "Compiler Options" window will open, make sure the "ISA86M: TIC Code For Intel" is selected, and then click on the "Upload" button. The "Prepare Project For Upload" window will open, click on the "Embed Source Code For Upload" checkbox and then click on the "OK" button.



VERY IMPORTANT NOTE:

Option “**Comments for not connected I/O channels**” must be choosed if “Directly represented variables” is used in this project (refer to section 3.4).

After you have checked the "Embed Source Code For Upload" checkbox and clicked on the "OK" button, you will need to recompile the project and download the project to the controller system.



IMPORTANT NOTE:

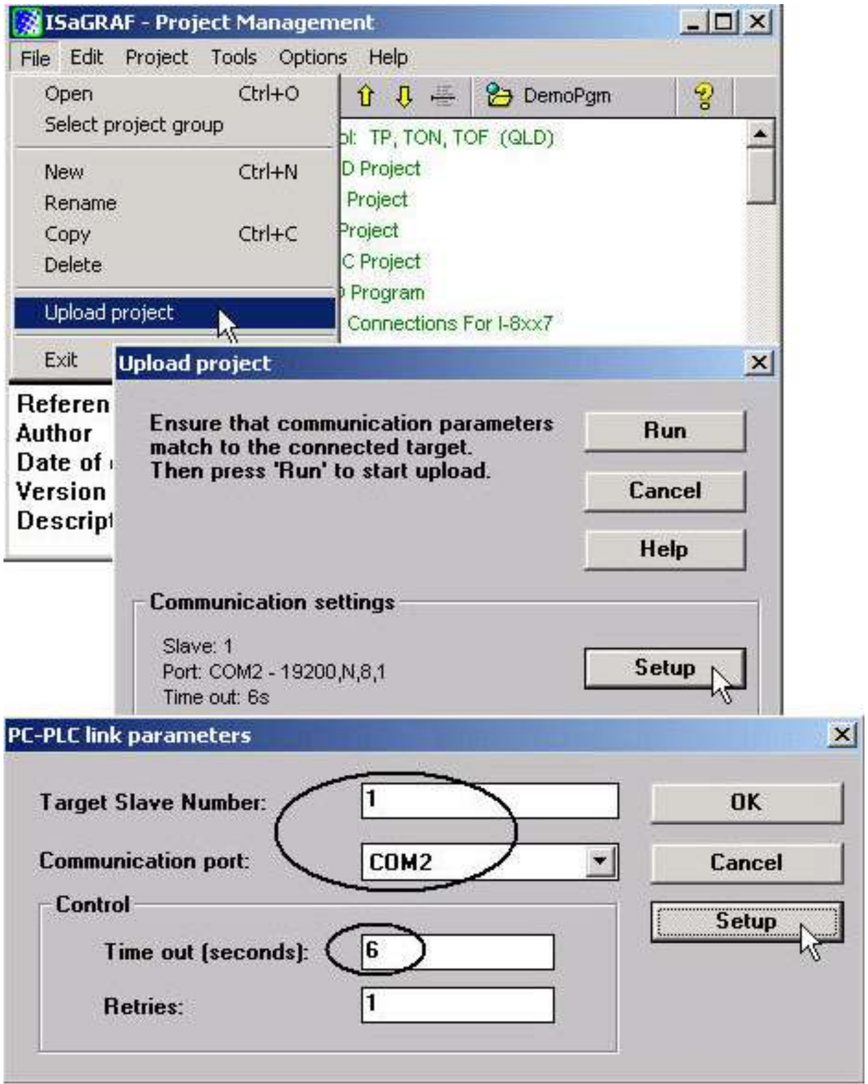
Once you have enabled the "Upload" option, the code generated by the compiler will larger than the original code that has not been set the “Upload” function. If the uploaded code size is larger than **64K** bytes, you will not be able to download the program to the I-8xx7, iP-8xx7, I-7188EG/XG, μPAC-5xx7, μPAC-7186EG and VP-2117 controller system. (The code size limitation is 512KB for W-8xx7 ; 1 MB for WP-8xx7, WP-5xx7, VP-25W7/23W7 ; 2 MB for XP-8xx7-CE6, XP-8xx7-Atom-CE6 controller system.)

Before trying to download the program it is advisable that you check the size of the uploaded program. Then, go to the appropriate sub-directory that the application program resides in. As an example, the "SIMPLELD" program that was create resides in the C:\isawin\demopgm\simpleld program sub-directory.

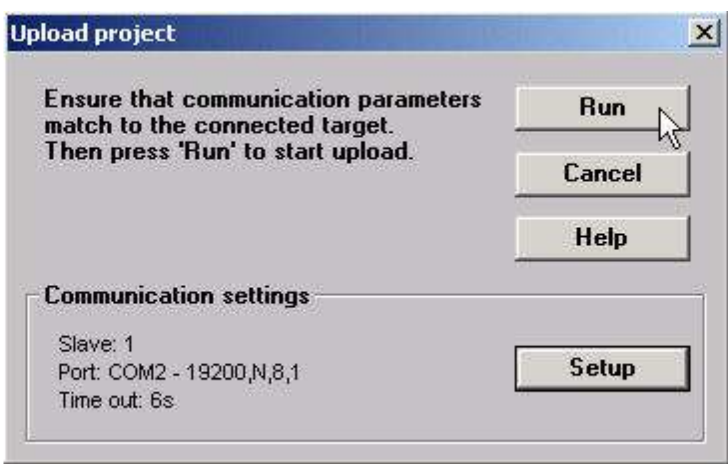
Remember, the "DEMOPGM" sub-directory is the **Project** group that the SIMPLELD program resides in, and the "SIMPLELD" sub-directory is where the actual application code files reside in. Look for the file named "**APPLI.X8M**" and check the size of this file. The "APPLIC.X8M" file is the file that contains the actual code that will be uploaded or downloaded to the controller system. Make sure the sizes of this file DOES NOT exceed 64K byte for I-8xx7 , iP-8xx7, I-7188EG/XG, μPAC-5xx7, μPAC-7186EG, VP-2117. (The code size limitation is 512 KB for W-8xx7 ; 1 MB for WP-8xx7, WP-5xx7, VP-25W7/23W7 ; 2 MB for XP-8xx7-CE6, XP-8xx7-Atom-CE6.)

UPLOADING AN ISaGRAF PROJECT

To upload an ISaGRAF project from a controller system, open the [File] → [Upload Project] in the "ISaGRAF Project Management" window, and check if the communication between your development PC and the controller system is working properly. If the communication is not in normal, please click on the "Setup" button to configure the proper communication settings.



Once you have made sure that the communication settings are properly configured, click on the "RUN" button in the "Upload Project" windows.

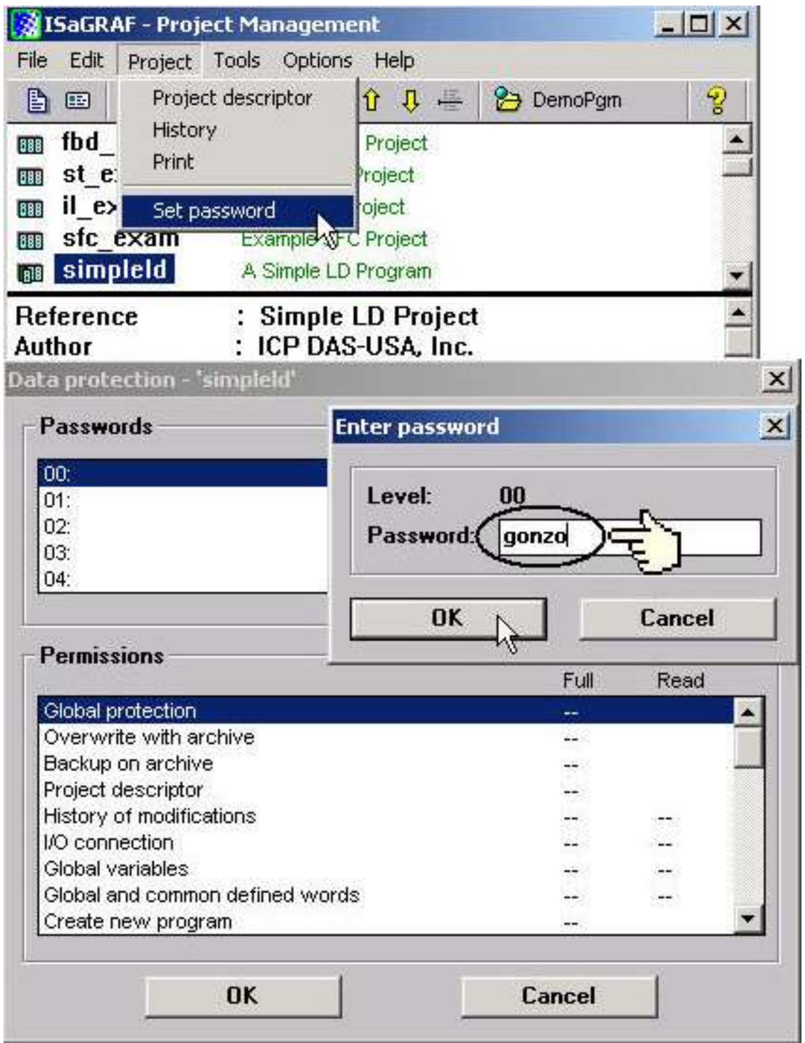


9.3: Setting An ISaGRAF Password

An ISaGRAF Workbench project can be password protected by configuring a user-defined password.

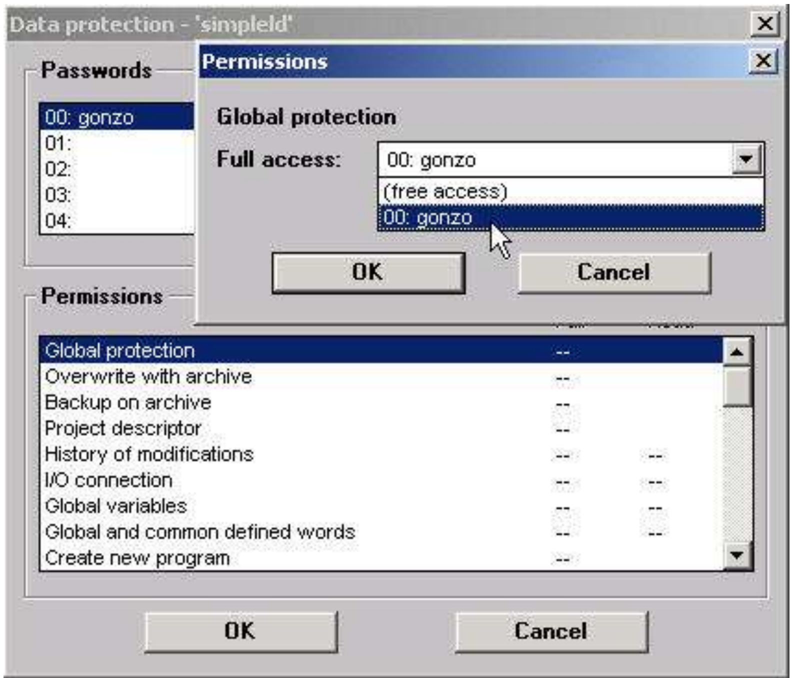
To configure an ISaGRAF password, open the "ISaGRAF Project Window", select "Project" from the menu bar, and then click on "Set Password". The "Data Protection" window will open and then select on of the passwords from "00 to 15" to configure a password (this means that up to 16 passwords can be assigned with the ISaGRAF Workbench program).

You will also need to select the type of data protection you are creating for your ISaGRAF project. In the example below we are defining the "Global Protection" for this ISaGRAF project.

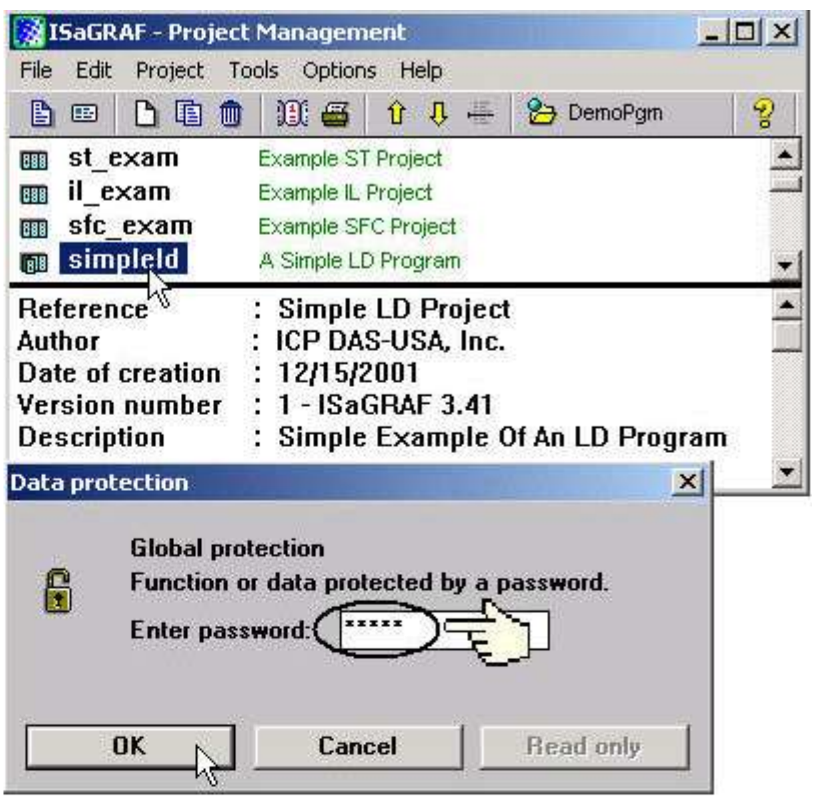


When you click on the "OK" button from the "Enter Password" window your new password will now be associated with the ISaGRAF project.

The next item you need to define is the type of data protection "Permissions" that will define for your ISaGRAF project. Double click on new password you have created and the "Data Protection Permissions" window will open. To allow full access WITH password protection, click on the "Full Access" scroll bar and click on the new password name you have created.

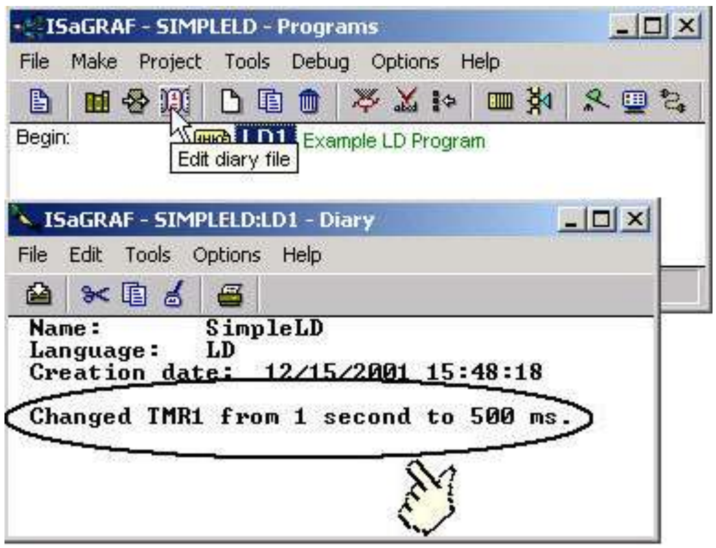


To verify that your password protection is now set for your ISaGRAF program, close all of ISaGRAF windows and then open the "ISaGRAF Project Management" window. Double click on the ISaGRAF program that you have created the password protection for. A "Data Protection" window will now open requiring you to enter the password for the ISaGRAF program you are attempting to open.



9.4: Creating An ISaGRAF Program Diary

When you modify an ISaGRAF program you can keep track of these revisions by entering a comment into the "Edit Diary" window. This affords the programmer the opportunity to add comments about program modifications and then save a record of these changes using the "Edit Diary" facility for enhanced program management capability.



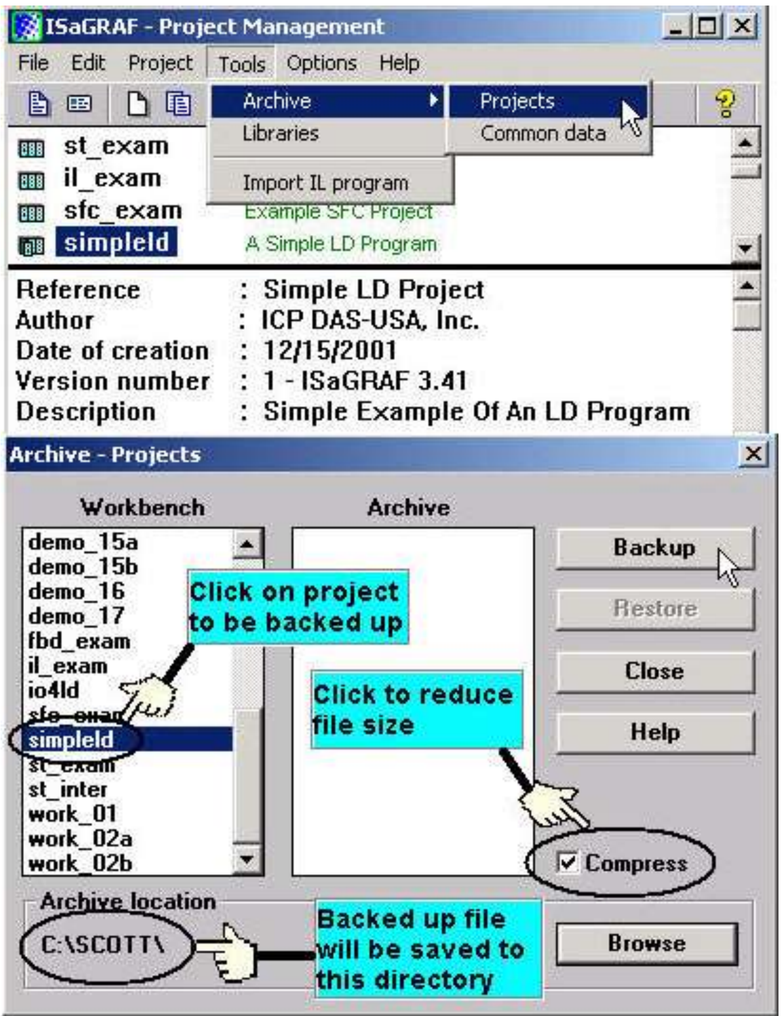
When you have completed entering information in the "ISaGRAF Diary" file, just click on the "Save" icon for your revision notes to be saved.

9.5: Backing Up & Restoring An ISaGRAF Project

For archiving purposes you can "Back Up" and "Restore" an ISaGRAF project. For example, you may want someone to test your program or email to service@icpdas.com for ICP DAS's ISaGRAF technical service.

Backing Up An ISaGRAF Project

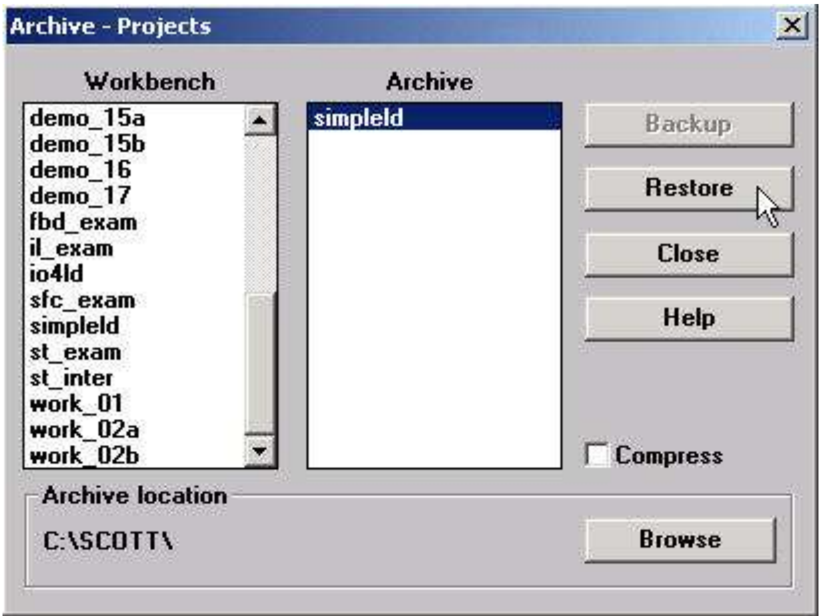
Open the "ISaGRAF Project Management" window, select "Tools" from the menu bar, click on "Archive", and then click on "Projects". An "Archive Projects" window will open which allows you to designate where you want to save the ISaGRAF project to. Click on the name of the ISaGRAF project you want to backup in the "Workbench" , and then click on the "Backup" button. You can compress the size of the file you have backed up by clicking on the "Compress" checkbox BEFORE you click on the "Backup" button.



You will now find the backed up ISaGRAF project file in the "Archive" location you have designated. In the example above, the name of the backed up file is "simpleld.pia".

Restoring An ISaGRAF Project

To restore an ISaGRAF project from a backed up file, use the same method as above to access the "Archive Projects" window, click on “Browse” to find the location of your backup ISaGRAF project, then select the project name you want to restore that listed in the “Archive” window, then click on the "Restore" button. The project will now be restored to the ISaGRAF.



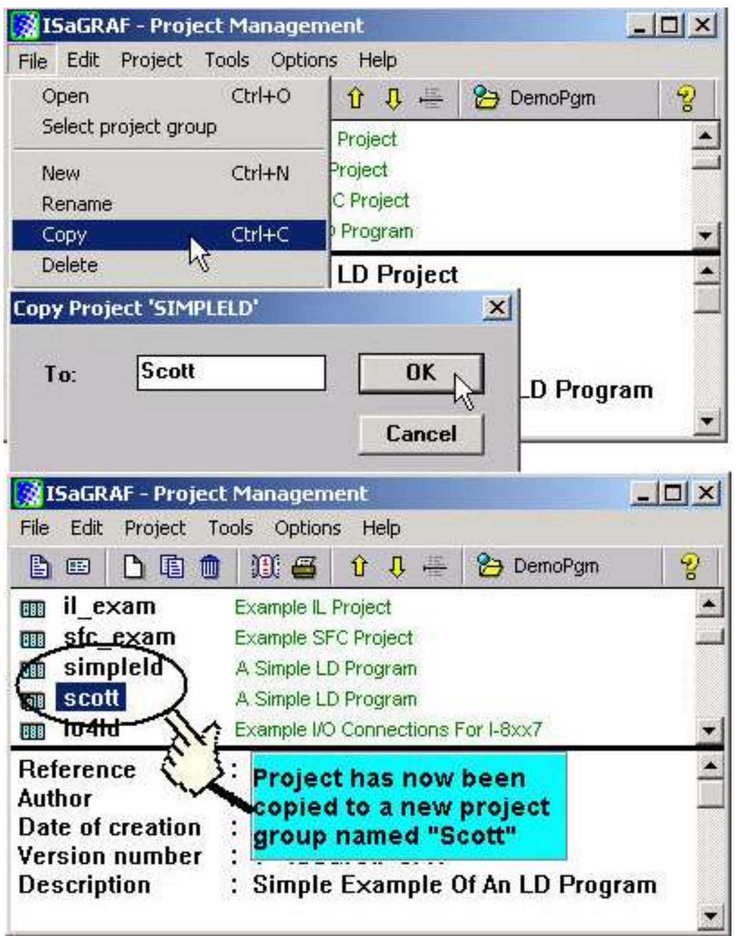
You can now open, edit and download the restored ISaGRAF project file.

9.6: Copying & Renaming An ISaGRAF Project

The ISaGRAF Workbench program has the capability of copying and renaming an ISaGRAF project or program. This is useful if you want to maintain a copy of an ISaGRAF project or program in a secondary directory.

Copying An ISaGRAF Program

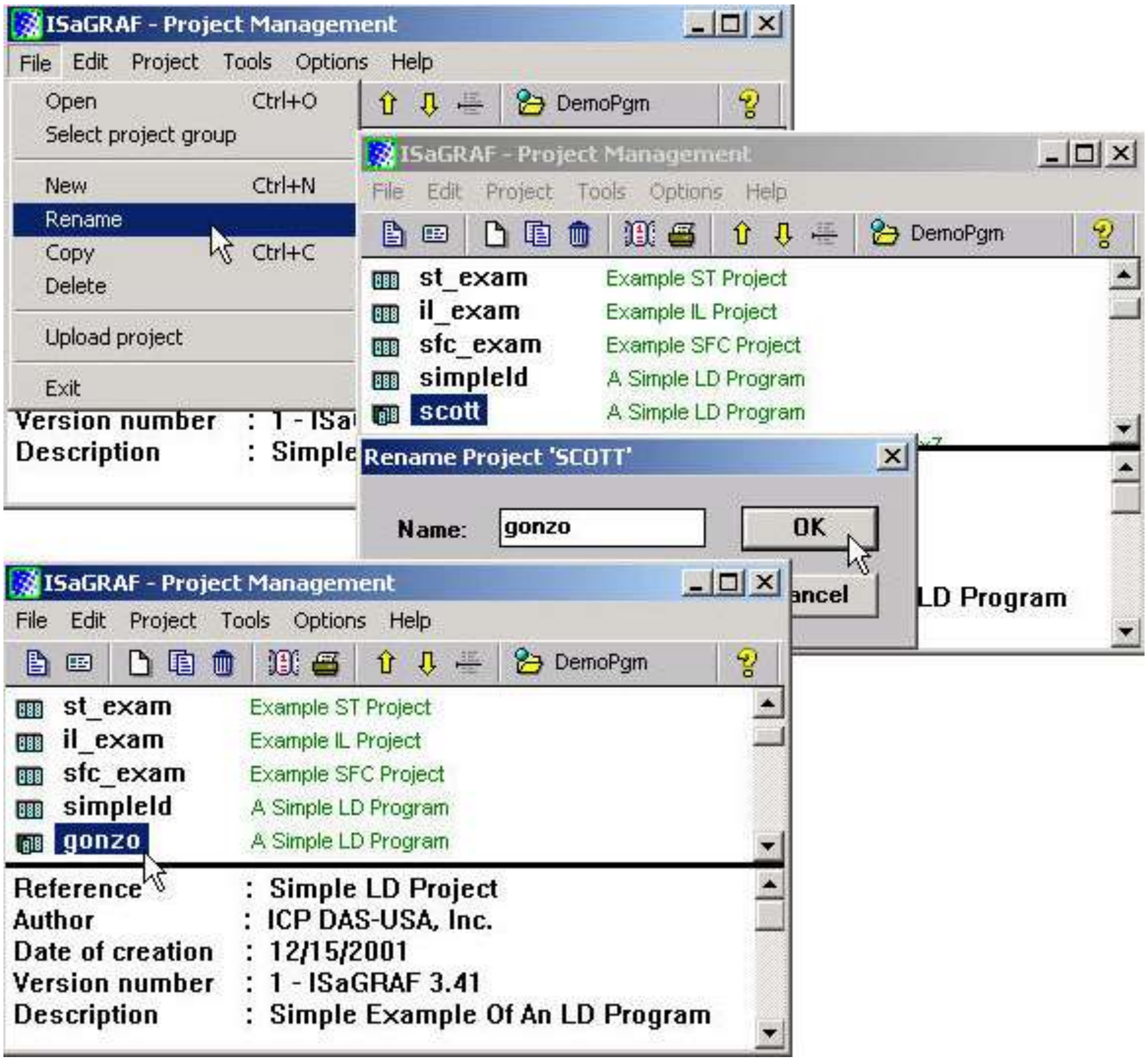
To copy an ISaGRAF program open the "ISaGRAF Project Management" window, first click on the name of the ISaGRAF program you want to copy, then select "File" from the menu bar, and then click on "Copy". When you click on "Copy" the "Copy Project" window will open, and now you can enter the name of the program you have selected to where you want to copy the program. If the new program name does not already exist, ISaGRAF will create the project name for you.



Note in the bottom screen that ISaGRAF has created a new program named "Scott" and placed a copy of all the files from "simpleld" into the "Scott" program group.

Renaming An ISaGRAF Program

To rename an ISaGRAF program open the "ISaGRAF Project Management" window, click on the name of the ISaGRAF program you want to rename, then select "File" from the menu bar, and then click on "Rename". When you click on "Rename" the "Rename Project" window will open, and now you can enter the new name for the ISaGRAF program.

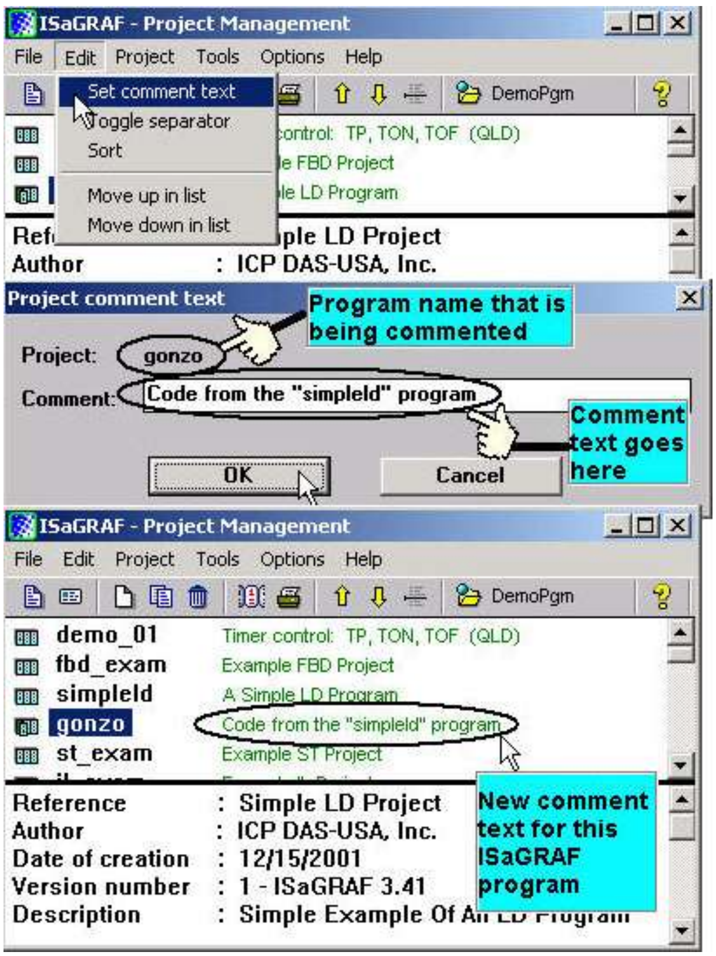


The former program named "scott" has now been changed to "gonzo", but it still has all the files from the "simpleld" program.

9.7: Setting Comment Text For An ISaGRAF Project

A useful feature of the ISaGRAF Workbench program is the ability to create "Comment Text" that will be placed next to an ISaGRAF program name in the "ISaGRAF Project Management" window. This way you can provide additional information about the purpose and any other additional comments regarding a particular ISaGRAF program.

To create "Comment Text" for an ISaGRAF program first open the "ISaGRAF Project Management" window, click on the name of the ISaGRAF program you want to create the comment text for, then select "Edit" from the menu bar, and then click on "Set Comment Text". When you click on "Set Comment Text" the "Project Comment Text" window will open, and now you can enter any comments and information you desire for the ISaGRAF program you have selected.

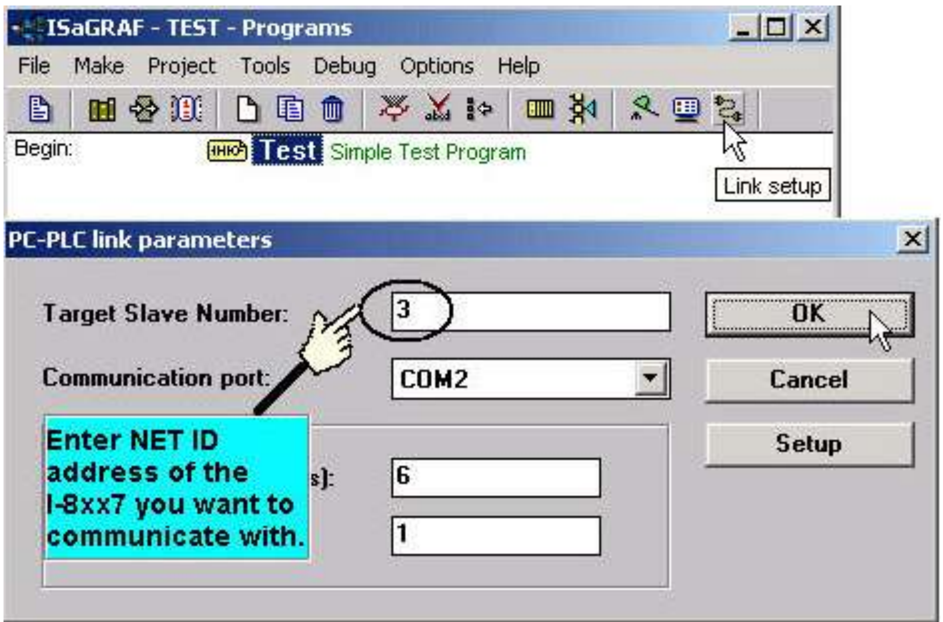


9.8: Setting The Slave ID For An ISaGRAF Controller

Each ISaGRAF controller system has a "NET ID" address that must be set to identify the controller to the ISaGRAF Workbench program. By default the NET ID address is "1" when it is shipped out.

If you need to communicate with multiple controller systems via RS-485 network, you must set the NET ID address in the ISaGRAF program for the specific controller system you want to communicate with. To communicate with different controller systems from one development PC open the "ISaGRAF Programs" window and click on the "Link Setup" icon.

When you click on the "Link Setup" icon, the "PC-PLC Link Parameters" window will open. Enter the "Target Slave Number" of the controller system you want to communicate with.

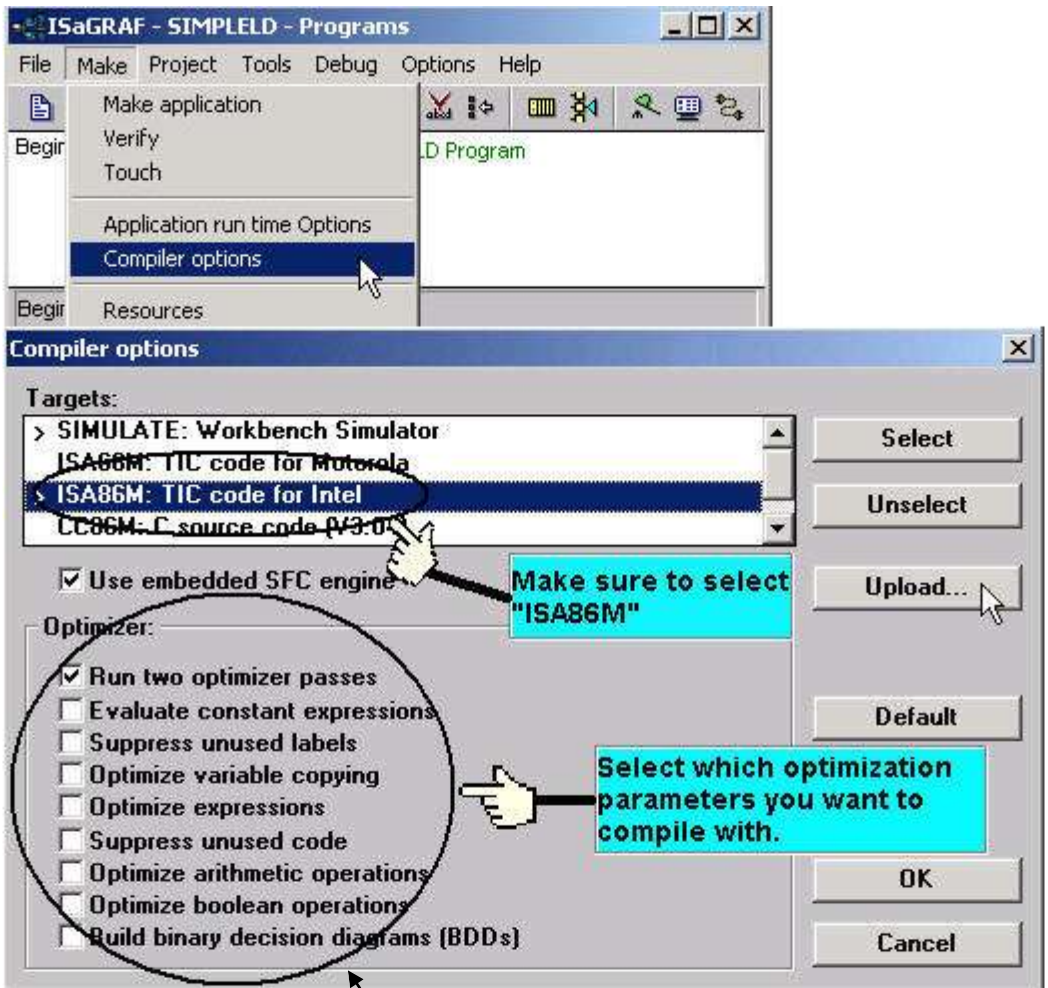


IMPORTANT NOTE

To set Net-ID for the controller, please refer to their respective “Getting Started Manual” delivered with the product.

9.9: Optimizing The ISaGRAF Code Compiler

The ISaGRAF Workbench program allows you to modify the settings for the "Compiler Options" to optimize the ISaGRAF program when you compile your project. To access the "Compiler Options" open the "ISaGRAF Programs" window and select "Make" on the menu bar, and then click on "Compiler Options". The "Compiler Options" window will open, and now you can select which optimization parameters you want for when you compile your ISaGRAF program.



If using "Variable Array" in the program, please DO NOT check the 2nd , 7th , 8th and 9 th Optimizer options, or the value of the Variable array will be incorrect. Recommend to check only the 1st – "Run two optimizer passes" option. (Please refer to Chapter 2.6)

Selecting the "Run Two Optimizer Passes" will insure that the code is compiled into the smallest possible program code.

9.10: Using The ISaGRAF Conversion Table

Note:

The conversion table is only for Input & Output attribution variables, not for internal variables. You may refer to Appendix A.4 for "**A4_20_to**" , "**To_A4_20**" to convert the analog value of 4 to 20 mA to application engineering value. Or "**V0_10_to**" , "**To_V0_10**" for converting analog value of 0 to 10 Volt to application engineering value.

Conversion Table Example

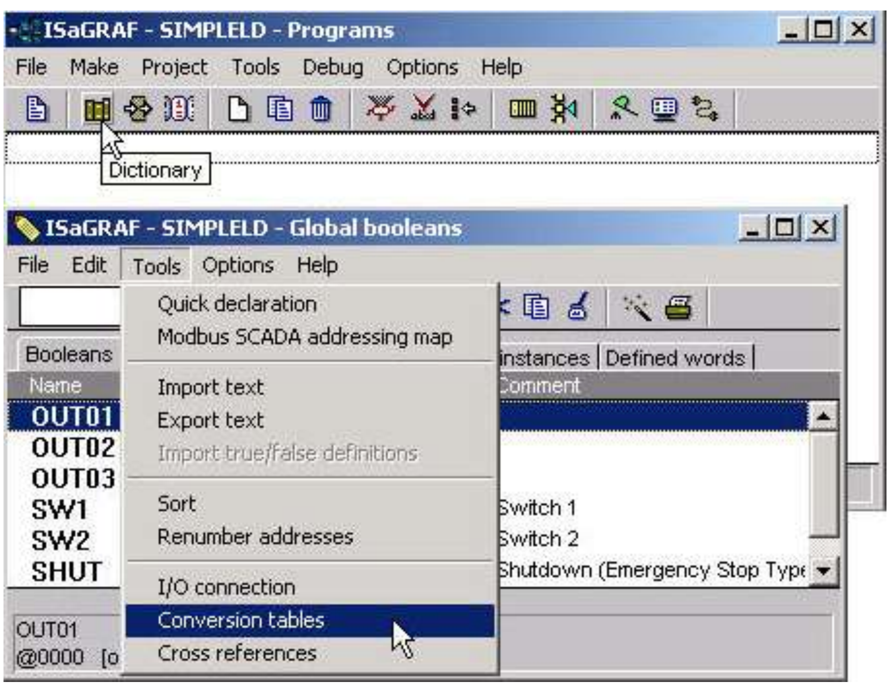
In this "Conversion Table" example the value from an I-87017 (an eight channel analog input module) board needs to be converted. The I-87017 is configured to receive a –10v to +10v signal, where –10v equals a value of "-32768", and a +10v signal equals a value of "+32767". You may refer to Appendix D to see the translation table of each analog board.

In this example we will use the "Conversion Table" to reconfigure the I-87017 so that a –10v signal will equal a value of "-10000" and a +10v signal will equal a value of "10000". In this example a value of +2.573v signal will equal a value of "2573".

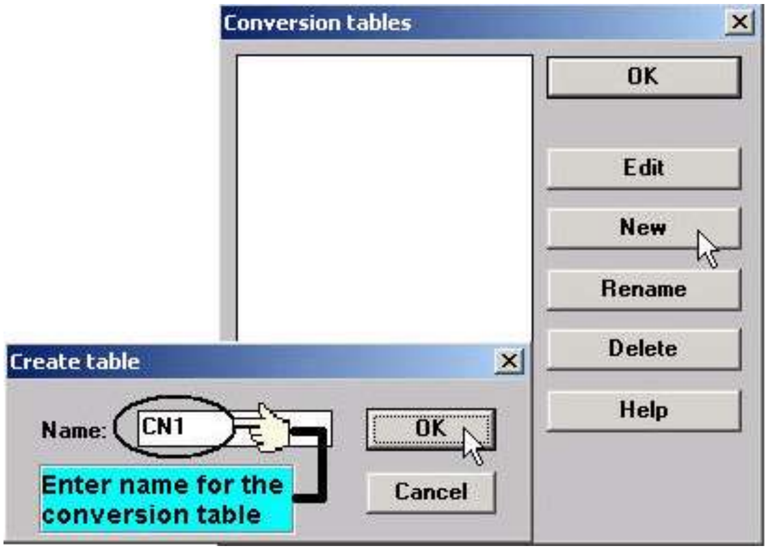
Note:

The ISaGRAF controller only supports the value before conversion within –32768 to +32767, and the value after conversion within **–10000 to +10000**. Setting conversion table out of these range may cause errors.

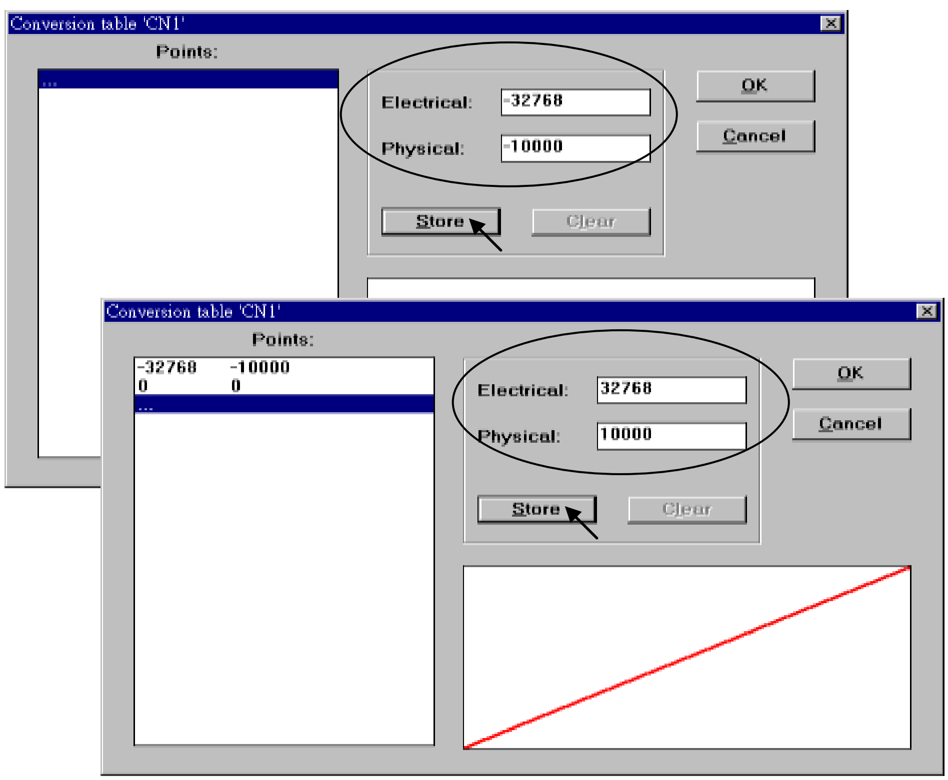
To configure a "Conversion Table" open the "ISaGRAF Programs" window and click on the "Dictionary" icon. This will open the "ISaGRAF Global Variables" window, select "Tools" from the menu bar, and then click on "Conversion Tables".



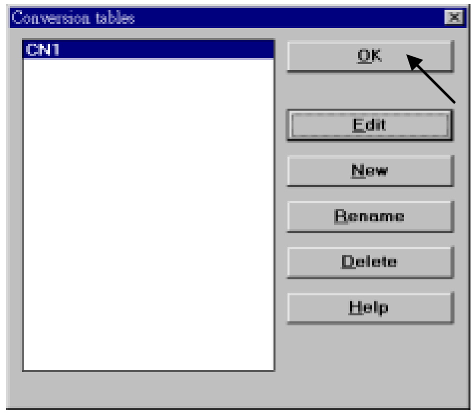
When you click on the "Conversion Tables" selection the "Conversion Tables" window will open. Next, click on the "New" button and then the "Create Table" window will now open. In the "Create Table" window enter the name for the conversion table you are creating.



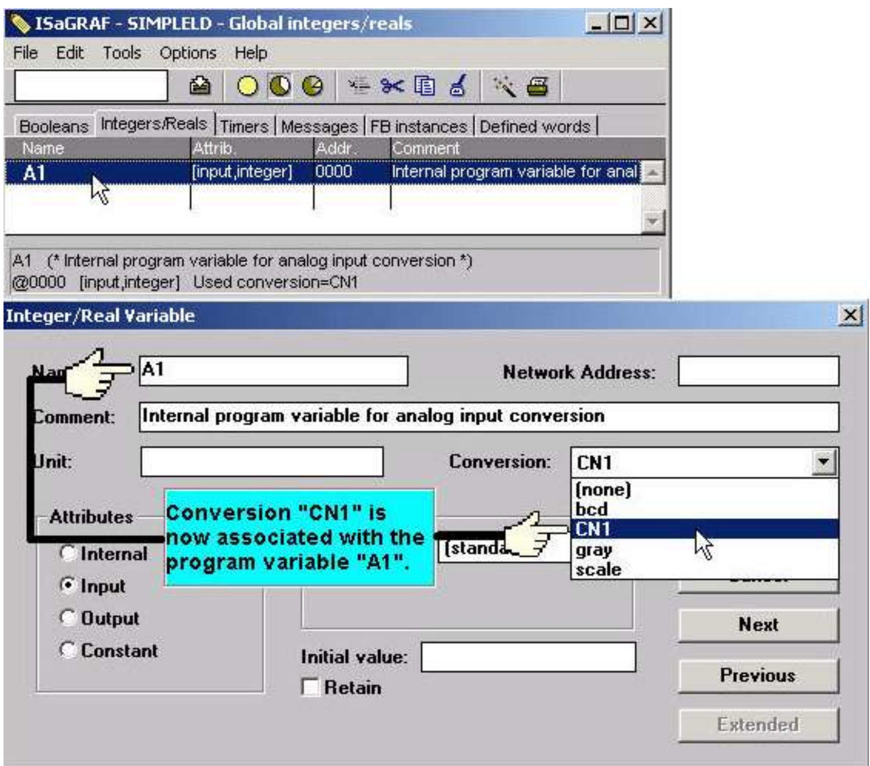
To properly create our example "Conversion Table" at least two values must be defined. The "Electrical" field means the original value BEFORE conversion and the "Physical" field is for the value AFTER conversion. The two points defined in this example are (-32768, -10000 "lower limit") and (+32767, 10000 "upper limit"). Click on the "STORE" button to save each entry.



When you have completed entering in the two value points, click on the "OK" button to save the entered values.



The last step is to assign the conversion table "CN1" to a program variable that will be used in an ISaGRAF program.

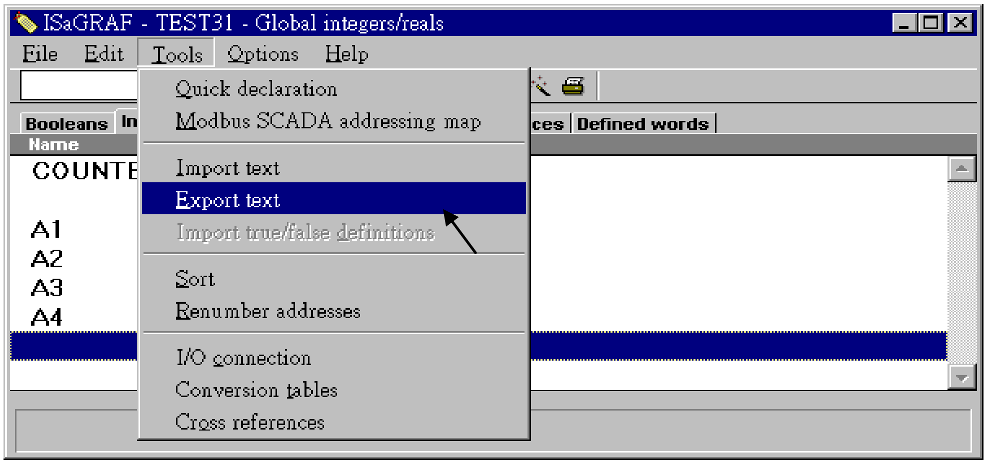


Note:
Only integer variable declared as input or output attribution can be assigned a conversion table. The user can use “Bin2Eng” to convert the internal variables (please refer to A.4).

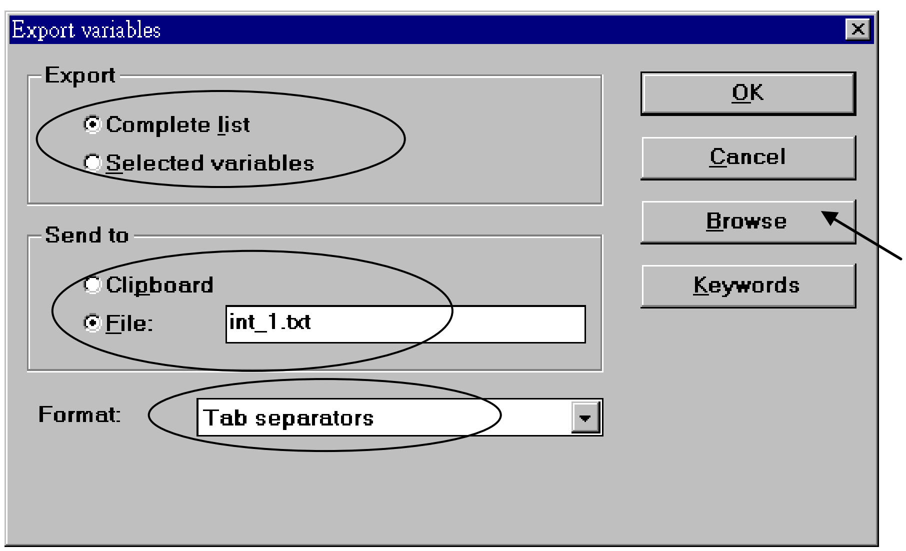
9.11: Export / Import Variable Declarations Via Microsoft Excel

Variables can be defined in Microsoft Excel and then be imported to ISaGRAF workbench. And also they can be exported from ISaGRAF to Excel.

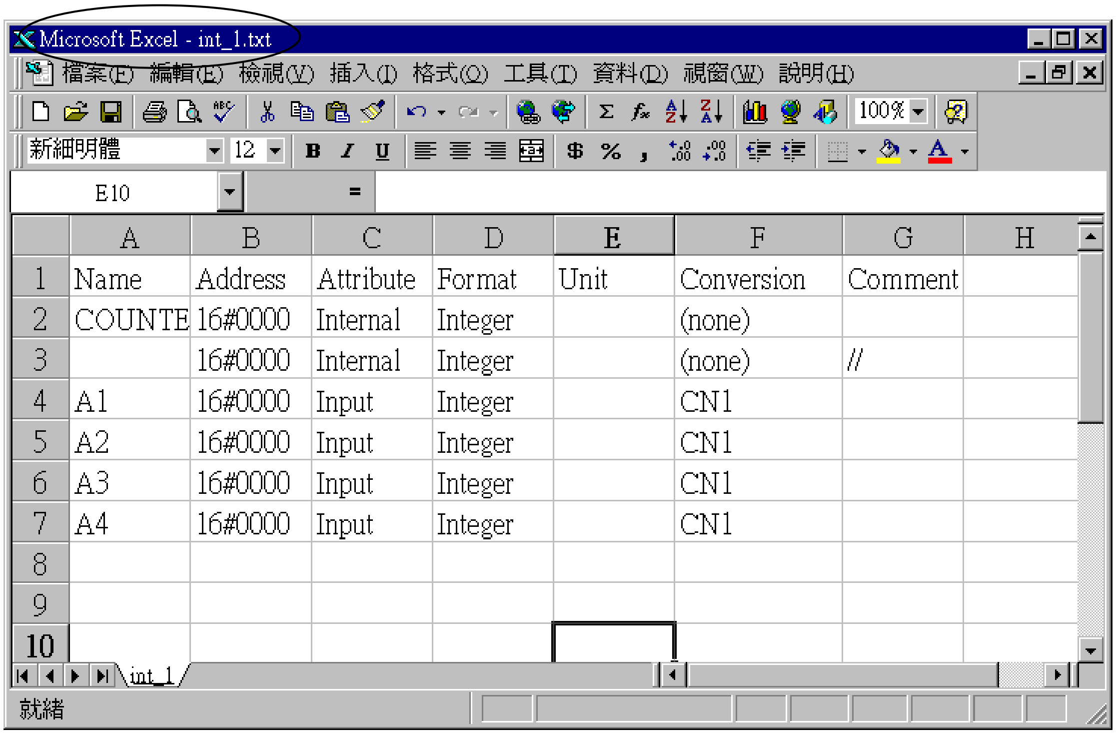
To export to a text file, with an extension name "**.txt**", run "Tools" - "Export text" from the "dictionary" window.



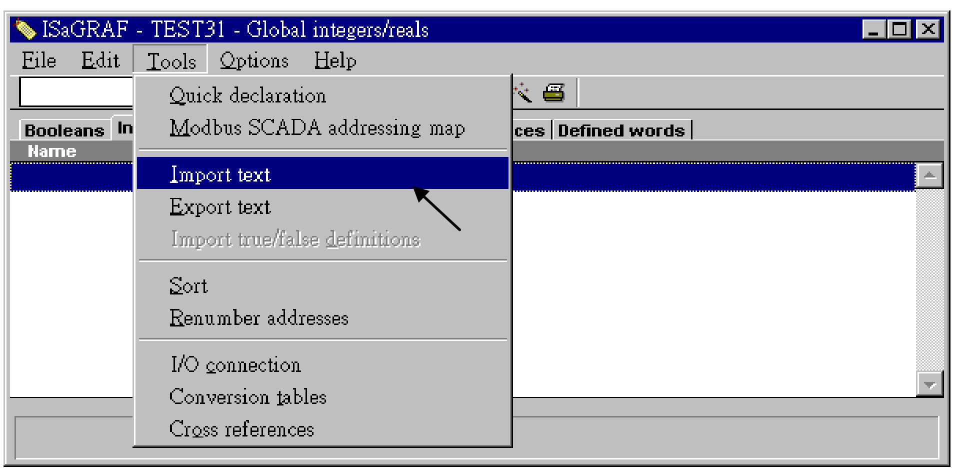
Select "File" and given a name to it, "int_1.txt" in this sample. Then click on "Browse" to select the directory where this txt file will be saved.



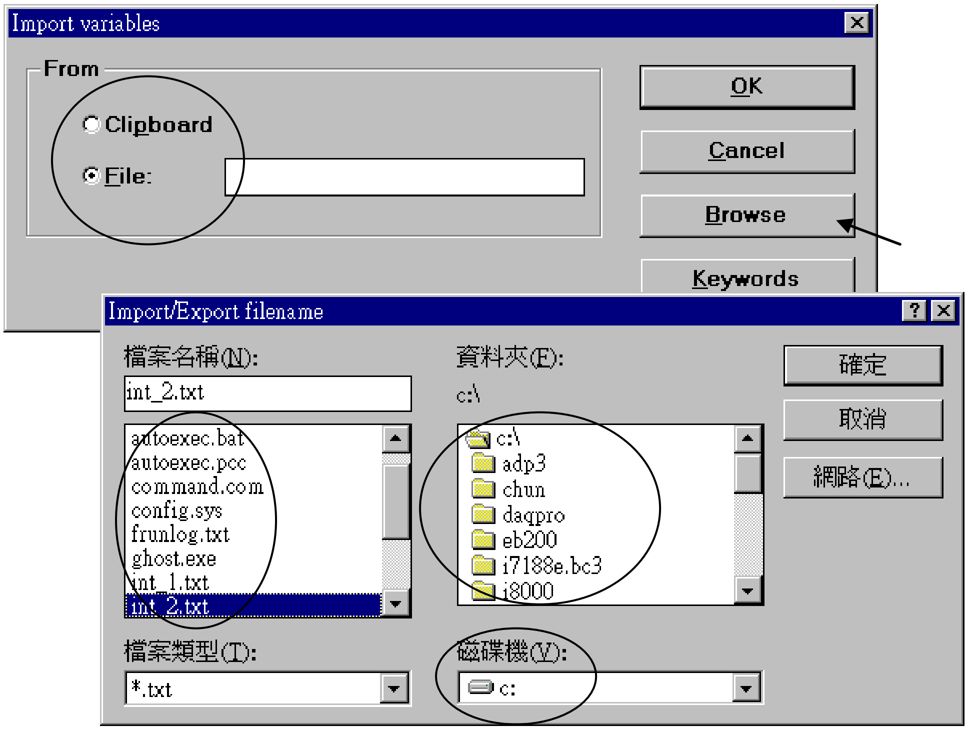
You may open and edit the file from the Excel. Please make sure to save this file with an extension ".txt".



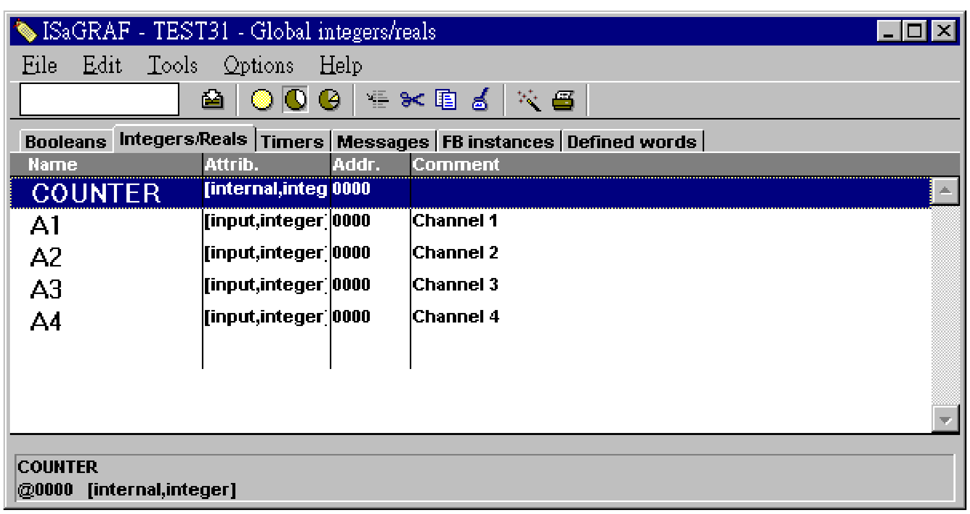
To **import** a text file to ISaGRAF, with an extension name ".txt", run "Tools" - "Import text" from the dictionary window.



Then click on “Browse” to select the associated text file.



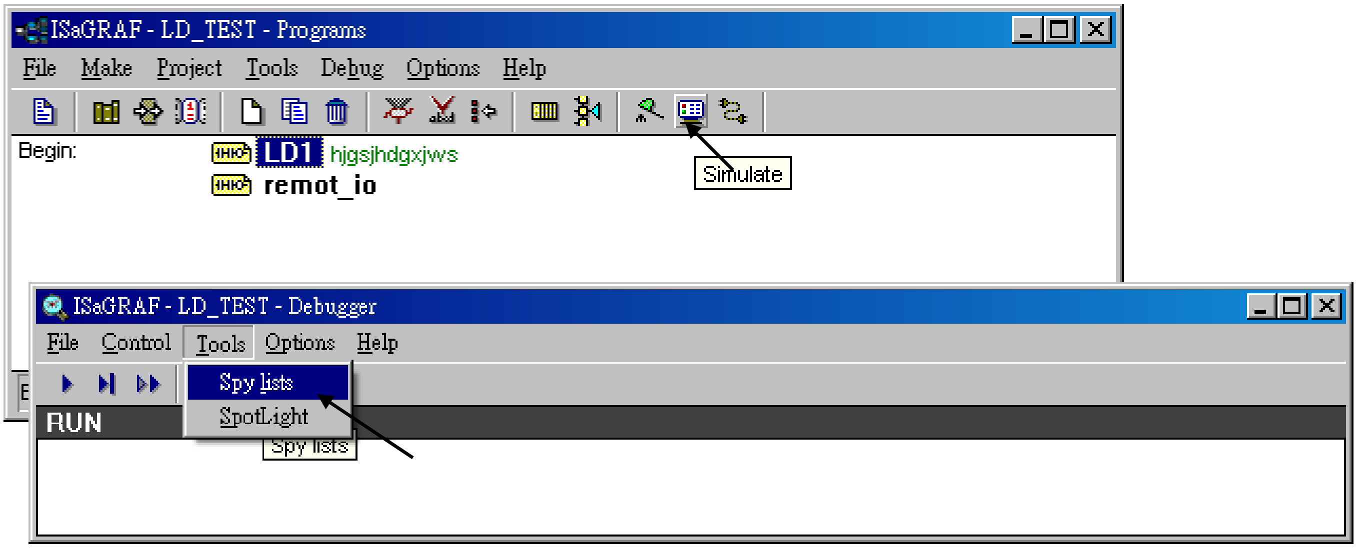
And then it is done as below.



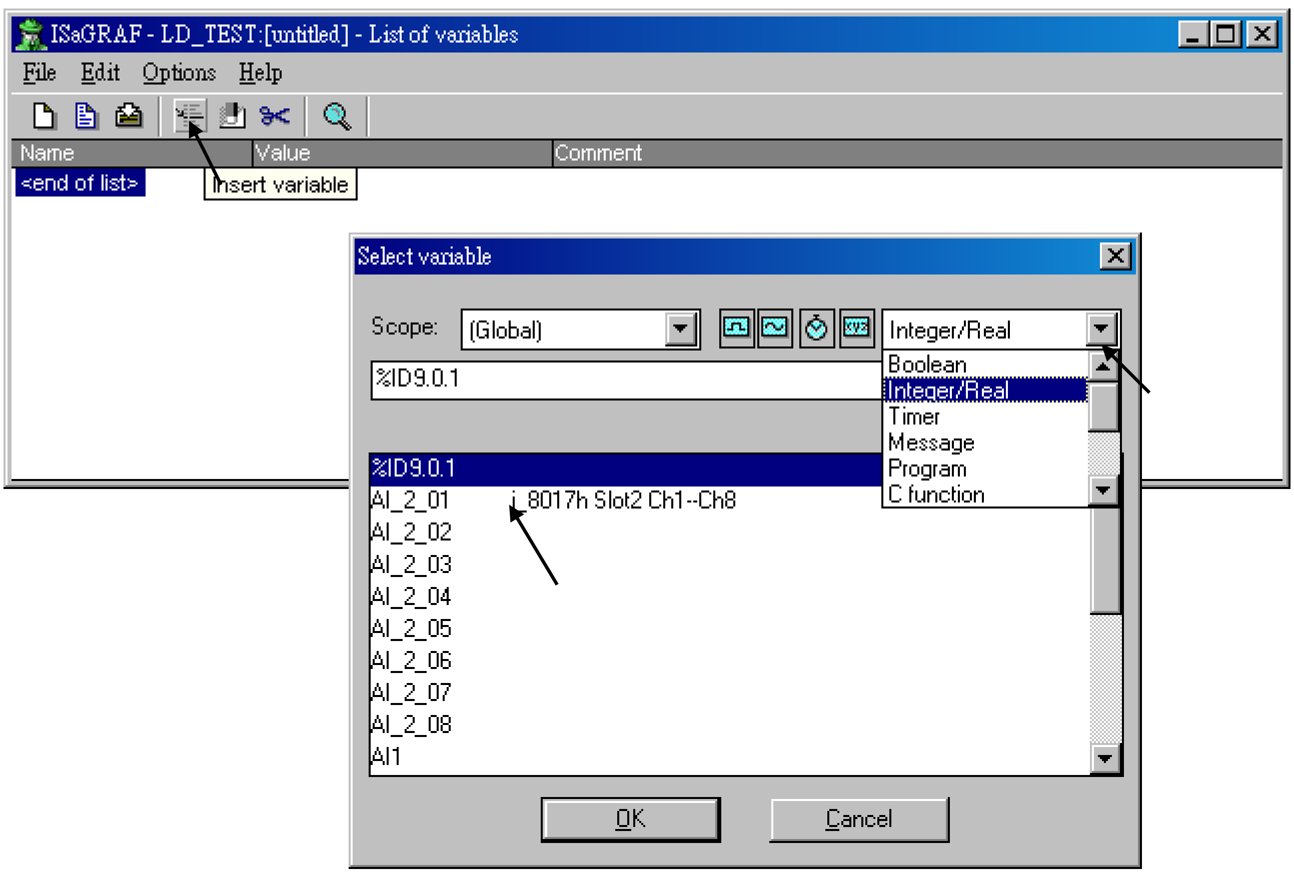
9.12: Spy list

ISaGRAF supports "Spy list" to spy some specific variables when linking to the controller. Please follow below steps to create a "spy list".

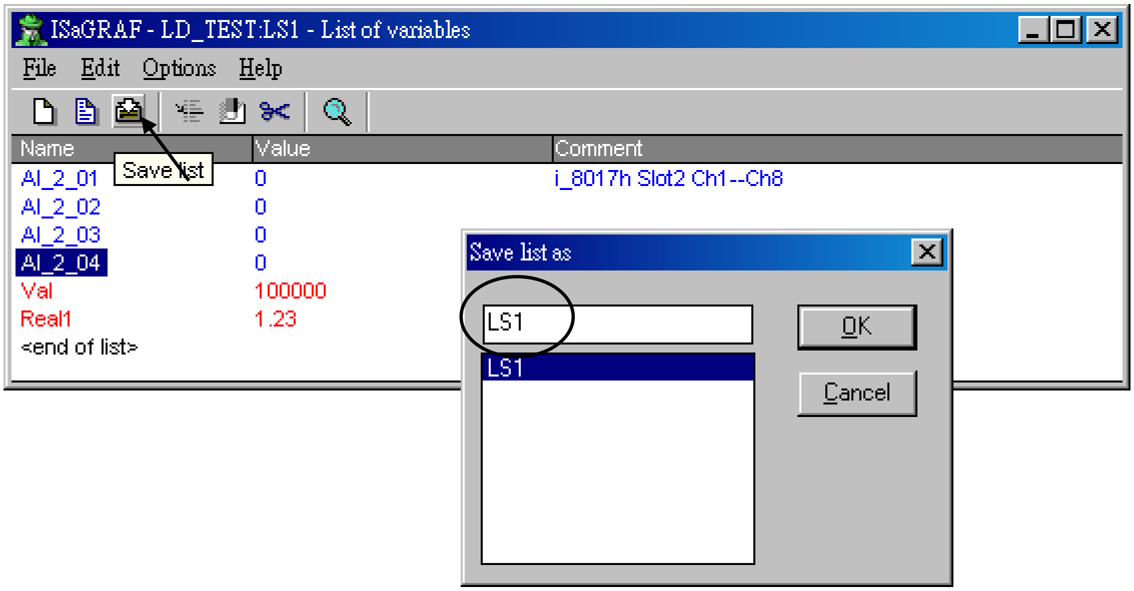
First click on "Simulate", then click on "Tools – Spy list".



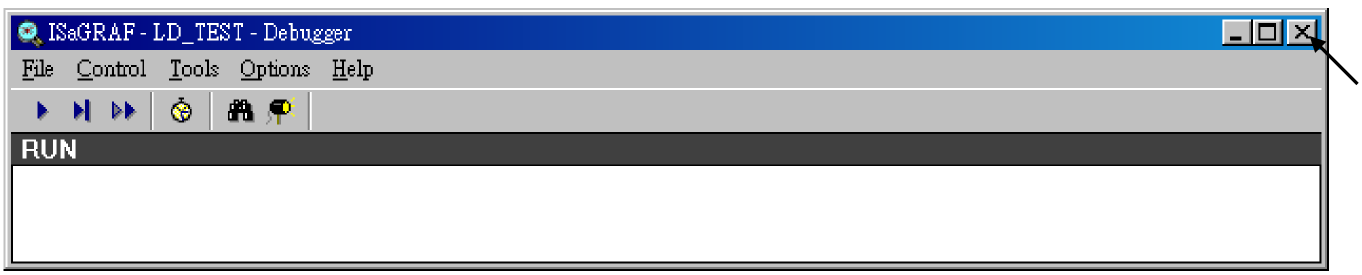
Next click on "Insert variable" to insert the variable to be spied.



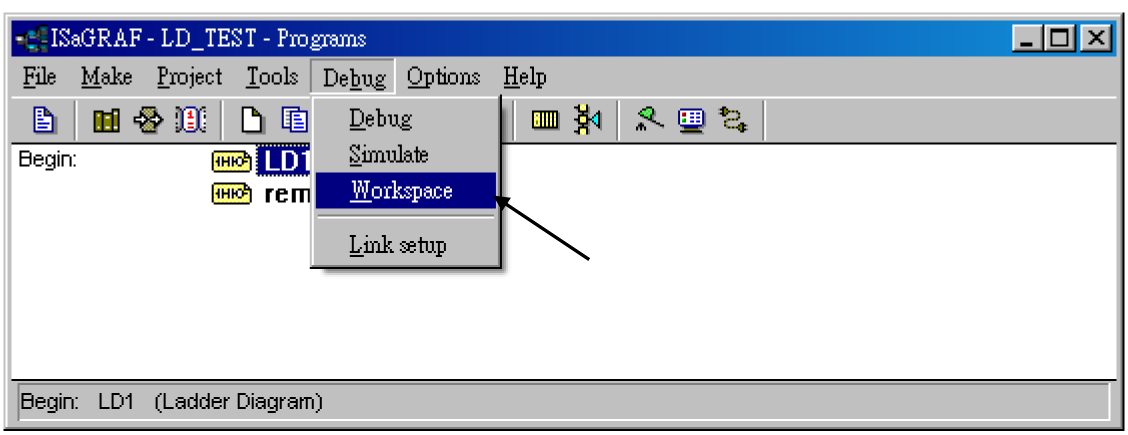
When all spied variables are inserted, remember to click on “Save list”.



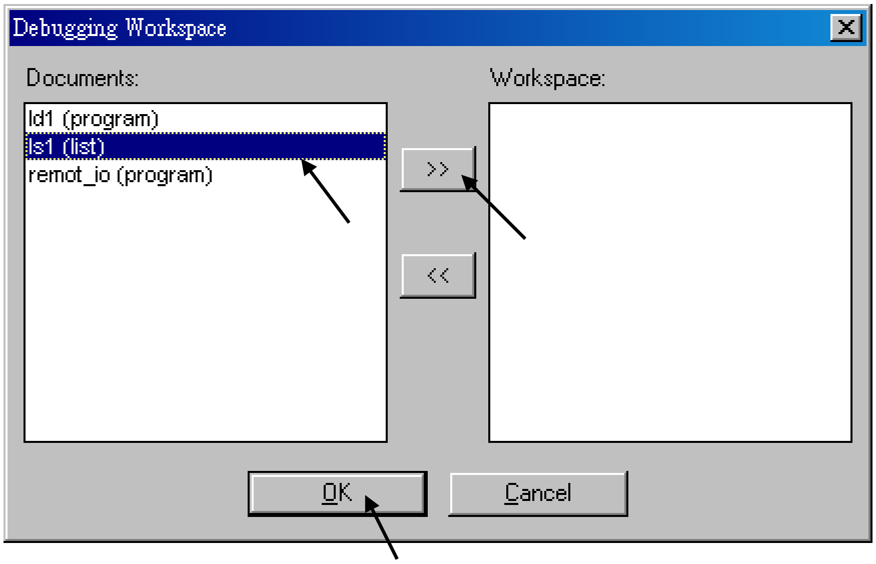
Then close the ”Debugger” window.



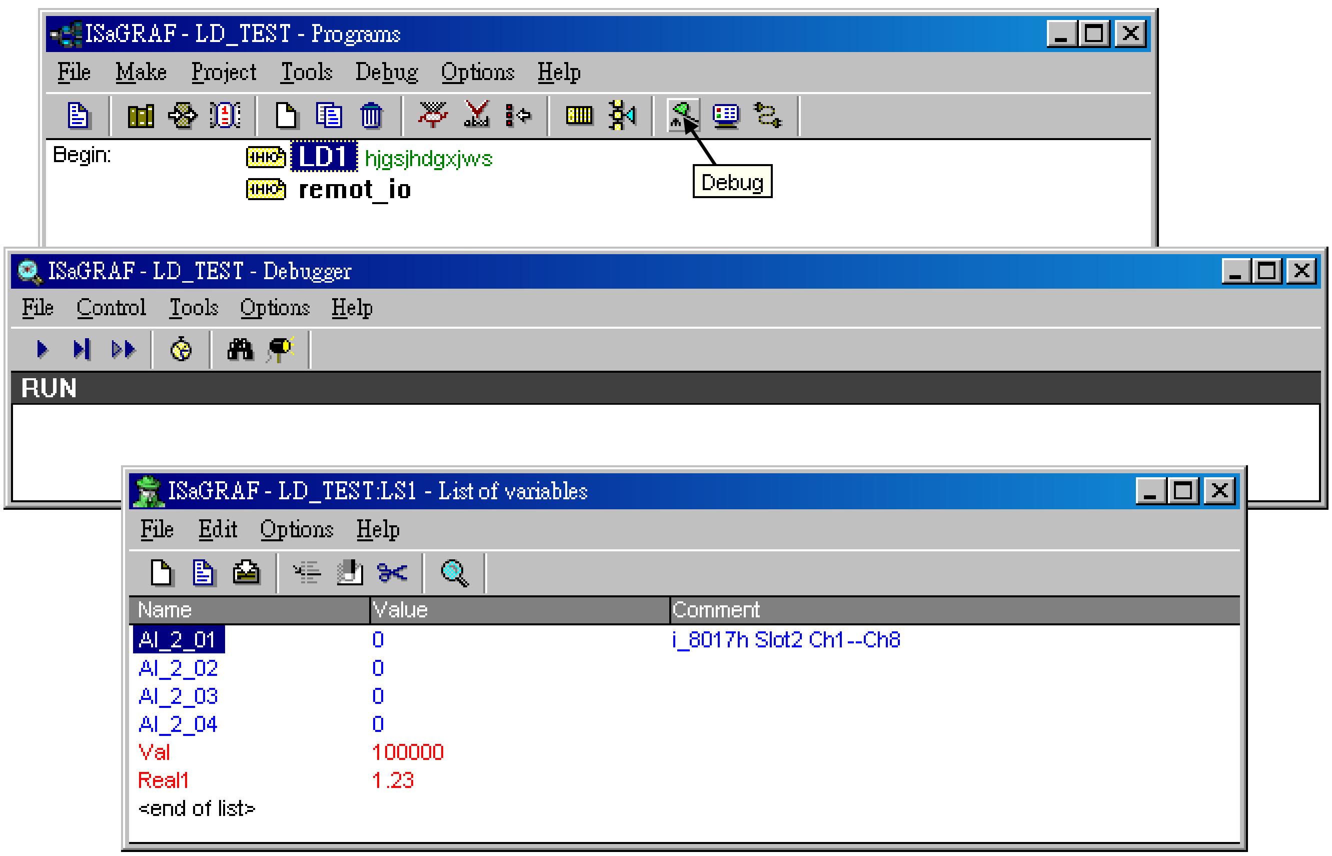
Click on “Debug – Workspace”



Move all “List” to the right hand side.

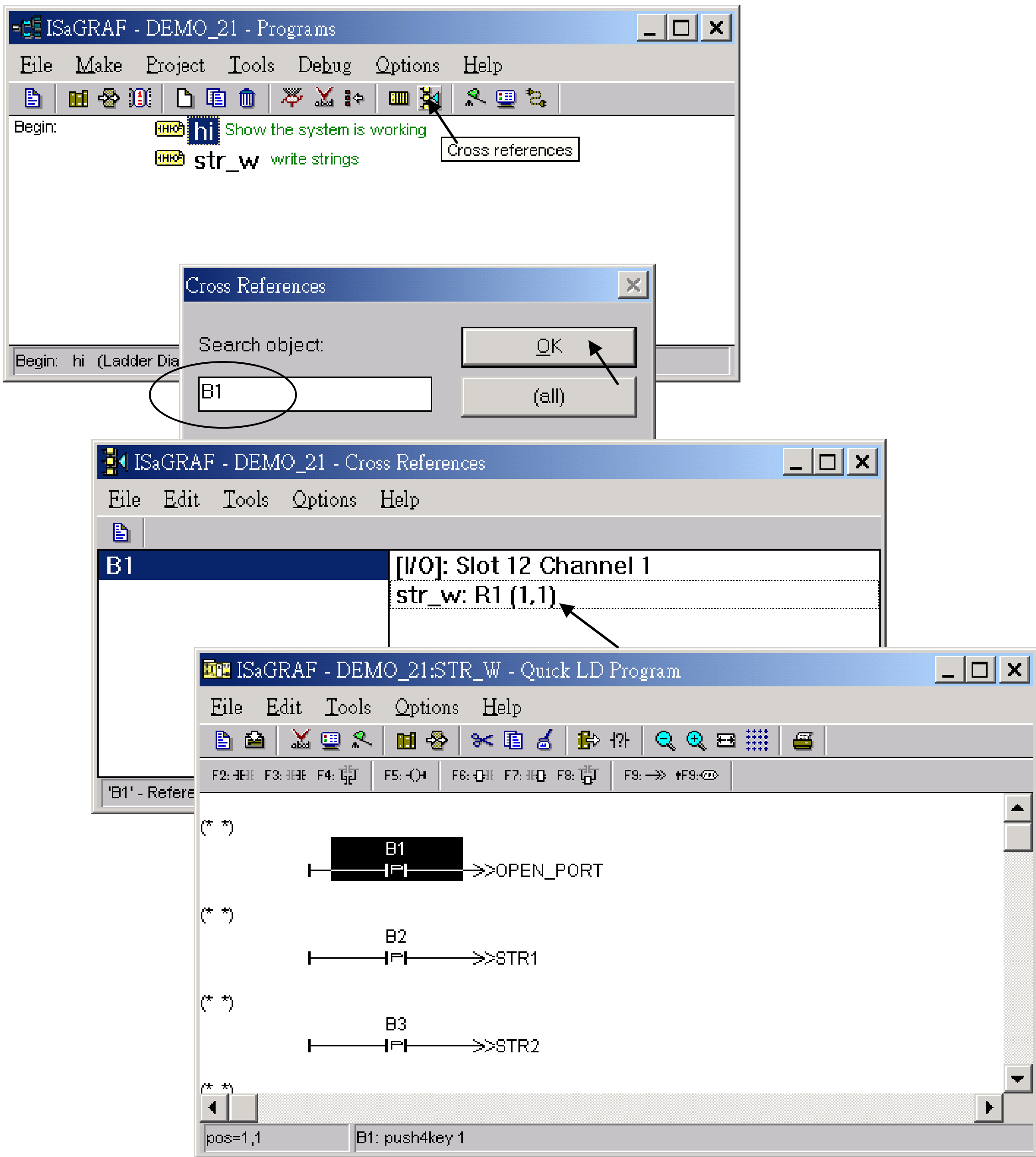


Then, you will see the “spy list” will automatically display when ISaGRAF linking to the controller.



9.13: How to search a variable name in an ISaGRAF project ?

Please click on "Cross references" and then entering the name you would like to search. The location will appear on the right hand side. Just click on it to get into it.



Chapter 10. The Retained Variable And Data Backup

10.1: The Retain Variable

Note: Read floating point value (RETAIN_F, RETAIN_X, RETAIN_A) from battery backup memory (or S-256/512 & X607/608) may cause controller fault if there is no floating point value saved inside. Please refer to Section 10.6 – “Controller Fault Detection”.

New Retain Function:

The I-8417/8817/8437/8837, I-7188EG/XG, W-8xx7, iP-8xx7, WP-8xx7, WP-5xx7, XP-8xx7-CE6, XP-8xx7-Atom-CE6, VP-2xW7, VP-2117, µPAC-7186EG, µPAC-5xx7 supports new retain function since below driver version.

I-7188EG + X607 or X608:	driver ver. 2.05 or later (better to be 2.17 or later)
I-7188XG + X607 or X608:	driver ver. 2.04 or later (better to be 2.15 or later)
µPAC-7186EG+ X607 or X608:	driver ver. since they are released around Jan.2008
µPAC-5xx7+XW-boards:	driver ver. since they are released.
I-8xx7+ S256 or S512 :	driver ver. 3.07 or later (better to be 3.19 or later)
W-83x7/83x6+ S256 or S512 :	driver ver. 3.18 or later (better to be 3.36 or later) with new back-plane of WB-831 (For 3-slot, released in 2006): Rev 2.6
W-87x7/87x6+ S256 or S512 :	driver ver. 3.18 or later (better to be 3.36 or later) with new back-plane of WB-871 (For 7-slot, released in 2006): Rev 2.8

The following controllers with a built-in battery backup SRAM:

iP-8xx7:	driver ver. since they are released.
VP-2117:	driver ver. since they are released.
WP-8xx7:	driver ver. since they are released.
WP-5xx7:	driver ver. since they are released.
VP-2xW7:	driver ver. since they are released.
XP-8xx7-CE6:	driver ver. since they are released.
XP-8xx7-Atom-CE6:	driver ver. since they are released.

The iP-8xx7, VP-2117, WP-8xx7, VP-25W7/23W7, XP-8xx7-Atom-CE6 and XP-8xx7-CE6 have built-in the 512 KB battery backup SRAM (no need to purchase the S-256/512).

If battery backup SRAM is found in the back-plane of the controller (I-8xx7: S256/S512, I-7188EG/XG: X607/X608, WinCon-8xx7: S256/S512), the maxinum number of retained variables for new retain function are listed as below. New retain variable is supported by below ISaGRAF “C-function” :

Target 1 : µPAC-7186EG /I-7188EG/XG+X607/608, I-8xx7+S256/512, iP-8xx7, VP-2117
Target 2 : W-8xx7+S256/512 (with new WinCon back-plane released in 2006), WP-8xx7, VP-2xW7, XP-8xx7-Atom-CE6 and XP-8xx7-CE6.

Retain_B : retain Boolean variable.	Target 1: max. 256 variables, Target 2: max. 1024.
Retain_N : retain Integer variable.	Target 1: max. 1024 variables, Target 2: max. 4096.
Retain_F : retain Real variable.	Target 1: max. 1024 variables, Target 2: max. 4096.
Retain_T : retain Timer variable.	Target 1: max. 256 variables, Target 2: max. 1024.

Retain_X : retain variable by using its Network address.
Retain_A : retain variable array by using its Network address. (Refer to Section 2.6.2)

Please refer to below ST examples to read the status of two batteries in the back-plane of the iP-8xx7, VP-25W7/23W7, WP-8xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6 controller.

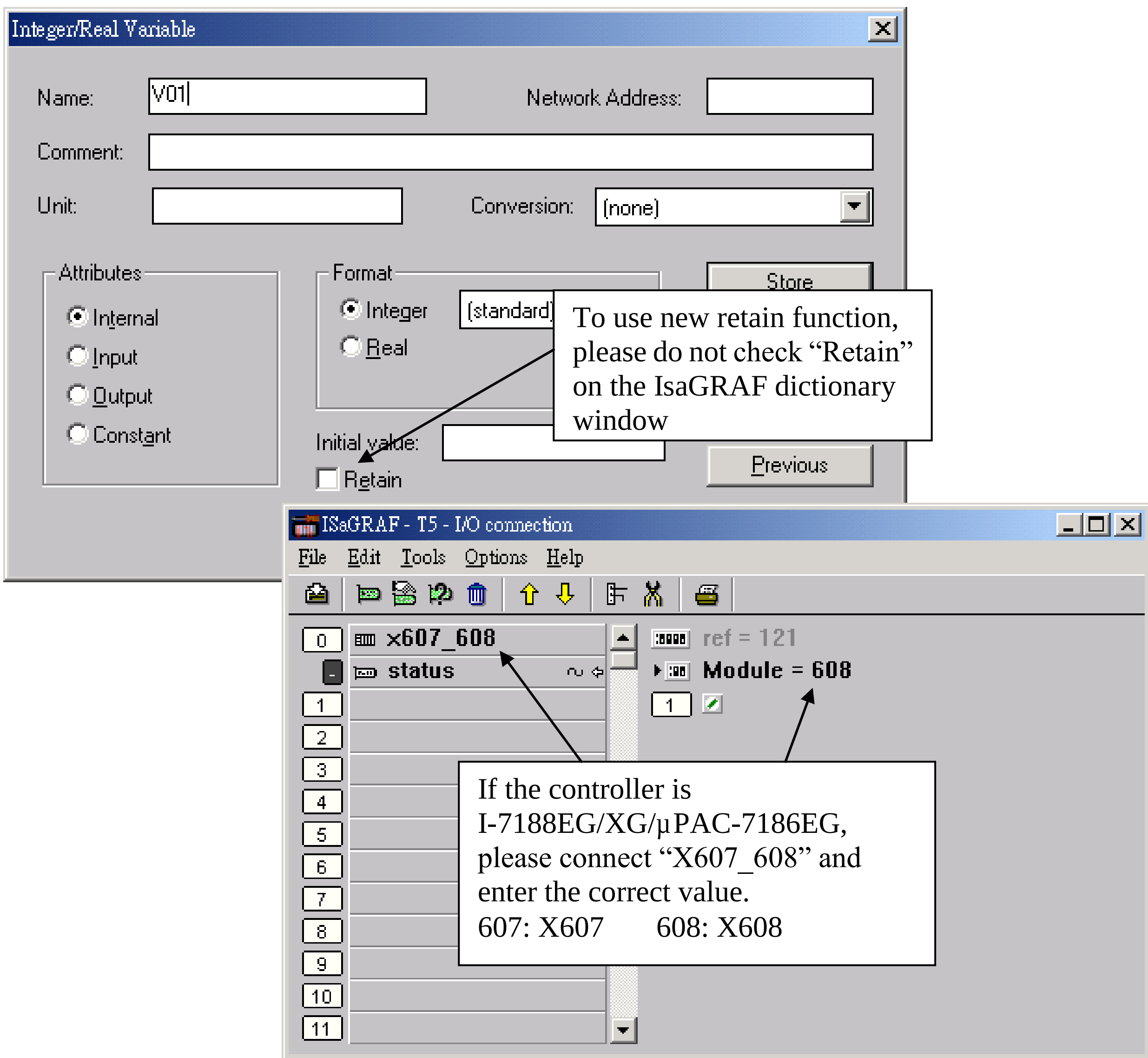
(* battery_state1 and battery_state2 are declared as an internal integer *)
battery_state1 := R_MB_ADR(1,9992) (Status of battery1, return 0: Low power 99: Power ok)
battery_state2 := R_MB_ADR(1,9993) (Status of battery2, return 0: Low power 99: Power ok)

The advantage for new retain variable:

1. The retain value will keep alive always whatever controller's power is off, or re-compiling & download a new ISaGRAF project unless the S-256/512, X607/608 has run out of battery. (Old Retain Method: the value will be cleared to 0 when re-download a ISaGRAF project.)
2. The amount of new retain variable can be stored is greater than old retain variable.

Important:

1. To use new retain function, please **do not** check "Retain" on the ISaGRAF dictionary window.
2. If your controller is**µPAC-7186EG or I-7188EG/XG,** please connect IO complex equipment "**X607_608**" in the IO connection windows.

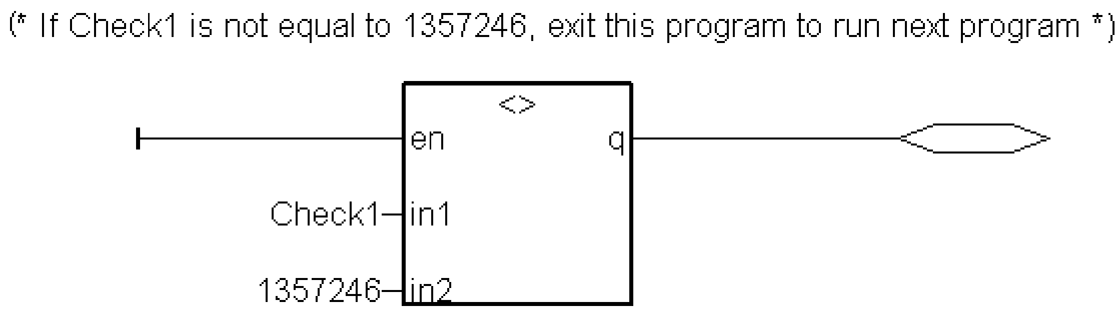


Example1: **(*** Set by variable name ***)**

```
(* Please set all retain variables in the first PLC scan cycle as the below code. and place this code on
   the top position of the project *)

(* To_Retain is declared as an internal boolean variable with an initial value TRUE *)
(* TMP is declared as an internal boolean variable *)
(*
   Check1 is declared as an internal integer. This "Check1" is for detecting if all the retain values have
   been well initialized. For example, you can define the value "1357246" to mean all the initial value
   of retain variables have been well set. And then if the "Check1" value is 1357246 , then allow the
   process to start. If the "Check1" value is not equal to "1357246", it means some retain variables
   havn't been set a proper value yet. So the process can not start. To run a program without the correct
   initial value for retain variables may cause some error. So user must set proper value for all retain
   variables at least once. And then remember to set this "Check1" as "1357246" to well start the
   proces . Then after, the program can start to run at every reboot because all value of retain variables
   (includes the "Check1") have been well setup.
*)

(* B1 , B2 is declared as internal Boolean variable, Do not check "Retain" *)
(* N1 , N2 is declared as internal Integer variable, Do not check "Retain"   *)
(* F1 , F2 is declared as internal Real variable, Do not check "Retain"   *)
(* T1 , T2 is declared as internal Timer variable, Do not check "Retain" *)

  if  To_Retain  then (* To set retained variables when controller is start running *)

     To_Retain  :=  False ;    (* Only do it once *)
     Tmp  :=  Retain_N( Check, 1); (* This variable used as a Tag *)
     Tmp  :=  Retain_B( B1 , 1 ) ;         Tmp  :=  Retain_B( B2 , 2 ) ;
     Tmp  :=  Retain_N( N1 , 11 ) ;        Tmp  :=  Retain_N( N2 , 12 ) ;
     Tmp  :=  Retain_F( F1 , 1 ) ;         Tmp  :=  Retain_F( F2 , 2 ) ;
     Tmp  :=  Retain_T( T1 , 1 ) ;         Tmp  :=  Retain_T( T2 , 2 ) ;
  end_if ;

(* To read the status of two batteries in the back-plane of the iP-8xx7, VP-25W7/23W7, WP-8xx7 and
   XP-8xx7-CE6 *)

(* battery_state1 and  battery_state2 are declared as an internal integer *)
battery_state1 := R_MB_ADR(1,9992)  (Status of battery1, return 0: Low power  99: Power ok)
battery_state2 := R_MB_ADR(1,9993)  (Status of battery2, return 0: Low power  99: Power ok)

  (* After then B1, B2, N1, N2, F1, F2, T1, T2 will be automatically retained in the program *)
```

Then, you can judge whether the "Check1" value is correct and then start to do some actions in the Ladder or ST program. (The aim is to set proper retain value at least once, then set this "Check1" as "1357246" to start the program as below.)

Then, you can add the “Check1” condition into the first row of the Ladder program.



■ ■ ■

Then, you can add the “Check1” condition into the first row of the ST program.

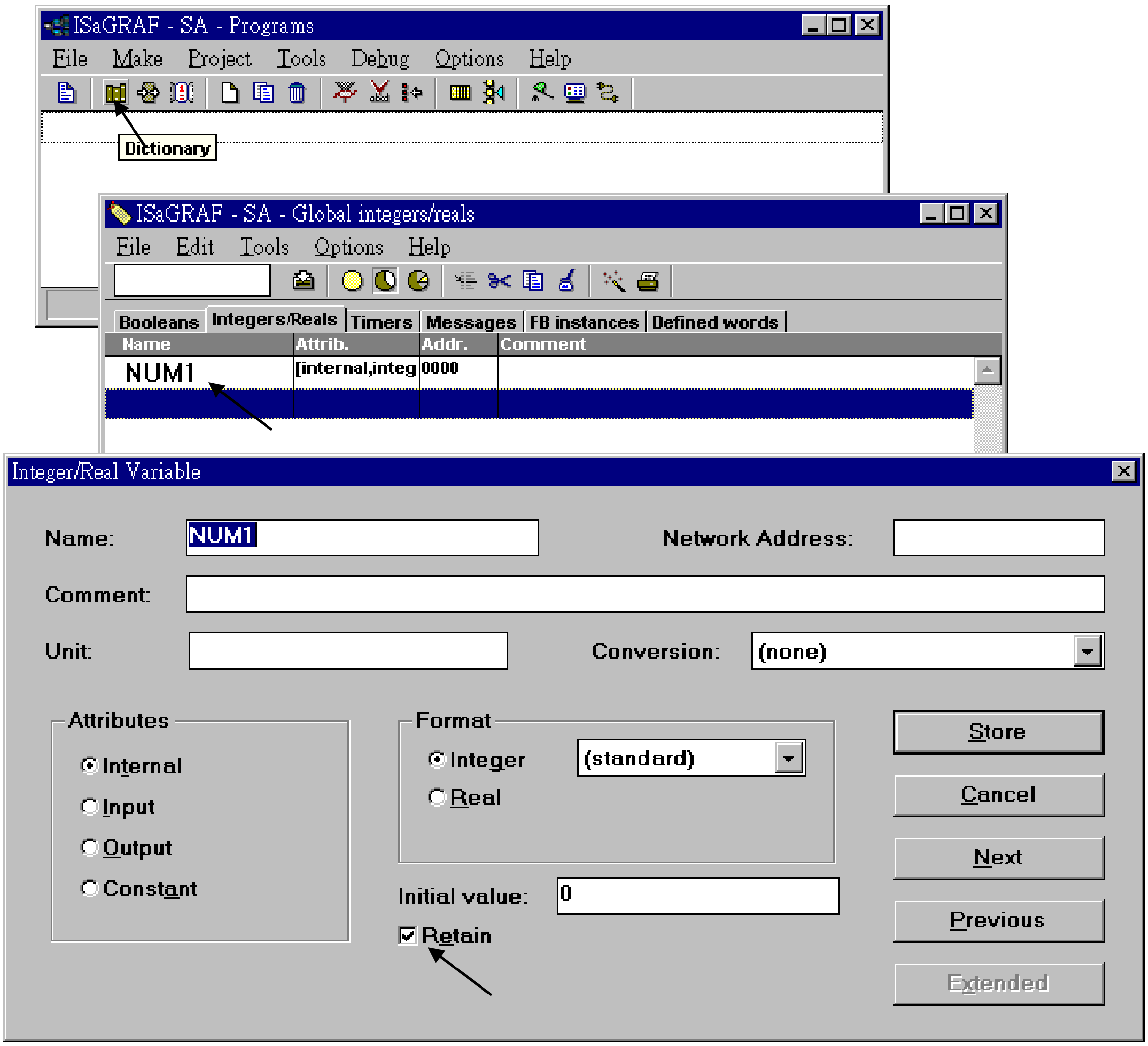
(* If Check1 is not equal to 1357246, you may write some code to set some safe and initial value to those retain variables or exit this program to run the next program *)

```
if   Check1  <>   1357246   then
    return ;
end_if  ;

 ...
```

Old Retain Method:

If the controller doesn't find the battery backup SRAM in the back-plane of the controller (I-8xx7: S256/S512, I-7188EG/XG/µPAC-7186EG: X607/X608, WinCon-8xx7: S256/S512). The I-8xx7 and I-7188EG/XG/µPAC-7186EG supports old retain variable, while WinCon supports no retain variable. There is a 31-byte "NVSRAM" in the I-8xx7 & I-7188EG/XG/µPAC-7186EG 's CPU board . Up to **six Integers/Reals** (signed 32-bit) and **sixteen Booleans** can be retained with this 31-byte NVSRAM.

To enable the old retained function, click on "Retain" for each associated variable.



Note:
If battery backup SRAM is found in the controller (I-8xx7: S256/S512, I-7188EG/XG/µPAC-7186EG: X607/X608, WinCon-8xx7: S256/S512), Please use **new retain function** listed in the former section. **The old retain method has two disadvantage**: (1) The data will lose when download a modified ISaGRAF project. (2) Its retain variable amount is less than new method.

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. It can be read freely however can be written only about to 100,000 times. So, it apply for saving the configured data.

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

EEP_B_R	Read one boolean
EEP_BY_R	Read one byte
EEP_WD_R	Read one word (2 bytes, signed, -32768 to +32767)
EEP_N_R	Read one integer (4 bytes, signed)
EEP_F_R	Read one REAL (4 bytes, float)

(If the related data saved in the EEPROM is not REAL format, using “EEP_F_R” to read it may generate a controller local fault No. = 114. Please refer to Chapter 10.6)

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	Remove the protection of EEPROM, then write operation is allowed.
EEP_PR	Set the protection of EEPROM, then write operation is not allowed.
EEP_B_W	Write one boolean.
EEP_BY_W	Write one byte (Byte: 0 to 255).
EEP_WD_W	Write one word (2 bytes, signed, -32768 to +32767).
EEP_N_W	Write one integer (4 bytes, signed).
EEP_F_W	Write one REAL (4 bytes, float).

Note:

1. “EEP_F_R” , “EEP_N_R” , “EEP_N_W” and “EEP_F_W” all use the same EEPROM memory, please DO NOT operate the same EEPROM address as Integer and also as REAL at the same time.
2. If the related data saved in the EEPROM is not REAL format, using “EEP_F_R” to read it may generate a controller local fault No. = 114. Please refer to Chapter 10.6

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. For different PAC, use the different memory size of the EEPROM. **For I-8xx7 & I-7188EG/XG,** there are total 1,512 bytes and 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.

For I-8xx7 & I-7188EG/XG:

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

For iP-8xx7, µPAC-7186EG, µPAC-5xx7, VP-2117, WinCon-8xx7:

```
Byte    4n-3, 4n-2, 4n-1, 4n        (* n = 1, 2, …3568 *)
Word    2n-1, 2n
Integer n
```

For WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6, VP-25W7, VP-23W7:

```
Byte    4n-3, 4n-2, 4n-1, 4n        (* n = 1, 2, …1536 *)
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 every PLC scan cycle (normally, the cycle time is about 3 to 60 ms depends on the application and controller model No.) .

```
(* 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 in the EEPROM will consume a lot of CPU time of 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 controller is powered up, and then updated the associated address in the EEPROM when the value is changed. Please refer to a sample program in Chapter 11 – "**demo_17**" & "**Wdemo_10**". For those data which are frequently changed are not suitable to be stored in the EEPROM.

10.3: Battery Backup SRAM

Note: The WP-8xx7, VP-25W7, XP-8xx7-CE6, XP-8xx7-Atom-CE6 PAC support new retain variable (refer to Section 10.1) but not support S_X_X function (listed in this section) to read/write battery backup SRAM. Read floating point value from S-256/512 & X607/608 may cause controller fault if no floating point value saved inside. Please refer to Section 10.6 – “Controller Fault Detection”

The I-8xx7 controllers can integrate with a S256 or S512 battery backup SRAM to store data, alarm, and information, while X607 & X608 for the I-7188EG/XG controller. The data stored in these SRAM is always retained unless their battery running out of energy. Their memory size is as below. The upper 12K is reserved (while 64 KB is reserved by W-8337/8737/8336/8736).

I-8417/8817/8437/8837, I-8437-80 / I-8837-80		**I-7188EG/XG/µPAC-7186EG**	
S256	244K bytes (256-12=244)	X607	116K bytes (128-12=116)
S512	500K bytes (512-12=500)	X608	500K bytes (512-12=500)
W-8337/8737/8336/8736			
S256	192K bytes (256-64=192)		
S512	448K bytes (512-64=448)		

If battery backup SRAM is found in the controller, the maxinum number of retained variables for new retain function “Retain_X” , “Retain_A” , “Retain_B”, “Retain_N”, “Retain_F” & “Retain_T” can be extend to as below (please refer to Section 10.1).

I-7188EG/XG+X607/608 and I-8417/8817/8437/8837+S256/512		
	New Retain function	**old retain method**
Boolean	256	256
Integer	1024	256 (Integer + Real)
Real	1024	
Timer	256	32
W-8337/8737/8336/8736+S256/512 with new WinCon back-plane (section 10.1)		
	New Retain function	**old retain method**
Boolean	1024	1024
Integer	4096	4096 (Integer + Real)
Real	4096	
Timer	1024	1024

ICP DAS provides an utility “**ICPDAS UDloader**” that can be installed in the PC to upload and download data from/to the ISaGRAF controller. Please copy “**UDloader.exe**” from the ICP DAS’s CD-ROM:\napdos\isagraf\some_utility\ to your PC / windows.

The I-8417/8817/8437/8837 supports S256/S512 since its driver version of 2.25 (better to be 3.19 or later), while I-7188EG supports X607/608 since its driver version of 1.18 (better to be 2.17 or later), and version 1.16 for I-7188XG (better to be 2.15 or later). W-8337/8737/8336/8736 supports S256/S512 since its driver version 3.18 (better to be 3.36 or later) (Please refer to section 10.1). If your driver is older one, please upgrade the hardware driver to the associate version or a higher version. The driver can be found from the ICP DAS’s web site: http://www.icpdas.com/products/PAC/i-8000/isagraf.htm

10.3.1: Access to the SRAM

The SRAM can store boolean, byte, word, integer, real & message. Their format is as below.
(Please refer to Chapter 11.3.7 for a demo program using S-256/512 by UDloader.exe)

Boolean:	True=1, False=0	1 byte
Byte:	0 ~ 255	1 byte
Word:	-32768 ~ 32767	2 bytes
Integer:	signed 32-bit	4 bytes
Real:	float	4 bytes
Message:	string (len<=255)	len bytes

To access to the SRAM, the below functions can be used (Please refer to Appendix A).

S_B_R, S_B_W, S_BY_R , S_BY_W, S_M_R, S_M_W
S_WD_R, S_WD_W, S_N_R , S_N_W, S_R_R, S_R_W
S_MV

10.3.2: Upload data stored in the SRAM

For PC to upload data stored in the volatile SRAM of the ISaGRAF controllers, the SRAM should be divided into 1 or up to 8 files. Each file has a ID No. of 1 to 8 and a name of up to 8 characters and 3 file extension. The below functions are for handling file format inside the SRAM.

S_FL_INI, S_FL_AVL, S_FL_RST, S_FL_STS

Please use functions of S_FL_INI & S_FL_AVL to arrange the file resident location & current available location (Please refer to Appendix A & demo_40, 41 or 42).

The volatile SRAM is consisted of bytes. The total number of bytes available depends on which module is used as below. The upper 12K is reserved.

Module name	Byte No.
I-8xx7: S256	1 ~ 249,856 (244K), (256-244=12K is reserved)
I-8xx7: S512	1 ~ 512,000 (500K), (512-500=12K is reserved)
I-7188XG/EG: X607	1 ~ 118,784 (116K), (128-116=12K is reserved)
I-7188XG/EG: X608	1 ~ 512,000 (500K), (512-500=12K is reserved)

A file can be located at any place inside these bytes. Each file's location can be described as (**Begin**, **End**). Begin is the lower limit byte No. of the associated file, while End is the upper limit byte No., and Begin is always less than End.

A file inside the SRAM has a current available area (**Head**, **Tail**). Head is the starting position of the file, Tail is the ending position. Head can be larger, less than or equal to Tail.

For example, a file resides at (Begin, End) = (1, 20000)

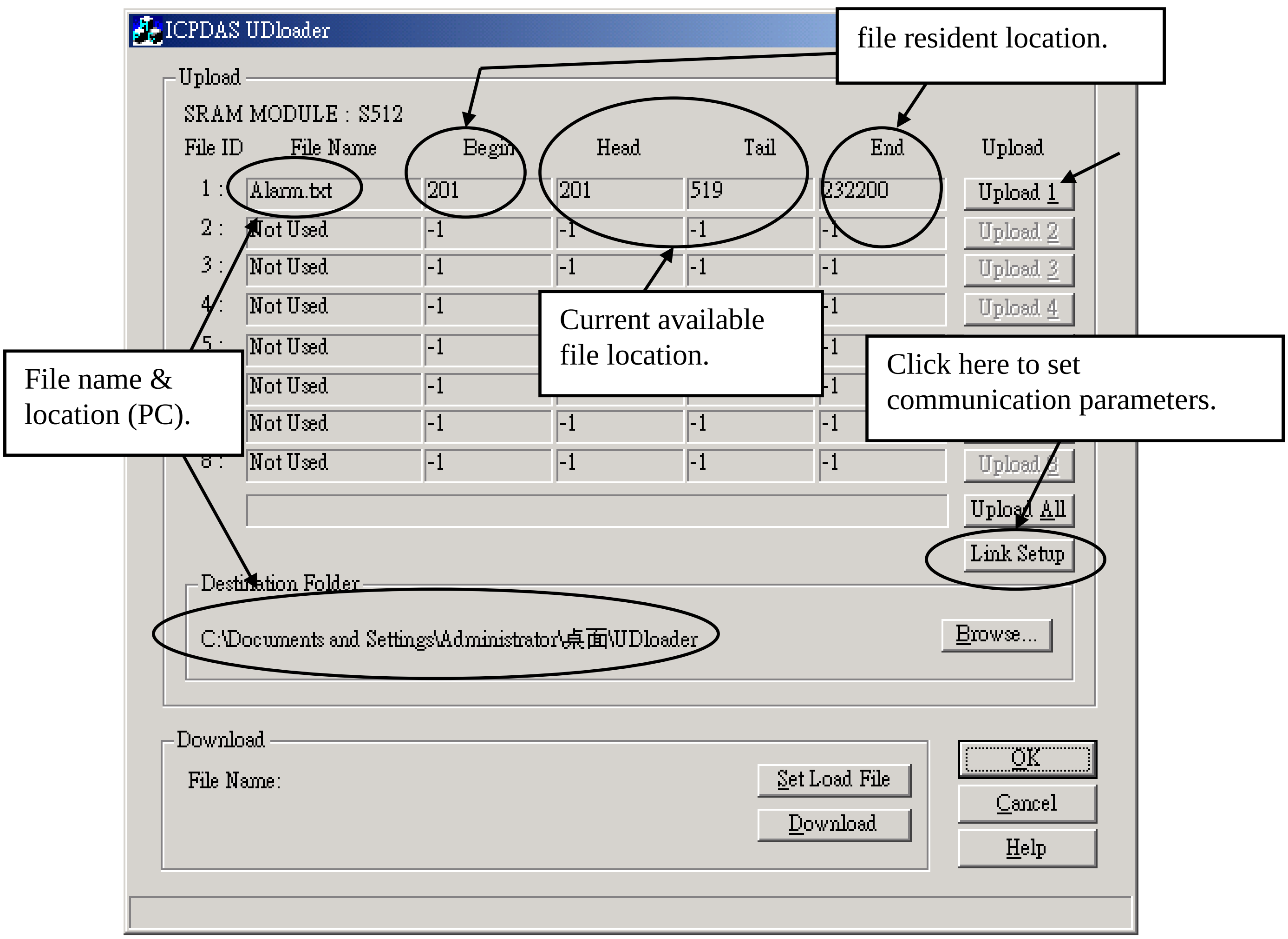
1. If (Head, Tail) = (1001,5100), it means the available data of the file is starting from byte No. of 1001 to 5100. The available file contains 4100 bytes.
2. If (Head, Tail) = (10001,5000), it means the available data of the file is starting from byte No. of 10001 to 20000 and then continued with 1 to 5000. The available file contains 15000 bytes.

3. If (Head, Tail) = (5001,5000), it means the available data of the file is starting from byte No. of 5001 to 20000 and then continued with 1 to 5000. The available file contains 20000 bytes.
4. If (Head, Tail) = (5000,5000), it means the available data of the file is empty, 0 byte.
5. If (Head, Tail) = (-1,-1), it means the available data of the file is empty, 0 byte.

To upload the data stored in the SRAM, please make sure you have installed the “ICPDAS UDloader” on your PC.

To upload data stored in the SRAM of the ISaGRAF controller to PC, please run “UDloader.exe”, then click on “Link Setup” to set proper communication parameters, then click on “Upload 1” to upload it.

Example:
Please download demo_41 to one I-8417/8817/8437/8837. Then push button 1 or 2 or 3 or 4 several times. Then upload the file stored in the SRAM.



10.3.3: Download data to the SRAM

For PC to download data to the volatile SRAM of the ISaGRAF controllers. The below functions can be used. Please refer to Appendix A & demo_44.

S_DL_EN, S_DL_DIS, S_DL_RST, S_DL_STS

Please call "S_DL_EN" to enable it.

The Controller accepts only the binary format for String, Byte, Word, Int & Real.

Byte:	0 ~ 255	1 byte	
Word:	-32768 ~ +32767	2 byte	[low bye] [high byte]
Int:	32-bit, signed integer	4 byte	[lowest] [2nd] [3rd] [highest]
Real:	32-bit float	4 byte	[lowest] [2nd] [3rd] [highest]
String:		up to 255 bytes	

If using the "UDloader.exe" to download data to the volatile SRAM, the data to be downloaded should be edited as a text file. Its format should follow the below rules.

The first row should be a No. indicate that to download to which starting Byte No. of the SRAM. Valid starting byte No is as below.

S256:	1 ~ 249,856	S512:	1 ~ 512000
X607:	1 ~ 118,784	X608:	1 ~ 512000

The other rows are the data.

A. **String**
String should start and end with the character of ' , for ex. 'Abcd123' (7 byte). The $NN (NN in hexidecimal and should not equal to 0), could be used to indicate the ASCII character. For ex, 'ABC$0D' contains 4 bytes, the 4th byte is <CR>.

B. **Byte**
Byte should start with (and end with) , for ex. (0) , (123), (255). Valid byte range is from (0) to (255).

C. **Word**
Word should be start with [and end with] , for ex. [-100] , [20000], [32767]. Valid word range is from [-32768] to [32767].

D. **Integer**
Integer should be start with { and end with } , for ex. {-1234567} , {200000}. Valid integer range is from {-2147483648} to {2147483647}.

E. **Real**
Real value should be start with < and end with > , for ex. <123> , <1.56E-2>, <-123.456>.

The character between each Byte, Word, Integer, Real, String at the same line should be at least one space character <SP> or , <Comma> or, <Tab>

For example,

201 ← to download to the SRAM which staring from byte No. 201
'Hello' (10) (20) (30) (40) [-10000] {70000} 'End' ← data (total 18 bytes)

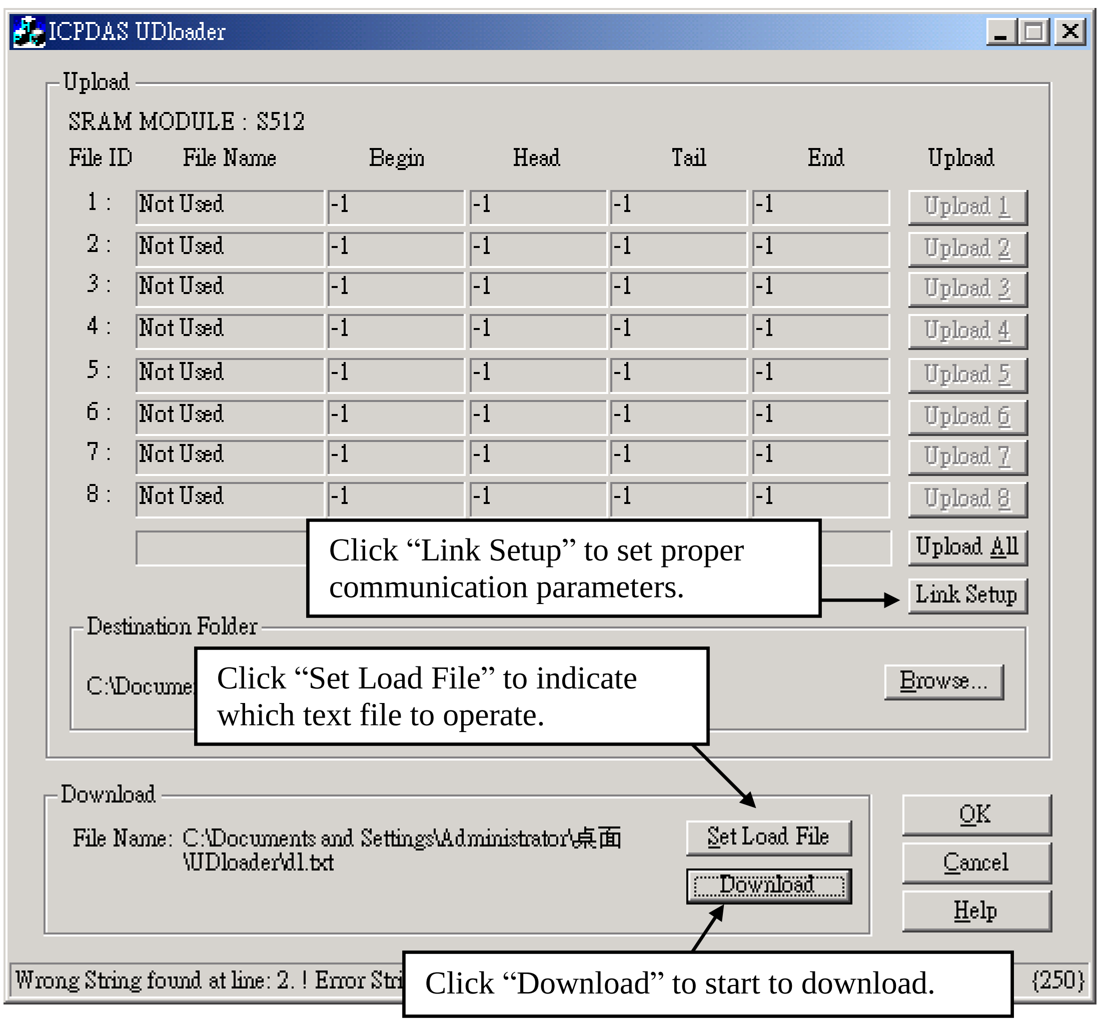
1 ← to download to the SRAM which staring from byte No. 1
(23) ← data (total 57 bytes)
{-1},{2},{-3},{4},{-5},{6} {-7} {8} {-9}   {10}   ← comma, <SP> & <Tab> are all acceptable
<0.123> <456.789> <100> , <2.3E3>

Example:
Please download demo_44 to one I-8417/8817/8437/8837. Then edit a text file as below.

1
{1000} {250} {100} ‘sTART’

The {1000} means the blinking period of L1 is 1000 ms.
The {250} means the blinking period of L2 is 250 ms.
The {100} means the blinking period of L3 is 100 ms. .

Then run “UDloader.exe”. You will see something change on the led of the controller.



10.3.4: Operation Functions for the battery backup SRAM

The below functions are for the ISaGRAF controller to access to the volatile SRAM. More information listed at Appendix A.4

S_FL_INI   Init one file's name & location for the volatile SRAM
S_FL_AVL   Set one file's current available byte No. for the volatile SRAM
S_FL_STS   Get file's Status, end byte No. that has been load by PC for the volatile SRAM
S_FL_RST   Reset file's Status to "Not been load by PC yet" for the volatile SRAM

S_B_R: Read one Boolean (TRUE, FALSE)
S_BY_R: Read one Byte (0 ~ 255)
S_WD_R: Read one Word (-32768 ~ +32767)
S_N_R: Read one Integer (32 bit, signed)

S_R_R: Read one Real (32 bit, float)
(If the data in related address of the battery backup SRAM is not a REAL value, using "S_R_R" to read it may generate a controller local fault No. = 102. please refer to Chapter 10.6)

S_M_R: Read one String

S_B_W: Write one Boolean (TRUE, FALSE)
S_BY_W: Write one Byte (0 ~ 255)
S_WD_W: Write one Word (-32768 ~ +32767)
S_N_W: Write one Integer (32 bit, signed)
S_R_W: Write one Real value (32 bit, float)
S_M_W: Write one String

S_DL_EN   Enable the download permission for PC to download data to the volatile SRAM
S_DL_DIS   Disable the download permission for PC to download data to the volatile SRAM
S_DL_STS   Get PC's Download Status for the volatile SRAM
S_DL_RST   Reset the Download Status to "-1:No action" for the volatile SRAM

S_MV   copy data in the volatile SRAM

10.4: Using I-8073 - MultiMediaCard to store data

The I-8072 / 8073 is not support by ISaGRAF PAC.

10.5: Reading & Writing File

Note:

1. If the data type in the related file position is not REAL type (32-bit float format), using "F_READ_F" function to read this data may generate a local controller fault No = 117 (please refer to Chapter 10.6).
2. Only WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6, VP-25W7/23W7 support file operating functions. Not for I-8xx7, I-7188EG/XG, µPAC-7186EG, µPAC-5xx7, iP-8xx7 and VP-2117.
3. If the file path is inside the '\System_Disk\' or '\Micro_SD\' (for WP-8xx7, WP-5xx7, VP-25w7 and VP-23W7) or '\System_disk2' (for XP-8xx7-Atom-Ce6 and XP-8xx7-CE6) folder, for example, '\Micro_SD\data1.txt' , the file will continue exist even the controller 's power is switched Off. However, it consums lots of CPU time to Read / Write files in the above listed directories.
4. If the file location belongs to RAM, for example – '\Temp\data2.txt', it will be stored in the RAM memory. The file saved in controller's RAM will be lost when power is switched OFF. The advantage of RAM memory is that the file read/write speed is much faster.

WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6, VP-25W7/23W7 support below ISaGRAF standard functions.

F_ROPEN	Open file in Binary format for read operation (file should exist already) .
F_WOPEN	Open file in Binary format for read and write operation (file should exist)
F_CLOSE	Close a file.
F_EOF	Test if reach the End-Of-File position.
FA_READ	Read one binary long integer (4-bytes, signed) from file.
FA_WRITE	Write one binary long integer (4-bytes, signed) to file.
FM_READ	Read one message (string) from file.
FM_WRITE	Write one message (string) with <CR> <LF> char. at end of message to file.

WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6, VP-25W7/23W7 support below ICP DAS c-functions.

F_APPEND	Append one file to the end of the other file.
F_COPY	Copy one file to another file.
F_CREAT	Create a new file.
F_DELETE	Delete a file.
F_DIR	Create a new directory (folder).
F_END	move current file position to the End-Of-File position.
F_EXIST	Test if a directory or a file exist.
F_SEEK	Move current file position to a specified position.
F_READ_B	Read one binary byte (0 - 255) (1 byte, unsigned) from file.
F_WRIT_B	Write one binary byte (0 - 255) (1 byte, unsigned) to file.
F_READ_W	Read one binary word (-32768 to +32767) (2 byte, signed) from file .
F_WRIT_W	Write one binary word (-32768 to +32767) (2 byte, signed) to file.
F_READ_F	Read one binary REAL (4-bytes, Float) from file. Like 123.45, -2.15E-03, …
F_WRIT_F	Write one binary REAL (4-bytes, Float) to file .

Please refer to section 11.3.6 or below for demo program.
WP-8xx7 CD-ROM: \napdos\isagraf\wp-8xx7\demo\ "wpdmo_54.pia" , 55, 56, 51, 50, 1 or 2 or
ftp://ftp.icpdas.com/pub/cd/winpac-8xx7/napdos/isagraf/wp-8xx7/demo/

10.5.1: Wpdmo_51: Read 10 REAL values from a file. Total 10 rows, each contains one REAL value

The "Wpdmo_51.pia" can be found at
WP-8xx7 CD-ROM:\napdos\isagraf\wp-8xx7\demo\ or
ftp://ftp.icpdas.com/pub/cd/winpac-8xx7/napdos/isagraf/wp-8xx7/demo/

If functions of Msg_F , Msg_N , ARY_F_R, AFY_F_W are not found in your PC / ISaGRAF, please download "ICP DAS utilities For ISaGRAF" at http://www.icpdas.com/products/PAC/i-8000/isagraf.htm Driver. Then run "setup.exe" to restore them to your ISaGRAF workbench.

The "Wpdmo_51" program will read 10 REAL values from "\System_Disk\data51.txt" when the controller is just power up or user set the "RE_LOAD" value to become "TRUE" at any time .
(To read / write file in the System_Disk take lots of CPU time, please do not read / write it frequently. And please always close the file after the operation. If user read / write file in every PLC scan cycle, the PLC scan time will become a very big time. It will be a bad performance !)

To test this sample program, please edit a text file "data51.txt" in your PC by , for example – "Notepad". Please enter 10 rows, each contains one Real value. Then download this "data51.txt" to WP-8xx7's "\ System_Disk \" folder by "ftp" utility. The content of the "data51.txt" looks like below.

```
2.345
999.03
-1.01
456.789
2
456.77
5.9E-12
32.3
45.1
33.3
```

Variables:

Name	Type	Attribute	Description
RE_LOAD	Bool	Internal	Set as True to read File once, init as TRUE
TMP	Bool	Internal	Internal use
File_name1	Message	Internal	Len is 64, init as \System_Disk\data51.txt
Msg1	Message	Internal	Len is 128, File processing state
str1	Message	Internal	Len is 255, internal use
F_VAL[0..9]	REAL	Internal	Variable array, Dim is 10. The 10 REAL value.
TMP_F	REAL	Internal	Internal use
File1	Integer	Internal	File ID
ii	Integer	Internal	Index of "for" loops

ST program:

```
if  RE_LOAD  then       (* Read file once if "RE_LOAD" is TRUE  *)

 RE_LOAD  :=  FALSE  ;

 File1 := f_wopen( File_name1 )  ;      (*  Open file in Read & Write mode *)

 if  File1 = 0  then       (* 0: open file fail *)
  Msg1  :=  'Can not Open file '  +  File_name1 ;
  return ;      (* Cannot open file, just exit this ST program *)
 end_if  ;

 for  ii  :=  0  to  9  do       (* Total 10 rows *)

   if   f_eof(File1)   =   TRUE    then       (* test if reaches the End-Of-File *)
     Msg1  :=  'Data number is not enough in '  +  File_name1 ;
     Exit  ;       (* Exit this "for" loops *)
   end_if  ;

   str1  :=  fm_read(File1)  ;      (* read one string in the File *)
   TMP_F  :=  str_real(str1) ;      (* convert string to a REAL value *)
   if  TMP_F  =  1.23E-20   then       (* if returns 1.23E-20, it means format error  *)
     Msg1  :=  'The '  +  Msg(ii+1)  +  'th Data format is not correct !'  ;
     exit  ;      (* Exit this "for" loops *)
   end_if ;

   F_VAL[ii]  :=  TMP_F  ;      (* Read & Convert Ok. Store value to F_VAL[0..9] *)

 end_for ;

 TMP  :=  f_close(File1)  ;       (* always close File after its operation *)
 If   ii=10   then       (* All data is succefully read and converted, 10 rows *)
   Msg1 := 'Read ' + File_name1 + ' Ok ' ;
 end_if ;

end_if  ;
```

10.5.2: Wpdmo_54: Read 20 REAL values from a file. Total 4 rows, each contains 5 REAL values

The "Wpdmo_54.pia" can be found at
WP-8xx7 CD-ROM:\napdos\isagraf\wp-8xx7\demo\ or
ftp://ftp.icpdas.com/pub/cd/winpac-8xx7/napdos/isagraf/wp-8xx7/demo/

If functions of Msg_F , Msg_N , ARY_F_R, AFY_F_W are not found in your PC / ISaGRAF, please download "ICP DAS utilities For ISaGRAF" at
http://www.icpdas.com/products/PAC/i-8000/isagraf.htm > Driver . Then run "setup.exe" to restore them to your ISaGRAF workbench

The "Wpdmo_54" program will read 20 REAL values from "\System_Disk\data54.txt" when the WP-8xx7 is just power up or user set the "RE_LOAD" value to become "TRUE" at any time .
(To read / write file in the System_Disk take lots of CPU time, please do not read / write it frequently. And please always close the file after the operation. If user read / write file in every PLC scan cycle, the PLC scan time will become a very big time. It will be a bad performance !)

To test this sample program, please edit a text file "data54.txt" in your PC by , for example – "Notepad". Please enter 4 rows, each contains 5 Real values. Then download this "data54.txt" to WP-8xx7's "\ System_Disk\" folder by "ftp" utility. The content of the "data54.txt" looks like below.

```
23 ,  65.9 , 0.012 , 5.87 , 88.2
0.34 , 8.0005 , -2.0E8 ,  4.08 , 5.32E-6
2 ,  -7 ,  6666.8 , 456.07 ,  1.01
5 ,  6 , 7 , 8 ,  9
```

Variables:

Name	Type	Attribute	Description
RE_LOAD	Bool	Internal	Set as True to read File once, init as TRUE
TMP	Bool	Internal	Internal use
File_name1	Message	Internal	Len is 64, init as \System_Disk\data54.txt
Msg1	Message	Internal	Len is 128, File processing state
str1	Message	Internal	Len is 255, internal use
F_VAL[0..19]	REAL	Internal	Variable array, Dim is 20. The 20 REAL value
NUM1	Integer	Internal	Get return of Msg_F( ), -1 means format error
File1	Integer	Internal	File ID
ii	Integer	Internal	Index of "for" loops
jj	Integer	Internal	Index of another "for" loops

ST program:

```
if RE_LOAD then        (* Read file once if "RE_LOAD" is TRUE *)

  RE_LOAD := FALSE ;

  File1 := f_wopen( File_name1 ) ;      (* Open file in Read & Write mode *)

  if File1 = 0 then      (* 0: open file fail *)
    Msg1 := 'Can not Open file ' + File_name1 ;
    return ;      (* Cannot open file, just exit this ST program *)
  end_if ;

  for ii := 0 to 3 do      (* total 4 rows *)

     if f_eof(File1) = TRUE then      (* test if reaches the End-Of-File *)
        Msg1 := 'There should be at least 4 rows in ' + File_name1 + ' !!!' ;
        Exit ;      (* exit this "for" loops *)
     end_if ;

     str1 := fm_read(File1) ;      (* read one row as string from file *)

    (* Convert string to become serval REAL values and store them into No. 1 Float array *)
     NUM1 := Msg_F(str1 , 1) ;

    (* If the amount of the converted REAL values is not 5 , it lacks of data. -1 means format error *)
     if NUM1 <> 5 then
        Msg1 := 'The ' + Msg(ii+1) + 'th row data format is not correct or data number is not 5 !' ;
        Exit ;      (* exit this "for" loops *)
     end_if ;

     for jj := 0 to 4 do
        (* Get 5 REAL values from No.1 Float array's addr=1 to 5 , and store them to F_VAL[0..19] *)
        F_VAL[ 5 * ii + jj] := ARY_F_R( 1 , jj + 1 ) ;
     end_for ;

  end_for ;

  TMP := f_close(File1) ;   (* always close File after its operation *)

  If ii = 4 then      (* All data is succefully read and converted, 4 rows *)
     Msg1 := 'Read ' + File_name1 + ' Ok ' ;
  end_if ;

end_if ;
```

10.5.3: Wpdmo_55: Read 20 Integer values from a file. Total 2 rows, each contains 10 Integer values

The "Wpdmo_55.pia" can be found at
WP-8xx7 CD-ROM:\napdos\isagraf\wp-8xx7\demo\ or
ftp://ftp.icpdas.com/pub/cd/winpac-8xx7/napdos/isagraf/wp-8xx7/demo/

If functions of Msg_F , Msg_N , ARY_F_R, AFY_F_W are not found in your PC / ISaGRAF, please download "ICP DAS utilities For ISaGRAF" at
http://www.icpdas.com/products/PAC/i-8000/isagraf.htm > Driver. Then run "setup.exe" to restore them to your ISaGRAF workbench

The "Wpdmo_55" program will read 20 Integer values from "\System_Disk\data55.txt" when the WP-8xx7 is just power up or user set the "RE_LOAD" value to become "TRUE" at any time .
(To read / write file in the System_Disk take lots of CPU time, please do not read / write it frequently. And please always close the file after the operation. If user read / write file in every PLC scan cycle, the PLC scan time will become a very big time. It will be a bad performance !)

To test this sample program, please edit a text file "data55.txt" in your PC by , for example – "Notepad". Please enter 2 rows, each contains 10 Integer values. Then download this "data55.txt" to WP-8xx7's "\System_Disk\" folder by "ftp" utility. The content of the "data55.txt" looks like below.

```
-1 , 1 , 2 , 3 , 4 , 5 , -6 , 7 , 8 , 9
100001 , 200002 , +300003 , 404 , -505 , 606 , 7007 , 8008 , 9009 , 10
```

Variables:

Name	Type	Attribute	Description
RE_LOAD	Bool	Internal	Set as True to read File once, init as TRUE
TMP	Bool	Internal	Internal use
File_name1	Message	Internal	Len is 64, init as \System_Disk\data55.txt
Msg1	Message	Internal	Len is 128, File processing state
str1	Message	Internal	Len is 255, internal use
N_VAL[0..19]	Integer	Internal	Variable array, Dim is 20. The 20 Integer values
NUM1	Integer	Internal	Get return of Msg_N( ), -1 means format error
File1	Integer	Internal	File ID
ii	Integer	Internal	Index of "for" loops
jj	Integer	Internal	Index of another "for" loops

ST program:

```
if  RE_LOAD  then     (*Read file once if "RE_LOAD" is TRUE *)

  RE_LOAD  :=  FALSE  ;

  File1 := f_wopen( File_name1 )  ;    (*  Open file in Read & Write mode *)

  if  File1 = 0  then     (* 0: open file fail *)
    Msg1  :=  'Can not Open file '  +  File_name1 ;
    return ;    (* Cannot open file, just exit this ST program *)
  end_if  ;

  for  ii   :=   0   to   1   do   (* total 2 rows *)

    if   f_eof(File1)  =  TRUE  then    (*test if reaches the End-Of-File *)
      Msg1   :=   'There should be at least 2 rows in '  +  File_name1  + ' !!!'  ;
      Exit  ;   (* exit this "for" loops *)
    end_if  ;

    str1  :=  fm_read(File1)  ;    (* read one row as string from file *)

    (* Convert string to become serval Integer values and store them into No. 2 Integer array *)
    NUM1  :=  Msg_N(str1 , 2)  ;

   (* If the amount of the converted Integer values is not 10 , it lacks of data. -1 means format error *)
    if   NUM1  <>  10   then
      Msg1 :=  'The '  + Msg(ii+1)  +  'th row data format is not correct or data number is not 10 !'  ;
      Exit  ;    (* exit this "for" loops *)
    end_if  ;

    for  jj   :=   0  to   9   do

      (* Get 10 Integer values from No.2 Integer array's addr=1 to 10 , and store them to N_VAL[0..19] *)
      N_VAL[ 10 * ii + jj]  :=  ARY_N_R( 2 ,  jj + 1 )  ;
    end_for ;

  end_for ;

  TMP   :=   f_close(File1)  ;   (*always close File after its operation *)

  If   ii = 2  then   (*All data is succefully read and converted, 2 rows *)
    Msg1  :=  'Read '  +  File_name1  +  ' Ok '  ;
  end_if  ;

end_if  ;
```

10.5.4: Wpdmo_56: Retain values of 1 to 255 Real variable in CompactFlash card

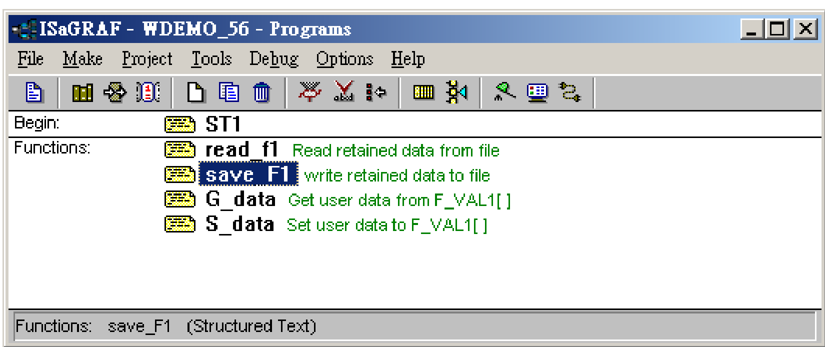
The “Wpdmo_56.pia” can be found at
WP-8xx7 CD-ROM:\napdos\isagraf\wp-8xx7\demo\ or
ftp://ftp.icpdas.com/pub/cd/winpac-8xx7/napdos/isagraf/wp-8xx7/demo/

If functions of Msg_F , Msg_N , ARY_F_R, AFY_F_W are not found in your PC / ISaGRAF, please download “ICP DAS utilities For ISaGRAF” at
http://www.icpdas.com/products/PAC/i-8000/isagraf.htm > Driver. Then run “setup.exe” to restore them to your ISaGRAF workbench.

The “Wpdmo_56” program will read 1 to 255 REAL values from “\System_Disk\data56.txt” to the related variable when the WP-8xx7 is just power up . If this “data56.txt” doesn’t exist, all 1 to 255 values will be init as 0.0 . At run time, if any value of these variable is modified, all 1 to 255 values will be written once to the “data56.txt” to make sure these variable’s value are well retained in file. If the file doesn’t exist, this program will create it.

To read / write file in the System_Disk take lots of CPU time, please do not read / write it frequently. And please always close the file after the operation. If user read / write file in every PLC scan cycle, the PLC scan time will become a very big time. It will be a bad performance ! If user need fast retain function, please refer to Chapter 10.1 for retaining data in the S256/S512.

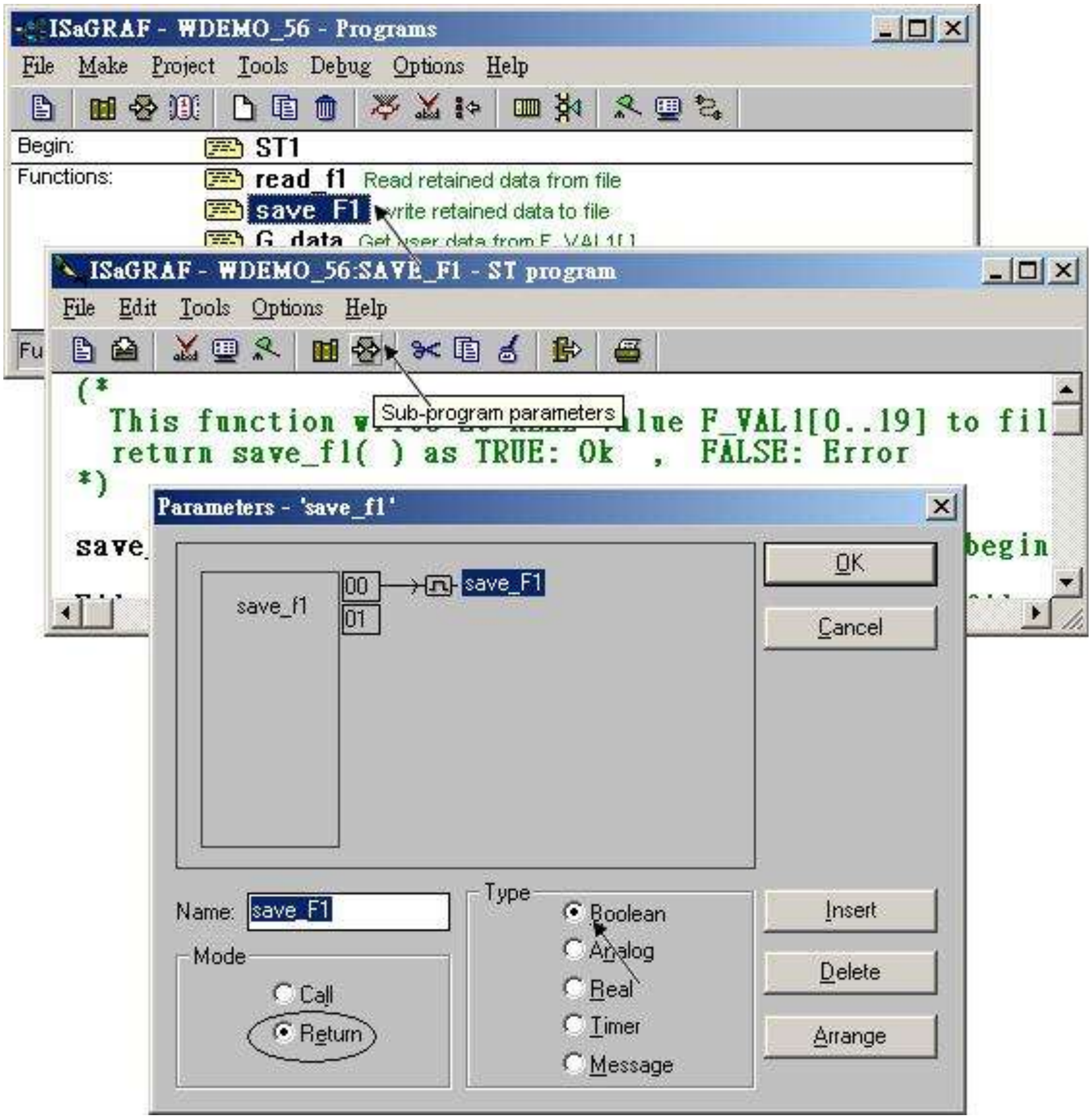
Project Architecture:



There are 5 ST programs in this “Wpdmo_56” project. Four of them are ISaGRAF user-defined functions – “reaf_f1” , “save_f1” , “G_data” and “S_data” .

Important note:
1. User may modify the constant value of “SIZE1” in the ISaGRAF “dictionary” window to a value between 1 ~ 255 according his own application. And then remember to compile it .
2. Please also modify the “Dim” value of the “F_VAL1[ ]” and “Old_F_VAL1[ ]” variable array in the ISaGRAF “dictionary” window to become the same value as the “SIZE1” . And also please modify the “G_data” and “S_data” program.
3. There is one advantage of retaining vale in the System_Disk. The data file can be edited in advance in PC. Then using “ftp” utility to download it to WP-8xx7. The file path name of this example is “\ System_Disk \data56.txt” . Then set “RE_LOAD” value to TRUE once, all related variable will update to the new value.

The following ST programs are all declared as functions. They are “read_f1” , “save_f1” , “G_data” and “S_data” . They all return a Boolean value. Please refer to below figure to declare function’s return-value type (more description in the Chapter 15)



The “read_f1” and “save_f1” program use “local variables” as below .

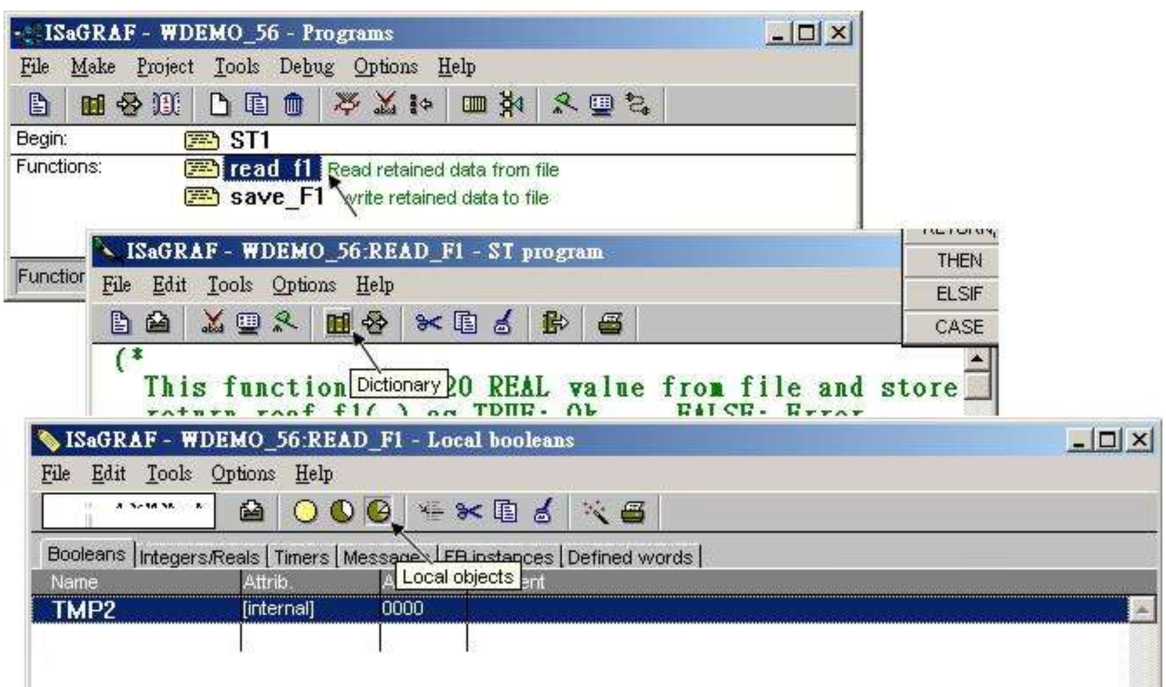
read_f1 :

Name	Type	Attribute	Description
TMP2	Bool	Internal	Internal use
ii2	Integer	Internal	Index of “for” loops
jj2	Integer	Internal	Index of “for” loops
num2	Integer	Internal	Internal use

save_f1 :

Name	Type	Attribute	Description
TMP2	Bool	Internal	Internal use
ii2	Integer	Internal	Index of “for” loops
jj2	Integer	Internal	Index of “for” loops
num2	Integer	Internal	Internal use

To declare “local variable”, please double click “read_f1” to get into this program. Then get into the “Dictionary” window. Then click on “Local objects” to declare them.



Variables (Global variable) :

Name	*Type*	*Attribute*	*Description*
SIZE1	Integer	Constant	The number of retained variables. Can be 1 to 255 . Please also modify the "Dim" value of the "F_VAL1[ ]" and "Old_F_VAL1[ ]" to the same value as "SIZE1". Here we use "SIZE1" as 17.
num_row1	Integer	Internal	How many rows in the file ? This value is automatically calculated by "SIZE1". Each row should have 10 REAL values, except the last row.
Last_num1	Integer	Internal	How many data in the last row ? This value is automatically calculated by "SIZE1".
RE_LOAD	Bool	Internal	Set as True to read File once, init as TRUE
TMP	Bool	Internal	Internal use
Data_Ok1	Bool	Internal	TRUE means File Ok
Flag_to_save	Bool	Internal	If program want to save data, it will set this value to TRUE.
File_name1	Message	Internal	Len is 64, init as \System_Disk\data56.txt
Msg1	Message	Internal	Len is 128, File processing state
str1	Message	Internal	Len is 255, Internal use
F_VAL1[0..16]	REAL	Internal	Variable array, "Dim" should be init as the same value as "SIZE1"
Old_F_VAL1 [0..16]	REAL	Internal	Old value of "F_VAL1[ ]" Variable array, "Dim" should be init as the same value as "SIZE1" .
NUM1	Integer	Internal	Get return of Msg_F( ), -1 means format error
File1	Integer	Internal	File ID
ii	Integer	Internal	Index of "for" loops
jj	Integer	Internal	Index of "for" loops
Data1 ~ Data5 And Data06 ~ Data17	REAL	Internal	The User Data variable. Here we have 17 variables in the demo program. User can declare them to different variable name. If name is modified, the "G_data" and the "S_data" program should be modified also.

ST program - ST1:

```
-----------------------------------------------------------------------------------------------------------------
if   RE_LOAD   then       (* if RE_LOAD is TRUE, get retained data from file *)

 RE_LOAD   :=   FALSE  ;    (* Set RE_LOAD as FALSE *)

 (* caculate number of rows and data number of the last row *)
 num_row1  :=  SIZE1 / 10  ;
 last_num1  :=  SIZE1 -  10  *  num_row1 ;
 if   last_num1  <>  0   then
   num_row1  :=  num_row1 + 1  ;     (* if last_row has data, num_row1 must plus 1 *)
 else
   last_num1  :=  10 ;
 end_if  ;

 (* Get retained value from file when controller is powered up *)
 TMP  :=  read_F1(  )  ;

 if   TMP = FALSE   then    (* Read file error or file not exist *)

  for   ii  :=  0   to   SIZE1 - 1   do
     F_VAL1[ ii ]  :=  0.0  ;      (* set all F_VAL1[ ] 's value as 0.0 *)
  end_for ;

  Data_Ok1  :=  FALSE  ;     (* set data is not Ok *)
  Msg1  :=  'File : '  +  File_name1  +  ' not exist or data error ! or File is open now'  ;

 else    (* Read data Ok *)

  Data_Ok1  :=  TRUE  ;     (* set data is Ok *)
  Msg1  :=  'Get Retained data from file Ok '  ;

 end_if  ;

 (* Update Old_F_VAL1[ ] *)
 for    ii   :=   0   to    SIZE1 - 1    do
  Old_F_VAL1[ ii ]   :=   F_VAL1[ ii ]  ;
 end_for  ;

 (* Get user data from F_VAL1[ ] when controller is just powered up *)
 TMP  :=  G_DATA( ) ;

end_if  ;

(* At run time, Set user data to F_VAL1[ ] *)
TMP  :=  S_DATA(  )  ;
```

```
(* At run time, test any value of F_VAL1[ ] is modified *)
for   ii  :=  0  to  SIZE1 - 1  do

  if  Old_F_VAL1[ ii ]  <>  F_VAL1[ ii ]   then     (* if any value is modified *)
    Flag_to_save  :=  TRUE  ;     (* now save command is given *)
    Old_F_VAL1[ ii ]  :=  F_VAL1[ ii ]  ;    (* Update Old_F_VAL1[ ] if it is modified *)
  end_if  ;

end_for  ;

(* if save command is given, it means value is modified *)
if   Flag_to_save   then

  TMP  :=  save_f1(  )  ;     (* save data to file *)

  (* if save file failed, keep this save command *)
  if   TMP  =  FALSE   then
    Msg1  :=  'Can not save data to file. May be file is open now by WinCon screen ! '  ;

  (* Save Ok, cancel this save command *)
  else
   Flag_to_save  :=  FALSE  ;     (* Set as "No save" at the beginning *)

  end_if  ;

end_if  ;
----------------------------------------------------------------------------------------------------------------
```

ST functions – G_data :

```
------------------------------------------------------------------------------------------------------------------
(* If any name of  Data1 to Data17 is modified or value of "SIZE1" is modified, User must modify the
below code *)
Data1    :=  F_VAL1[0] ;      (* get variable value from F_VA1L[0..16] *)
Data2    :=  F_VAL1[1] ;
Data3    :=  F_VAL1[2] ;
Data4    :=  F_VAL1[3] ;
Data5    :=  F_VAL1[4] ;
Data06   :=  F_VAL1[5] ;
Data07   :=  F_VAL1[6] ;
Data08   :=  F_VAL1[7] ;
Data09   :=  F_VAL1[8] ;
Data10   :=  F_VAL1[9] ;
Data11   :=  F_VAL1[10] ;
Data12   :=  F_VAL1[11] ;
Data13   :=  F_VAL1[12] ;
Data14   :=  F_VAL1[13] ;
Data15   :=  F_VAL1[14] ;
Data16   :=  F_VAL1[15] ;
Data17   :=  F_VAL1[16] ;
G_data  :=  TRUE   ;       (* function  returns TRUE *)
------------------------------------------------------------------------------------------------------------------
```

ST functions – S_data :

```
------------------------------------------------------------------------------------------------------------------
(*If any name of  Data1 to Data17 is modified or value of "SIZE1" is modified, User must modify the
below code *)
F_VAL1[0]  :=  Data1 ;     (* store variable value to F_VAL1[0..16] *)
F_VAL1[1]  :=  Data2 ;
F_VAL1[2]  :=  Data3 ;
F_VAL1[3]  :=  Data4 ;
F_VAL1[4]  :=  Data5 ;
F_VAL1[5]  :=  Data06 ;
F_VAL1[6]  :=  Data07 ;
F_VAL1[7]  :=  Data08 ;
F_VAL1[8]  :=  Data09 ;
F_VAL1[9]  :=  Data10 ;
F_VAL1[10]  :=  Data11 ;
F_VAL1[11]  :=  Data12 ;
F_VAL1[12]  :=  Data13 ;
F_VAL1[13]  :=  Data14 ;
F_VAL1[14]  :=  Data15 ;
F_VAL1[15]  :=  Data16 ;
F_VAL1[16]  :=  Data17 ;
S_data  :=  TRUE ;          (* function returns TRUE *)
------------------------------------------------------------------------------------------------------------------
```

ST functions - read_f1 :

```
---------------------------------------------------------------------------------------------------------------------
(*   This function read "SIZE1" number of REAL value from file and store them to F_VAL1[ ]
  return reaf_f1( ) as TRUE: Ok  ,  FALSE: Error  *)
read_f1  :=  FALSE  ;                    (* set as FALSE: Error at the beginning *)
File1  :=  f_wopen( File_name1 )  ;   (* Try to open file in Read & Write mode *)

if  File1 = 0  then      (* File doesn't exists *)
   return  ;    (* exit this function *)
end_if  ;

 (* max  "num_row1"  rows to read these "SIZE1" number of REAL values, Each row in the file contains
10 REAL values *)
for  ii2  :=  0  to  num_row1 - 1  do

  if  f_eof( File1 )  =  TRUE   then   (* test if End_Of_File reached *)
    exit  ;       (* Reach End Of File,  exit "for" loop *)
  end_if  ;

  str1  :=  fm_read( File1 )  ;   (* Read one row as String (message) *)

  (* Convert this string to some REAL values and store them into No.1 Float array *)
  NUM1  :=  Msg_F( str1 , 1 ) ;

  (* if data number of last row is not correct *)
  if  ( ( ii2 = num_row1 – 1 )   and   ( NUM1  <>  last_num1 )  )  or

     (* non-last row must have 10 REAL values *)
     ( ( ii2  <>  num_row1 – 1 )   and   ( NUM1  <>  10 )  )    then

    (* error, it means the format is not correct REAL values or data number is not enough *)
    exit ;      (* exit for loop *)

  end_if;

  (* conversion Ok, store these REAL values to F_VAL1[ ] *)
  if   ii2  =  num_row1 - 1   then      (* last row *)
     num2  :=  last_num1  ;        (* last row has only "last_num1" number of data *)
  else
     num2 := 10 ;    (* non-last row has 10 data *)
  end_if ;
  (* Get these converted REAL values from No.1 Float array 's addr. 1 to 10 (or 1 to last_num1 for last
row) *)
  for   jj2  :=  0   to   num2 - 1   do
      F_VAL1[ 10*ii2  +  jj2 ]   :=   ARY_F_R( 1 ,  jj2  +  1 )  ;
  end_for  ;

end_for  ;
```

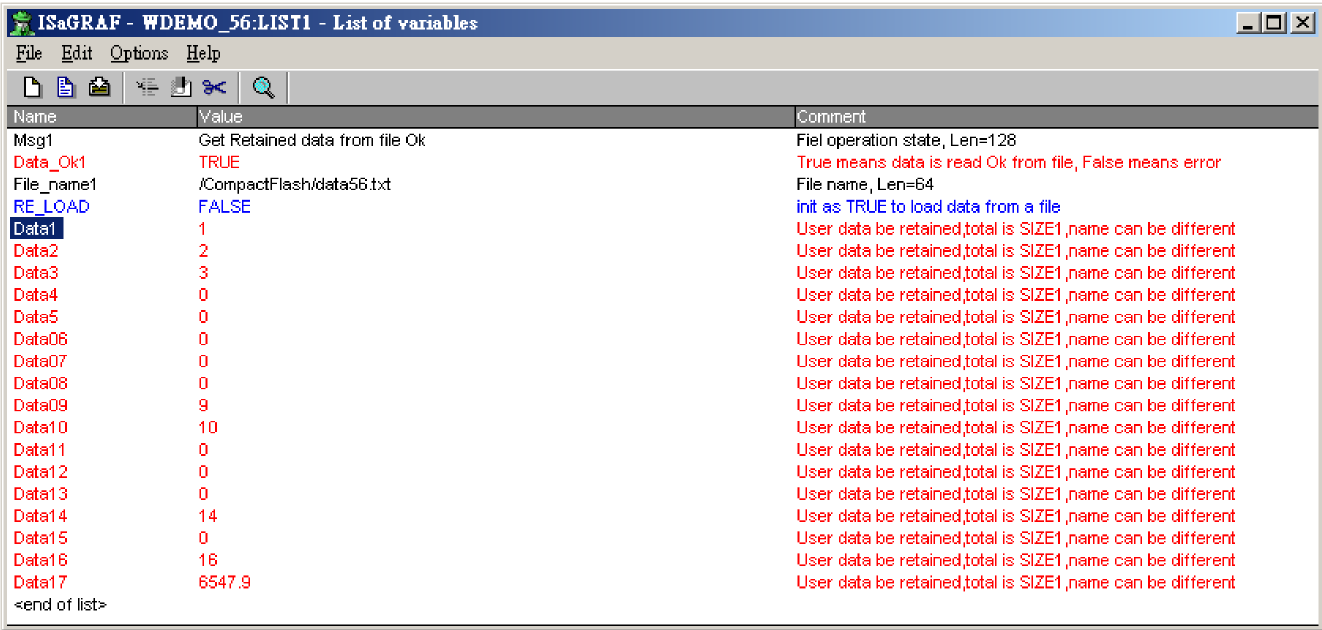
```
(* Any file been open should be closed by f_close( ) *)
TMP2  :=  f_close( File1 )  ;

(* All rows are read Ok *)
if   ii2  =  num_row1    then
   read_F1   :=   TRUE  ;      (* return value as TRUE:Ok *)
end_if  ;
```

ST functions – save_f1 :

```
(*  This function write 20 REAL value F_VAL1[0..19] to file
    return save_f1( ) as TRUE: Ok  ,  FALSE: Error  *)
save_f1  :=  FALSE  ;   (* set as FALSE: Error at the beginning *)
File1  :=  f_creat( File_name1 )  ;   (* Creat a new file to write *)

if   File1 = 0 then
  return ;  (*creat failed , exit this function *)
end_if  ;

(* max "num_row1"  rows to save these REAL values, Each row in the file contains 10 REAL values *)
for  ii2  :=  0  to  num_row1 - 1  do

  str1  :=  ' '  ;   (* set initial value of str1 *)

  if   ii2  =  num_row1 – 1   then        (* last row *)
   num2   :=   last_num1 ;      (* last row has only "last_num1" number of data *)
  else      (* non-last row *)
   num2  := 10 ;    (* non-last row has 10 data *)
  end_if  ;

  for    jj2  :=  0  to  num2 - 2    do
   str1  :=  str1  +  REAL_STR(  F_VAL1[ 10 * ii2  +  jj2 ]  )  +  ' , '   ;
  end_for ;

  (* the last data in each row should end with <CR> <LF> character *)
  str1  :=  str1  +  REAL_STR(  F_VAL1[ 10 * ii2  +  num2 - 1]  )  +  '$0D$0A'  ;
  TMP2  :=  f_writ_s( File1  , str1 )  ;     (* write one row to file *)

end_for ;

(* Any file been open should be closed by f_close( ) *)
TMP2  :=  f_close( File1 ) ;

save_f1  :=  TRUE  ;   (* return value as TRUE:Ok *)
```

How to test this “Wpdmo_56” project ?

1. Please download “Wpdmo_56” to WP-8xx7, then the “Spy list” window will pop-up as below.



Name	Value	Comment
Msg1	Get Retained data from file Ok	Fiel operation state, Len=128
Data_Ok1	TRUE	True means data is read Ok from file, False means error
File_name1	/CompactFlash/data56.txt	File name, Len=64
RE_LOAD	FALSE	init as TRUE to load data from a file
Data1	1	User data be retained,total is SIZE1 ,name can be different
Data2	2	User data be retained,total is SIZE1 ,name can be different
Data3	3	User data be retained,total is SIZE1 ,name can be different
Data4	0	User data be retained,total is SIZE1 ,name can be different
Data5	0	User data be retained,total is SIZE1 ,name can be different
Data06	0	User data be retained,total is SIZE1 ,name can be different
Data07	0	User data be retained,total is SIZE1 ,name can be different
Data08	0	User data be retained,total is SIZE1 ,name can be different
Data09	9	User data be retained,total is SIZE1 ,name can be different
Data10	10	User data be retained,total is SIZE1 ,name can be different
Data11	0	User data be retained,total is SIZE1 ,name can be different
Data12	0	User data be retained,total is SIZE1 ,name can be different
Data13	0	User data be retained,total is SIZE1 ,name can be different
Data14	14	User data be retained,total is SIZE1 ,name can be different
Data15	0	User data be retained,total is SIZE1 ,name can be different
Data16	16	User data be retained,total is SIZE1 ,name can be different
Data17	6547.9	User data be retained,total is SIZE1 ,name can be different
<end of list>		

You may modify any value of USER Data - Data1 to Data17. Then the new value will be saved once into file of “\System_Disk\data56.txt”. Then you can open this file on the WP-8xx7’s monitor screen by double click on the file name. You will see the related value is modified. (Please do not always keep this file open. Please close it later, or the new modified data will not be saved . That is because the file is open, write operation is not allowed)

2. Recycle the power of WP-8xx7. You will see the value keep at the last modified value when WinCon is boot up well.

3. Edit a “data56.txt” file on PC as below by “NotePad” utility. (total 17 data)

1.1 , 2.2 , 3.3 , 4.4 , 5.5 , 6.66 , 7.77 , 8.88 , 9.99 , 10.01
0.01 , 0.02 , 0.03 , 0.04 , 0.05 , 0.06 , 0.07

Then please download this “data56.txt” file to WP-8xx7’s \System_Disk\ path by “ftp” utility. Then set “RE_LOAD” to become TRUE on ISaGRAF “Spy list” window. You will see the related variable value is updated.

10.5.5: Record I-8017H 's Ch.1 to Ch.4 voltage input in a user allocated RAM memory in the ISaGRAF PAC ? The sampling time is one record every 0.01 second. The record period is 1 to 10 minutes. Then PC can download this record and display it as a trend curve diagram by M.S. Excel.

Please refer to Chapter 11.3.6 (the fastest sampling rate is 25 Hz)
Please refer to Chapter 11.3.10 (the fastest sampling rate is 100 Hz)

10.6: Controller Fault Detection

There is some event may cause “controller fault” happens. For example, value divided by zero or reading a floating point value from EEPROM or S256 or file which has no floating point value saved inside.

ICP DAS ISaGRAF controllers support Controller Fault detection since below driver version.
(The VP-2117, μPAC-7186EG, μPAC-5xx7, iP-8xx7, WP-8xx7, WP-5xx7,VP-25W7/23W7, XP-8xx7-CE6, XP-8xx7-Atom-CE6 are supported.)

I-7188EG	2.05	I-7188XG	2.04
I-8417/8817/8437/8837	3.07	W-8037/88337/8737	3.18

There is two type of controller fault. One is called “Global” fault. The other is “Local” fault.
When Global fault happens, the ISaGRAF project will stop running. Waiting the new modified project to be downloaded. When Local fault happens, the ISaGRAF project still runs.

PC/HMI/OPC Server can request the controller fault state by using Modbus protocol.

Word address of 9999 is the controller fault state. 0: Ok , 1: Controller fault.
R_MB_ADR(1, 9999) to get controller_state

Word address of 9998 is the controller fault type.
R_MB_ADR(1, 9998) to get fault_type.

101 : **Global fault**

(other value is Local fault)
102 : S_R_R Float error
103 : R_MB_REL Float error
104 : INT_REAL Float error
105 : RETAIN_F Float error
106 : RETAIN_X Float error
107 : Real value divided by 0
108 : Integer value divided by 0
109 : RETAIN_A Float error
110 : Real value multiplication is overflow (exceeds valid range of 32-bit float)
111 : Real value division is overflow (exceeds valid range of 32-bit float)
112 : Real value addition is overflow (exceeds valid range of 32-bit float)
113 : Real value subtraction is overflow (exceeds valid range of 32-bit float)
114 : EEP_F_R Float error
115 : EBUS_F_R Float error
116 : FBUS_F_R Float error
117 : F_REAF_F Float error (Only in WinCon-8xx7)
118 : Can not find I-87K I/O board in slot 0 to 7
119 : ARY_F_R Float error
121 : ANA( ) operation error. For ex, ANA(1.23E20) , ANA(-2.0e25)
122 : TMR( ) operation error. For ex, TMR(1.23E20) , TMR(-100)
123 : Floatng point calculation error. For ex, pow(1.23E20 , 3.0) , expt(5.0, 10000000)
124 : PID_AL( ) floating point calculation error. (exceeds valid range of 32-bit float)
125 : REAL “Variable array” float error. It may be the array index out of the declared range

When Local fault happens, the project is still running, the ISaGRAF project can use

R_MB_ADR(1, 9999) to get controller_state
R_MB_ADR(1, 9998) to get fault_type.

To clear the value in Network address 9999 & 9998, please use **W_MB_ADR(1, 9999, 0)** and **W_MB_ADR(1, 9998, 0)**. Please refer to below example.

Example:

```
(* When controller "Local Fault" happens, the ISaGRAF program can detect it and then program can do
the right action *)

(* is_fault & fault_type are declared as internal integer *)
(* tmp is declared as internal boolean *)
(* PC / HMI can request controller fault state & type by Modbus protocol at No.=9999 & 9998 *)

(* to get controller state *)
is_fault := R_MB_ADR(1, 9999) ;     (* 0: Ok ,  1: controller fault happens *)

(* To get controller fault type *)
fault_type := R_MB_ADR(1, 9998) ;

if  is_fault=1  then

  (* Do action here when  "Local Fault" happens *)
  (* ...                                        *)

  (* Only for WinCon-8x37: Stop program running & reset all output in slot 1 to 7 *)
  (*  tmp := Stop_APL( );  *)

  (* To clear the value in Network address 9999 & 9998 when Local fault happens *)
    tmp :=  W_MB_ADR(1, 9999, 0) ;
    tmp :=  W_MB_ADR(1, 9998, 0) ;

end_if ;
```

Chapter 11. ISaGRAF Programming Examples & FAQ

When you receive the ISaGRAF controller system, ICP DAS has created a number of ISaGRAF programming examples for them. These example programs are useful for understanding how to program the controller system with the ISaGRAF Workbench software program.

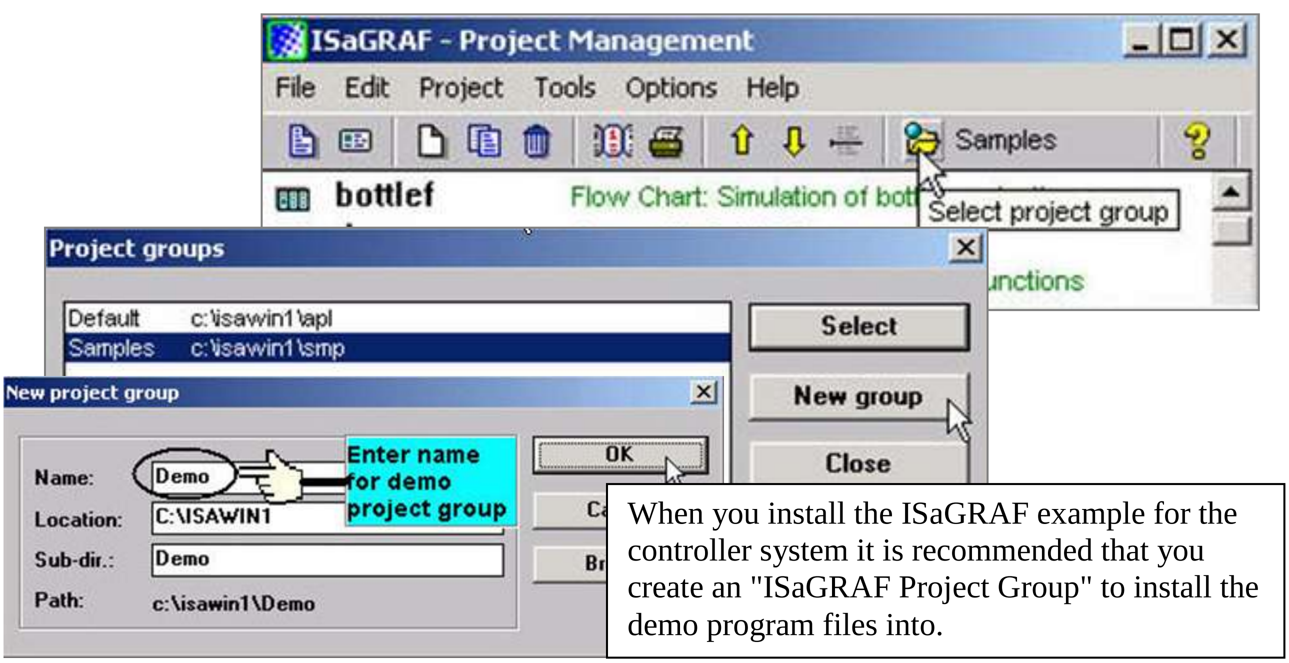
11.1: Installing The ISaGRAF Programming Examples

The ISaGRAF programming examples are installed on the same CD-ROM which the “ICP DAS Utilities For ISaGRAF” resides. The CD-ROM is delivered with the product. You will find the programming example files in the below sub-directory in the CD-ROM.

I-8xx7:	I-8000 CD-ROM: \napdos\isagraf\8000\demo\
I-7188EG, µPAC-7186EG:	I-8000 CD-ROM: \napdos\isagraf\7188eg\demo\
I-7188XG:	I-8000 CD-ROM: \napdos\isagraf\7188xg\demo\
iP-8xx7:	I-8000 CD-ROM: \napdos\isagraf\iP8000\demo\
VP-2117:	I-8000 CD-ROM: \napdos\isagraf\vp2k\demo\
XP-8xx7-CE6:	XP-8xx7-CE6 CD-ROM: \napdos\isagraf\xp-8xx7-ce6\demo\
XP-8xx7-Atom-CE6:	XP-8xx7-Atom-CE6 CD-ROM: \napdos\isagraf\xp-8xx7-Atom-ce6\demo\
WP-8xx7:	WP-8xx7 CD-ROM: \napdos\isagraf\wp-8xx7\demo\
WP-5xx7:	WP-5xx7 CD-ROM: \napdos\isagraf\wp-5xx7\demo\
VP-25W7/VP-23W7:	VP-2xW7 CD-ROM: \napdos\isagraf\vp-25w7-23w7\demo\

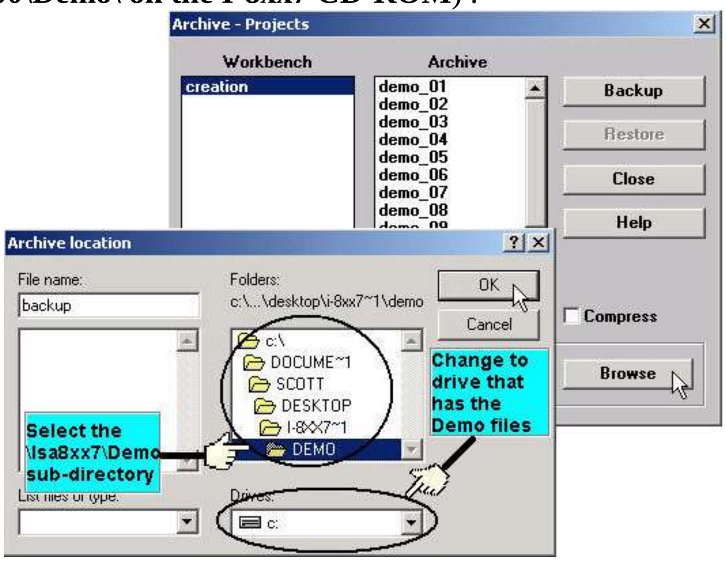
Or you may download them from below web site:

µPAC-7186EG:	ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/7188eg/demo/
µPAC-5xx7:	ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/up5000/demo/
iP -8xx7 :	ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/iP-8000
VP-2117:	ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/vp2k/demo/
XP-8xx7-CE6:	ftp://ftp.icpdas.com/pub/cd/xp-8xx7-ce6/napdos/isagraf/xp-8xx7-ce6/demo/
WP-8xx7:	ftp://ftp.icpdas.com/pub/cd/winpac-8xx7/napdos/isagraf/wp8xx7/demo/
VP-25W7/VP-23W7:	ftp://ftp.icpdas.com/pub/cd/vp-25w7-23w7/napdos/isagraf/vp-25w7-23w7/demo/

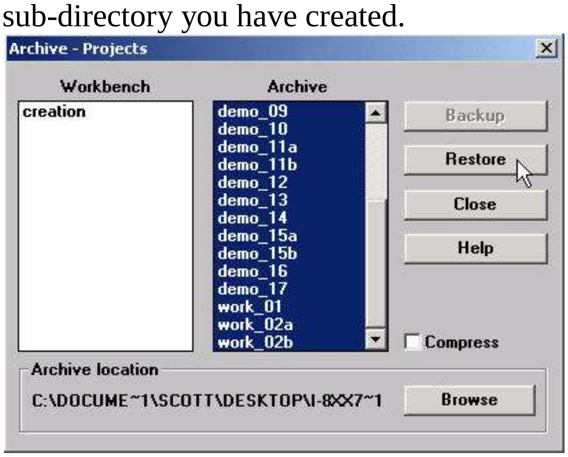


To install the demo programs into the project you have created, open the "ISaGRAF Project Management" window to select "Tools" from the menu bar, then select the "Archive" option and then click on "Projects".

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 (**For example: Napdos\ISaGRAF\8000\Demo\ on the I-8xx7 CD-ROM**) .



To install all of the Demo files, click on the "demo_01" 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.



11.2: ISaGRAF Demo Example Files

The example program for VB, µPAC-7186EG, µPAC-5xx7, iP-8xx7, WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6 and XP-8xx7-CE6 are listed in this section. For other PAC, please refer to their respective "Getting Started Manual".

http://www.icpdas.com/products/PAC/i-8000/isagraf_demo_list.htm#VBNET
www.icpdas.com > product > solutions > software > Development Tools > ISaGRAF > Demo Files

Visual Basic example program:

I-8000 CD-ROM:\napdos\isagraf\vb_demo\
ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/vb_demo

Demo_1	PC Read / Write data in the I-8437/8837 by Modbus TCP/IP	I-8437/8837 I-8054
Demo_2	PC use Modem + phone line to link to remote I-8437/8837 (please refer Chapter 13)	I-84x7/88x7 I-87064 Modem Phone line
Demo_3	PC run "VB.net 2005" or "VB 6.0" program to Read / Write data in the contoller (I-8x37-80, I-7188EG, µPAC-7186EG, µPAC-5xx7, VP-2117, iP-8xx7, WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6, or VP-25W7/ VP-23W7 by Modbus TCP/IP protocol. Please refer to http://www.icpdas.com/faq/isagraf.htm 051 & 052	
Demo_4	PC run "VB 6.0" program to Read / Write data in "I-8x37-80, I-7188EG, µPAC-7186EG, µPAC-5xx7, VP-2117, iP-8xx7, WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6, or VP-25W7/ VP-23W7" + I-7018Z by Modbus TCP/IP protocol to display temperature information. Please refer Chapter 11.3.9	I-7018z
Demo_6	PC run "VB 6.0" program to link to (I-8437-80 / 8837-80) + I-8024 & I-8017H by Modbus TCP/IP. Please refer Chapter 11.3.7	Slot2: I-8024 Slot3: I-8017H

µPAC-7186EG, I-7188EG/XG example program:

µPAC-7186EG, I-7188EG:
CD-ROM: \napdos\isagraf\7188eg\demo or
ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/7188eg/demo/

I-7188EG:
CD-ROM: \napdos\isagraf\7188xg\demo or
ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/7188xg/demo/

Project name	Description	I/O board used
Demo_01	Receive data and send to Com2 and Com3	X503/4/5/6
Demo_02	Send one string to COM5 and COM6 in X503 board	X503
Demo_03	Receive message and then send to Com6 or Com7 (using "Comary_r" and "Comary_w")	X503
Demo_04	Linking remote I-7000 and using X107 board	Bus7000b X107
Demo_05	Timer control, TP, TON, TOF	X304
Demo_06	Display a value to S-MMI by "VAL10LED"	X304
Demo_07	Using X107 and remote I-7060D Relay I/O	Bus7000b X107
Demo_08	Receive message and then send to COM3 in X507/8/9 and control Diital Output .	X507_8_9
Demo_09	Using S-MMI and Timer control command "tStart" , "tStop" and Reset to 0	
Demo_10	Using S-MMI	X107
Demo_11	Linking other Modbus RTU device	mbus
Demo_12	Training box demo 1	Bus7000b
Demo_13	Trainning box demo 2	Bus7000b
Demo_18	PID control by "PID_AL". This program can not simulate in PC, please download to controller.	
Demo_21	Write one string to Com3 and Com4	Xbi8 (set as virtual) X50x
Demo_22	Receive message and send to Com3 and Com4	X50x
Demo_23	Receive command from PC and return a Integer value. Comary_R , Comary_W	X50x
Demo_35a	Time synchronization by using Fbus between two or more controllers. "Demo_35A" should be used with "Demo_35B" demo. If the time is modified in 35A, the time in controller running 35B will be automatically modified. (**User can modify the program to use Ebus**)	Fbus_m

Project name	Description	I/O board used
Demo_35b	Time synchronization by using Fbus between two or more controllers.	Fbus_s
Demo_36	Get driver version of the I-8xx7, 7188EG/XG	
Demo_41	Record alarm (text) in X607/X608, then PC can download this record by "ICPDAS UDloader"	X607_608 Xbi8 (virtual D/I) Xbo8 (virtual D/O)
Demo_43	SMS demo, Please modify to your own phone number in the ISaGRAF dictionary window	SMS
Demo_43a	Same as Demo_43 but sending SMS to two cell. phone	SMS
Demo_44	PC download data to X607/X608	X607_608 Xbo8 (virtual D/O)
Demo_48a	Redundancy: I-7188XG redundant Master	Bus7000b Ebus_m
Demo_48b	Redundancy: I-7188XG redundant slave	Bus7000b Ebus_s
Demo_50	PWM I/O demo. (Pulse Width Modulation)	X107
Demo_51a	Redundancy: I-7188EG redundant Master	Bus7000b Ebus_m
Demo_51b	Redundancy: I-7188EG redundant slave	Bus7000b Ebus_s
Demo_61	D/I counters using DI_CNT, I-7188 + X107, Do something when D/I signal happens	X107
Demo_70	Send message to COM2 or COM3 when Alarm 1 to 8 happens	
Demo_72	Controller link one RS-485 remote I-7018z, and also PC can run "VB 6.0" program to become an HMI screen. (please refer to Chapter 11.3.9)	I-7018z

NOTE:
Demo_18 uses PID_AL which is provided by CJ International for evaluation. Please refer to "ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/8000/english_manu PID_AL.ComplexPIDalgorithm implementation.htm".

iP-8xx7, I-8417/8817/8437/8837 example program:

iP-8xx7:
I-8000 CD-ROM: \napdos\isagraf\ip8000\demo or
ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/ip8000/demo/

I-8417/8817/8437/8837:
I-8000 CD-ROM: \napdos\isagraf\8000\demo or
ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/8000/demo/

Project name	Description (iP-8xx7 / I-8xx7)	I/O board used
Demo_01	Timer control, TP, TON, TOF	Push4Key, Show3Led
Demo_01a	To do something at some second later when an event happens	Push4Key, Show3Led
Demo_02	Start, Stop and Reset a Time Timer, tStart , tStop	Push4Key, Show3Led
Demo_03	Read / Write Date & Time SYSDAT_R, SYSDAT_W, SYSTIM_R, SYSTIME_W To output at a time interval. Like, Moday, 09:00 ~ 18:00, Sunday, 10:00 ~ …	NONE
Demo_04	Calculate empy cycle time	NONE
Demo_05	Blinking output, TP, Blink	Push4Key, Show3Led
Demo_06	Change output mode	Push4Key, Show3Led
Demo_07	Display a value to S-MMI, VAL10LED, tStart , tStop	Push4Key, Show3Led
Demo_08	Input a value fromS-MMI , INP10LED	Push4Key, Show3Led
Demo_09	+ , - , * , /	NONE
Demo_10	Display analog input value to S-MMI	I-87017, I-87024, Push4Key
Demo_11a	Fbus Master, NET_ID = 1	Fbus_m, Push4Key, Show3Led
Demo_11b	Fbus Slave, NET_ID = 2	Fbus_s, Push4Key
Demo_12	Using COM3 to receive data from PC	Show3Led
Demo_13	Send data to Com3 every 3 seconds	I-87017
Demo_14	Convert I-7K & I-87K protocol to Modbus protocol	Bus7000b
Demo_15a	Link other Modbus device	Mbus
Demo_15b	Simulate I-8417 as a modbus device for Demo_15a to link to this project	None

Demo_16	Send Modbus command to device once every second	Push4Key, Mbus
Demo_17	Read / Write EEPROM	None
Demo_18	PID control by “PID_AL”. This program can not simulate in PC, please download to controller.	None
Demo_21	Send string to Com5 and Com6	Push4Key, Show3Led
Demo_22	Receive data from Com5 or Com6 (RS-232) and echo back	Show3Led
Demo_23	Receive user defined protocol from PC	Show3Led
Demo_27	Motion control: x axis, slot 0: I-8091, Slot1: I-8090, Napdos\ISaGRAF\8000\Driver\motion.pdf	I-8091 I-8090 Show3Led
Demo_27a	To move some pulse at x-axis of I-8091 of slot 1	I-8091
Demo_28	Motion control: x , y axes, slot0: I-8091, slot1: I-8090, Napdos\ISaGRAF\8000\Driver\motion.pdf	I-8091 I-8090 Show3Led
Demo_29	Store 1200 short integer values every 75 seconds and then send to PC via COM3	I-87017
Demo_30	Store 2880 short integer values every 18 seconds and then send to PC via COM3	I-8017h
Demo_33	Read / Write user defined protocol via COM3	Show3Led
Demo_35a	Time synchronization by using Fbus between two or more controllers. “Demo_35A” should be used with “Demo_35B” demo If the time is modified in 35A, the time in controller running 35B will be automatically modified (**User can modify the program to use Ebus**)	Fbus_m
Demo_35b	Time synchronization by using Fbus between two or more controllers.	Fbus_s
Demo_37	Spotlight demo (Simple HMI) . please refer to Chapter 14	Push4Key Show3Led
Demo_38	I-8xx7 link MMICON , demo 1, please refer to Chapter 16	
Demo_39	I-8xx7 link MMICON , demo 2, please refer to Chapter 16	
Demo_40	Store 8 A/I (binary) to S256 per minute, then PC can load it by "ICPDAS UDloader"	
Demo_41	Record Alarm (text) to S256/512 & PC can load it by "ICPDAS UDloader"	
Demo_42	Store 8 A/I (text) to S256 per min, then PC can load it by "ICPDAS UDloader"	
Demo_43	SMS demo, Please declare your own phone No. in the dictionay, message type	SMS
Demo_43a	Same as demo_43, but send to many cell. phones.	SMS
Demo_44	Demo of PC to download data to the S256/512	
Demo_46	Motion control:	I-8091

	Pulse move at a specified speed	I-8090
Demo_49a	Redundant : 8437/8837 redundant Master	Bus7000 Ebus_m
Demo_49b	Redundant : 8437/8837 redundant slave	Bus7000 Ebus_s
Demo_50	PWM I/O demo. (Refer to section 3.7)	I-8055
Demo_52	Parallel D/I counter demo 1 at slot 0 (Refer to section 3.8). (Counter Value is retained in this demo)	I-8051 Push4Key
Demo_53	Parallel D/I counter demo 2 at slot 0 (Refer to section 3.8) (Not retained)	I-8051 I-8056 Push4key
Demo_54a	Modbus Master	
Demo_54b	Modbus Slave	
Demo_55	PWM I/O demo 2. (Refer to section 3.7)	I-8055
Demo_61	DI counters using DI_CNT, 8xx7 + 8051 (Refer to section 3.8) Do somethig when DI signal happens	I-8051
Demo_70	Send string to COM3 when alarm 1 to 8 happens (Access to variables as array)	
Demo_71	Recording I-8017H 's Ch.1 to Ch.4 voltage input in S-256 / 512 in I-8437-80 or I-8837-80 . The sampling time is one record every 0.05 second. The record period is 1 to 10 minutes. Then PC can download this record and display it as a trend curve diagram by M.S. Excel	I-8024 I-8017H
Demo_72	Demo_72: Connecting I-7018z and I-7188EGD to get 6 channels of 4 to 20 mA input and 4 channles of Thermo-couple temperature input. And then also display the value on PC by VB 6.0 program.	I-7018Z

NOTE:
Demo_18 uses PID_AL which is provided by CJ International for evaluation. Please refer to "CD\Napdos\isagraf\8000\english_manu\ PID_AL.ComplexPIDalgorithm implementation.htm".

WP-8xx7 example program:

The Soft-GRAF Studio software listed in the FAQ-146 is more useful than the way listed in the FAQ-131.

WP-8xx7 CD-ROM: \napdos\isagraf\wp-8xx7\demo\ or
ftp://ftp.icpdas.com/pub/cd/winpac-8xx7/napdos/isagraf/wp-8xx7/demo/

Project name	**Description (WP-8xx7)**	**I/O used**
demo01~ demo07	Soft-GRAF HMI demo01 ~ demo07 : Please refer to the FAQ-146.	
example	A simple Web HMI example	slot 0: I-87055W
wp_vb01	VB.net 2008 demo 01 for WP-8xx7 : DIO demo (Please refer to Chapter 6 of the "WP-8xx7 Getting Started").	slot 0: I-87055W
wp_vb02	VB.net 2008 demo 02 for WP-8xx7. Analog I/O (Please refer to Chapter 6 of the "WP-8xx7 Getting Started").	slot 1: I-87024W Slot 2: I-87017HW
wp_vb03	VB.net 2008 demo 03 for WP-8xx7. Read / Write long integer, float & Timer Please refer to Chapter 6.	
wpdmo_01	WinPAC demo_01: R/W float value from file. (FAQ-060)	
wpdmo_02	WinPAC demo_02: R/W long integer from file (FAQ-060)	
wpdmo_03	To output at a time interval: SYSDAT_R, SYSDAT_W, SYSTIM_R, SYSTIM_W (ST+QLD)	
wpdmo_04	WinPAC demo_04: User defined Modbus protocol (No using "Mbus")	
wpdmo_05	To do something at some sec later when an event happens. (FAQ-017)	slot 0: I-87055W
wpdmo_06	Using Message Array - MsgAry_r , MsgAry_w	
wpdmo_07	Convert float value to string, using real_str & rea_str2	
wpdmo_08	PID control, refer to WinPAC-8xx7 CD: \napdos\isgraf\wp-8xx7\english_manu\"PID_AL...htm"	
wpdmo_09	Store & backup boolean & long integer value To/From files	
wpdmo_10	Store & backup boolean & long integer value To/From EEPROM	
wpdmo_11	Dir is \Micro_SD ,save 3 values to 3 files per 10 minutes ,change file name per month	
wpdmo_14	Retain variable by Retain_b, Retain_N, Retain_f, Retain_t . (FAQ-074)	
wpdmo_16	Dir is \Micro_SD ,save 3 values to 1 file every minute ,change file name every day	
wpdmo19	Send UDP String to PC when alarm happens (using variable array),Time_Gap is 1 sec	slot0: I-87055W

	(Chapter 19.2 of the “ISaGRAF User's Manual”)	
wpdmo19a	Send UDP String to PC 3 sec later, Time_Gap is 250ms (Chapter 19.2 of the “ISaGRAF User's Manual”)	slot0: I-87055W
wpdmo19b	Send UDP Str to PC 3 sec later (wpdmo19a is better), Time_Gap is 250 ms (Chapter 19.2 of the “ISaGRAF User's Manual”)	slot0: I-87055W
wpdmo_20	receive String coming from remote PC or controller via UDP/IP	
wpdmo_21	using "com_MRTU" to disable/enable Modbus RTU slave port,	
wpdmo_22	PWM I/O demo. (Pulse Width Modulation), minimum scale is 2ms for WinPAC	slot 0: I-8055W
wpdmo_23	Send Time String to COM3:RS-232 every second by using COMOPEN, COMSTR_W . (FAQ-059)	
wpdmo_24	Send string to COM3 when alarm 1 to 8 happens	slot 0: I-87055W
wpdmo_26	To move some pulse at x-axis of I-8091W of slot 1 in WP-8xx7 (Chapter 18 of the “ISaGRAF User's Manual”)	slot 1: I-8091W
wpdmo_27	Motion x (Chapter 18 of the “ISaGRAF User's Manual”)	slot 1: I-8091W slot 2: I-8090W
wpdmo_28	Motion x-y (Chapter 18 of the “ISaGRAF User's Manual”)	slot 1: I-8091W slot 2: I-8090W
wpdmo_29	Moving to he Abs. position when CMD is given (Chapter 18 of the “ISaGRAF User's Manual”)	slot 1: I-8091W slot 2: I-8090W
wpdmo_30	WP8xx7(10.0.0.102) link two i8KE8 + I/O , one is 10.0.0.108, one is 10.0.0.109 . (FAQ-042)	
wpdmo_31	WP8xx7(10.0.0.2) link one i8Ke8 + I/O (10.0.0.109) (FAQ-042)	
wpdmo_32	Set up WP8xx7 as TCP/IP Client & link to other TCP/IP server (1 connection) (Chapter 19.3 of the “ISaGRAF User's Manual”)	slot 0: I-87055W
wpdmo_33	Same as Wpdmo_32 but send message only when event last for larger than 3 seconds	slot 0: I-87055W
wpdmo_36	Read Real Val from Modbus RTU device (www.icpdass.com > FAQ > Software > ISaGRAF > 47 & 75)	
wpdmo_37	Write Real Val to Modbus RTU device. (FAQ-047 & 75)	
wpdmo_38	Using Modbus function code 6 to write 16 bits. (FAQ-046 & 75)	
wpdmo_39	WP-8xx7 + I-8172W connecting FRnet I/O modules. (FAQ-082)	
wpdmo_41	COM3 connecting 1:M7053D + 2:M7045D (MBRTU format, baud=9600)	

	(Chapter 21 of the "ISaGRAF User's Manual")	
wpdmo_42	COM3 connecting 1:M-7053D to get D/I counter value (MBRTU format, baud=9600)	
wpdmo_43	COM3 connecting 1:M7017R + 2:M7024 (MBRTU format, baud=9600)	
wpdmo_44	COM3 connecting 1:M7017RC , Current input, +/- 20mA, 4-20mA (Modbus format)	
wpdmo_45	COM3 connecting 1:M-7019R (set as T/C K-type input) (MBRTU format, baud=9600)	
wpdmo_46	COM3 connecting 1:M7080 (MBRTU format, baud=9600)	
wpdmo_48	VB.net 2005 demo - "MBTCP_demo" (FAQ-051)	
wpdmo_50	Non-linear conversion. like give P to find V (P , V relation listed in a file)	
wpdmo_51	Read 10 REAL value from a file,10 rows,each row has 1 REAL value, use str_real	
wpdmo_52	Msg_F. i8xx7 since 3.19. I-7188EG/XG since 2.17/2.15. W8xx7 since 3.36, WP-8xx7	
wpdmo_53	Msg_N. i8xx7 since 3.19. I-7188EG/XG since 2.17/2.15. W8xx7 since 3.36, WP-8xx7	
wpdmo_54	Read 20 REAL values from a file,4 rows,each row has 5 REAL values,uses msg_f . (FAQ-060)	
wpdmo_55	Read 20 Integers from a file,2 rows, each row has 10 Integers,uses msg_n	
wpdmo56	Retain 17 REAL value in a file, 2 rows, Each row has 10 REAL value	
wpdmo56a	Retain 2 Boo + 17 REAL in a file, 2 rows, Each row has 10 REAL value	
wpdmo56b	Retain 25 Integer in a file, 2 rows, Each row has 10 integer value	
wpdmo56c	Retain 2 Boo + 25 Integer in a file, 2 rows, Each row has 10 integer value. (FAQ-060)	
wpdmo56d	Retain 17 Real + 2 Boo + 10 Integer in 2 file, Each row has 10 value	
wpdmo56e	Retain more than 255 Real, 255 Boo,255 Integer in 2 file, up to 1024.	
wpdmo_61	i8xx7, WP8xx7: AutoReport data to PC via UDP.Controller=10.0.0.103,PC=10.0.0.91	
wpdmo_62	Send email via Ethernet port. (To one receiver without attached file) (FAQ-067, 71, 72, 76, 77)	
wpdmo_63	For WP-8xx7 & W-8xx7 only. Send email to one receiver with one attached file. (FAQ-067, 71, 72, 76, 77)	
wpdmo64a	station 1001 , Time synchronization of many controllers via Ethernet.	
wpdmo64b	station 1002 , Time synchronization of many controllers via Ethernet.	
wpdmo65a	WP8xx7: Record temperature per minute to a file. Then send it by email per day. (FAQ-067, 71, 72, 76, 77)	slot 2: I-87018z
wpdmo65b	WP8xx7: Same as wdmo_65a but add time synchronization and state report to PC. (FAQ-067, 71, 72, 76, 77)	slot 2: I-87018z

wpdmo_66	Record 1 to 4-Ch. i8017HW voltage per 20ms, then send this record file by Email	slot 2: I-8024W slot 3: I-8017HW
Wpdmo_70	FRnet : WP-8xx7 or iP-8447, slot1: I-8172W, Port0, FR-2057(adr=4), FR-2053(adr=8)	slot 1: I-8172W FR-2057 FR-2053
Wpdmo_76	SMS : WP-8447, COM4: GTM-201-RS232	GTM-201-RS232
wpdmo71a	WP-8xx7 COM4 connects I-7530 -- "CANopen" ID=1 device (8DI, 8DO, 4AO, 8AI) . (FAQ-086)	
wpdmo71b	Similiar as wdmo_71A but connecting two I-7530. One is at COM5, one is at COM6	
wpdmo71c	WP8xx7 COM4 – 7530 -- CAN device to get string (with float or integer data inside)	
wpdmo71d	Similiar as wdmo_71c but connecting two I-7530. One is at COM5, one is at COM6	
wpdmo71e	WP-8xx7: COM5 --- I-7530 --- CANopen device. COM6 --- I-7530 --- CAN device	
wpdmo72a	New WP-8xx7 redundant system with RU-87P4 + I-87K I/O (Without Touch HMI). (FAQ-093)	
wpdmo72b	Same as wpdmo72a but setup COM1 as Modbus RTU slave port to connect one RS-232 Touch HMI. (FAQ-093)	
wpdmo72c	New WP-8xx7 redundant system with I-8KE8-MTCP I/O (Without Touch HMI)	
wpdmo74a	get average value of one REAL value. (FAQ-099)	
wpdmo74b	get average value of one Integer value. (FAQ-099)	
wpdmo75	Using the I-8088W(8-ch, PWM output) in slot0	slot 0: I-8088W
wpdmo75b	Connect the I-87088W (I-7088) (addr=1,baud=115200) via WP-8xx7's COM2:RS485	I-87088W (I-7088)
wpdmo77a	sending / Receiving UDP bytes by using eth_udp and eth_send( ) and eth_recv( )	
wpdmo77b	sending / Receiving TCP bytes by using eth_tcp and eth_send( ) and eth_recv( )	
wpdmo78	WP-8xx7 COM2 Mbus Master---M-7011 (ID=1, baud=9600) to get AI,DI (FAQ-118)	M-7011
wpdmo79a	AP1 of FAQ119: Mbus RTU Master (Central station)	
wpdmo79b	AP1 of FAQ119: Mbus RTU Slave (local 1),Must set PAC ID (Slave Number) to 1	
wpdmo79c	AP1 of FAQ119: Mbus RTU Slave (local 2),Must set PAC ID (Slave Number) to 2	

wpdmo80a	AP2 of FAQ119: Mbus TCP Master (Central station)	
wpdmo80b	AP2 of FAQ119 (local 1), Must set ID to 1, LAN1=192.168.1.178, LAN2=192.168.1.179	
wpdmo80c	AP2 of FAQ119 (local 2), Must set ID to 1, LAN1=192.168.1.180, LAN2=192.168.1.181	
wpdmo81	WP-8xx7+slot 1: I-8017HW (single-End) to get Moving Average (refer to FAQ-120)	slot 1: I-8017HW

VB.NET 2008 example program: Running with ISaGRAF program in the same WP-8xx7

WP-8xx7 CD-ROM: \napdos\isagraf\wp-8xx7\vb.net_2008_demo\ or
ftp://ftp.icpdas.com/pub/cd/winpac-8xx7/napdos/isagraf/vb.net_2008_demo/

Project name	**Description** **WP-8xx7 & VB.NET 2008**	**I/O board used**
wp_vb01	Digital I/O demo (The related demo project name: “wp_vb01.pia”).	slot 0: I-87055W
wp_vb02	Analog I/O demo (The related demo project name: “wp_vb01.pia”).	slot 1: I-87024W Slot 2: I-87017HW
wp_vb03	Read/Write ISaGRAF internal integers, timers & real variables. (The related demo project name: “wp_vb03.pia”).	

WP-8xx7 Web HMI example program :

Web HMI example program:
WinPAC-8xx7 CD-ROM: \napdos\isagraf\wp-8xx7\wp_webhmi_demo\ or
ftp://ftp.icpdas.com/pub/cd/winpac-8xx7/napdos/isagraf/wp-8xx7/wp_webhmi_demo/

Related ISaGRAF program:
WP-8xx7 CD-ROM: \napdos\isagraf\wp-8xx7\demo\ or
ftp://ftp.icpdas.com/pub/cd/winpac-8xx7/napdos/isagraf/wp-8xx7/demo/

Name	Description (WP-8xx7 Web HMI)	IO board
sample	A Web HMI sample	No I/O board
example1	A simple example listed in Chapter 4	slot 0: I-87055W
wphmi_01	Display controller's date & time	No I/O board
wphmi_02	DI & DO demo	slot 0: I-87055W
wphmi_03	Read / Write Long, float & Timer value	No I/O board
wphmi_04	Read / Write controller's String	No I/O board
wphmi_05	Multi-Pages demo Page menu is on the Left	slot 0: I-87055W
wphmi_05a	Multi-Pages demo Page menu is on the Top	slot 0: I-87055W
wphmi_06	AIO demo, scaling is in ISaGRAF	slot 2: I-87024W slot 3: I-8017HW
wphmi_07	AIO demo, scaling is in PC	slot 2: I-87024W slot 3: I-8017HW
wphmi_08	download controller's file to PC	slot 0: I-87055W
wphmi_09	pop up an alarm window on PC	slot 0: I-87055W
wphmi_11	Trend curve.	slot 2: I-87024W slot 3: I-8017hW
wphmi_12	Record 1 to 8 Ch. I-8017HW 's volt every 50 ms and draw trend curve by M.S.Excel	slot 3: I-8017hW slot 2: I-8024W
wphmi_13	Record 1 to 4-Ch. I-8017HW's voltage every 10 ms and draw trend curve by M.S.Excel	slot 3: I-8017hW slot 2: I-8024W

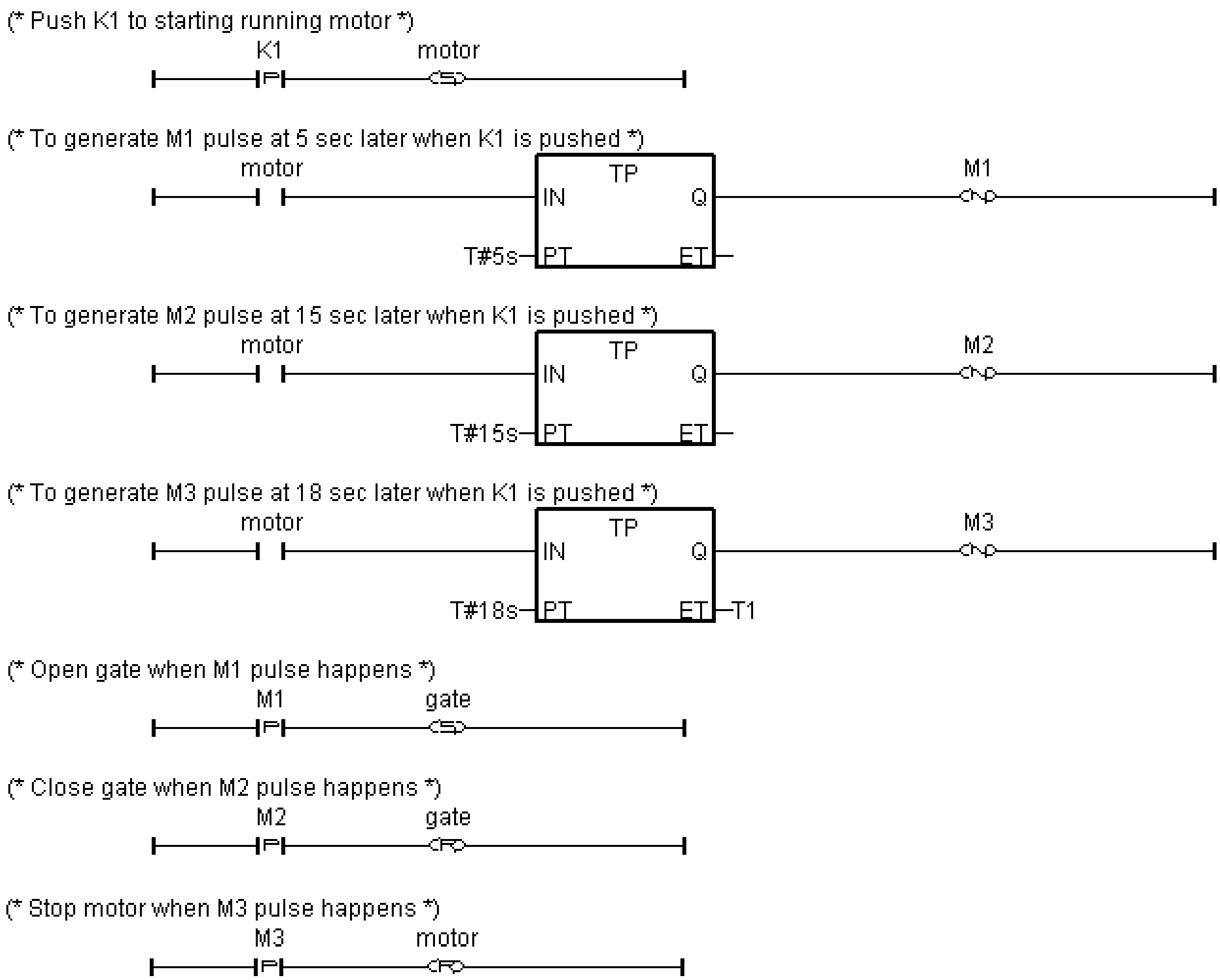
11.3: Description Of Some Demo Examples

11.3.0 Demo_01A & Demo_03: Do something at specific time

Demo_01A: Do something at some seconds later when an event happens.
Location: I-8000 CD-ROM: \napdos\isagraf\8000\demo\“demo_01a.pia”

Variables :

Name	Type	Attribute	Description
K1	Boolean	**Input**	push K1 to start running motor (pushbutton 1 on the I-8xx7)
Motor	Boolean	**Output**	True means to run motor, False means to stop motor
Gate	Boolean	**Output**	True means to open gate, False means to close gate
M1	Boolean	**Internal**	event generated at 5 sec later when K1 is pushed
M2	Boolean	**Internal**	event generated at 15 sec later when K1 is pushed
M3	Boolean	**Internal**	event generated at 18 sec later when K1 is pushed
T1	Timer	**Internal**	Time past



Demo_03: Do something at specific weekday & some time interval

Location: I-8000 CD-ROM: \napdos\isagraf\8000\demo\ "demo_03.pia"

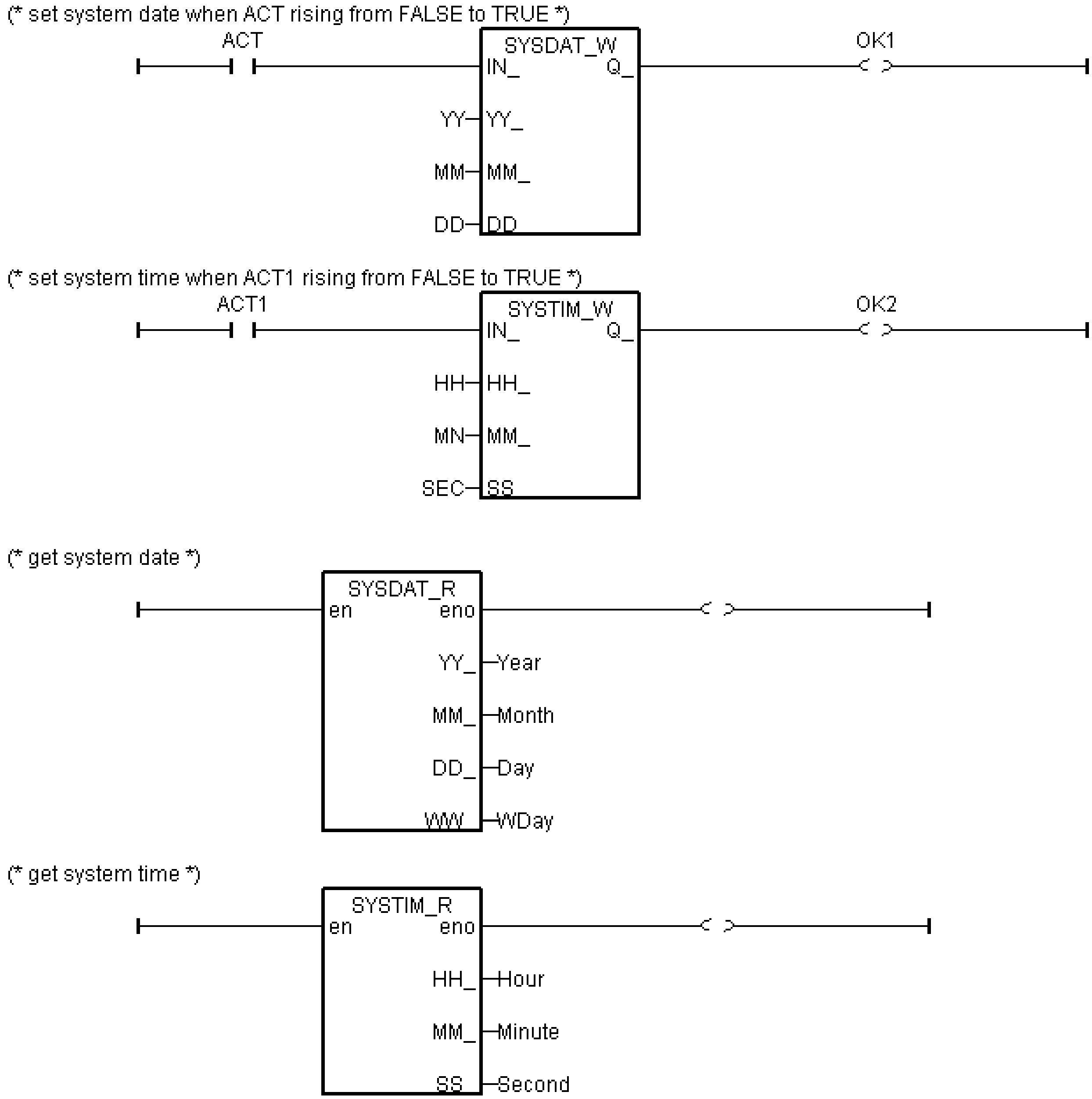
Variables :

Name	Type	Attribute	Description
Year	Integer	**Internal**	System year, 2001 ~
Month	Integer	**Internal**	System Month, 1 ~ 12
Day	Integer	**Internal**	System date, 1 ~ 31
Wday	Integer	**Internal**	System Wday, 1:Monday ~ 6:Saturday, 7:Sunday
Hour	Integer	**Internal**	System hour, 0 ~ 23
Minute	Integer	**Internal**	System minute, 0 ~ 59
Second	Integer	**Internal**	System second, 0 ~ 59
YY	Integer	**Internal**	New system year to set
MM	Integer	**Internal**	New system month to set
DD	Integer	**Internal**	New system date to set
HH	Integer	**Internal**	New system hour to set
Mn	Integer	**Internal**	New system minute to set
Sec	Integer	**Internal**	New system second to set
Act	Boolean	**Internal**	Trigger to set new date
Act1	Boolean	**Internal**	Trigger to set new time
OK1	Boolean	**Internal**	Read back of "SYSDAT_W"
OK2	Boolean	**Internal**	Read back of "SYSTIM_W"
L1 ~ L3	Boolean	**Internal**	Simulate Boolean Output 1 to 3
Time_val	Integer	**Internal**	unit is sec, = 3600 x hour + 60 x minute + sec, every day = 0~86399

Operation action:

1. Monday ~ Saturday, L1 ~ L3, 09:00:00 ~ 18:00:00 ON
2. Sunday, L1, 13:00:00 ~ 20:00:00 ON
3. Other time, L1 ~ L3 are all OFF

Ladder program : get_time



ST program : control

```
time_val  :=  3600 * hour  +  60 * minute  +  second ;    (* calculate time in sec. *)

(* set as False at the beginning of this ST program*)
L1  :=  False ;
L2  :=  False ;
L3  :=  False ;

(* Monday ~ Saturday, L1 ~ L3, 09:00:00 ~ 18:00:00 ON *)
IF  ( Wday  >= 1)  AND  ( Wday  <=  6 )  THEN
   IF  (time_val >= 32400 )  AND  ( time_val  <=  64800 )   THEN
      L1  :=  True ;
      L2  :=  True ;
      L3  :=  True ;
   END_IF ;
END_IF ;

(* Sunday, L1, 13:00:00 ~ 20:00:00 ON *)
IF  ( Wday  =  7 )  THEN
   IF  ( time_val  >=  46800 )  AND  ( time_val <= 72000 )   THEN
      L1  :=  True ;
   END_IF ;
END_IF ;
```

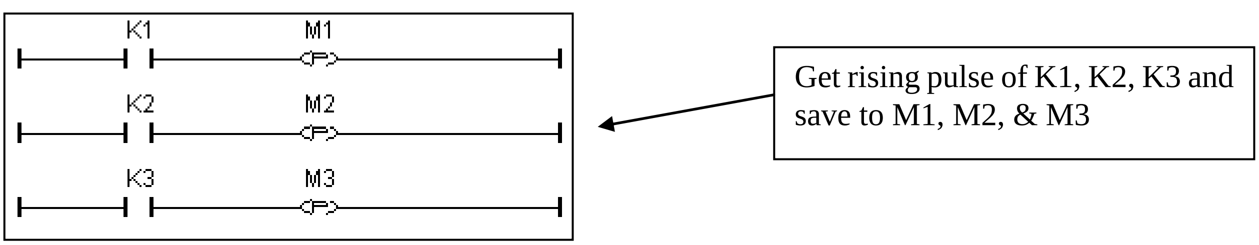
11.3.1 Demo_02 : Start, Stop And Reset Timer

Project architecture:

Variables :

Name	Type	Attribute	Description
M1	Boolean	**Internal**	Indicate a rising pulse of K1
M2	Boolean	**Internal**	Indicate a rising pulse of K2
M3	Boolean	**Internal**	Indicate a rising pulse of K3
K1	Boolean	**Input**	Pushbutton 1
K2	Boolean	**Input**	Pushbutton 2
K3	Boolean	**Input**	Pushbutton 3
L1	Boolean	**Output**	Output 1
L2	Boolean	**Output**	Output 2
L3	Boolean	**Output**	Output 3
T1	Timer	**Internal**	Operation timer, **initial value is set at "T#0s"**

LD program "prg1" :



ST program “demo” :

```
(* Start timer *)
IF  M1  THEN
  TSTART(T1) ;
END_IF ;
```

“TSTART” will start ticking the “T1” timer

```
(* Stop timer *)
IF  M2  THEN
  TSTOP(T1) ;
END_IF ;
```

“TSOP” will stop ticking “T1” timer

```
(* Reset timer *)
IF  M3  THEN
  T1 := T#0s ;
END_IF ;
```

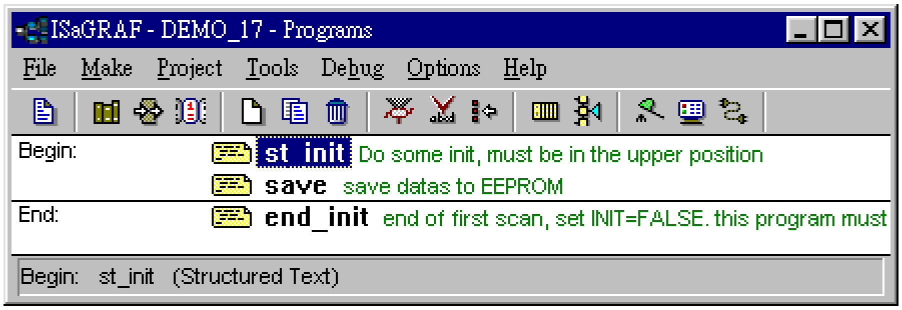
Reset “T1” timer to 0 sec.

```
(* Output L1 ~ L3 *)
L1 := ( T1 > T#2s ) AND ( T1 < T#15s ) ;
L2 := L1 ;
L3 := L1 ;
```

“L1” will be TRUE between 2 and 15 sec of the value of “T1”

11.3.2 Demo_17 : R/W Integer Value From/To The EEPROM

Project architecture:



Variables: (Please refer to Chapter 2.6 for more information about Variable Array)

Name	Type	Attribute	Description
V[0..7]	Integer	**Internal**	Variable Array, Dim is 8 If modifying the value of V[0..7], the new value will be stored to the EEPROM
Old_V[0..7]	Integer	**Internal**	Variable Array, Dim is 8 Old value of V[0..7]
TEMP	Boolean	**Internal**	Internal use
ii	Integer	**Internal**	Index of "for" loops
INIT	Boolean	**Internal**	If controller is just powered up, **initial value is TRUE**

ST program "st_init" :

```
if  INIT = TRUE  then   (* First scan cycle *)

  (* Read 8 integers from EEPROM *)
  (* save them to Old_V[0..7] , V[0..7] *)
  for  ii  :=  0  to  7  do
     V[ii]  := eep_n_r( ii+1 )  ;
     Old_V[ii]  :=  V[ii]  ;
  End_for  ;

  (* remove protection of EEPROM *)
  TEMP  :=  eep_en( ) ;

end_if  ;
```

Read long integers stored at the position from 1 to 8 of the eeprom at the first scan cycle.

Remove the protection of EEPROM, so that it can be written later.

ST program “save” :

```
 (* save V[0..7] to EEPROM if it is modified *)
(* You will find write to EEPROM take lots of time, about 23ms for each eep_n_w *)
for ii := 0 to 7 do

   IF V[ii] <> Old_V[ii] THEN
      TEMP := eep_n_w( ii+1 , V[ii] ) ;
      Old_V[ii] := V[ii] ;
   END_IF ;

End_for ;
```

The value will be saved to eeprom only when the current value is changed.

Then update Old value to the new value.

ST program “end_init” :

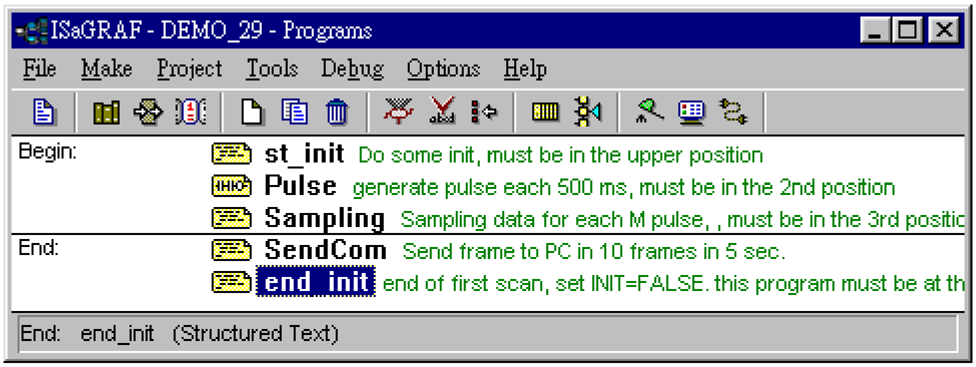
```
INIT := FALSE ;
```

Set “INIT” to False, so that “INIT” is only TRUE at the first scan cycle since it is declared with the initial value - TRUE.

11.3.3 Demo_29: Store 1200 Short Int Every 75 sec & Send To PC Via Com3

This demo program is to save the 8 analog input value (8 samples) of the I-87017 to the short-integer array every 500ms. Then when the number of samples reach 1200, these samples will be divided in 10 frames, each frame contain 120 samples, and sent to one PC via COM3 (RS-232/RS-485).

Project architecture:



Variables :

Name	Type	Attribute	Description
M	Boolean	**Internal**	pulse to store a sample
M1	Boolean	**Internal**	pulse to send frame
M2	Boolean	**Internal**	To generate M1 pulse
INIT	Boolean	**Internal**	If controller is just powered up, **initial value is TRUE**
TMP	Boolean	**Internal**	For temporal use
A1	Integer	**Input**	Connect to Ch. 1 of I-87017
A2	Integer	**Input**	Connect to Ch. 2 of I-87017
A3	Integer	**Input**	Connect to Ch. 3 of I-87017
A4	Integer	**Input**	Connect to Ch. 4 of I-87017
A5	Integer	**Input**	Connect to Ch. 5 of I-87017
A6	Integer	**Input**	Connect to Ch. 6 of I-87017
A7	Integer	**Input**	Connect to Ch. 7 of I-87017
A8	Integer	**Input**	Connect to Ch. 8 of I-87017
count	Integer	**Internal**	No. of sample(1~1200) that is processing, **init value=1**
position	Integer	**Internal**	position in current short integer array, 1 ~ 256
No	Integer	**Internal**	current short integer array No. which is processing
Frame_No	Integer	**Internal**	only = 0 ~ 10
TMP_VAL	Integer	**Internal**	For temporal use

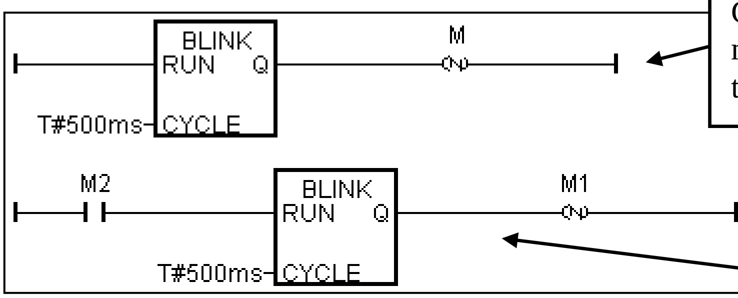
ST program “st_init” :

```
if  INIT = TRUE  then  (* First scan cycle *)

  COUNT := 1 ;   (* init count to 1 *)
  FRAME_NO := 0 ;  (* init to 0 *)
  M2  :=  False ;  (* init M2 to false *)
  TMP := comopen( 3 , 9600 , 8 , 0 , 1 ) ;

end_if ;
```

Do some init at first scan cycle

Open Com3 as baud=9600, char. size=8, no parity & stop bit=1

LD program “Pulse” :



Generate M pulse every 500 ms, M pulse is used to record the A/I sample value

Generate M1 pulse every 500 ms when “M2” is set to TRUE, M1 pulse is used to send one frame to PC

ST program “Sampling” :

```
(* no M pulse, return *)
if  M = False  then
  return ;
end_if ;

No := ( ( COUNT – 1 ) / 240 ) + 1 ;
Position := COUNT – 240 * ( No – 1 ) ;

(* Store I-87017 : 8 A/I value to short integer array *)
(* Please keep in mind, the max No of short int arry can be used is 12 *)
TMP := ary_w_w(No , position , A01) ;
TMP := ary_w_w(No , position+1 , A02) ;
TMP := ary_w_w(No , position+2 , A03) ;
TMP := ary_w_w(No , position+3 , A04) ;
TMP := ary_w_w(No , position+4 , A05) ;
TMP := ary_w_w(No , position+5 , A06) ;
TMP := ary_w_w(No , position+6 , A07) ;
TMP := ary_w_w(No , position+7 , A08) ;

If  (COUNT+7 >= 1200)  then
  FRAME_NO := 1 ;     (* set FRAME_NO=1 *)
  M1 := True ;
  M2 := True ;
  COUNT := 1 ;  (* reset COUNT to 1 *)
else
  COUNT := COUNT+8 ; (* next sampling *)
end_if ;
```

This will make the following statement only be processed when M pulse is generated

Get current array No. and the start pos. of the array which the samples will save to. Each array can store up to 240 samples

Store I-87017 : 8 A/I value to short integer array

If number of stored samples reach 1200, send to PC via Com3 in 10 frames

set M1=True to send first frame at next ST program - "SendCom"

set M2=True to generate M1 pulse at previouse LD program - "Pulse"

If the number of stored samples hasn’t reached 1200 yet, pulse “COUNT” by 8 for next sampling.

ST program “SendCom” :

```
If  M1 = False  then
  Return ;
end_if ;
```

This will make the following statement only be processed when M1 pulse is generated

User defined frame format : Each contains 120 short integers

	STX	DLE	FRAME_NO	DATA	ETX
number of bytes	1	1	1	120x2	1
value	0x2	0x10	1~10	?	0x03

When “FRAME_NO” is between 1 and 10

```
If ( (FRAME_NO >= 1)  and  ( FRAME_NO <= 10) )  then

  No := ( FRAME_NO – 1 ) / 2 + 1 ;
```

Get the short integer array No to process. Keep in mind, each array strore up to 240 samples. (in other word -- 2 frames)

```
  case  (FRAME_NO – 2 * ( No – 1 ) )  of
    1:  position := 1 ;
    2:  position := 121 ;
  end_case ;
```

Get starting position inside the array

Send one frame via Com3

```
  TMP := comwrite( 3 , 16#2) ;     (* write one byte = STX to Com3 *)
  TMP := comwrite( 3 , 16#10) ;    (* write one byte = DLE to Com3 *)
  TMP := comwrite( 3 , FRAME_NO) ;    (* write frame No = 1 ~ 10 to Com3 *)

  (* write 120 short integers inside the array to Com3 *)
  TMP := comay_ww( 3 , No , 120 , position ) ;

  TMP := comwrite( 3 , 16#3) ;     (* write one byte = ETX to Com3 *)

  M1 := False;
```

500 ms later, send next frame. “M1” will be turned ON after 500 ms later at LD program - "Pulse"

```
  If  (FRAME_NO = 10)  then
    FRAME_NO := 0 ;
    M2 := False ;
  else
    FRAME_NO := FRAME_NO + 1;   (* for next cycle *)
  end_if ;

end_if ;
```

If all frames are sent, reset “FRAME_NO” to 0 , and set “M2” to FALSE to stop to generate “M1”

If some frames have not been sent yet, plus “FRAME_NO” by 1 to send next frame when next “M1” is generated

ST program “end_init” :

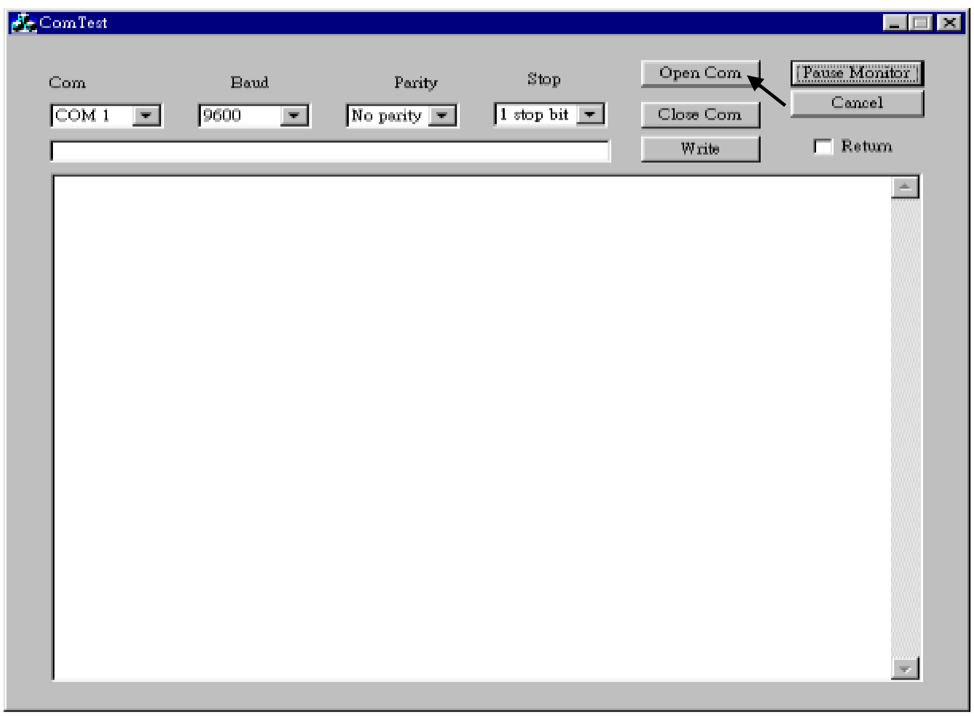
```
INIT := FALSE ;
```

Set “INIT” to False, so that “INIT” is only TRUE at the first scan cycle since it is declared with the initial value - TRUE.

How to test ?

1. Plug one I-87017 in the slot 0 of the I-8xx7 controller.
2. Download Demo_29 to the controller.
3. Prepare a RS-232 cable to connect Com3 of the controller to Com1 of your PC.
4. There is one ultilty named “ComTest.exe” located in the ICP DAS’s CD-ROM. Copy it to your PC. “\Napdos\ISaGRAF\some_utility\Comtest.exe” or you may obtain it from below site.
   ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/some_utility/

5. Execute “ComTest” and select the parameter to “COM1” , “9600” , “No parity” , “1 stop bit” and then click on “Open Com”.



You will receive 10 frames coming from the target controller every 75 seconds.

11.3.4 Demo_33 : R/W User Defined protocol Via Com3:RS-232/RS-485

This demo program can let I-8xx7, iP-8xx7, WP-8xx7, WP-5xx7, VP-25W7 accept commands coming from PC via a RS-232 cable. The command protocol format can be defined by the user. We use the below protocol format in this example.

Command is case insensitive, that means M1 & m1 are same

Protocol Format:

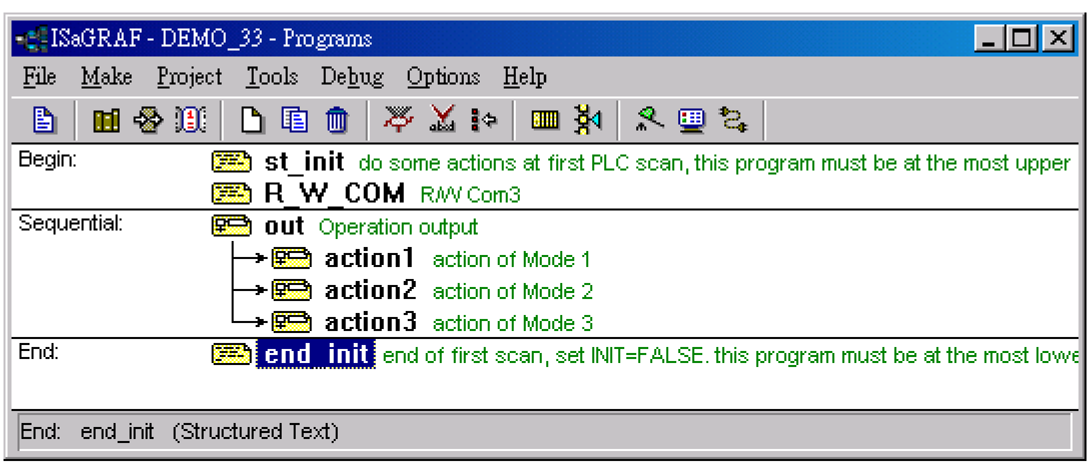
PC req.
M1<CR> : Change to Mode 1
M2<CR> : Change to Mode 2
M3<CR> : Change to Mode 3
Txxxx<CR> : Change Period time to xxxx ms
for ex. T250<CR> will change period time to 250ms
Controller Ans.
OK<CR>

PC req.
M?<CR> : Request the current Mode
Controller Ans.
Mx<CR> : for ex. M1 means Mode 1

PC req.
T?<CR> : Request the current Period time
Controller Ans.
Txxxx<CR> : for ex. T1500 means Period time is 1500ms

Timeout:
a valid command should be completely sent in 5 sec.

Project architecture:



Variables :

Name	Type	Attribute	Description
L1	Boolean	**Output**	Output 1
L2	Boolean	**Output**	Output 2
L3	Boolean	**Output**	Output 3
INIT	Boolean	**Internal**	If controller is just powered up, **initial value is TRUE**
TMP	Boolean	**Internal**	For temporal use
Mode	Integer	**Internal**	Operation Mode, range from 1 to 3
Step	Integer	**Internal**	Processing step
NUM	Integer	**Internal**	Received valid byte number
Num_com3	Integer	**Internal**	return value of Comary_R
byt	Integer	**Internal**	Current operating byte
index	Integer	**Internal**	Index of byte array
CMD	Integer	**Internal**	command type, M, m, T, or t
TMP_val	Integer	**Internal**	for temporal use
ii	Integer	**Internal**	for temporal use
T1	Timer	**Internal**	Period time, valid range is 50 ~ 9999 ms
tout	Timer	**Internal**	timer to measure timeout, tick when first valid byte recved

ST program “st_init” :

```
If   INIT = TRUE   then

  (* Init *)                                Do some init at the first scan cycle
  Mode := 1 ;
  STEP := 0 ;
  T1 := T#500ms ;
  NUM := 0 ;
  tout := T#0s ;


  (* Open Com3 as baud=9600, char. size=8, no parity & stop bit=1 *)
  TMP := comopen( 3 , 9600 , 8 , 0 , 1 ) ;

end_if ;
```

ST program “R_W_COM” :

```
(* STEP:                                          *)
(*      0: no valid data coming                   *)
(*      1: first valid byte received              *)
(*      2: receive '?' command                    *)
(*     10: check if other byte is '0' ~ '9'       *)
(*     21: receive complete command               *)

num_com3 := 0 ; (* reset to 0 *)
if  ComReady(3)  then                       <-- test if data coming from Com3
  num_com3 := Comary_R(3 , 1) ;             <-- read all coming bytes to byte array 1
end_if ;

(* if data coming, process it *)
index := 1 ;
while  num_com3 > 0  do                     <-- Process all coming bytes

  num_com3 := num_com3-1 ;
  byt := array_r(1 , index) ;               <-- get current operating byte from array 1
  index := index + 1 ;

  case  STEP  of                            STEP 0 : check if 1st byte valid or not
   0:
     Case  byt  of   (* check 1st byte *)
       16#4D, 16#6D, 16#54, 16#74 :    (* 1st byte is valid,  M, m, T, or t *)
          STEP := 1 ;     (* for next STEP *)
          NUM := 1 ;      (* plus valid received byte number by 1 *)
          TSTART(tout);   (* start ticking tout *)
          CMD := byt ;   (* record command type *)
          TMP := Array_w(2, NUM, byt) ;   (* save 1st valid byte to byte array 2*)
     end_case ;
   1 :                                      STEP 1 : check if 2nd byte valid or not
     Case  byt  of   (* check 2nd byte *)
       16#3F :    (* 2nd byte is '?' *)
          STEP := 2 ;     (* for next STEP *)
          NUM := 2 ;      (* plus valid received byte number by 1 *)
          TMP := Array_w(2, NUM, byt) ;   (* save 2nd valid byte to byte array 2*)
     else
       if  (CMD = 16#4D  or  CMD = 16#6D )  (* cmd is M, m *)
         and  (byt  >=  16#31)  and  (byt  <= 16#33)   then  (* '1' ~ '3' *)
          STEP := 10 ;    (* for next step *)
          NUM := 2 ;      (* plus valid received byte number by 1 *)
          TMP := Array_w(2, NUM, byt) ;   (* save 2nd valid byte to byte array 2*)
       Elsif  ( CMD=16#54  or  CMD=16#74 )  (* cmd is T, t *)
         And  (byt >= 16#30)  and  (byt <= 16#39)  then  (* '0' ~ '9' *)
          STEP := 10 ;    (* for next step *)
          NUM := 2 ;      (* plus valid received byte number by 1 *)
          TMP := Array_w(2 , NUM, byt) ;   (* save 2nd valid byte to byte array 2*)
```

```
    else
      STEP := 0 ;  (* not valid data, reset STEP to 0 *)
      TSTOP(tout) ;   (* stop ticking "tout" *)
      tout := T#0s ;  (* reset "tout" *)
      NUM := 0 ;      (* reset NUM *)
    end_if ;
  end_case ;
```

STEP 2 : after receive 2^{nd} byte = '?' , check if 3^{rd} byte is <CR>

```
 2 :
  if  byt = 16#0D   then  (* check 3rd byte is <CR> or not *)
    STEP := 21 ;  (* complete command is received *)
    (* send answer to Com3 *)
    case  CMD  of
      16#4D , 16#6D :    (* M or m *)
        TMP := ComWrite(3, 16#4D) ;          (* M *)
        TMP := ComWrite(3, Mode+16#30 ) ;    (* Mode *)
        TMP := ComWrite(3, 16#0D ) ;         (* <CR> *)
      16#54, 16#74 :    (* T or t *)
        TMP := ComWrite(3, 16#54) ;          (* T *)
        TMP := ComStr_w(3, MSG(ANA(T1))) ;   (* Timer value *)
        TMP := ComWrite(3, 16#0D ) ;         (* <CR> *)
    end_case ;
  else
    STEP := 0 ;  (* not valid data, reset STEP to 0 *)
    TSTOP(tout) ;   (* stop ticking "tout" *)
    tout := T#0s ;  (* reset "tout" *)
    NUM := 0 ;      (* reset NUM *)
  end_if ;
```

Receive valid "M?" command, reply "Mx", x = '1' ~ '3'

Receive valid "T?" command, reply "Txxxx", x = '0' ~ '9'

STEP 10 : check 3^{rd} and other byte when command is "Mx" or "Txxxx"

```
10 :
  If  (byt = 16#0D)  then      (* received <CR> *)
    STEP := 21 ;  (* complete command is received *)
    case CMD of
      16#4D, 16#6D :    (* M or m *)
        Mode := Array_r(2, 2) - 16#30 ;  (* Change Mode *)
        (* send answer to Com3 *)
        TMP := ComStr_w(3, 'OK') ;
        TMP := ComWrite(3, 16#0D ) ;         (* <CR> *)
      16#54, 16#74 :    (* T or t *)
        (* get Period *)
        TMP_val := 0 ;
        for  ii := 1  to  NUM-1  do
          TMP_val :=  10*TMP_val + (Array_r(2 , 1+ii) - 16#30 ) ;
        end_for ;
        if  (TMP_val >= 50)  and  (TMP_val < 10000)  then (* must be in 50 ~ 9999 ms *)
          T1 := TMR(TMP_val) ;  (* Change T1 *)
          (* send answer to Com3 *)
          TMP := ComStr_w(3, 'OK') ;
          TMP := ComWrite(3, 16#0D ) ;         (* <CR> *)
```

Receive valid "Mx" command, chang Mode value and reply "OK" to PC

Receive valid "Txxx" command, change T1 value and reply "OK" to PC

```
      end_if ;
    end_case ;

   elsif (byt >= 16#30) and (byt <= 16#39) then   (* '0' ~ '9' *)

     STEP := 10 ;       (* for next step *)
     NUM := NUM+1 ;      (* plus valid received byte number by 1 *)
     TMP := Array_w(2, NUM, byt) ;  (* save other valid byte to byte array 2*)

     if NUM > 5 then     (* command is too long, drop it *)
       STEP := 0 ;     (* reset STEP *)
       TSTOP(tout) ;  (* stop ticking "tout" *)
       tout := T#0s ; (* reset "tout" *)
       NUM := 0 ;      (* reset NUM *)
       EXIT;          (* exit while loop *)
     end_if ;

   else
     STEP := 0 ; (* not valid data, reset STEP to 0 *)
     TSTOP(tout) ;  (* stop ticking "tout" *)
     tout := T#0s ; (* reset "tout" *)
     NUM := 0 ;      (* reset NUM *)

   end_if ;

 end_case ;

end_while ;

(* Check timeout *)
If tout > T#5s then  (* if timeout *)
  STEP := 0 ;     (* reset STEP *)
  TSTOP(tout) ;  (* stop ticking "tout" *)
  tout := T#0s ; (* reset "tout" *)
  NUM := 0 ;      (* reset NUM *)
end_if ;

(* reset STEP to 0 *)
if STEP = 21 then
  TSTOP(tout) ;  (* stop ticking "tout" *)
  tout := T#0s ; (* reset "tout" *)
  NUM := 0 ;      (* reset NUM *)
  STEP := 0 ;
end_if ;
```

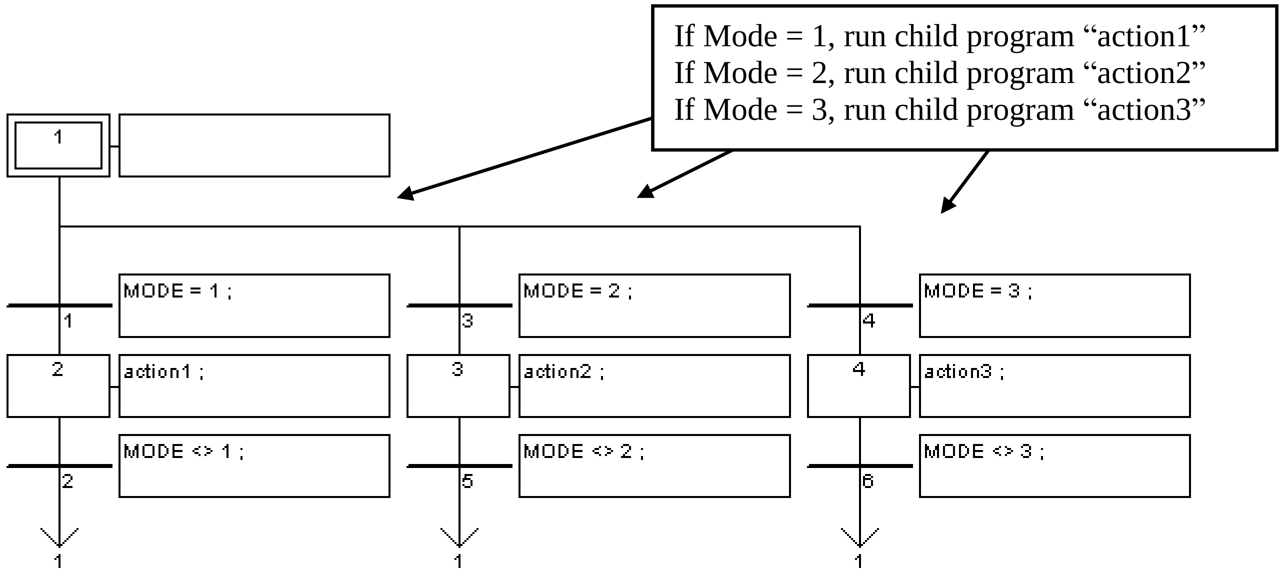
Receive ‘0’ ~ ‘9’, command is not completely received yet, process next byte

Check timeout, a valid complete command should be received in 5 seconds

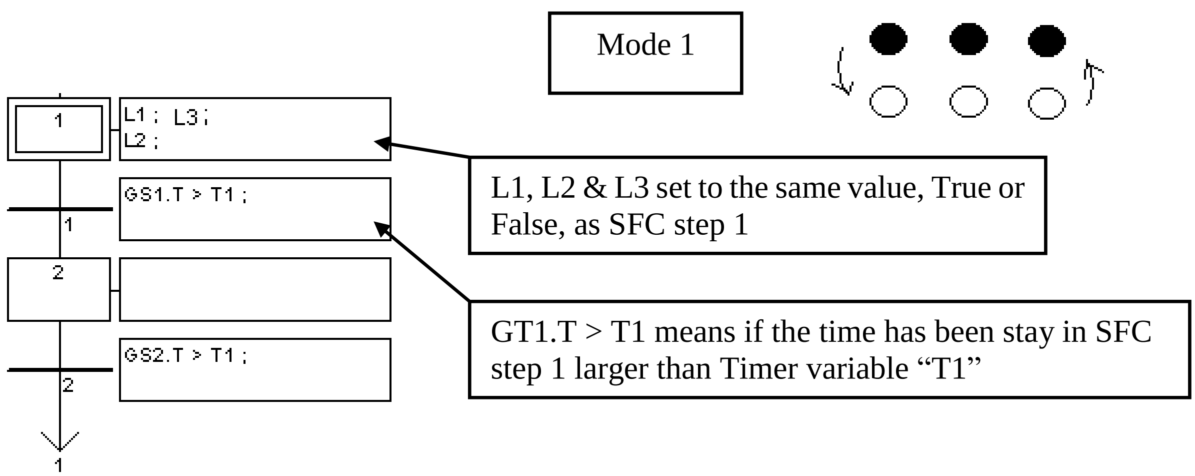
Valid command has been processed, reset to STEP 0

SFC program “Out” :

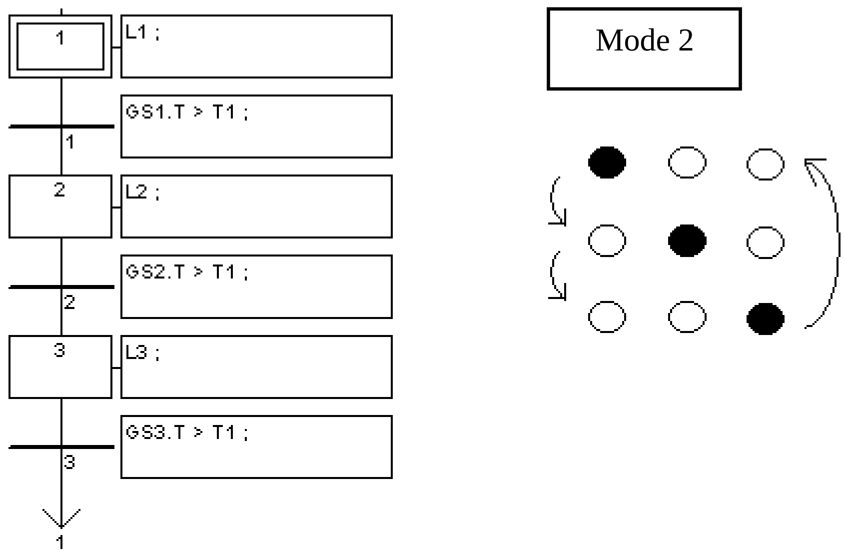
Each statement should end with a colon “;”



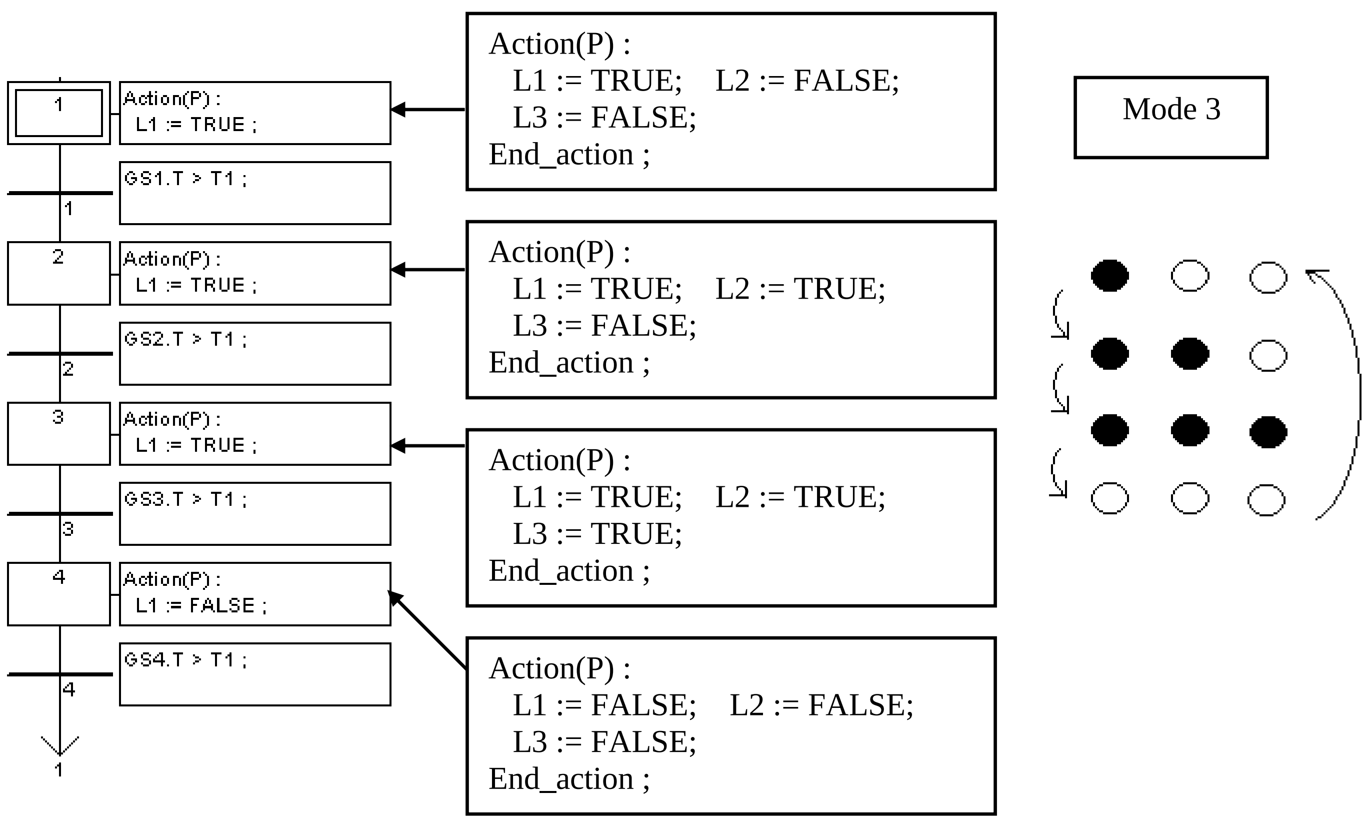
SFC child program “action1” :



SFC child program “action2” :



SFC child program “action3” :



ST program “end_init” :

```
INIT := FALSE ;
```

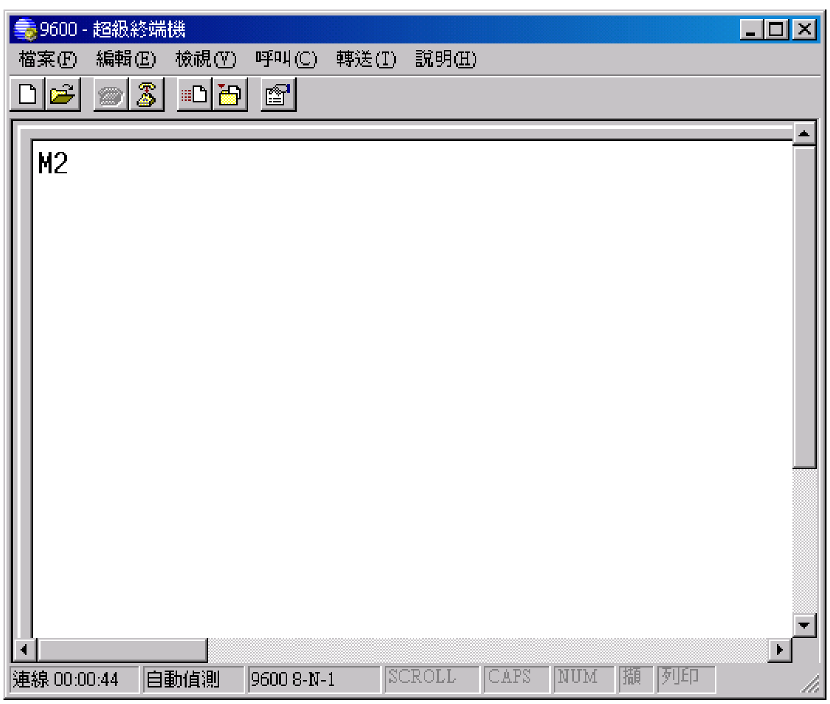
Set “INIT” to False, so that “INIT” is only TRUE at the first scan cycle since it is declared with the initial value - TRUE.

How to test ?

1 . Download Demo_33 to the controller.
2. Prepare a RS-232 cable to connect Com3 of the controller to Com1 of your PC.
3. You may open a “Hyper Terminal” with Com1, 9600, N, 8, 1 and “No flow control” to type the following command to test

M2<CR> : change to mode 2
T?<CR> : request current period time
T200<CR> : change to 200ms
T1500<CR> : change to 1500ms
M?<CR> : request current mode

<CR> is the return char.



11.3.5 Wdemo_24: Send string to COM2 when alarm 1 to 8 happens

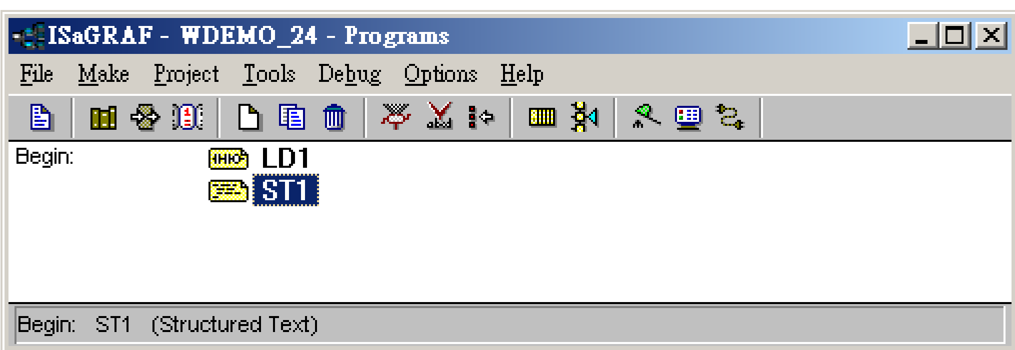
This demo program can be running in WinCon-8xx7/8xx6 or in I-8xx7. Please init “PORT” as 2 if your target is WinCon, while 3 for I-8xx7.

Location: W-8xx7 CD-ROM:\napdos\isagraf\wincon\demo\ “wdemo_24” or
ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/demo/ “wdemo_24”

Variables : (Please refer to Chapter 2.6 for more informayion about Variable Array)

Name	Type	Attribute	Description
INIT	Boolean	**Internl**	Init as TRUE, True indicates first PLC scan cycle
TMP	Boolean	**Internal**	Internal use
Tick1	Boolean	**Internal**	pulse generated every 1 sec to counting time
IN[0..7]	Boolean	**Input**	Variable Array, Dim is 8 The input signal
OLD_IN[0..7]	Boolean	**Internal**	Variable Array, Dim is 8 The old value of IN[0..7]
ii	Integer	**Internal**	Index of “For” loops
Port	Integer	**Internal**	A COM PORT Number to open, init as 2 for WinCon
CNT[0..7]	Integer	**Internal**	Variable Array, Dim is 8 To count the elapsed seconds
Msg1	Message	**Internal**	Message to send to COM2, init length as 128

Project architecture:



Operations:

1. If IN[0..7] rising from False to True and hold in True for at least 3 seconds, send one message = ‘Alarm N’ + <LF> <CR> to COM2. N= 1,2, … 8 depends on which Input is triggered. For ex, if IN[2] is rising and hold in True longer than 3 seconds, send ‘Alarm 3’ + <LF> <CR> to COM2

2. If after IN[0..7] ‘s first alarm is sent and then continusly hold in True for 30 seconds, then send one more messge after every 30 second past to COM2 until the state of IN[0..7] is falling to FALSE. The string is for ex, ‘Alarm 3 , 30 sec past !’

How to test ?

1. Please download wdemo_24 to W-8xx7+ slot 1: I-8077 (or demo_70 for I-8xx7+slot 0: I-8077)

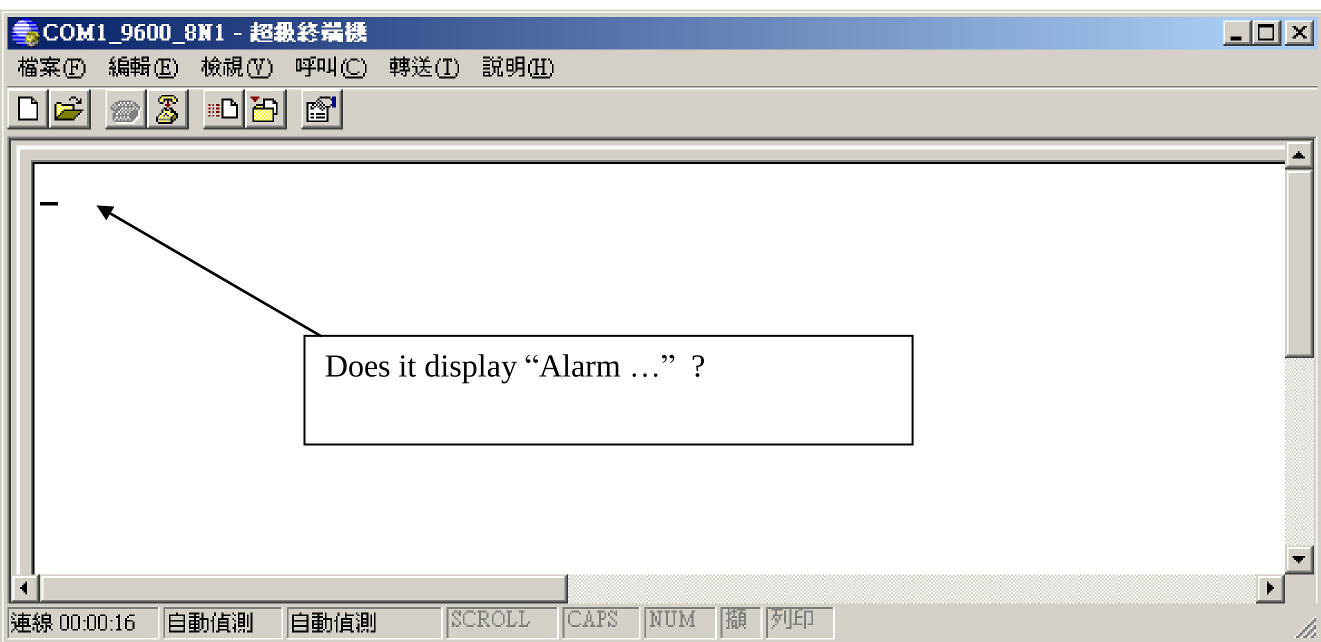
2. Connect a RS-232 cable between W-8xx7's COM2 to your PC's COM1

Wincon (COM2)	PC (COM1)
2 RXD	2 RXD
3 TXD	3 TXD
5 GND	5 GND

Or if you are using I-8xx7's COM3 to your PC's COM1

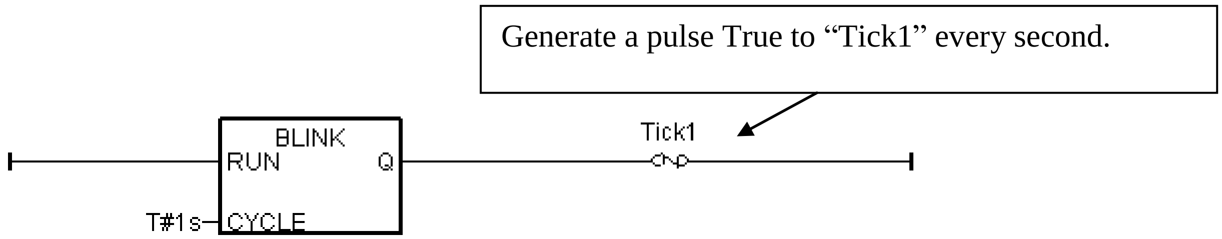
I-8xx7 (COM3)	PC (COM1)
2 TXD	2 RXD
3 RXD	3 TXD
5 GND	5 GND

3. Open PC's Hyper terminal at COM1 with 9600, 8 char. size, no parity, 1 stop bit and No flow control. And then please switch I-8077's Input1 or 2 or … from FALSE to TRUE and wait about three seconds. If it works, there should be a message "Alarm …" displayed. And then please hold this input TRUE more than 30 seconds, there should be one another message "Alarm …, 30 sec past !" displayed.



Program description:

LD1 program:



ST1 program:

```
(* only do it in 1st PLC scan *)
if  INIT  then
  INIT := FALSE ;   (* No more 1st PLC scan cycle *)
  TMP := COMOPEN( PORT , 9600 , 8 , 0 , 1 ) ;   (* open COM2, 9600,8,N,1 *)

  (* init value of CNT[0..7] to -7 *)
  For  ii := 0  to  7  do
    CNT[ii]  :=  -7  ;
  end_for ;

end_if ;

(* test all IN[0..7] if rising from False to True *)
for   ii := 0  to  7   do

  (* test if IN[0..7] signal rising *)
  If   ( IN[ii] = True )  and  ( OLD_IN[ii] = False )  then

    (* set related CNT[ ] value to -3 when Input event is trigered *)
    (* if CNT[ ] value is not -7, it means "INPUT been trigered" *)
(* the CNT[ ] value will plus 1 every 1 sec past later, except the related INPUT become False *)
    CNT[ii] := -3 ;

  end_if ;

  (* if INPUT is cleared or "if related INPUT become FALSE", the related CNT[ ] value
                 will reset to -7: "No input event happens at that INPUT channel" *)
  if IN[ii] = False then  (* signal is becoming FALSE *)

    (* set related CNT[ ] value to -7: "No input event happens at that INPUT channel" *)
    CNT[ii] := -7 ;

  end_if ;
```

```
 if   Tick1   then   (* Tick1 is generated as pulse "True" every second in "LD1" program *)

  (* if CNT[ ] value is not -7, means the related input is trigered *)
  if   CNT[ii] > -7   then

    CNT[ii]  :=  CNT[ii]  + 1 ;   (* plus 1, Tick1 = True means 1 sec has passed *)

    (* ----------------------------------------------------------------------------- *)
    (* INPUT event happens and 3 sec past, send 1st alarm message to COM2 *)
    if  ( CNT[ii] = 0 )  then   (* send 1st alarm when CNT[ ] is from -3, -2, -1 ---> 0 *)
      CNT[ii] := 0 ;  (* re-start from 0 and then count to 30 second to send alarm *)
      (* send one message to COM2 *)
      msg1 := 'Alarm ' + MSG(ii+1) + ' $0A$0D' ;
      TMP := comstr_w( PORT , msg1 ) ;
    end_if ;
    (* ----------------------------------------------------------------------------- *)

    (* ----------------------------------------------------------------------------- *)
    (* INPUT event happens and every 30 second past, send one alarm message *)
    If  ( CNT[ii] = 30 )  then   (* send one alarm when CNT[ ] is from 0, 1, 2, ..., 30 *)
      CNT[ii] := 0 ;  (* re-start from 0 and then count to 30 second to send alarm *)
      (* send one message to COM2 *)
      msg1  :=  'Alarm '  +   MSG(ii+1)  +  ', 30 sec past ! $0A$0D'  ;
      TMP := comstr_w(PORT , msg1) ;
    end_if ;
    (* ----------------------------------------------------------------------------- *)

  end_if ;  (*  "if CNT[ ] > -7 then"  *)

 end_if ;  (*  "if Tick1 then"  *)

 (* Update OLD_IN[ ] *)
 OLD_IN[ii] := IN[ii] ;

end_for ;
```

11.3.6 Whmi_12: Recording I-8017H 's Ch.1 to Ch.4 voltage input in a RAM memory in the WinCon-8xx7. The sampling rate is one record every 0.05 second. The record period is 1 to 10 minutes. Then PC can download this record and display it as a trend curve diagram by M.S. Excel

Note:
If using WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6 and VP-25W7/VP-23W7, the features described in this section can be applied, but you need to use the different contents and files in individual PAC CD. (Their content is similar, but unlike the WinCon CD). Moreover, these controllers - WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6 and VP-25W7 / VP-23W7, do not use this path (\CompactFlash\Temp\HTTP\WebHMI\).
The WP-8xx7, WP-5xx7 and VP-25W7 / VP-23W7 use the path - \Micro_SD\Temp\HTTP\WebHMI\; the XP-8xx7-CE6, XP-8xx7-Atom-CE6 use the path - \System_Disk\Temp\HTTP\WebHMI\. For more information, please refer to the website:

WP-8xx7 : "setup_web_hmi_demo.pdf"
ftp://ftp.icpdas.com/pub/cd/winpac-8xx7/napdos/isagraf/wp-8xx7/wp_webhmi_demo

VP-25W7 / VP-23W7 : "setup_web_hmi_demo.pdf"
ftp://ftp.icpdas.com/pub/cd/vp-25w7-23w7/napdos/isagraf/vp-25w7-23w7/vp-webhmi-demo/ 內

XP-8xx7-CE6 : "xpce6_setup_web_hmi_demo.pdf"
ftp://ftp.icpdas.com/pub/cd/xp-8xx7-ce6/napdos/isagraf/xp-8xx7-ce6/xpce6-webhmi-demo/ 內

This demo is Whmi_12.pia, it can only be used for WinCon-8xx7/8xx6 with driver version 3.36 or later.
New driver: http://www.icpdas.com/products/PAC/i-8000/isagraf-link.htm
The demo program is stored in W-8xx7 CD-ROM:\napdos\isagraf\wincon\demo\ or
ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/demo/

The W-8xx7 provides three ways to record the data. One way is to read/write the file in the RAM Disk (like this example) and this applies to the application that its mininum sampling time is 40 ms (the fastest 25Hz). The files in the RAM Disk will disappear after power off.

Another way is to put the recording data in the "\CompactFlash\" path. The read/write speed for CompacFlash is much slower than RAM Disk. The advantage of CompacFlash is that the files will not disappear after shutdown, but the disadvantage is the read/write speed is too slow.

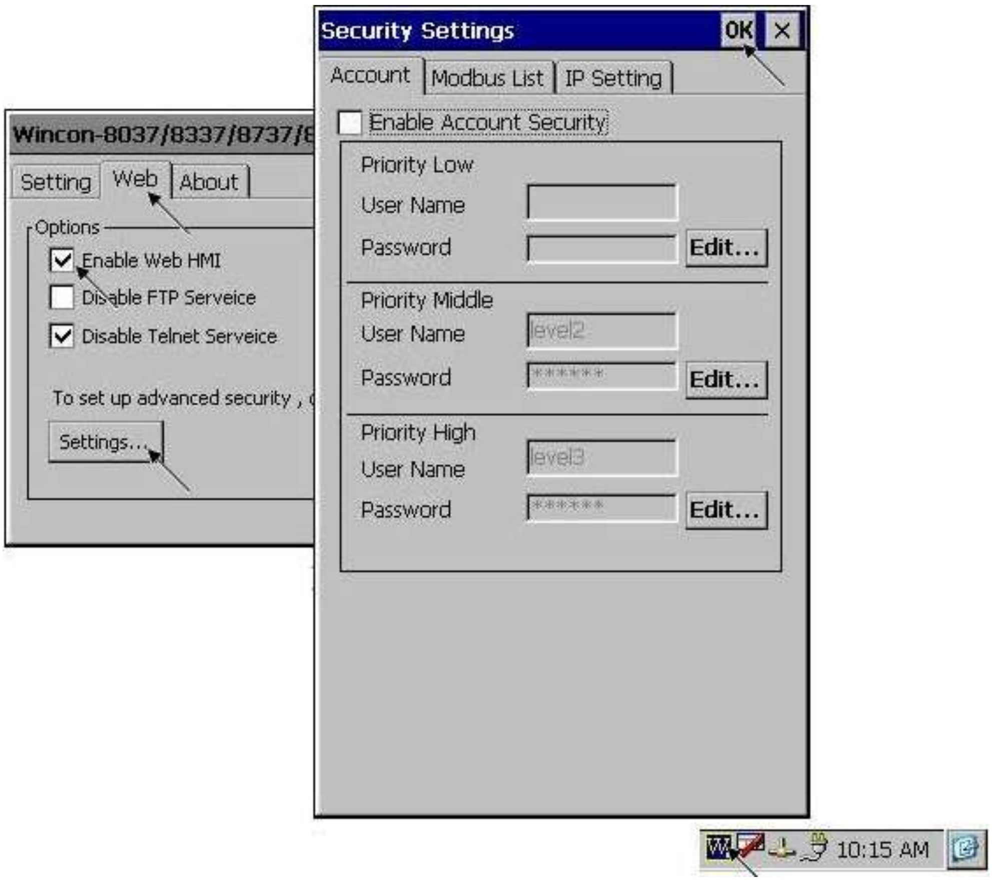
The third is the fastest way, the application with the minimum sampling time can reach 10 ms that means the fastest 100 Hz. Please refer to Section 11.3.10 for the example illustrates. But this minimum sampling time is related to program complexity. If the PLC user program scan time is 100 ms, the sampling time should be greater than 100 ms.

The source code for Web HMI is stored in (Please refer to chapter 3, 4, 5 of WinCon ISaGRAF Getting Started: W-8xx7 CD-ROM:\napdos\isagraf\wincon\english_manu\ "getting_started_w8337.pdf")
W-8xx7 CD-ROM:\napdos\isagraf\wincon\WebHMI_Demo\ whmi_12 or
ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/webhmi_demo/

If you can't find these functions, such as Msg_F, Msg_N, ARY_F_R, AFY_F_W, etc. which installed in ISaGRAF on your PC. Please visit to http://www.icpdas.com/products/PAC/i-8000/isagraf.htm to download the "ICP DAS utilities For ISaGRAF" zip file. After unzip it, run the "setup.exe" file to install these new functions into the ISaGRAF.

How to test this demo:

1. Please plug one I-8024 in W-8xx7's Slot 2, one I-8017H in Slot 3. Then connect Ch1. to Ch.4 voltage output of I-8024 to Ch1. to Ch.4 of I-8017H. Then power up WinCon, Check "Enable Web HMI" option as below. For demo purpose, please don't check "Enable Account Security"."



2. Copy all files of Web HMI's Demo_12 to WinCon's \CompactFlash\Temp\HTTP\WebHMI\ folder by ftp utility (For example, run ftp://10.0.0.103 in Internet Explorer)

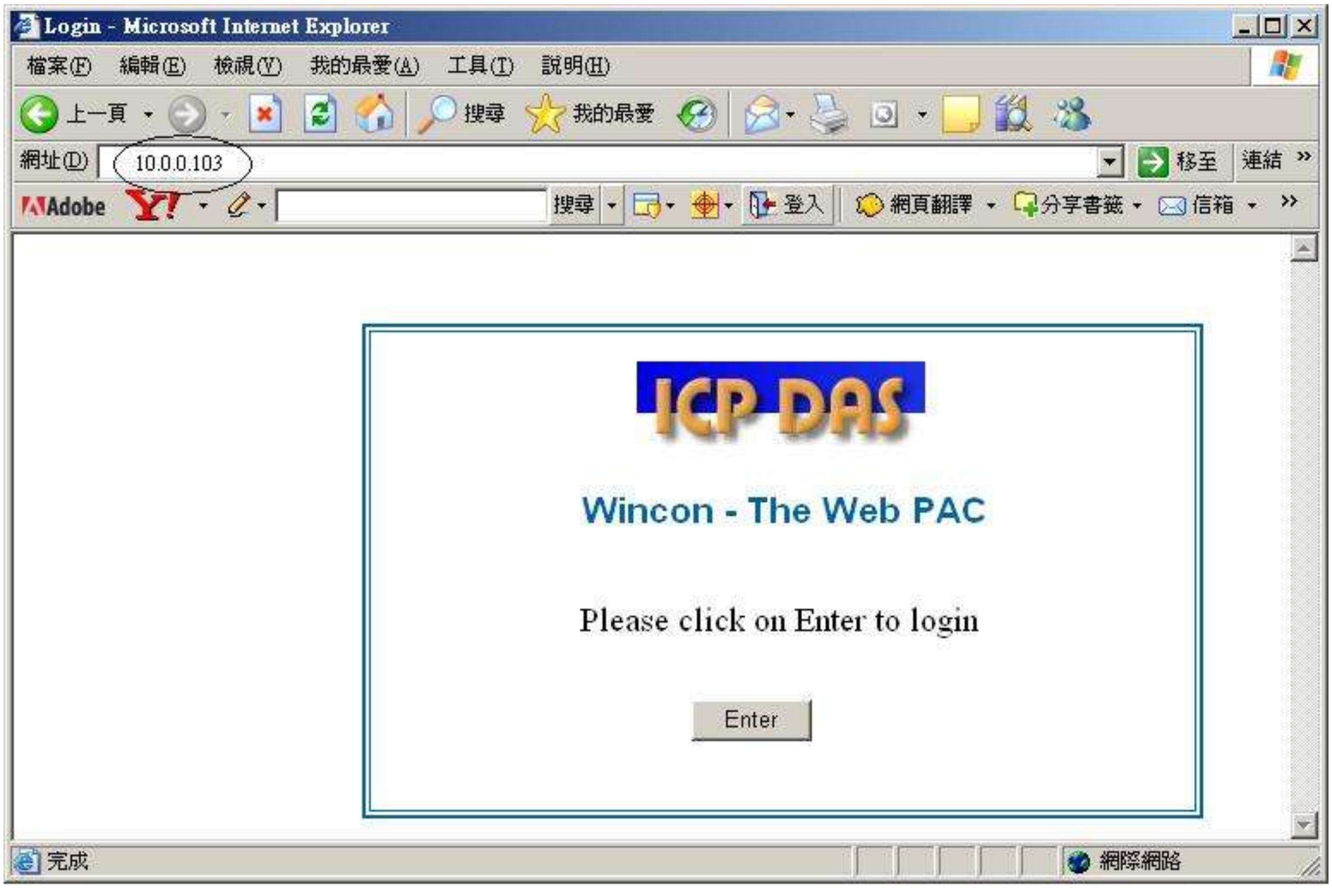
   Web HMI codes resides at
   W-8xx7 CD-ROM:\napdos\isagraf\wincon\WebHMI_Demo\ "whmi_12" folder or
   ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/webhmi_demo/

   Ther are 7 Files plus 2 folder should be copied to WinCon's \CompactFlash\Temp\HTTP\WebHMI\
   Main.htm , menu.htm , index.htm , login.htm , main.dll , login.dll , whmi_filter.dll
   "img" & "msg" folder

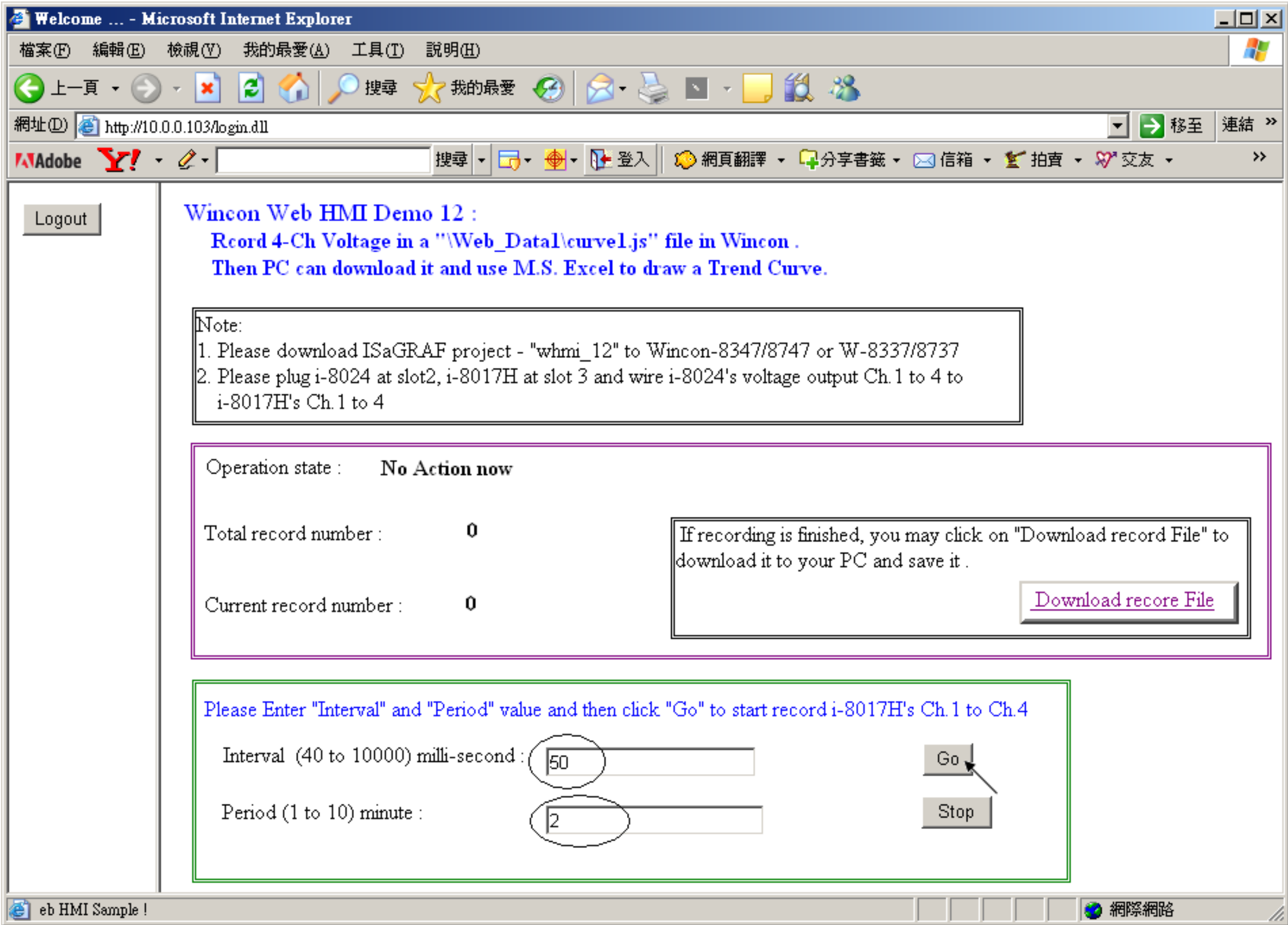
3. Download ISaGRAF project "whmi_12" to W-8xx7. (If using Web HMI as HMI, please finish procedure listed in step 2 first, then do step 3)

   The "Whmi_12.pia" can only run in WinCon-8xx7/8xx6 (not in I-8xx7). It resides at W-8xx7 CD-ROM:\napdos\isagraf\wincon\demo\ or
   ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/demo/

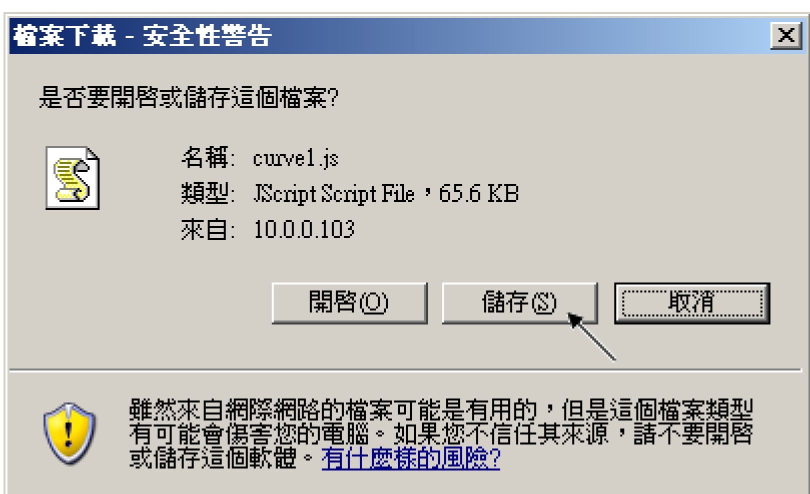
4. PC run Internet Explorer (I.E. version shoulde be 5.0 or later version). Enter W-8xx7 IP. If connecting well, click on "Enter"



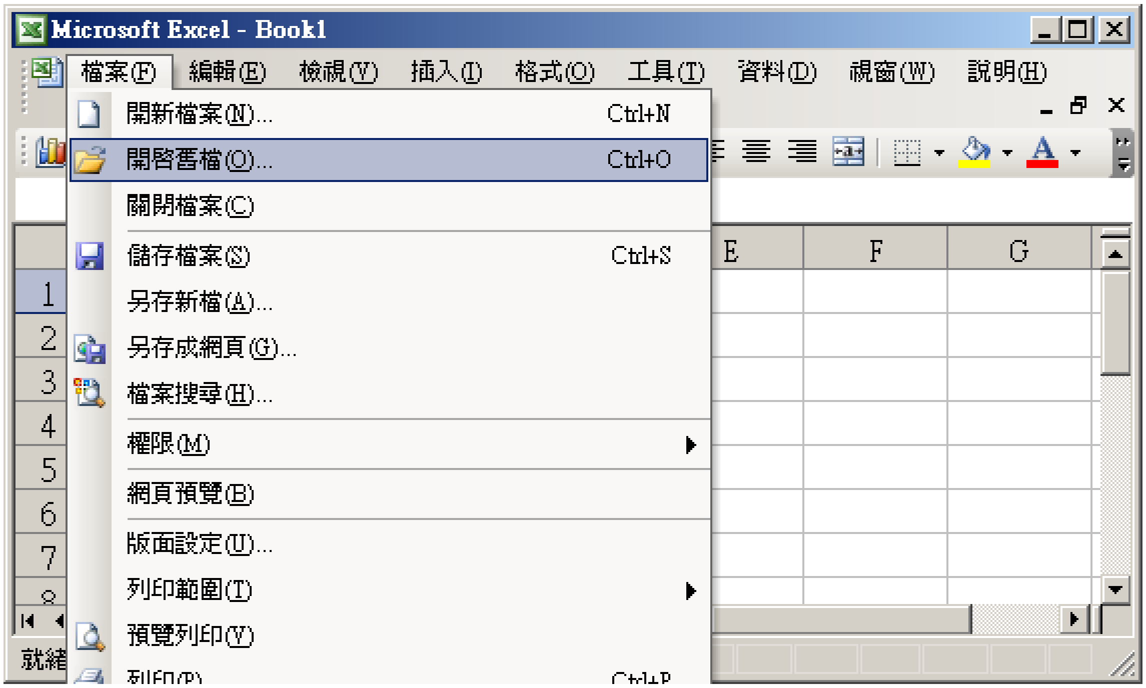
5. Then please enter proper "Interval" value. Unit is 0.001 second (1 milli-second). For example, if enter 50, it means to store one record every 0.05 second. The "Period" is the time period to record. Unit is minute. Then please click on "Go" to start recording. W-8xx7 will then output different voltage in I-8024 Ch.1 to Ch.4 . If user has finished procedure listed in step 1 – "connect Ch1. to Ch.4 voltage output to Ch1. to Ch.4 of I-8017H", the I-8017H Ch1. to Ch.4's voltage input will also change during this period. And they will be recorded.



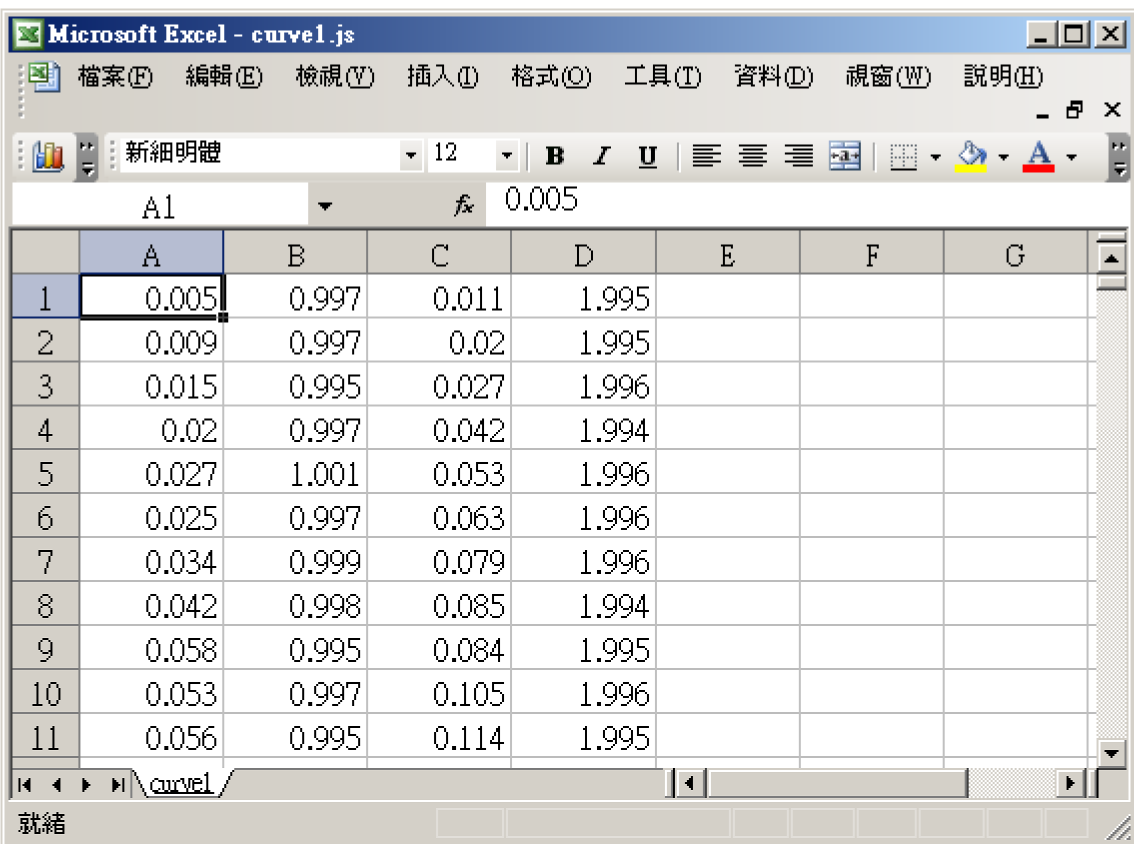
During the recording period, the “Current record number” value will count up. If it reaches the value of “Total record number”, it means recording is finished. Then the ISaGRAF program will store these records to a RAM file automatically. You can see the progress in “Saving state” . If all done, please click on “Download record File” to download this record file to your PC.



6. Then please open this record file – “curver1.js” on M.S. Excel.

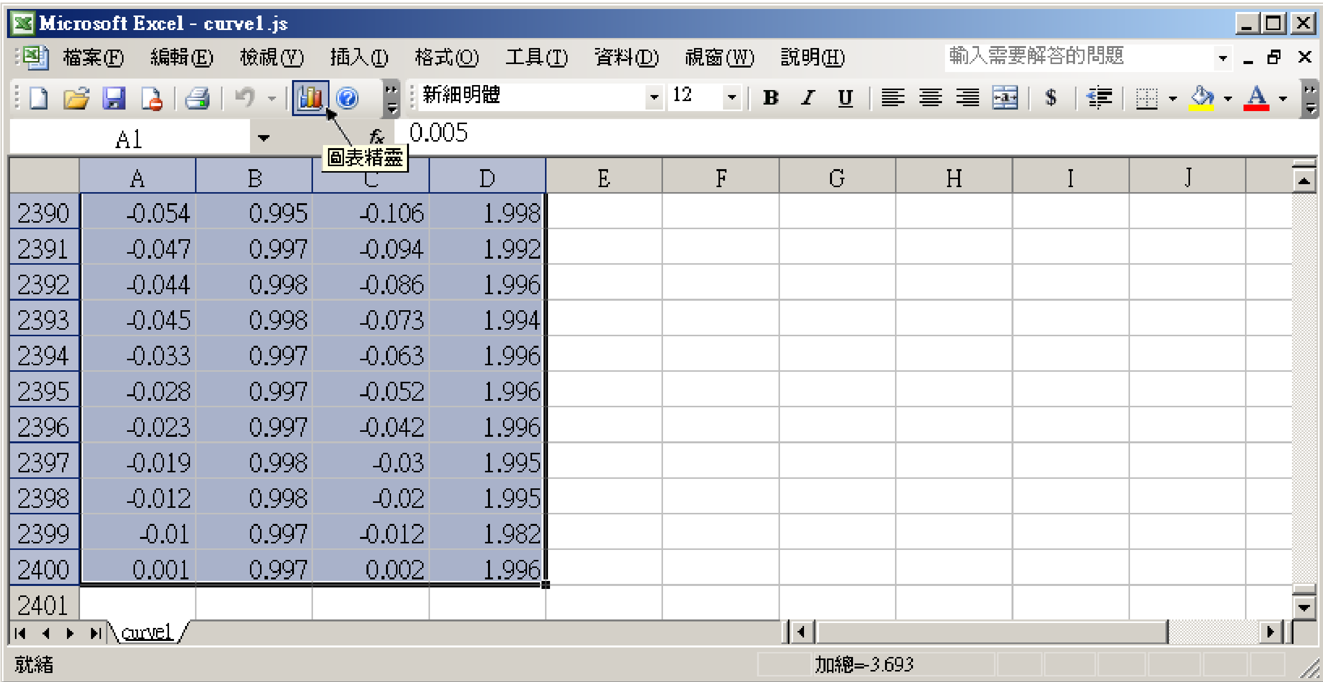


Please click on the first data at the left-top position. Then press and hold in “Shift”, and at the same time press “Ctrl” – “End”. You will see all data been selected.

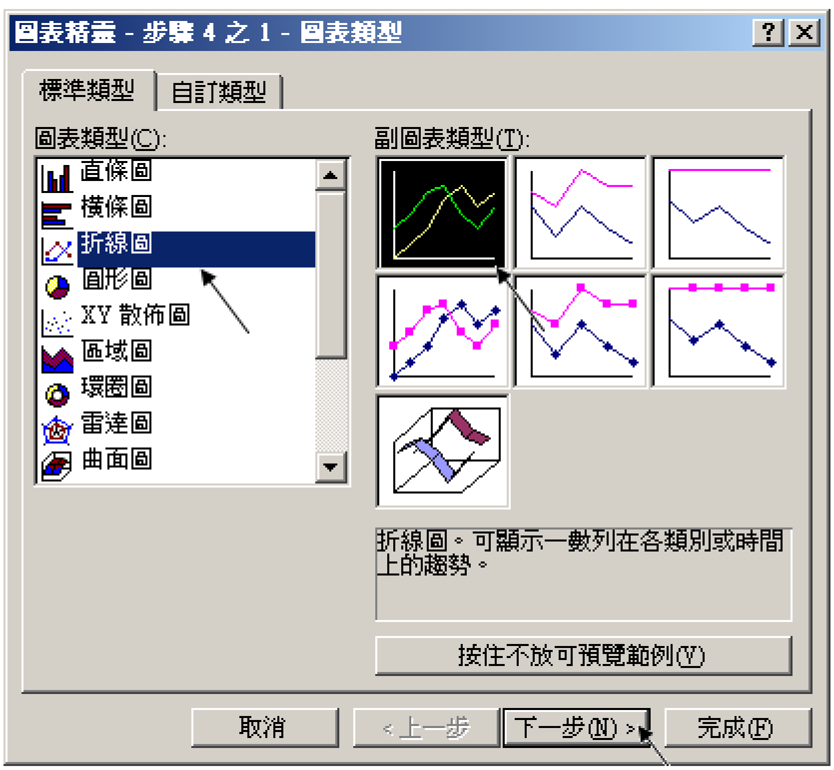


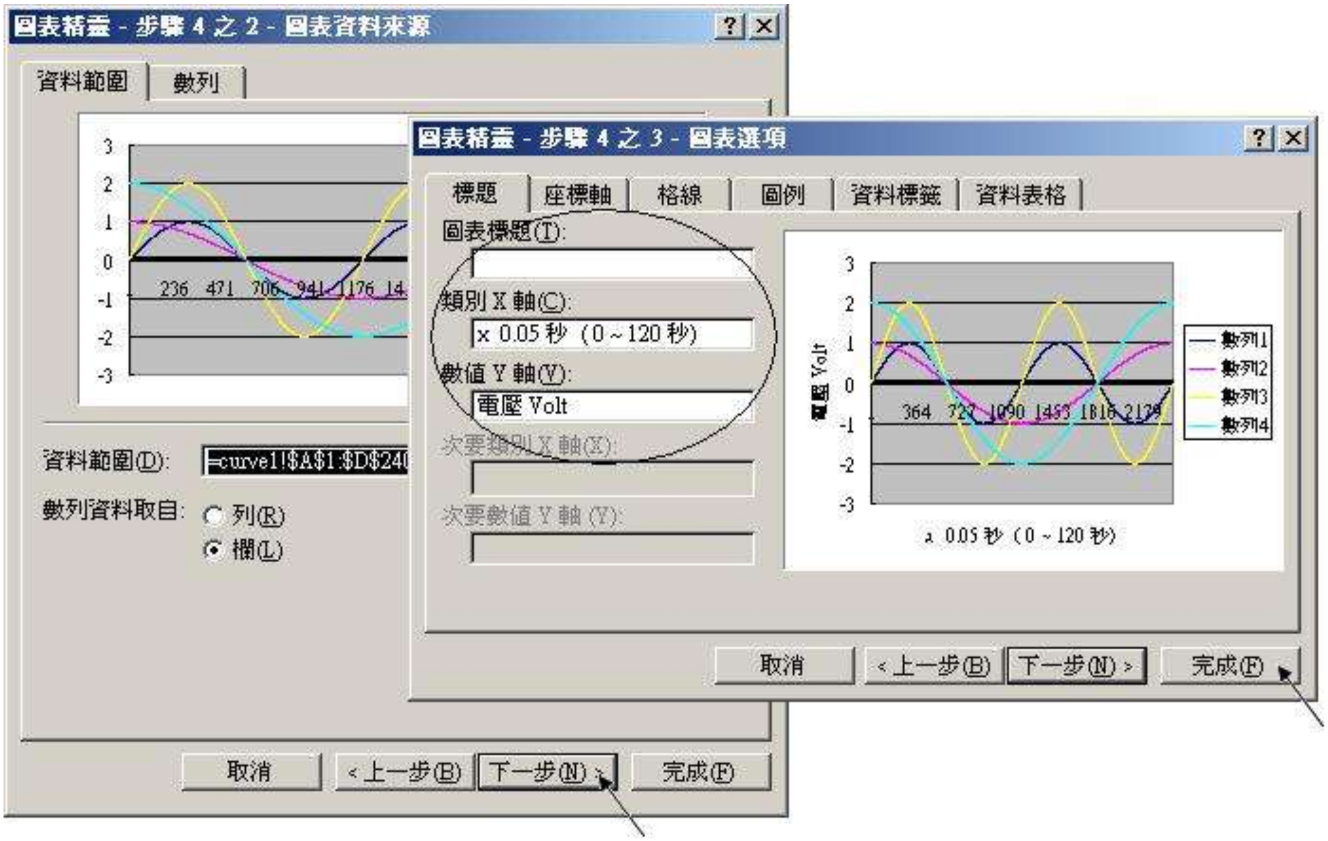
	A	B	C	D	E	F	G
1	0.005	0.997	0.011	1.995			
2	0.009	0.997	0.02	1.995			
3	0.015	0.995	0.027	1.996			
4	0.02	0.997	0.042	1.994			
5	0.027	1.001	0.053	1.996			
6	0.025	0.997	0.063	1.996			
7	0.034	0.999	0.079	1.996			
8	0.042	0.998	0.085	1.994			
9	0.058	0.995	0.084	1.995			
10	0.053	0.997	0.105	1.996			
11	0.056	0.995	0.114	1.995			

Then click on  (Chart Wizard).

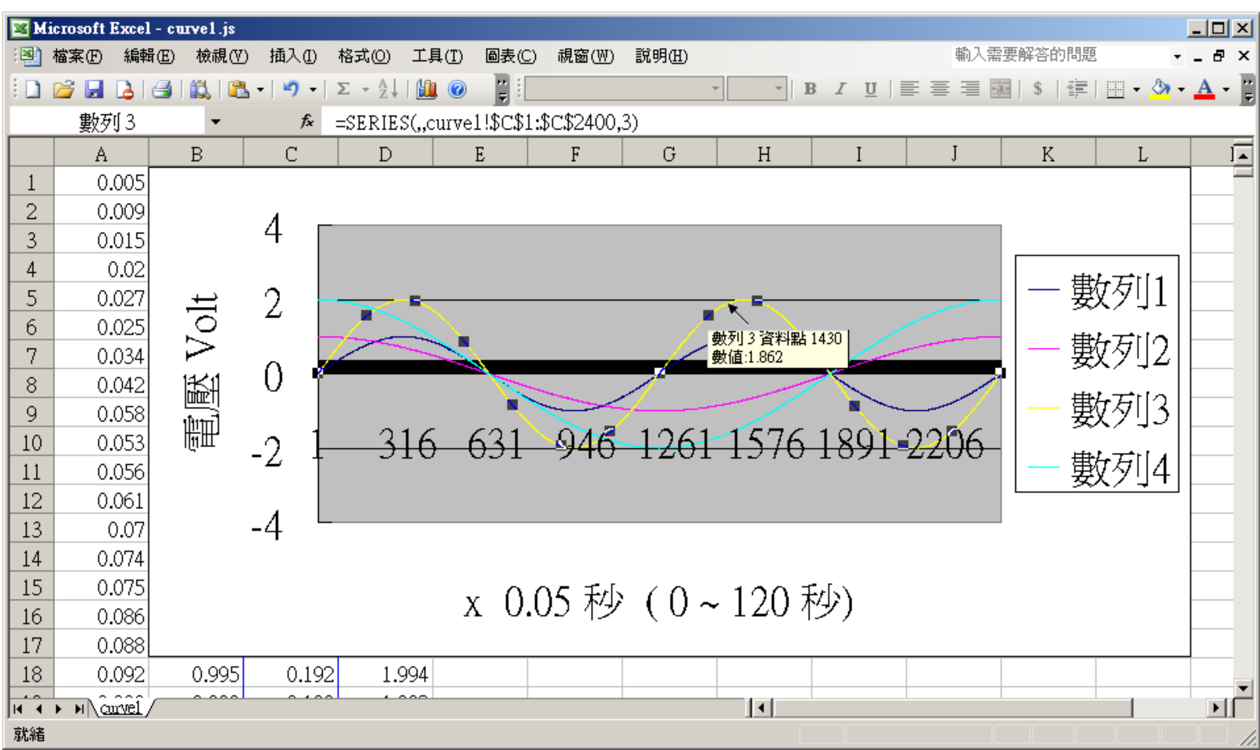


Please select the “Line Chart” on the left-hand side. And select the left-top type on the righthand side. Then go Next.

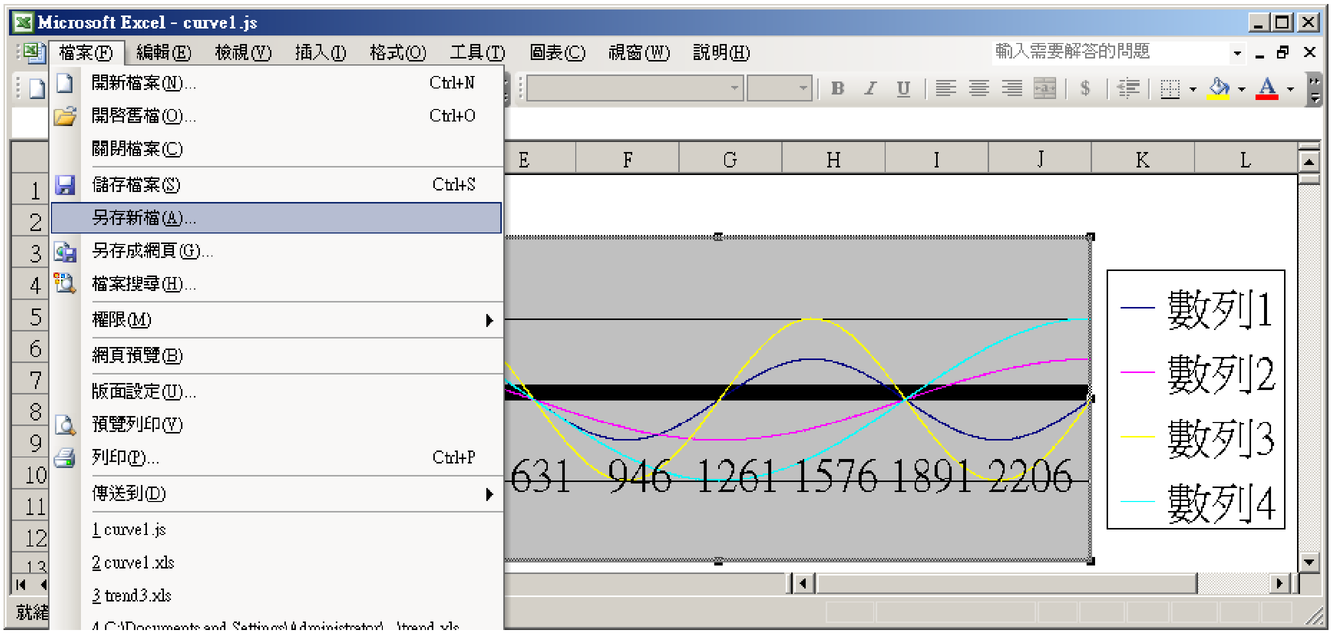


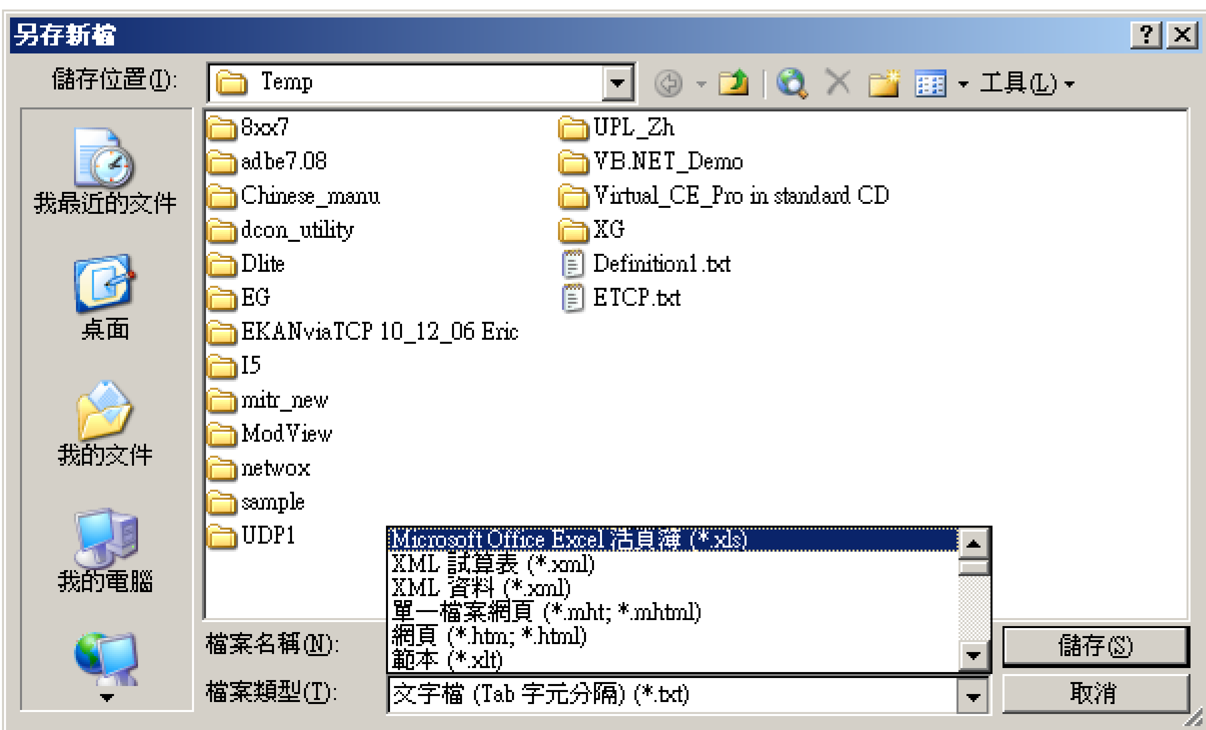


By the procedure, you will get the trend curve as the below window. You can modify its size, or check at any trend line. If you move your mouse to point at some position at the trend line, the related data is shown.

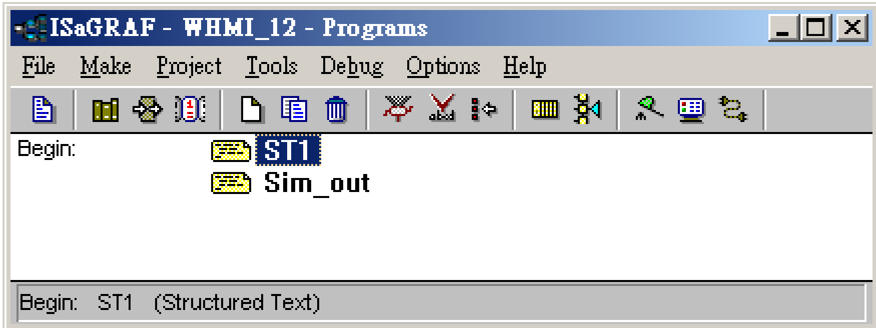


Please save this trend curve diagram as a “Microsoft Office Excel (*.xls)” format. Then at any later time, you can open it to display the trend curve directly.





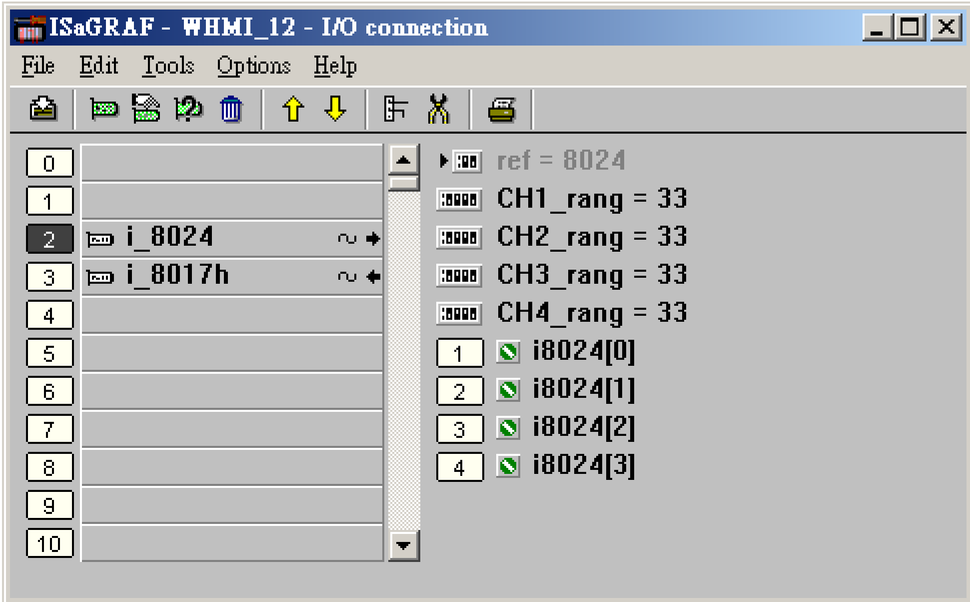
ISaGRAF project architecture:

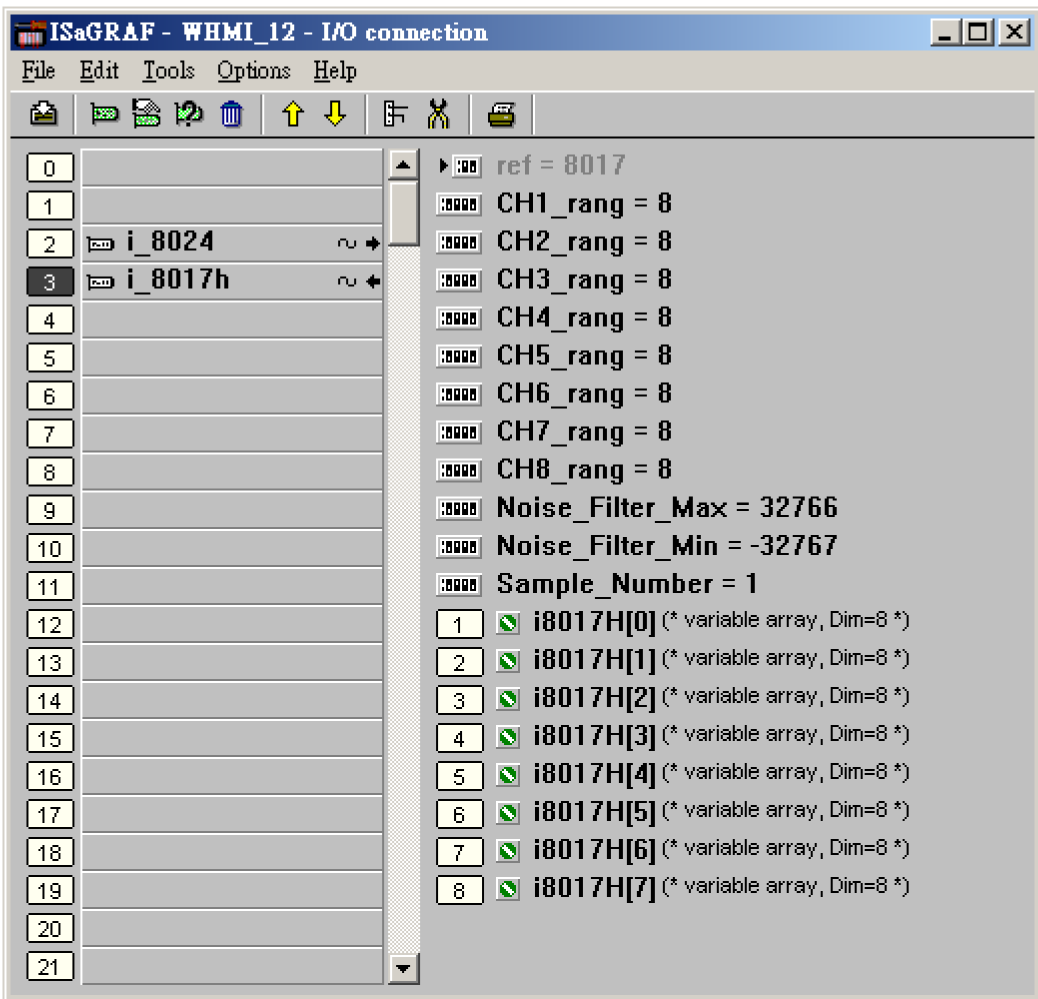


Variables:

Name	Type	Attribute	Description
Go1	Boolean	**Internl**	Set as True to start, **addr defined as 21** (Hex. is 15)
Stop1	Boolean	**Internal**	Set as True to stop, **addr defined as 22** (Hex. is 16)
TMP	Boolean	**Internal**	For temporal use
MUM_CH	Integer	**Constant**	How many chanels in I-8017H to record ? We use 4 channels in this demo (Ch.1 to 4)
File1	Integer	**Internal**	File ID
STEP1	Integer	**Internal**	Recording state. 0: No action , 1: recording , 2: finished
Period1	Integer	**Internal**	How long to record? unit is minute, **addr as 3**
Interval1	Integer	**Internal**	How long to save a record ? unit is ms, **addr as 1**
Total_record1	Integer	**Internal**	How many records in this recording action ? This value is calculated by the ISaGRAF program automatically. **addr declared as 5**
Record_cnt1	Integer	**Internal**	Current finished record count. **addr declared as 7**
ii	Integer	**Internal**	To use in "for" loops
i8017H[0..7]	Integer	**Input**	Variable array, Dim as 8. link to I-8017H's Ch.1 to 8.
Volt1[0..7]	REAL	**Internal**	Variable array, Dim declared as 8. The voltage value converted from "i8017H[0..7]"
i8024[0..3]	Integer	**Output**	Variable array, Dim declared as 4. link to I-8024's Ch1 to 4.
T1	Timer	**Internal**	Operation Timer
T1_next	Timer	**Internal**	The time to get and save next record
T1_Interval	Timer	**Internal**	The interval time between two record
File_name1	Message	**Internal**	File name, Len is 64, init as \Web_Data1\curve1.js Web HMI support only RAM Disk File in \Web_Data1\ , If the file is in CompactFlash File, Web HMI support only in \CompactFlash\Temp\HTTP\Data\ (Please refer to Chapter 11.2 - Whmi_08 demo)
Msg1	Message	**Internal**	Operation state message, Len is 255, init as "No Action now", addr as 41 (Hex. is 29)
Str1	Message	**Intenal**	Len is 255, internal use

I/O Connection:





ST Program – Sim_out

```
-----------------------------------------------------------------------------------------------------------------
(* Output I-8024's Ch1 to Ch4 as different Sin , Cos voltage curve *)

(*   2 * Pi * T1 / 60000  =  T1 * 1.047197E-4    *)
(*   2 * Pi * T1 / 120000 = T1 *  5.235985E-5 *)

i8024[0]  :=  ANA(  sin( REAL(T1) *  1.047197E-4 ) * 3276.8  )  ;
i8024[1]  :=  ANA(  cos( REAL(T1) *  5.235985E-5  ) * 3276.8  ) ;
i8024[2]  :=  ANA(  sin( REAL(T1) *  1.047197E-4 ) * 6553.6  ) ;
i8024[3]  :=  ANA(  cos( REAL(T1) *  5.235985E-5  ) * 6553.6  ) ;
-----------------------------------------------------------------------------------------------------------------
```

ST Program – ST1

```
-----------------------------------------------------------------------------------------------------------------

(*  If  set "Stop1" as TRUE, stop the "T1" timer and set the "STEP1" as 0 *)
if   Stop1   then
  Stop1  :=  False  ;
  STEP1   :=  0  ;      (* set the "STEP1" as 0;  0: No action *)
  TStop(T1)   ;     (* stop the "T1" timer *)
  T1 := T#0s  ;
  Msg1  :=  'User stop recording !'  ;
end_if  ;

(*  If  set "Go1" as TRUE, it will start to action *)
if   Go1   then

   Go1  :=  False  ;

   (* STEP1 :  0: No action , 1: recording , 2: finished *)
   if    STEP1 = 1   then

       Msg1 := 'It is still recording now ...' ; (* Update the operation status *)

   else

     (* Check if the value "Interval1" is correct; the value is between 40 ~ 10000; Unit is 0.001 second *)
       If   ( Interval1  <  40 )   or   ( Interval1  >  10000 )  then
         Msg1  :=  'Wrong Interval value, it should be in 40 to 10000 milli-second !'  ;

     (* Check if the value "Period1" is correct; the value is between 1 ~ 10; Unit is minutes *)
       elsif  ( Period1  <  1 )   or  ( Period1  >  10 )   then
         Msg1  :=  'Wrong Period value, it should be in 1 to 10 minute !'  ;

       else
```

```
        (*  The value is corrent and then create a new file *)
        File1  :=  F_creat( File_name1 )  ;
        if  File1=0  then    (* If failed to create a new file, No actions *)

          Msg1  := 'Create File ' + 'File_nam1 Error !!!'  ;

        else        (* If successful to create a new file, start actions *)

          TMP   :=  F_close( File1 )  ;      (* Close file after completing the actions *)

          total_record1 := ( Period1 * 60000 ) / Interval1 ;  (* Calculate total record number *)
          record_cnt1  :=  0  ;     (* Reset current record count as 0 *)
          STEP1  :=  1  ;     (* Set STEP1 as 1: recording *)
          Msg1  :=  'Recording now ...'   ;

          (* Start ticking T1 from 0 second *)
          T1  :=  T#0s  ;
          T1_Interval  :=  TMR( Interval1 )  ;
          T1_next  :=  T1  +  T1_Interval  ;
          TStart( T1 ) ;

        end_if;

      end_if ;

    end_if ;

  end_if ;

  if  STEP1=1  then         (*  1: Recording *)

    if T1 >= T1_next then  (* Start to prepare one record when T1 is reach to the next record time *)

      T1_next  :=  T1_next  +  T1_Interval  ;  (* Re-calculate next T1 *)

(* The timing range is 23 hours, 59 minutes, 59 seconds, so we reset it as 0 in advance just over 20 hours. *)
(* The content below is used for user's reference, the timer limit is only 10 minutes in this example *)
(* If the WinCon's RAM Disk only run the ISaGRAF, it can support up to 16MB file record*)
      (* ---------------------------------------------------------------- *)
      if  T1 >=  T#20h  then
        T1  :=  T#0s  ;
        T1_next  :=  T1  +  T1_Interval  ;
      end_if  ;
      (* ---------------------------------------------------------------- *)

      (* Open a file to write a record *)
      (* ---------------------------------------------------------------- *)
```

```
File1  :=  F_wopen( File_name1 )  ;     (* Enable read/write mode *)
   if  File1=0  then
     (* If failed to open file, No actions *)
     Msg1  :=  'Save File '  +  File_name1  + ' Error !!!'  ;
     STEP1  :=  0  ;
     Tstop( T1 ) ;      (* stop T1 timer *)
     return  ;          (* leave the  ST program *)
     T1 := T#0s  ;
   end_if  ;
   (* ------------------------------------------------------------ *)

   str1  :=  ''  ;      (* Init str1 as empty string *)
   for   ii   :=   0   to   NUM_CH - 1   do

     (* Convert the i8017H's analog input value to voltage value *)
     Volt1[ii] := Real( i8017H[ii] ) * 0.000305176  ;  (* 10.0 / 32768 = 0.000305176 *)
     str1 := str1 + Rea_Str2( Volt1[ii] , 3 ) + '$09' ;  (* The separator is <TAB> *)

   end_for  ;

   str1  :=  str1  +  '$0D$0A'  ;  (* add <CR> <LF> at the end of each row *)
   TMP  :=  F_end( File1 )  ;    (* Move to the end of the file to write *)
   TMP  :=  F_writ_s( File1  ,  str1 )  ;    (* To write stings *)
   TMP  :=  F_close( File1 ) ;       (* Close file after complete the operation *)

   record_cnt1  :=  record_cnt1 + 1  ;    (* Current record count plus 1 *)

   (* Check if record numbers reach the end *)
   if  ( record_cnt1  >=  total_record1 )   then
     STEP1 := 2 ;     (* Set "STEP1" as 2: Recond Finished *)
     Msg1 := 'Record is finished ! You may download the record file to your PC now !' ;
     Tstop( T1 ) ;    (* Stop T1 timer *)
     T1 := T#0s  ;
   end_if  ;

  end_if ;

end_if;
```

11.3.7 Demo_71: Recording I-8017H 's Ch.1 to Ch.4 voltage input in S-256 / 512 in I-8437-80 or I-8837-80 . The sampling time is one record every 0.05 second. The record period is 1 to 10 minutes. Then PC can download this record and display it as a trend curve diagram by M.S. Excel

This demo is the "Demo_71" can run in I-8437-80 or in I-8837-80 (80MHz) . The controller driver should be version of 3.19 or later version.

New drive: http://www.icpdas.com/products/PAC/i-8000/isagraf-link.htm
"demo_71.pia" resides at I-8000 CD-ROM:\napdos\isagraf\8000\demo\ or
ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/8000/demo/ or

Note: If using iP-8xx7, you must instead use the I-8017HW and I-8024W I/O modules.

I-8437-80 and I-8837-80 controller 's CPU is running at 80MHz . The speed is about 2 to 4 time faster than I-8437 and I-8837 (40MHz CPU). So it can record minimum to 25 milli-second sampling data. While using I-8437 and I-8837 (40MHz) , it can record only 100 ms or above sampling data. This minimum sampling time depends on your ISaGRAF program 's PLC scan time. If the PLC scan time is large, like 200 ms, then you can do sampling only larger than 200 ms.

How to test this demo ?

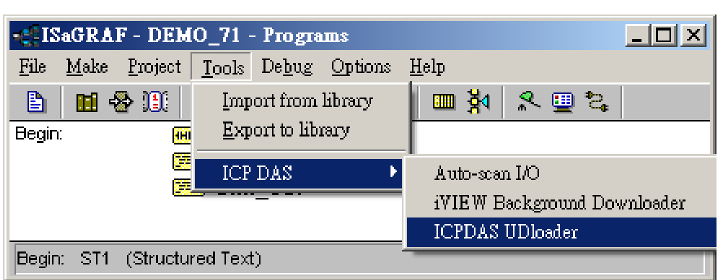
You may run VB 6.0 - "Demo_6" in your PC to on-line control this I-8837-80 via ethernet (refer to section 11.2). Or just push the pushbutton on the front panel of the I-8437-80/8837-80 to start / stop it.

1. Please plug one I-8024 in I-8437-80 's Slot 2, one I-8017H in Slot 3. (Note: the left-most I/O slot No. of the I-8xx7 is 0 , not 1) .Then connect Ch1. to Ch.4 voltage output of I-8024 to Ch1. to Ch.4 of I-8017H. Then power up this I-8437-80.

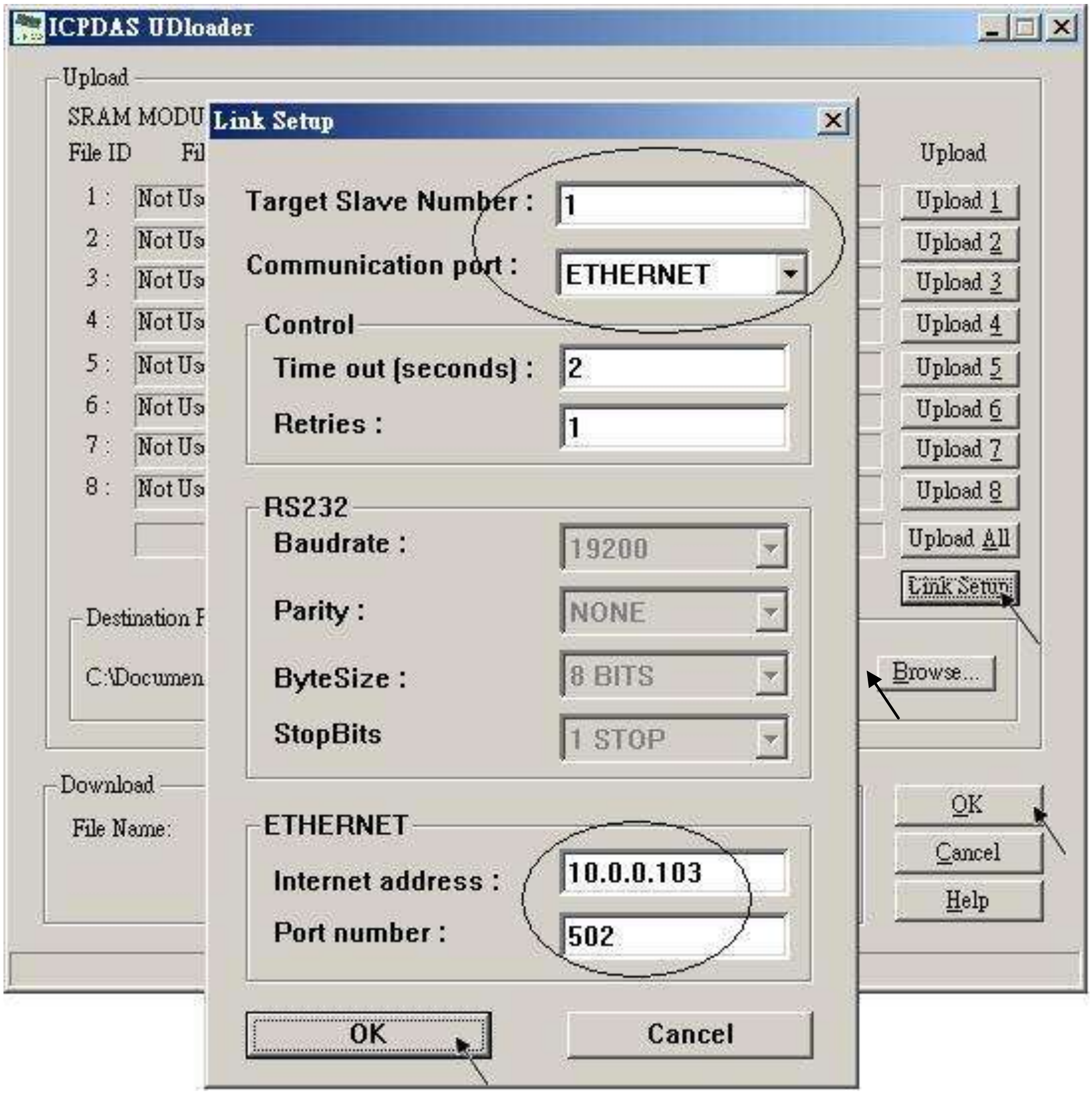
2. Download the ISaGRAF project - "Demo_71" to the I-8437-80 .

3. At run time, you may press pushbutton 1 to start recording. Then it will record data during 2 minutes. You can see the displayed number on the front panel decreasing to 0 and blinking. If recording is finished, the 3 Leds on the front panel will blink and the displayed number will be 0. To stop at anytime, just press pushbutton 4 once.

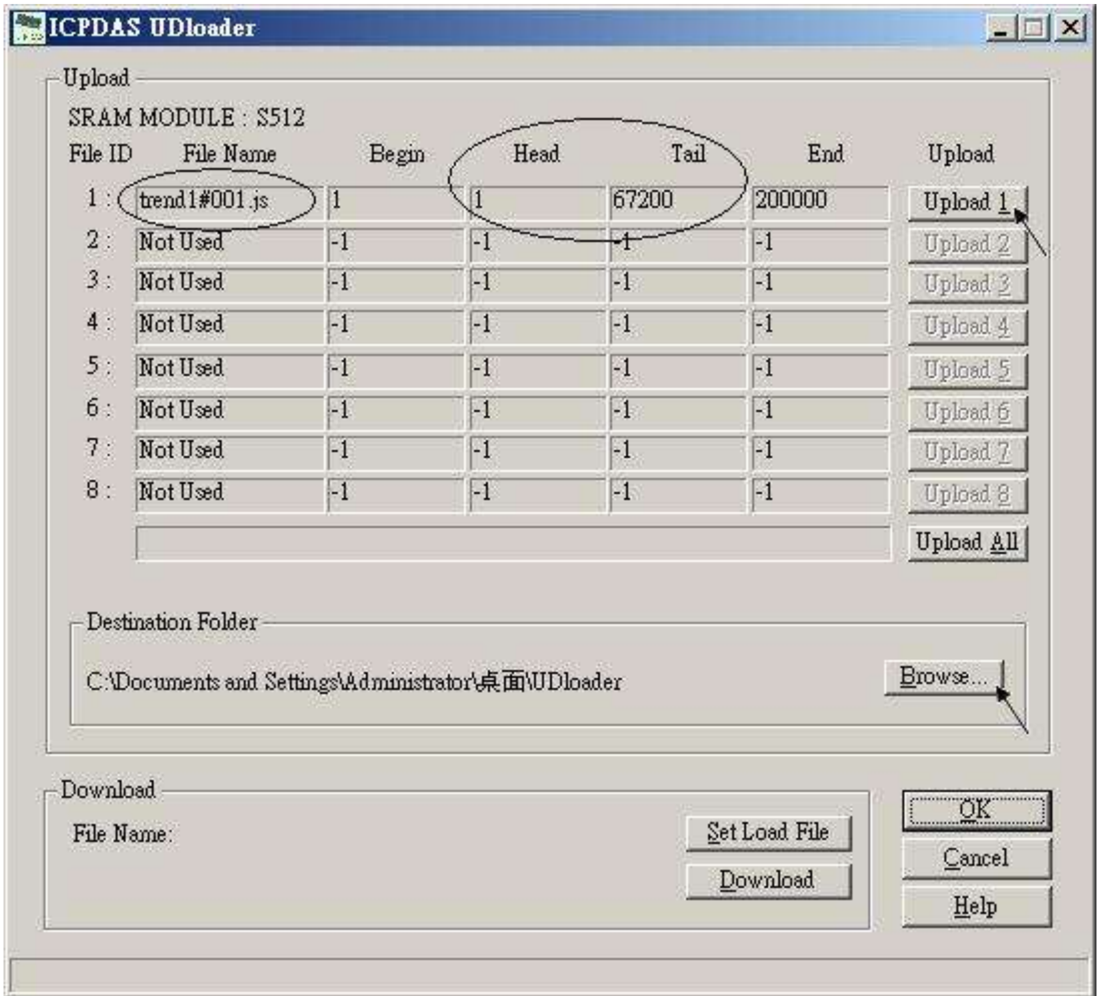
4. Whe recording is finished, please run UDLoader in your PC to upload the record file in the S-256 / 512 to PC. If your PC is currently running ISaGRAF workbench, please run "Tools" – "ICP DAS" – "ICPDAS UDloader". Then you will see the window listed in step (5) below.



If your PC is not running ISaGRAF workbench, please copy i-8000-CD:\napdos\isagraf\some_utility\ "udloader.exe" to for example PC windows 's desktop. Then please run it. Set proper "Link Setup" (If click on "Browse" , you may modify the file upload location path). Then click on "Ok" and "Ok" to save this setting, then run it again.



5. If the controller is well connected, you will see a File Name displayed on the below window. Value of "Head" and "Tail" is the current size of the record file in the S-256 / S-512. The below example shows 67,200 bytes. (Note: S-256 / S-512 has size limitation, please refer to Chapter 10.3). you may modify the file upload location path by click on "Browse …" . Then please click on "Upload 1" to upload this record file to your PC.

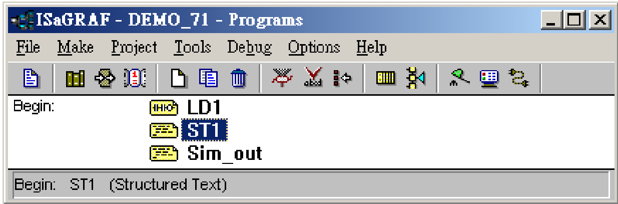


Then you may check if the record file is uploaded to your PC at the same path.

6. Then please open this record file - "trend1.js" on M.S. Excel. Please refer to section 11.3.6 – step (6).

Note:
You may run VB 6.0 - "Demo_6" in your PC to on-line control this I-8837-80 via ethernet (refer to section 11.2).

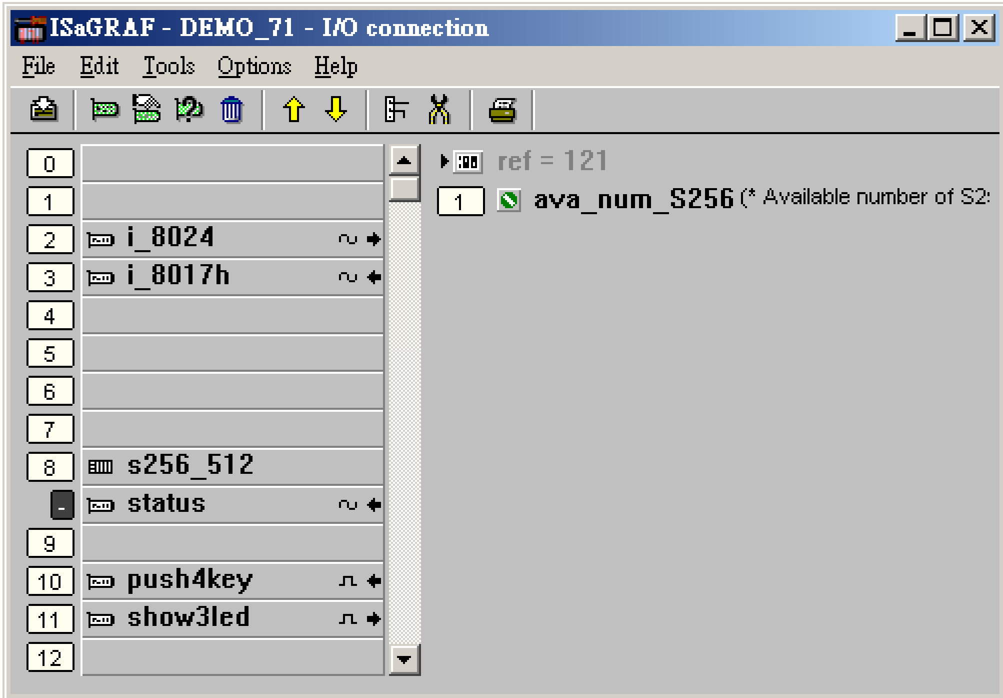
ISaGRAF project architecture:



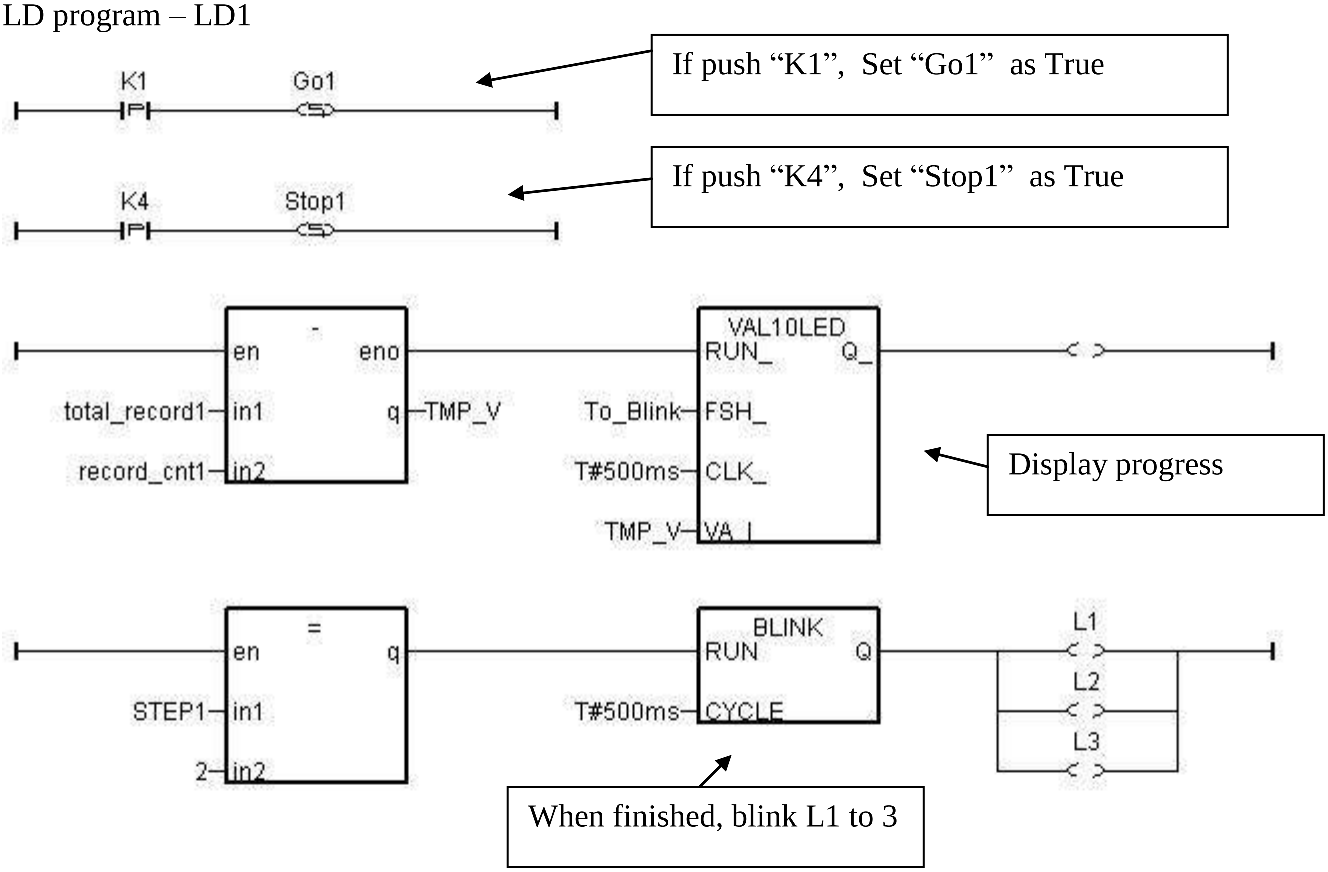
Variable :

Name	Type	Attribute	Description
INIT	Boolean	**Internal**	Init as True
Go1	Boolean	**Internl**	Set as True to start, **addr defined as 21** (Hex. is 15)
Stop1	Boolean	**Internal**	Set as True to stop, **addr defined as 22** (Hex. is 16)
TMP	Boolean	**Internal**	Internal use
L1 , L2 , L3	Boolean	**Output**	connect to show3Led 's Ch.1 to Ch.3
K1 , K4	Boolean	**Input**	Connect to push4key 's Ch.1 to Ch.4 Push K1 to start recording. K4 to stop
To_Blink	Boolean	**Internal**	To control blinking of the number on the front pannel
MUM_CH	Integer	**Constant**	How many chanels in I-8017H to record ? We use 4 channels in this demo (Ch.1 to 4)
Ava_num_s256	Integer	**Input**	Connect to "S256_512" 's Ch.1 . if value is 0, it means can not find S-256 / 512
Current_pos1	Integer	**Internal**	Current operating byte address in S-256 / 512
Len1 , TMP_V	Integer	**Internal**	Internal use
File_begin1 File_end1	Integer	**Constant**	The Begin & End byte address in the S-256 / S-512 allocated for the record file
STEP1	Integer	**Internal**	Recording state. 0:No action , 1:recording , 2:finished
Period1	Integer	**Internal**	How long to record ? unit is minute, **addr as 3**
Interval1	Integer	**Internal**	How long to save a record ? unit is ms, **addr as 1**
Total_record1	Integer	**Internal**	How many records in this recording action ? This value is calculated by the ISaGRAF program automatically. **addr declared as 5**
Record_cnt1	Integer	**Internal**	Current finished record count. **addr declared as 7**
ii	Integer	**Internal**	To use in "for" loops
i8017H[0..7]	Integer	**Input**	Variable array, Dim as 8. link to I-8017H 's Ch1 to Ch. 8
Volt1[0..7]	REAL	**Internal**	Variable array, Dim declared as 8. The voltage value converted from "i8017H[0..7]"
i8024[0..3]	Integer	**Output**	Variable array, Dim declared as 4. link to I-8024 's Ch1 to Ch. 4
T1	Timer	**Internal**	For counting time
T1_next	Timer	**Internal**	The time to get and save next record
T1_Interval	Timer	**Internal**	The interval time between two record
Msg1	Message	**Internal**	Operation state message, Len is 255, init as "No Action now", **addr as 41** (Hex. is 29)
Str1	Message	**Intenal**	Len is 255, internal use

IO connection:



LD program – LD1



ST program – Sim_out

```
-----------------------------------------------------------------------------------------------------------------
(* Output I-8024 's Ch1  to Ch4 as different voltage curve *)
(*   2 * Pi * T1 / 60000  =  T1 * 1.047197E-4    *)
(*   2 * Pi * T1 / 120000 = T1 *  5.235985E-5 *)
i8024[0]  :=  ANA(  sin( REAL(T1) *  1.047197E-4 ) * 3276.8  )  ;
i8024[1]  :=  ANA(  cos( REAL(T1) *  5.235985E-5  ) * 3276.8  ) ;
i8024[2]  :=  ANA(  sin( REAL(T1) *  1.047197E-4 ) * 6553.6  ) ;
i8024[3]  :=  ANA(  cos( REAL(T1) *  5.235985E-5  ) * 6553.6  ) ;
-----------------------------------------------------------------------------------------------------------------
```

ST program – ST1

```
-----------------------------------------------------------------------------------------------------------------
if  INIT  then

  INIT := FALSE  ;      (* set as False to only do it once at 1st PLC scan *)

  if   ava_num_S256 = 0  then   (* S256 / S512 is not installed in I-8xx7, return *)
    Msg1  :=  'S256 / S512 is not installed in I-8xx7 controller !' ;
    Return ;
  end_if  ;

  (* Allocate S256/512 memory of byte No.1 to 200,000 for file ID = 1 , name='trend1.js' *)
  TMP  :=  S_FL_INI( 1 , 'trend1.js' , File_begin1 , File_end1) ;
  TMP  :=  S_FL_AVL( 1 , -1 , -1 ) ;    (* Init file content as No data at the beginning *)

end_if  ;

if   ava_num_S256 = 0  then   (* S256 / S512 is not installed in I-8xx7, return *)
  return ;
end_if ;

(* If stop command is gived *)
if  Stop1  then
  Stop1 := False ;
  STEP1 := 0 ;    (*  0: no action *)
  TStop(T1)  ;     (* stop T1 *)
  T1 := T#0s ;
  Msg1 := 'User stop recording !' ;
  To_Blink := FALSE ;  (* Set as FALSE not to blink the display value *)
end_if ;

(* Get file status in S256 or S512 :  -1: PC hasn't loaded the file yet *)
(* others: the end byte No. that PC has load the file *)
TMP_V  :=  S_FL_STS(1) ;
if   TMP_V <> -1  then       (* PC has load the file *)
  TMP  :=  S_FL_RST(1)  ;      (* reset status to -1 ( PC hasn't load the file yet) *)
end_if  ;
```

```
(* If start command is gived *)
if  Go1  then
  Go1 := False ;

  (* STEP1 :  0: no action , 1: recording , 2: recond finished *)
  if   STEP1=1   then
     Msg1  :=  'It is still recording now ...'  ;
  else

     (* Check interval valid or not *)
     (* we assume 25 to 10000 ms is valid in this example *)
     (* If your average PLC scan time is larger, for example, near 20 ms,
        Please use Interval larger than 25 ms. Or the record time won't be correct *)
     if   (Interval1 < 25)  or  (Interval1 > 10000)  then
       Msg1  :=  'Wrong Interval value, it should be in 25 to 10000 milli-second !' ;

     (* Check period valid or not *)
     (* we assume 1 to 10 minute is valid in this example *)
     Elsif  ( Period1 < 1)   or  (Period1 > 10)  then
       Msg1  :=  'Wrong Period value, it should be in 1 to 10 minute !' ;

     else

       (* parameter is correct, start recording *)
       total_record1 := (Period1 * 60000) / Interval1 ;  (* calculate total record number *)
       record_cnt1 := 0 ;  (* reset current record count as 0 *)
       STEP1 := 1 ;  (* set step as 1:recording *)
       Msg1  :=  'Recording now ...'  ;

       (* start ticking T1 from 0 second *)
       T1 := T#0s ;
       T1_Interval  :=  TMR(Interval1) ;
       T1_next := T1 + T1_Interval ;
       TStart(T1) ;   (* ticking now *)
       Current_pos1 := 1 ;  (* reset current data position in S256/S512 as 1 *)
       To_Blink := TRUE ;  (* Set as TRUE to blink the display value *)

     end_if  ;
  end_if  ;
end_if  ;

 (* in reconrding state *)
if    STEP1=1    then
 (* store one record *)
 if   T1 >=  T1_next   then

   (* Re-calculate next T1 *)
   T1_next  :=  T1_next  +  T1_Interval ;
```

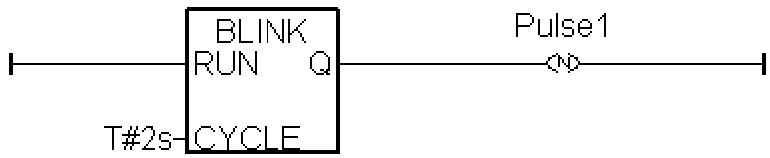
```
(* T1 will be overflow after T#23h59m59s999ms, so reset it at T#20h *)
if   T1 >= T#20h   then
  T1  :=   T#0s ;
  T1_next   :=   T1 + T1_Interval ;
end_if ;

str1  :=  '' ;  (* init str1 as empty string *)
for   ii  :=  0  to  NUM_CH-1  do

   (* convert i8017H analog input value to Volt value *)
   Volt1[ii]  :=  Real(i8017H[ii]) * 0.000305176 ;    (* 10.0 / 32768 = 0.000305176 *)
   str1  :=  str1 + Rea_Str2(Volt1[ii] , 3) +  '$09' ;     (* delimiter is <TAB> character *)

end_for ;

str1  :=  str1 + '$0D$0A'  ;  (* add <CR> <LF> at the end of each row *)
Len1  :=  MLEN(str1) ;  (* get string length *)

(*  data number larger than file's max. allocated memory *)
if   (Current_pos1 + Len1 - 1)  >  File_end1   then
   STEP1 := 0;  (* 0: no action *)
   Msg1 := 'File allocated memory is not enough to hold the data ! '  ;
   Tstop(T1) ;
   T1 := T#0s ;
   To_Blink := FALSE ;  (* Set as FALSE not to blink the display value  *)
   Return ;
end_if  ;

TMP  :=  S_M_W( Current_pos1 , Len1 , str1 ) ;   (* write all bytes in str1 to S256/S512 *)
Current_pos1  :=  Current_pos1 + Len1 ;          (* Current position move on *)
TMP := S_FL_AVL( 1 , 1 , Current_pos1-1 ) ;     (* Re-caculate File 's Head & Tail *)

(* Check if record number reach the end *)
record_cnt1  :=  record_cnt1 + 1 ;  (* current record count plus 1 *)
if   (record_cnt1 >= total_record1)   then
   STEP1  :=  2  ;      (* 2: recond finished *)
   Msg1 :=  'Record is finished ! You may download the record file to your PC now !' ;
   Tstop(T1)  ;
   T1  :=  T#0s   ;
   To_Blink  :=   FALSE ;  (* Set as FALSE not to blink the display value *)
end_if ;

 end_if ;

end_if  ;
```

11.3.8: How to do periodic operation in ISaGRAF controllers ?

The “BLINK” function block can apply to generate a Pulse True periodically. So it can apply in some periodic operations like as below.

(* LD program *)

“Pulse1” is Boolean Internal
Periodic operation every 2 sec



(* ST program *)

```
IF   Pulse1   THEN    (* above LD program will generate a pulse TRUE in “pulse1” variable *)

     (* do operation *)
     (*  ……       *)

END_IF ;
```

The above program has a disadvantage. When the periodic interval time is short, for example – 200ms or smaller, or the controller ‘s PLC scan time is bigger , the operation time will not be precise. For example to do a periodic operation every 50 milli-second. Because 50ms is a shorter interval , it is much closer to the PLC scan time compared with interval time of 250 ms or 2 seconds, the result time will not be precise. To improve this, following codes can be applied.
ST program:

```
IF   INIT   THEN
     INIT  :=  False ;
     T1  :=  T#0s ;
     T1_next  :=  T1  +  T#50ms ;
     Tstart (T1) ;
END_IF ;

IF   T1 >=  T1_next   THEN

     IF       T1  >  T#22h   THEN
        T1  :=  T#0s ;
        T1_next  :=  T#0s  ;
     END_IF  ;
     T1_next := T1_next  +  T#50ms  ;    (* calculate next operation time *)

     (* do operation *)
     (*  ……       *)

END_IF  ;
```

“INIT” is declared as Boolean Internal
And init as TRUE
“T1” and “T1_next” are Timer Internal

Timer will be overflow if it is ticking to T#23h59m59s999ms. So we can reset it to 0 second when it just reach the “22h” or “16h” whatever a bigger time you like.

11.3.9: Demo_72: Connecting I-7018z and I-7188EGD to get 6 channels of 4 to 20 mA input and 4 channles of Thermo-couple temperature input. And then also display the value on PC by VB 6.0 program .

The ISaGRAF demo project name is "Demo_72" . It can run in the I-7188EG, μPAC-7186EG, μPAC-5xx7. If user want to run in I-8xx7, XP-8xx7-CE6, XP-8xx7-Atom-CE6 or WinCon-8xx7, please set the "com_port" parameter of "Bus7000b" in the IO connection window to COM3 and then re-compile the project. (If using WP-8xx7, WP-5xx7, VP-25W7/23W7, set the "com_port" parameter of "Bus7000b" to COM2)

"demo_72.pia" resides at I-8000 CD-ROM:\napdos\isagraf\8000\demo\ or
ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/8000/demo/ or

VB 6.0 project - "Demo_4" resides at
i-8000 CD: \napdos\isagraf\vb_demo\demo_4\ or
http://www.icpdas.com/faq/isagraf.htm FAQ-055

I-7188EG 's COM2: RS-485 can connect I-7000 or I-87K/4/5/8/9 expansion base plus I-87xxx I/O boards in it. One I-7188EG can connect max. 64 pcs. of I-7000 modules (or I-87xxx I/O boards, the total amount of "I-7000 + I-87xxx" is up to 64 pcs.). To use I-8xx7's COM3: RS-485 to connect I-7000 + I-87xxx is the same as I-7188EG, the total amount is also 64 pcs. While max. 255 pcs. for using W-8xx7 's COM3: RS-485 to connect I-7000 + I-87xxx .

The more RS-485 I/O modules connected, the more I/O scan time will be. For example, if setting baud-rate as 9600 bps (Bit Per Second), one RS-485 D/I & D/O module will consume about 20 to 40 milli-second to scan its I/O channels. If connecting RS-485 A/I & A/O module, one will consume about 40 to 60 ms (The I/O scan time of the remote RS-485 I/O module depends on the module's type and function. If there are more than one I/O type in the module, the time consumed will be longer than the above value. For example, the I-7050D is a 7-Ch digital Input plus 8-ch digital output module, it will consume more than 20 to 40 ms). If connecting 20 pcs. of D/I/O modules, the appromate I/O scan time of all channels in these I/O modules will be about 0.4 to 0.8 second. If connecting 20 pcs. of A/I/O modules, the I/O scan time is about 0.8 to1.2 second. To have better (shorter) remote I/O scan time, here recommend not to connect more than 24 pcs. of I/O modules in the I-7188EG/XG and I-8xx7, while 64 pcs. in the WinCon-8xx7.

How to test this demo ?

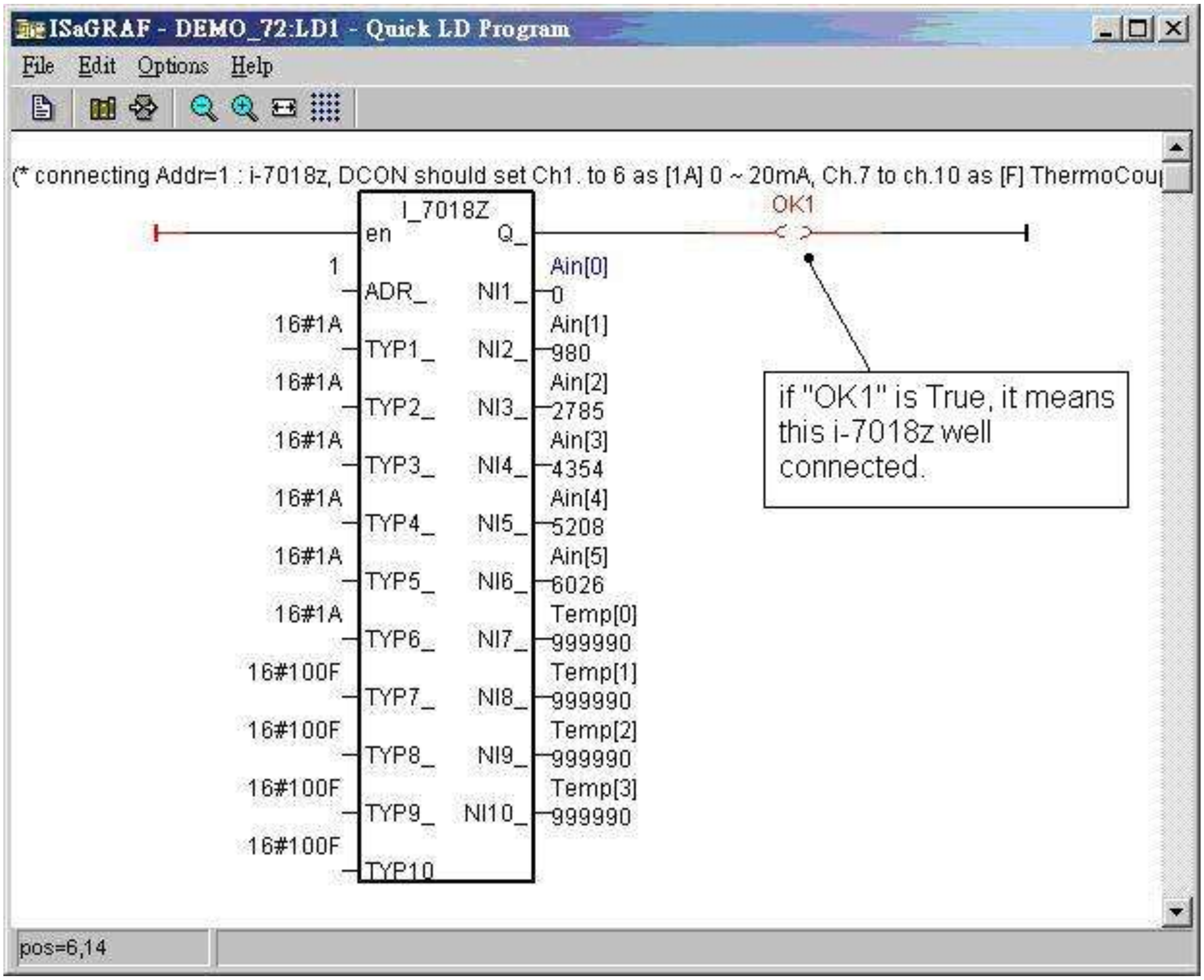
1. To configure I-7018z and I-87018z, please install **DCON utility (Version should be 4.4.3 or later version)** in your PC. The new released DCON utility can be found in the i-8000 CD-ROM or at ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/driver/dcon_utility/ "setup" folder .

2. Please do initial configuration in I-7018z, (please refer to step (1) to (4) in chapter 6.1). Set I-7018z 's Address as 1, baud rate as 9600, Format as "2's complement" , Checksum disable. And also set Ch.**1** to Ch.**6** type as "**[1A] : 0 ~ 20 mA**", while Ch.**7** to Ch.**10** type as "**[0F] : T/C K-Type**" . If initial setting is finished, please switch the "Dip Switch" on the back of I-7018z to "Normal" and recycle its power.

3. Please set the I-7188EG 's IP as 192.168.1.3 (refer to Appendix B), NET-ID as 1. Then power OFF the I-7188EG, connecting its COM2 to the I-7018z. Then power up I-7188EG and I-7018z. (To

connect this I-7188EG well in the local network, PC 's IP should be in the same domain as 192.168.1.x. For example, setting PC 's IP as192.168.1.2 , Mask=255.255.255.0)

4. PC run ISaGRAF to download "demo_72" project to the I-7188EG via ethernet. (If you don't know how to do it, please refer tp section 2.1.5 or 1.3.8) Then open the Ladder program window in the ISaGRAF to check if I-7018z is well connected.



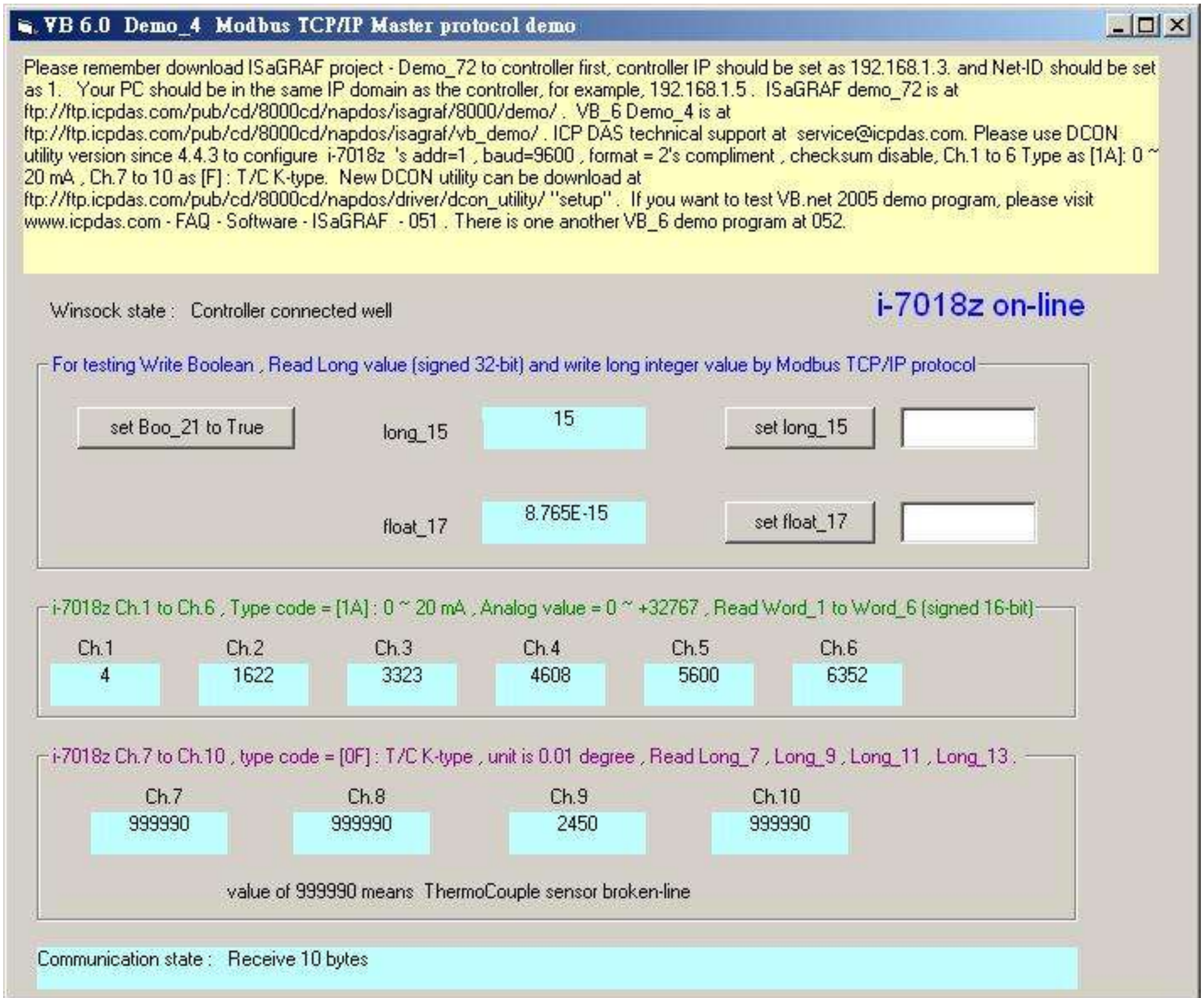
5. Then please run VB 6.0 – "Demo_4.exe" in your PC. It resides at
   i-8000 CD: \napdos\isagraf\vb_demo\demo_4\demo_4.exe or
   http://www.icpdas.com/faq/isagraf.htm > FAQ-055

(As the figure in the next page)

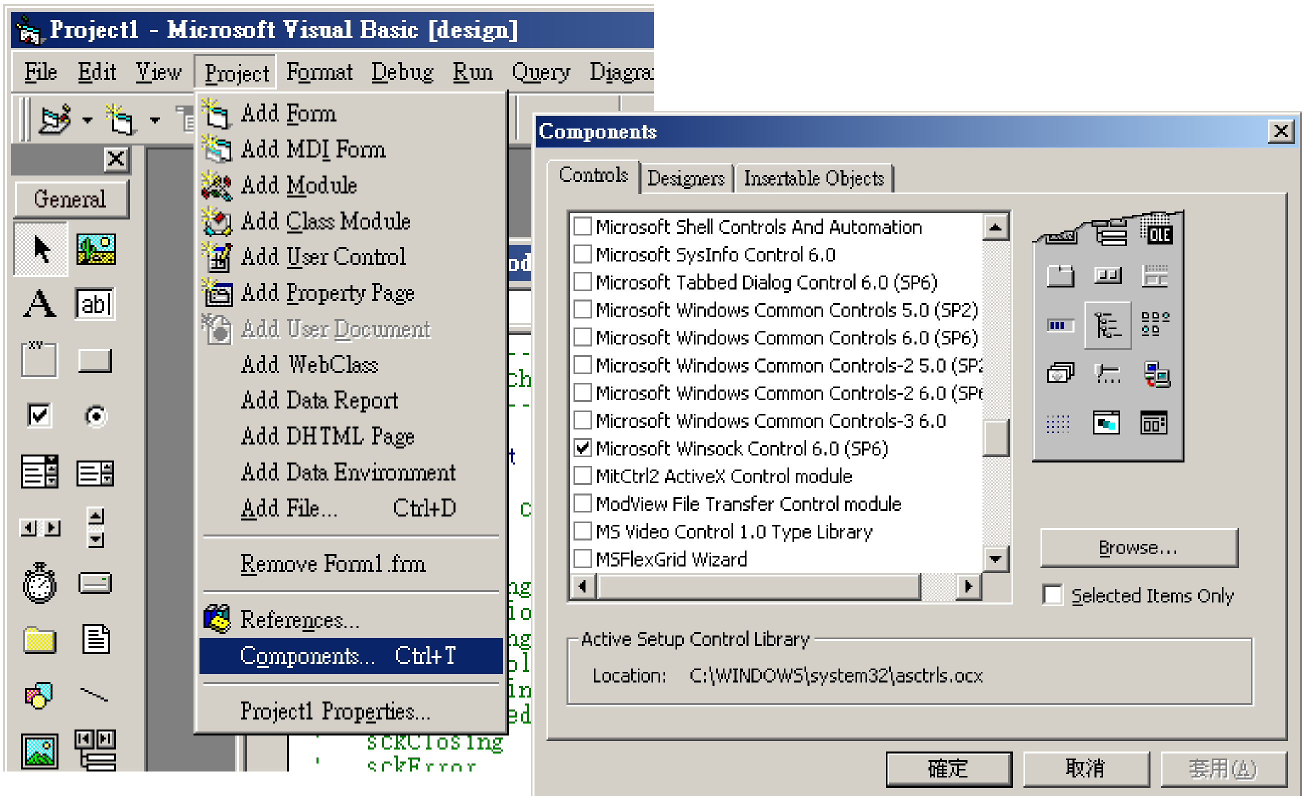
There is one another VB.net 2005 demo project can be study. Please visit
http://www.icpdas.com/faq/isagraf.htm > FAQ-051 or
(www.icpdas.com – FAQ – Software – ISaGRAF – 051)

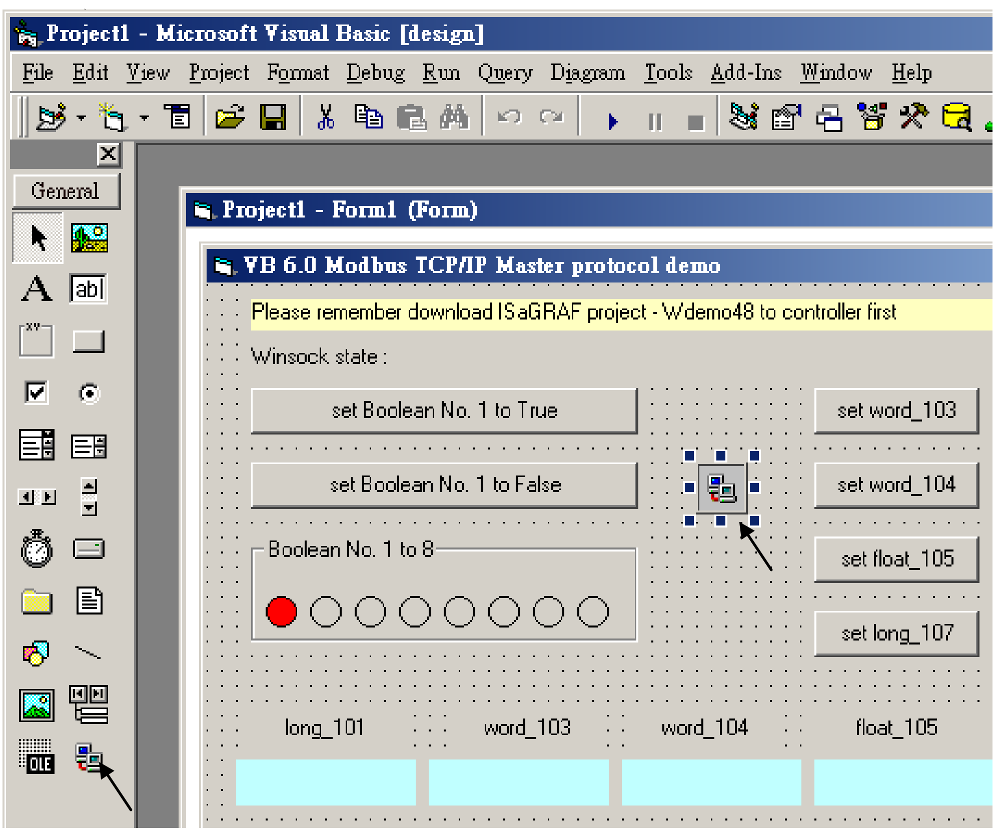
If PC can not link the I-7188EG well, the “Communication state” at the bottom will display the related error message. If the I-7188EG can not connect I-7018z well, there will be a “I-7018z not on-line” message displayed in red color.

You may click on “set Boo_21 to True” button. One click will increase the “long_15” value by 1. You may also enter a value to “set long_15” column, then click on “set long_15”

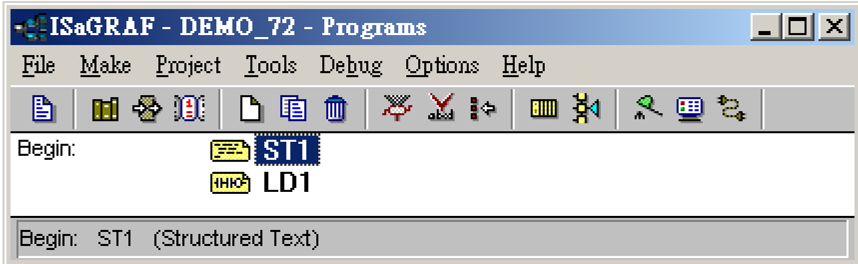


At designing time of the VB 6.0 program, please add "Winsock control" to your VB 6.0 project as below. Then ethernet operation will be possible in the project.





ISaGRAF Project architecture:



Variables :

Name	Type	Attribute	Description
INIT	Boolean	**Internl**	Set initial value as True
OK1	Boolean	**Internl**	Communication state of I-7018z, **addr as 31** (Hex. is 1F)
M1	Boolean	**Internal**	For testing by VB 6.0 , **addr as 21** (Hex. is 15)
TMP	Boolean	**Internal**	Internal using
Ain[0..5]	Integer	**Internal**	Variable array, Dim as 6, **addr as 1** To get the input value of I-7018z ‘s Ch.1 to Ch.6
Temp[0..3]	Integer	**Internal**	Variable array, Dim as 4, **addr as 7** To get the temperature input of I-7018z ‘s Ch.7 to Ch.10
CNT1	Integer	**Internal**	For testing by VB 6.0, **addr as 15** (Hex. is F)
Float_17	Integer	**REAL**	For testing by VB 6.0, **addr as 17** (Hex. is 11) Set initial value as 1.02345

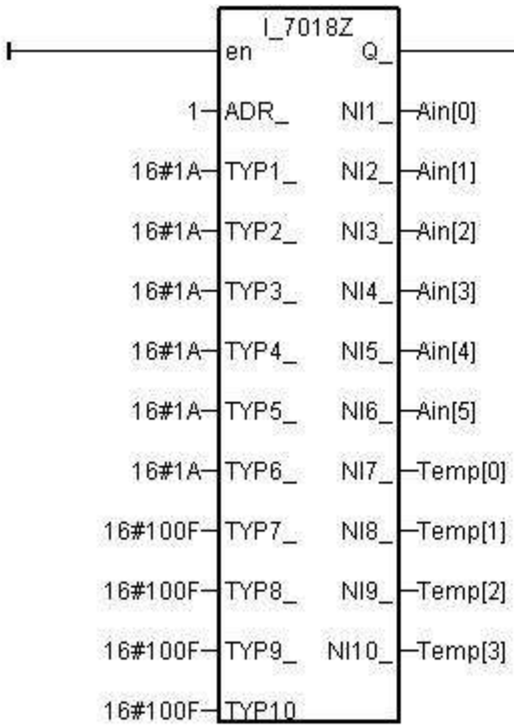
STprogram – ST1

```
-------------------------------------------------------------------------------------------------------
if  INIT  then

  INIT  :=  False  ;

  (* Configure Ain[0..5] ‘s network addr as 1, 2, 3, 4, 5, 6, the initial addr. 1 should be assigned when
doing variable declaration in the ISaGRAF dictionary window *)
  TMP :=  S_MB_ADR( 1 , 6 , 0 )  ; (* the 3rd parameter 0 means setting as continuous addr. *)

(*Configure Temp[0..3] ‘s network addr as 7, 9, 11, 13, the initial addr. 7 should be assigned when doing
variable declaration in the ISaGRAF dictionary window *)
  TMP :=  S_MB_ADR( 7 , 4 , 1 )  ;   (*the 3rd parameter 1 means setting as jummping addr. *)

end_if  ;

if   M1   then
   M1  :=  False  ;
   CNT1  :=  CNT1  +  1  ;  (* if M1 is set as TRUE by VB 6.0 program, increase CNT1 by 1 *)
end_if  ;
-------------------------------------------------------------------------------------------------------
```

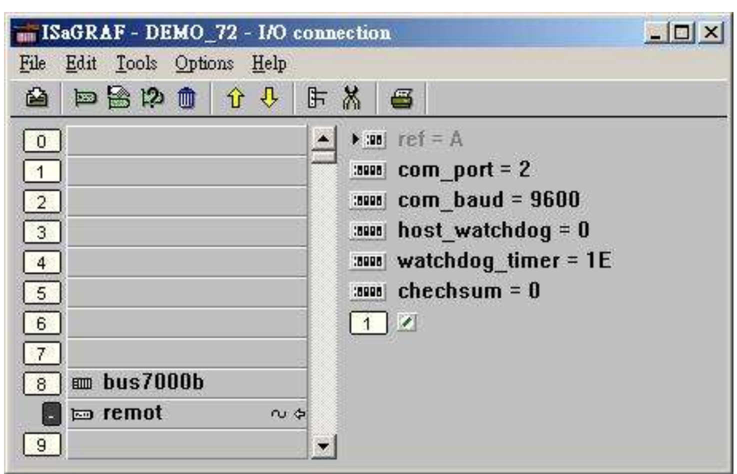
LD program – LD1

The “TYP1_” to “TYP6_” parameter of the I-7018z block should be set as the same type code value in the DCON utility (Here we use [1A] 0 ~ 20 mA in this demo). And “TYP7_” to “TYP10_” set as 16#100F (This demo set [0F] T/C K-Type in the DCON utility) . Because we want to convert the temperature value to Celsius degree, so we use 16#100F here (unit is 0.01 degree). (If applying as Degree Fahrenheit, please set as 16#200F). If any converted value of the Temp[0] to Temp[3] returns 999990, it means the related channel’s temperature input sensor is break.

If the I-7018z is connected well, OK1 will be True.



IO connection:



11.3.10: Whmi_13: Recording I-8017H 's Ch.1 to Ch.4 voltage input in a user allocated RAM memory in the WinCon-8xx7 . The sampling time is one record every 0.01 second. The record period is 1 to 10 minutes. Then PC can download this record and display it as a trend curve diagram by M.S. Excel.

Note: This demo also apply to WP-8xx7, WP-5xx7, XP-8xx7-CE6, XP-8xx7-Atom-CE6 and VP-25W7/VP-23W7, the difference is the required contents and files (both are similar, but different with WinCon CD) are on respective CD. Moreover, the WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6 and VP-25W7/VP-23W7 do not have the path \CompacFlash\Temp\HTTP\WebHMI\, the WP-8xx7, WP-5xx7 and VP-25W7/VP-23W7 use the path \Micro_SD\Temp\HTTP\WebHMI\ and the XP-8xx7-CE6, XP-8xx7-Atom-CE6 use the path \System_Disk\Temp\HTTP\WebHMI\. Please refer to the following website to get more details:

WP-8xx7 : "setup_web_hmi_demo.pdf"
ftp://ftp.icpdas.com/pub/cd/winpac-8xx7/napdos/isagraf/wp-8xx7/wp_webhmi_demo

VP-25W7 / VP-23W7 : "setup_web_hmi_demo.pdf"
ftp://ftp.icpdas.com/pub/cd/vp-25w7-23w7/napdos/isagraf/vp-25w7-23w7/vp-webhmi-demo/ 內

XP-8xx7-CE6 : "xpce6_setup_web_hmi_demo.pdf"
ftp://ftp.icpdas.com/pub/cd/xp-8xx7-ce6/napdos/isagraf/xp-8xx7-ce6/xpce6-webhmi-demo/ 內

The "Whmi_13.pia" can run in WinCon-8xx7/8xx6 with driver version of 3.36 or later version.

> New drive: http://www.icpdas.com/products/PAC/i-8000/isagraf-link.htm
> "whmi_13.pia" resides at W-8xx7 CD-ROM:\napdos\isagraf\wincon\demo\ or
> ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/demo/

VB6 - "Demo_5" code at

> W-8xx7 CD-ROM:\napdos\isagraf\wincon\vb6_demo_pc\ or
> ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/vb6_demo_pc/

If using Web HMI in this demo, the Web HMI codes resides at below location. (Please refer to Chapter 3, 4 and 5 in the "WinCon ISaGRAF Getting Started manual" or
W-8xx7 CD-ROM:\napdos\isagraf\wincon\english_manu\ "getting_started_w8337.pdf")

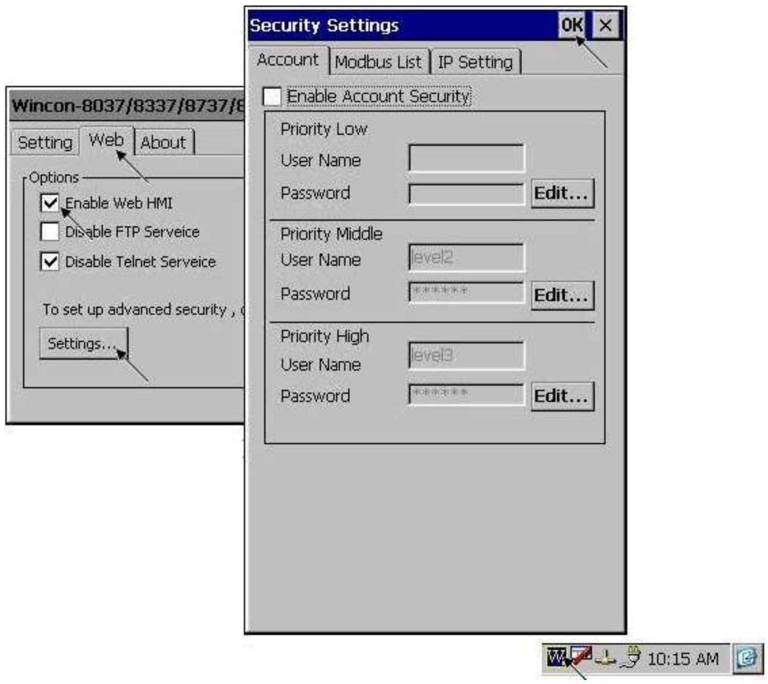
> W-8xx7 CD-ROM:\napdos\isagraf\wincon\WebHMI_Demo\ whmi13 or
> ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/webhmi_demo/

If new c-function of Msg_F , Msg_N , ARY_F_R and AFY_F_W doesn't exist in ISaGRAF in your PC, please visit http://www.icpdas.com/products/PAC/i-8000/isagraf.htm to download the "ICP DAS utilities For ISaGRAF" . Then run "setup.exe" inside it to re-install all new ISaGRAF c-function & I/O boards definition to your ISaGRAF workbench.

How to test this demo ?

The following steps is only for using Web HMI as Human-Machine-Interface. If you are using VB 6.0 – "Demo_5" as HMI , please run it in your PC and only do procedure listed in step 1 (not necessary to enable "Web HMI") , step 3 and step 6 .

1. Please plug one I-8024 in W-8xx7 's Slot 2, one I-8017H in Slot 3. Then connect Ch1. to Ch.4 voltage output of I-8024 to Ch1. to Ch.4 of I-8017H. Then power up WinCon, Check "Enable Web HMI" option as below. For demo purpose, please don't check "Enable Account Security"



2. Copy all files of Web HMI 's Demo_13 to WinCon 's \CompactFlash\Temp\HTTP\WebHMI\ folder by ftp utility (For example, run ftp://10.0.0.103 in Internet Explorer)

   Web HMI codes resides at
   W-8xx7 CD-ROM:\napdos\isagraf\wincon\WebHMI_Demo\ "whmi_13" folder or
   ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/webhmi_demo/

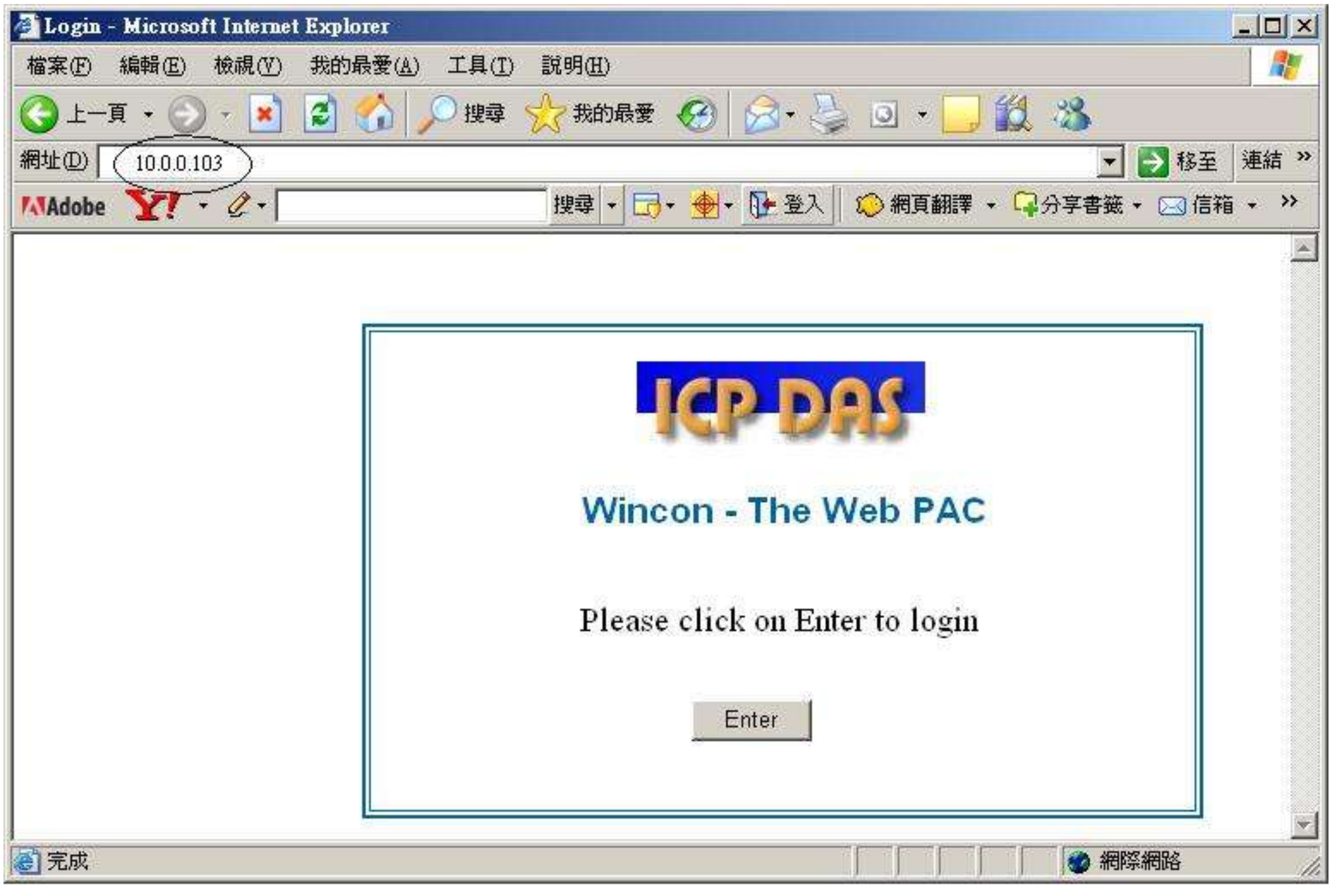
   Ther are 7 Files plus 2 folder should be copied to WinCon's \CompactFlash\Temp\HTTP\WebHMI\

   Main.htm , menu.htm , index.htm , login.htm , main.dll , login.dll , whmi_filter.dll
   "img" & "msg" folder

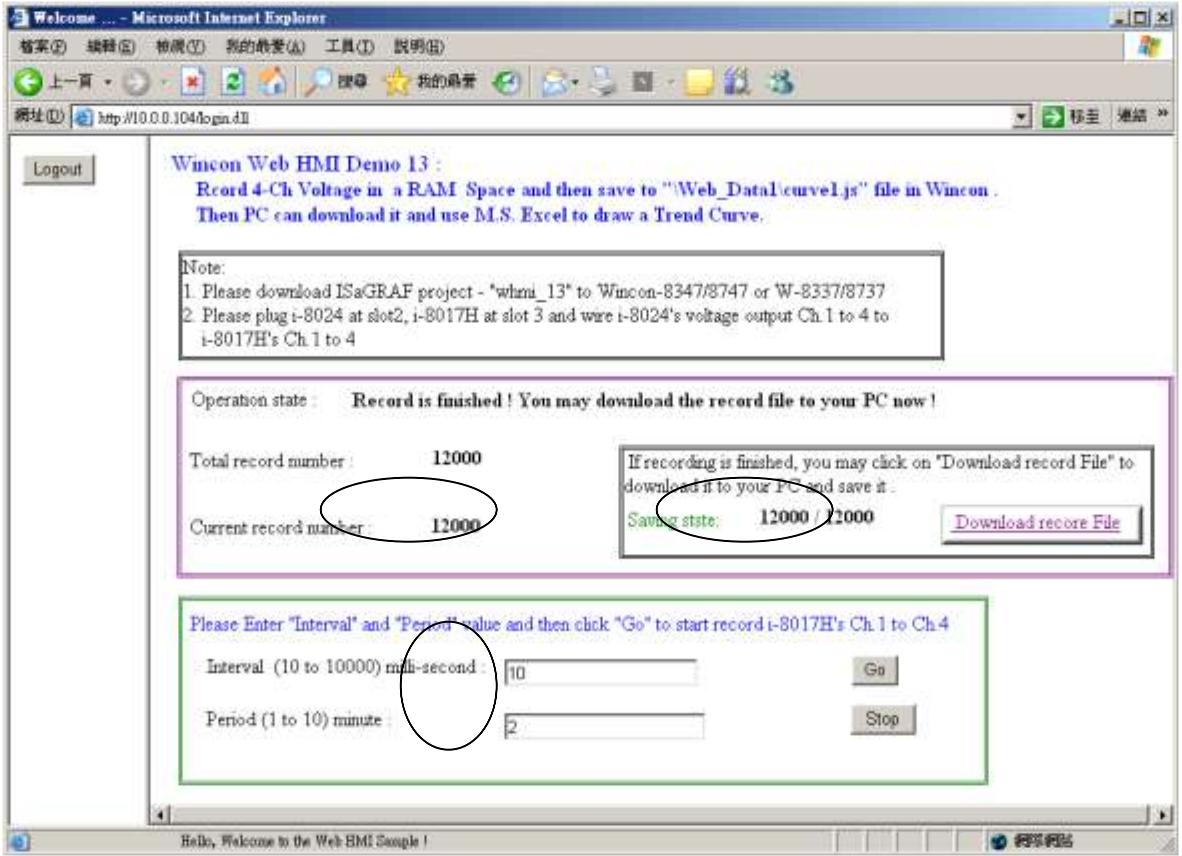
3. Download ISaGRAF project "whmi_13" to W-8xx7. (If using Web HMI as HMI, please finish procedure listed in step 2 first, then do step 3)

   The "Whmi_13.pia" can only run in WinCon-8xx7/8xx6. It resides at
   W-8xx7 CD-ROM:\napdos\isagraf\wincon\demo\ or
   ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/demo/

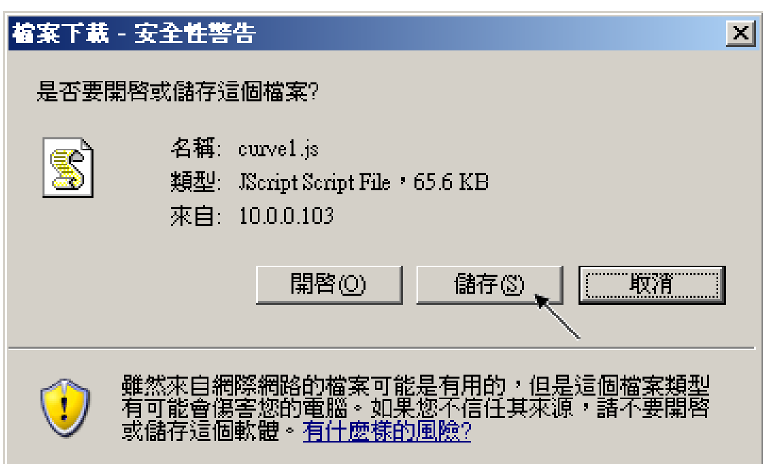
4. PC run Internet Explorer (I.E. version shoulde be 5.0 or later version). Enter W-8xx7 IP. If connecting well, click on “Enter”



5. Then please enter proper “Interval” value. Unit is 0.001 second (1 milli-second). For example, if enter 10, it means to store one record every 10 ms. The ”Period” is the time period to record. Unit is minute. Then please click on “Go” to start recording. W-8xx7 will then output different voltage in I-8024 Ch.1 to Ch.4 . If user has finished procedure listed in step 1 – “connect Ch1. to Ch.4 voltage output to Ch1. to Ch.4 of I-8017H”, the I-8017H Ch1. to Ch.4 ‘s voltage input will also change during this period. And they will be recorded. (As figure below)

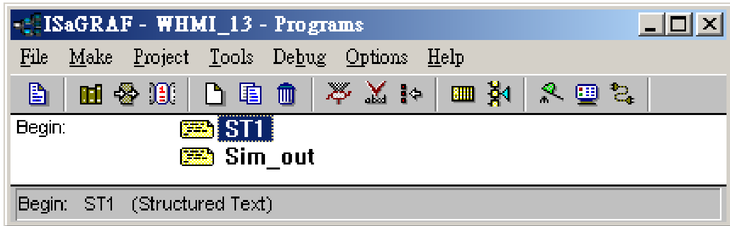


During the recording period, the "Current record number" value will count up. If it reaches the value of "Total record number" , it means recording is finished. Then the ISaGRAF program will store these records to a RAM file automatically. You can see the progress in "Saving state". If all done, please click on "Download record File" to download this record file to your PC.



6. Then please open this record file - "trend1.js" on M.S. Excel. Please refer to section 11.3.6 – step (6).

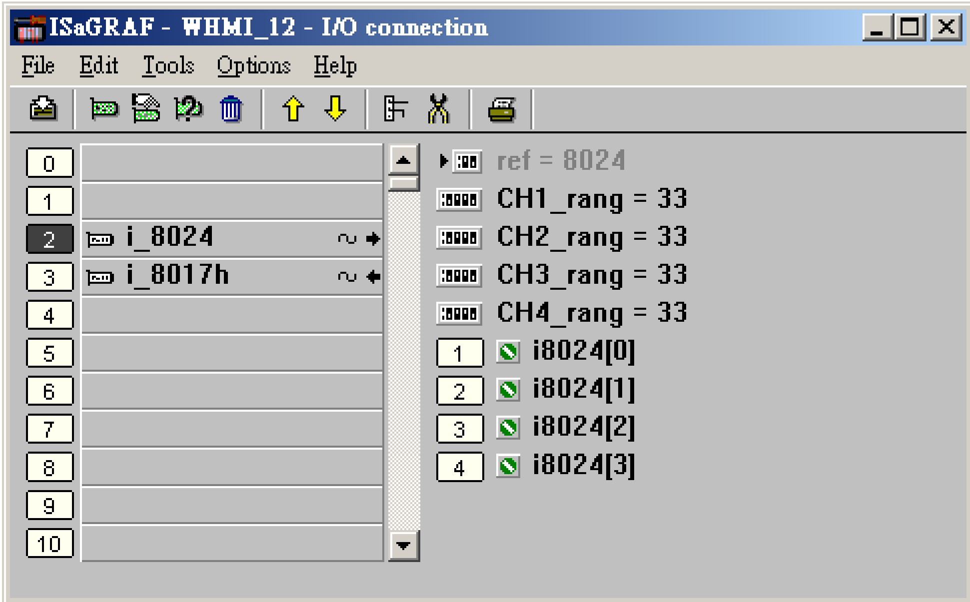
ISaGRAF project architecture:

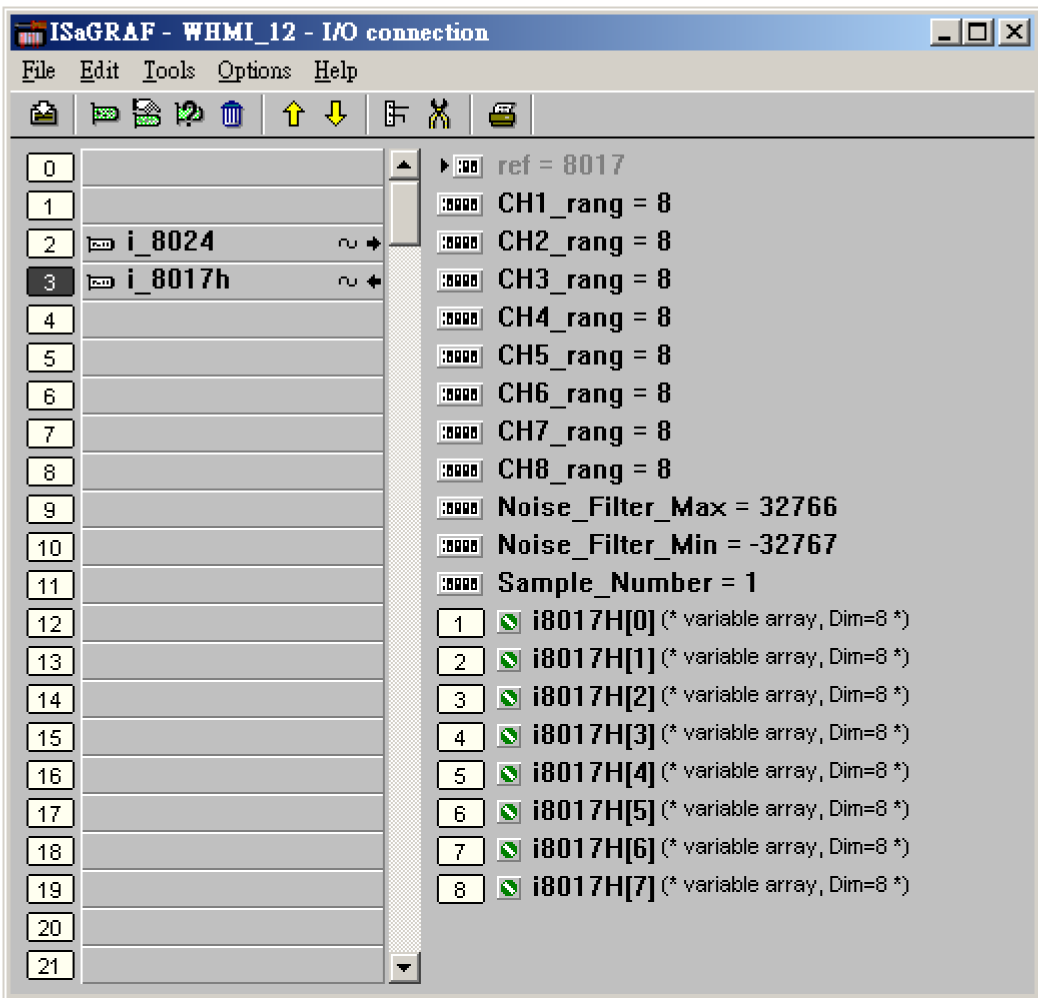


Variables :

Name	Type	Attribute	Description
Go1	Boolean	**Internl**	Set as True to start, **addr defined as 21** (Hex. is 15)
Stop1	Boolean	**Internal**	Set as True to stop, **addr defined as 22** (Hex. is 16)
TMP	Boolean	**Internal**	Internal use
INIT	Boolean	**Internal**	Init as True
Save_file1	Boolean	**Internal**	The ISaGRAF program will set this value to True to store records to a RAM Disk File
MUM_CH	Integer	**Constant**	How many chanels in I-8017H to record ? We use 4 channels in this demo (Ch.1 to 4)
File1	Integer	**Internal**	File ID
STEP1	Integer	**Internal**	Recording state. 0:No action , 1:recording , 2:finished
Period1	Integer	**Internal**	How long to record ? unit is minute, **addr as 3**
Interval1	Integer	**Internal**	How long to save a record ? unit is ms, **addr as 1**
Total_record1	Integer	**Internal**	How many records in this recording action ? This value is calculated by the ISaGRAF program automatically. **addr declared as 5**
Record_cnt1	Integer	**Internal**	Current finished record count. **addr declared as 7**
ii & ii2	Integer	**Internal**	To use in “for” loops
i8017H[0..7]	Integer	**Input**	Variable array, Dim as 8. link to I-8017H ‘s Ch1 to Ch. 8
Volt1[0..7]	REAL	**Internal**	Variable array, Dim declared as 8. The voltage value converted from “i8017H[0..7]”
i8024[0..3]	Integer	**Output**	Variable array, Dim declared as 4. link to I-8024 ‘s Ch1 to Ch. 4
Save_cnt1	Integer	**Internal**	Current saving record amount in the RAM disk File, **addr declared as 9**
TMP_v	Integer	**Internal**	Internal use
T1	Timer	**Internal**	For counting time
T1_next	Timer	**Internal**	The time to get and save next record
T1_Interval	Timer	**Internal**	The interval time between two record
File_name1	Message	**Internal**	File name, Len is 64, init as \Web_Data1\curve1.js Web HMI support only RAM Disk File in \Web_Data1\ , If the file is in CompactFlash File, Web HMI support only in \CompactFlash\Temp\HTTP\Data\ (Please refer to Chapter 11.2 - Whmi_08 demo)
Msg1	Message	**Internal**	Operation state message, Len is 255, init as “No Action now”, **addr as 41** (Hex. is 29)
Str1	Message	**Intenal**	Len is 255, internal use

IO connection:





ST program – Sim_out

```
------------------------------------------------------------------------------------------------------------------
(* Output I-8024 's Ch1  to Ch4 as different voltage curve *)

(*   2 * Pi * T1 / 60000  =  T1 * 1.047197E-4    *)
(*   2 * Pi * T1 / 120000 = T1 *  5.235985E-5 *)

i8024[0]  :=  ANA(  sin( REAL(T1) *  1.047197E-4 ) * 3276.8  )  ;
i8024[1]  :=  ANA(  cos( REAL(T1) *  5.235985E-5  ) * 3276.8  ) ;
i8024[2]  :=  ANA(  sin( REAL(T1) *  1.047197E-4 ) * 6553.6  ) ;
i8024[3]  :=  ANA(  cos( REAL(T1) *  5.235985E-5  ) * 6553.6  ) ;
------------------------------------------------------------------------------------------------------------------
```

ST program – ST1

```
------------------------------------------------------------------------------------------------------------------

(* W-8xx7 can have max. speed of 100Hz to record data ( minimum sample interval is 10 ms) *)
(* This example assume max. 8-Ch. , so 1 second will record 100 x 8 REAL value *)
(*   1 minute will record  100 x 8 x 60 = 48,000 REAL value *)
(* If period is set as 10 minute, we need 48,000 x 10 = 480,000 REAL value memory = 480,000 x 4 =
1,920,000 bytes *)

if  INIT  then
  INIT := False ;

  (* Allocate 500,000 integer (or 32-bit REAL) space to store records up to 10 minutes. total 500,000 x 4
= 2,000,000 bytes  , W-8xx7 support only No.1 Arcreate( ) up to 3,000,000 integer space, that is
12,000,000 bytes . The first parameter in ARcreate( ) should be 1, it doesn't support 1st parameter as 0, 2,
3, ... , 15  *)
  (* Arcreate( ) can be called only once in the ISaGRAF program *)
  TMP_v := ARcreate(1, 500000) ;
  if  TMP_v <> 1  then
    Msg1 := 'Parameter error or can not allocate memory by ARcreate( ) function!' ;
  end_if  ;

  TMP := PLC_mode(-1) ;     (* Set W-8xx7 ISaGRAF driver running at fatest mode *)
end_if ;

(* If stop command is gived *)
if  Stop1  then
  Stop1 := False ;
  STEP1 := 0 ;  (*  0: no action *)
  TStop(T1)  ;  (* stop T1 *)
  T1 := T#0s ;
  Msg1 := 'User stop recording !' ;
  save_cnt1 := 0 ;
end_if  ;
```

```
(* If start command is gived *)
if  Go1  then

  Go1 := False ;

  (* STEP1 :  0: no action , 1: recording , 2: recond finished *)
  if  STEP1=1  then

     (* It is still recording now *)
     Msg1 := 'It is still recording now ... Please wait' ;

  else

     (* Check interval valid or not *)
     (* we assume 10 to 10000 ms is valid in this example *)
     (* If your average PLC scan time is larger, for example, near 10 ms,
         Please use Interval larger than 10 ms. Or the record time won't be correct *)
     if  (Interval1 < 10)  or  (Interval1 > 10000)  then
       Msg1 := 'Wrong Interval value, it should be in 10 to 10000 milli-second !' ;

     (* Check period valid or not *)
     (* we assume 1 to 10 minute is valid in this example *)
     elsif ( Period1 < 1)  or  (Period1 > 10)  then
       Msg1 := 'Wrong Period value, it should be in 1 to 10 minute !' ;

     else

       (* parameter is correct, start recording *)

       total_record1 := (Period1 * 60000) / Interval1 ;  (* calculate total record number *)
       record_cnt1 := 0 ;  (* reset current record count as 0 *)
       STEP1 := 1 ;  (* set step as 1:recording *)
       Msg1 := 'Recording now ... Please wait'  ;

       (* start ticking T1 from 0 second *)
       T1 := T#0s ;
       T1_Interval := TMR(Interval1) ;
       T1_next := T1 + T1_Interval ;
       TStart(T1) ;   (* ticking now *)
       save_cnt1 := 0 ;

     end_if ;

  end_if ;

end_if ;
```

```
(* in reconrding state *)
if  STEP1 = 1  then

  (* store one record *)
  if  T1 >= T1_next  then

    (* Re-calculate next T1 *)
    T1_next := T1_next + T1_Interval ;

    (* T1 will be overflow after T#23h59m59s999ms, so reset it at T#20h *)
    if   T1 >= T#20h  then
      T1 := T#0s ;
      T1_next := T1 + T1_Interval ;
    end_if ;

    (* record data *)
    for  ii  :=  0  to  NUM_CH-1  do
      Volt1[ii] := Real(i8017H[ii]) * 0.000305176 ;    (* convert to voltage *)

(* using Real_int( ) to map REAL value to become integer value & then store it by ARwrite( ) *)
      TMP_v := ARwrite( 1 ,  NUM_CH * record_cnt1 + ii , Real_int(Volt1[ii]) ) ;

      (* check if ARwrite( ) correct *)
      if TMP_v <> 1  then
        Msg1 := 'Can not operate ARwrite( ) !' ;
        STEP1 := 0 ;    (*  0: no action *)
        TStop(T1)  ;    (* stop T1 *)
        T1 := T#0s ;
      end_if ;

    end_for ;

    (* Check if record number reach the end *)
    record_cnt1 := record_cnt1+1 ;      (* current record count plus 1 *)
    if  (record_cnt1 >= total_record1)   then

 (* record is finished, prepare to save records to a RAM disk file in serval separate PLC scans *)
      STEP1 := 0 ;   (* set step as 0 at the beginning of saving *)
      Tstop(T1) ;
      T1 := T#0s ;

      (* Create a new file *)
      File1 := F_creat(File_name1) ;
      if   File1 = 0  then
        (* Can not create file *)
        Msg1 := 'Create File ' + 'File_nam1 Error !!!' ;
      else
```

```
    (* Because saving lots of data to file take lots of PLC scan time , we are not going to
       save all data in a single PLC scan. We will save it in serval separate PLC scans *)

    Msg1 := ' Please wait ...   Saving data to file : ' + File_name1 + ' ...' ;
    save_file1 := True ;  (* set as True to start saving RAM disk file *)
    save_cnt1 := 0 ;   (* from 0 to total_record1-1 *)
   end_if ;

  end_if ;

 end_if ;

end_if ;

(* Because saving lots of data to file take lots of PLC scan time , we are not going to
   save all data in a single PLC scan. We will save it in serval separate PLC scans *)

(* save records to a RAM disk file in serval separate PLC scans *)
if  save_file1  then

  for  ii2 := 0  to 50  do      (* we limit one PLC scan can save max. 50 records *)

    if  save_cnt1 < total_record1  then

      str1 := '' ;     (* init str1 as empty string *)
      for  ii  :=  0  to  NUM_CH - 1  do

        (* delimiter is <TAB> character *)
        str1 := str1 + Rea_Str2( Int_real(ARread(1, NUM_CH * save_cnt1 + ii)) , 3  ) + '$09' ;

      end_for ;
      str1 := str1 + '$0D$0A'  ;  (* add <CR> <LF> at the end of each row *)
      TMP := F_writ_s(File1 , str1) ;
      save_cnt1 := save_cnt1 + 1 ;

    else

      (* saving is finished *)
      save_file1 := False ;
      TMP := F_close(File1) ;      (* Close file *)
      STEP1 := 2 ;                 (* 2: recond finished *)
      Msg1 := 'Record is finished ! You may download the record file to your PC now !' ;

    end_if ;
  end_for ;
end_if ;
```

11.4: Frequently Asked Questions

FAQ (ISaGRAF Ver.3 FAQ: Questions/Descriptions/Demo programs)
http://www.icpdas.com/faq/isagraf.htm
www.icpdass.com > FAQ > Software > ISaGRAF Ver.3 (English)

FAQ Table:

No.	English ISaGRAF Ver.3 FAQ
1	Q: How to get counter value built in I-7000 & I-87xxx remote I/O modules?
2	Q: How to search I/O boards and declare variables automatically for I-8xx7 controllers?
3	Q: How to build a HMI screen by using ISaGRAF?
4	Q: Can I create my own functions inside ISaGRAF?
5	Q: Can I use more than 32 I/O in my ISaGRAF project if I don't have ISaGRAF-256 or ISaGRAF-L?
6	Q: Can I use ISaGRAF controller (I-8417/8817/8437/8837, I-7188EG/XG) as a Modbus Master controller to gather data from other Modbus devices?
7	Q: Can I write my own protocol or third-party protocol to apply on ISaGRAF controllers?
8	Q: What is the limitation of program size of I-8417/8817/8437/8837, I-7188EG & I-7188XG?
9	Q: Can not fine I/O boards in the ISaGRAF I/O connection window?
10	Q: I Want to email my ISaGRAF program to someone. How can I archive one ISaGRAF project to a single file?
11	Q: How can I implement motion control in I-8417/8817/8437/8837?
12	Q: My HMI software wants to access to float values and long word values inside the I-8417/8817/8437/8837, 7188EG & 7188XG. How?
13	Q: PWM: Can I generate D/O square pulse up to 500Hz with I-8417/8817/8437/8837, 7188EG & 7188XG controllers? How?
14	Q: Can I use 8K Parallel D/I board to get counter Input up to 500Hz? How ?
15	Q: How to output something at a time interval? For ex. Turn ON at 09:00~18:00 on Monday to Saturday , while 13:00~20:00 on Sunday.
16	Q: How to determine a D/I if it has bouncing problem?
17	Q: How to trigger something at some seconds later when one event happens?
18	Q: Does the ISaGRAF-256 software have I/O Tag limitation? Why not using "ISaGRAF-L" Large version?
19	Q: Why my I-8417/8817/8437/8837 or I-7188EG/XG stop running?
20	Q: How to search a variable name in an ISaGRAF project?
21	Q: When closing my ISaGRAF window, it holds for long time. Why?
22	Q: How to use Proface HMI (Touch panel) to link to I-7188EG/XG, I-8xx7 and WinCon-8x37?
23	Q: How to reduce ISaGRAF code size? How to directly Read / Write ISaGRAF variables by using Network address?

No.	English ISaGRAF Ver.3 FAQ
24	Q: How to scale Analog Input and Output of 4 to 20 mA to my engineering format? How to scale Analog Input and Output of 0 to 10 V to my engineering format?
25	Q: How to detect controller Fault?
26	Q: New ISaGRAF retained variable is better than old one.
27	Q: How to link to Modbus ASCII Slave device?
28	Q: How to use multi-port Modbus Master in the WinCon-8037/8337/8737 & WinCon-8036/8336/8736?
29	Q: How to send/receive message from ISaGRAF PAC to remote PCs or Controllers via Ethernet UDP communication?
30	Q: Setting special “range” parameter of temperature input board to get clear “Degree Celsius” or “Degree Fahrenheit” input value. For ex, “1535” means 15.35 degree.
31	Q: Setting a special “ADR_” parameter of remote I-7000 & I-87K temperature input module to get clear “Degree Celsius” or “Degree Fahrenheit” input value. For ex, “8754” means 87.54 degree.
32	Q: How to access to ISaGRAF variables as array? (A demo program of sending string to COM2 or COM3 when alarm 1 to 8 happens)
33	Q: Setting up more Modbus RTU Slave ports in WinCon ISaGRAF PACs.
34	Q: Compiling error result in different ISaGRAF version?
35	Q: Slow down ISaGRAF driver speed to work better with InduSoft software in W-8036/8336/8736 & W-8046/8346/8746?
36	Q: Redundancy Solution in WinCon-8xx7.
37	Q: I-7188EG/XG support remotely downloads via Modem Link.
38	Q: Setting I-7188EG/XG's COM3 as Modbus RTU Slave port.
39	Q: ISaGRAF version 3.4 & 3.5 now supporting "Variable Array" !!!
40	Q: Setting I-8437/I-8837/I-8437-80/I-8837-80's COM3 as Modbus RTU Slave port.
41	Q: How to connect PC / HMI to a Redundancy system with a single IP address?
42	Q: How to use WinCon connecting to Ethernet I/O? The I/O scan rate is about 30 to 40 msec for 3000 to 6000 I/O channels.
43	Q: How to setup WinCon-8xx7 as TCP/IP Client to communicate to PC or other TCP/IP Server device? Or WinCon automatically report data to PC via TCP/IP?
44	Q: WinCon-8xx7/8xx6 automatically report data to PC/InduSoft or PC/HMI?
45	Q: ISaGRAF controllers display message to EKAN Modview LED.
46	Q: How to Write 16-bits to Modbus RTU devices by Mobus function call No. 6?
47	Q: How to Read or Write Floating Point value to Modbus RTU Slave device?
48	Q: How to use WinCon-8xx7 / 8xx6 to control FRnet I/O?
49	Q: Setting a special “CODE_” parameter of “MBUS_R” & ”MBUS_R1” to get a clear “Degree Celsius” or “Degree Fahrenheit” input value of M-7000 temperature module. For ex, “3012” means 30.12 degree.
50	Q: How to connect an ISaGRAF controller to M-7000 Remote I/O?
51	Q: VB.net 2005 Demo program using Modbus TCP/IP protocol to control ISaGRAF PACs

No.	English ISaGRAF Ver.3 FAQ
52	Q: VB 6.0 Demo program using Modbus TCP/IP protocol to control ISaGRAF PACs.
53	Q: Performance Comparison Table of ISaGRAF PACs.
54	Q: iPAC-8xx7 and µPAC-7186EG support Data Logger function.
55	Q: How to connect I-7018z to get 6 channels of 4 to 20 mA Input and 4 channles of Thermo-couple temperature Input? And also display the value on PC by VB 6.0 program?
56	Q: How to do periodic operation in ISaGRAF PACs?
57	Q: How to record I-8017H's Ch.1 to Ch.4 voltage Input in a user allocated RAM memory in the WinCon-8xx7? The sampling time is one record every 0.01 second. The record period is 1 to 10 minutes. Then PC can download this record and display it as a trend curve diagram by M.S. Excel.
58	Q: How to record I-8017H's Ch.1 to Ch.4 voltage input in S256 / 512 in I-8437-80 or I-8837-80? The sampling time is one record every 0.05 second. The record period is 1 to 10 minutes. Then PC can download this record and display it as a trend curve diagram by M.S. Excel.
59	Q: Some skill to operate RS-232/422/485 serial COM Port by COM functions
60	Q: How to read / write file data in WinCon?
61	Q: How to connect RS-485 Remote I-7000 and I-87K I/O modules in I-8xx7, I-7188EG/XG and WinCon-8xx7 PAC? How to program RS-485 remote I-7017RC, I-87017RC and I-7018Z?
62	Q: How to setup a redundant system with Ethernet I/O?
63	Q: Why my RS-485 remote I-7000 and I-87K Output module's host watchdog function doesn't work to reset its output channels to safe output value while the RS-485 communication cable is broken?
65	Q: ICP DAS release Stable and Cost-effective Data Acquisition Auto-Report System. (VC++ 6.0, VB 6.0 and ISaGRAF demo program are available)
66	Q: How to process the Integer or Real value coming from the RS-232 / RS-485 device? Like the device of Bar-Code reader or RS-232 weight meter.
67	Q: How to send email with one attached file by WinCon-8xx7 or iPAC-8447 / 8847 or µPAC-7186EG?
68	Q: Why the W-8xx7 or I-8xx7 or I-7188EG/XG always reset? How to fix it?
69	Q: Why my PC can not run "ftp" to connect W-8347 or W-8747?
70	Q: How to do Time Synchronization and record state of many ISaGRAF PACs?
71	Q: Application: Record 10-Ch. temperature value into a file in W-8xx7 every minute. When 24 hour recording is finished, send this record file by email every day.
72	Q: Application sample: Record Voltage / Current input by W-8xx7 every 20 ms for 1 to 10 minutes. Then send this record file by email.
73	Q: Why does the I-7017 or I-87017's Current Input reading value become double or incorrect?
74	Q: How to use ISaGRAF new Retain Variable? What is its advantage?
75	Q: Why my ISaGRAF project can not connect Modbus Slave device correctly?
77	Q: Application sample: Record Voltage / Current input by µPAC-7186EG every second for

No.	English ISaGRAF Ver.3 FAQ
	1 to 10 minutes. Then send this record file by email.
80	Q: Application: Record 10-Ch. temperature value into a file in µPAC-7186EG every minute. When 24 hour recording is finished, send this record file by email every day.
81	Q: How to measure +/-150VDC in ISaGRAF controllers plus the I-87017W-A5 I/O card?
82	Q: An easy way to program the fast FRnet remote I/O modules.
83	Q: How to set I-8x37, I-8x37-80, I-7188EG and µPAC-7186EG's TCP recycling time?
84	Q: Application: A Cost Effective and Hot-Swap Redundancy System by µPAC-7186EG or I-8437-80 plus RU-87P4/8.
86	Q: The WinCon-8347 / 8747 , µPAC-7186EG and iP-8447 / 8847 connecting one or several I-7530 to link many CAN or CANopen devices and sensors.
87	Q: What does it mean and how to fix it when the 7-segment LED shows error messages of Err00, Err02, Err03, Err90 or E.0001 after booting the PAC?
88	Q: Function Modifications: The W-8347/8747, µPAC-7186EG, I-8x37-80, I-8xx7 and I-7188EG/XG with S256/512 and X607/608 no longer support old retain method, please change to use the better new retain method to retain variables.
089	Q: Why my µPAC-7186EG unable to renew the driver and ISaGRAF application?
090	Q: How to use I-7017Z module in ISaGRAF PAC?
091	Q: How to use ISaGRAF PAC plus I-87089-the VW sensor Master card to measure the Vibration Wire frequency to calculate the stress of constructions?
092	Q: Setting µPAC-7186EG's and I-7188EG/XG's COM3 or COM2 as Modbus RTU Slave port.
093	Q: New Hot-Swap and Redundant solution for the WinCon-8347 / 8747.
094	Q: How to update the WinCon-8347/8747's OS?
095	Q: The WinCon-8xx7 supports Max. 32 Modbus TCP/IP connections since Its Driver version 4.03.
096	Q: Release two C-Function-Blocks to read max. 24 Words or 384 Bits from Modbus RTU / ASCII devices.
097	Q: How to modify the IP, NET-ID and Modbus RTU Slave port setting of the W-8347 / 8747 by an USB pen drive (without Mouse and VGA)?
098	Q: Application: Link Serial COM Port to the Modbus RTU device by COM functions .
099	Q: How to get an average value of a Real or Integer variable which is samlped every fixed interval (or sampled in every PLC scan) ?
100	Q: How to use I-8084W (4 / 8 – Ch. Counter or 8-Ch. frequency) ?
101	Q: How to read max. 120 Words or max. 60 Long-Integers or max. 60 Real value from Modbus RTU / ASCII devices by using MBUS_XR or MBUS_XR1 function block (for WP-8xx7 / 8xx6 and VP-25W7/23W7/25W6/23W6 and WinCon-8xx7 / 8xx6 only) ?
102	Q: Why PC can not connect the WP-8xx7 or VP-25W7/23W7 's FTP server ?
103	Q: Using RS-232 Or USB Touch Monitor With WinPAC.
104	Q: Why my PC running ISaGRAF can not connect the ISaGRAF PAC correctly ?
105	Q: Program The 8-Channel PWM Output Board : I-8088W In WP-8xx7, VP-25W7/23W7 And iP-8xx7 PAC.

No.	English ISaGRAF Ver.3 FAQ
106	Q: How to display the frequency trend curve by running ISaGRAF and C# .net 2008 program in the WinPAC-8xx7 plus I-8084W?
107	Q: How to do auto-time-synchronization and measure the local Longitude and Latitude by using the I-87211W GPS I/O module in ISaGRAF PAC ?
108	Q: How to display the temperature trend curve by running ISaGRAF and C# .net 2008 program in the WinPAC-8xx7 plus I-87018z?
109	Q: How to adjust the system time of some ISaGRAF PACs via Ebus by using ISaGRAF PAC and I-87211w?
110	Q: ZigBee Wireless Application: How to control remote I/O and acquire data?
111	Q: How to use the GTM-201-RS-232 to send a short message in user's local language ?
112	Q: Program the I-8093W (3-axis high speed Encoder input module) by ISaGRAF.
113	Q: Linking ISaGRAF PAC to Modbus TCP/IP Slave Devices By Modbus TCP Master Protocol.
114	Q: How to avoid garbled content when printing ISaGRAF PDF documents?
115	Q: Working eLogger HMI with ISaGRAF SoftLogic in the WP-8xx7, VP-2xW7 and XP-8xx7-CE6 PAC. (the document version is 1.03 released on Jul.15,2010)
116	Q: How to enable the second to fifth Modbus RTU slave port of the WP-8xx7 and VP-2xW7 without modifying the ISaGRAF project ?
117	Q: How to install the ISaGRAF Ver. 3 on Windows Vista or Windows 7?
118	Q: A M.S. VC++ 6.0 Demo Program To Connect One WP-8xx7 by Modbus TCP Protocol.
119	Q: How to implement the communication redundancy between the central control station and the local stations?
120	Q: How to calculate the moving average value of a variable by c-functions "Aver_N" or "Aver_F" ?
121	Q: How to install or remove the ISaGRAF development platform properly?
122	Q: How To Solve The USB-Freeze Problem Of The W-8x4x ? How To Update The W-8x4x 's OS Image ?
123	Q: How to move the InduSoft picture faster in the W-8xx6 / WP-8xx6 / VP-25W6 / XP-8xx6-CE6 ?
124	Q: A Web HMI Example for ISaGRAF Professional XPAC XP-8xx7-CE6-PRO – by FrontPage .
125	Q: XP-8xx7-CE6 And iDCS-8000 (Or ET-7000 Or Modbus TCP Slave device) Redundant System.
126	Q: How to use the WP-8847 to connect ET-7018Z and ET-7044D and develop the HMI program by InduSoft, VS2008 C# and VB.NET ?
128	Q: How to use The ISaGRAF PAC plus I-87113DW - the master card of the Carlson Strain Gauage Inputs ?
129	Q: How To Connect The ICP DAS Power Meter – PM-2133 and PM-2134 By The ISaGRAF PAC ?
130	Q: How to automatically synchronize the time of WP-8x47/VP-23W7 over a network ?
131	Q: Soft-GRAF : Create A Colorful HMI in The XP-8xx7-CE6 and WP-8xx7 and VP-2xW7 PAC (paper version: 1.3) .

No.	English ISaGRAF Ver.3 FAQ
132	Q: Motion Control - Using I-8094F/8092F/8094
133	Q: How to send and receive UDP / TCP data ?
134	Q: How to reset the ISaGRAF driver or reset the whole controller by software ?
135	Q: How to program ISaGRAF PAC to support SQL Client to write data to (or read data from) Microsoft SQL server ?
136	Q: HART Solution : ISaGRAF PAC plus I-87H17W
137	Q: How to connect to remote server and send network package via GPRS with uPAC-5000 series controller?
138	Q: How to program an XP-8xx7-CE6 redundant system (with I-87K8 expansion base or Modbus I/O or other I/O) ?
139	Q: How to install/use ISaGRAF 3.55 Demo Version and its limitations
140	Q: How to communicate between InduSoft local HMI and ISaGRAF PACs via Modbus TCP protocol?
141	Q: iP-8xx7/µPAC-7186EG/I-8xx7/I-8xx7-80 provide the Flash memory write protect feature
142	Q: How to protect your ISaGRAF program from used by the unauthorized people?
143	Q: How to Make “ISaGRAF WinCE PAC” to Connect to the Internet and Send Data by GPRS Dial-up?
144	Q: A new function block “Mbus12w” to write max. 12 words to Modbus salve devices.
146	Q: Soft-GRAF Studio : Create a Colorful HMI in the XP-8xx7-CE6 & WP-8xx7 & VP-2xW7 PAC
147	Q: How to use the VPD-130 to read the µPAC-7186EG’s system date and time via RS-485?
149	Q: How to make the ISaGRAF WinCE PAC play a sound ?
150	Q: ISaGRAF Tutorial Video .
151	Q: How to use FTP Client to upload log files to remote FTP Server on PC?
152	Q: How to control the IR module, IR-210/IR-712, with the ISaGRAF PACs?
153	Q: How to use the ISaGRAF PAC to communicate with a far away Modbus TCP server or a ftp server by the 3G or 2G wireless GPRS ?
155	Q: How to save the value of ISaGRAF variables to the Flash memory or Micro_SD memory in the WP-5xx7, XP-8xx7-CE6, WP-8xx7 and VP-25W7 / VP-23W7 PAC ?

Chapter 12. Sending Email

WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6, VP-25W7, VP-23W7, WinCon-8xx7, µPAC-7186EG, µPAC-5xx7 and iPAC-8447/8847 can send email via its Ethernet port since its following ISaGRAF driver version.

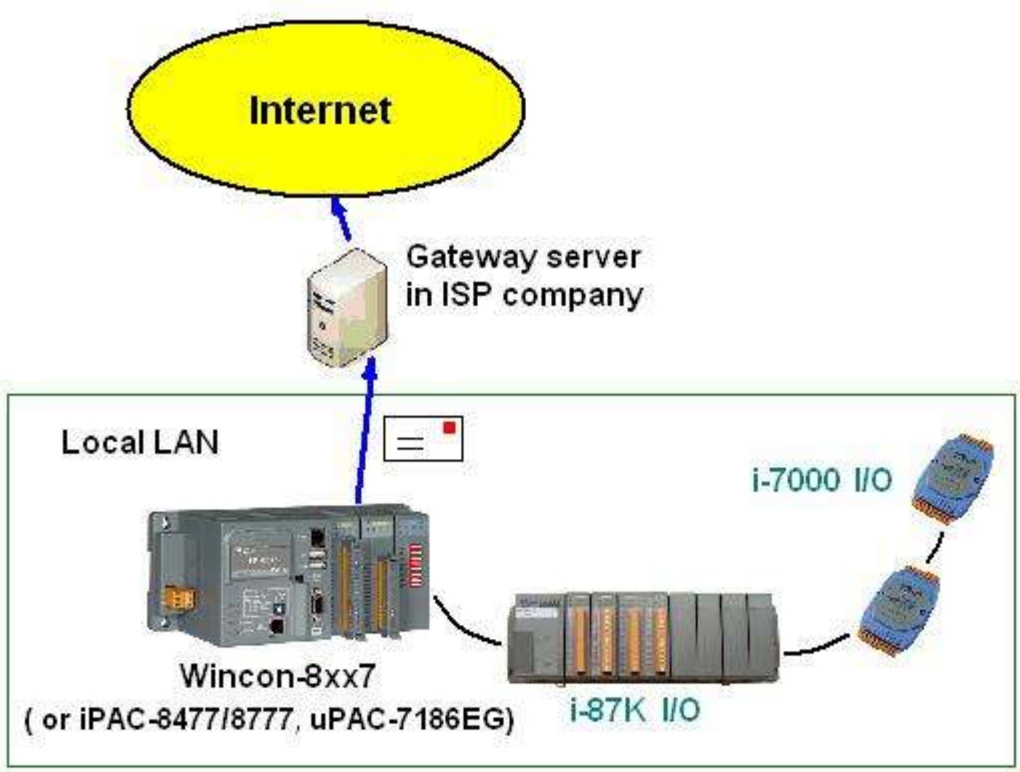
WP-8xx7/WP-5xx7:	since it is released
XP-8xx7-CE6/ XP-8xx7-Atom-CE6:	since it is released
VP-25W7, VP-23W7:	since it is released
WinCon-8xx7/8xx6:	3.42
iPAC-8447 / 8847 :	since it is released
µPAC-7186EG:	since it is released

These controllers must reside at a local network which can connect to the Internet, or sending email is not possible.

For sending Email or TCP, UDP data via 2G / 3G wireless communiction, please refer to http://www.icpdas.com/faq/isagraf.htm > FAQ-143 to setup the wireless devices.

New released ISaGRAF driver: http://www.icpdas.com/products/PAC/i-8000/isagraf-link.htm
Demo program: www.icpdas.com – FAQ – Software – ISaGRAF – 067, 071, 072, 076 and 077
or
Wdemo_62 , Wdemo_63, Wdmo_63a, Wdmo_65a, Wdmo_65b, demo_74a and demo_75a
or http://www.icpdas.com/faq/isagraf.htm

Controller can send email without or with one attach file



Features:

1. The sending Email can contain one attached file or without any attached file. The attached file format can be text or binary or any file format. The approximate max. file size is listed as the following.

WP-8xx7,WP-5xx7, VP-25W7:	2 MB
XP-8xx7-CE6, XP-8xx7-Atom-CE6:	2 MB
WinCon-8xx7:	2 MB
iPAC-8447 / 8847:	488KB
µPAC-7186EG + X607:	112KB
µPAC-7186EG + X608:	488KB

2. Email Title can be max. 128 bytes. Email content can be max. 510 bytes. Local language word can be used (English, Chinese, any language character which computer can use).

3. One email can send to 10 receivers at one sending.

4. Each email can be assigned as High , Low or Normal priority.

5. Please assign at least one Mail server IP in the ISaGRAF program. **Or for safety, assign two Mail servers IP. Then if one Mail server is out of service, the controller will send this email by the other Mail server**.

6. If controller model is WP-8xx7, WP-5xx7, XP-8xx7-CE6, XP-8xx7-Atom-CE6, iP-8xx7 or W-8x47 (dual LAN) and both LAN ports are enabled, the controller will automatically switch to the other Ethernet port to send email if one is broken or damaged.

7. If the sending email has one file attached, this file must be stored or copied to the correct file path before it is sent.

   WP-8xx7, WP-5xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6, VP-25W7:
   WinCon-8xx7:
   file should be stored in the path of '\Email_ETH\' , for ex, the '\Email_ETH\A1.txt'.

   For your reference usage:
   (To copy the file into the path '\Email_ETH\')

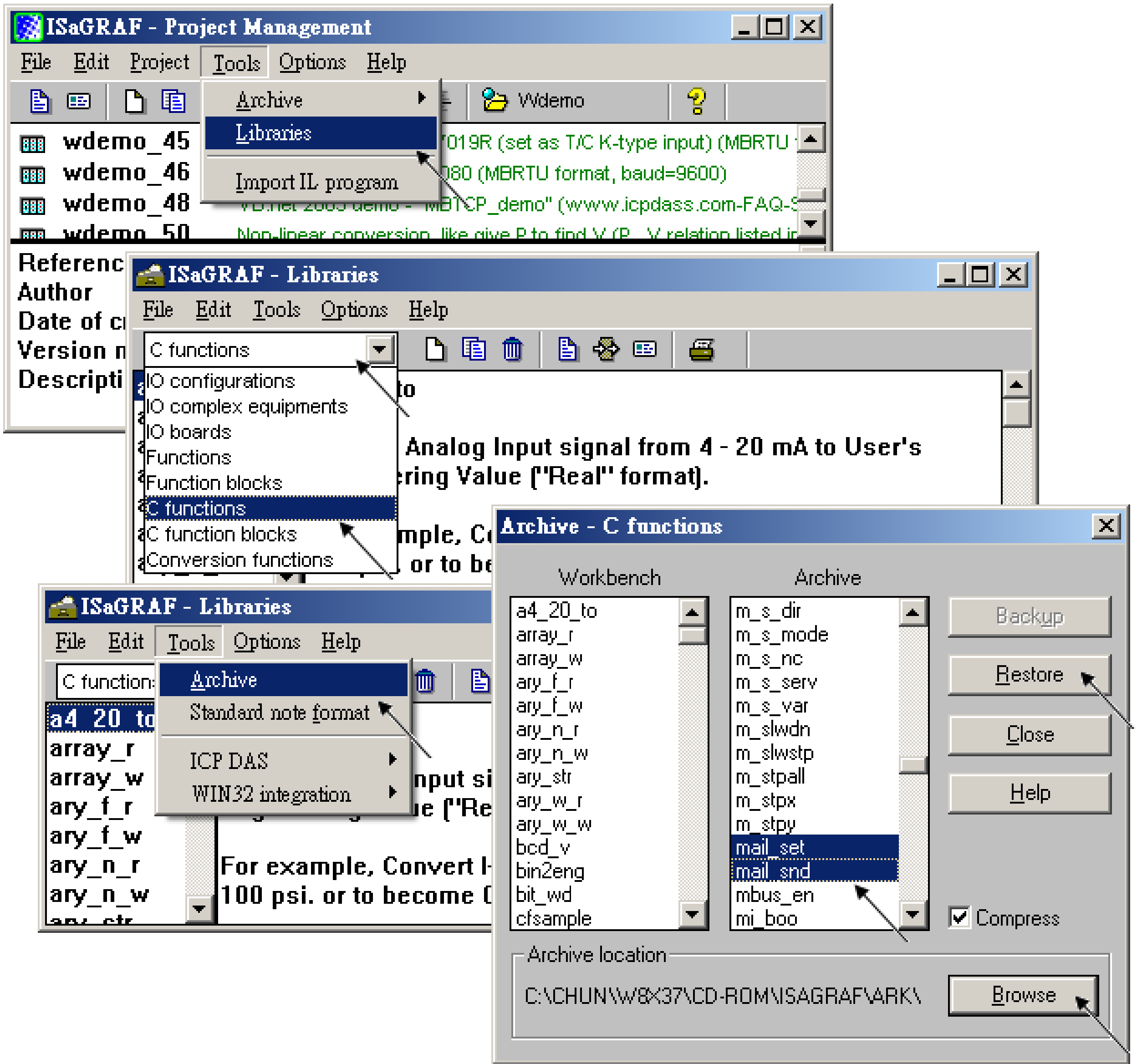
   TMP:=F_copy('Micro_SD\B9.jpg', '\Email_ETH\ B9.jpg) ;

   (After sending the email, you can delete the file)

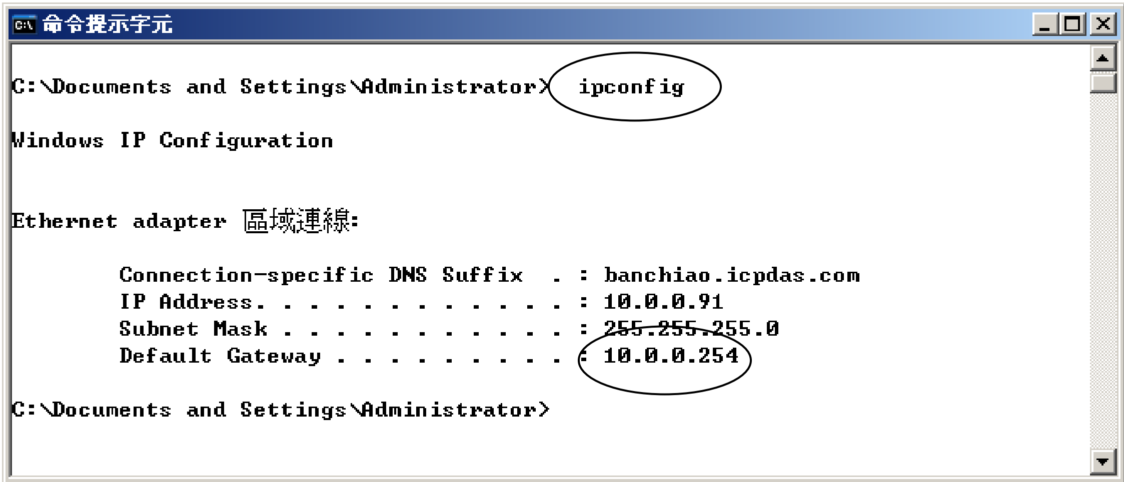
   TMP:=F_delete('\Email_ETH\ B9.jpg') ;

   iPAC-8447 / 8847 & µPAC-7186EG, µPAC-5xx7:
   file should be stored in the battery backup memory by the "S_xxx" functions, like the "s_fl_ini" , "s_fl_avl" , "s_m_r" , … (please refer to section 10.3 and appendix A.4)

Please make sure if your ISaGRAF software in PC has installed the ISaGRAF c-function of "Mail_snd" , "Mail_set" and "R_mb_adr". If not installed, please visit http://www.icpdas.com/faq/isagraf.htm – FAQ-067, 076 to download the Demo program. Then restore "Mail_snd.uia" , "Mail_set.uia" and "R_mb_adr.uia" to your ISaGRAF in PC by below steps.

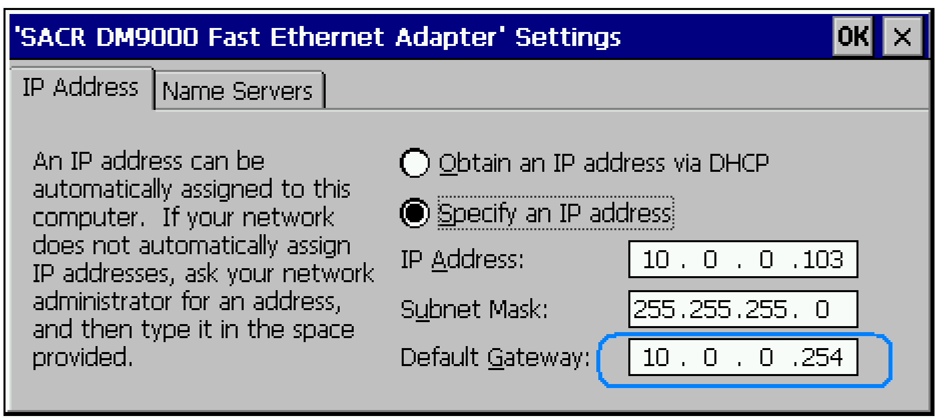


To send email correctly, please set proper Gateway IP in the controller's Ethernet port setting. Please type command "ipconfig" in a PC 's command prompt window at the same local network to get the Gateway IP setting as below. (Here is 10.0.0.254)

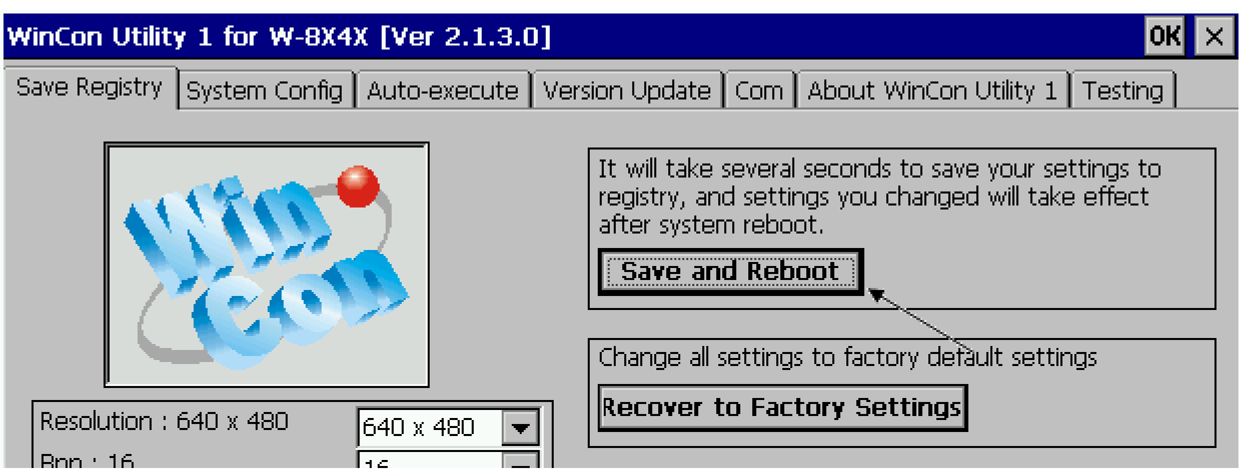


Then please fill-in this Gateway IP address to your controller's Ethernet port setting (If controller model is W-8347/8747, you can enable two Ethernet ports, then you need to fill-in both with the same Gateway IP)

WinCon-8xx7 / 8xx6:



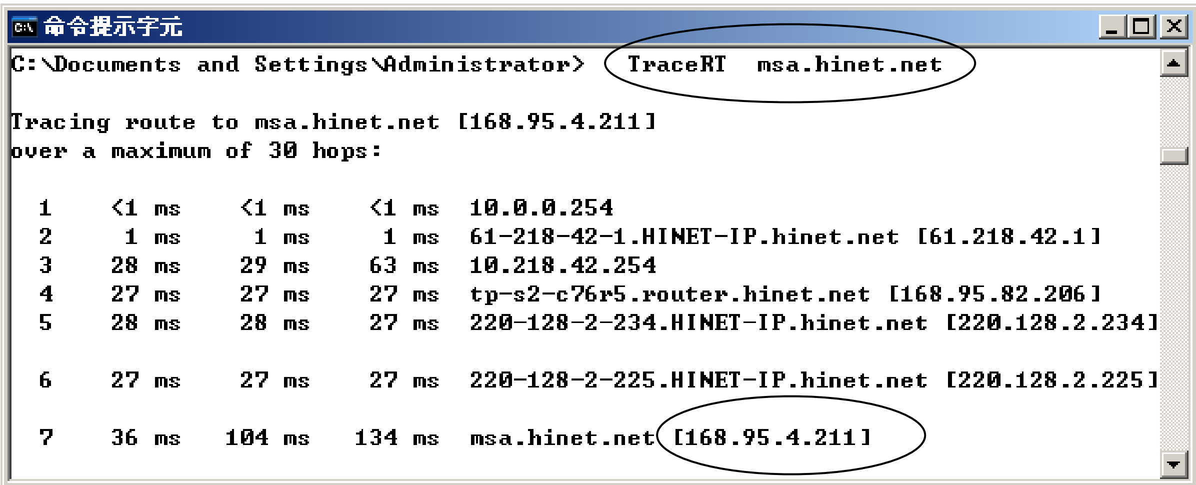
Then run WinCon Utility – Save and Reboot to store the IP setting. It will automatically re-boot once.



iPAC-8447 / 8847 & µPAC-7186EG & µPAC-5xx7 :

Please run “7188xw.exe” in the PC and give command for ex, “gateway 10.0.0.254” if the gateway IP is 10.0.0.254. (Please refer to ISaGRAF User’s Manual - appendix B)

The PC ‘s command prompt windows can also request the Mail server‘s IP address (We need it in the ISaGRAF program). For example, to request IP of msa.hinet.net , please type command **TraceRT msa.hinet.net** as below (Here is 168.95.4.211)



Email demo download from www.icpdas.com > FAQ > Software > ISaGRAF > 067, 076 has three example programs.

“Wdemo_62.pia” is the demo without attached file.
“Wdemo_63.pia” is the demo with one attached file
(For W-8xx7,WP-8xx7, WP-5xx7, VP-25W7, XP-8xx7-CE6, XP-8xx7-Atom-CE6).
“Wdmo_63a.pia” is the demo with one attached file
(For µPAC-7186EG, µPAC-5xx7 and iPAC-8447/8847) .

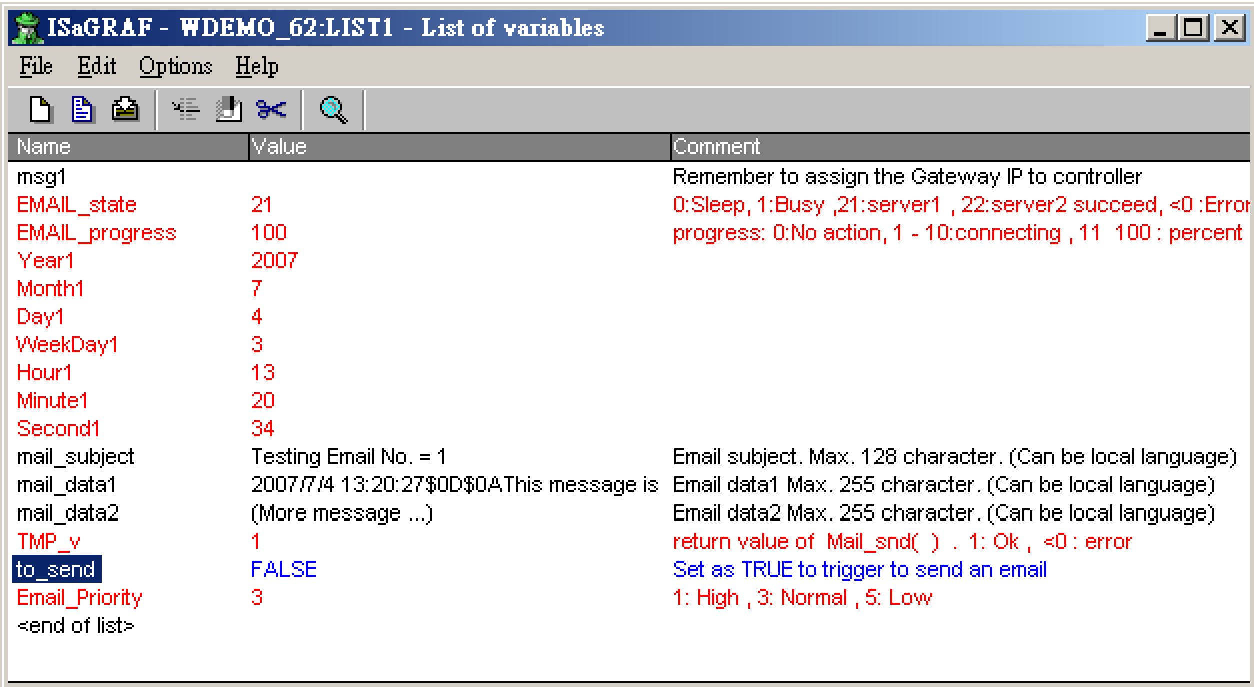
Please modify at least the below setting in the demo program to your own setting .

```
TMP := MAIL_SET( 1 , 'chun@icpdas.com' ) ;            (* Receiver 1. please modify it *)
TMP := MAIL_SET( 100 , 'go_mao@hotmail.com' ) ;  (* Sender. please modify it *)
TMP := MAIL_SET( 101 , '168.95.4.211' ) ;              (* Mail server 1‘s IP, please modify it *)
```

(* Some of the Mail Server required to login before sending mail, so the user need to set these two commands below. If Mail Server1 and Server2 no need to login, please do not set these commands. *)

```
TMP := MAIL_SET(104, 'MY_ACCOUNT')     (The account has been registered on Mail Server)
TMP := MAIL_SET(104, 'MY_PASSWORD')  (The password has been registered on Mail Server)
```

Then re-compile it and then download it to the controller to run. The below windows will show up. Please set “to_send” as TRUE to trigger to send one email. Few seconds later, value of “Email_state” will be 21 or 22 if succeed. However value of “Email_state” will be less than 0 if failed. When “Email_progress” reach value of 100, it means the email data is 100% sent.



Name	Value	Comment
msg1		Remember to assign the Gateway IP to controller
EMAIL_state	21	0:Sleep, 1:Busy ,21:server1 , 22:server2 succeed, <0 :Error
EMAIL_progress	100	progress: 0:No action, 1 - 10:connecting , 11 100 : percent
Year1	2007	
Month1	7	
Day1	4	
WeekDay1	3	
Hour1	13	
Minute1	20	
Second1	34	
mail_subject	Testing Email No. = 1	Email subject. Max. 128 character. (Can be local language)
mail_data1	2007/7/4 13:20:27$0D$0AThis message is	Email data1 Max. 255 character. (Can be local language)
mail_data2	(More message ...)	Email data2 Max. 255 character. (Can be local language)
TMP_v	1	return value of Mail_snd( ) . 1: Ok , <0 : error
to_send	FALSE	Set as TRUE to trigger to send an email
Email_Priority	3	1: High , 3: Normal , 5: Low
<end of list>		

Below is the description of the three ISaGRAF functions for sending email.

MAIL_Set(CMD_ , MSG_)

Parameters:

CMD_ Integer Can be the following value.

1 : Set receiver 1 , for example, TMP := Mail_set(1 , 'chun_tsai@icpdas.com') ;
Max. receiver length can not exceeds 48 characters.

2 to 10 : Set receiver 2 to 10 if they exist.

100 : Set the sender , for ex, TMP := Mail_set(100 , 'sender1@icpdas.com') ;
Max. sender length can not exceeds 48 characters.

101 : Set the mail server 1 's IP address , for ex, TMP := Mail_set(101 , '168.95.4.211') ;
102 : Set the mail server 2 's IP address if it exist.
103 : a new TCP port No. for sending email. (Default is 25 “SMTP protocol”)

******* Since the following ISaGRAF driver version support CMD_104, 105 and 106 *******

μPAC-7186EG:	(ISaGRAF driver Ver. 1.14 or later)
μPAC-5xx7:	(Since it is released)
iP-8xx7:	(ISaGRAF driver Ver. 1.10 or later)
WP-8xx7/8xx6:	(ISaGRAF driver Ver. 1.37 or later)
WP-5xx7/5xx6:	(Since it is released)
VP-25W7/23W7/25W6/23W6:	(ISaGRAF driver Ver. 1.29 or later)
XP-8xx7-CE6/ XP-8xx6-CE6:	(ISaGRAF driver Ver. 1.17 or later)
XP-8xx7-Atom-CE6/ XP-8xx6-Atom-CE6:	(Since it is released)

104 : Set the user_account for the Mail Server which required to login before sending mail.
For example, TMP := Mail_set(104, 'my_account')

105 : Set the user_password for the Mail Server which required to login before sending mail.
For example, TMP := Mail_set(105, 'my_password')

If Mail Server1 and Server2 no need to login, please do not set CMD_104, 105.

106 : Set “Timeout”, unit is second, the value can be 30 ~ 180, the default value for the driver version listed above is 60 second, the old version is 20 second.

MSG_ Message the related message setting according to the 1st parameter - CMD_

Return:

Q_ Boolean True : Ok .
False : the related setting is not correct or the "CMD_" value is not correct.

MAIL_snd(Start_ , Num_ , Subject_ , Prio_ , Data1_ , Data2_ , Attach_)

Parameters:

Start_	Integer	Starting receiver No. Can be 1 through 10.
Num_	Integer	Number of receivers. Can be 1 through 10.
Subject_	Message	Subject of the email. Max. length is 128 characters. For ex, 'Alarm of plant 1'
Prio_	Integer	Set Email Priority symbol. Value can be 1 : High , 3 : Normal , 5 : Low ; default setting is 3.
Data1_	Message	The email data 1 (Max. 255 characters). For ex, 'Pressure 1 is too high. Please check it soon ! ... '
Data2_	Message	The email data 2 (Max. 255 characters). More message behind the "Data1_". For ex, 'More message ...'
Attach_	Message	The attached file name or file ID if it exists. (It depends on controller) Please give '' (empty messge) if no attached file used.

WP-8xx7, WP-5xx7 & W-8xx7:
The file name can be Max. 64 characters and it must store in the '\Email_ETH\' folder . For ex, '\Email_ETH\A1.txt'

µPAC-7186EG : The file must store in the X-607, X-608 memory.
iPAC-8447/8847 : The file must store in the built-in battery SRAM in the backplane.

the valid value is '1' , '2' , … , '8' . the number is the file ID No. set by the "S_FL_AVL" function. (refer to section 10.3 or appendix A.4)

The max. attached file size are listed as following.

Wincon-8xx7:	2 MB ,	iPAC-8447 / 8847:	488 KB
µPAC-7186EG + X607:	112 KB ,	µPAC-7186EG + X608:	488 KB

Return:

Q_ Integer

1 : Ok, then start sending email.
< 0 : error
-1 : Busy . The earier email is still sending.
-2 : The first Receiver (No. = "Start_") is empty or error.
-3 : Mail server 1 is empty or error.
-4 : Sender is empty or error.
-5 : "Start_" value less than 1 or larger than 10
-6 : "Subject_" exceeds 128 characters.
-7 : Email system is not active yet, Please use "mail_set( )" to set at leaset one receiver email box address, one mail server IP and the sender email box address
-8 : "Num_" value less than 1 or larger than 10
-9 : The given attached file name doesn't exist or file path name > 64 characters or its size exceeds the allowed file size.

R_MB_ADR(1 , 9995) is to get the email sending state sent by "Mail_snd( )" .

The return value of R_MB_ADR(1 , 9995) will remain until next calling "Mail_snd( )"

Return :

0 : Sleep. No action
1 : Busy. one email is still sending now
21 : Email is successfully sent through Mail server 1
22 : Email is successfully sent through Mail server 2

< 0 : Error happens
-1 : Can not connect to the Mail server
-2 : Sender setting is rejected by the Mail server
-3 : Time out
-4 : Ethernet socket error
-5 : receiver setting is reject by the Mail server

R_MB_ADR(1 , 9994) is to get the current email sending progress sent by "Mail_snd( )" .

Calling "R_MB_ADR(1 , 9994)" can not get the Error No. when error happens. Please use "R_MB_ADR(1 , 9995)" to get it .
If error happens while sending email, the return value will stay at its last value until next calling "Mail_snd( )"

Return :

0 : No action
1 : Connecting to Mail server 1
2 : Mail server 1 connected . Sending "HELO "
3 : Sending "MAIL FROM: ..." to Mail server 1
4 : Sending "RCPT TO: ..." to Mail server 1
5 : Sending "DATA" to Mail server 1
6 : Connecting to Mail server 2
7 : Mail server 2 connected . Sending "HELO ..."
8 : Sending "MAIL FROM: ..." to Mail server 2
9 : Sending "RCPT TO: ..." to Mail server 2
10 : Sending "DATA" to Mail server 2

11 ~ 100 : the current progress of sending email data.

For ex,	25	means	25	/ 100 =	25 %
	36	means	36	/ 100 =	36 %
	95	means	95	/ 100 =	95 %
	100	means	100	/ 100 =	100 % (sent completely)

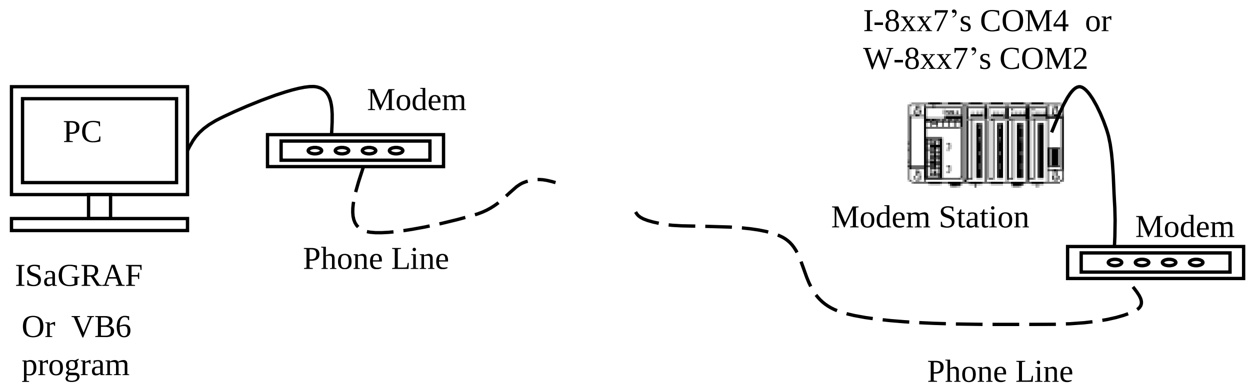
Chapter 13. Remotely Download Via Modem_Link

13.1: Introduction

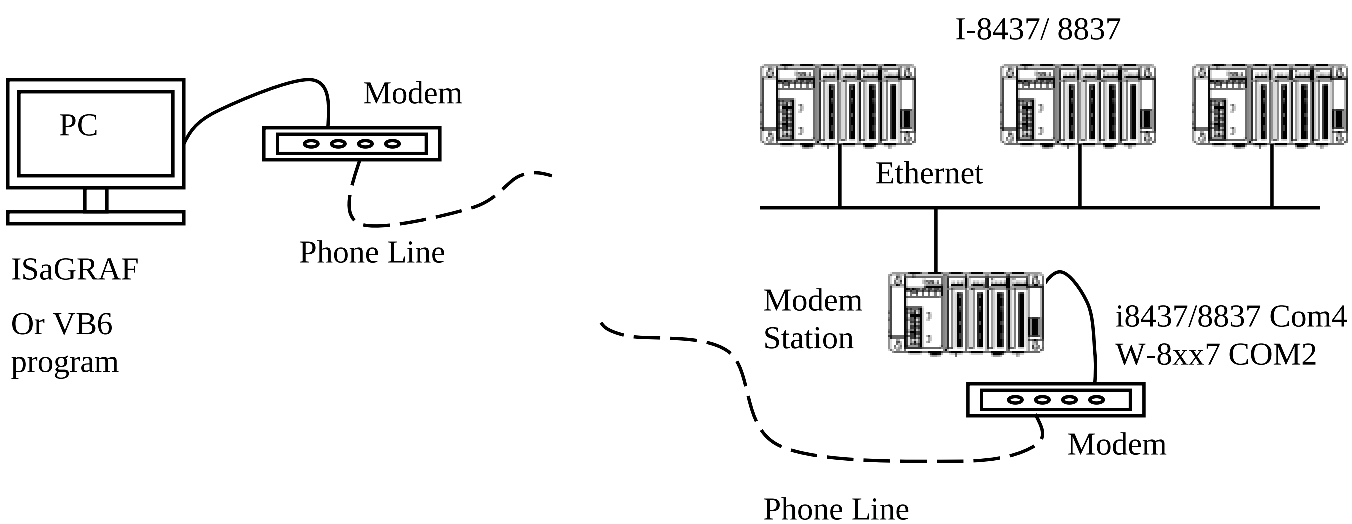
COM4 of the I-8417/8817/8437/8837 & COM2 of the W-8xx7 supports full modem signals. It has embedded the Modem_Link protocol for remotely download and monitoring since the I-8xx7 driver version of 2.14 & W-8xx7 driver version of 3.10. Please refer to Appendix C to make sure your controller driver version is the same or higher. You can obtain the new released driver from:

http://www.icpdas.com/products/PAC/i-8000/isagraf-link.htm

To Remotely download and monitor program via the Modem_Link, I-8xx7's COM4 & W-8xx7's COM2 has to link to a modem. They have exactly the same pin assignments as the Com1 (9-pin Dsub) of the PC.



We name the controller as "**Modem Station**" since it will pick up the phone call coming from the remote PC running ISaGRAF. If the controller is either I-8437 or I-8837 (Ethernet controller), The configuration can be extened to link many controllers together. Therefore, the PC running ISaGRAF can remotely download to anyone of them through the modem and the Modem station.



Note:

1. W-8xx7's COM2 can be set as Modbus RTU port, **please disable it if using as "modem_link" port**. Please refer to W-8xx7's "Getting Started" Manual.
2. WinCon-8xx7 just support Modem station (which connect to Modem)

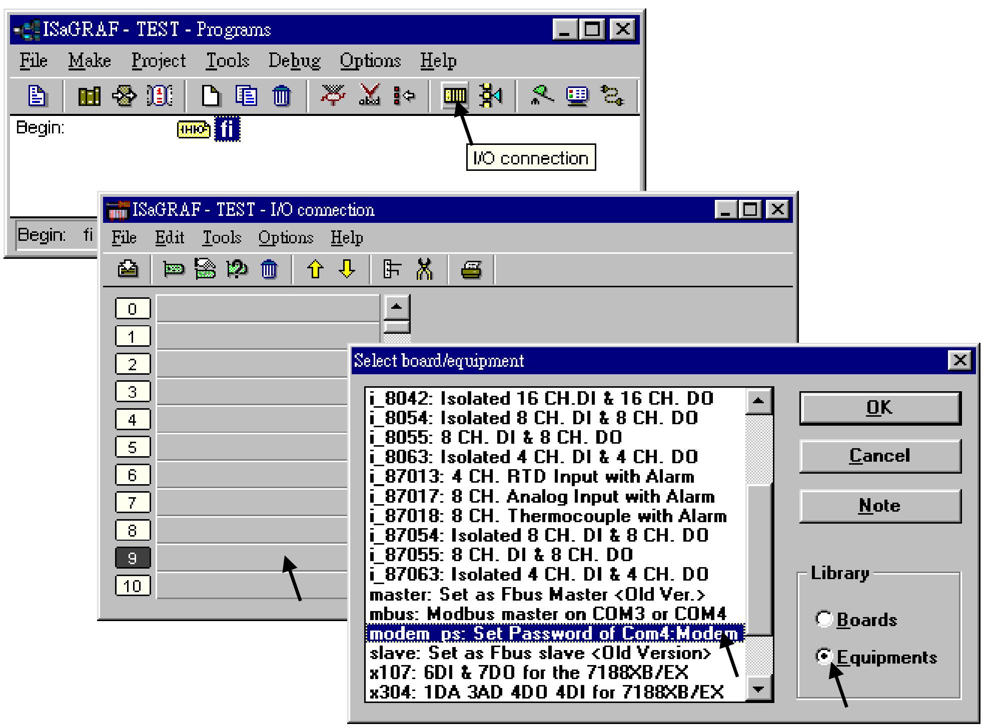
13.2: Download Program Via Modem_Link

Note:

1. **The quality of the phone line, busy situations and weather conditions may have dangerous effects on changing a program in the remote controller. So, please try not to download the program via Modem_Link unless necessary. But, the users can still monitor the operations of the controller and it's not dangerous.**
2. **The COM2 of W-8xx7 is used as a Modbus RTU port by default, please disable it if using as "modem_link" port. Please refer to W-8xx7's "Getting Started" Manual.**

Warnning:
Do not download a project which uses I-8xx7's COM4 & W-8xx7's COM2 to do other things to the "Modem station" controller. Because the "Modem_Link" function will be invalid and can not remotely connect unless you move to the controller and set the program again. For example, do not connect "Bus7000" & "Mbus" with port_no = 4 (for I-8xx7) & port_no=2 (for W-8xx7). And do not use "Comopen" to open Com4(for I-8xx7) & Com2(for W-8xx7). It will disable "Modem_Link". But, the "Email" function described in chapter 12 has no this limitation.

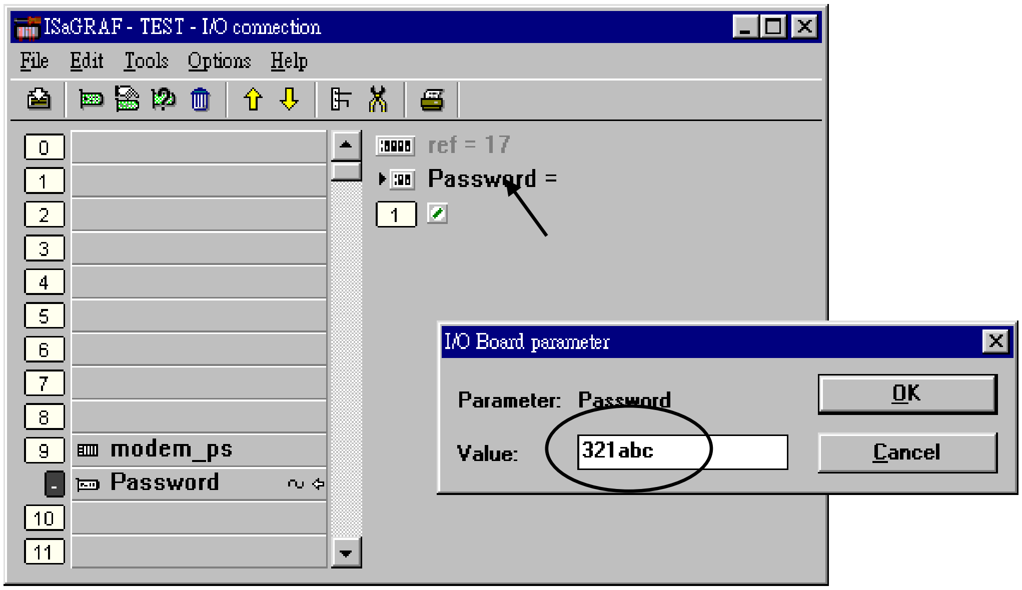
The first thing is to add a "modem password" to your ISaGRAF program of the "Modem station" controller for security. To do it, click on one empty slot No. from the I/O connection window. Then connect "Modem_PS" on the slot.



Then you got the window similar as below. Type in your prefered password for the "Modem station" controller. The password can contain up to 12 characters & can't use character " and '.
Then re-compile it and download it to the "Modem station" controller.

Note:
User can write Visual Basic program to acess to the I-8417/8817/8437/8837 via Modem. Please download VB6 demo source code at

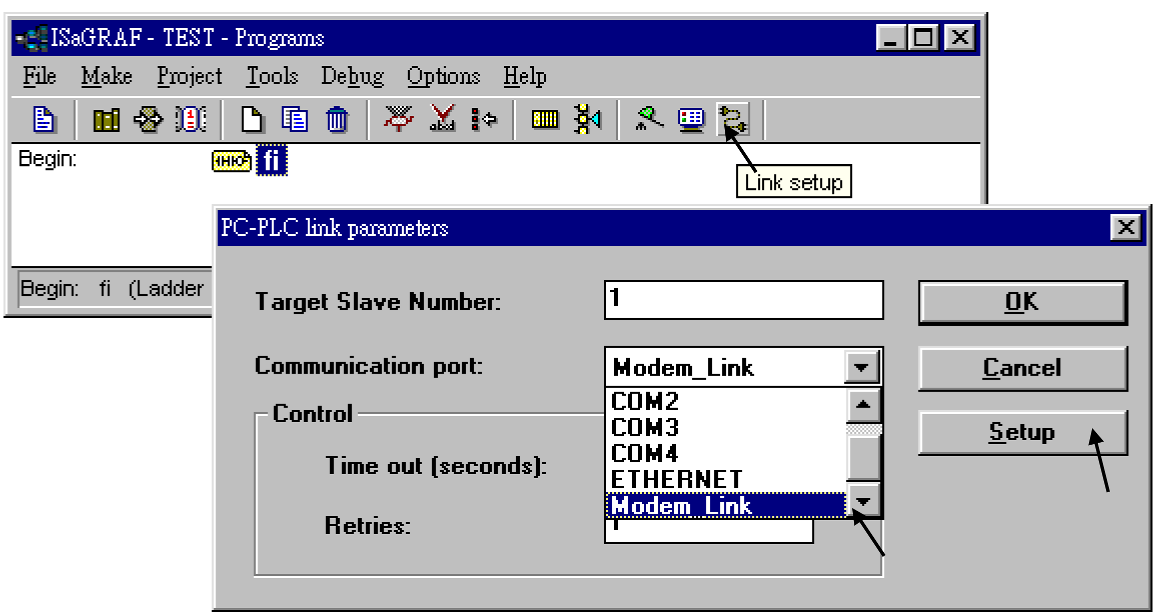
ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/vb_demo/ or
I-8000 CD-ROM:\napdos\isagraf\vb_demo\



Very Important:
If you don't assign the Modem password to the "Modem station" controller, anyone who has the phone No. of your "Modem station" controller can link to it to do anything. Be very careful.

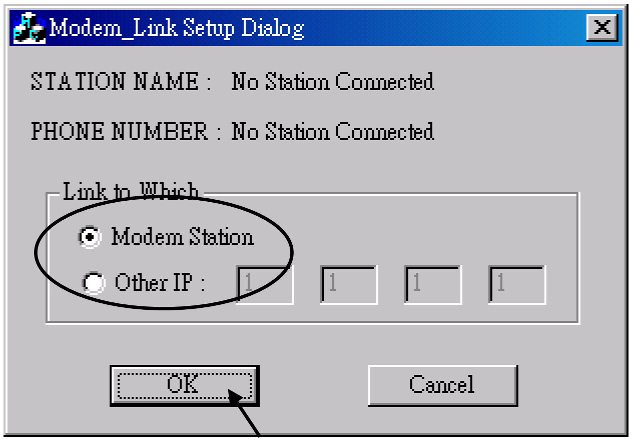
Now we are going to download and monitor the program of faraway controllers.

Click on "Link setup", select "Modem_Link", and then click on "Setup"

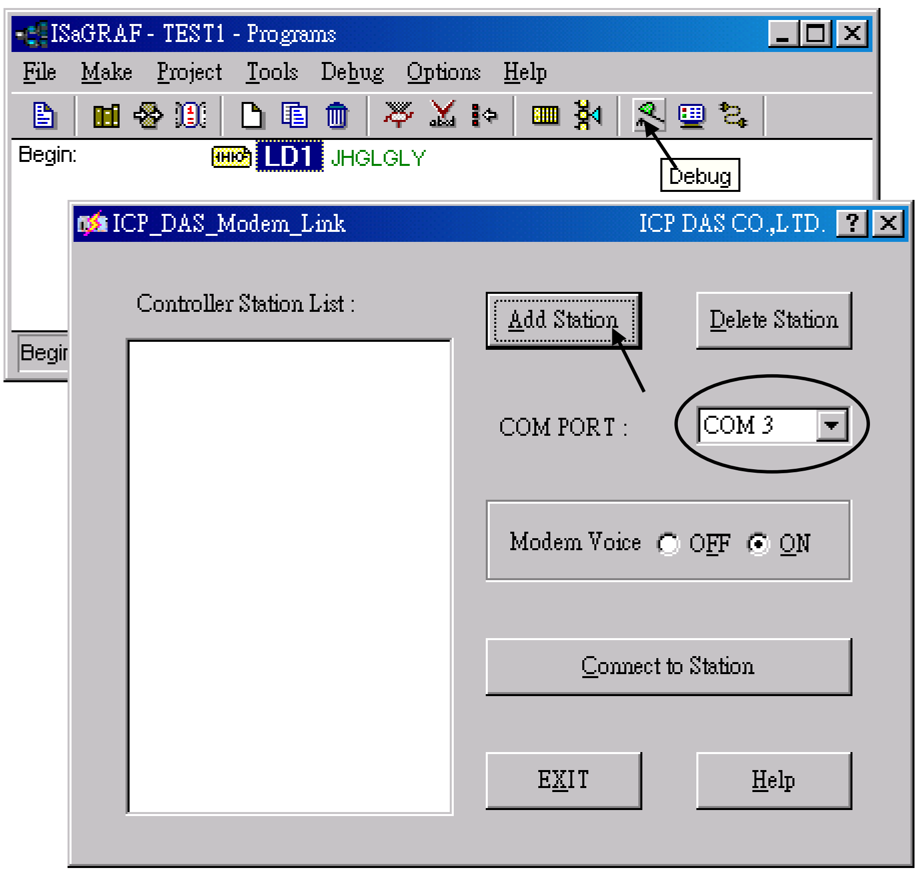


For windows NT, 2000 & XP users:
(For Windows95, 98 users, Please skip to next three page)

If you are going to connect the "Modem station" controller, check "Modem station", otherwise check "Other IP". "Other IP" means the target controller is not connect to a modem however connect to the "Modem station" controller via an ethernet cable, the IP address has to be assigned.

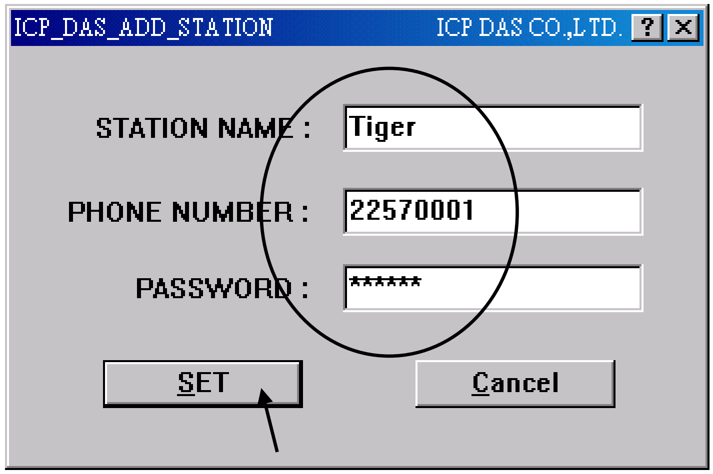


Then click on "debug". Select the correct COM port of your PC which will dial the modem. And then click on "Add Station" to add a station if you have none.

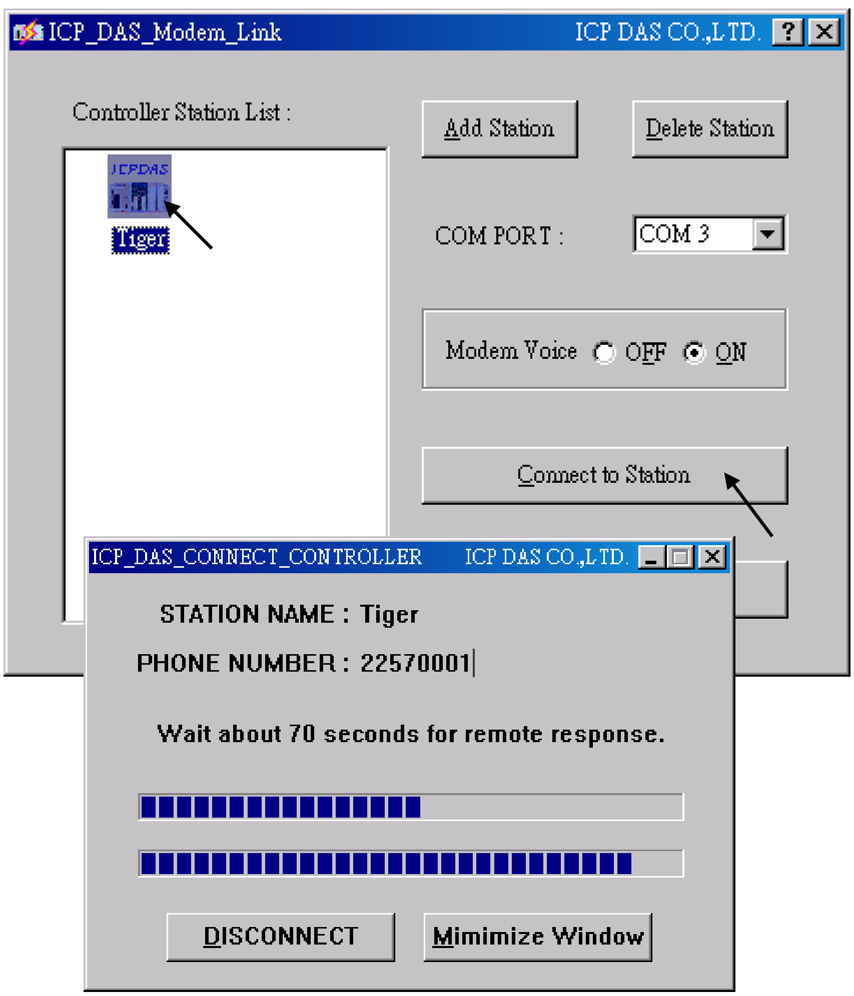


Then you will see the below window. Given a name for this new station and the target phone No. If you add a "," character inside the phone No. It will wait one second and then dial the rest No. For ex. Given

No. as “9,,22570001” will dial “9” first, then wait 2 seconds and then dial “22570001”. The password must set to the same password of the “modem station” controller.

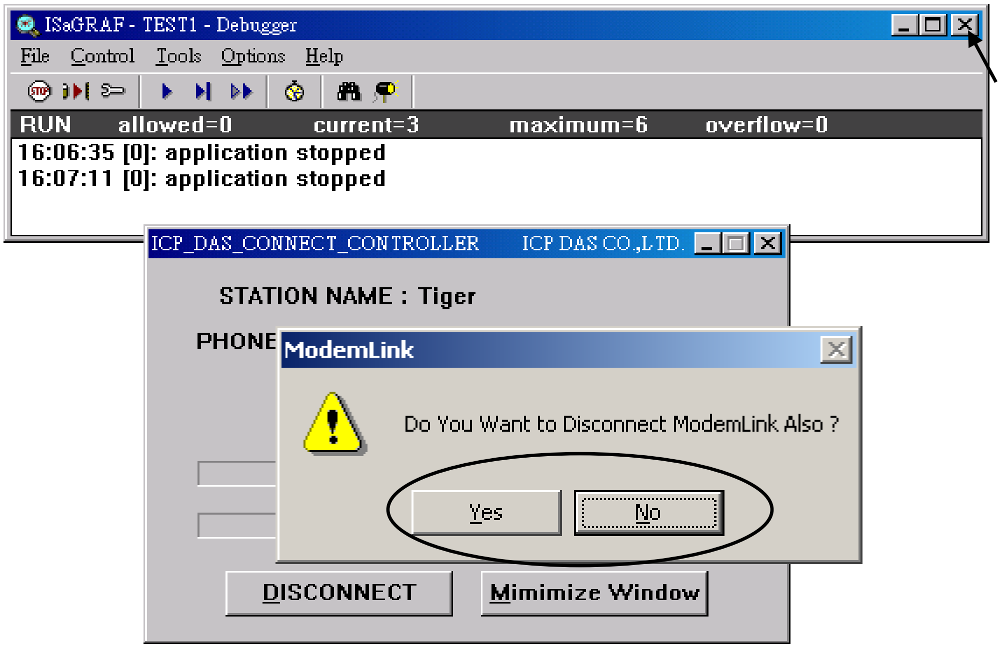


Click on the station you would like to connect first and then click on “Connect to Station” to command the modem dialing to the faraway controller.

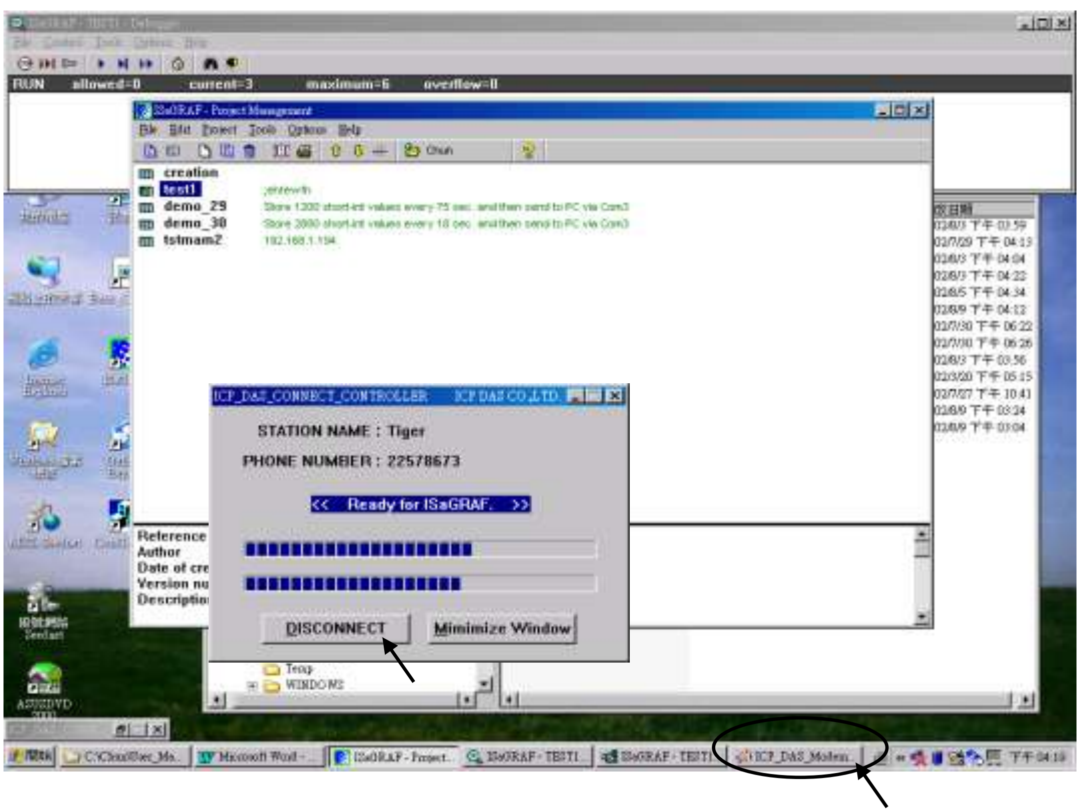


After the connection is Ok. You can download, monitor and change the variable value just like you did when the controller is near beside you.

To disconnect from the target controller, close the " … Debugger" window. Then you can choose "No" to keep the phone connected, or "Yes " to hang off phone. If you choose to keep the phone connected, you can open another ISaGRAF project to directly connect to another faraway target. The modem won't dial again.



However, keep in mind, remember to disconnect the modem_link when you finish your work, don't waste the money to the telecom company.

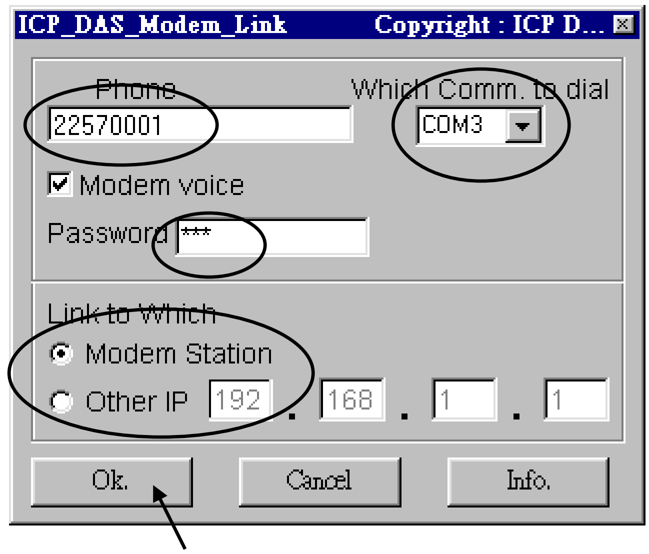


For windows 95 & 98 users:

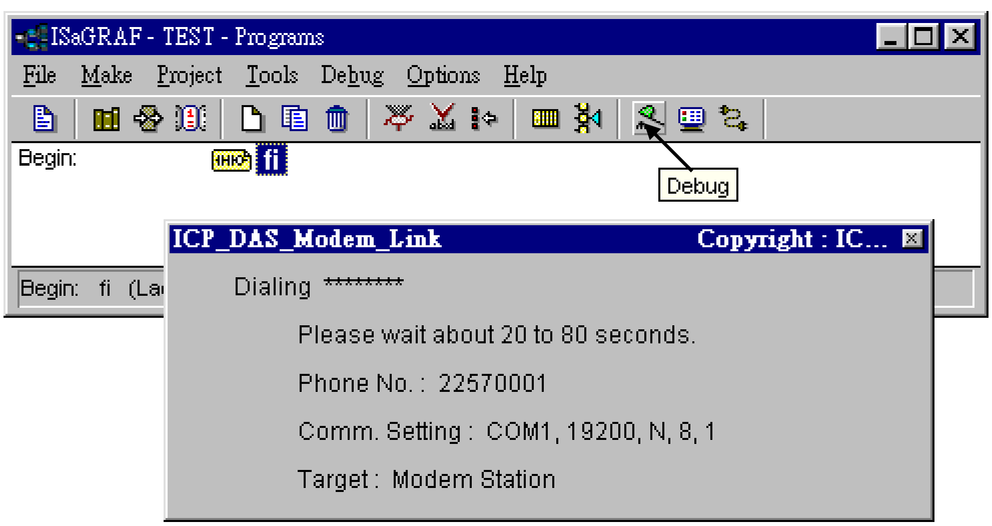
Given the correct target phone No. and the correct COM port of your PC which will dial the modem.

If you add a “,” character indise the phone No. It will wait one second and then dial the rest No. For ex. Given No. as “9,,22570001” will dial “9” first, then wait 2 seconds and then dial “22570001”. The password must set to the same password of the “modem station” controller.

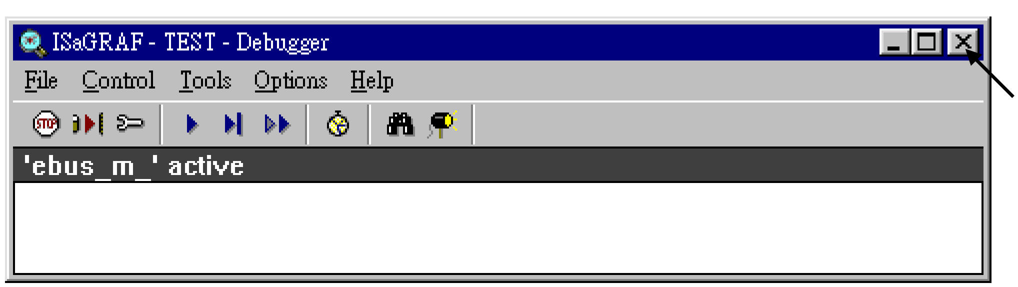
If you are going to connect the “Modem station” controller, check “Modem station”, otherwise check “Other IP”. “Other IP” means the target controller is not connect to a modem however connect to the “Modem station” controller via an ethernet cable, the IP address has to assign.



Then click on “debug” to start dialing the modem to connect to the faraway controller.



After the connection is Ok., you can download a new program, monitor the variable status just like you did when the controller is near beside you.When you close the “ … Debugger” window, the PC will command the modem to hang off the phone and disconnect with the faraway controller.



Chapter 14. Spotlight : Simple HMI

Spotlight is a simple HMI coming with ISaGRAF which allows user to build **Boolean Icon, Bar Graph, Trend Curve, Value Text, Bitmap Picture** to make application more friendly.

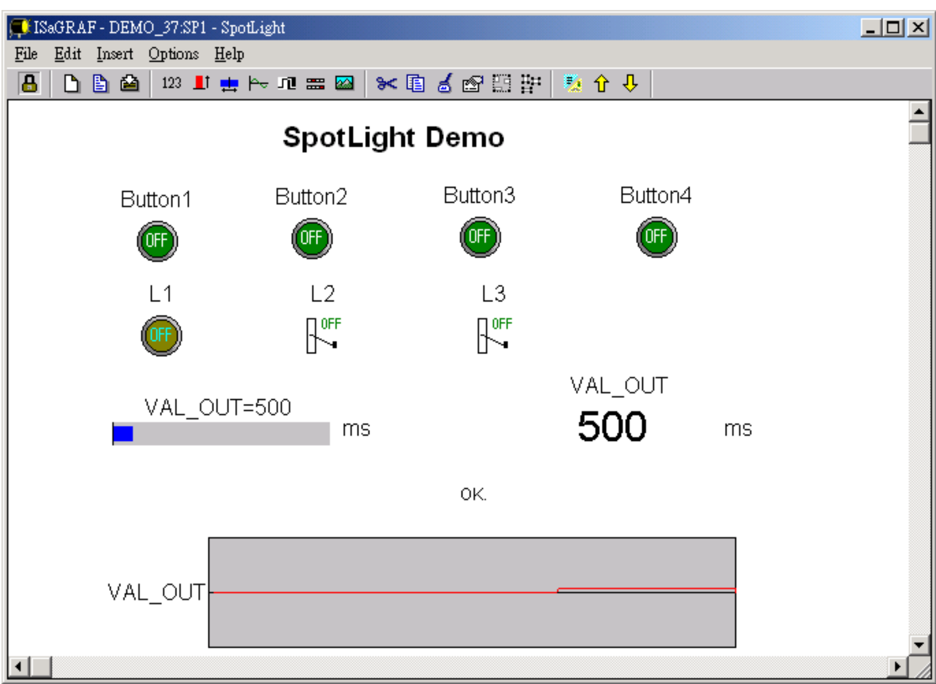
14.1 A Spotlight Example:

This Demo example can be restored from the ICP DAS's I-8000 CD-ROM - "demo_37". Please refer to Chapter 11 to restore it.

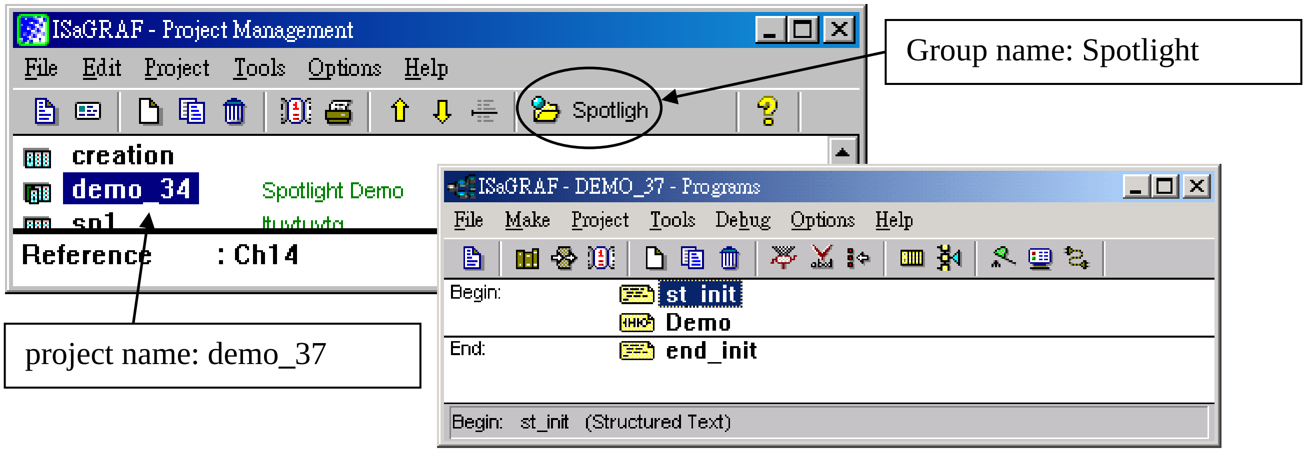
Variables used In the example:

Name	Type	Attribute	Description
INIT	Boolean	**Internal**	Only = TRUE at the 1st scan cycle, **INIT value is TRUE**
L1	Boolean	**Output**	Output 1, connect to Ch1 of "show3led"
L2	Boolean	**Output**	Output 2, connect to Ch2 of "show3led"
L3	Boolean	**Output**	Output 3, connect to Ch3 of "show3led"
Button1	Boolean	**Inpput**	Input 1, connect to Ch1 of "push4key"
Button2	Boolean	**Inpput**	Input 2, connect to Ch2 of "push4key"
Button3	Boolean	**Inpput**	Input 3, connect to Ch3 of "push4key"
Button4	Boolean	**Inpput**	Input 4, connect to Ch4 of "push4key"
VAL_OUT	Integer	**Internal**	to set blinking period, **initial value is set at 500 (unit:ms)**
OLD_VAL_OUT	Integer	**Internal**	Old value of VAL_OUT
T1	Timer	**Internal**	Time Period of blinking
MSG1	Message	**Internal**	Status report, please set its Maxinum Length to 48

HMI screen outline:



Project architecture:



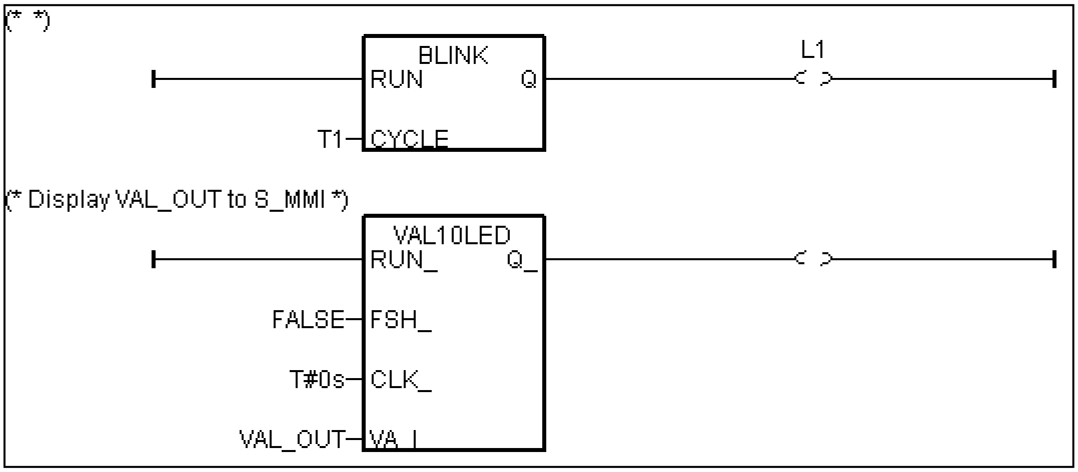
ST Program “st_init” in the “Begin” area :

```
(* Do some init action *)
if   INIT=TRUE   then
  T1 := TMR(VAL_OUT) ;   (* Convert integer:VAL_OUT to Timer:T1 in ms *)
  MSG1:='OK.' ;
  OLD_VAL_OUT := VAL_OUT ;  (* init OLD value *)
end_if ;

(* if set a new value to VAL_OUT *)
if  VAL_OUT <> OLD_VAL_OUT  then

  (* VAL_OUT is acceptable *)
  if  (VAL_OUT>=200) & (VAL_OUT<=5000)  then
    T1 := TMR(VAL_OUT) ;   (* Convert integer:VAL_OUT to Timer:T1 in ms *)
    MSG1 := 'OK.' ;
  else     (* VAL_OUT out of range *)
    MSG1 := 'VAL_OUT should be between 200 and 5000 .' ;
  end_if ;

  OLD_VAL_OUT  :=  VAL_OUT ;  (* update OLD value *)

end_if ;
```

ST Program “end_init” in the “End” area :

```
INIT := FALSE ;
```

LD Program "Demo" in the "Begin" area:



Operations :

- The status of four push buttons will be displayed on the HMI screen
- The first output will be blinking with the period defined by "VAL_OUT" in ms
- Value of "VAL_OUT" can be modified from the HMI screen and displayed on the front panel of the controller.
- The second and third output "L2" & "L3" can be controlled by the HMI screen.

Steps to build a Spotlight: HMI screen:

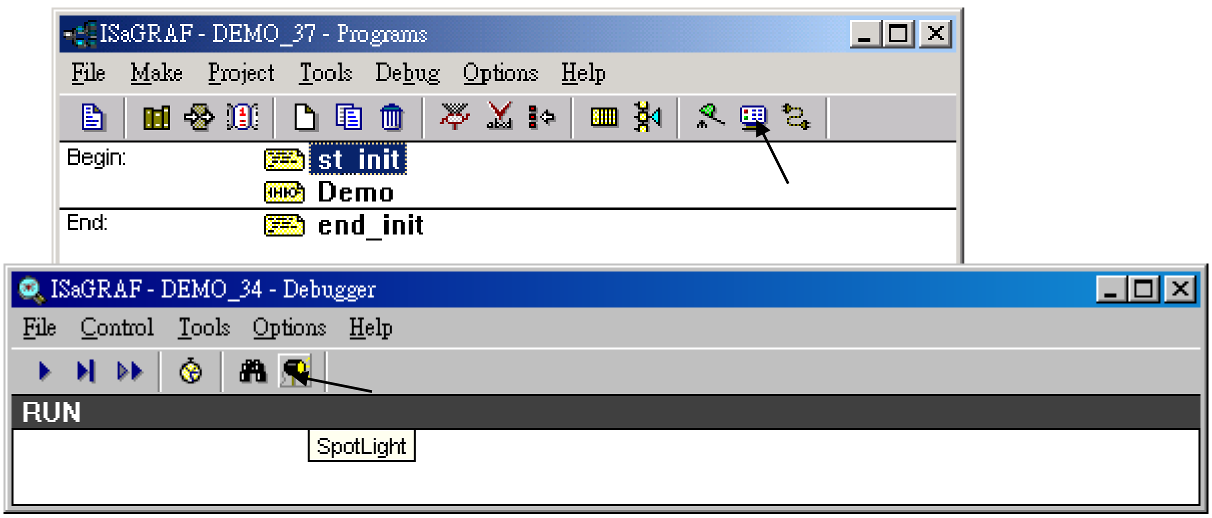
A. Complete this Demo project as described above. After you finish it. Compile it to make sure there is no error.

B. Copy all files inside “ICO” folder to the associate directory of your project.
The “ICO” folder contains some boolean icon files already bulit by ICP DAS. They can be found from the I-8000 CD-ROM : \napdos\isagraf\ICO\

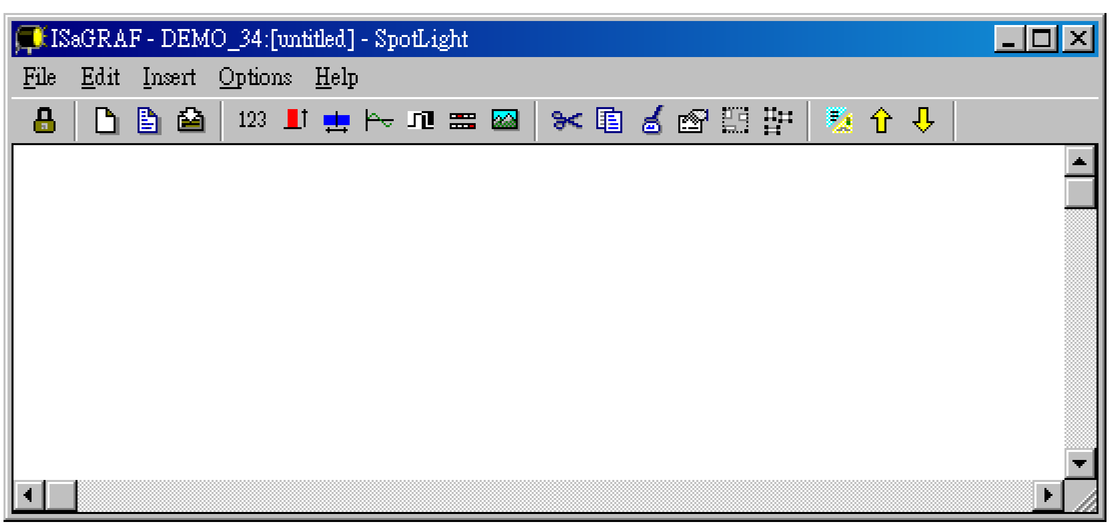
For example, this demo project is inside group “spotligh” and the project name is “demo_37”, then copy CD-ROM: \napdos\isagraf\ICO*.* to c:\isawin\spotligh\demo_37\

If the “ICO” folder is not found in your CD-ROM. Please download it from the below site.
ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/

C. Get into the Spotlight editor.
Click on “Simulate”, then click on “Spotlight” to open spotlight editor.

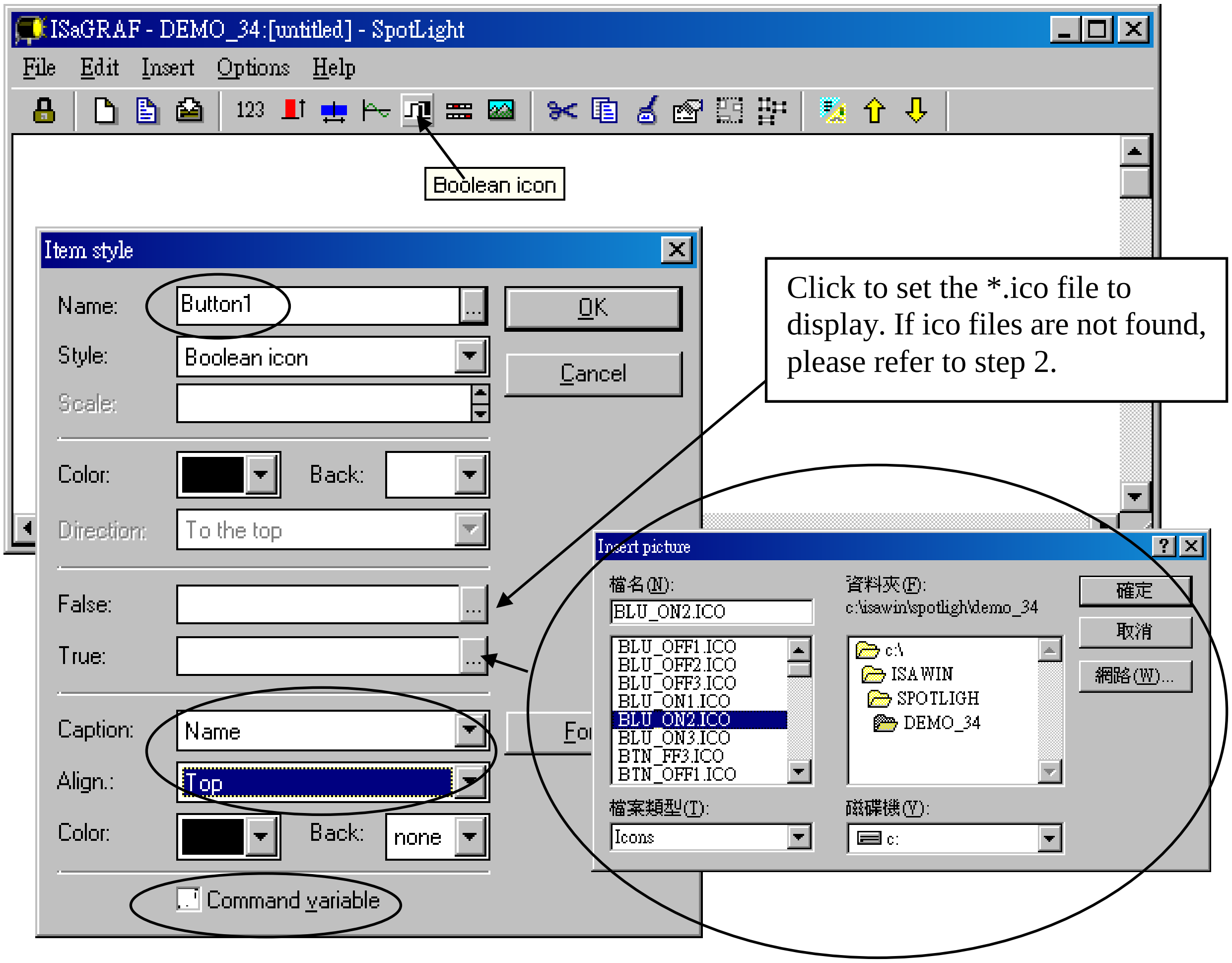


A “SpotLight” window will appear as below.

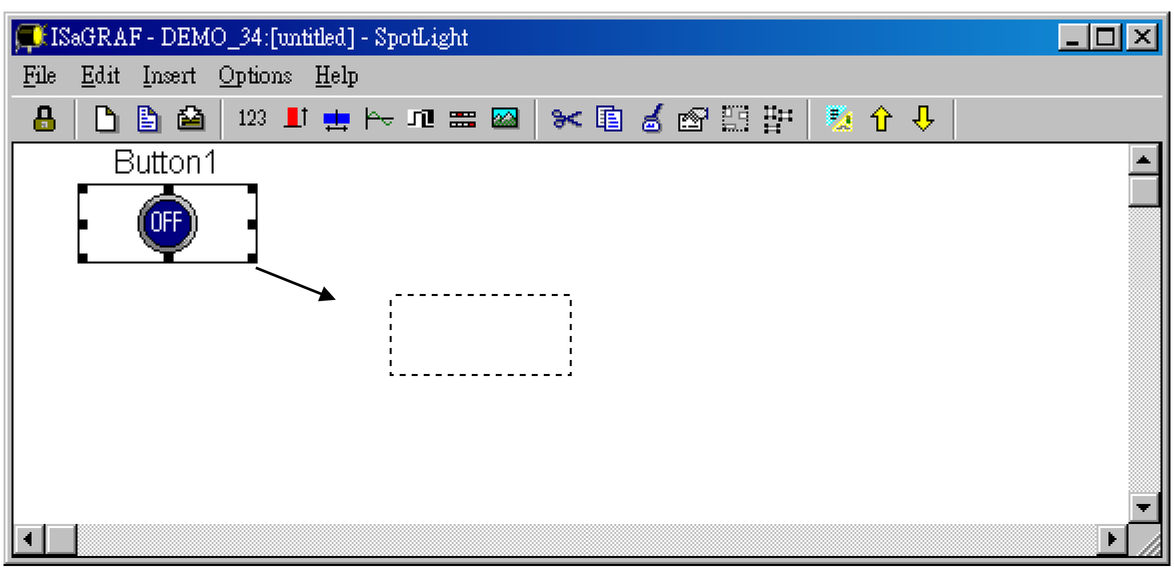


D. Add “boolean Icons”

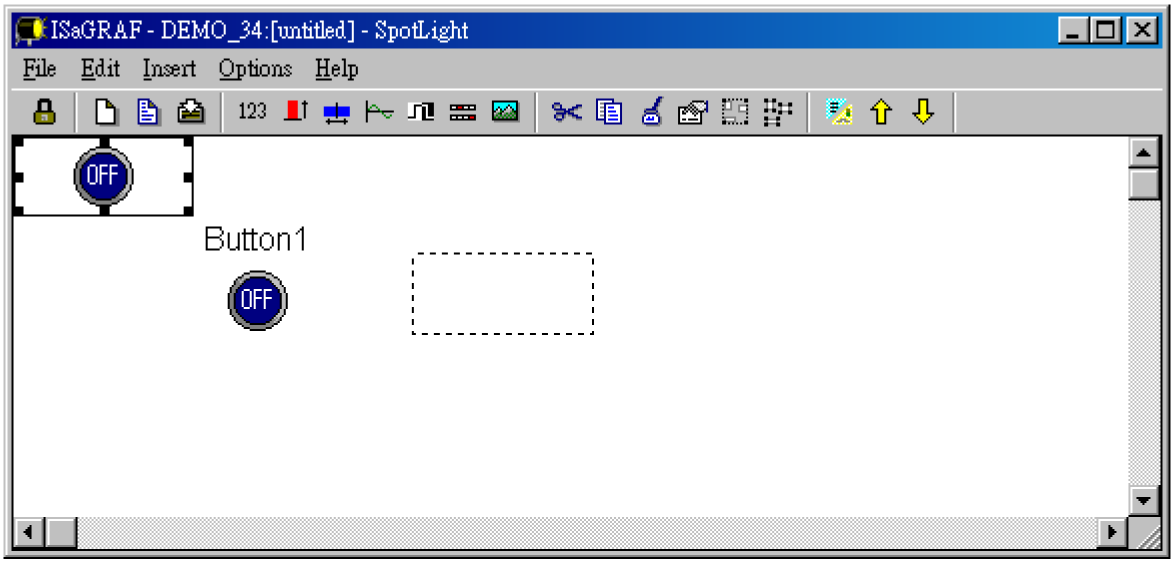
Click on “Boolean icon”, then set the associated Name as “Button1”, Caption as “Name”, Align as “Top” and then set the prefered *.ico file to display with “FALSE” and “TRUE”, and un-check “Command variable”.



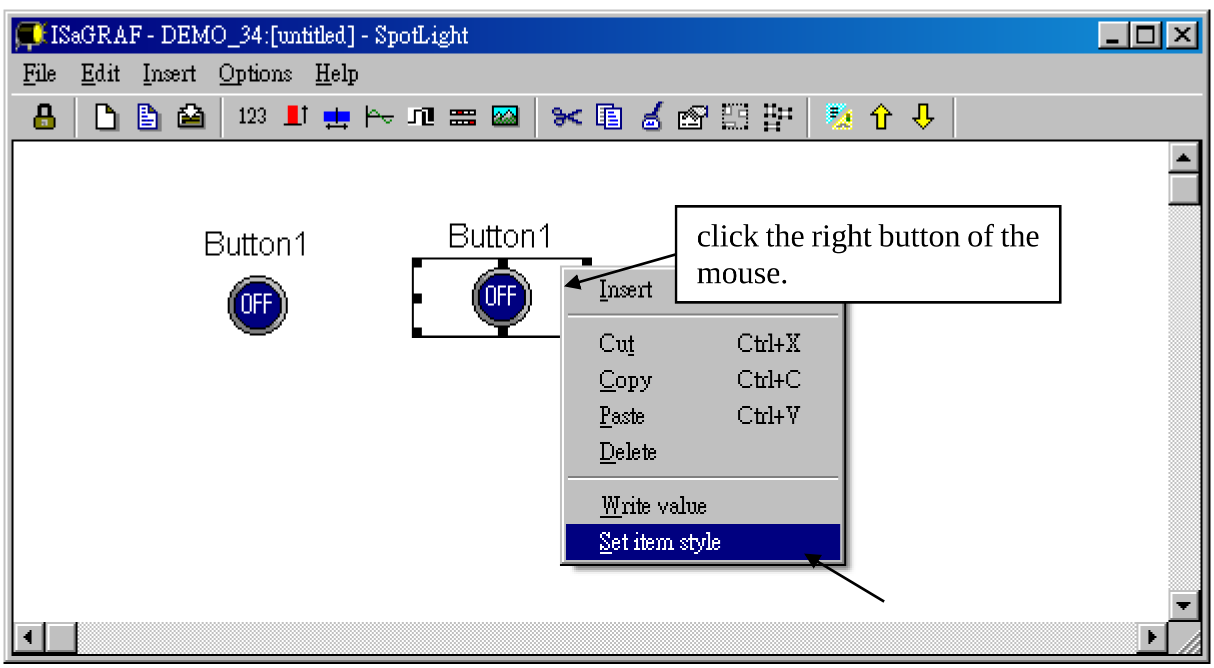
Then drag the boolean icon to appropriate place.



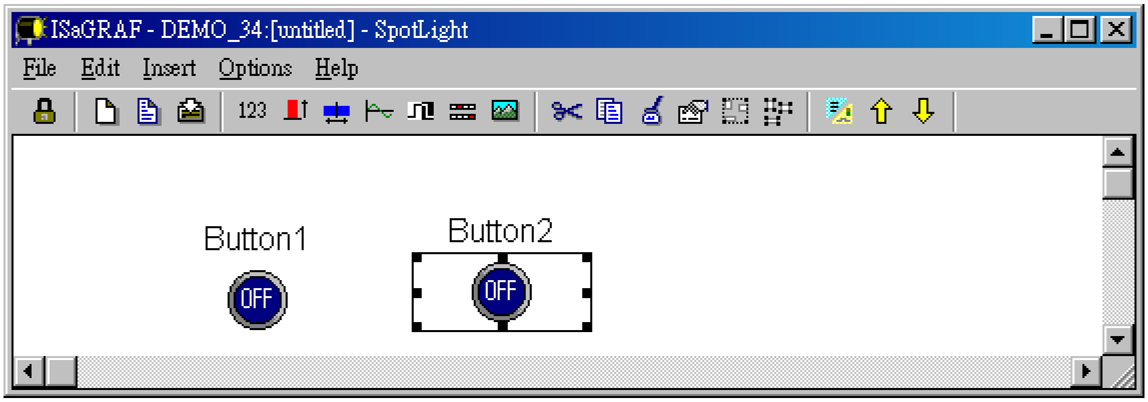
Check on the new created boolean icon, copy it(Ctrl+c) and then paste it (Ctrl+v) to reproduce one another boolean icon. Then drag it to the prefered place.



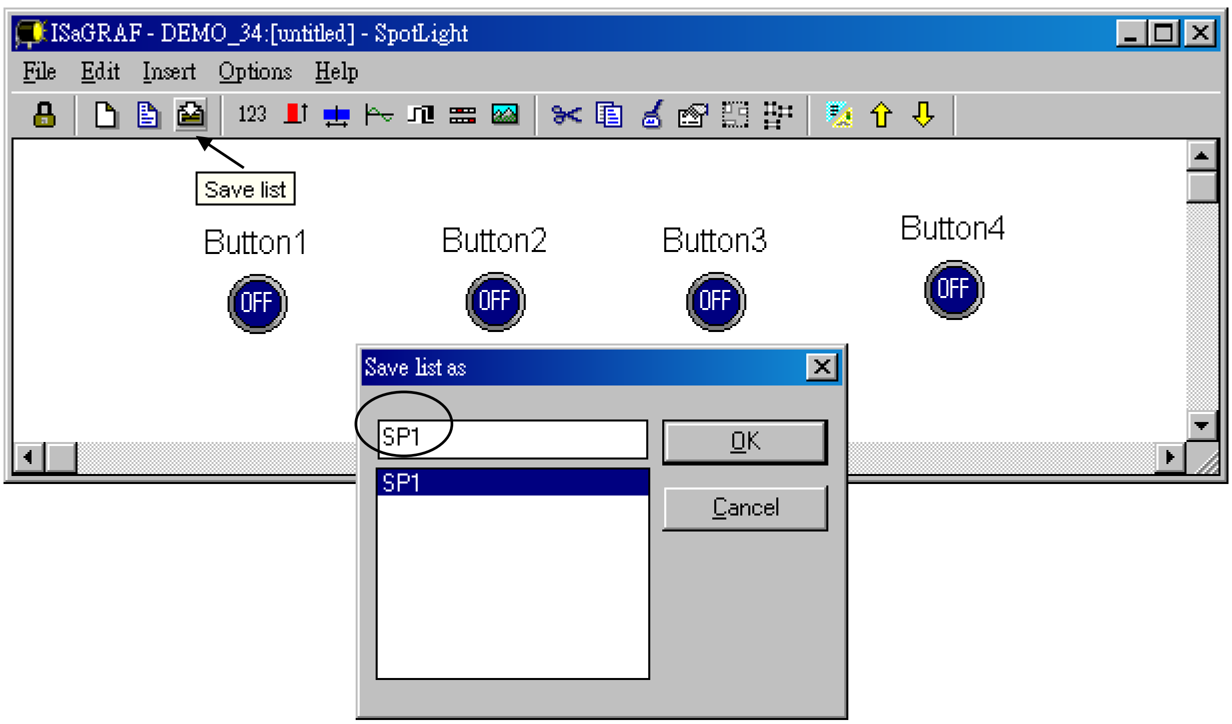
Check on the new created boolean icon, then click the right button of the mouse, select "Set item style" to modify the name to "Button2".



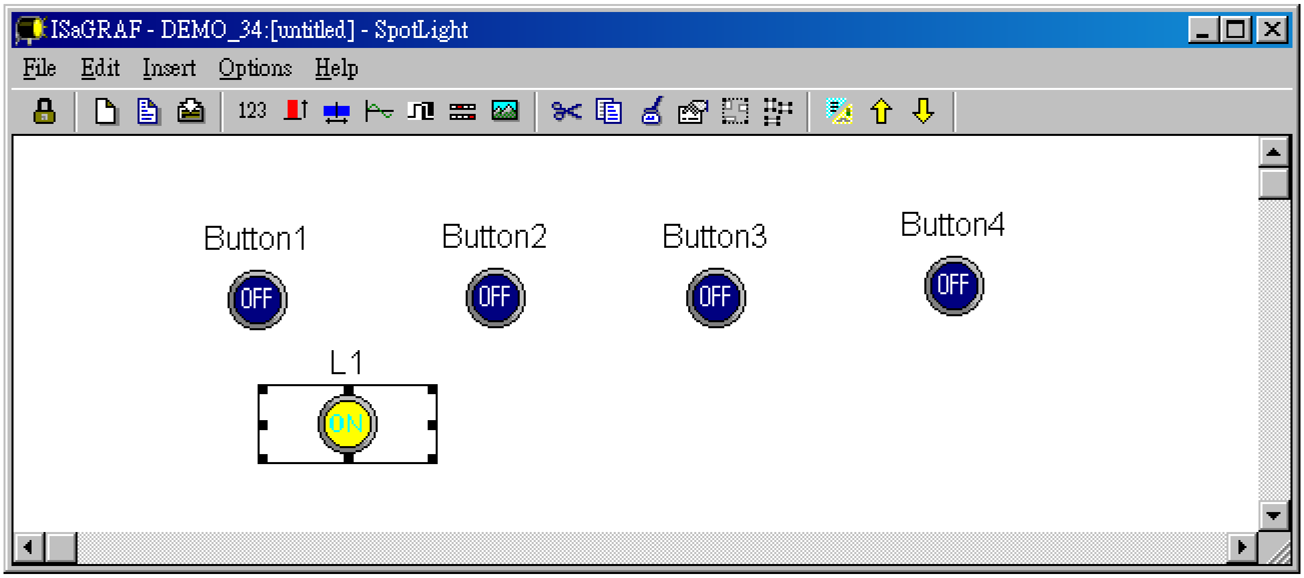
Then we have …



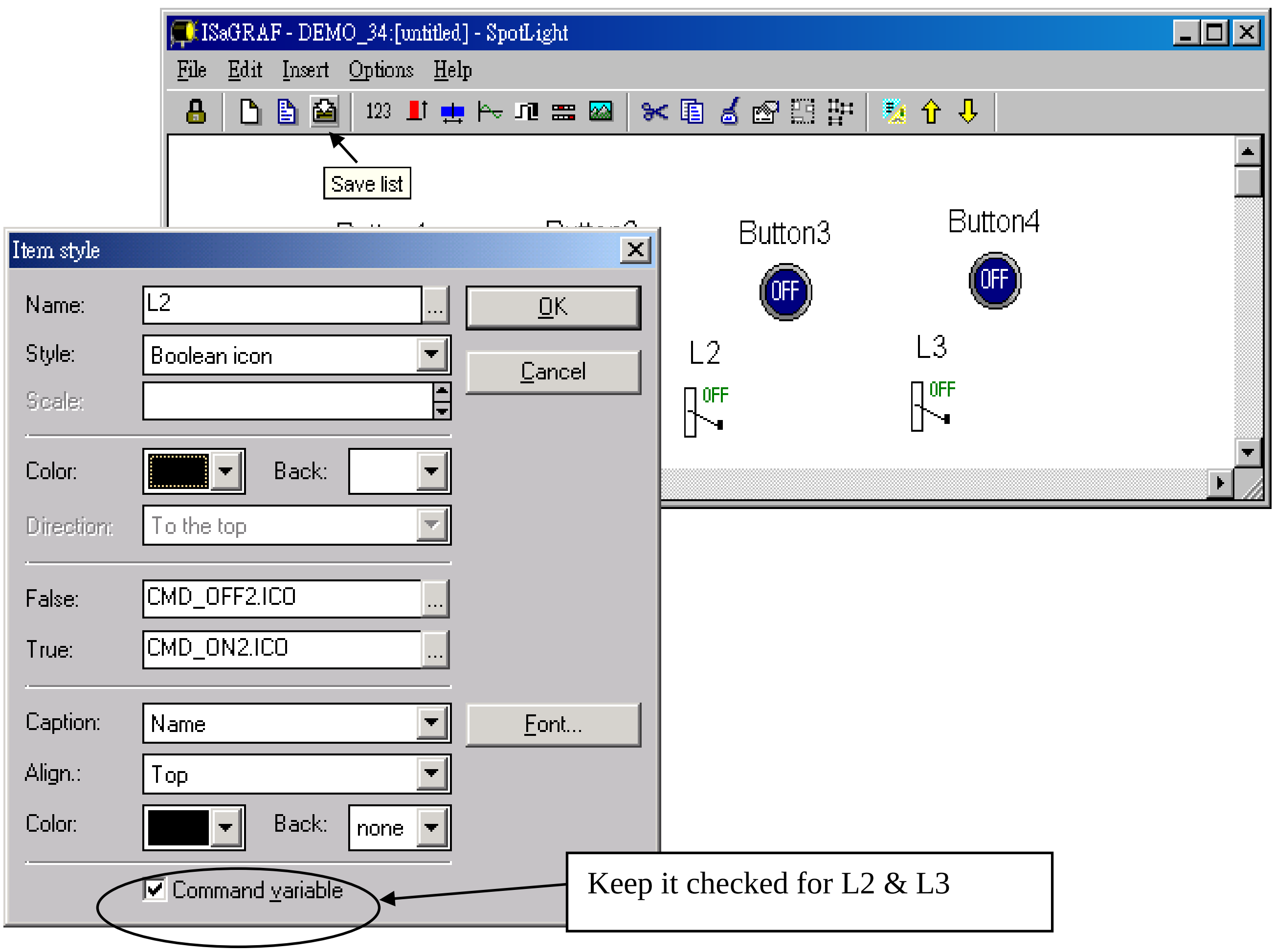
Follow the same method to create 4 boolean icons as below. Recommand to save it anytime for safety. Given a name to this screen.



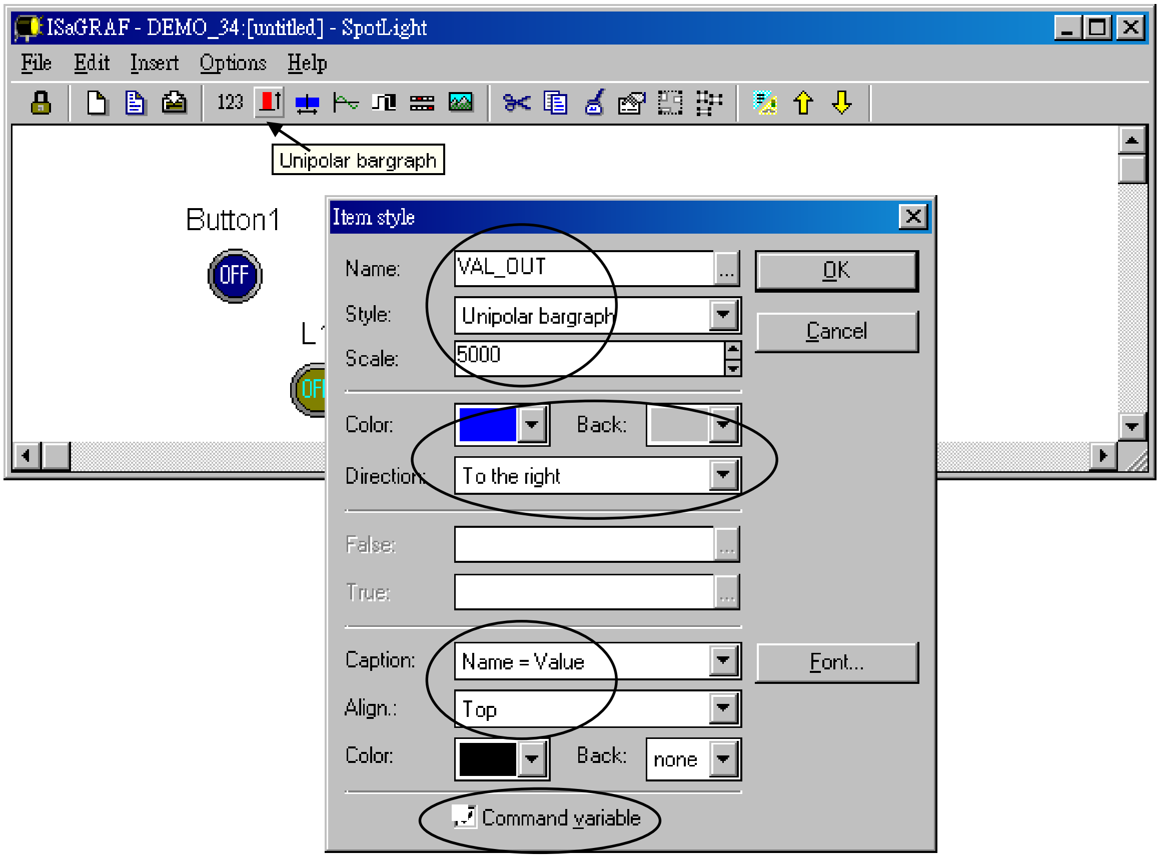
We need one another Boolean icon to display the status of “L1”. Create it with a different color (TRUE : “YEL_ON2.ico” , FALSE : “YEL_OFF2.ico”).



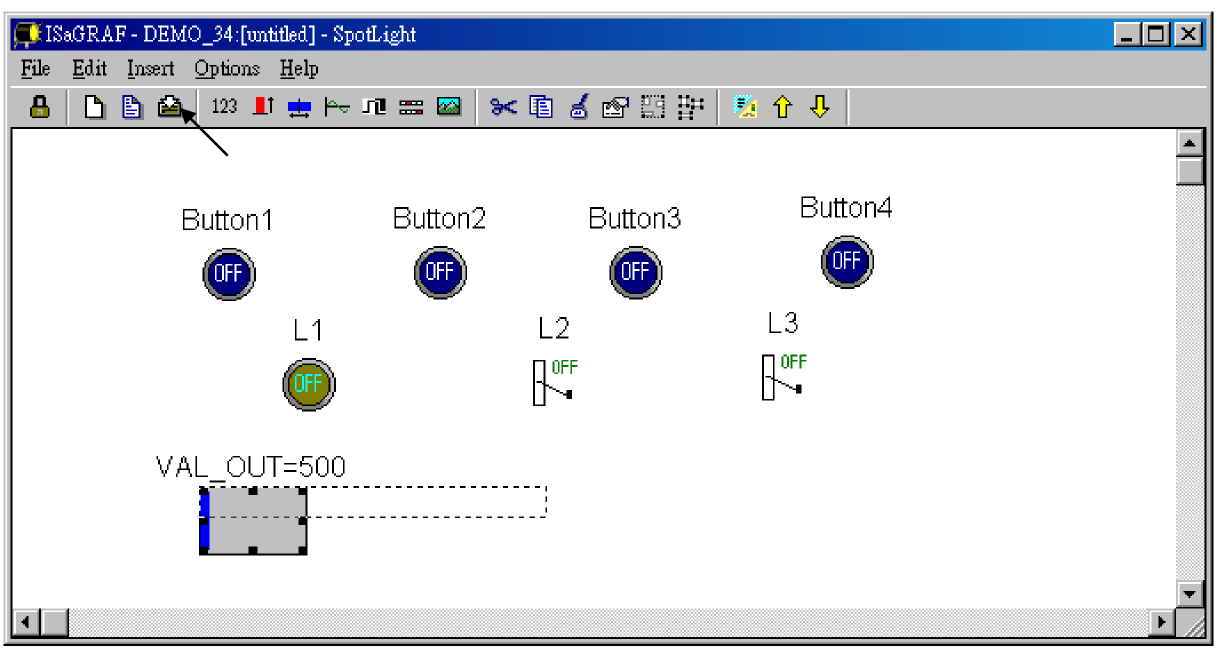
And then create L2 & L3 with TRUE:”CMD_ON2.ico” and FLASE: “CMD_OFF2.ico” as below. Save it anytime, **L2 & L3 should not un-check “Command variable”**.



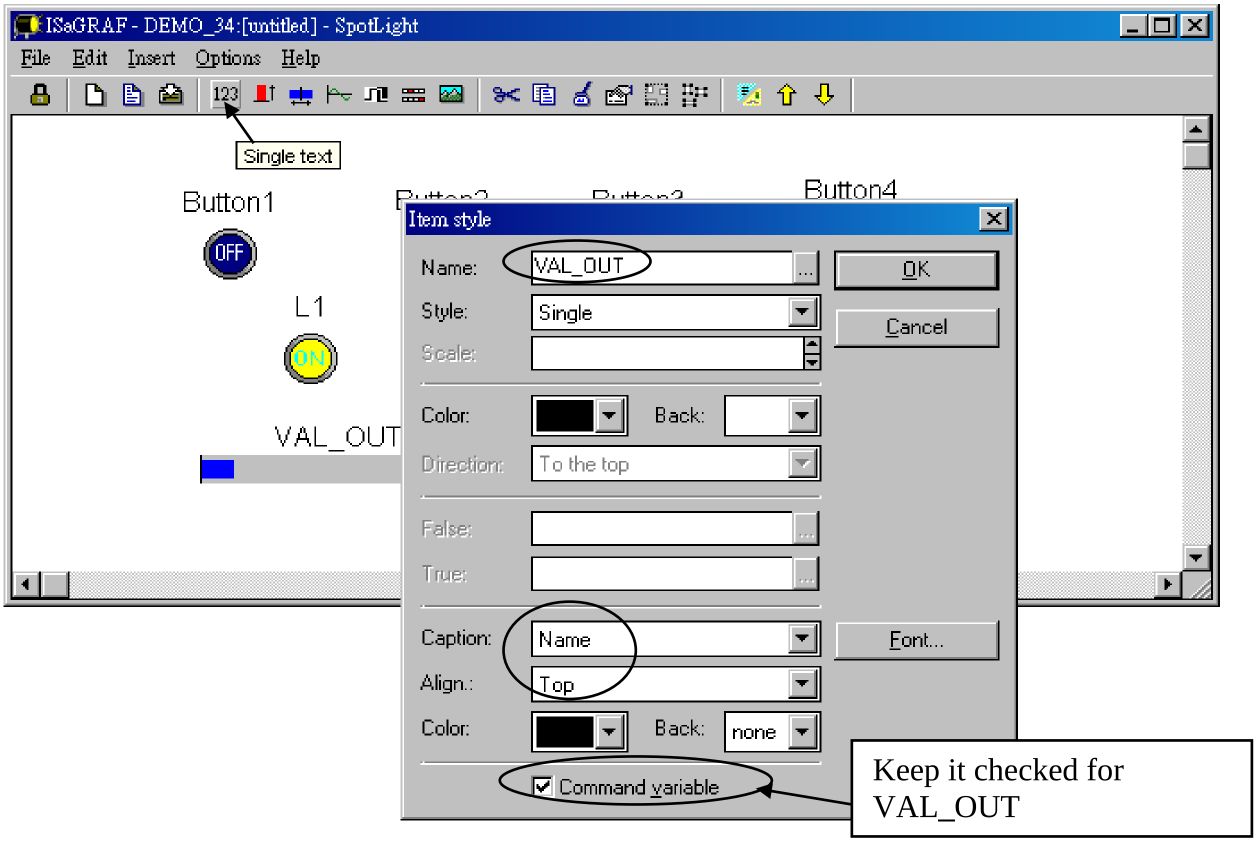
E. Add “Unipolar bargraph”
   Click on “Unipolar bargraph”, set the associated Name as “VAL_OUT”, Scale as “5000”, Color as blue, Back as gray, Direction as “To the right”, Caption as “Name=Value”, Align as “Top”, and un-check “Command variable”



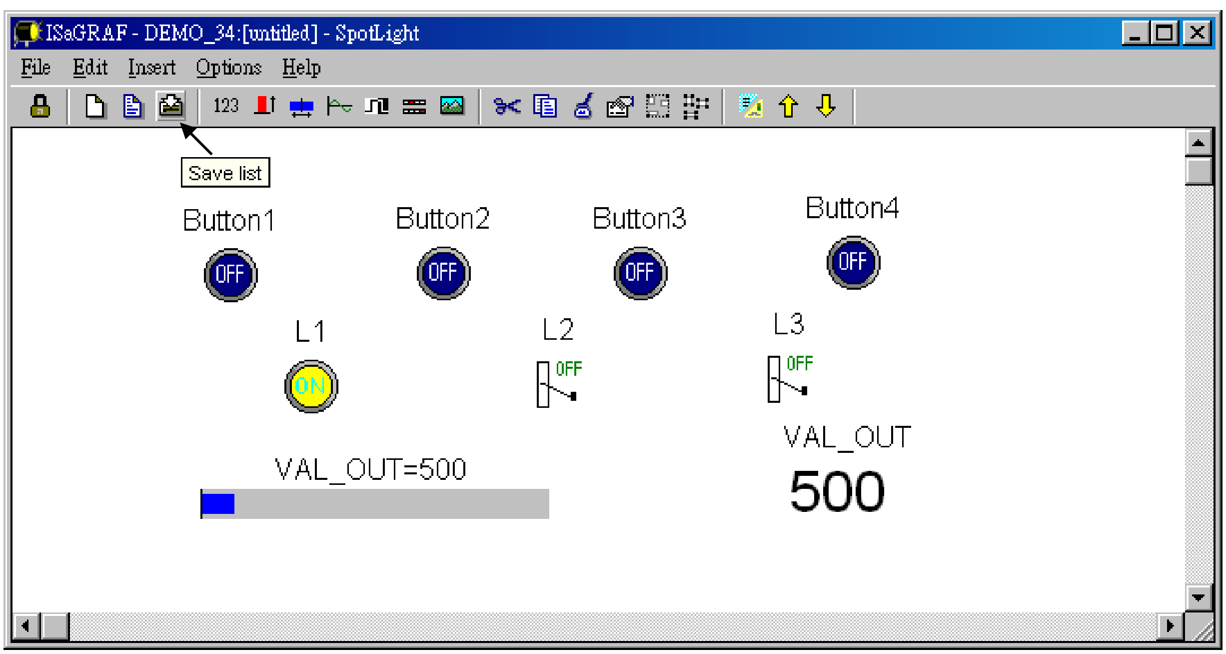
Click and hold on the left button of the mouse to change to the prefered shape as below. Save it anytime.



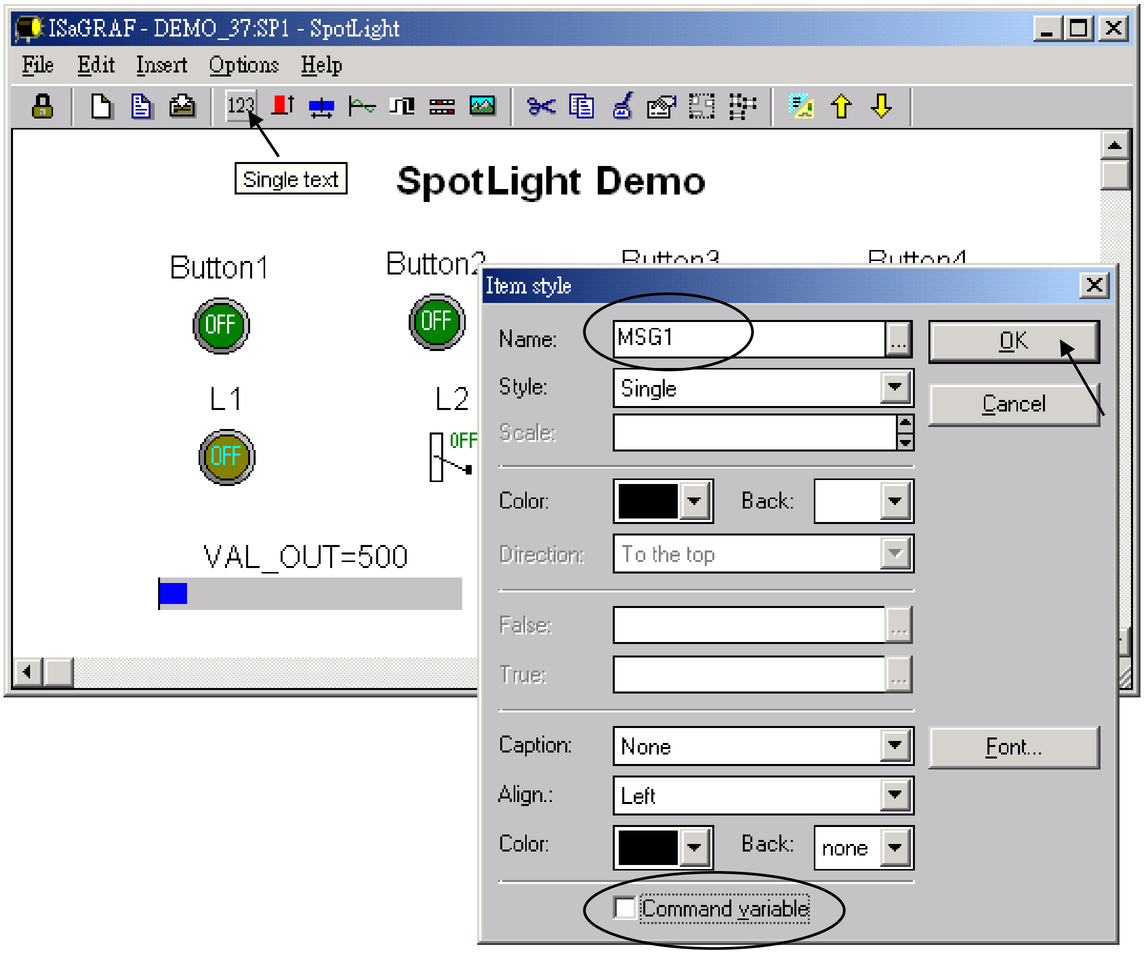
F. Add “Single text”
Click on “Single text”, set the associated Name as “VAL_OUT”, Caption as “Name”, Align as “Top”



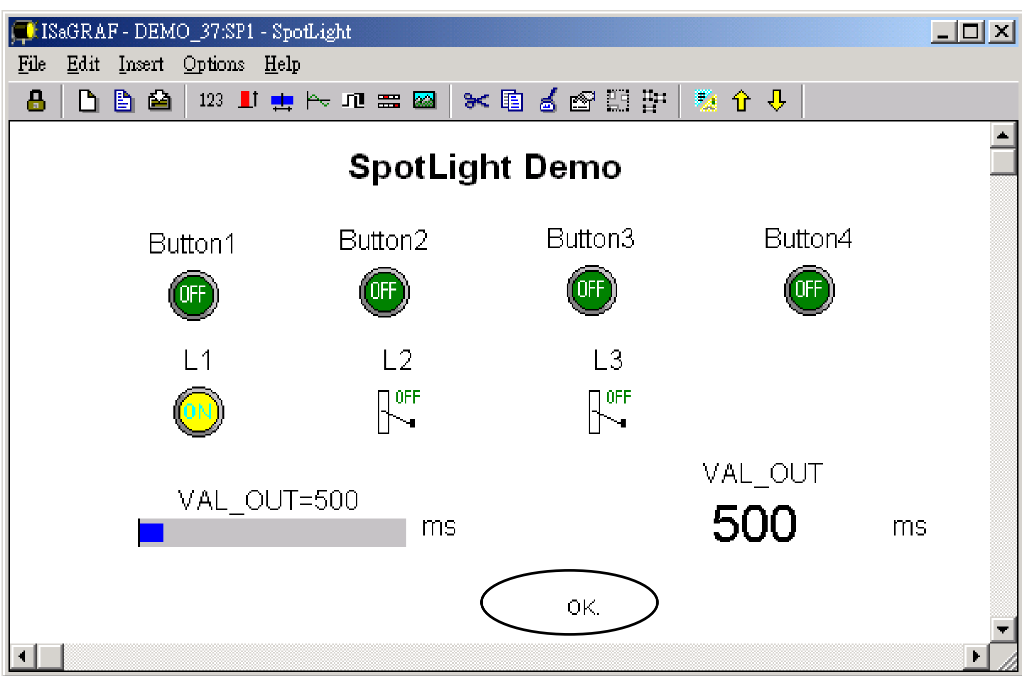
Move it to the prefered place and save it.



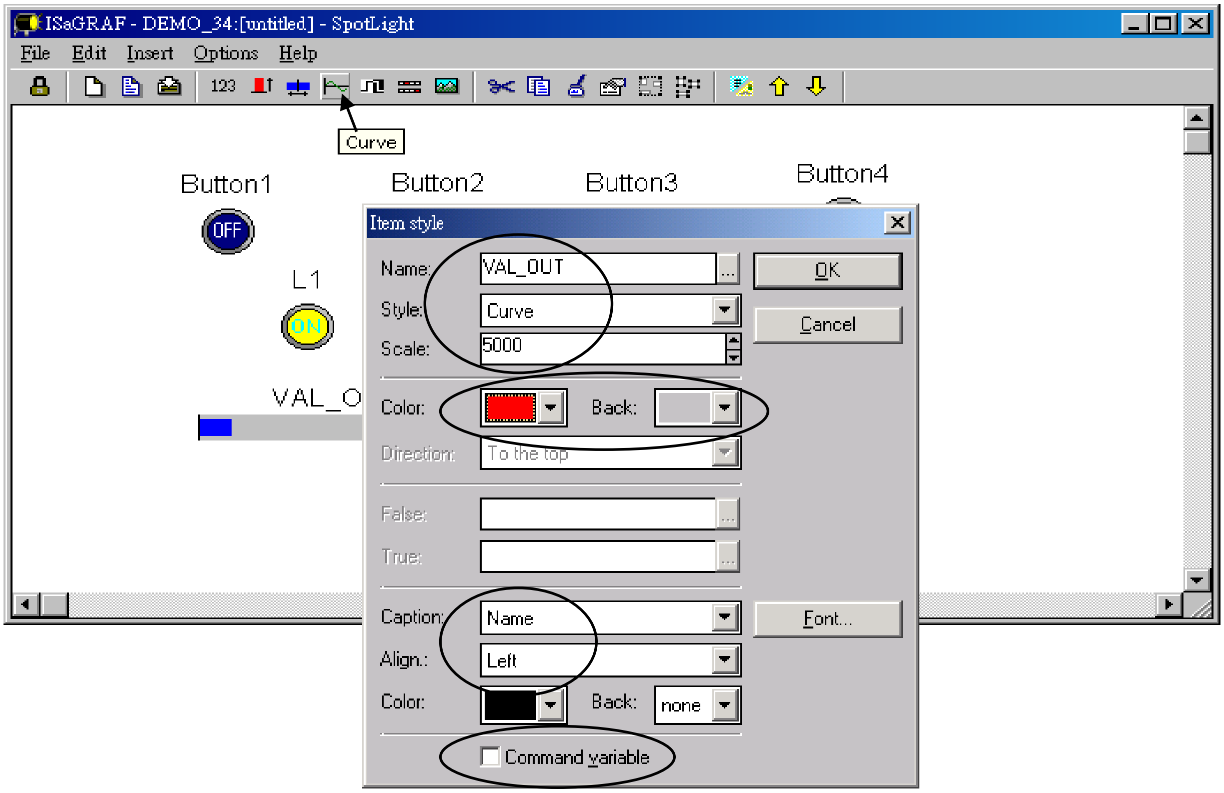
Click on “Single text” again, set the associated Name as “MSG1”, Caption as “None”, Align as “Left” and un-check “Command variable”.



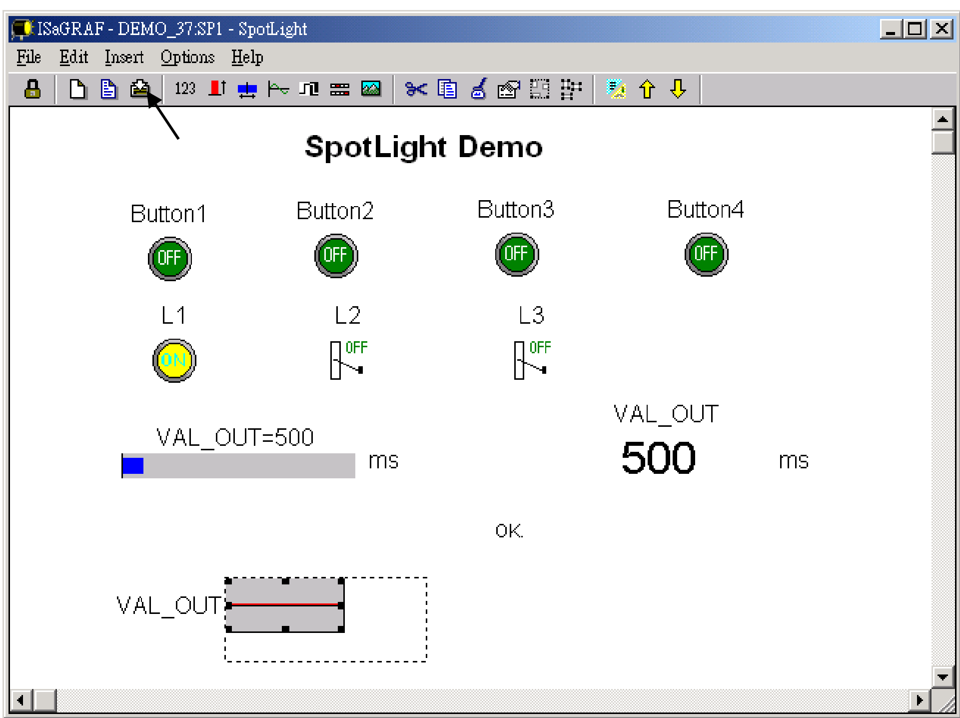
Move it to the prefered place and save it.



G. Add “Curve”
Click on “Curve”, set the associated Name as “VAL_OUT”, Scale as “5000”, Color as red, Back as gray, Caption as “Name”, Align as “Top”, and un-check “Command variable”



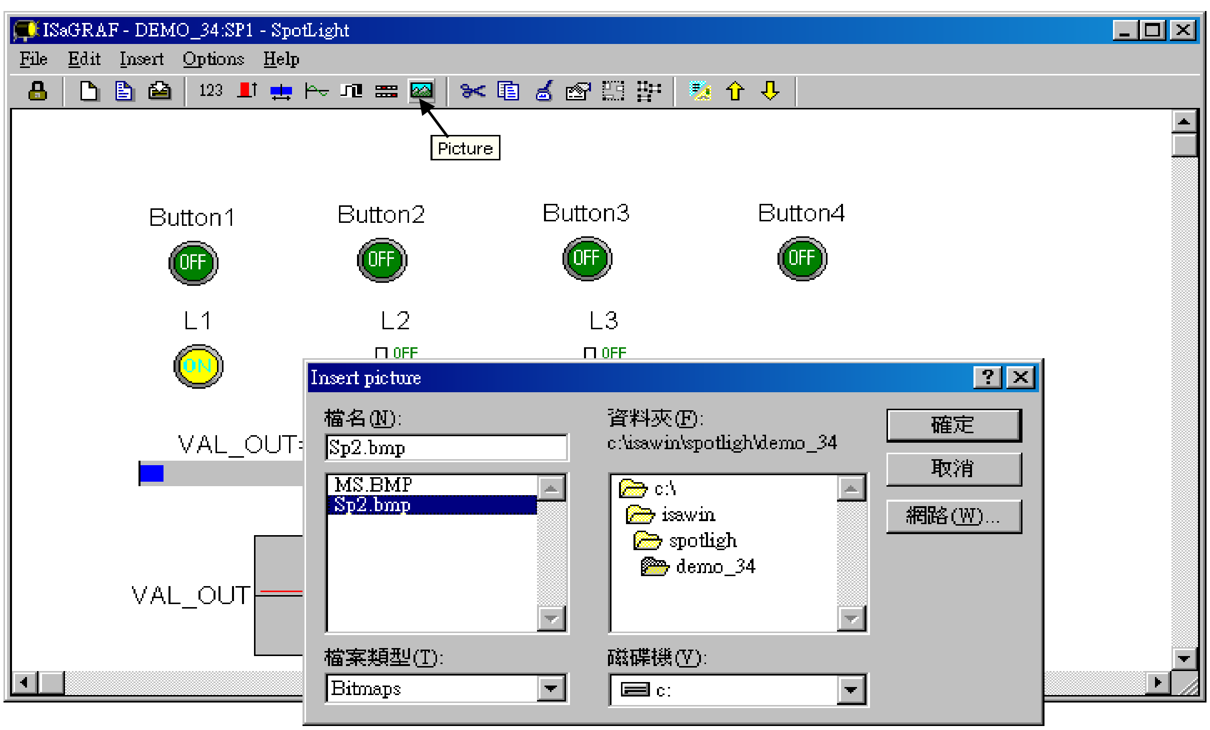
Click and hold on the left button of the mouse to change to the prefered shape as below. Save it anytime



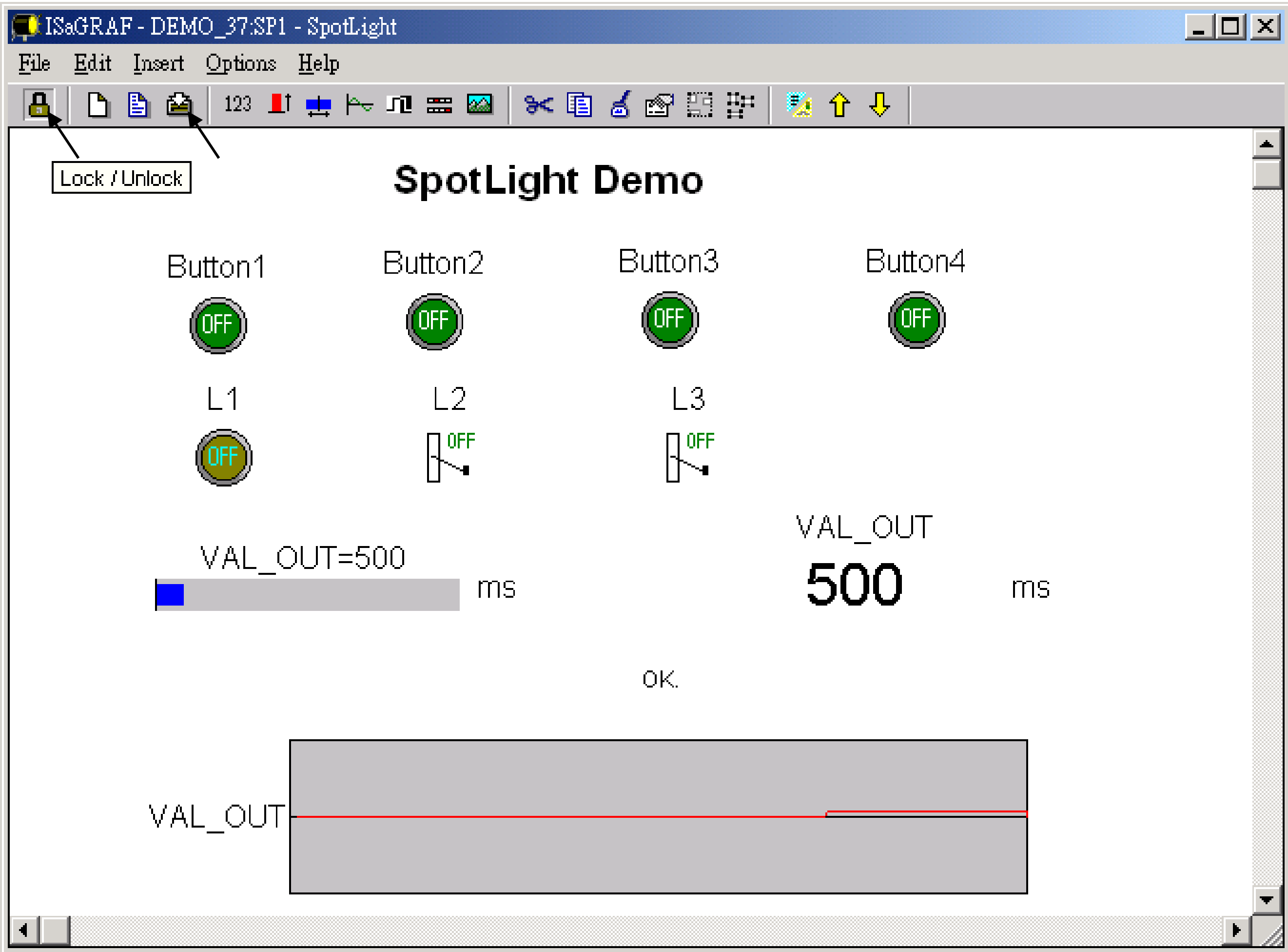
H. Add “picture”

Please build 2 bitmap pictures by MS painter as below. Then save them respectively with file names of “sp2.bmp” & “ms.bmp” to the associate project directory. (For this example “c:\isawin\spotligh\demo_37\”)

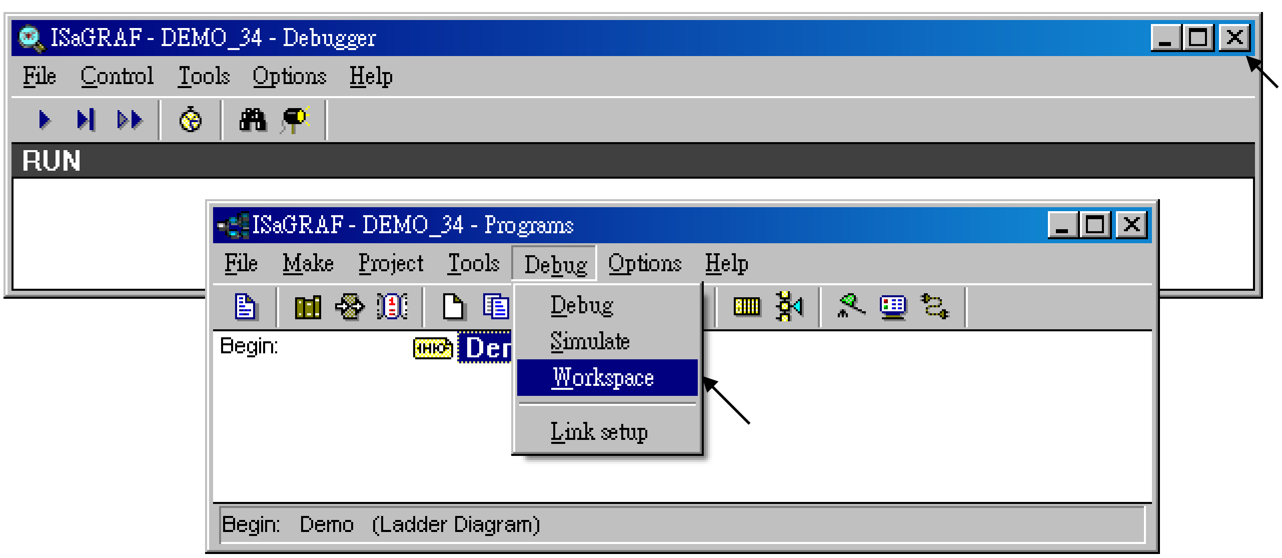
Click on “Picture”, Select the associate bmp file name.



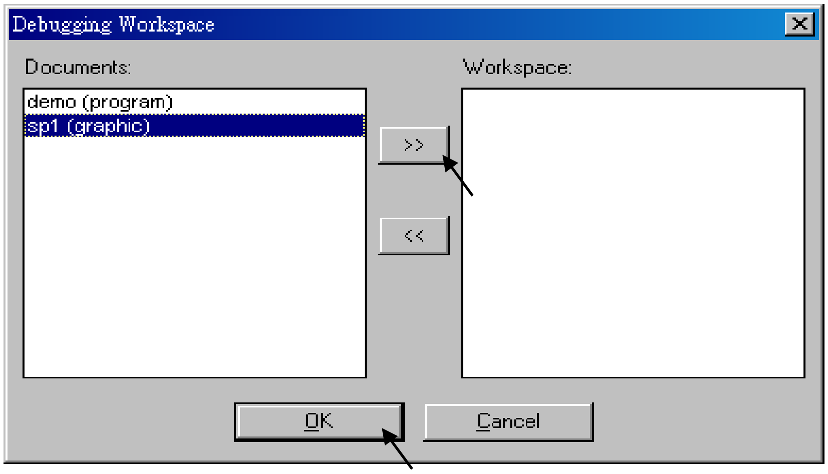
Add 2 pictures “sp2.bmp” and “ms.bmp” to the prefered place, then we got the below window. Click on “Lock” to protect it (No modification allowed). Save it anytime.



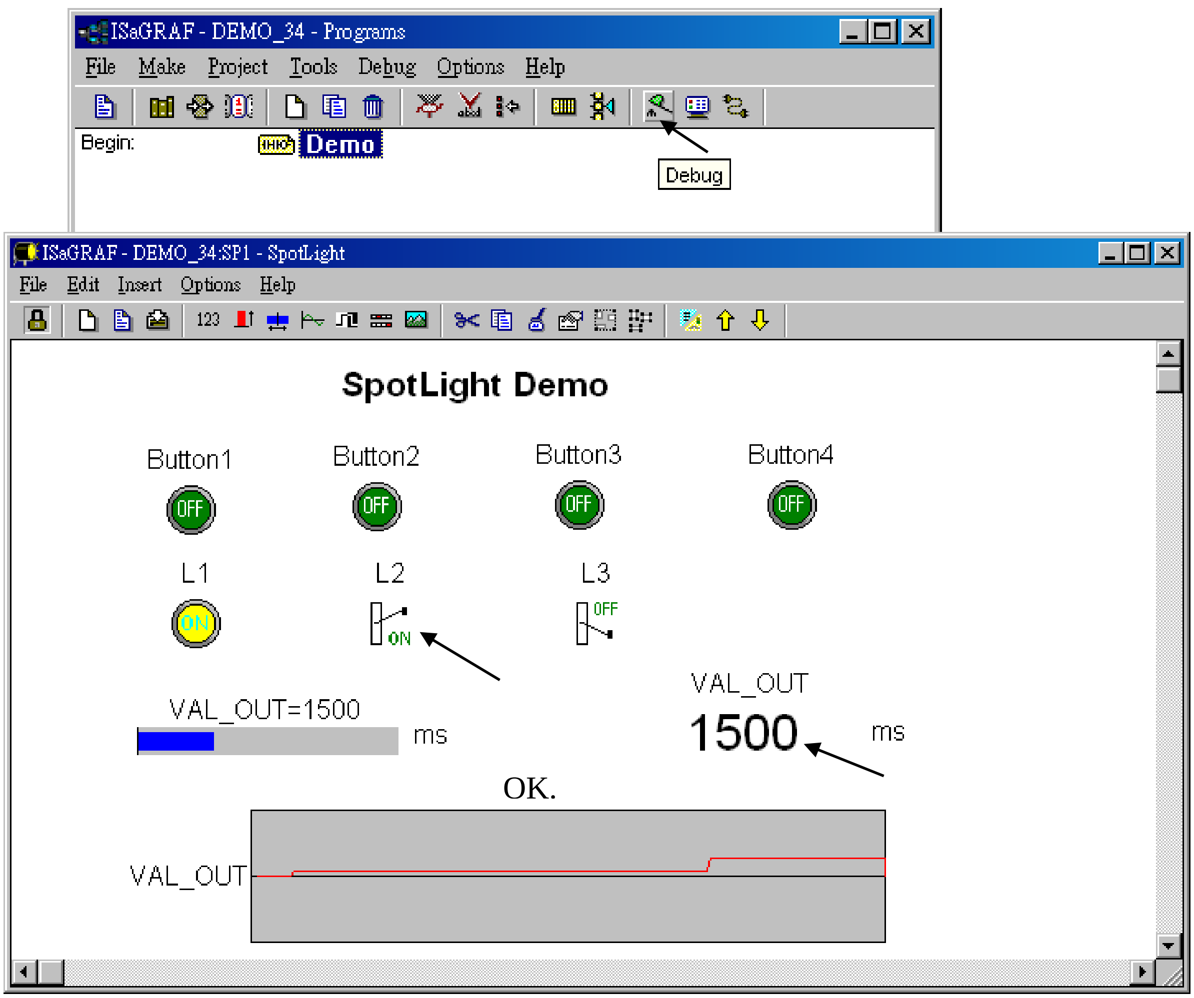
I. Add the HMI screen to the “Workspace”
Quit “simulation”, then run “Debug”-“Workspace”.



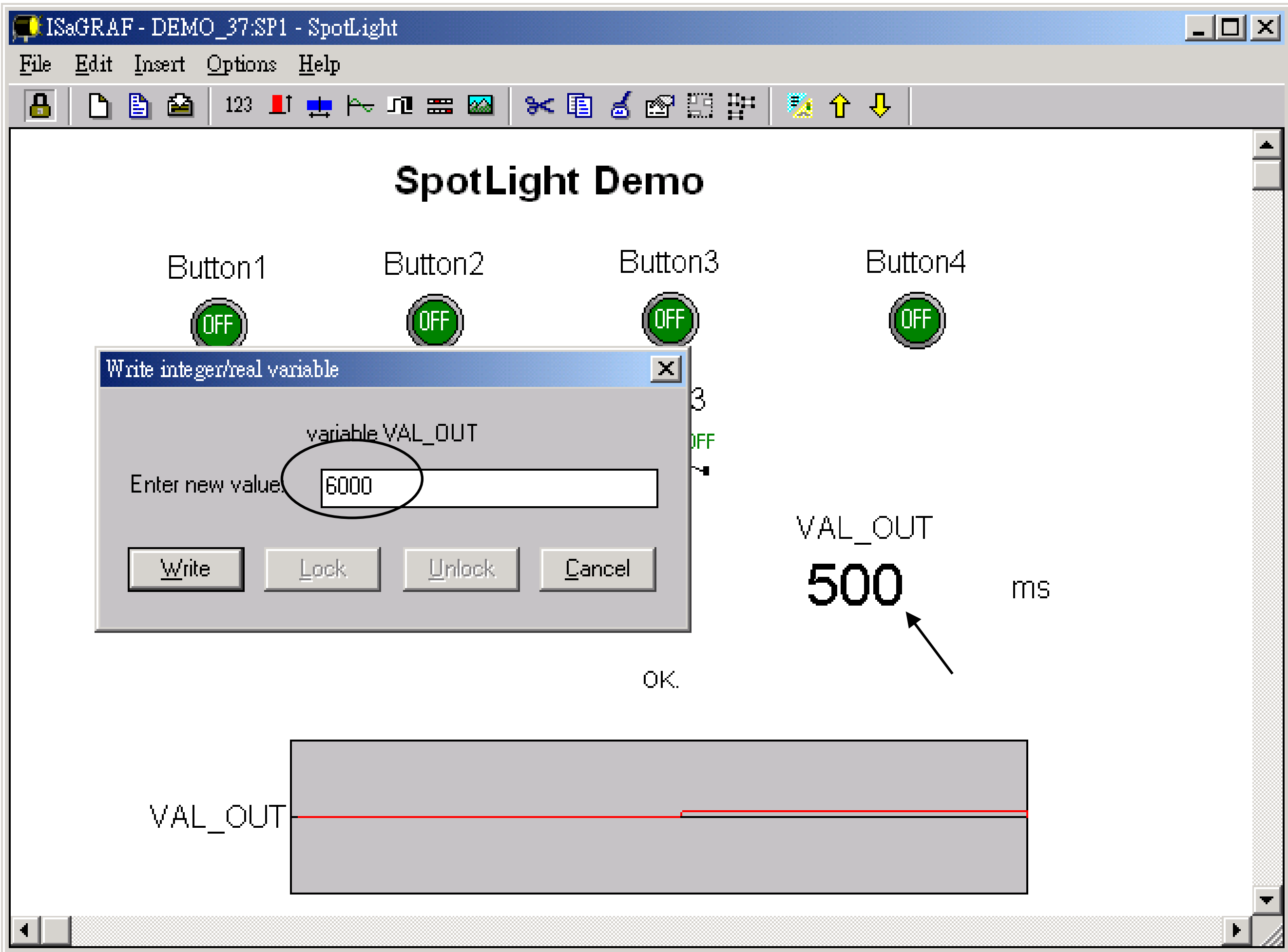
Move the HMI screen to the right (Workspace).



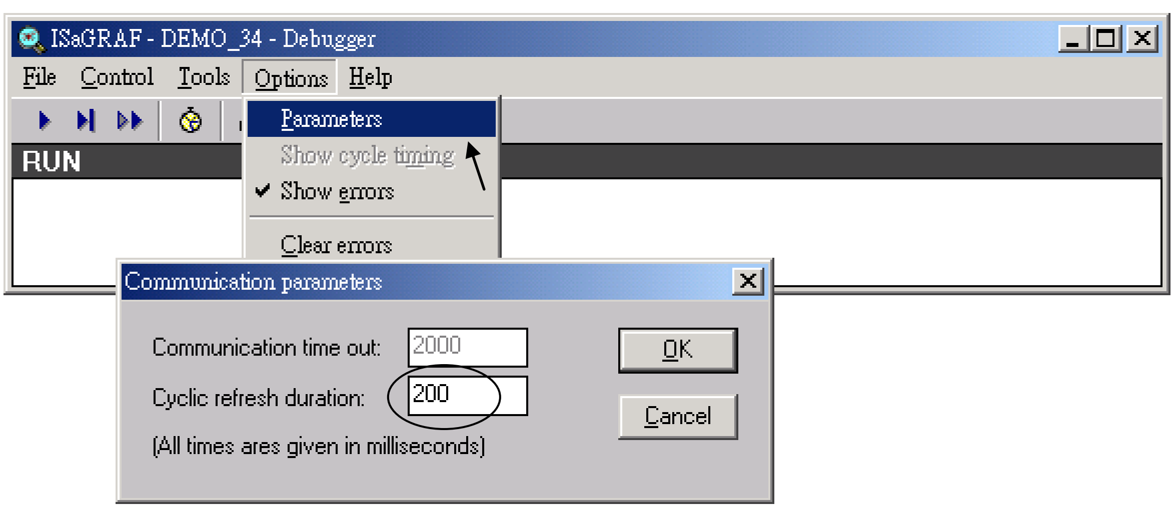
J. Time to download to the controller and test
Click on "Debug" to download the project to the controller and test it. You may double click on "L2", "L3" or "VAL_OUT" to modify the value and see what it happens on the controller. And also you can press the 4 pushbuttons on the controller.



You may double click on "VAL_OUT" and give a value large than 5000 to see what it happens.



Note: For quick response, user may click on "Options" – "Parameters", and then set the "Cyclic refresh duration to a smaller value. (Recommand not to set below 200 ms)



Chapter 15. Creating User-Defined Functions

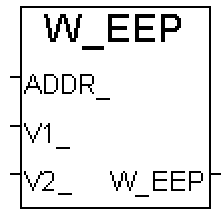
ISaGRAF supoorts functions written in ST, FBD, IL and QLD languages. User-defined functions are normally for some algorithm which been used again and again. A function always has an return value (outp ut parameter) and its name should be the same name as the function, and may have up to 31 input parameters. The code written inside functions can not call any function block, however can call other ISaGRAF standard functions and c functions provided by ICP DAS.

We are going to creating a function to save an integer value to the EEPROM. Its format is as the below.

Function name : W_EEP
Description: Save an integer to the EEPROM when its value changed

Input parameters:

- ADDR_ (integer) : the address of the EEPROM to write
- V1_ (integer) : New value
- V2_ (integer) : Old value



Return parameter:

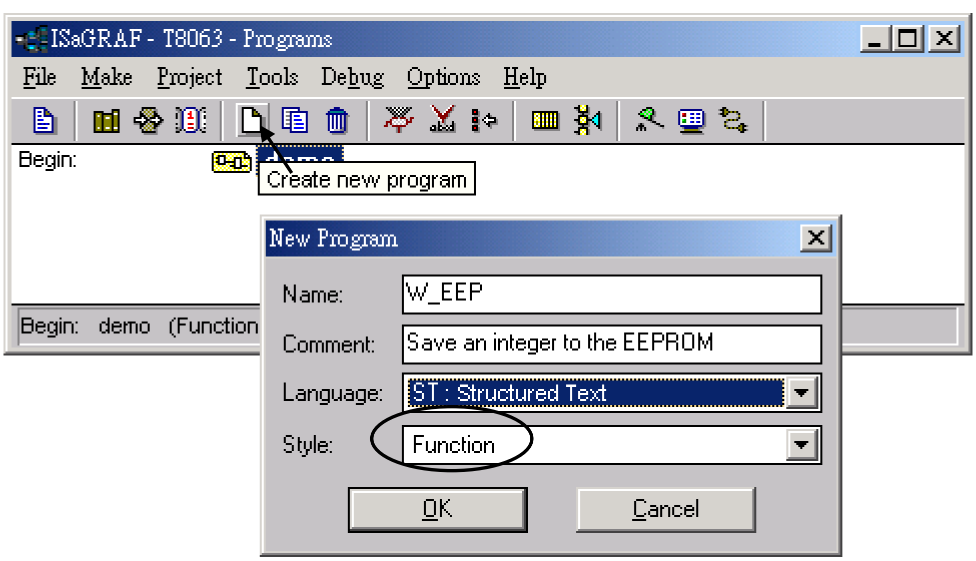
- W_EEP (integer): return the new value

Note: The parameter names been used will become reserved names. That's why we use ADDR_ , V1_ , V2_ rather than ADDR , V1 & V2.

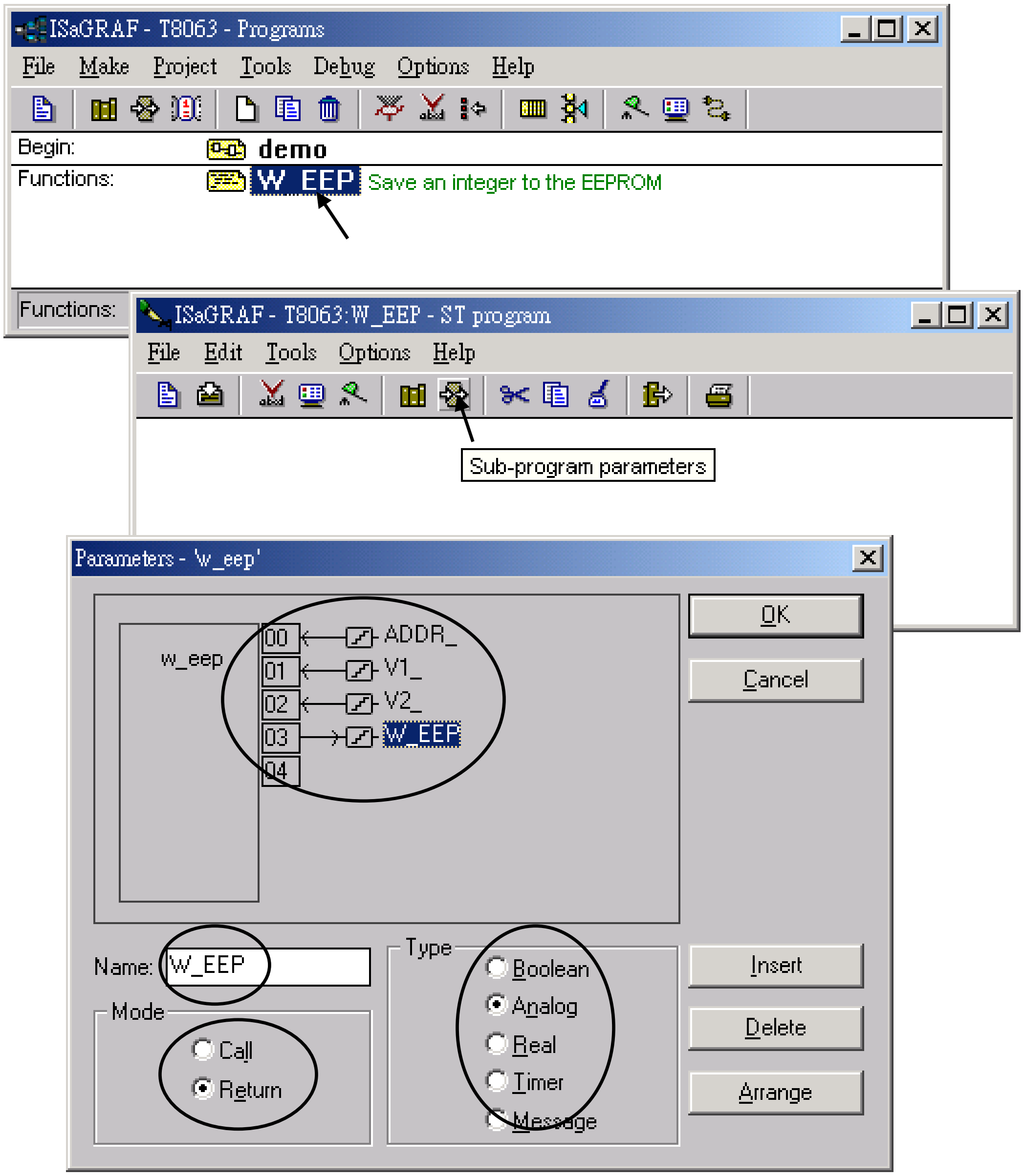
15.1: Creating functions inside one project

Functions created inside one project can be only called by other programs written in the same project.

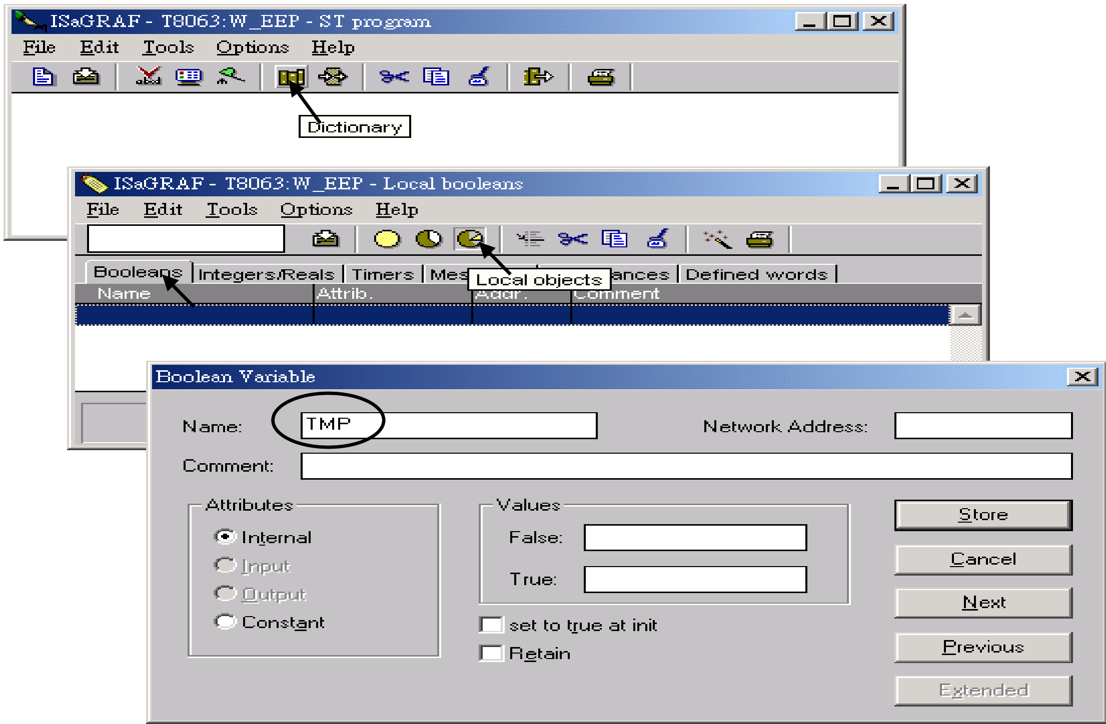
A. Click on "Create new program" inside the project. Given Name as "W_EEP", Language as "ST:…", Style as "Function".



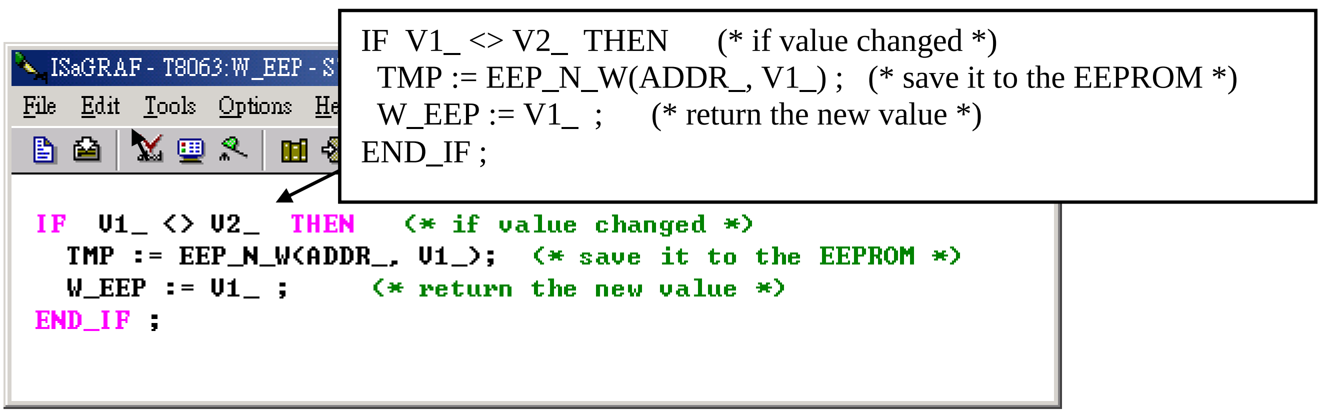
B. Double click on the function to get into it. Then click on “Sub-program parameters” to define input and output parameters.



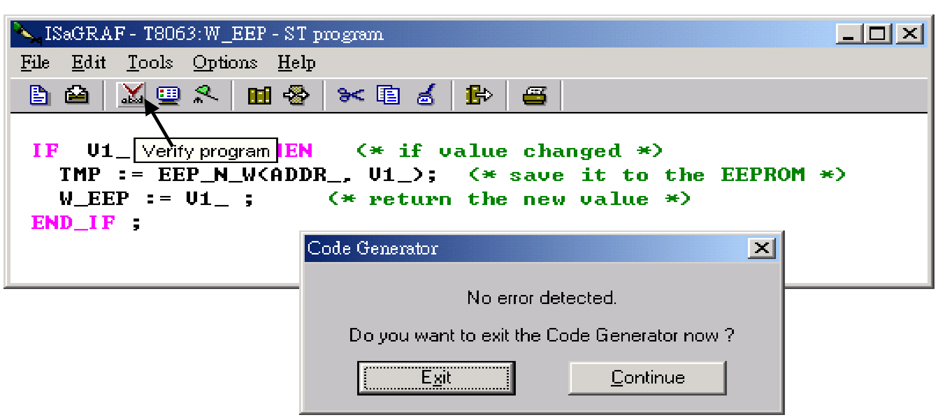
C. Declare local variables. We need a local **boolean internal** variable "**TMP**" in this example.



D. Enter function codes.



E. Verify the function.

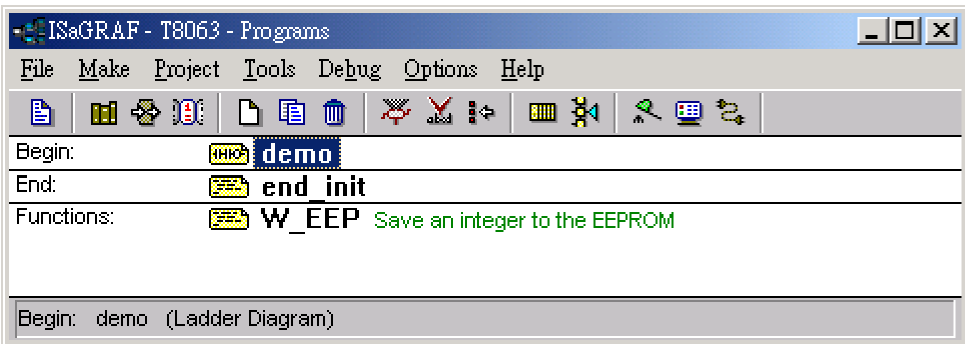


F. Call it in other programs in the same project.

Global variables used in the project:

Name	Type	Attribute	Description
INIT	Boolean	**Internal**	**initial value at “TRUE”**. TRUE means 1st scan cycle
K1	Boolean	**Input**	Connect to 1st ch. Of “push4key”, press it to get “Val”
New_Val	Integer	**Internal**	New value wish to save to the EEPROM
Old_Val	Integer	**Internal**	Old value
Val	Integer	**Internal**	Read back value of the EEPROM

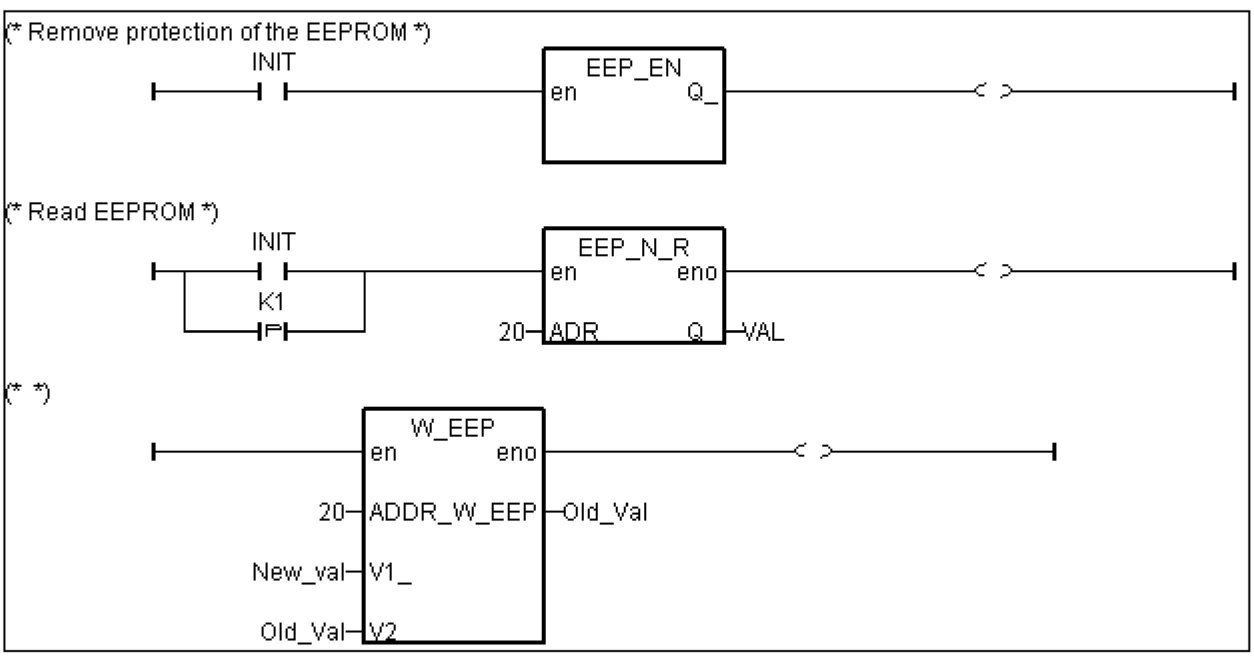
Project architecture:



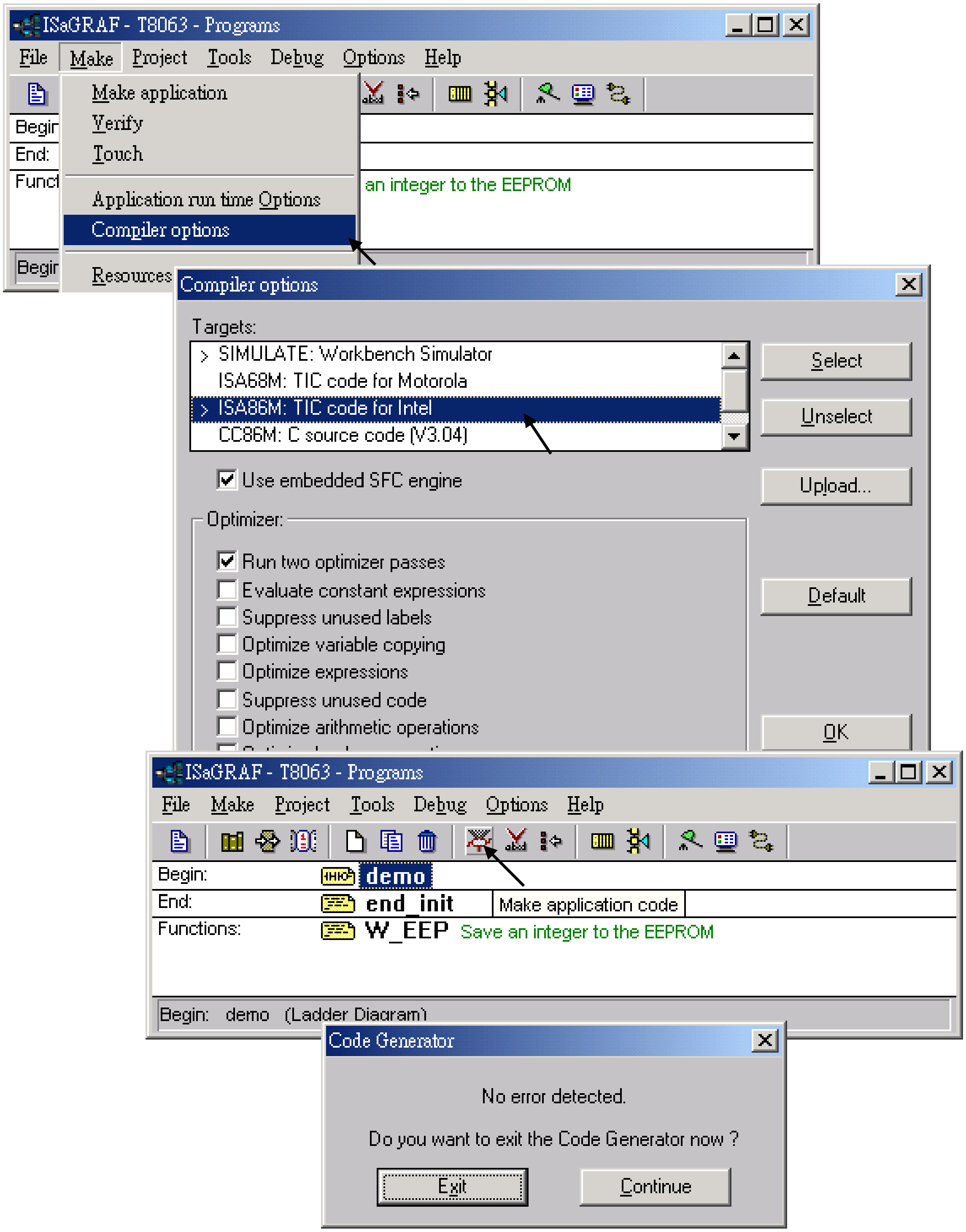
ST program – “end_init” in the “End” area :

```
IF  INIT=TRUE  THEN
        INIT := FALSE ;
END_IF ;
```

LD program – “demo” :



G. Set Compiler Options and compile the project.



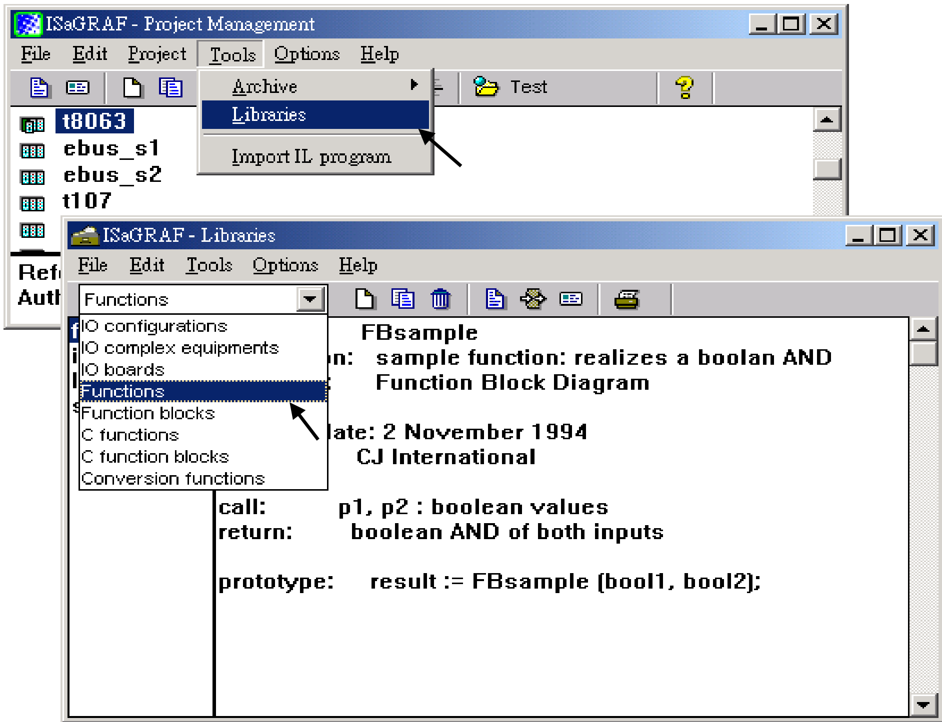
After download to the controller, you may change the “New_Val”, and then press “K1” to see what it happens.

15.2: Creating functions in the ISaGRAF library

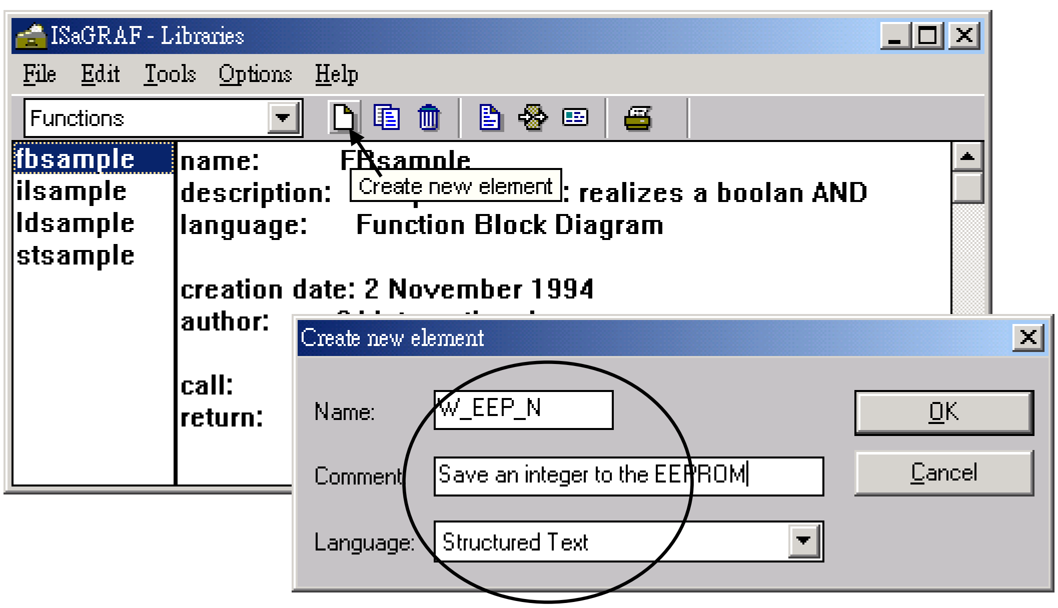
Functions created in the library can be called by programs in any project.

The steps is similar to the former section 15.1. Please refer to it in advance.

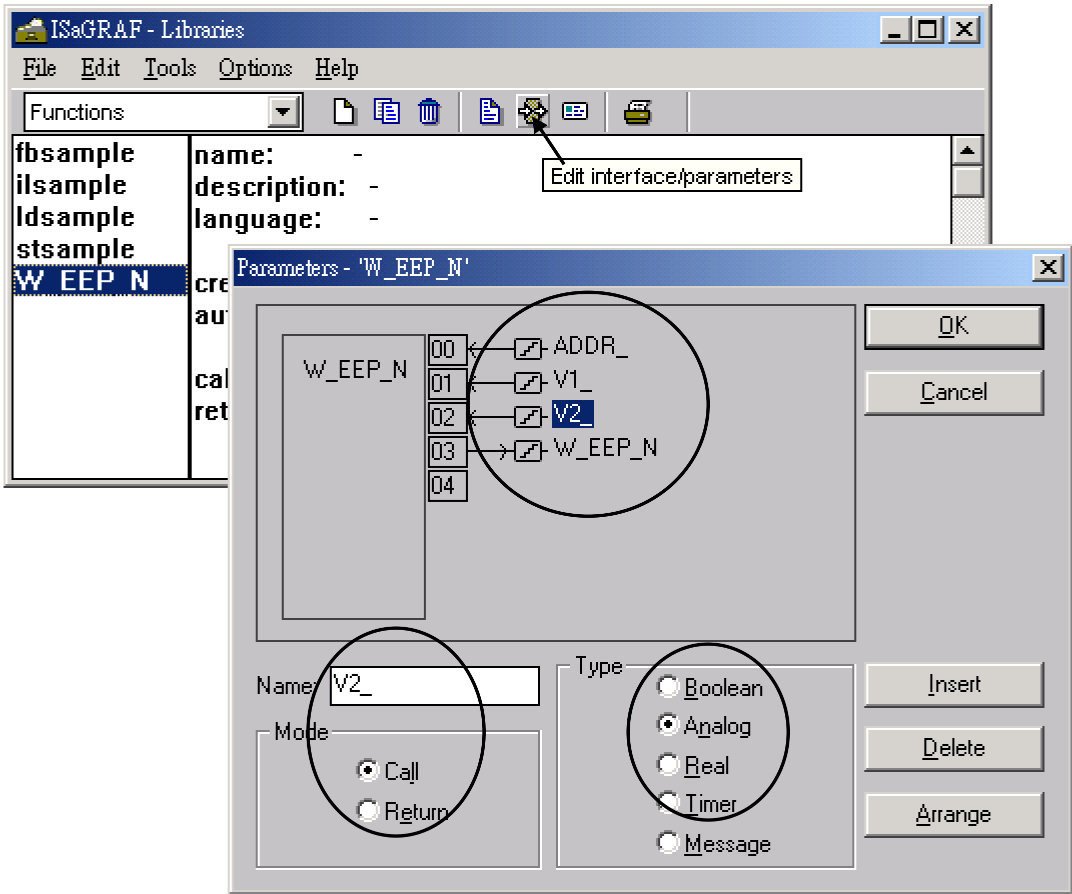
A. Get into the library. Then click on "Functions"



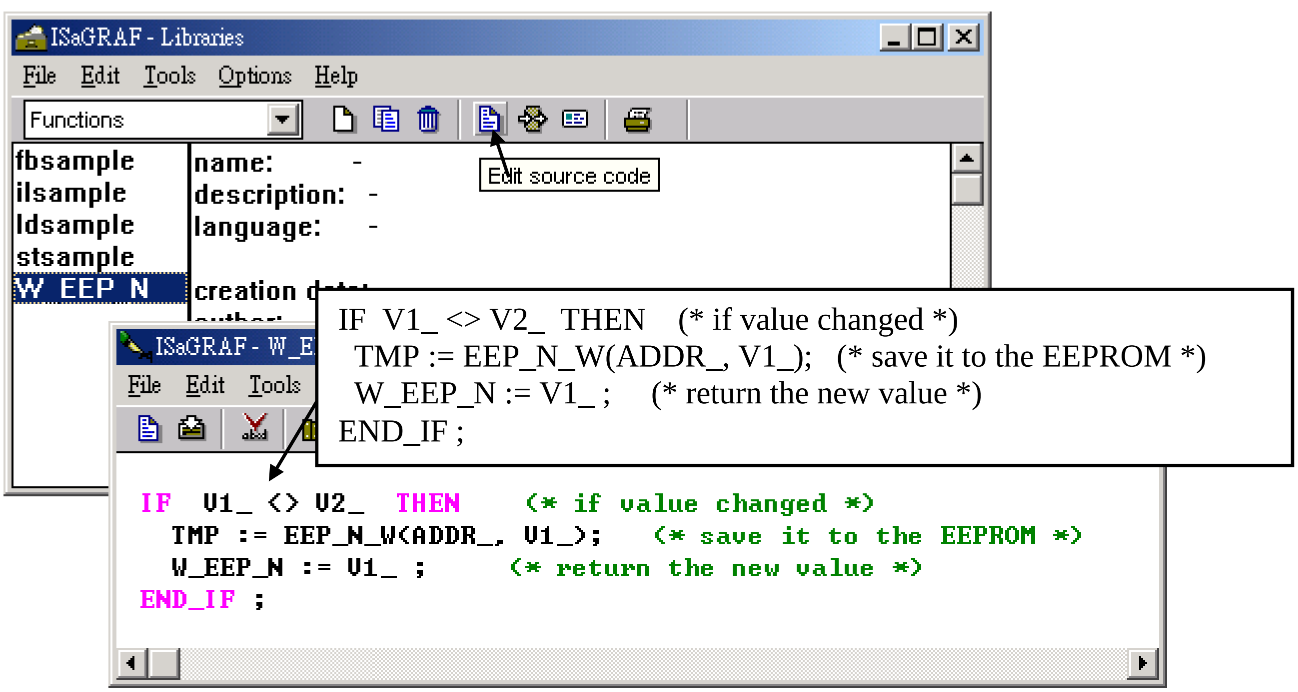
B. Create an new function and given Name as "W_EEP_N" , Language as "Structured Text".



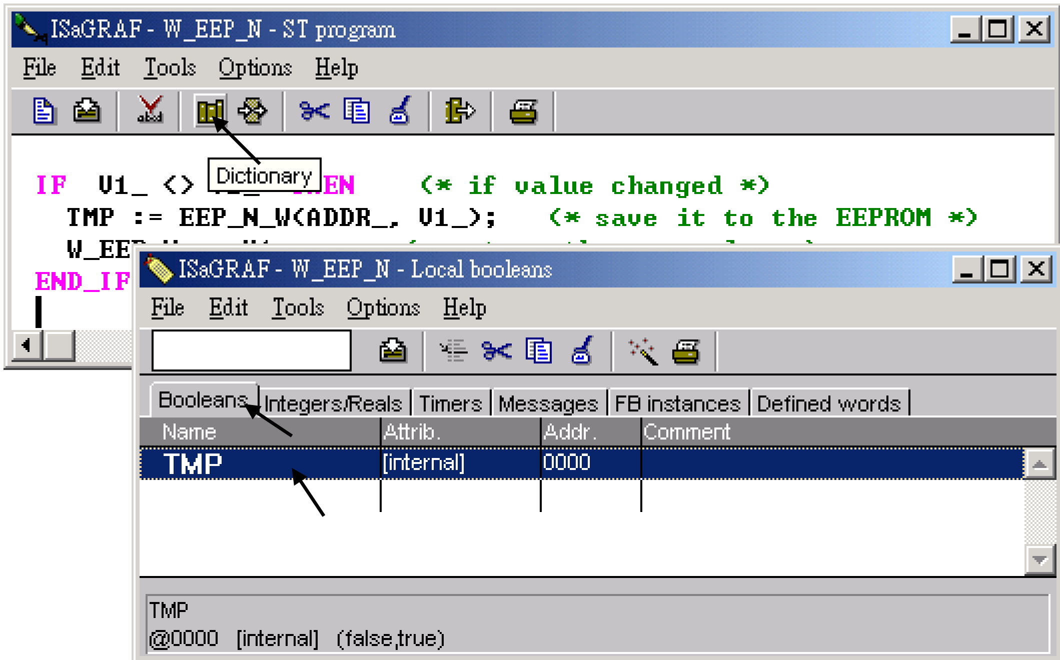
C. Define input and return parameters



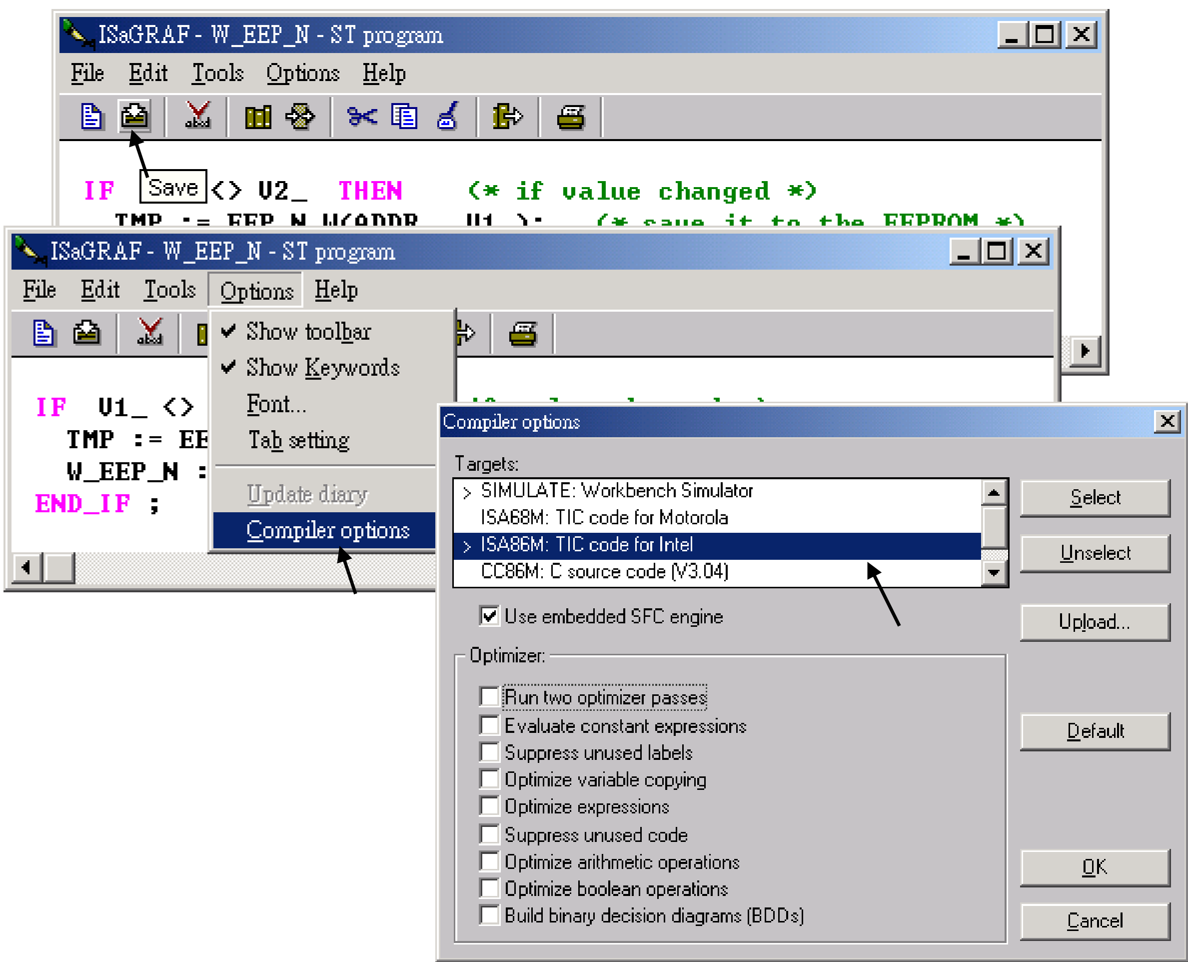
D. Add codes.



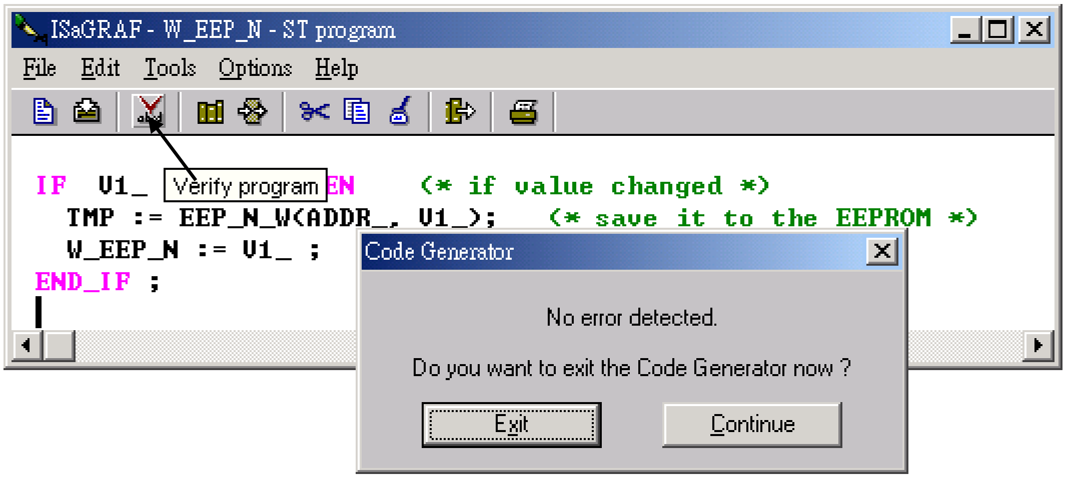
E. Declare local variables. We need a boolean internal variable – "TMP"



F. Save the function and set compiler options.



G. Verify the function.



Then you can call it in any project.

Chapter 16. Linking MMICON

The I-8417/8817/8437/8837, I-7188EG/XG and WinCon-8xx7 controller can integrate the ICP DAS's MMICON to become their Man Machine Interface. The MMICON is featured with a 240 x 64 dot LCD and a 4 x 4 Keyboard. User can use it to display picture, string, integer, float, and input a character, string, integer and float. All control logic is written in ISaGRAF program.

There is a better HMI tools - the Soft-GRAF studio - for the XP-8xx7-Atom-CE6, XP-8xx7-CE6, WP-8xx7, VP-25W7, VP-23W7 and WP-5xx7. Please refer to http://www.icpdas.com/faq/isagraf.htm > FAQ-146 and http://www.icpdas.com/products/Software/Soft-GRAF/soft-graf.htm .

16.1: Hardware Installation

Please refer to the "MMICON Hardware Manual" which is delivered with the hardware for more hardware details. http://www.icpdas.com/products/HMI/touch_lcd/man_machine_list.htm

1. The MMICON has a COM port. Please set as a RS-232 port. (Please look at the jumper "J7" & "J8" setting on the hardware) and the RS-232 cable which is delivered with the MMICON packaging can connect to the COM4 of the I-8417/8817/8437/8837 and CN3 of the MMICON.

   Pin assignment :

   I-8417/8817/8437/8837: COM3 & COM4 can be used.
   WinCon-8xx7: COM2

I-8xx7 (COM4) W-8xx7 (COM2)	MMICON (CN3) RS232
2 RXD	2 TXD
3 TXD	3 RXD
5 GND	5 GND

I-8xx7 (COM3) RS232	MMICON (CN3) RS232
3 RXD	2 TXD
2 TXD	3 RXD
5 GND	5 GND

   I-7188EG/XG: COM3 can be used. (COM3 is added on X503 ~ X51x board)

I-7188EG/XG RS232	MMICON (CN3) RS232
RXD	2 TXD
TXD	3 RXD
GND	5 GND

2. Please set Jumper "J2" of MMICON to position "INIT". I-8417/8817/8437/8837, I-7188EG/XG & W-8xx7 only support COM parameter "9600, 8, N, 1" and "address = 0" to talk to the MMICON.

Note:
If using W-8xx7's COM2 to connect to MMICON, please refer to W-8xx7's "Getting Started" Manual to disable its Modbus RTU function.

16.2: Create Background Picture Of the MMICON

Please refer to the "MMIDOS Software User Manual" which is delivered with the hardware for more software details.

The number of the background pictures depends on the ROM memory on the MMICON. It can up to 256 pages for EPROM like "27040", and 128 pages for "27020", and 64 pages for "27010".

Note: ROM/ EPROM/ EEPROM/ FLASH are all validate.

Please Install the "MMICON" folder from CD-ROM: \Napdos\others\mmicon\ to your hard disk or you can download it from the website:
ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/other/mmicom/

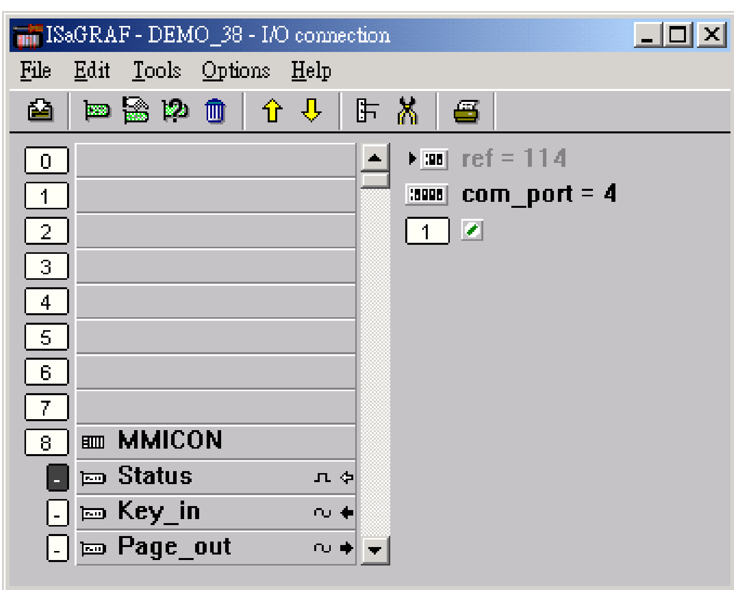
Note: Please change all these file's attribute : **removing "Read-only"**

- Create all the background pages by Microsoft painter (Please refer to "P0.bmp").
- Edit your "Autox.dat" file (Please refer to "Auto1.dat"). This file must remove its "Read-only" attribute. Run "MMIDOS.exe" to build the "romx.bin", For ex. "rom1.bin"
- Using your ROM programmer to burn this "romx.bin" image to the ROM memory. Then plug it into the socket on the MMICON.

Please refer to the "MMIDOS Software User Manual" which is delivered with the hardware for more software details.

16.3: Writing Control program

The I/O complex equipment "mmicon" should be connected to the I/O connection window first. You can find 3 boards under "MMICON".



Status:

Parameter "com_port" defines the COM No. to link to the MMOCON. 3 or 4 for I-8xx7, while 3 for I-7188EG/XG , and 2 for W-8xx7
1 channel of Digital Input: True means communication between the controller and the MMICON is Ok. FALSE means fail.

Key_in:

1 channel of Integer Input: The value is the key been pressed. And the value will last only for one scan cycle, then go back to 0.

Key	Key code value	Key	Key code value
0	16#30	**Enter**	16#0D
1	16#31	**.**	16#2E
2	16#32	**Left**	16#1B
3	16#33	**Right**	16#1A
4	16#34	**Up**	16#18
5	16#35	**Down**	16#19
6	16#36	**Back space**	16#08
7	16#37	**F1**	16#F1
8	16#38	**F2**	16#F2
9	16#39	**F3**	16#F3
A	16#41	**F4**	16#F4
B	16#42		
C	16#43		
D	16#44		
E	16#45		
F	16#46		

Page_out:

1 channel of Integer Output: The value output define the page No. to display.

The I-8417/8817/8437/8837, I-7188EG/XG & W-8xx7 controller provide below functions to control the action of the MMICON.

MI_BOO	Display a boolean value as "ON" or "OFF"
MI_INT	Display an integer value
MI_REAL	Display a real value
MI_STR	Display a string
MI_INP_N	To enter an integer
MI_INP_S	To enter a string
REAL_STR	Convert a real value to a string
STR_REAL	Convert a string to a real value

Please refer to the demo_38, dem_39 in chapter 11.

Chapter 17. SMS: Short Message Service

The ISaGRAF controller can integrate with a GSM Modem to support SMS: Short Message Service. This allows user to request information or control something from his own cellular phone to the ISaGRAF controller. Beside, the controller can also send information and alarms to user's cellular phone. The following is the COM port number of the GTM-201-RS232 for connecting different PAC:

I-8xx7: 4 or 5	I-7188EG/ µPAC-7186EG/µPAC-5xx7: 1 or 3 or 4	XP-8xx7-Atom-CE6/ XP-8xx7-CE6: 5 or 6
I-7188XG: 3 or 4	VP-2117: 3 or 5	WP-5xx7: 3
iP-8xx7: 4 or 5	VP-25W7 / VP-23W7: 3 or 5	WP-8xx7: 4 or 5

Note:
Recommend not to use the GTM-201-RS232 if your PAC is the WP-8xx7, VP-25W7 / VP-23W7 and XP-8xx7-CE6, XP-8xx7-Atom-CE6 using the I-8212W / I-8213W (GPRS/GSM board) is better. The I-8212W / I-8213W can be plugged in the leftmost I/O slot of the WP-8xx7, VP-25W7 / VP-23W7 (slot 0) and the XP-8xx7-CE6 / XP-8xx7-Atom-CE6 (slot 1). To enable the I-8212W / I-8213W, please refer to http://www.icpdas.com/faq/isagraf.htm > FAQ-143. For using the I-8212W / I-8213W, the related COM port is, WP-8xx7, VP-25W7 / VP-23W7: COM5, XP-8xx7-CE6, XP-8xx7-Atom-CE6: COM6.

17.1: Hardware Installation

The I-8417/8817/8437/8837 supports SMS since its driver version of 2.24, while version 1.14 for I-7188EG, and version 1.12 for I-7188XG. If your driver is older one, please upgrade the hardware driver to the associate version or a higher version. The driver can be found from the below ICP DAS's web site: http://www.icpdas.com/products/PAC/i-8000/isagraf-link.htm

The I/O library should be re-installed if yours is older one. Please refer to section 1.2. Or you can refer to Appendix A.2 to simply install "C functions" with the below items.

SMS_test, SMS_get, SMS_gets, SMS_send, SMS_sts and "I/O complex equipment" : SMS.

Recommend to use the GTM-201-RS232 as GSM Modem. You may purchase them from ICP DAS or from your local agent. ICP DAS is not sure for other GSM modems working or not.

Note: Please REMOVE the password setting in SIM card , then plug it into GSM modem.

iP-8xx7(COM4/5) W-8xx7(COM2)		GSM cable of GTM-201-RS232
2 RXD	————	2 TXD
3 TXD	————	3 RXD
5 GND	————	5 GND
4 DTR	-------------	4 DSR
7 RTS	-------------	7 CTS

7188EG/XG:COM3/4 RS232		GSM cable of GTM-201-RS232
RXD	————	2 TXD
TXD	————	3 RXD
GND	————	5 GND
DTR (or RTS)	-------------	4 DSR
DTR (or RTS)	-------------	7 CTS

17.2: A SMS demo example

The demo project is located at I-8xx7's demo_43, please refer to section 11.1 to install it to your ISaGRAF workbench. Or It can be downloaded at ICP DAS's FTP site.

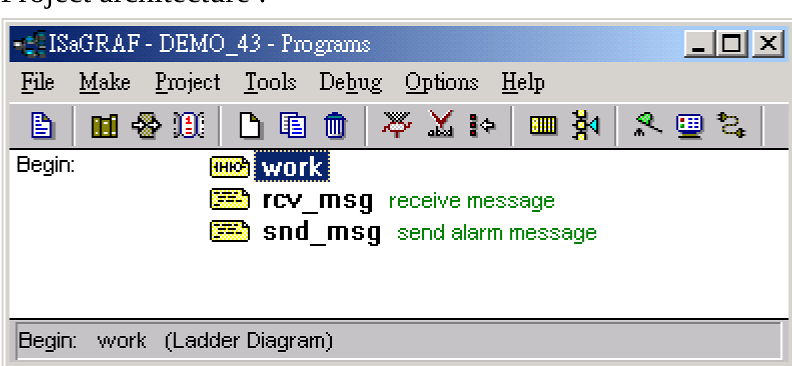
ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/8000/demo/

Moreover, the demo_43a is an example for sending SMS to multiple cellphone can be obtained at the same ftp site above.

Variables :

Name	Type	Attribute	Description
M1	Boolean	**Internal**	Trigger to send an alarm message when K1 is pushed
M2	Boolean	**Internal**	Trigger to send a report message when a message is coming
K1	Boolean	**Input**	Pushbutton 1, connect to push4key
L1	Boolean	**Output**	Output 1, connect to show3led
L2	Boolean	**Output**	Output 2, connect to show3led
L3	Boolean	**Output**	Output 3, connect to show3led
Q1	Boolean	**Internal**	Test if message is coming
TMP	Boolean	**Internal**	Temportary usage
SMS_available	Boolean	**Input**	is SMS available ? connect to SMS - status
T1	Timer	**Internal**	Blinking time of L1 to L3, init at T#500ms
data	Message	**Internal**	The coming Message
phone	Message	**Internal**	phone No. of sender
Date_time	Message	**Internal**	Message coming date & time in string format
To_who	Message	**Internal**	phone No of receiver, **please use your own No.**
Msg_to_send	Message	**Internal**	Message to send out
Year1	Integer	**Internal**	Message coming year
Mon1	Integer	**Internal**	Message coming month
Day1	Integer	**Internal**	Message coming date
Wday1	Integer	**Internal**	Message coming week date
Hour1	Integer	**Internal**	Message coming hour
Min1	Integer	**Internal**	Message coming minute
Sec1	Integer	**Internal**	Message coming second
Q1_cnt	Integer	**Internal**	Message coming count, declared as retained variable
Msg_status	Integer	**Internal**	Message sending status
TMP_v	Integer	**Internal**	temportary usage

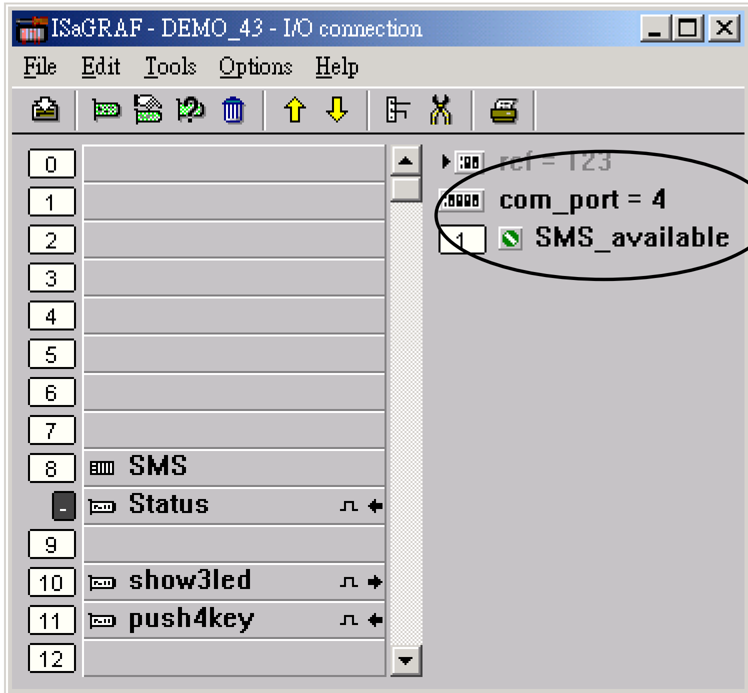
Project architecture :



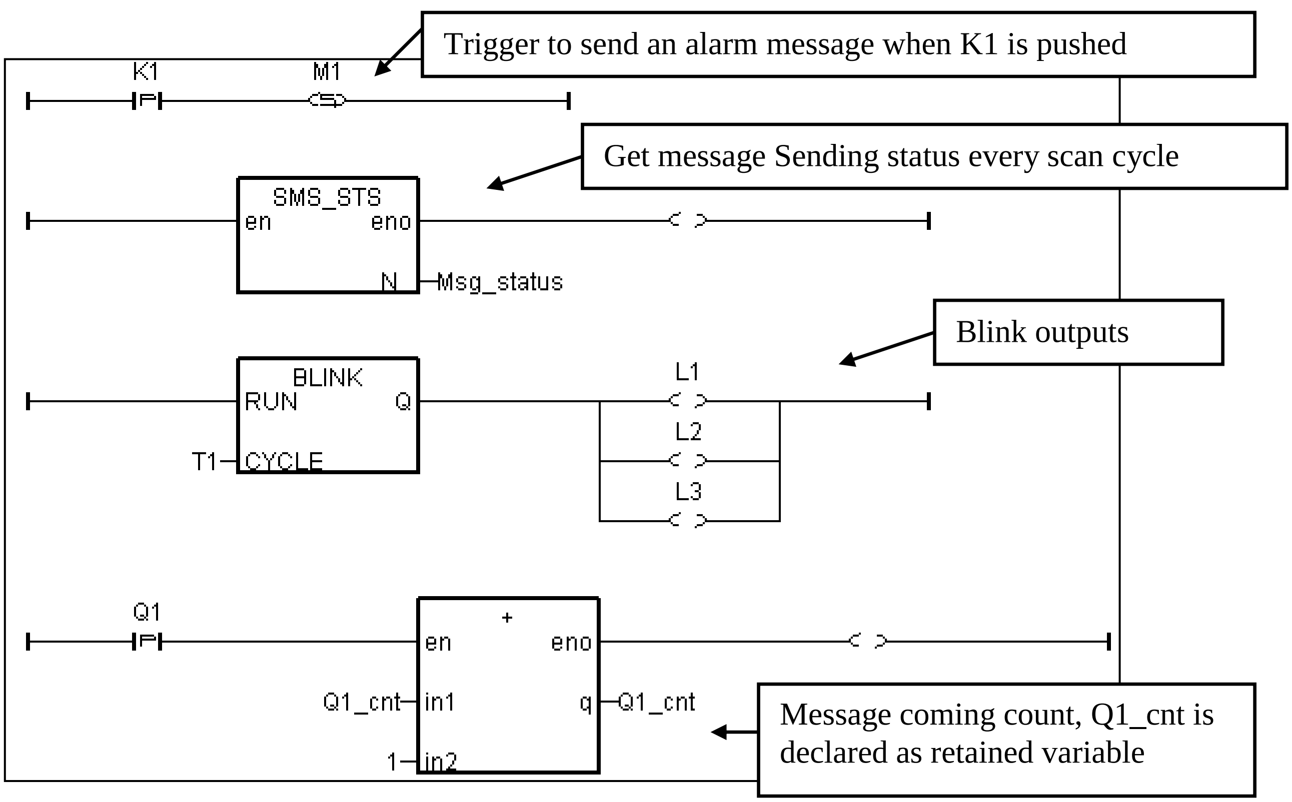
Operation actions:

1. If K1 is pushed, an Alarm message will be sent.
2. If the user send a message in format, for ex. T0200 or T1500 to the controller, the blinking period will change to 200ms and 1500ms. And then the controller will response a report message back to the user.

I/O connection:



LD program : work



ST program : rcv_msg

```
Q1 := SMS_test();
```

Test if a message is coming or not

```
if Q1 then
```

if a message coming

```
 Year1   := SMS_get(1);
 Mon1    := SMS_get(2);
 Day1    := SMS_get(3);
 Wday1   := SMS_get(4);
 Hour1   := SMS_get(5);
 Min1    := SMS_get(6);
 Sec1    := SMS_get(7);
```

call SMS_get to get message coming date & Time in integer format

get phone No. of sender

```
 phone     := SMS_gets(2);
 date_time := SMS_gets(3);
```

get message coming date & time in string format

```
 data      := SMS_gets(1);
```

get message data, SMS_gets(1) should be called in the last one, because it will reset SMS_test status to FALSE:No message coming

Check the coming message. For ex. T1500 will result T1=1500 ms, while T0300 result T1=300ms, however TAB10 will result T1=0 ms (not valid)

```
 if mid(data,1,1) = 'T' then
```

check 1st char is T or not

```
  TMP_v := ANA(mid(data,4,2));
```

extract 4 bytes starting from string position 2, and then convert to an integer

```
  (* valid format *)
  if TMP_v>=50 and TMP_v<=9999 then
    T1 := TMR(TMP_v);    (* convert to timer *)

    Msg_to_send := 'Current T1 change to ' + Msg(TMP_v) + ' ms.';
    M2 := TRUE;
```

Trigger to send a report message to sender

```
  else   (* invalid format*)

    Msg_to_send := '!!! Wrong command, Val should be between T0050 to T9999. Current T1 remains
at ' + Msg(Ana(T1)) + ' ms.';
    M2 := TRUE;
```

Trigger to send a report message to sender

```
  end_if;

 end_if;    (* if mid(data,1,1) = 'T' then *)

end_if;   (* if Q1 then *)
```

ST program : snd_msg

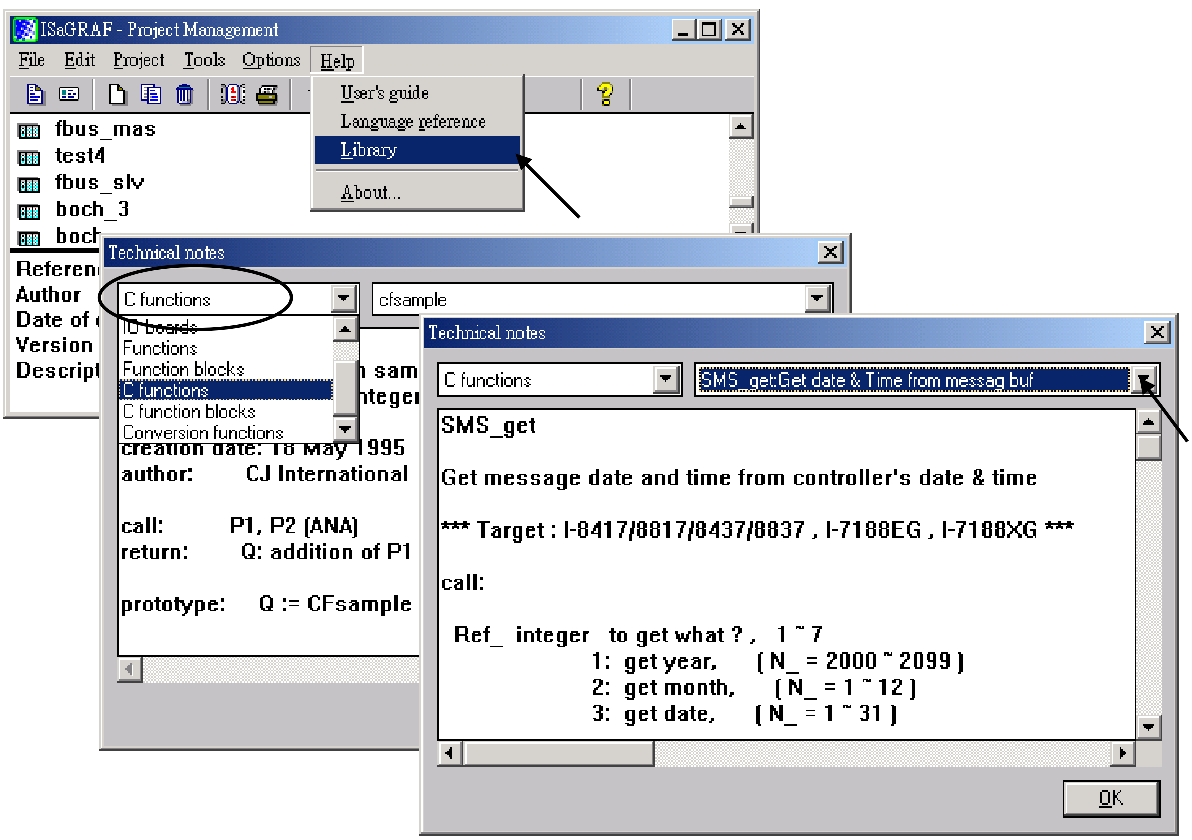
if message sending status is not 1:busy

```
if (Msg_status <> 1) and SMS_available then

 if M1 then   (* alarm triggering *)

  TMP := SMS_send(to_who,'K1 is pushed!');
  M1 := FALSE;

 elsif M2 then  (* Report triggering *)

  TMP := SMS_send(phone,Msg_to_send);  (* report message back *)
  M2 := FALSE;

 end_if;

end_if;
```

Message sending status:
0: waiting for a new sending request
1: busy. (message is processing now) 21: The message is sent successfullly
-1: SMS system is not available
-2: Timeout, No response.

Must disable it (set to FALSE) after SMS_send is called

More description of SMS_sts, SMS_send, SMS_test, SMS_get & SMS_gets, Please refer to ISaGRAF's On-line Help. "Library" – "C functions" – "SMS_xxxx"



Chapter 18. Motion

Note:

1. XP-8xx7-CE6 and XP-8xx7-Atom-CE6 support the I-8094 / 8094F (4-axis Motion & Encoder) and the I-8092F (2-axis Motion & Encoder) and that is the better solution than I-8091W, please refer to http://www.icpdas.com/faq/isagraf.htm > FAQ-132.

Limitation:

1. If using the I-8417/8817/8437/8837, the standard driver version is not support the I-8090 and I-8091. Therefore, the users need to update it to the "Motion" driver, please visit the website below: http://www.icpdas.com/products/PAC/i-8000/isagraf-link.htm

2. I-8437/8837 CAN NOT do Ethernet communication when using I-8091 to do motion control, while WinCon-8xx7 doesn't have this limitation. (The standard driver version for WinCon-8xx7 has supported the I-8090 and I-8091.)

3. Only one I-8091 board in I-8xx7 & WinCon-8xx7 can do X-Y dependent motion, other I-8091 boards should be moving independent. Moreover, the I-8xx7 can support max. two I-8091 and the WinCon can support max. four I-8091.

18.1: Install motion driver

Restriction of the motion driver of I-8417/8817/8437/8837:

The motion driver for I-8417/8817/8437/8837 doesn't support the Ethernet communication, however W-8337/8737 desen't have this limitation.

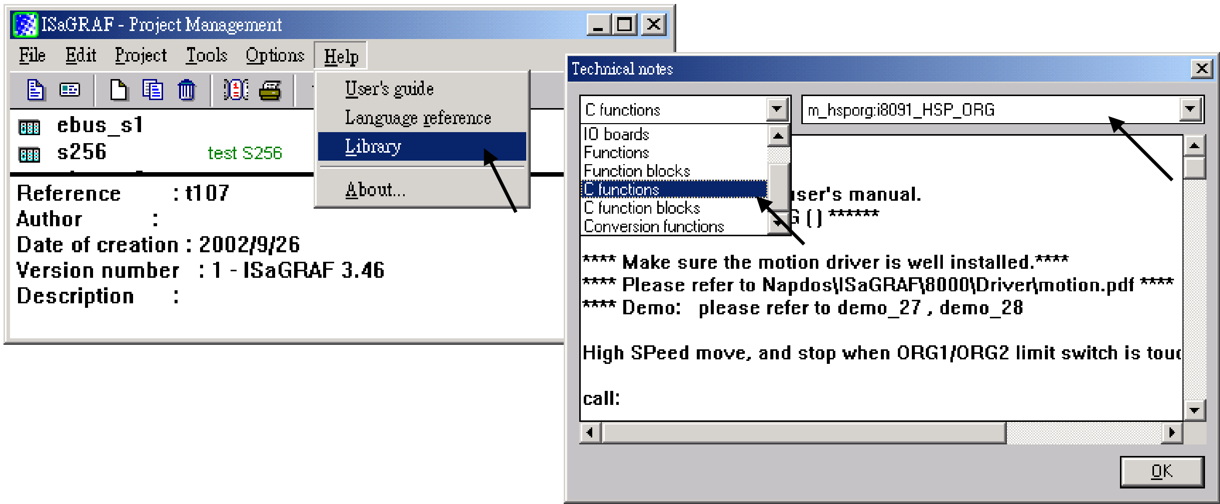
The ISaGRAF demo projects of motion for I-8417/8817/8437/8837 are "demo_27" , "demo_28", & "demo_46". They are located in the

I-8000 CD-ROM: \napdos\isagraf\8000\demo\" , or at
ftp.icpdas.com/pub/cd/8000cd/napdos/isagraf/8000/demo/

The ISaGRAF demo projects of motion for W-8337/8737/8347/8747 are "wdemo_26" , "wdemo_27", "wdemo_28" & "wdemo_29". They are located in the

Wincon CD-ROM: \napdos\isagraf\wincon\demo\" , or at
ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/demo/

All functions that trigger I-8091 & I-8090 are named as ”M_???” , Please refer to the On-line help from the ISaGRAF “Help” – “Library” - “C functions” for names starting with “M_???”.



Beside, please refer to “I-8091 & I-8090 User’s Manual” .It can be found in the package box of the I-8091, or

I-8000 CD-ROM: napdos\8000\motion\i8091\manual\
ftp site: ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/8000/motion/i8091/manual/

18.2: Introduction

18.2.1: System Block Diagram

The I-8091 stepping motor control card is a micro-computer controlled, 2-axis pulse generation card. It includes a 2Kbytes-FIFO to receive motion command from host, a micro-computer for profile generation and protection, 2-axis DDA chip to execute DDA function when interpolation command is used, 2500Vrms optical isolation inserted for industrial application.

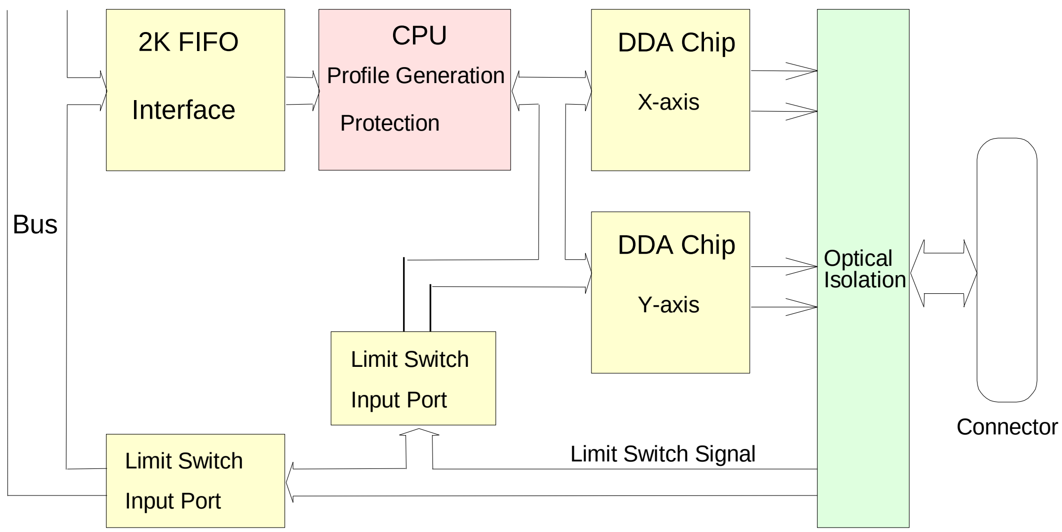


Fig.(1) block diagram of I-8091 card

18.2.2: DDA Technology

The DDA chip is the heart of I-8091 card, it will generate equal-space pulse train corresponding to specific pulse number during a DDA period. This mechanism is very useful to execute pulse generation and interpolation function. The DDA period can be determined by DDA cycle. Table(1) shows the relation among DDA cycle, DDA period and output pulse rate. When DDA cycle set to 1, the DDA period is equal to (1+1)x1.024ms = 2.048ms. The output pulse number can be set to 0~2047, therefore the maximum output pulse rate will be 1Mpps. The minimum output pulse rate is 3.83pps when set DDA cycle=254 (DDA period = (254+1)x1.024ms = 261.12ms).

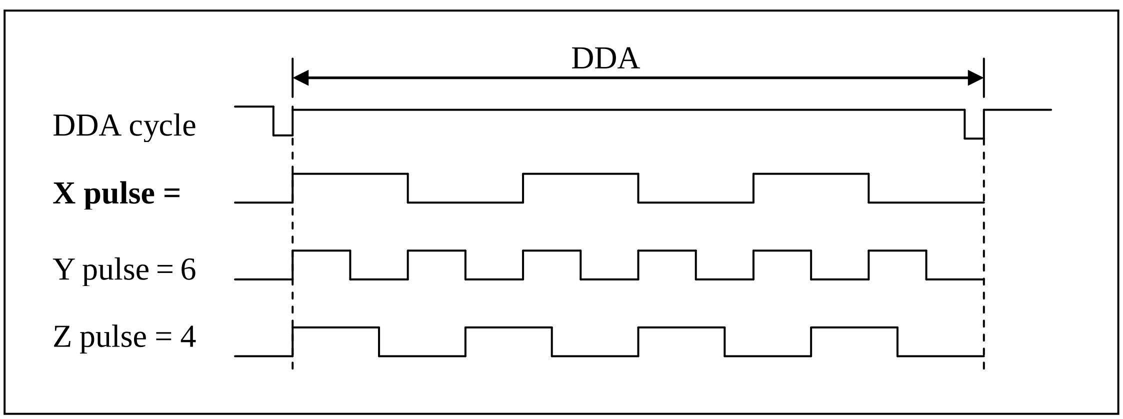


Fig.(2) DDA mechanism

Table(1) The Relation among DDA cycle, DDA period and output pulse rate.

DDA cycle	DDA period	Max. pulse rate(n=2047)	Min. pulse rate (n=1)
1	2.048ms	999511pps	488pps
2	3.072ms	666341pps	325pps
3	4.096ms	.	.
.	.	.	.
N	(N+1)*1.024ms	2047/(DDA period)	1/(DDA period)
.	.	.	.
254	261.12ms	7839pps	3.83pps

The DDA cycle can be set by i8091_SET_VAR() command which decribed in charpter 3. The selection criterion of DDA cycle was described as following.

1. The required max. output pulse rate.

PRmax = Vmax*N/60

$$\text{PRmax} = \frac{2047}{(DDAcycle + 1) * 1.024ms}$$

PRmax : max. output pulse rate.

Vmax : max. speed (rpm).

N : the pulse number of stepping motor per revolution (pulse/rev).

2. The required speed resolution.
   The maximum output pulse number is Np(0~2047), therefore the speed resolution is Vmax(max. speed)/Np. The DDA cycle can be obtained by following equation.

$$\text{PRmax} = \frac{Np}{(DDAcycle + 1) * 1.024ms}$$

3. When choose large DDA cycle (DDA period), it will occur vibration between different pulse input which generally can be observed during acceleration or deceleration. So, the small DDA cycle , the smooth acceleration/deceleration curve as long as the speed resolution is acceptable.

Example: Stepping Motor
The spec. of stepping motor is 500 pulse/rev, max. speed 500 rpm, speed resolution 2 rpm.

The required max. pulse rate
PRmax = 500 rpm*500/60 = 4166.67 pps

The maximum output pulse
Np = 500rpm/2rpm =250 pulse number

The DDA cycle can be calculated by follow equation

$$PRmax = \frac{Np}{(DDAcycle + 1) * 1.024ms}$$

$$4166.67 = \frac{250}{(DDAcycle + 1) * 1.024ms}$$

DDA cycle = 58

High Speed = 247 pulse (4166.67*58*0.001024)

The above results means that maximum speed is 500rpm when send command i8091_SET_VAR(0, 58, 2, 2, 247) to I-8091 card.

Example: Pulse type input Servo Motor
The spec. of servo motor is 8000 pulse/rev, max. speed 3000 rpm, speed resolution 2 rpm.

The required max. pulse rate

PRmax = 3000 rpm*8000/60 = 400,000 pps

The maximum output pulse

Np = 3000rpm/2rpm =1500 pulse number

The DDA cycle can be calculated by follow equation

$$PRmax = \frac{Np}{(DDAcycle + 1) * 1.024ms}$$

$$400{,}000 = \frac{1500}{(DDAcycle + 1) * 1.024ms}$$

DDA cycle = 3

High Speed = 1638 pulse (400,000*4*0.001024)

The above results means that maximum speed is 3000rpm when send command i8091_SET_VAR(0, 3, 2, 2, 1638) to I-8091 card.

18.3: Hardware

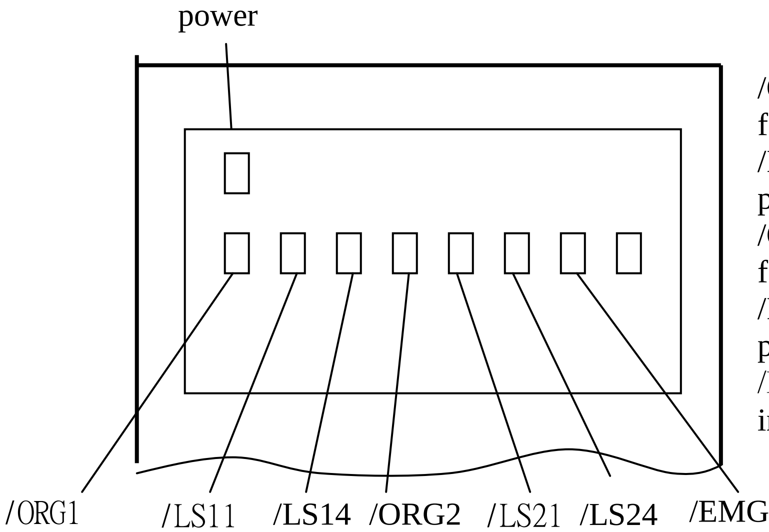
18.3.1: I-8000 hardware address

The hardware address of I-8000 main system is fixed as following table. There are 4 slots I-8000 and 8 slots I-8000.

	Slot 0	Slot 1	Slot 2	Slot 3	Slot 4	Slot 5	Slot 6	Slot 7
I-8000, 4 slot address	0x080	0x0A0	0x0C0	0x0E0	---	---	---	---
I-8000, 8 slot address	0x080	0x0A0	0x0C0	0x0E0	0x140	0x160	0x180	0x1A0

Fig.(3) I-8000 hardware address

18.3.2: LED Indicator



/ORG1: X-axis's original limit switch for machine home position.
/LS11, /LS14 : X-axis's negative and positive limit switches.
/ORG2: Y-axis's original limit switch for machine home position.
/LS21, /LS24 : Y-axis's negative and positive limit switches.
/EMG : system's emergency signal input.

Fig.(4) I-8091 LED indicator

18.3.3: Hardware Configuration

Limit switch configuration
Because the profile generation and protection is executed by the CPU on I-8091 card, the limit switches must configure as following diagram. The motion command just can work properly.

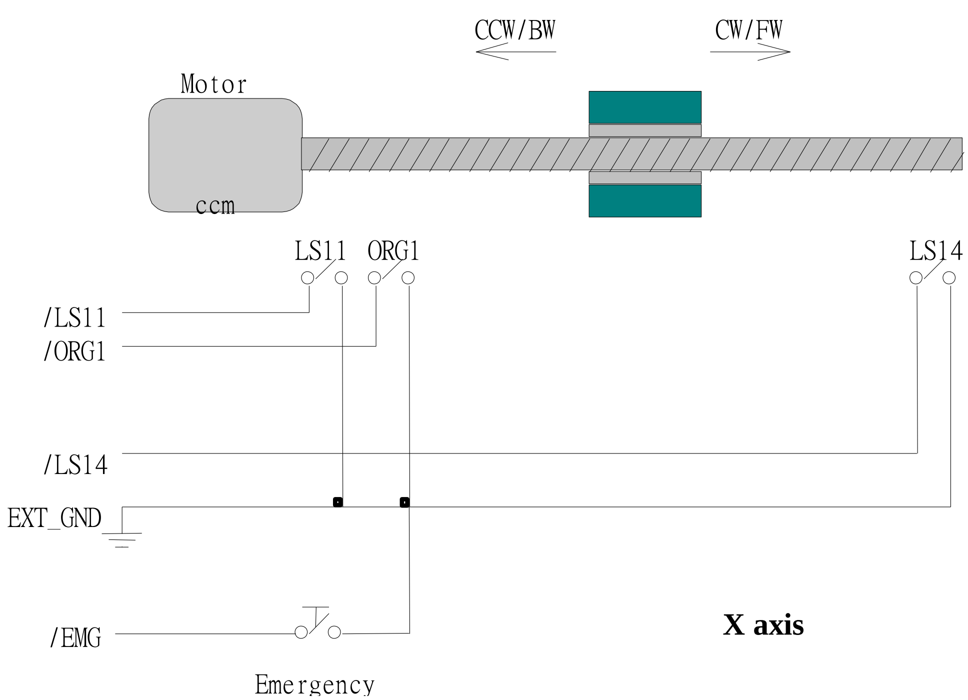
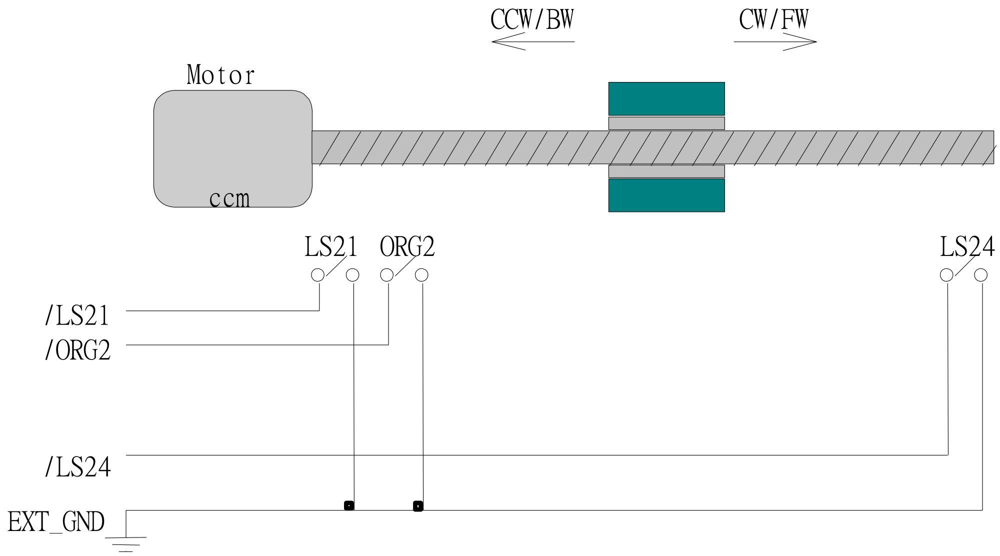


Fig.(5) Limit switch configuration of X axis



Y axis

Fig.(6) Limit switch configuration of Y axis

Output pulse mode configuration
I-8091 card provide two kind output method.

(a) CW/CCW mode
(b) Pulse/Direction mode

The command **M_s_mode(card_NO_, modeX_, modeY_)** provide parameters 0: CW_CCW and 1: PULSE_DIR to define output pulse mode.

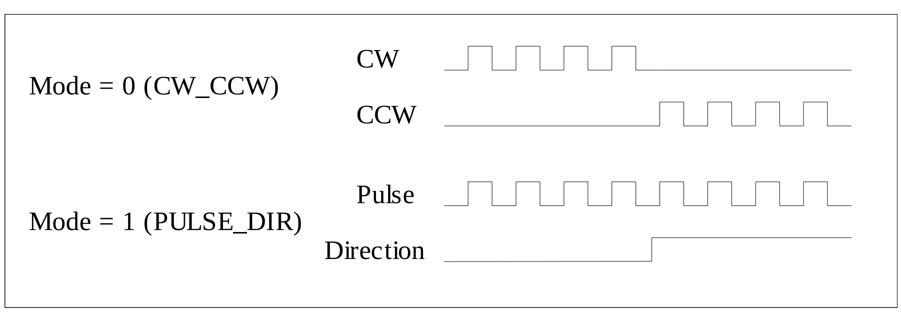


Fig.(7) Output pulse mode

Direction configuration

Sometimes, the output direction of X-axis, Y-axis is not in the desired direction due to the motor's connection or gear train. It is recommended to unify the output direction as shown in Figure(5)(6). The CW/FW direction is defined as toward outside from motor and the CCW/BW direction is defined as toward inside to motor. The **M_s_dir(card_NO_, defdirX_, defdirY_)** command provides parameters 0: NORMAL_DIR and 1:REVERSE_DIR to define the rotating direction of motor.

Turn Servo ON/OFF (Hold ON/OFF)

To turn servo motor into servo ON(OFF) state, or turn stepping motor into hold ON(OFF) state, the command **M_s_serv(card_NO_, sonX_, sonY_)** provide parameters 1:ON and 0:OFF to turn ON or OFF.

Automatic protection

The I-8091 card has a automatic protected system.

(a) If X-aixs command is executing and moving toward CW/FW direction, X-axis will immediately stop when LS14 is touched. To release this protection as long as X-axis move toward CCW/BW direction.

(b) If X-aixs command is executing and moving toward CCW/BW direction, X-axis will immediately stop when LS11 is touched. To release this protection as long as X-axis move toward CW/FW direction.

(c) If Y-aixs command is executing and moving toward CW/FW direction, Y-axis will immediately stop when LS24 is touched. To release this protection as long as Y-axis move toward CCW/BW direction.

(d) If Y-aixs command is executing and moving toward CCW/BW direction, Y-axis will immediately stop when LS21 is touched. To release this protection, as long as Y-axis move toward CW/FW direction.

(e) If the signal of the emergency limit switch /EMG was found in CPU firmware, all motion will be terminated and stop.

Set limit switch as normal close condition

The limit switches /EMG, /LS11, /LS14, /LS21, /LS24, /ORG1, /ORG2 is initially normal open condition, that is, these signal is active when connect it to ground. In industrial application, it might be recommended normal close condition, that is, these signal is active when open from ground.
The **M_s_nc(card_NO_, sw_)** command can be set sw=0 (default), for normal open condition. When set sw=1, for normal close condition.

18.3.4: Pin assignment of connector CN2

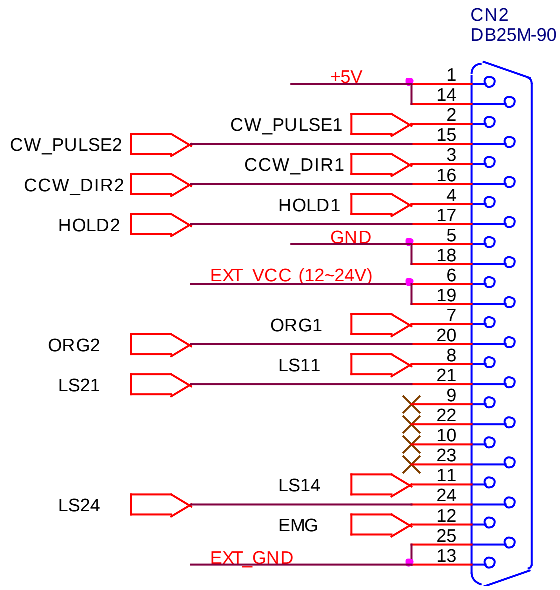


Fig.(8) CN2 connector of I-8091

Table of CN2 connector's pin assignment

pin name	pin number	Description
+5V	1	Internal +5V power, Max. output current: 50mA
CW_PULSE1	2	X-axis CW (Pulse) output pin
CCW_DIR1	3	X-axis CCW (Direction) output pin
HOLD1	4	X-axis HOLD (servo on) output pin
GND	5	Signal ground of pin 2,3,4
EXT_VCC	6	External power(12~24V) for limit switches
/ORG1	7	X-axis original (home) limit switch
/LS11	8	X-axis limit switch
	9,10	No used
/LS14	11	X-axis limit switch
/EMG	12	Emergency input
EXT_GND	13	External ground for limit switch
+5V	14	Internal +5V power, Max. output current: 50mA
CW_PULSE2	15	Y-axis CW (Pulse) output pin
CCW_DIR2	16	Y-axis CCW (Direction) output pin
HOLD2	17	Y-axis HOLD (servo on) output pin
GND	18	Signal ground of pin 15,16,17
EXT_VCC	19	External power(12~24V) for limit switches
/ORG2	20	Y-axis original (home) limit switch
/LS21	21	Y-axis limit switch
	22,23	No used
/LS24	24	Y-axis limit switch
EXT_GND	25	External ground for limit switch

The internal circuit of CW_PULSE, CCW_DIR, HOLD

When output these signal as 1, it can source 15mA(max.).
When output these signal as 0, it can sink 50mA(max.)

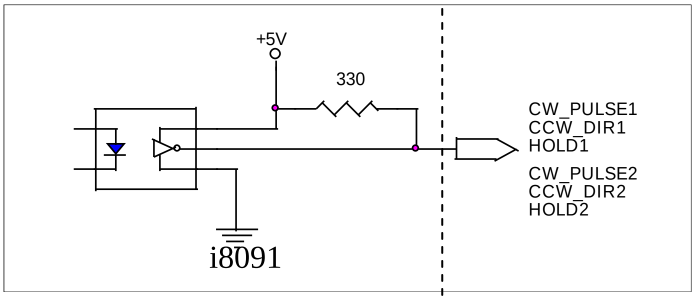


Fig.(9) internal circuit of pulse output pin

The internal circuit of limit switch input

Initially, the limit switch inputs of I-8091 board are normal open (N.O.), the I-8091 board will automatic protect when limit switch pin connect to EXT_GND. The user can use the command **M_s_nc(card_NO_, 1)** to let those limit switch input as normal close condition at the beginning of the user's program.

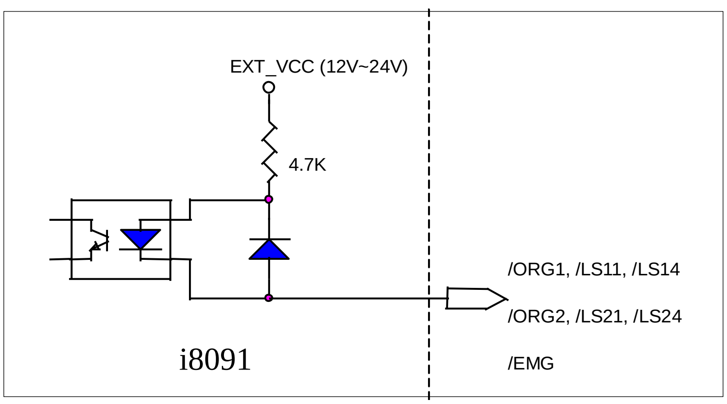


Fig.(10) internal circuit of limit switch input pin

Example of connection

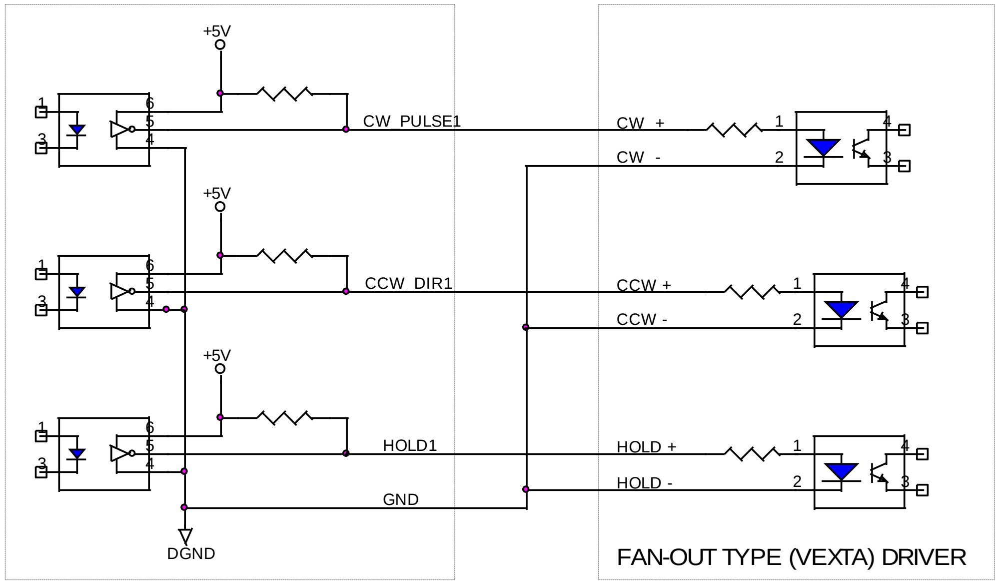


Fig.(11) fan-out type driver (VEXTA's motor driver)

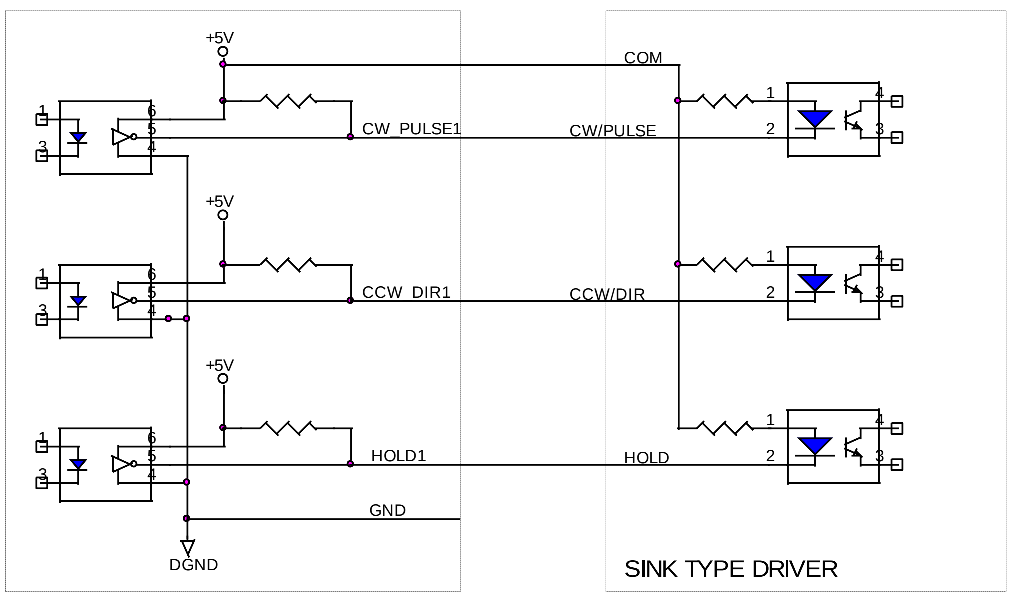


Fig.(12) Sink type driver

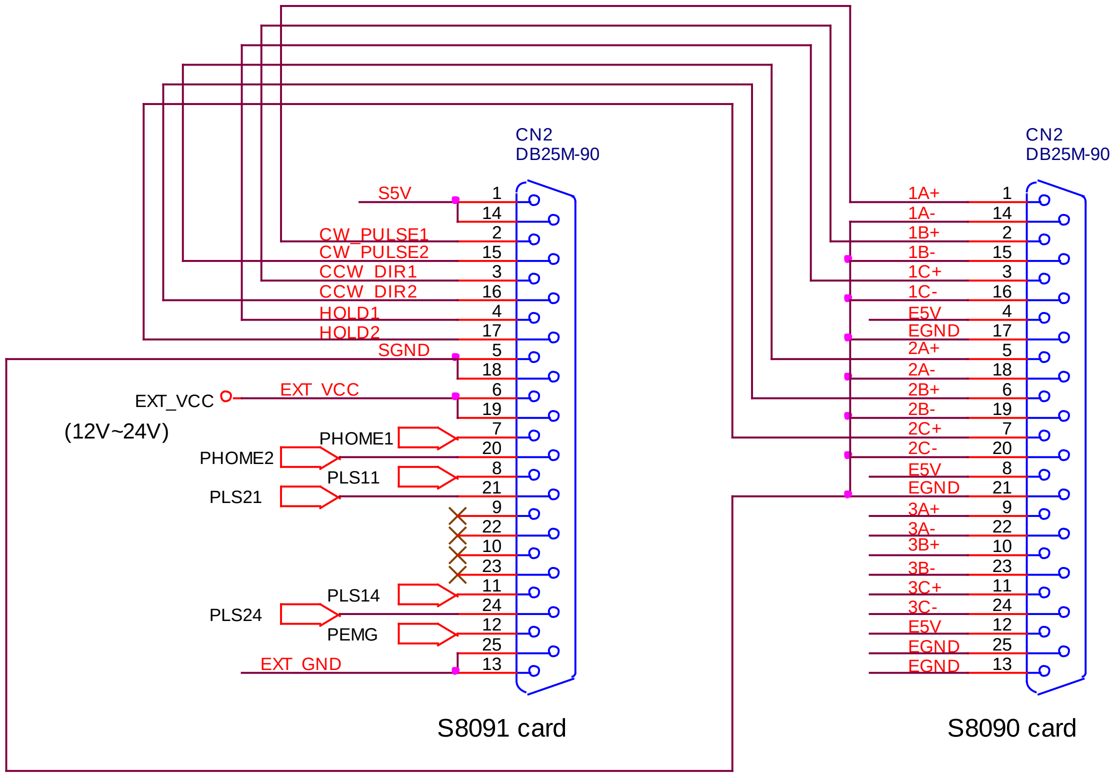
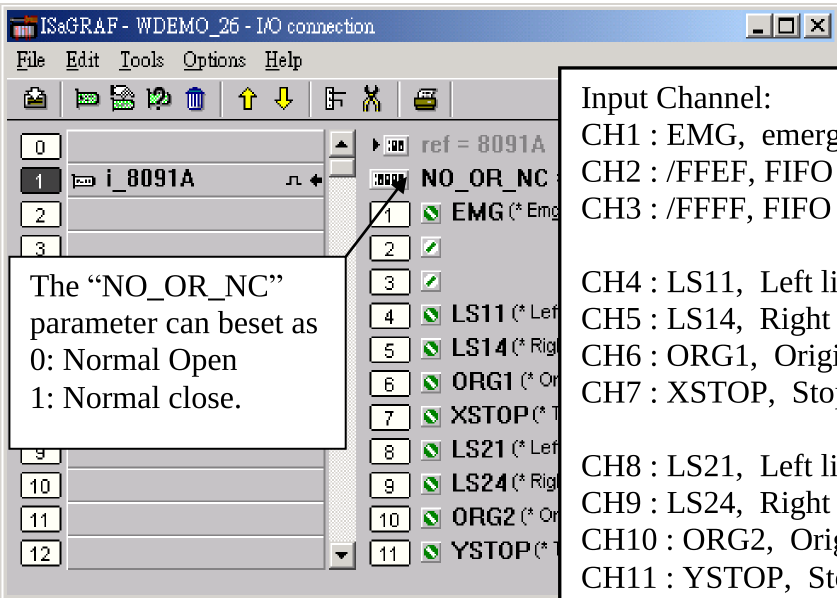


Fig.(13) The connection between I-8090 and I-8091 for function testing or pulse feedback by I-8090 encoder card.

18.4: Software

I/O connection:

The “**I-8091A**” connected on the I/O connection window contains 11 digital input channels.



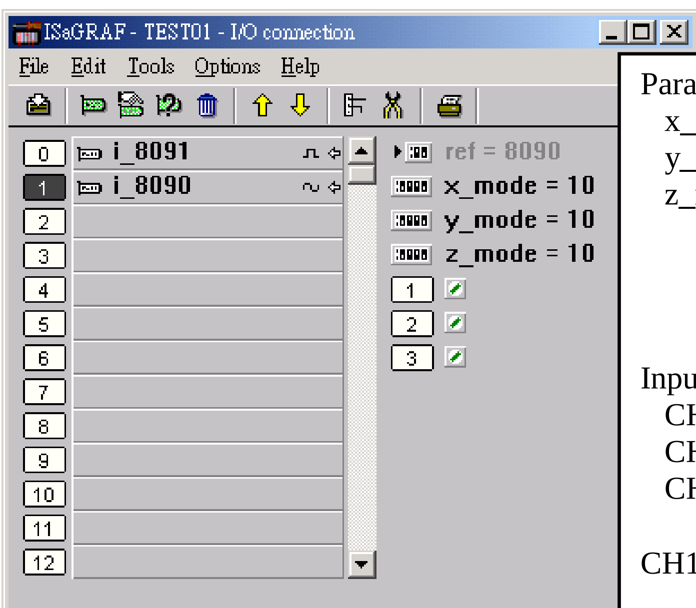
The “NO_OR_NC” parameter can beset as
0: Normal Open
1: Normal close.

Input Channel:
CH1 : EMG, emergency stop
CH2 : /FFEF, FIFO is empty or not, TRUE: empty
CH3 : /FFFF, FIFO is full or not, TRUE: full

CH4 : LS11, Left limit swtch of X-axis
CH5 : LS14, Right limit swtch of X-axis
CH6 : ORG1, Original position swtch of X-axis
CH7 : XSTOP, Stop or not of X-axis, TRUE: stop

CH8 : LS21, Left limit swtch of Y-axis
CH9 : LS24, Right limit swtch of Y-axis
CH10 : ORG2, Original position swtch of Y-axis
CH11 : YSTOP, Stop or not of Y-axis, TRUE: stop

I-8090 contains 3 analog input channels.



Parameter:
x_mode : integer counting mode of X-axis
y_mode : integer counting mode of Y-axis
z_mode : integer counting mode of Z-axis
00: quadrant counting mode
10: CW/CCW counting mode
20: pulse/direction counting mode

Input Channel:
CH1 : encorder value of X-axis
CH2 : encorder value of Y-axis
CH3 : encorder value of Z-axis

CH1 to CH3 are signed 32-bit integer format

Setting commands:

M_regist Register one I-8091

In order to distinguish more than one I-8091 card in I-8417/8817/8437/8837 platform, the I-8091 cards should be registrated before using it. This command will assign a card number = "card_NO_" to I-8091 card at that "address_" . If there is no I-8091 at the given address, this command will return FALSE.

Note: If using "I_8091A" rather than "I_8091" on the I/O connection window, user don't need to call "m_regist" & "m_s_nc", they are ignored. The card_NO of "I-8091A" is equal to its slot No. I-8xx7: 0 ~ 7. W-8xx7: 1 ~ 7.

Parameters:

card_NO_	integer	valid is 0 ~ 19.
address_	integer	the plugged slot address of the i8091 card slot 0: 16#80 slot 1: 16#A0 slot 2: 16#C0 slot 3: 16#E0 slot 4: 16#140 slot 5: 16#160 slot 6: 16#180 slot 7: 16#1A0

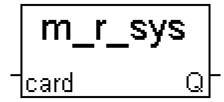
Return:

Q_	boolean	TRUE: Ok , FALSE: Fail

Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

```
(* declaration:   INIT as boolean <internal> and has initial value of TRUE      *)
   (*  TMP as boolean <internal>                                              *)
(* cardNO as integer <internal> and has intial value of 1          *)
   (* Do some init setting at 1st scan cycle *)
   if INIT then
         INIT := FALSE;
         TMP := M_regist(cardNO,16#80);        (* plug i8091 in slot 0 *)
         TMP := M_r_sys(cardNO);          (* reset i8091's setting *)
         TMP := M_s_var(cardNO,4,2,5,100);
         TMP := M_s_dir(cardNO,0,0);           (* Normal direction *)
         TMP := M_s_mode(cardNO,1,1);          (* pulse_dir mode *)
         TMP := M_s_serv(cardNO,1,1);          (* X & Y server ON *)
         TMP := M_s_nc(cardNO,0);         (* Normal open *)
   end_if;
```

M_r_sys Reset all setting



To reset I-8091 card, this command will terminate the running command in I-8091 card. User can use this command as software emergency stop. This command also will clear all of setting, so, all I-8091 card's parameter should be set again.

Parameters:

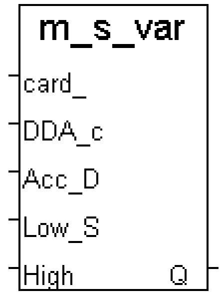
card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19

Return:

Q_	boolean	always return TRUE.

Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

M_s_var Set motion system parameters



To set DDA cycle, accelerating/decelerating speed, low speed and high speed value.

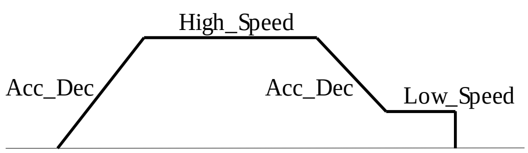
Parameters:

card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19
DDA_cycle_	integer	DDA cycle , valid is 1 ~ 254
Acc_Dec_	integer	Acc/Dec speed , valid is 1 ~ 200
Low_Speed_	integer	low speed , valid is 1 ~ 200 , Low_Speed_ >= Acc_Dec_
High_Speed_	integer	high speed , Low_Speed_ <= High_Speed <= 2047

Return:

Q_	boolean	always return TRUE.

Note:
The lower “DDA_cycle_” is given, the smaller delay time between /ORG1 ON and /X_STOP ON (or /ORG2 ON and /Y_STOP ON) when using M_hsporg & M_lsporg command. For ex, DDA_cycle_ set to 4, the delay time is about 5 to 13 ms.



Restriction:

$1 \le DDA\_cycle \le 254$

$1 \le Acc\_Dec \le 200$

$1 \le Low\_Speed \le 200$

$Low\_Speed \le High\_Speed \le 2047$

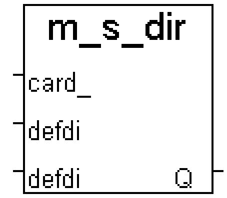
Low_Speed >= Acc_Dec

Default value
DDA_cycle = 10
Acc_Dec = 1
Low_Speed = 10
High_Speed = 100

Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

```
TMP := M_s_var(1, 5, 2, 10, 150);
(* DDA_cycle = 5      --> DDA period = (5+1)*1.024ms = 6.144ms
   Acc_Dec = 2        --> Acc/Dec speed = 2/(6.144ms)^2 = 52981 p/s^2
  Low_Speed = 10      --> low speed = 10/6.144ms = 1628pps
  High_Speed = 150    --> high speed = 150/6.144ms = 24414pps *)
```

M_s_dir Define output direction of axes



Sometimes, the output direction of X-axis, Y-axis is undesired direction due to the motor's connection or gear train. In order to unify the output direction as shown in Fig.(5) and Fig.(6). Where CW/FW direction is defined as toward outside from motor, CCW/BW direction is defined as toward inside from motor. This command provide parameters to define the rotating direction of motor.

Parameters:

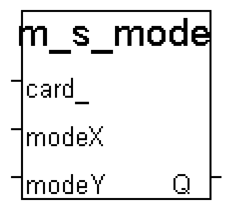
card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19
defdirX_	integer	X axis direction definition , valid is 0 ~ 1
defdirY_	integer	Y axis direction definition , valid is 0 ~ 1 0: normal direction, 1: reverse direction

Return:

Q_	boolean	always return TRUE.

Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

M_s_mode Set output mode

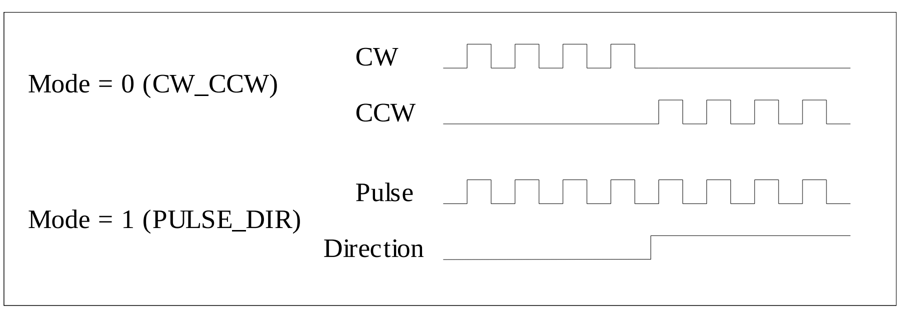


Parameters:

card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19
modeX_	integer	X axis mode, valid is 0 ~ 1
modeY_	integer	Y axis mode, valid is 0 ~ 1 0: CW_CCW, 1: PULSE_DIR

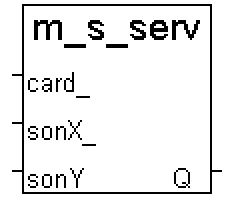
Return:

Q_	boolean	always return TRUE.



Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

M_s_serv Set servo ON/OFF



Parameters:

card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19
sonX_	integer	X axis servo/hold on switch , valid is 0 ~ 1
sonY_	integer	Y axis servo/hold on switch , valid is 0 ~ 1

0: OFF, 1: ON

Return:

Q_ boolean always return TRUE.

Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

M_s_nc Set N.O. / N.C.

To set all of the following limit switches as N.C.(normal close) or N.O.(normall open). If set as N.O., those limit switches are active low. If set as N.C., those limit switches are active high. The auto-protection will automatically change the judgement whatever it is N.O. or N.C..

Limit switches: ORG1, LS11, LS14, ORG2, LS21, LS24, EMG.

Note: If using “I_8091A” rather than “I_8091” on the I/O connection window, user don’t need to call “m_regist” & “m_s_nc”, they are ignored. The card_NO of “I-8091A” is equal to its slot No. I-8xx7: 0 ~ 7. W-8xx7: 1 ~ 7.

Parameters:

card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19
sw_	integer	0: N.O. (default) , 1: N.C.

Return:

Q_ boolean always return TRUE.

Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

Note: If using “I_8091A” in the ISaGRAF IO connection window, there is a “NO_OR_NC” parameter can be set to define as 0:Normal Open , 1:Normal Close. So user no more need to call this “m_s_nc” function if using “I_8091A”.

Stop commands:

M_stpx Stop X axis

Parameters:
card_NO_ integer the card No. has been set by **M_regist**, valid is 0 ~ 19

Return:
Q_ boolean always return TRUE.

Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

M_stpy Stop Y axis

Parameters:
card_NO_ integer the card No. has been set by **M_regist**, valid is 0 ~ 19

Return:
Q_ boolean always return TRUE.

Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

M_stpall Stop X & Y axes

This command will stop X & Y axes and clear all of commands pending in the FIFO.

Parameters:
card_NO_ integer the card No. has been set by **M_regist**, valid is 0 ~ 19

Return:
Q_ boolean always return TRUE.

Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

Simple motion commands:

M_lsporg Low speed move to ORG

Low speed move , and stop when **ORG1/ORG2** limit switch is touched.

Parameters:

card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19
DIR_	integer	0: CW , 1: CCW
AXIS_	integer	1: X axis , 2: Y axis

Return:

Q_	boolean	always return TRUE.

M_hsporg High speed move to ORG

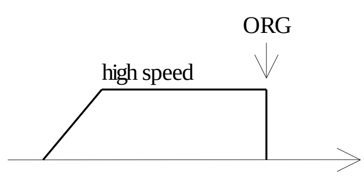
High speed move , and stop when **ORG1/ORG2** limit switch is touched.

Parameters:

card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19
DIR_	integer	0: CW , 1: CCW
AXIS_	integer	1: X axis , 2: Y axis

Return:

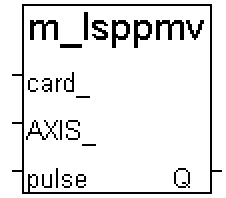
Q_	boolean	always return TRUE.



Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

Note:
The lower "DDA_cycle_" is given, the smaller delay time between /ORG1 ON and /X_STOP ON (or /ORG2 ON and /Y_STOP ON) when using M_hsporg & M_lsporg command. For ex, DDA_cycle_ set to 4, the delay time is about 5 to 13 ms.

M_lsppmv Low speed pulse move



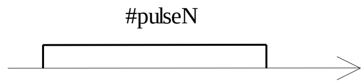
Low speed move a specified “pulse”

Parameters:

card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19
AXIS_	integer	1: X axis , 2: Y axis
Pulse_	integer	number of pulse to move. if > 0, move toward CW/FW dir. if < 0, move toward CCW/BW dir.

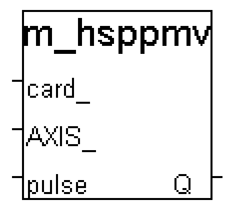
Return:

Q_	boolean	always return TRUE.



Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

M_hsppmv High speed pulse move



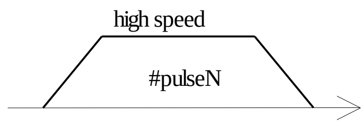
High speed move a specified “pulse”

Parameters:

card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19
AXIS_	integer	1: X axis , 2: Y axis
Pulse_	integer	number of pulse to move. if > 0, move toward CW/FW dir. if < 0, move toward CCW/BW dir.

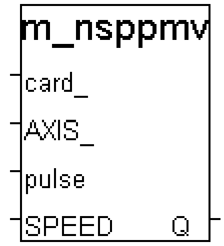
Return:

Q_	boolean	always return TRUE.



Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

M_nsppmv Normal speed pulse move



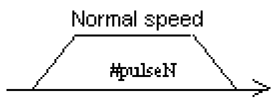
Normal speed move a specified “pulse”

Parameters:

card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19
AXIS_	integer	1: X axis , 2: Y axis
Pulse_	integer	number of pulse to move. if > 0, move toward CW/FW dir. if < 0, move toward CCW/BW dir.
SPEED_	integer	Speed, low speed <= SPEED_ <= high speed

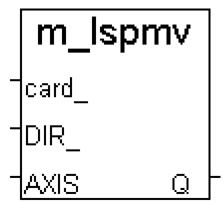
Return:

Q_	boolean	always return TRUE.



Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

M_lspmv Low speed move



Low speed move toward the direction specified. It can be stop by **M_stpx** or **M_stpy** or **M_stpall** command

Parameters:

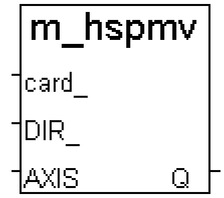
card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19
DIR_	integer	direction. 0: CW , 1: CCW
AXIS_	integer	1: X axis , 2: Y axis

Return:

Q_	boolean	always return TRUE.

Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

M_hspmv High speed move



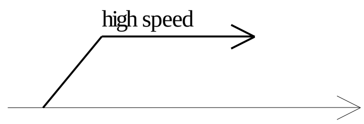
High speed move toward the direction specified. It can be stop by **M_stpx** or **M_stpy** or **M_stpall** command

Parameters:

card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19
DIR_	integer	direction. 0: CW , 1: CCW
AXIS_	integer	1: X axis , 2: Y axis

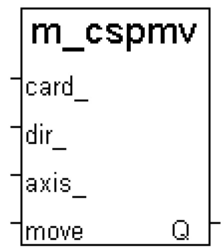
Return:

Q_	boolean	always return TRUE.



Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

M_cspmv Change speed move



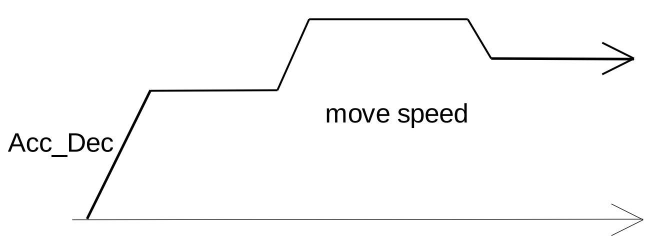
This command will accelerate/decelerate the selected axis's motor to the "move_speed". This command can be continuously send to I-8091 to dynamicly change speed. The rotating motor can be stop by the command **M_stpx**, **M_stpy**, **M_stpall**, or **M_slwstp**

Parameters:

card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19
dir_	integer	direction. 0: CW , 1: CCW
axis_	integer	1: X axis , 2: Y axis
move_speed_	integer	0 < move_speed_ <= 2040

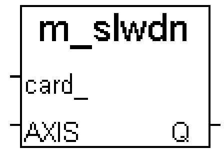
Return:

Q_	boolean	always return TRUE.



Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

M_slwdn Slow down to low speed



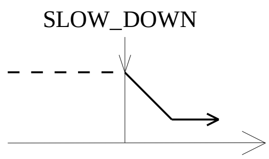
To decelerate to slow speed until **M_stpx** or **M_stpy** or **M_stpall** is executed.

Parameters:

card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19
AXIS_	integer	1: X axis , 2: Y axis

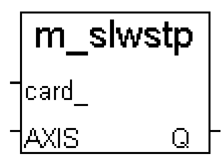
Return:

Q_	boolean	always return TRUE.



Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

M_slwstp Slow down to stop



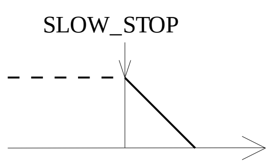
To decelerate to stop.

Parameters:

card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19
AXIS_	integer	1: X axis , 2: Y axis

Return:

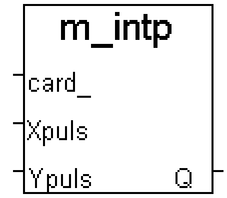
Q_	boolean	always return TRUE.



Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

Interpolation commands:

M_intp Move a short distance on X-Y plane



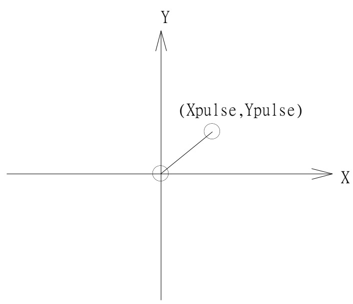
This command will move a short distance (interpolation short line) on X-Y plane. This command provided a method for user to generate an arbitrary curve on X-Y plane.

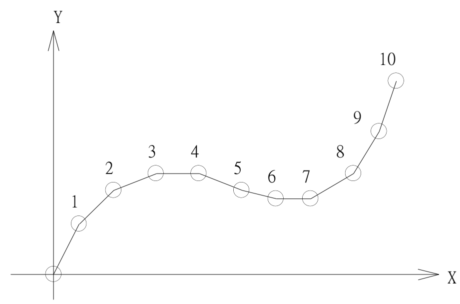
Parameters:

card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19
Xpulse_	integer	-2047 <= Xpulse_ <= 2047
Ypulse_	integer	-2047 <= Ypulse_ <= 2047

Return:

Q_	boolean	always return TRUE.





Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

NOTE:
For a lot of **M_intp** call set at the same time, please check if the FIFO is not full. Call it if FIFO is not full. FIFO indicator is a Digital Input resides at CH3 of I-8091.

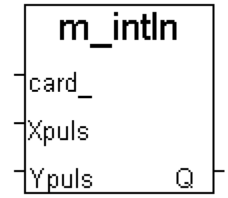
I-8091 D/I channel on ISaGRAF I/O connection window:

CH1 : EMG, emergency stop
CH2 : /FFEF, FIFO is empty or not, TRUE: empty
CH3 : /FFFF, FIFO is full or not, TRUE: full

CH4 : LS11, Left limit swtch of X-axis
CH5 : LS14, Right limit swtch of X-axis
CH6 : ORG1, Original position swtch of X-axis
CH7 : XSTOP, Stop or not of X-axis, TRUE: stop

CH8 : LS21, Left limit swtch of Y-axis
CH9 : LS24, Right limit swtch of Y-axis
CH10 : ORG2, Original position swtch of Y-axis
CH11 : YSTOP, Stop or not of Y-axis, TRUE: stop

M_intln   Move a long distance on X-Y plane



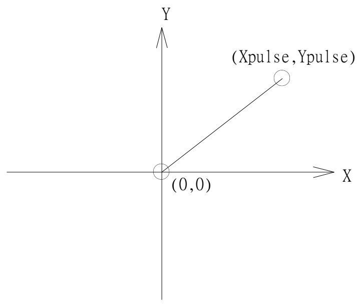
This command will move a long distance (interpolation line) on X-Y plane. The CPU on I-8091 card will generate a trapezoidal speed profile of X-axis and Y-axis, and execute interpolation by way of DDA chip.

Parameters:

card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19
Xpulse_	integer	-524287 <= Xpulse_ <= 524287
Ypulse_	integer	-524287 <= Xpulse_ <= 524287

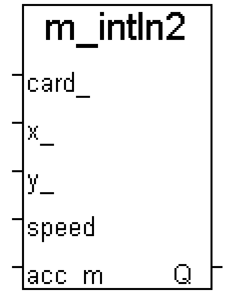
Return:

Q_	boolean	always return TRUE.



Example: I-8417/8817/8437/8837: demo_46, demo_27, demo_28
W-8337/8737: wdemo_26, wdemo_27, wdemo_28, wdemo_29

M_intln2 Move a long distance on X-Y plane



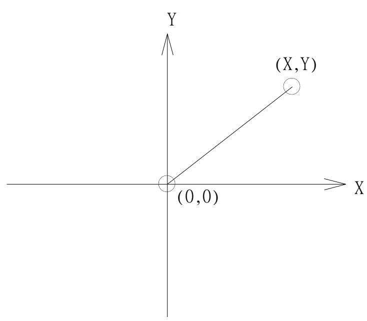
This command will move a long interpolation line on X-Y plane. It will automatically generate a trapezoidal speed profile of X-axis and Y-axis by state-machine-type calculation method.

Parameters:

card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19
x_, y_	integer	end point relate to present position
speed_	integer	0 ~ 2040
acc_mode_	integer	0: enable acceleration/deceleration profile 1: disable acceleration/deceleration profile

Return:

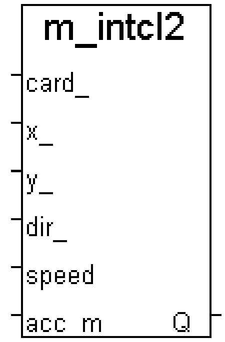
Q_	boolean	always return TRUE.



NOTE:

1. Only one of **M_intln2**, **M_intcl2** & **M_intar2** command can be called at one time, the other motion moving commands related to the same I-8091 card should not be called unless it is completed. (Please use **M_intstp** to test command of **M_intln2**, **M_intcl2** & **M_intar2** completed or not).
2. One controller can only drive one I-8091 to move by **M_intln2** , **M_intcL2** , **M_intar2** command. Two or more I-8091 cards in the same controller to use **M_intln2** , **M_intcL2** , **M_intar2** at the same time is not possible.

M_intcl2 Move a circle on X-Y plane



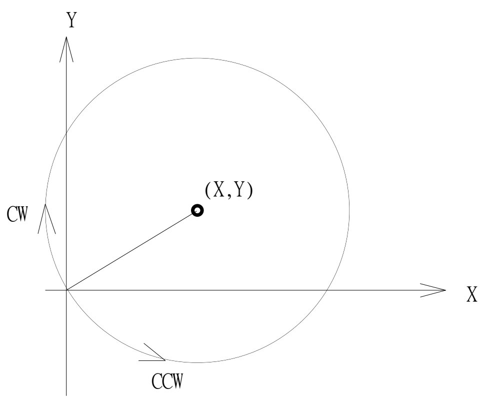
This command will generate an interpolation circle on X-Y plane. It will automatically generate a trapezoidal speed profile of X-axis and Y-axis by state-machine-type calculation method.

Parameters:

card_NO_	integer	the card No. has been set by **M_regist**, valid is 0 ~ 19
x_, y_	integer	center point of circle relate to present position
dir_	integer	moving direction. 0: CW , 1: CCW
speed_	integer	0 ~ 2040
acc_mode_	integer	0: enable acceleration/deceleration profile 1: disable acceleration/deceleration profile

Return:

Q_	boolean	always return TRUE.

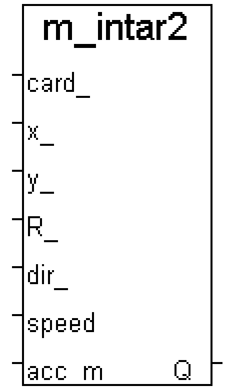


where radius = sqrt(X^2 + Y^2)

NOTE:

1. Only one of **M_intln2**, **M_intcl2** & **M_intar2** command can be called at one time, the other motion moving commands related to the same I-8091 card should not be called unless it is completed. (Please use **M_intstp** to test command of **M_intln2**, **M_intcl2** & **M_intar2** completed or not).
2. One controller can only drive one I-8091 to move by **M_intln2** , **M_intcL2** , **M_intar2** command. Two or more I-8091 cards in the same controller to use **M_intln2** , **M_intcL2** , **M_intar2** at the same time is not possible.

M_intar2 Move a arc on X-Y plane



This command will generate an interpolation arc on X-Y plane. It will automatically generate a trapezoidal speed profile of X-axis and Y-axis by state-machine-type calculation method.

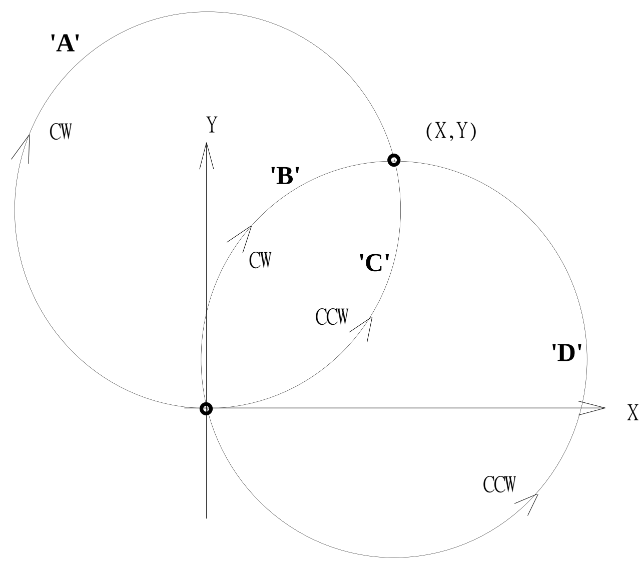
Parameters:

card_NO_ integer the card No. has been set by **M_regist**, valid is 0 ~ 19
x_, y_ integer end point of arc relate to present position
R_ integer radius of arc, if > 0, the arc < 180 degree, if < 0, the arc > 180 degree
R_ must > (square root of (X_*X_+Y_*Y_)) / 2
dir_ integer moving direction. 0: CW , 1: CCW
speed_ integer 0 ~ 2040
acc_mode_ integer 0: enable acceleration/deceleration profile
1: disable acceleration/deceleration profile

Return:

Q_ boolean always return TRUE.

R	dir	path of curve
R>0	CW	'B'
R>0	CCW	'C'
R<0	CW	'A'
R<0	CCW	'D'



NOTE:
1. Only one of **M_intln2**, **M_intcl2** & **M_intar2** command can be called at one time, the other motion moving commands related to the same I-8091 card should not be called unless it is completed. (Please use **M_intstp** to test command of **M_intln2**, **M_intcl2** & **M_intar2** completed or not).
2. One controller can only drive one I-8091 to move by **M_intln2** , **M_intcL2** , **M_intar2** command. Two or more I-8091 cards in the same controller to use **M_intln2** , **M_intcL2** , **M_intar2** at the same time is not possible.

M_intstp Test X-Y plane moving command

To test the below 3 commands completed or not.

M_intln2 , M_intcL2 , M_intar2

It will return FALSE for interpolation command completed while return TRUE for busy - not completed yet.

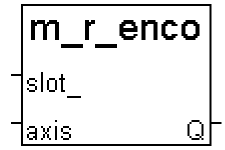
Return:

Q_ boolean TRUE: busy , FALSE: completed

NOTE:

1. Only one of **M_intln2**, **M_intcl2** & **M_intar2** command can be called at one time, the other motion moving commands related to the same I-8091 card should not be called unless it is completed. (Please use **M_intstp** to test command of **M_intln2**, **M_intcl2** & **M_intar2** completed or not).
2. One controller can only drive one I-8091 to move by **M_intln2** , **M_intcL2** , **M_intar2** command. Two or more I-8091 cards in the same controller to use **M_intln2** , **M_intcL2** , **M_intar2** at the same time is not possible.

I-8090 encorder commands:



M_r_enco Reset I-8090's encorder value to 0

Parameters:

slot_	integer	the slot No. where the i8090 is plugged, 0 ~ 7
axis_	integer	1: x-axis, 2: y-axis, 3: z-axis

Return:

Q_	boolean	always return TRUE.

Example: demo_27, demo_28, demo_46

Chapter 19. Ethernet Communication and Security

ISaGRAF PAC has built in high flow-rate protection in the Ethernet communication. This protect the ISaGRAF program running well when TCP SYN, TCP FIN, … flood attack happens.

19.1: Ethernet Security

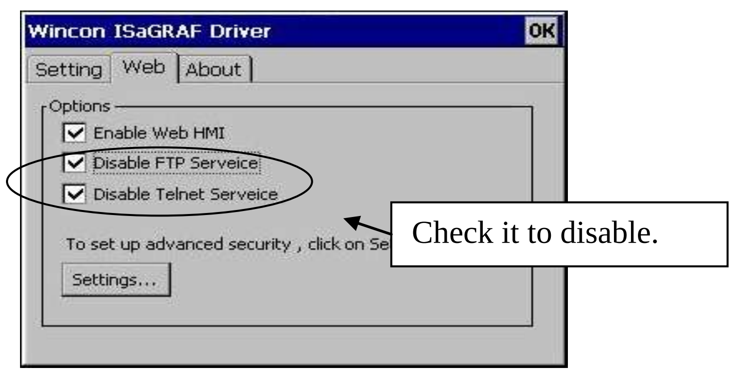
There are some ways user can get access to the XP-8xx7-Atom-CE6, XP-8xx7-CE6, WP-8xx7, WP-5xx7, VP-25W7/23W7, W-8xx7 via its Ethernet port.

1. Using Modbus TCP protocol at port No.= 502. (ISaGRAF and other HMI can do this)
2. Using ftp (for example, keyin “ftp://10.0.0.103” on the Internet Explorer)
3. Using telent (for example, keyin “telnet 10.0.0.103 in the “command” window)
4. Using the Web Server (The Web HMI does)

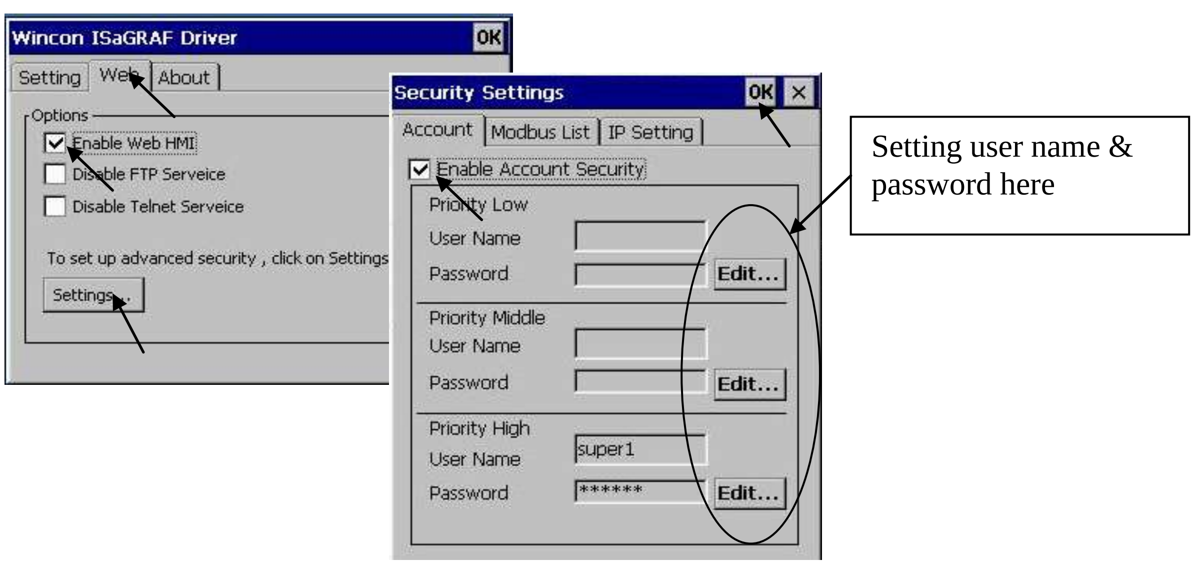
Note:

1. **While for I-8xx7, I-7188EG, µPAC-7186EG, µPAC-5xx7, VP-2117 and iP-8x47 only item 1 is possible.**
2. **If the controller is WP-8xx7, WP-5xx7, XP-8xx7-CE6, XP-8xx7-Atom-CE6, VP-25W7/23W7, W-8xx7, when using “ftp”, “telnet”, “Web HMI” & “Modbus TCP/IP”, please connect your PC/HMI to its “LAN1” port, and please use “NS-205” or “NS-208” Ethernet switch.**

For safety, recommend to disable item 2 and 3 at run time for WinCon.

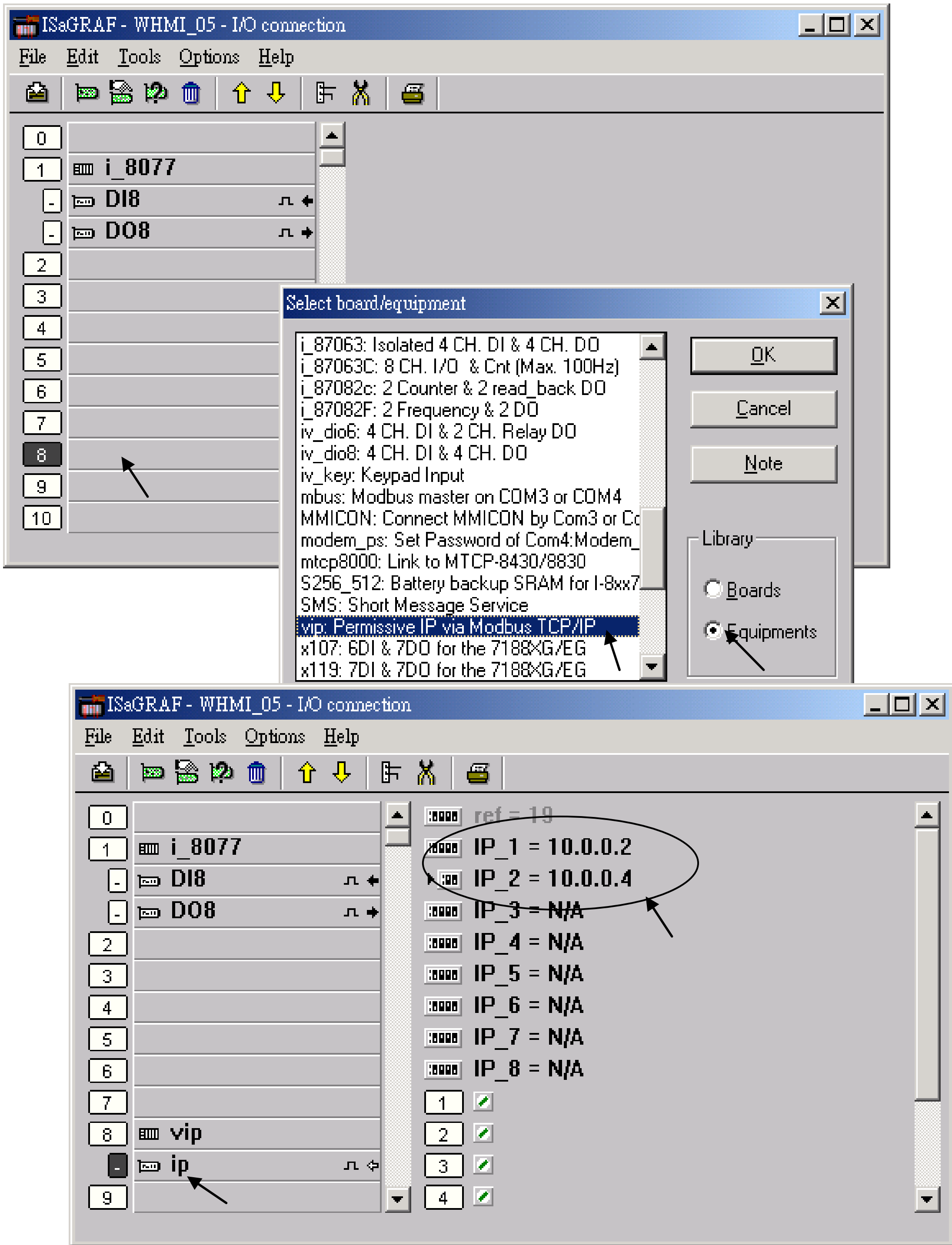


And about item 4, please set proper username & password for the WinCon Web HMI.



About item 1, user may set up to 8 IP address for ISaGRAF or other HMI to get access to the controller via the Modbus TCP/IP protocol as below.

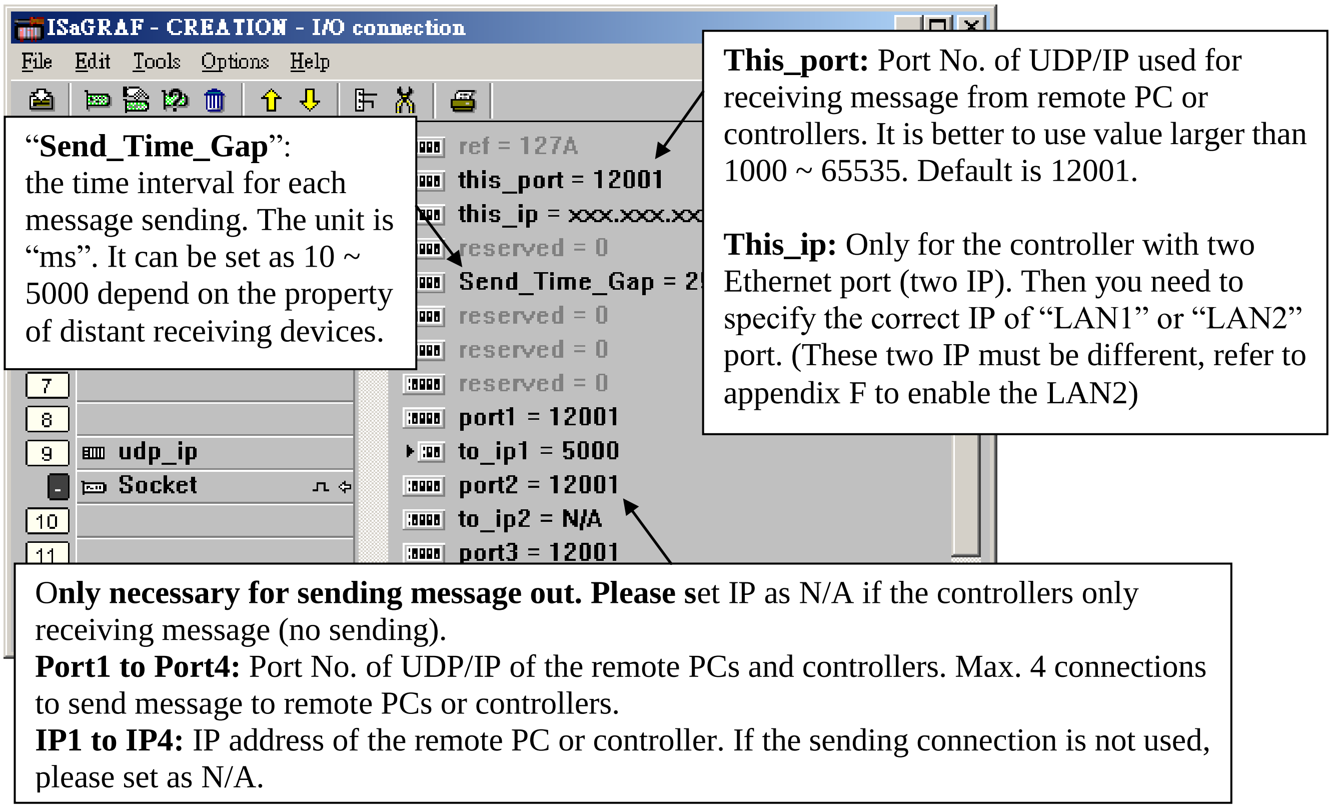
On the IO connection window of ISaGRAF, please connect “vip” and entering the IP which can get access to the controller via Modbus TCP/IP protocol. If “vip” is not connected, any remote IP can get access to your controller via Modbus TCP/IP protocol. If “vip” is connected and No IP is entered (all assigned as “N/A”), No HMI and ISaGRAF can get access to it anymore.



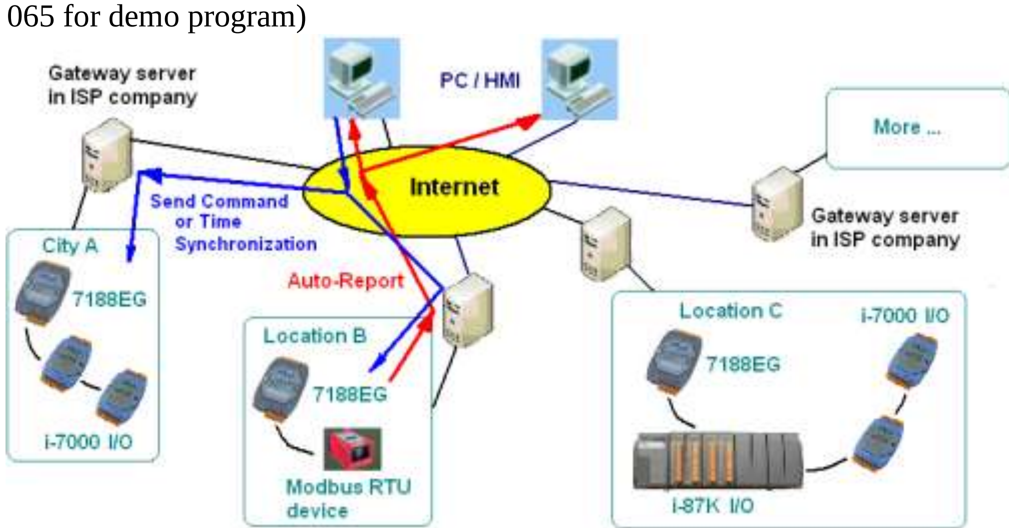
19.2: Delivering Message via UDP

When using WP-8xx7, WP-5xx7, XP-8xx7-CE6, XP-8xx7-Atom-CE6 and iP-8x47, you may connect ethernet cable at "LAN1" or "LAN2" port. Please use NS-205/208 Ethernet switch for them.

Since I-7188EG: driver 2.18, I-8437/8337:driver 3.20 , W-8xx7 :driver 3.37 or later Ver. Please connect "udp_ip" before using "udp_recv" and "udp_send" functions.



I-7188EG, µPAC-7186EG, uPAC-5xx7, I-8437-80/8837-80, VP-2117, iP-8xx7, WP-8x47, WP-5xx7, XP-8xx7-CE6, XP-8xx7-Atom-CE6 and VP-2xW7 can use its UDP IP to auto-report "acquisition data" or "control data" to local or remote Internet PC / HMI . The PC / HMI can also send "control commands" or "time synchronization command" to the controller. **The advantage is every Controller in different location doesn't need a fixed "Internet IP"** (Please refer to www.icpdas.com – FAQ – Software – ISaGRAF – 065 for demo program)



UDP_Recv:

To receive message from remote PCs or controllers, please use “udp_recv” function.

For example:

```
(* test if message is coming from UDP *)
(* Msg1 is declared as Message variable *)
(* if return = '' (empty message), that means no message coming *)
Msg1 := udp_recv( ) ;
```

Note:
1. The receiving buffer size for WP-8xx7, WP-5xx7, XP-8xx7-CE6, XP-8xx7-Atom-CE6, VP-25W7/23W7 and W-8xx7 is 8192 bytes - include one extra message end: 1 byte in each message. While for I-7188EG, µPAC-7186EG, µPAC-5xx7 and iP-8x47 is 2048 bytes.
2. If the receiving buffer is full, the oldest received message will be overwritten.

UDP_Send:

To send message to remote PCs or controllers, please use “udp_send” function.

For example:

```
(* TMP is declared as Internal / Boolean *)
(* 1st parameter: To which connection - defined in IO connection "udp_ip", can be 1 to 4 *)
(* 2nd parameter: the message to send out *)
(* Return True:Ok,  False: sending buffer is full or connection not defined well in “udp_ip” *)
TMP := udp_send(1, ‘Alarm1’ );
```

Note:
1. The sending buffer size for WP-8xx7, WP-5xx7, XP-8xx7-CE6, XP-8xx7-Atom-CE6, VP-25W7/23W7 and W-8xx7 is 2048 bytes - include extra message end: 1 byte. That means max. 2048 bytes in one ISaGRAF PLC scan can be sent to remote IP. While for iP-8x47, I-7188EG, µPAC-7186EG, µPAC-5xx7, VP-2117 and I-8x37-80 is 1024 bytes.
2. Please do not send lots of bytes in one PLC scan cycle too frequently. The controller driver will actually send only one message out each PLC scan when there is message in the sending buffer. For example, if there is 100 messages in the sending buffer, the controller will send over these 100 message in 100 PLC scan cycles.

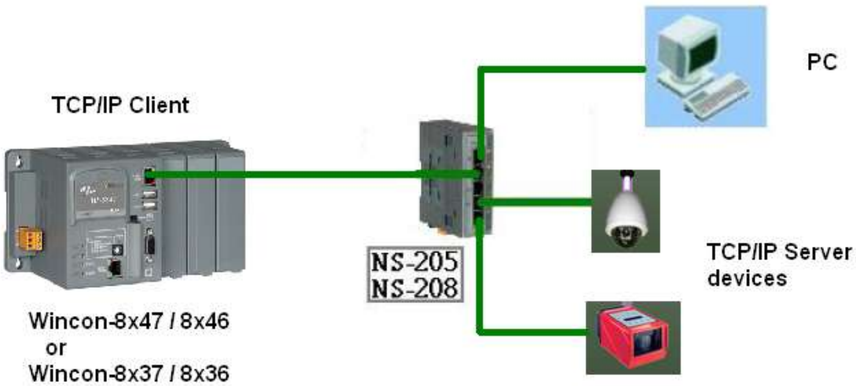
Example:
Please refer to WinCon CD-ROM:\napdos\isagraf\wincon\demo**wdemo_19 & Wdemo_20** or ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/demo/

If you can not find “udp_ip” , “udp_recv” and “udp_send” in your ISaGRAF, please visit http://www.icpdas.com/products/PAC/i-8000/isagraf.htm to download “ICP DAS Utilities For ISaGRAF.zip”. Then, run the “Setup.exe” to restore it to your ISaGRAF installed in PC.

Test Utility: there is a useful utility “udp.exe” can be used on PC to receive message coming from UDP IP. Please run it in command shell. W-8xx7 CD-ROM:\napdos\isagraf\some_utility\udp_test\udp.exe or ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/some_utility/udp_test

19.3: To Send/ Receive/ Auto-Report data via TCP/IP

The WinCon-8xx7 / 8xx6 supports TCP/IP Client since its driver version of 3.37. Please visit http://www.icpdas.com/products/PAC/i-8000/isagraf-link.htm for new released driver. And please also visit http://www.icpdas.com/products/PAC/i-8000/isagraf.htm to download the "ICP DAS Utilities For ISaGRAF.zip" to remove it and restore it to your ISaGRAF installed in PC. Then you will find IO connection - "Tcp_Clie" & c-function - "Tcp_send" & "Tcp_recv"

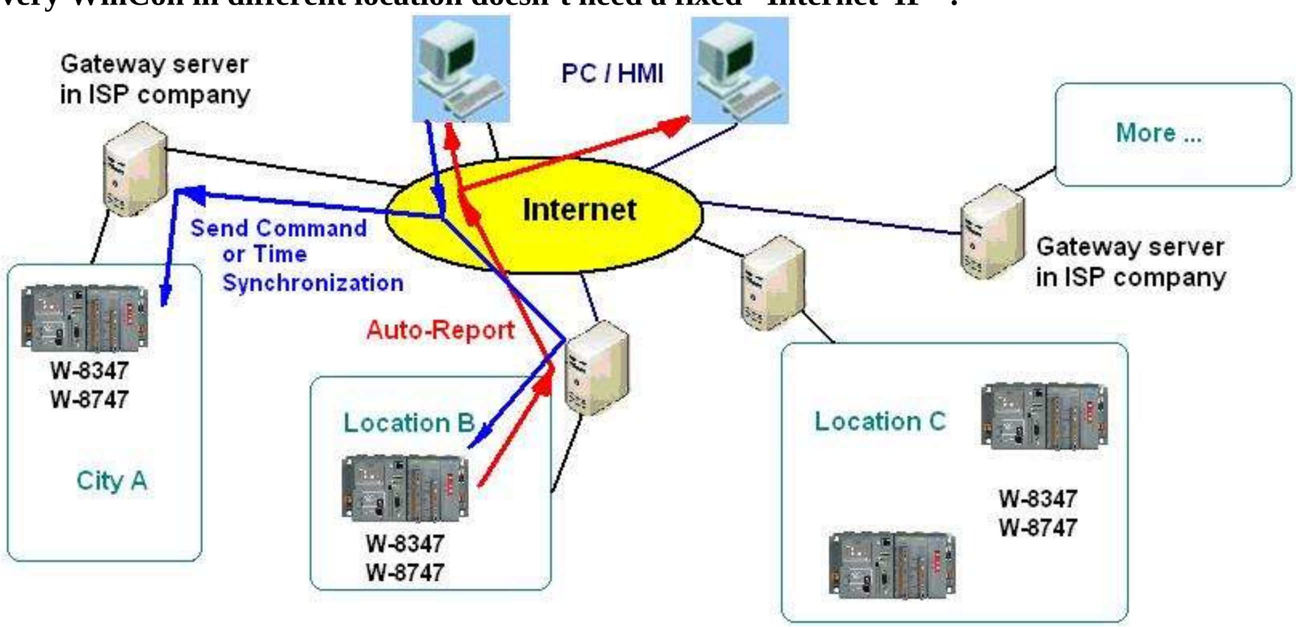


Note: The WP-8xx7, XP-8xx7-CE6, XP-8xx7-Atom-CE6, VP-25W7/23W7 support the TCP/IP client function.
The ISaGRAF demo program is Wdemo_32.pia & Wdemo_33.pia & Wdemo_60.pia (It can be download at ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/demo/)

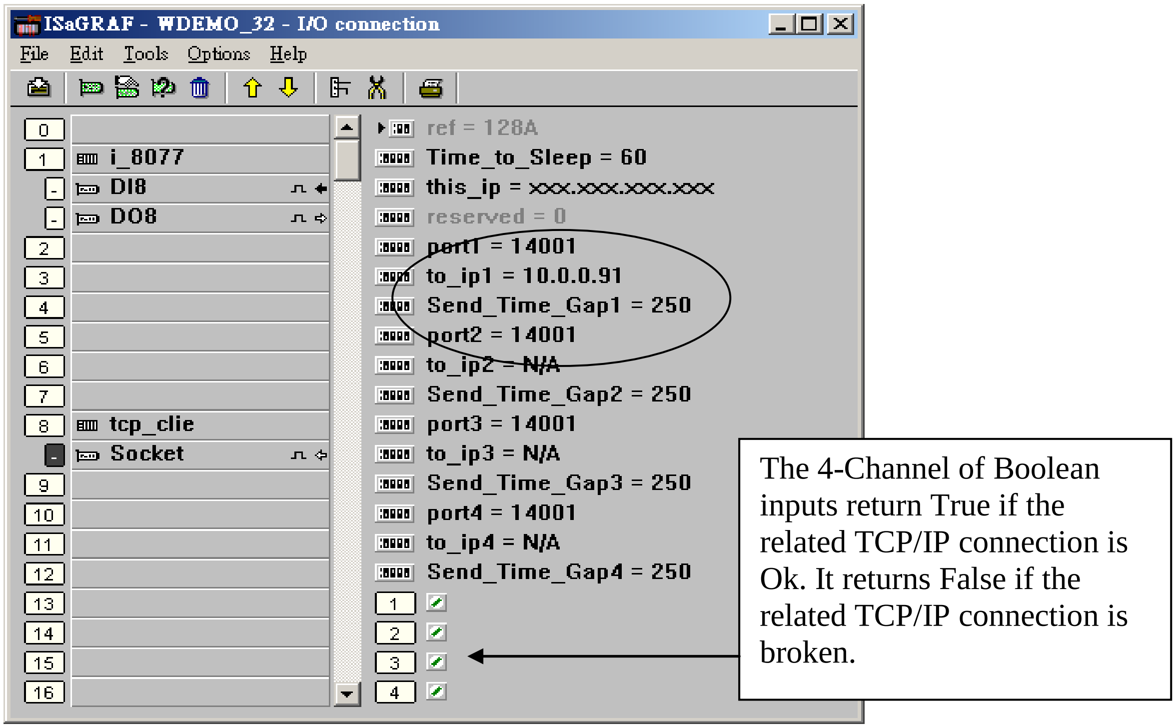
Note: The remote PC or device must support the TCP/IP Server function for connecting the TCP/IP Client –W-8xx7.

******** Useful Application ********

(Please refer to www.icpdas.com – FAQ – Software – ISaGRAF – 064 for demo program)
W-8347 and W-8747 can use its TCP / IP Client to auto-report "acquisition data" or "control data" to local or remote Internet PC / HMI . The PC / HMI can send back "control commands" or "time synchronization command" to the WinCon after receiving the data from WinCon. **The advantage is every WinCon in different location doesn't need a fixed "Internet IP" .**



To setup W-8xx7 as TCP/IP Client, please connect IO complex equipment - “Tcp_clie” first in the IO connection windows as below. Max 4 TCP/IP Client can be setup in one WinCon-8xx7.



The “**Time_to_sleep**” setting ranges from 10 to 600, or 0. unit is second. If setting as 0, The TCP/IP sending connection is always connected unless the TCP/IP has communication problem. Setting as 10 to 600 sec, means if no message is sending persist for such a long time, the connection will be disconnected (TCP/IP connection will be disconnected). If there is new message requested to send, the TCP/IP connection will be connected again.

The “**this_ip**” setting is only necessary if your WinCon is W-8x47 / W-8x46. Please enter the “IP address of the ethernet port in this controller”. For W-8x37/8x36 has only one Ethernet port (One IP address only), you don't need to specify it.

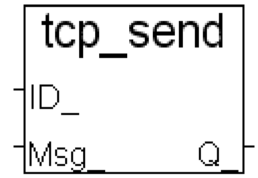
The “**port**?” specify the remote PC or device 's TCP/IP server's port number.

The “**to_ip**?” setting default is “N/A”, which means “Not Avaliable”. If setting an IP address to it, the related TCP/IP client will be setting up.

The “**Send_Time_Gap**?” setting is the minimum “Time Gap” in milli-second to send out each TCP/IP message one by one. For example, if setting as 250 , the 2nd message will not be sent out if the “Time Gap” since the first message sent out is less than 250 ms. The value can be 10 to 5000 ms depends on the remote device or PC 's TCP/IP server property.

After the “TCP_clie” is well setup. Use may call “Tcp_send” to send message out. To receive response from remote TCP/IP server, please call “Tcp_recv”

Tcp_send



TCP Client send message to remote PCs or device's TCP/IP server (via ethernet)
*** Target : WinCon-8xx6/8xx7 (since driver version of 3.36)

Note:
The sending buffer for WinCon is 4096 bytes. That means max. 4096 bytes in one PLC scan can be sent to remote IP. If sending buffer is full, the oldest mesage will be drop to release space for new "tcp_send( )" request.

parameter：

ID_	Integer	send to which connection, can be 1 to 4. The related "ip address" and "port No." is defined in "tcp_clie" (IO complex equipment)
Msg_	Message	The message to send

return value:

Q_	Boolean	True: send OK , False: parameter error(For ex, setting ID_ as 8) or related connection is not defined in IO connection - "Tcp_clie" .

Tcp_recv

TCP Client receive message from remote PC or device's TCP/IP server (via ethernet)
*** Target : WinCon-8xx6/8xx7 (since driver version of 3.36)

Note:
The receiving buffer size is 4096 bytes. If the receiving buffer is full, the oldest message will be drop to release space for receiving new coming data.

Parameter:

ID_	Integer	send to which "Tcp_Clie" connection, can be 1 to 4. The related "ip address" and "port No." is defined in IO connection: "Tcp_clie"

return value:

Msg_	Message	the received message. If Msg_ = '' (empty message), it means no message coming.

There is an useful “Tcp_Server” tool can be run in PC to simulate the TCP/IP server device. It resides at WinCon-8xx7 CD-ROM:\napdos\isagraf\some_utility\ (or visit ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/some_utility/ to download), please copy “Tcp_server folder to your PC”, then open a command prompt, and key-in “ tcp3 <port_No> “ , for example “tcp3 14001”

This utility wait TCP/IP Client (W-8xx7) requesting to connect, and then receive message from it, and then reply a same message back to the TCP/IP client.

The ISaGRAF demo program is Wdemo_32.pia & Wdemo_33.pia & Wdemo_60.pia (It can be download at ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/demo/)

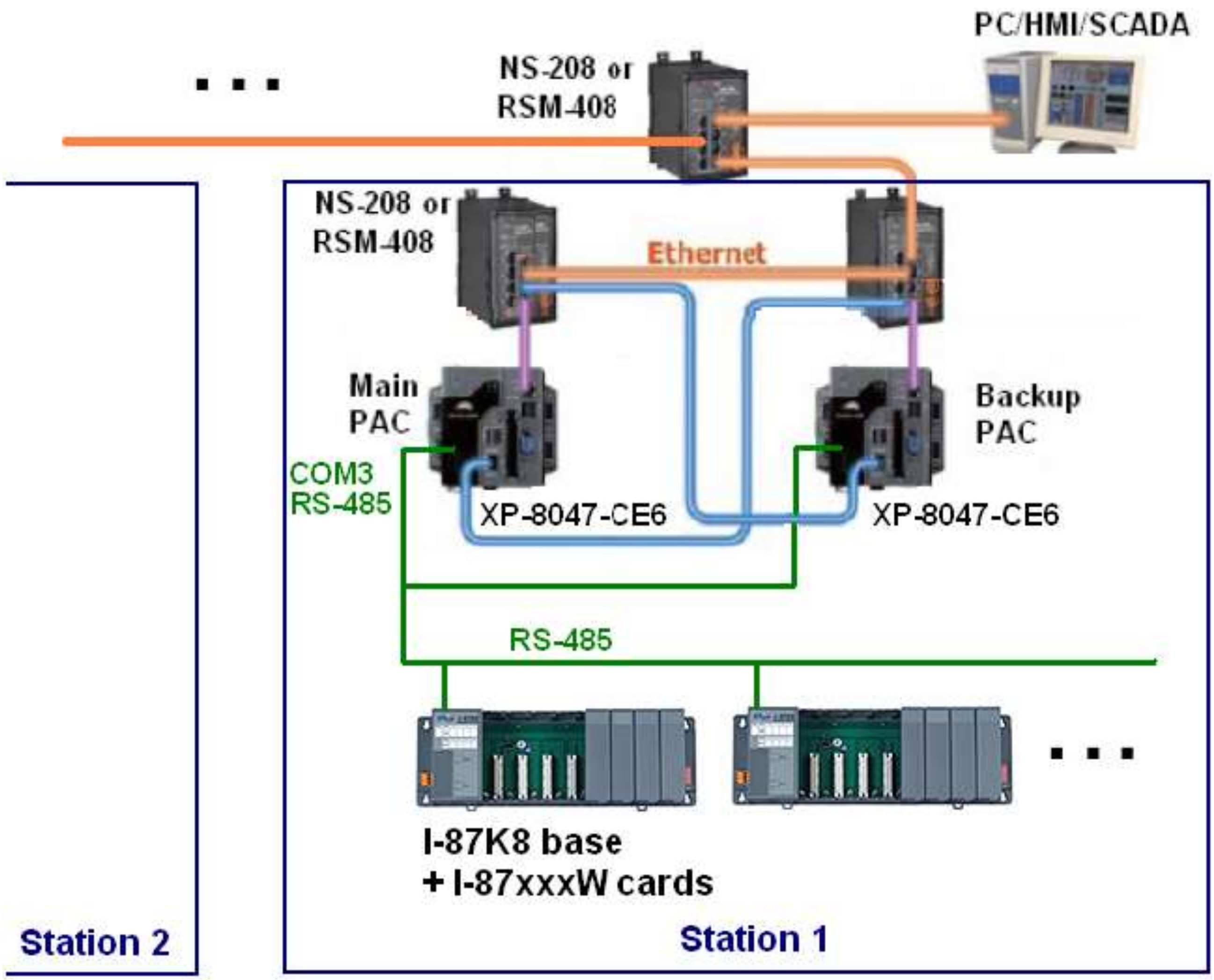
Chapter 20. Redundancy Solutions

20.1: XP-8xx7-CE6 Redundant System

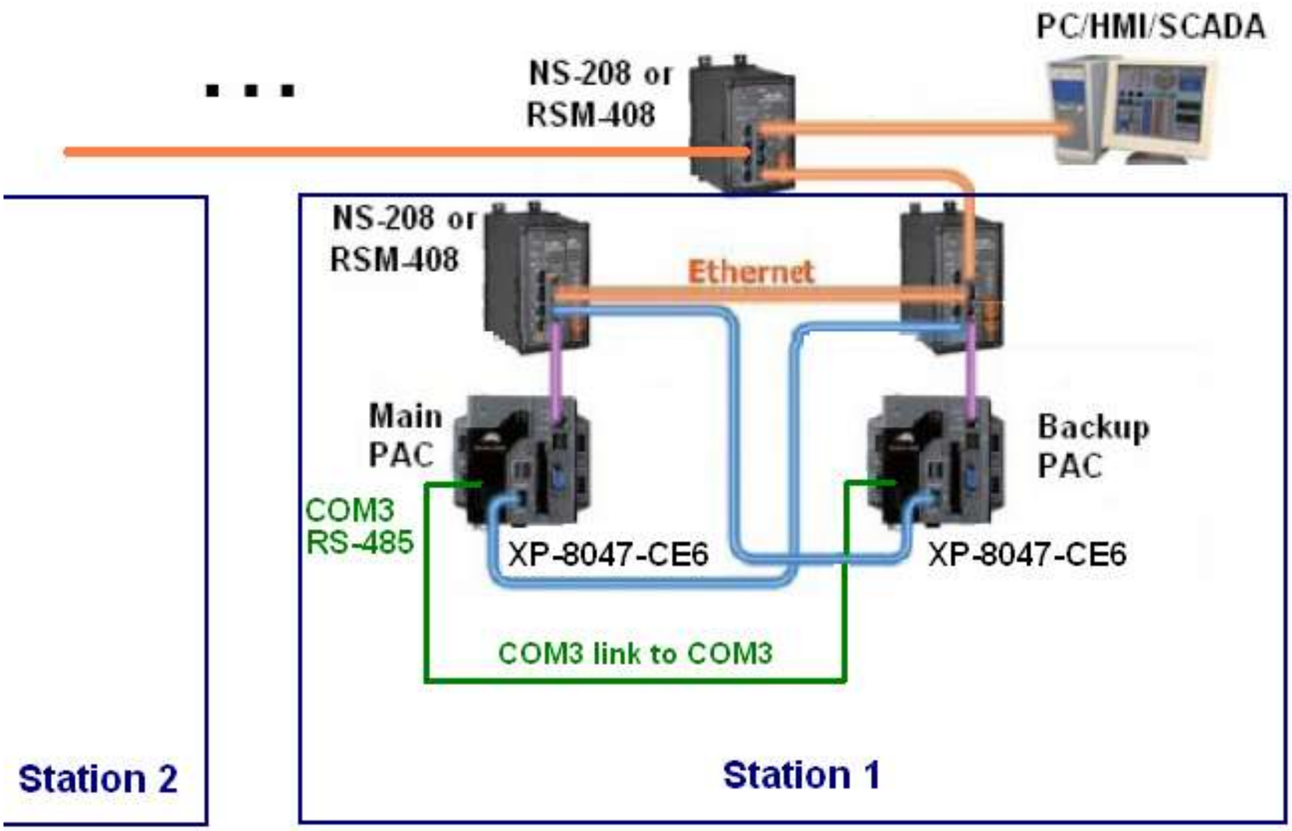
Please visit http://www.icpdas.com/faq/isagraf.htm > FAQ-138 for more information.

Advantage more than the WP-8xx7 redundant system, the PC/ HMI/ SCADA just need to connect to one IP address (the “active_IP1” address) of the XP-8xx7-CE6/ XP-8xx7-Atom-CE6 redundant system (The “Active_IP1” address will auto-switch to the active XP-8xx7-CE6/ XP-8xx7-Atom-CE6 's LAN1 or LAN2 port). On the other hand, the PC/ HMI/ SCADA need to connect to two IP address of the WP-8xx7 redundant system and they need auto-switch to the second IP (“Active_IP2”) if the first IP is disconnected. For many SCADA software, some are impossible and some are not easy (need to create two sets of data tag) to do that.

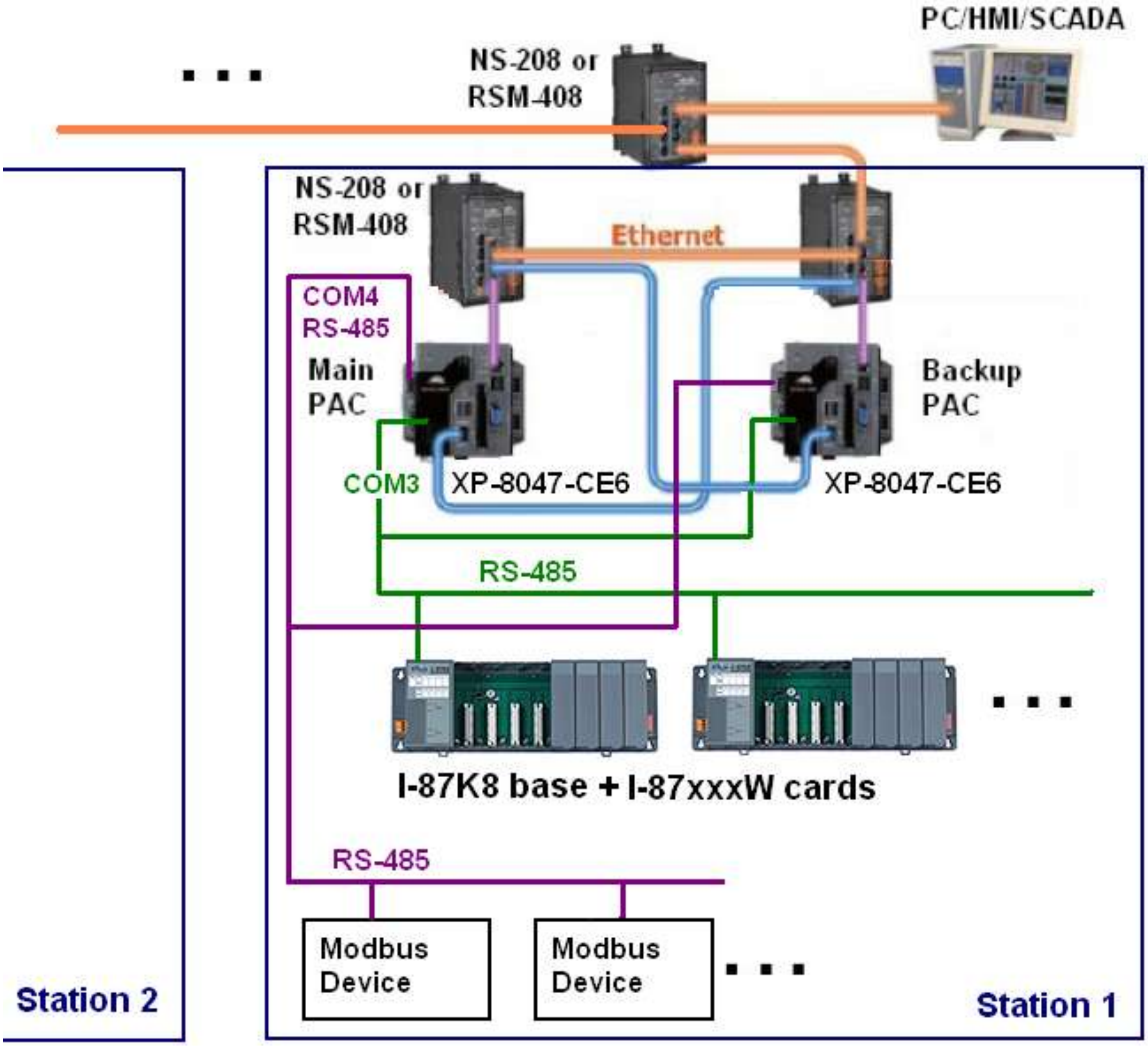
The first configuration is using two XP-8xx7-CE6/ XP-8xx7-Atom-CE6 PAC to connect one or more I-87K8 expansion base (each I-87K8 base can have max. eight I-87xxxW I/O cards in it). One or more stations can join together as the following figure. Each station contains one or two NS-208 (or RSM-208) and two XP-8xx7-CE6 controllers and one or more I-87K8 base with I-87xxxW I/O cards.



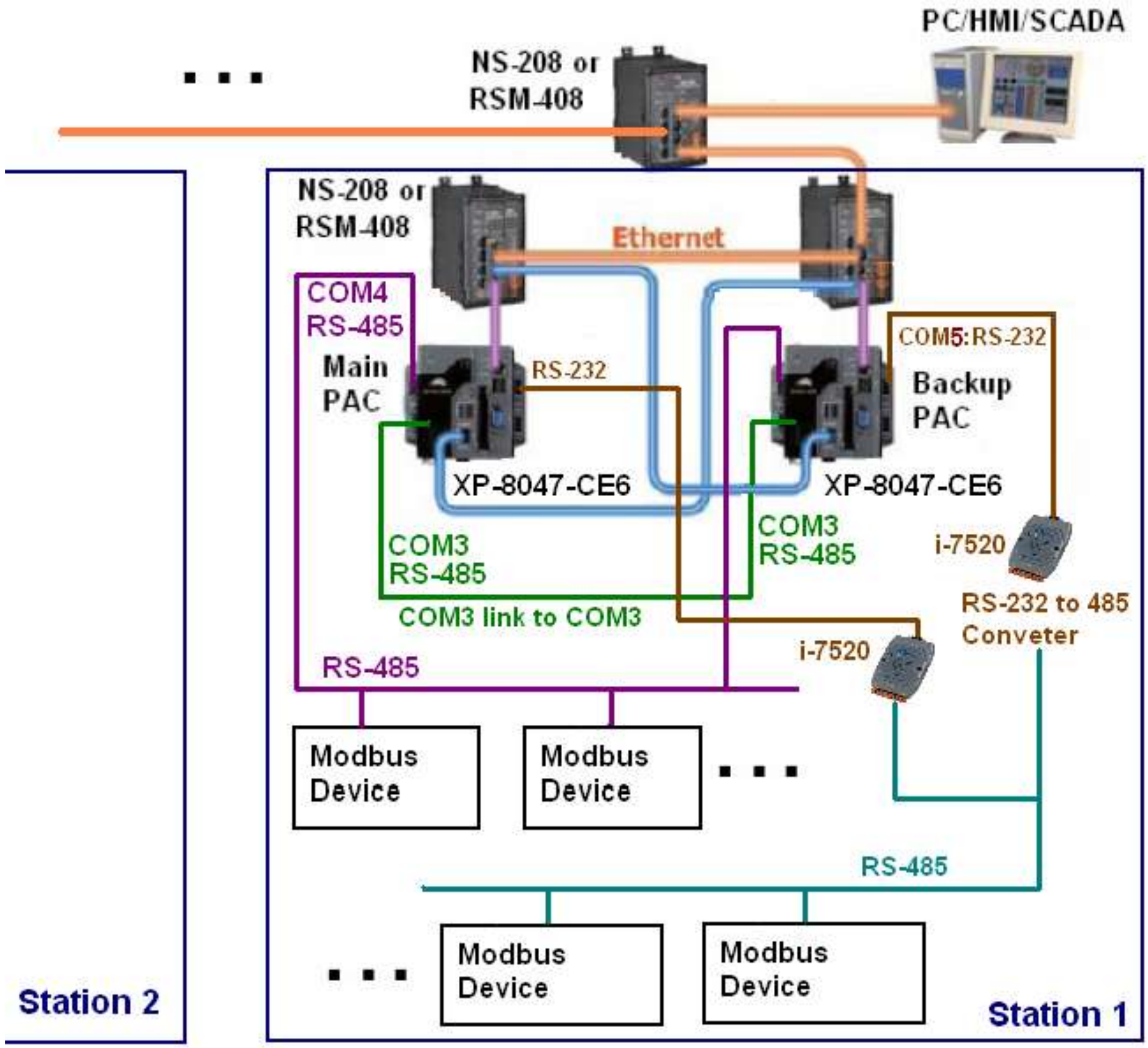
The second configuration is using two XP-8xx7-Atom-CE6/XP-8xx7-CE6 PAC without I/O or connects some other devices by serial ports (for example, using both XP-8xx7-Atom-CE6/XP-8xx7-CE6's COM4: RS-232 to link one I-7530 respectively to become CAN signal to connect other CAN/CANopen devices). Both PAC's COM3: RS-485 should connect to each other in this configuration. Can use only one NS-208 (or RSM-208) or two in each station.



The third configuration is using two XP-8xx7-Atom-CE6/XP-8xx7-CE6 PAC to connect one or more I-87K8 expansion bases and connect some Modbus RTU devices (or Modbus ASCII devices) by one or more serial ports. Can use only one NS-208 (or RSM-208) or two in each station.



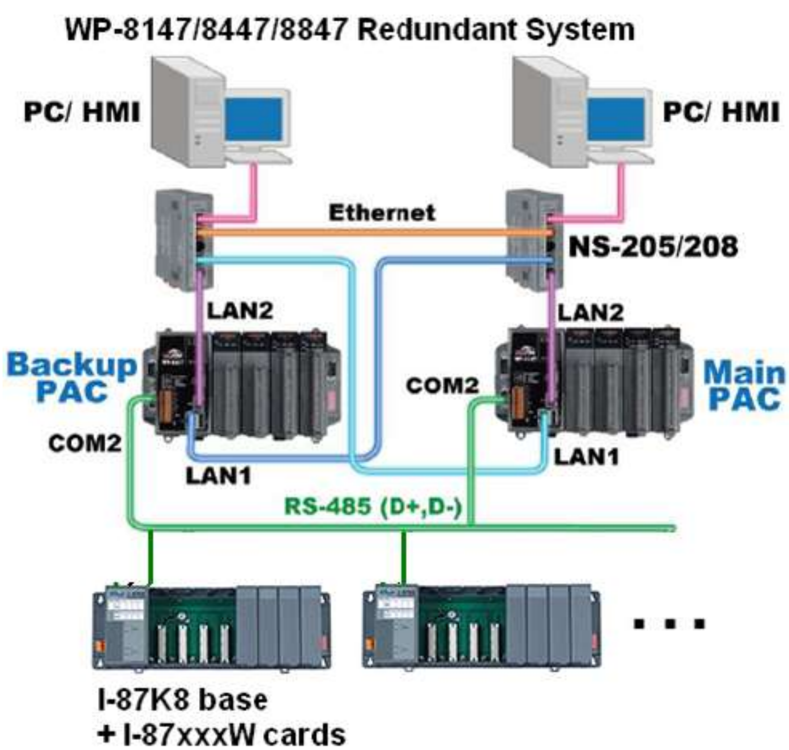
The fourth configuration is using two XP-8xx7-Atom-CE6/XP-8xx7-CE6 PAC to connect some Modbus RTU device (or Modbus ASCII devices). Both PAC's COM3: RS-485 should connect to each other in this configuration. Can use only one NS-208 (or RSM-208) or two in each station.



20.2: WP-8xx7 Redundant System

Please visit http://www.icpdas.com/faq/isagraf.htm > FAQ-093 for more information.

Advantage more than the WP-8xx7 redundant system, the PC/ HMI/ SCADA just need to connect to one IP address (the “active_IP1” address) of the XP-8xx7-CE6 redundant system (The “Active_IP1” address will auto-switch to the active XP-8xx7-CE6 's LAN1 or LAN2 port). On the other hand, the PC/ HMI/ SCADA need to connect to two IP address of the WP-8xx7 redundant system and they will auto-switch to the second IP (“Active_IP2”) if the first IP is disconnected. For many SCADA software, some are impossible and some are not easy (need to create two sets of data tag) to do that.



Chapter 21. Connecting M-7000 Series I/O Modules

ISaGRAF controllers support M-7000 remote RS-485 I/O since below driver version.

Controller	***Driver version***
W-8037 / 8337 / 8737 / 8036 / 8336 / 8736	3.35 or later version
W-8047 / 8347 / 8747 / 8046 / 8346 / 8746	3.35 or later version
I-8417 / 8817 / 8437 / 8837	3.19 or later version
I-7188EG / 7188EGD	2.17 or later version
I-7188XG / 7188XGD	2.15 or later version
µPAC-7186EG, µPAC-5xx7, iP-8xx7, VP-2117, WP-8xx7, WP-5xx7, VP-25W7/23W7, XP-8xx7-CE6, XP-8xx7-Atom-CE6	Since its released version

Please visit http://www.icpdas.com/products/PAC/i-8000/isagraf-link.htm to download them & follows steps listed in “ReadMe.txt” or “Update_w8xx7.pdf” to update them to your controller if your controller's driver is older.

The M-7000 series modules are RS-485 remote I/O modules which support Modbus RTU slave protocol. Please visit http://www.icpdas.com/products/Remote_IO/m-7000/m-7000_list.htm for more information.

User can write ISaGRAF program to support Modbus RTU Master protocol to connect to M-7000 I/Os. Please refer to Chapter 8 of the “ISaGRAF user's Manual”.

The Ethernet port of WP-8xx7, WP-5xx7, VP-2xW7, XP-8xx7-Atom-CE6 and XP-8xx7-CE6 suppport Modbus TCP Master, please refer to http://www.icpdas.com/faq/isagraf.htm > FAQ-113

If the Modbus device can’t be connected well, please refer to FAQ-075 for troubleshooting. (or set the specific “Delay_Time”)

The example program for connecting the M-7000 I/O modules are stored in the WP-8xx7 CD-ROM:\napdos\isagraf\wp-8xx7/demo

These programs all use COM3 port to connect to M-7000 I/O. You may change the “port_no” setting of the “mbus” to fit your controller.

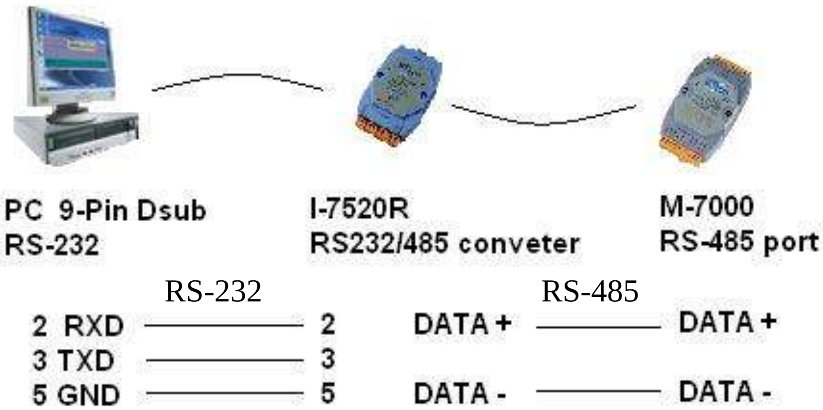
Wdemo_41	COM3 connecting 1: M-7053D (16-Ch. D/I) + 2:M-7045D (16-Ch. D/O)
Wdemo_42	COM3 connecting 1: M-7053D to get D/I counter value (16-bit, 0- 65535)
Wdemo_43	COM3 connecting 1: M-7017R (8-Ch. A/I) + 2:M-7024 (4-Ch. A/O)
Wdemo_44	COM3 connecting 1: M-7017RC (8-Ch. Current Analog Input)
Wdemo_45	COM3 connecting 1: M-7019R (8-Ch. Universal A/I, thermocouple or voltage input or current input) to get temperature value
Wdemo_46	COM3 connecting 1: M-7080 (2-Ch counter or frequency)

21.1: Using DCON utility to do initial setting for M-7000

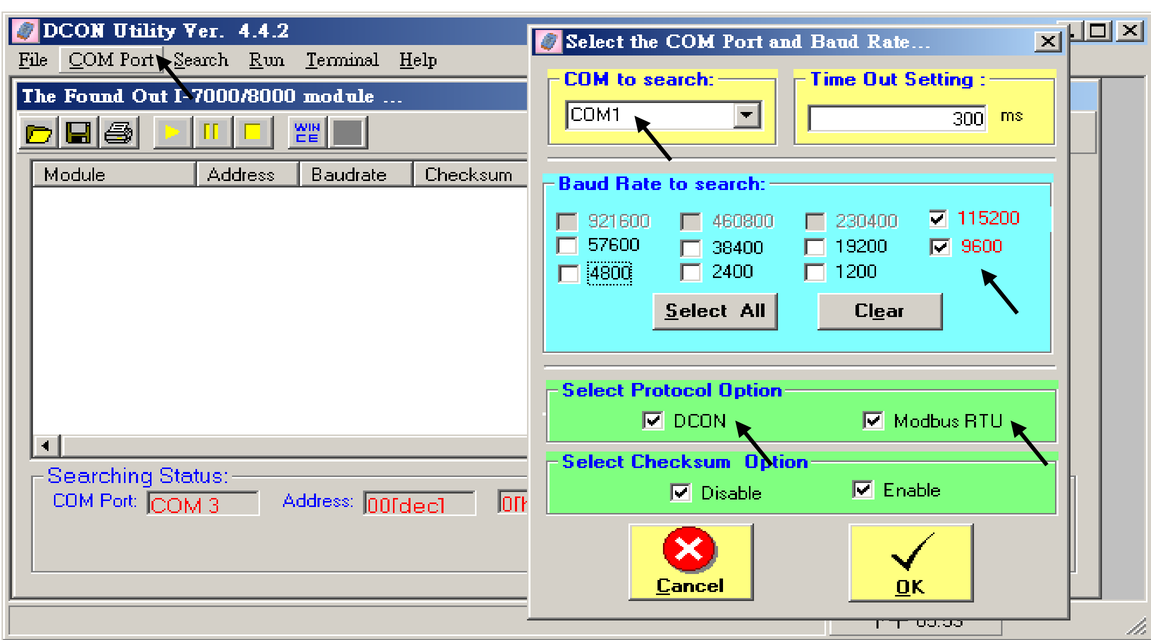
Before we starting at programming Modbus Master port, please run "DCON utility" to well configure M-7000's "Slave No" (or called Address), "Baudrate" for every D/I/O & A/I/O module and channel range or type setting for Analog input & output module. The "Procotol" setting should be "Modbus". You may install "Dcon Utility" from the I-8000 CD-ROM or visit http://www.icpdas.com/download/7000/7000.htm to download and then install it.

Steps to configure each M-7000 module.

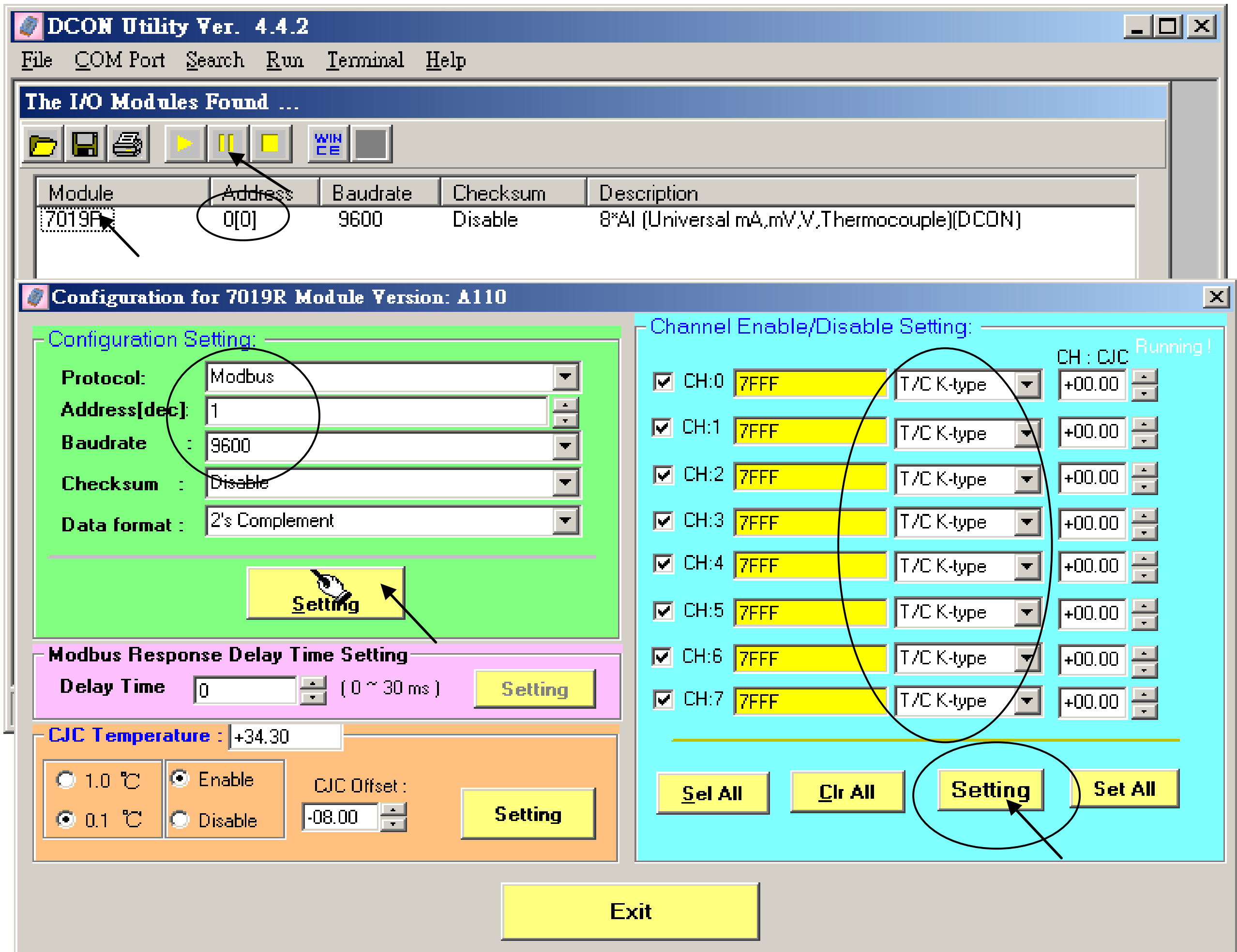
Step 1: Power off M-7000. Connect one RS-232 cable from PC's COM1 (or other COM port) to one RS-232/485 converter,for example I-7520R at http://www.icpdas.com/products/Industrial/communication_module/communication_list.htm , then connect this converter to the M-7000 module. **Please short "INIT*" to "GND"**. This means to make the M-7000 to be in initial state (Address will be 0, baudrate=9600). Some M-7000 module provide a "Init – Normal" dip switch on its back to replace the "INIT*" pin.



Step 2: Power on M-7000. Run "Dcon utility", click "COM Port" menu to select proper COM port, baudrate , check on "DCON" & "Modbus RTU", ... Then click "start Search" to search M-7000.



Step 3: The only one connected M-7000 should be found at Address=0 (because it is in initial state), Click "Stop" to stop searching when found it. Please set protocol as "Modbus", proper "Address" (Slave no) , "Baudrate" . And if the M-7000 is Analog I/O, please set proper type & range, then click on "Setting"



Step 4: Power off M-7000. Remove connection between "INIT*" and "GND" . **Then power it ON again . Run DCON utility to search & then check if the setting is correct or not**.
If the setting is not correct, modify them and click on "setting" again.

If this M-7000 Module is M-7041 or M-7044 or M-7050 or M-7053 or M-7060 or M-7063 or M-7065 (or M-7041D or M-7044D or M-7050D or M-7053D or M-7060D or M-7063D or M-7065D), please go to step 5. If the module is not in the above item numbers, then this M-7000 is well configured.

Note:

1. **Every M-7000 must be configured to a unique "Address number" (1 to 247) and the same "Baudrate" and other proper setting before using it.**
2. **User may refer to the attatched manual in the product box, or visit** http://www.icpdas.com/products/Remote_IO/m-7000/m-7000_list.htm **to get each M-7000 Module's Manual to find their "Analog Input Type and Data Formate Table" (Type code setting).**

Important Step5:

After the initial configuration is completed (Step 1 to 4), please give below Modbus command to below M-7000 modules 's Digital input channels to invert them.

01 46 29 01 (4-byte command, each byte is 2 Hex-number)

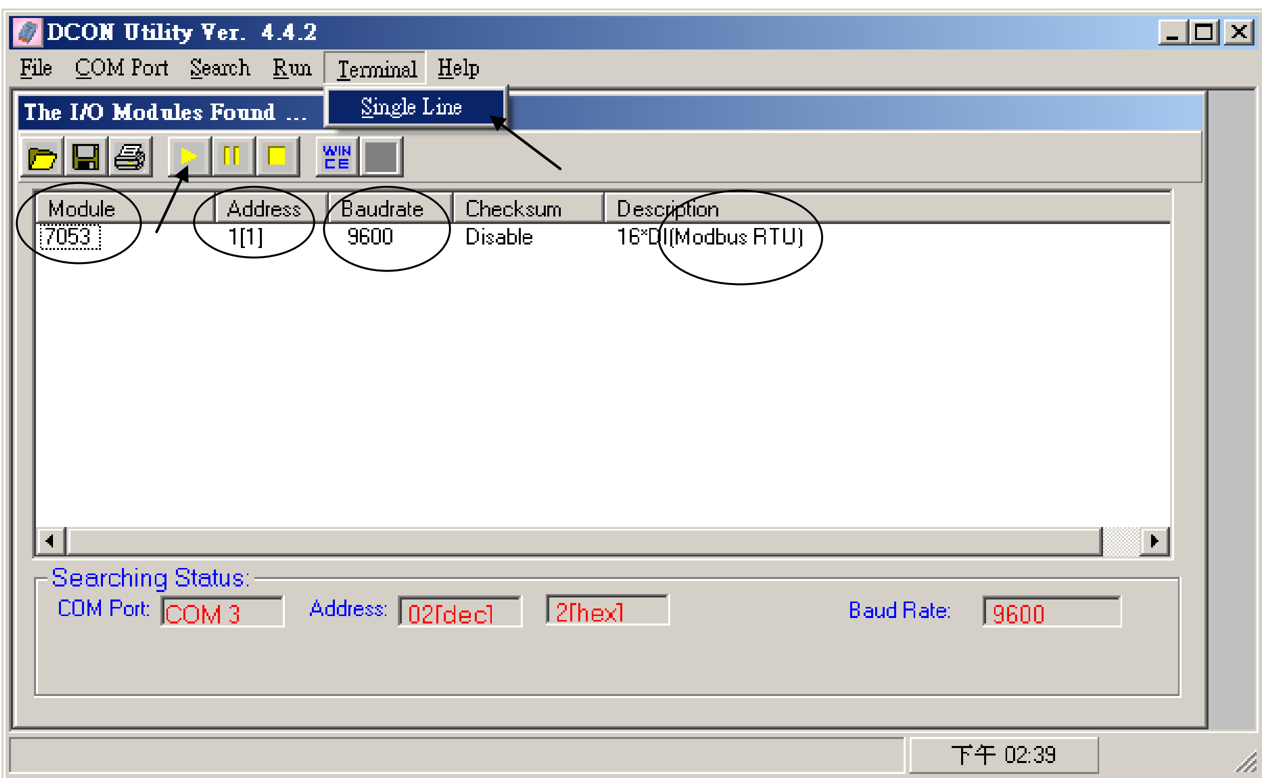
The first byte is the M-7000 Address number been set by DCON utility, it may be 01, 02, 03, ..., 0F, ... to F7 depends on your setting of the related M-7000. The other 3 bytes "46 29 01" should be always same.

M-7000 Modules should be inverted						
M-7041,	M-7044,	M-7050,	M-7053,	M-7060,	M-7063,	M-7065
M-7041D,	M-7044D,	M-7050D,	M-7053D,	M-7060D,	M-7063D,	M-7065D

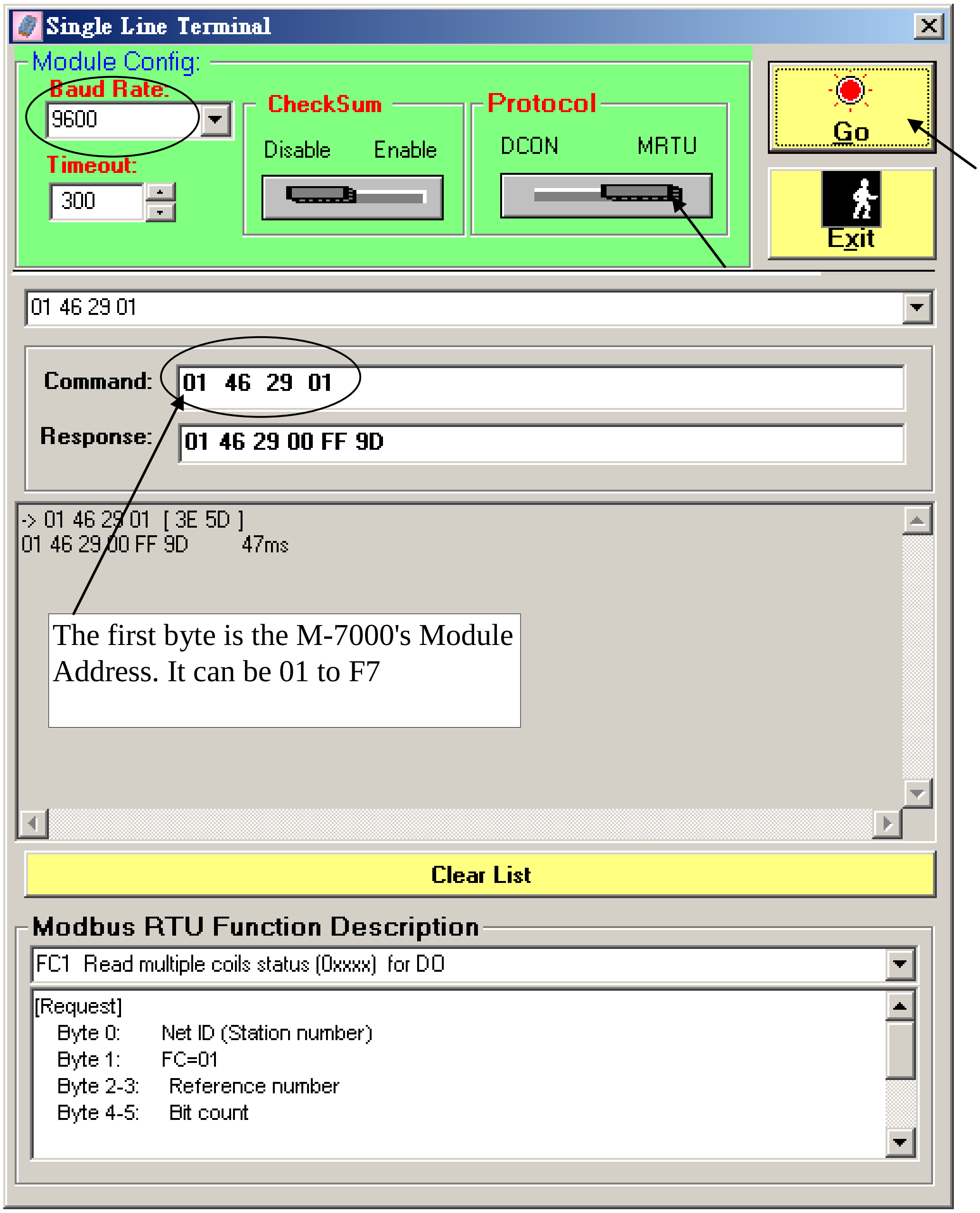
Please Do Not give the upper command to other M-7000 modules which are not in the above lists.

Steps to invert the digital input channels:

After Step 4 is finished, power on M-7041 or M-7044 or M-7050 or M-7053 or M-7060 or M-7063 or M-7065 again. Run DCON utility to search the module first. If the module is found. Stop search. Make sure the Module name is one of M-7041 or M-7044 or M-7050 or M-7053 or M-7060 or M-7063 or M-7065. Then goto "Terminal" - "Single Line"



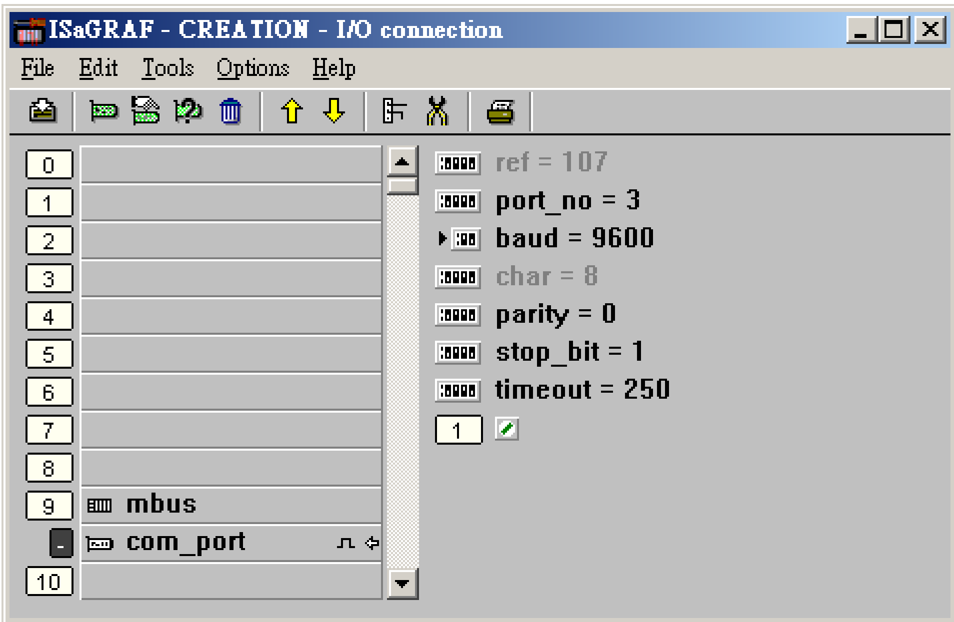
Select the correct baudrate, Protocol should be set to “MRTU”. Then type the inverted command as below, the first byte should be the Module's Address number. It can be 01 to F7. And then click “Go” . If the response is “01 46 29 ...” , it means command succeed. Power off this M-7000 modules. And it is well configured.



21.2: Writting program to connect to I-7000 modules

Important : If your M-7000 is **M-7041** or **M-7044** or **M-7050** or **M-7053** or **M-7060** or **M-7063** or **M-7065** (or **M-7041D** or **M-7044D** or **M-7050D** or **M-7053D** or **M-7060D** or **M-7063D** or **M-7065D**), please follow the former section's Step5 to invert their digital input channels.

To program Modbus RTU Master, please connect "mbus" in the ISaGRAF IO connection windows as below. Please set proper "port_no" , "baud" & "timeout". "timeout" setting default is 500 ms, you can specify 250 ms if connecting only M-7000 I/O modules.



Then please create an Ladder program or function block program to access to each M-7000 I/O channels. ICP DAS ISaGRAF controllers can access to M-7000 modules by using "Mbus_r" , "Mbus_r1" , "Mbus_b_w" , "Mbus_wb" & "Mbus_n_w" .

Mbus_R	1. Read max. 12 word-value (-32768 ~ +32767) using Modbus function call **4** to read M-7000 **Analog input** channels or read D/I counter value. And, it can also used to read six 32-bit int-value (-2,147,483,648 ~ +2, ,147,483,647) using "WD_LONG" function block to convert two Word to one 32-bit interger. 2. Read max.192 bit-value using Modbus function call **2** to read M-7000 **Digital input** channels. Using "WD_Bit" function block one Word to 16 Boolean-value.
Mbus_R1	Same as Mbus_R but with one extra setting – Period. (It can be set as 1 ~ 600) Read words or bits with a specified period time (unit is second)
MBUS_N_W	Write max. 4 word-value (-32768 ~ +32767) using Modbus function code **6** or **16** to witre M-7000 **Analog output** channels. (write 1 word using code **6**, write 2 ~ 4 words using code **16**)
MBUS_B_W	Write max. 4 bit-value using Modbus function code **5** or **15** to witre M-7000 **Digital output** channels. (write 1 bit using code **5**, write 2 ~ 4 bits using code **15**)
MBUS_WB	Write max. 16 bit-value using Modbus function code **15** to witre M-7000 **Digital output** channels.

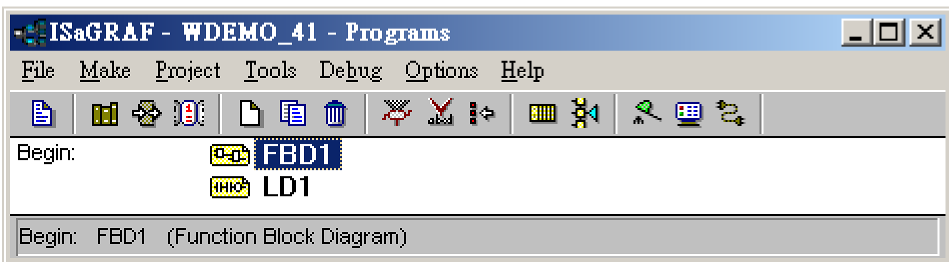
Example 41: Connecting 1: M-7053D (16-Ch. D/I) + 2: M-07045D (16-Ch. D/O)
(This example is "Wdemo_41").

Please follow former section 's step 1~ 5 to do the initial setting for the M-7053 module, and step 1 ~ 4 for the M-7045D module.

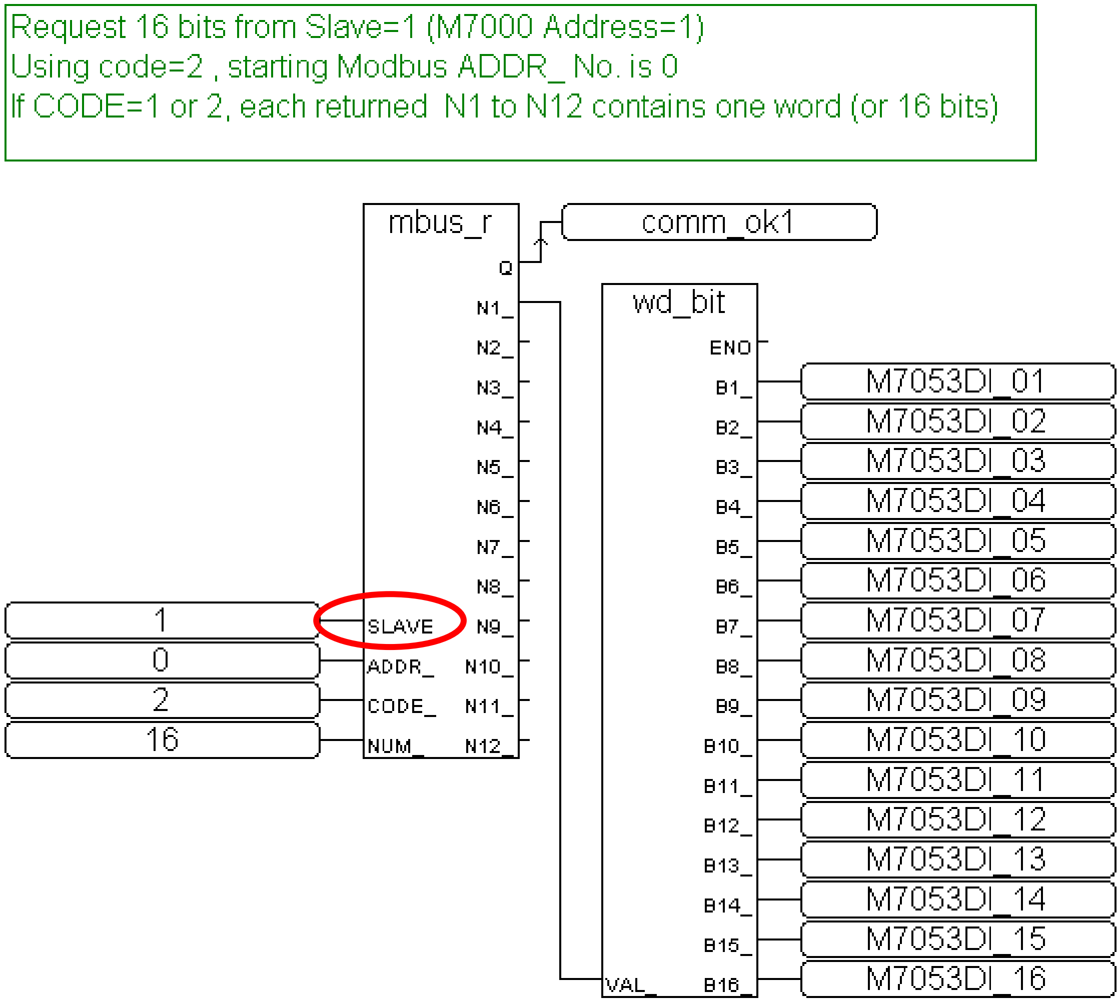
Variables:

Name	*Type*	*Attribute*	*Description*
comm_ok1	Bool	Internal	Communication state of the related M-7053D
comm_ok2	Bool	Internal	Communication state of the related M-7045D
M7053DI_01 to M7053DI_16	Bool	Internal	Total 16 boolean internal variables D/I Ch. 1 to 16 of M-7053D
M7045DO_01 to M7045DO_16	Bool	Internal	Total 16 boolean internal variables D/O Ch. 1 to 16 of M-7045D

Project: One function block program + one Ladder program

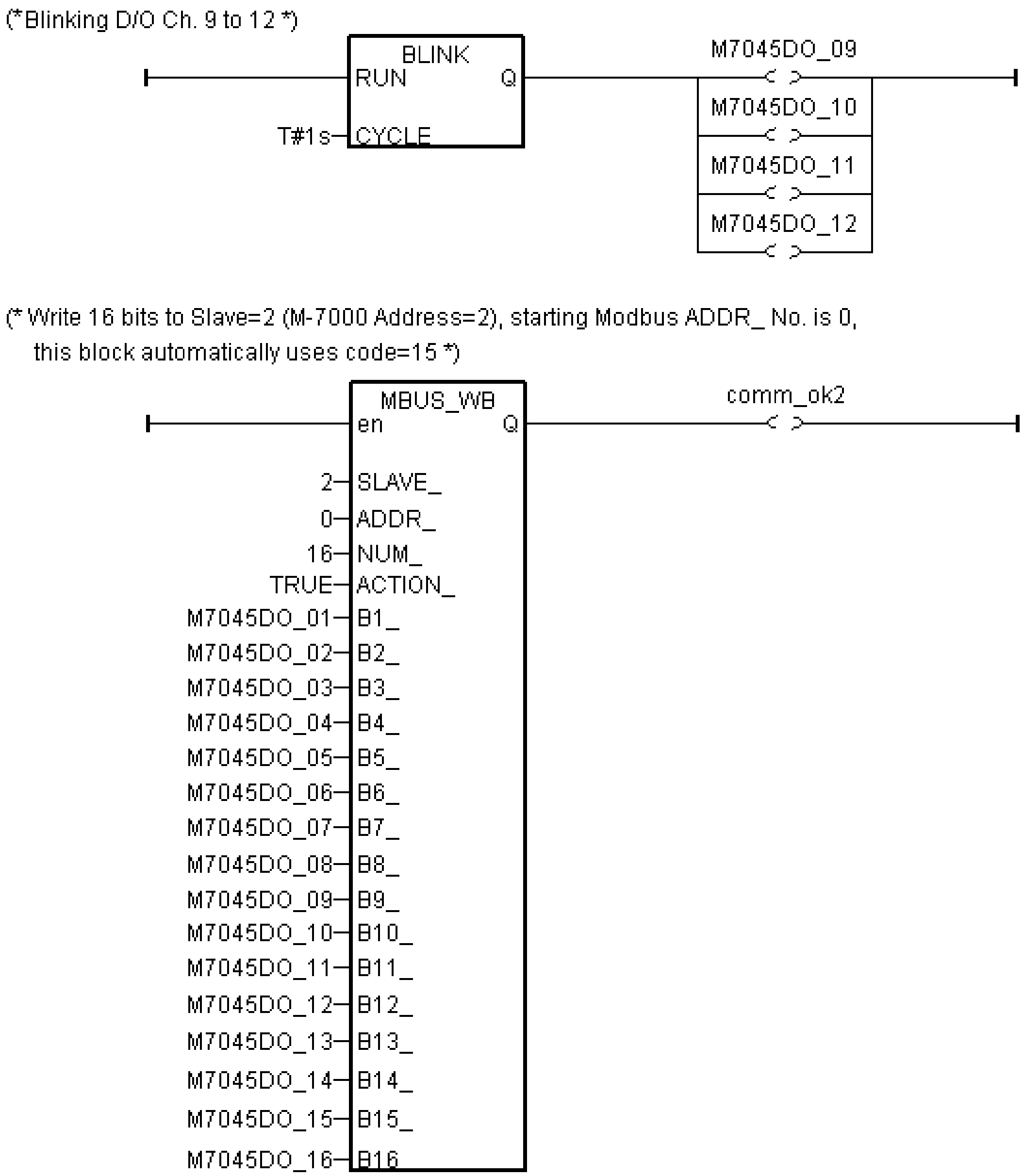


Function block program:

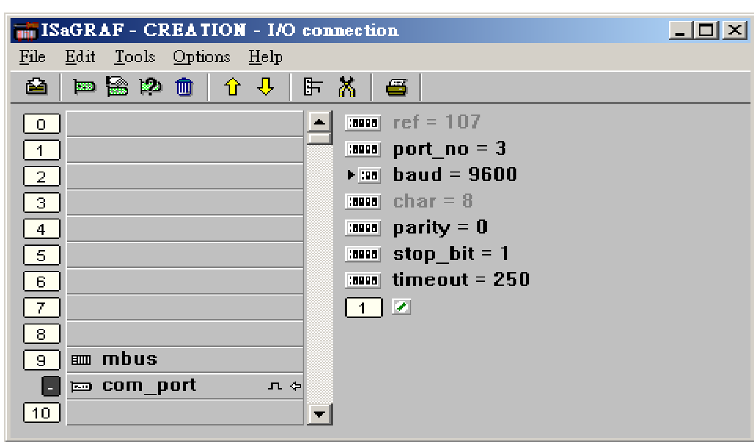


If the “SLAVE” be set as “3002”, means using COM3 to connect the Mosbus Slave device that its Net-ID is 2. Set it as “2001” means using COM2 to connect the Mosbus Slave device that its Net-ID is 1.

Ladder Program:



I/O connection:

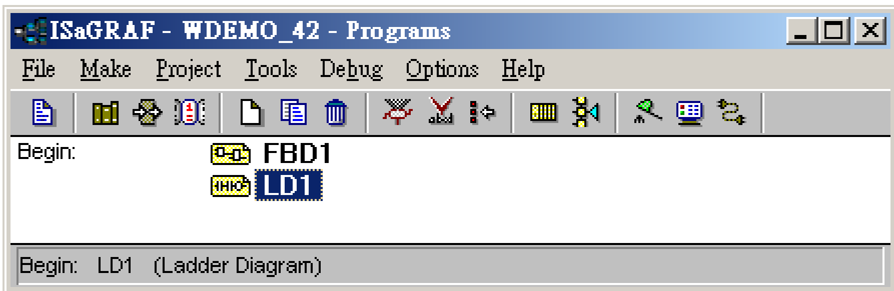


Example 42: Connecting 1: M-7053D to get D/I counter value (This example is "Wdemo_42")
Remember to do the initial setting for the M-7053D modules in the section 21.1 (steps 1 ~ 5).

Variables:

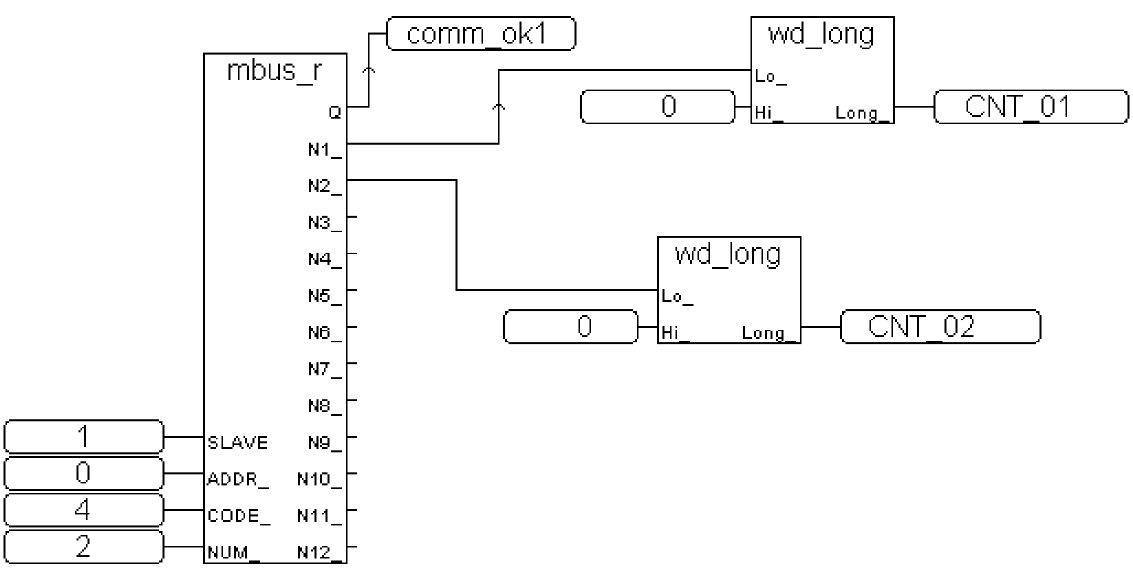
Name	***Type***	***Attribute***	***Description***
comm_ok1	Bool	Internal	Communication state of the related M-7053D
RS1	Bool	Internal	Set as True to reset Ch1. D/I counter value to 0
RS2	Bool	Internal	Set as True to reset Ch2. D/I counter value to 0
CNT_01	Integer	Internal	Ch1 D/I counter value
CNT_02	Integer	Internal	Ch2 D/I counter value

Project: One Function block program + one Ladder program



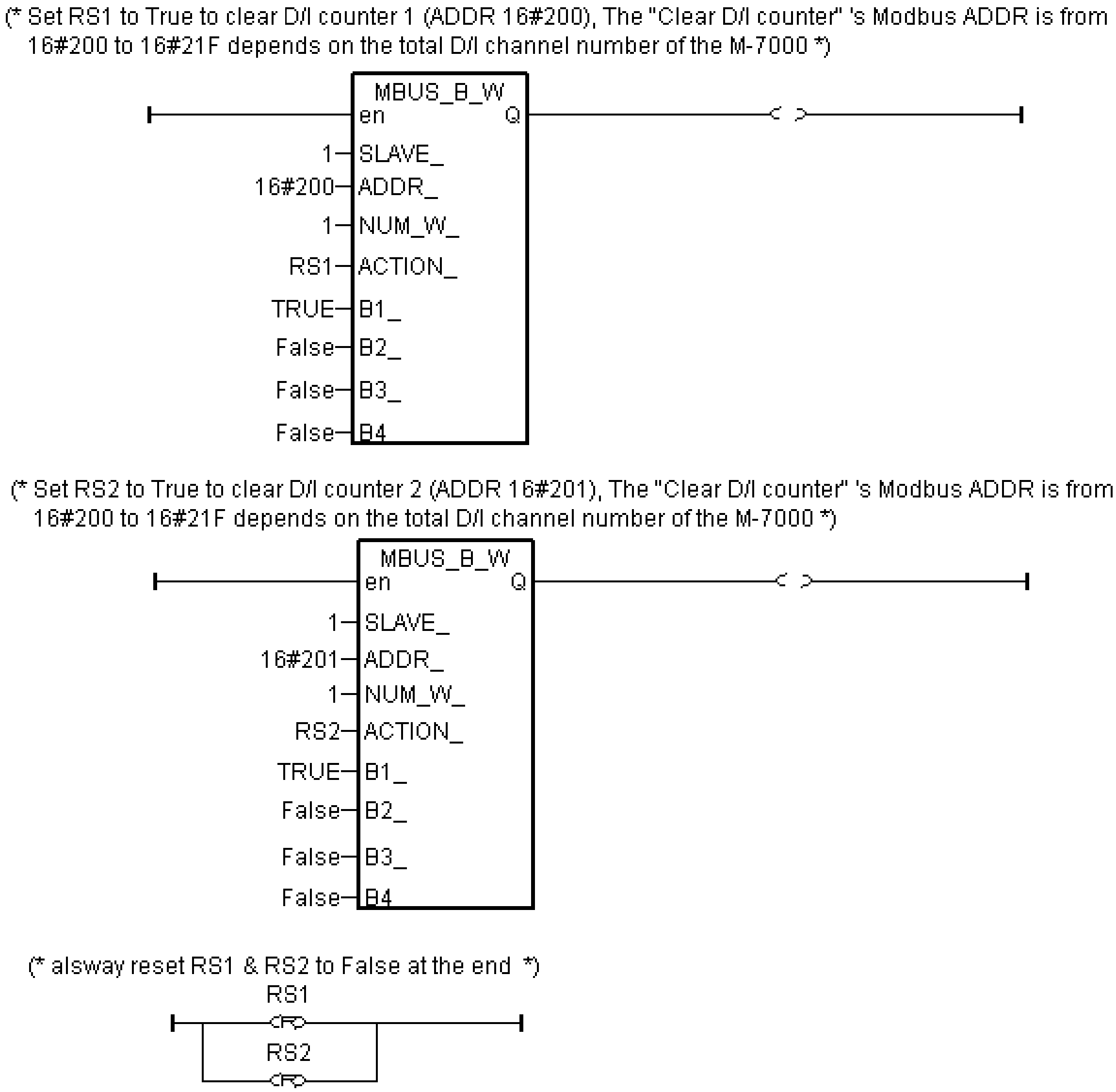
Function block program:

Using Code=4 to request M-7000 D/I counter value, Starting from Modbus ADDR No=0
NUM can be 1 to 12 depends on how many D/I counter channel in the M-7000 to be read
The M-7000 D/I counter value is from 0 ~ 65535 contained in one word.
Since Mbus_r & Mbus_r1 can only return word value as -32768 to +32767, so please
use "wd_long" to convert this word to become a long integer value. Then the converted
counter value will be 0 to 65535

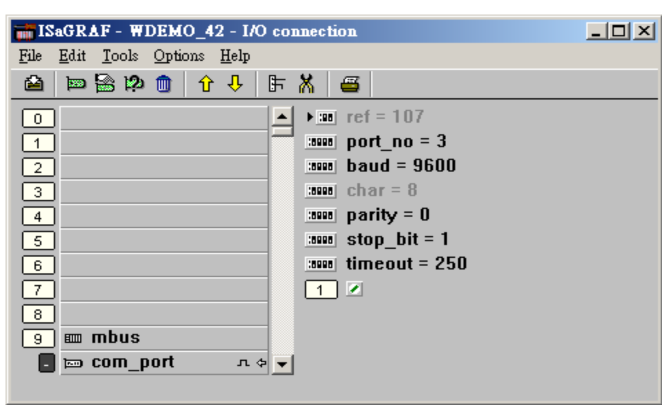


To reset M-7000's D/I counter value to 0, please write bit value 1 (TRUE) to coil Modbus No. 16#200 to 16#21F . Reset Ch1. Is to write to No. 16#200, Ch2 is 16#201, ..., Ch.32 is 16#21F.

Ladder Program:



I/O connection:



Example 43: Connecting 1: M-7017R & 2: M-7024 (This example is "Wdemo_43")

Please set M-7017R's Input range & Type to +/- 10V
M-7024's Output range & Type to +/- 10V

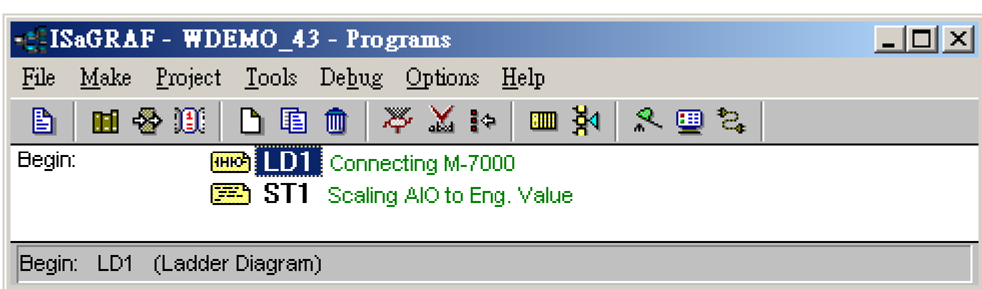
User may refer to the attatched manual in the product box, or visit http://www.icpdas.com/products/Remote_IO/m-7000/m-7000_list.htm to get each M-7000 Module's Manual to find their "Analog Input Type and Data Format Table" (Type code setting)

We use the "variable array" in this example, please refer to Section 2.6 for the details on it.

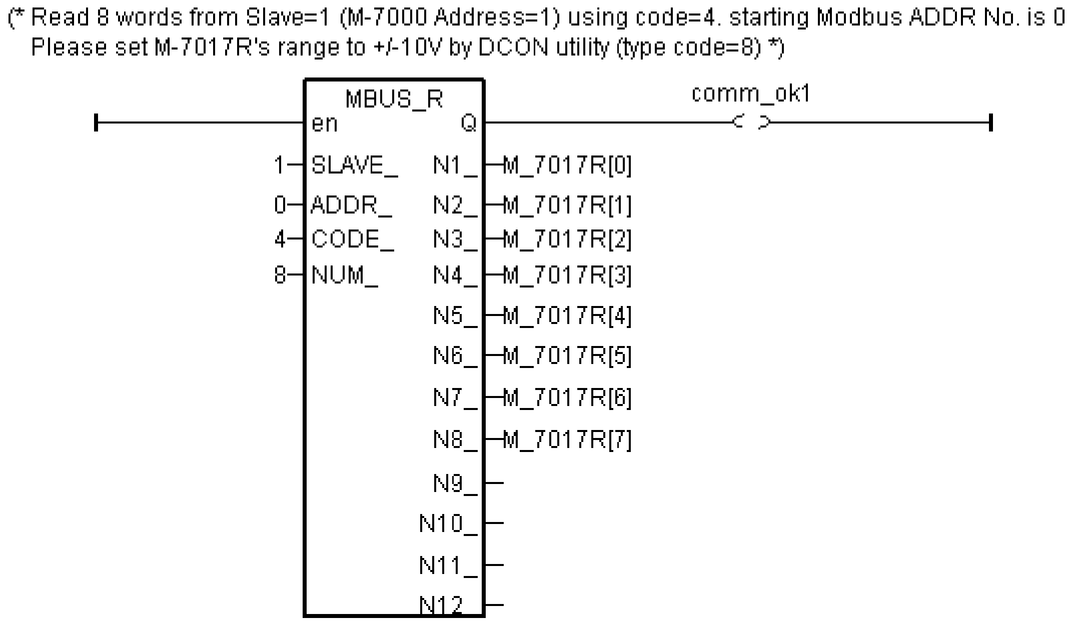
Variables:

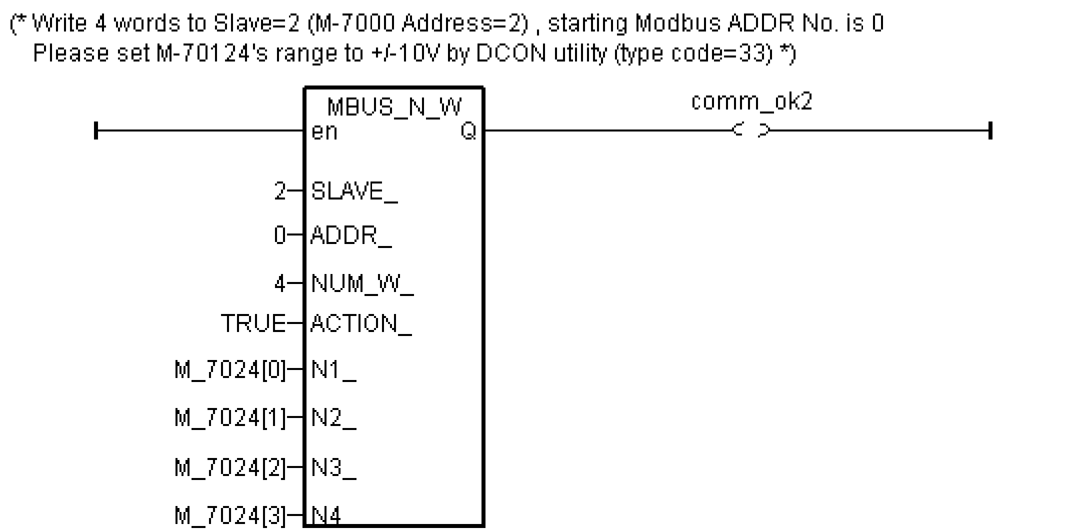
Name	*Type*	*Attribute*	*Description*
comm_ok1	Bool	Internal	Communication state of the related M-7053D
comm_ok2	Bool	Internal	Communication state of the related M-7045D
M_7017R[0..7]	Integer	Internal	**Variable Array, Dim = 8** M-7017R's Analog Input value (-32768 to +32767) means (-10 to +10) V
M_7024[0..3]	Integer	Internal	**Variable Array, Dim = 4** M-7024's Analog Output value (-16384 to +16383) means (-10 to +10) V
In_Val[0..7]	Integer	Internal	**Variable Array, Dim = 8** Engineering value converted from M_7017R[0..7] (-32768 to +32767) converter to (-10000 to +10000)
Out_Val[0..3]	Integer	Internal	**Variable Array, Dim = 4** Engineering value to be converted to M_7024[0..3] (-1000 to +1000) converter to (-16384 to +16383)
ii	Integer	Internal	index

Project: One Ladder program + one ST program



Ladder program:

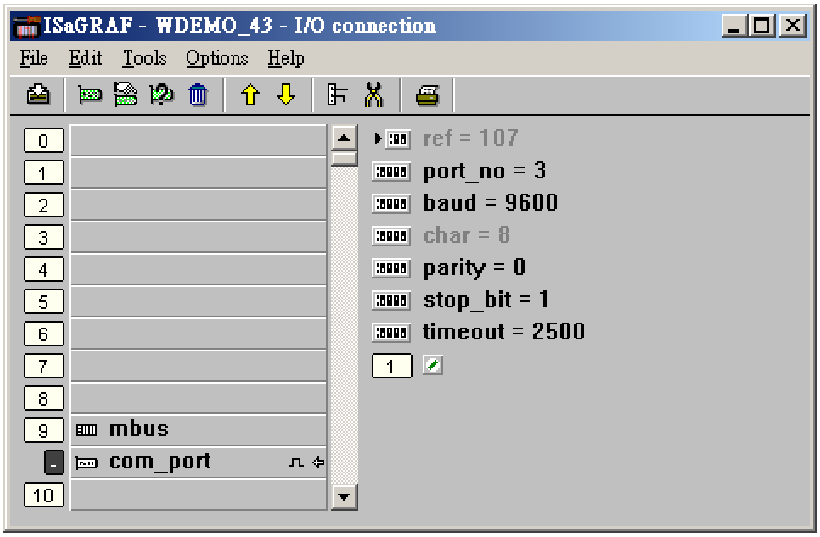




ST program:

```
(* Please configure this M-7017R as +/- 10V range (type code=8) *)
(* convert M-7017R's A/I value (-32768 to +32767) to become engineering value
   of (-10000 to +10000) *)

for ii := 0 to 7 do
 IN_Val[ii] := Bin2Eng( M_7017R[ii] , 32767 , -32768 , 10000 , -10000 ) ;
end_for ;

(* Please configure this M-7024 as +/- 10V range (type code=33) *)
(* convert  OUT_Val of (-1000 to +1000) to become M-7024's A/O value
   of (-16384 to +16383) *)

for ii := 0 to 3 do

 if OUT_Val[ii] > 1000 then
  M_7024[ii] := 16383 ;

 elsif OUT_Val[ii] < -1000 then
  M_7024[ii] := -16384 ;

 elsif OUT_Val[ii] >= 0 then
  M_7024[ii] := (OUT_Val[ii] * 16383) / 1000 ;

 elsif OUT_Val[ii] < 0 then
  M_7024[ii] := (OUT_Val[ii] * -16384) / 1000 ;

 end_if ;

end_for ;
```

I/O connection:



Example 44: Connecting 1: M-7017RC (This example is "Wdemo_44")

Please set M-7017RC 's Input range & Type to +/- 20 mA

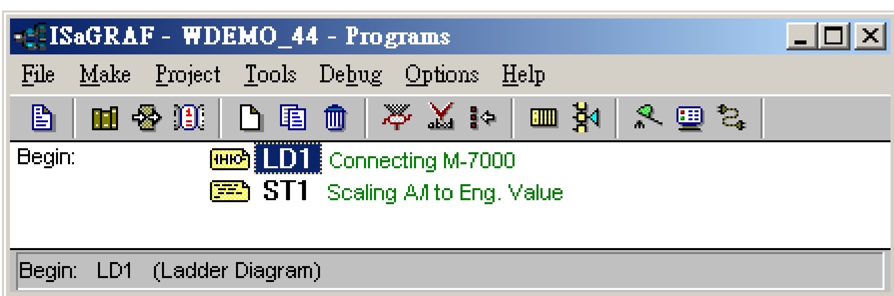
User may refer to the attatched manual in the product box, or visit http://www.icpdas.com/products/Remote_IO/m-7000/m-7000_list.htm to get each M-7000 Module's Manual to find their Analog I/O Value mapping to physical I/O (Type code setting)

We use the "variable array" in this example, please refer to Section 2.6 for the details on it.

Variables:

Name	***Type***	***Attribute***	***Description***
comm_ok1	Bool	Internal	Communication state of the related M-7053D
M7017RC[0..7]	Integer	Internal	**Variable Array, Dim = 8** M-7017RC 's Analog Input value (-32768 to +32767) means (-20 to +20) mA if setting Input range & Type to +/- 20 mA
In_Val[0..7]	**REAL**	Internal	**Variable Array, Dim = 8** (REAL format) Engineering value converted from M7017RC[0..7] 4 to 20 mA converting to (0.0 to 1000.0) psi
VAL[0..7]	Integer	Internal	**Variable Array, Dim = 8** (Integer format) Engineering value converted from M7017RC[0..7] 4 to 20 mA converting to (0 to 10000), unit is 0.1 psi
ii	Integer	Internal	Index (using in "For" loop)

Project: One Ladder program + one ST program



Analog input Table of M-7017RC:

+/- 20 mA type (type code=16#D) : -32768 to +32767
4 to 20 mA type (type code=16#7) : 0 to +32767

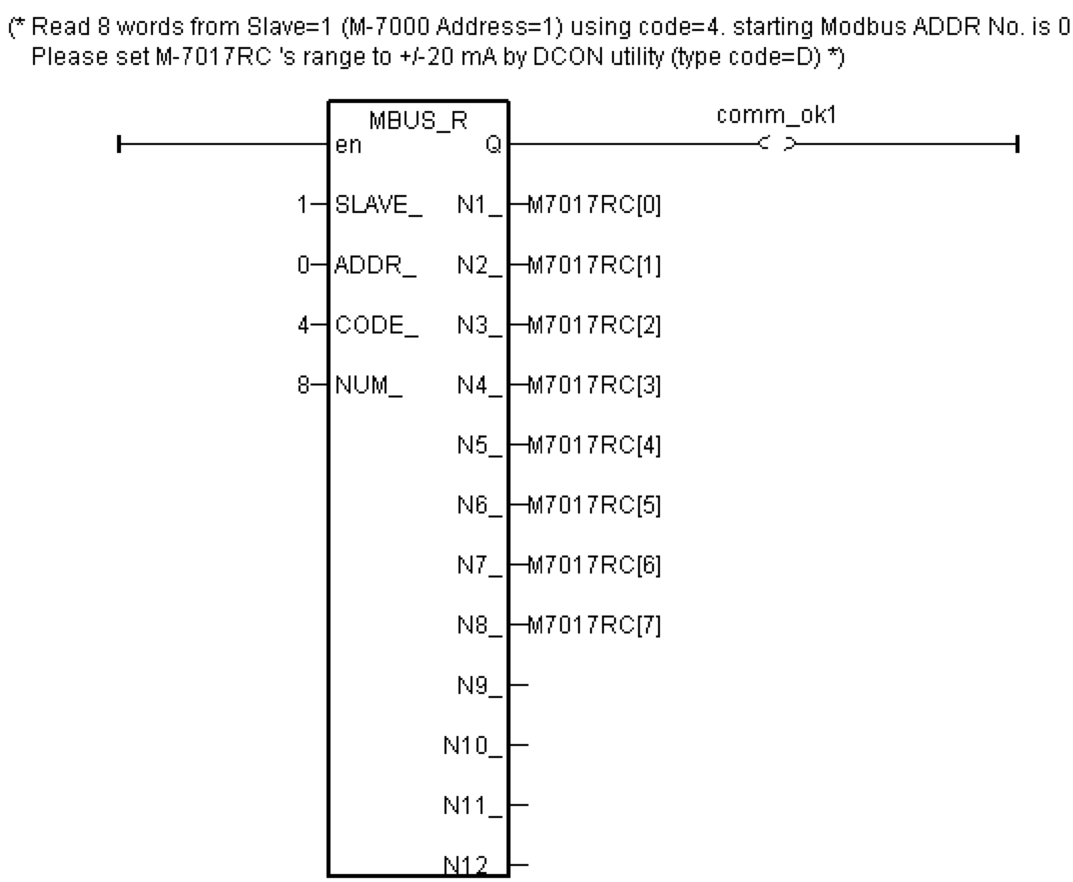
If the input sensor type is 4 to 20 mA , it is better to set M-7017RC as +/- 20 mA type .
(It is not good to set M-7017RC as "4 to 20 mA" type.)

The reason is, when the sensor is broken, the analog input of M7017RC[0..7] will be near to 0.
If setting M-7017RC 's range type as 4 to 20 mA type, the value near 0 can mean 4 mA, and also can mean sensor broken. So no way to distinguish them.

However if setting M-7017RC as +/- 20 mA type, the value near 0 only means sensor broken if the communication of M-7017RC is well.
Because if sensor is well, the input is 4 to 20 mA, value should be (6553 to 32767).
Value near 0 means sensor broken if the communication of M-7017RC is well.

For safe reason, please set M-7017RC as +/- 20 mA type.
So you can say if the value of M7017RC[0..7] < 5000 " or < 4000" , then it means sensor broken.

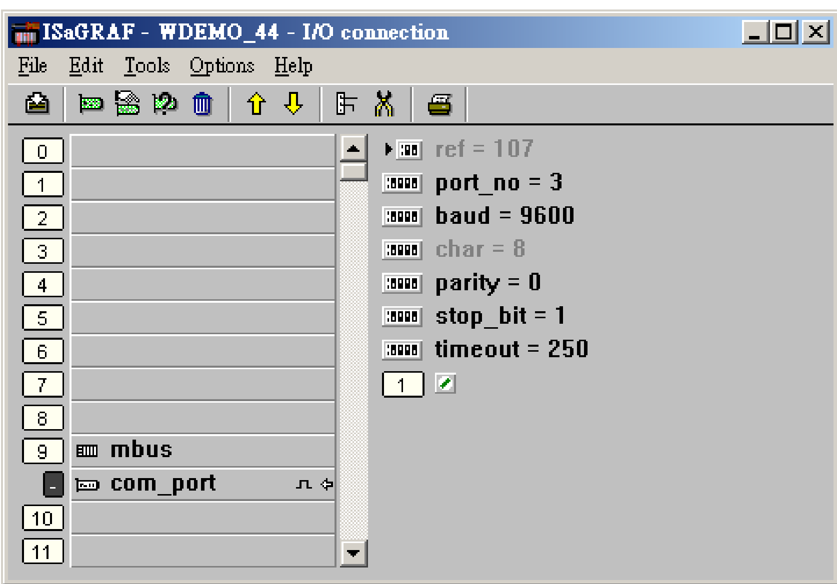
One Ladder program:



One ST program:

```
(* Please configure this M-7017RC as +/- 20 mA range (type code=D) *)
(*  We will convert  (4 , 20 mA) to become  (0.0 , 1000.0 Psi), Real format  *)
for  ii  :=  0  to  7  do
  IN_Val[ii]  :=  A4_20_To( M7017RC[ii] , 16#D , 1000.0 , 0.0 );
end_for  ;

(* or you may use Bin2Eng( ) to convert (4 to 20mA) to become (0 to 10000) as below, unit is 0.1 psi
*)
(* Please declare Val[0..7] as Integer format *)
for  ii  :=  0  to  7  do
  Val[ii]  :=  Bin2Eng(  M7017RC[ii]  ,  32767  ,  6553  ,  10000  ,  0  ) ;
end_for  ;



(* You can do something if the sensor is broken or communication is break *)
if  comm_ok1 and  ( M7017RC[ii] < 5000 )  then

  (* You may do someting if 4-20 mA sensor is broken *)

elsif  comm_ok1=False  then

  (* You may do someting if communication between controller & M-7017RC is break *)

end_if ;
```

I/O connection:



Example 45: Connecting 1: M-7019R to get temperature val (This example is "Wdemo_45")

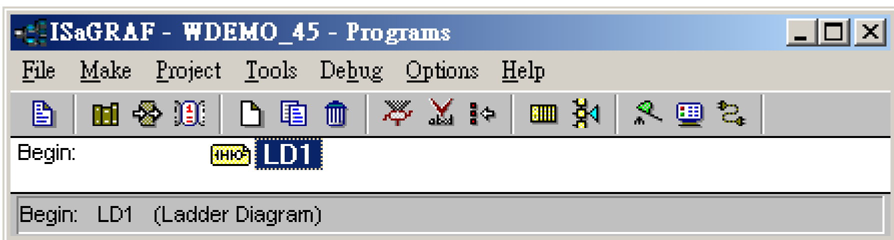
Please use DCON utility to configure M-7019R's range & type to Thermocouple, K-Type (Type code=0F)

User may refer to the attatched manual in the product box, or visit http://www.icpdas.com/products/Remote_IO/m-7000/m-7000_list.htm to get each M-7000 Module's Manual to find their "Analog Input type and data Format Table" (Type code setting)

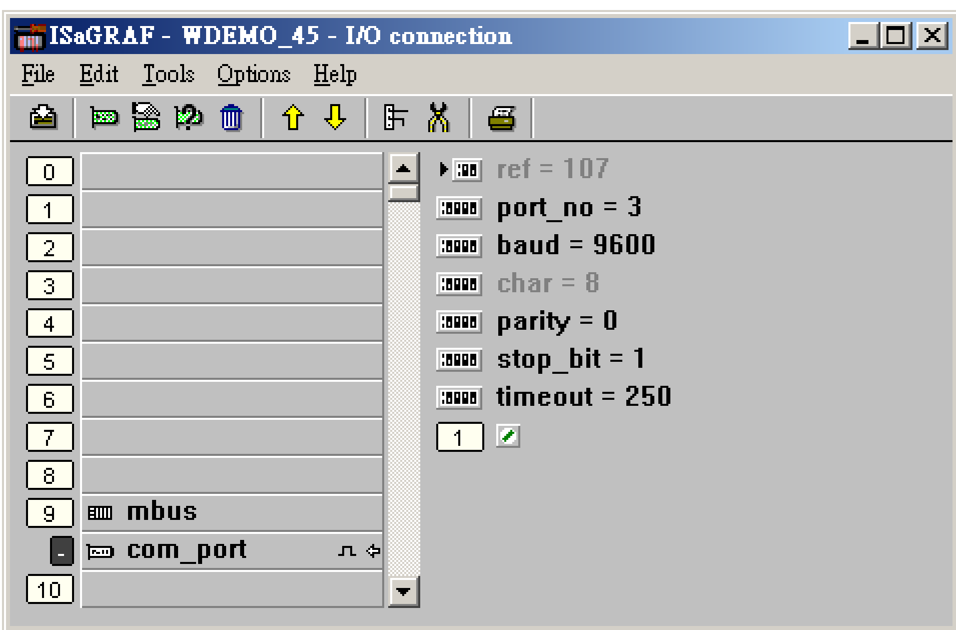
Variables:

Name	***Type***	***Attribute***	***Description***
comm_ok1	Bool	Internal	Communication state of the related M-7019R
Temper_1 to Temper_8	Integer	Internal	Temperature input value of Ch1. To 8 of M-7019R

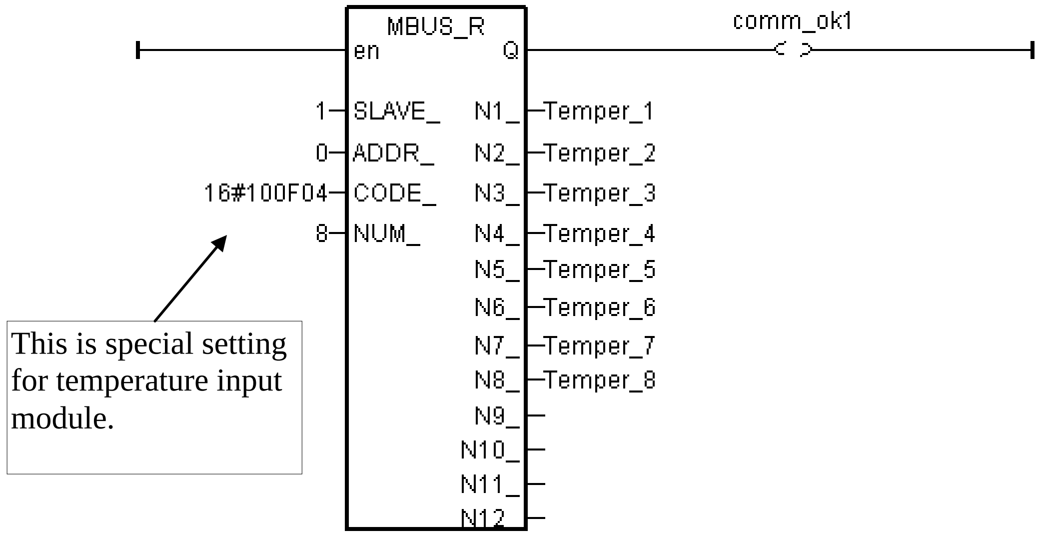
Project: One Ladder program



I/O connection:



Ladder program:



The "CODE_" parameter of "MBUS_R" & "MBUS_R1" can be "standard" or "special" setting.

In the **"standard" setting case**,
Setting "CODE_" as 1 or 2 , each returned N1_ to N12_ contains 16-bits data (or 16 Digital Input)
Setting "CODE_" as 3 or 4 , each returned N1_ to N12_ is normally from –32768 to +32767.

The **"special" setting case** is for M-7000 temperature input modules like M-7015, M-7018R & M-7019R , Please set "CODE_" to a special value defined as below.

Format : TTRRCC (Hex.)
TT=10 (Convert to "Degree Celsius")
TT=20 (Convert to "Degree Fahrenheit")
TT=00 (standard setting, -32768 to +32767. RR should be set as 00 if TT=00)
RR : "Type Code" setting of the related temperature input module
CC : Modbus function code 1 to 4 of the related Modbus device

The temperature input value unit is 0.01 degree. For ex, if returned "3012" , it means 30.12 degree. If returned 999990, it means "sensor broken line"

For example, setting "CODE_" as below to read the temperature value of M-7019:

A. 16#100F04 : (TT=10, RR=0F CC=04, Hex) the input value will be "Degree Celsius", unit is 0.01 degree, range= "0F: Thermocouple K Type, -270 ~1372 degree Celsius", code=04(Dec.). That results input value of "2356" = 23.56 Degree Celsius, "-489" = -4.89 Degree Celsius, "999990" = sensor broken-line.

B. 16#200F04 : (TT=20, RR=0F, CC=04, Hex)) the input value will be "Degree Fahrenheit ", unit is 0.01 degree, range= "0F :Thermocouple K Type, -270 ~1372 degree Celsius", code=04(Dec.). That results input value of "4512" = 45.12 Degree Fahrenheit,"500" = 5.00 Degree Fahrenheit, "999990" = sensor broken line.

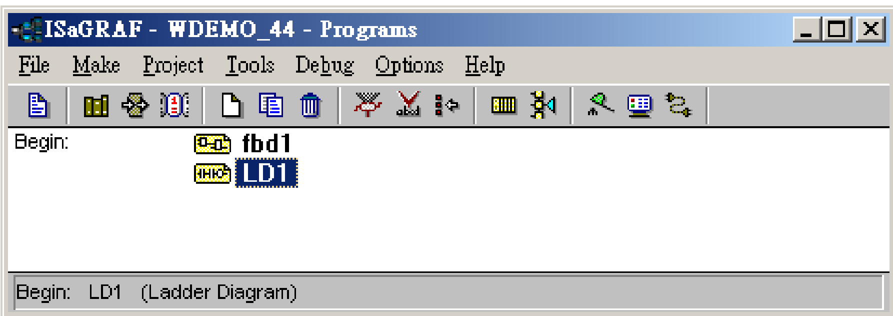
C. 16#04 : (TT=00, RR=00, CC=04) standard setting.

Example 46: Connecting 1: M-7080-D to get counter value (This example is “Wdemo_46”)

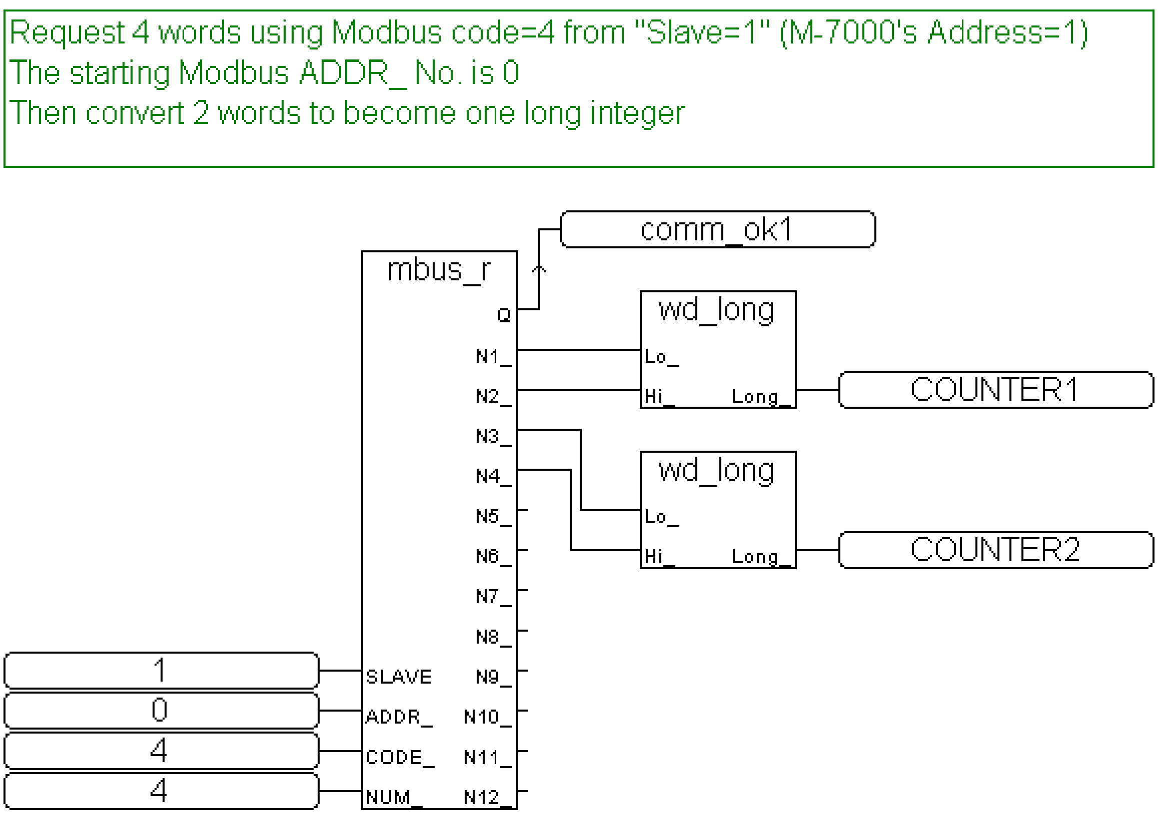
Variables:

Name	***Type***	***Attribute***	***Description***
comm_ok1	Bool	Internal	Communication state of M-7080D
RS1	Bool	Internal	set as True to reset counter 1 as 0
RS2	Bool	Internal	set as True to reset counter 2 as 0
COUNTER1	Integer	Internal	1st Counter or frequency value of M-7080D
COUNTER2	Integer	Internal	1st Counter or frequency value of M-7080D

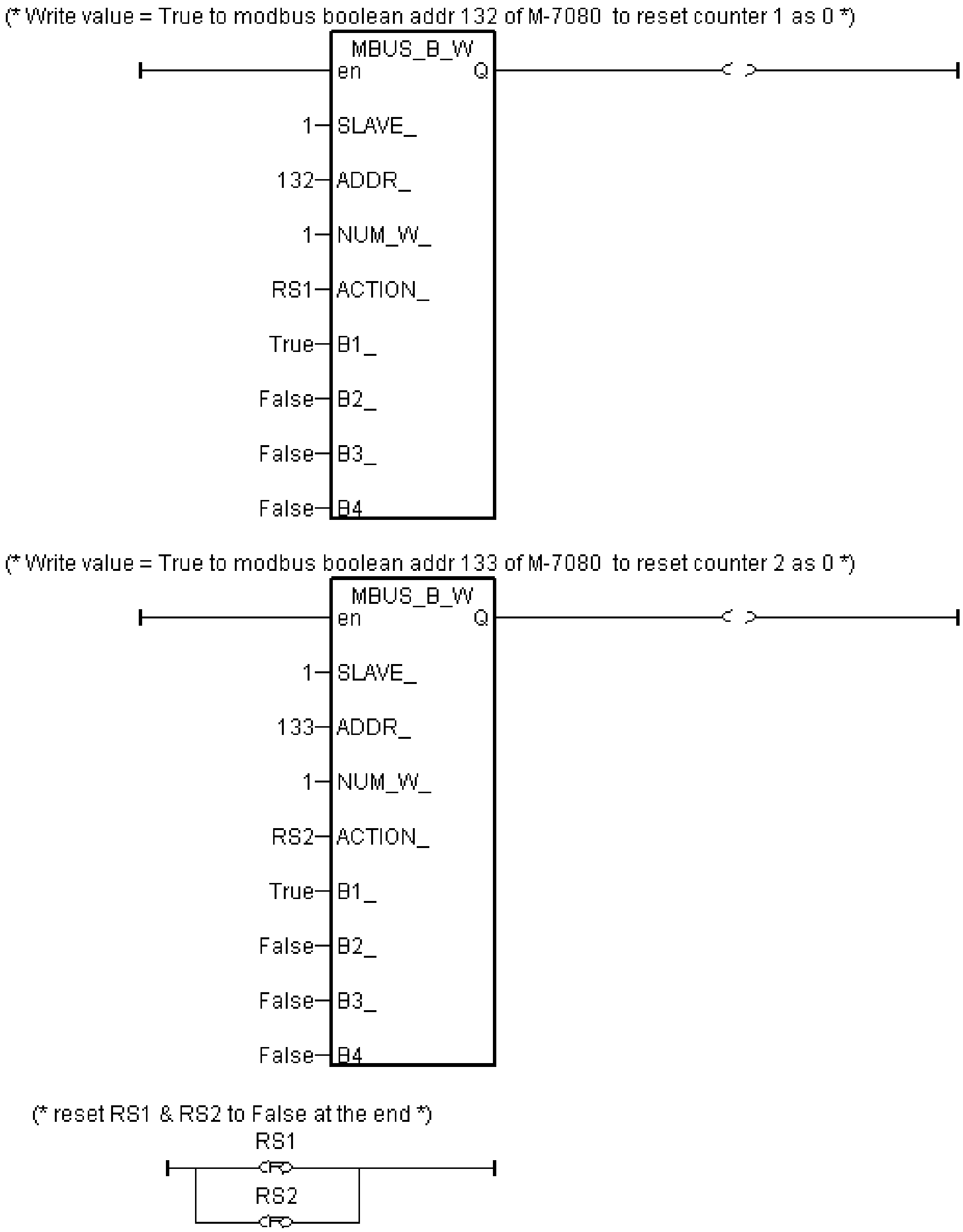
Project: One function block program + one Ladder program



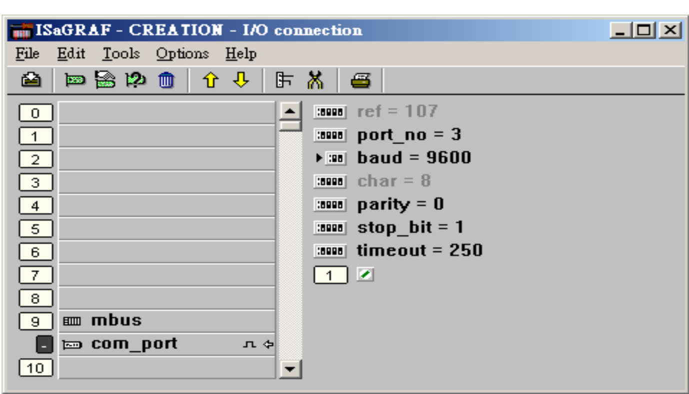
Function block program:



Ladder program:



I/O connection:



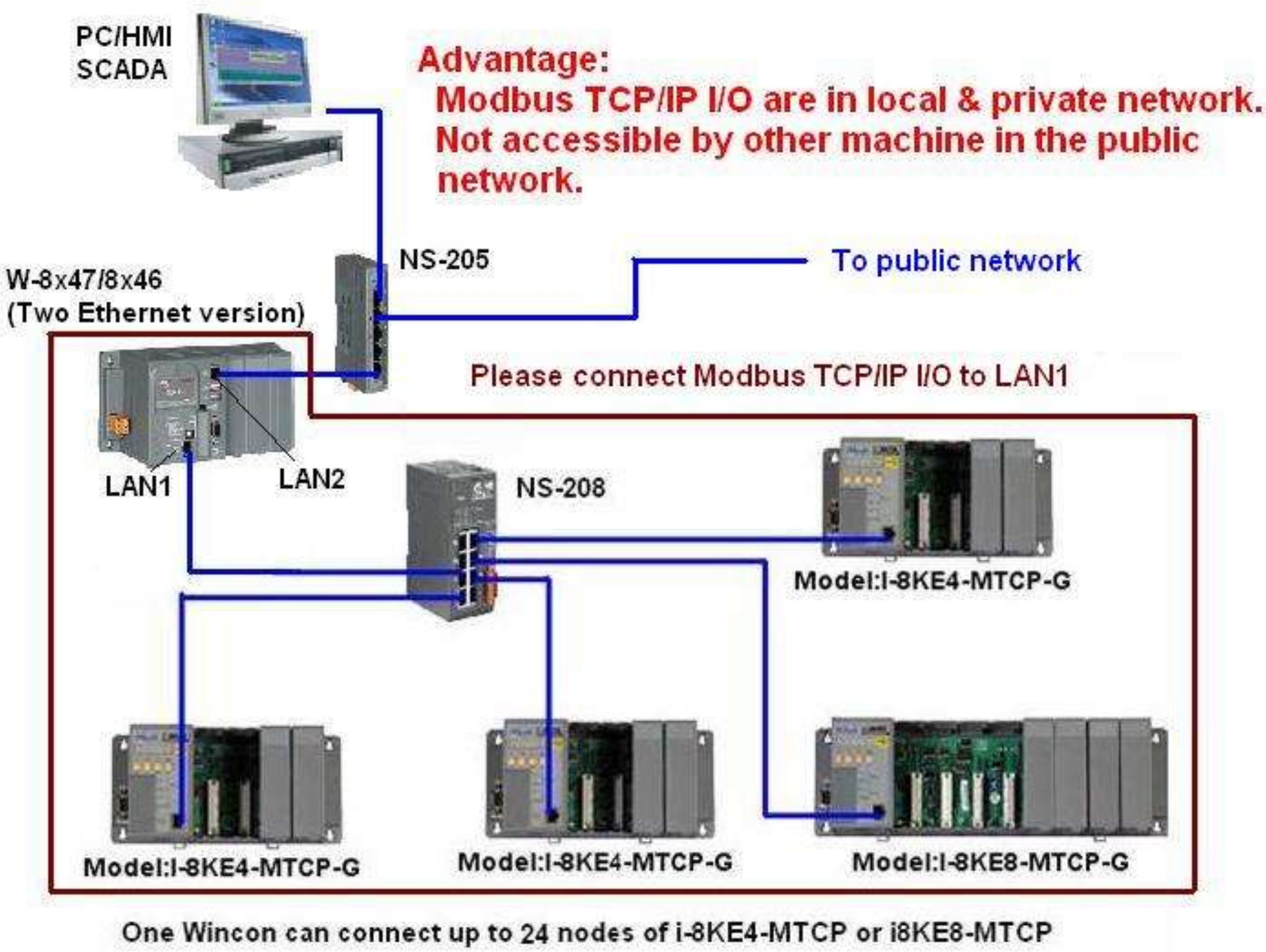
Chapter 22. Connecting Modbus TCP/IP I/O

W-8x47/8x46 supports I-8KE4-MTCP & I-8KE8-MTCP ethernet I/O since its ISaGRAF driver version 3.32B. (The WP-8xx7, WP-5xx7,VP-25W7/23W7, XP-8xx7-CE6, XP-8xx7-Atom-CE6 also support Ethnet I/O)
I-8KE4/8-MTCP: http://www.icpdas.com/products/PAC/i-8000/i-8KE4_8KE8_MTCP.htm
WinCon ISaGRAF driver: http://www.icpdas.com/products/PAC/i-8000/isagraf-link.htm
NS-205 / NS-208 : http://www.icpdas.com/products/Switch/industrial/ethernet_switch.htm

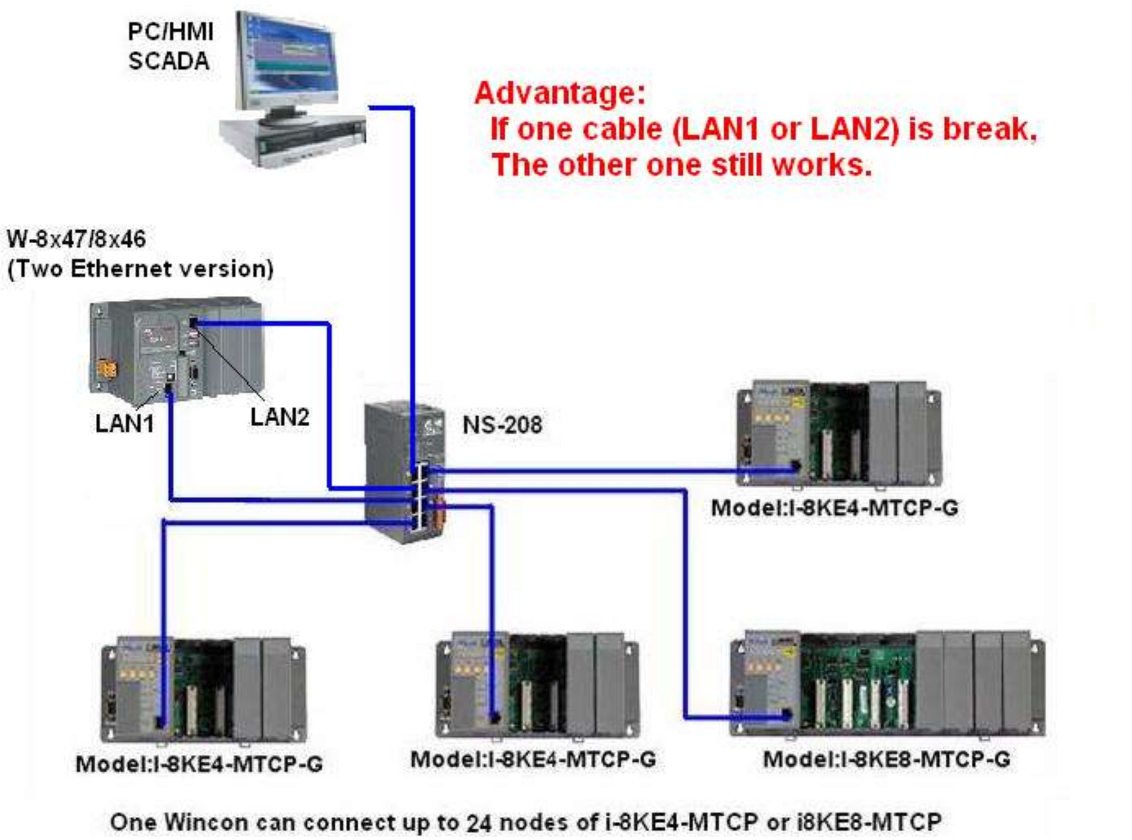
22.1: Induction of the I-8KE8-MTCP I/O

One W-8x47/8x46 can connect max. 24 nodes of I-8KE4-MTCP and I-8KE8-MTCP. The Ethernet I/O scan time of the 3000 ~ 6000 I/O Channels is about 30 to 40 ms. If connecting less than 10 nodes of I-8KE4/8-MTCP, the Ethernet I/O scan time is about 20 ms. However, it still depends on how big (complex) of your logic program. (**The Ethernet I/O scan time of one W-8x36 / 8x37 is about twice of the W-8347 / 8747. That means W-8x36 / 8x37 is slower than W-8347 / 8747 when connecting Ethernet I/O**)

Configure1: W-8347 / 8747 (Dual Ethernet version) connecting Modbus TCP/IP I/O in a safe-local-private network .

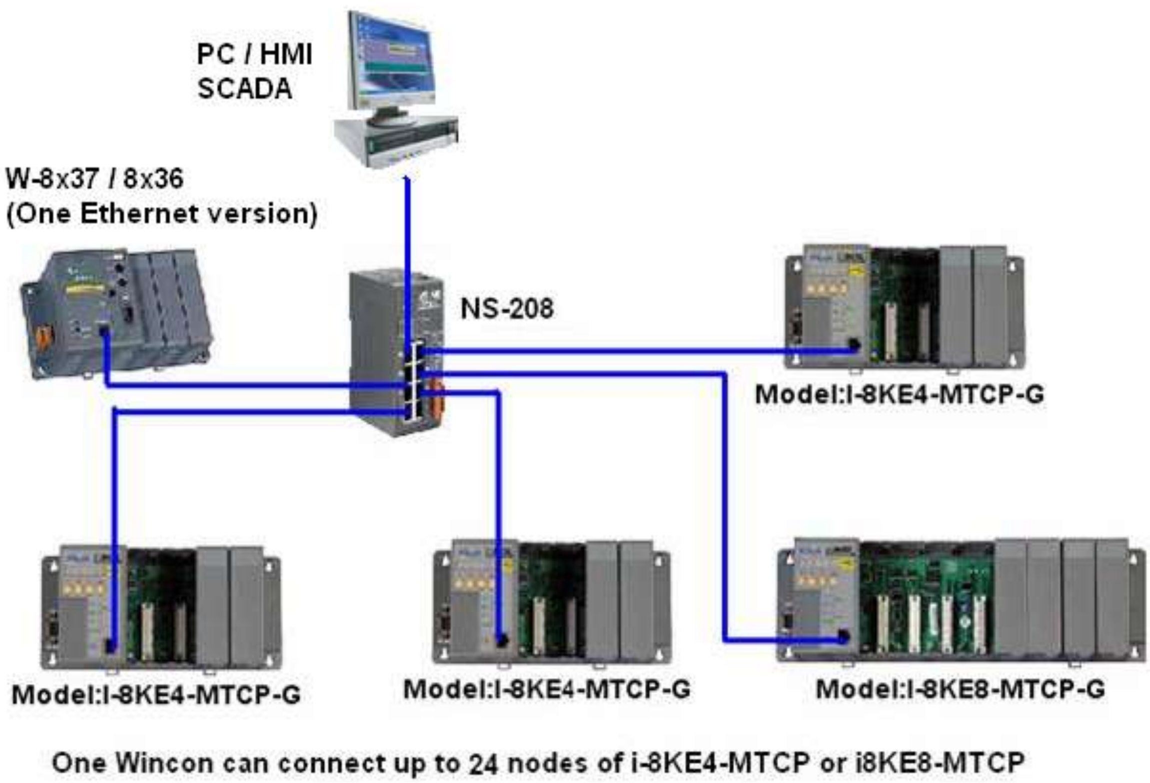


Configure2: W-8x47/8x46 (Dual Ethernet) connecting Modbus TCP/IP I/O in both two ports.



Configure3: W-8x37/8x36 (One Ethernet version) connecting Modbus TCP/IP I/O.

This configure doesn't have the advantage of configure 2 . And if the NS-208 is connected to public network, then it doesn't have the advantage of configure 1.



22.2: Programming to Control the I-8KE8-MTCP I/O

Step 1.

The first step is to assign an unique IP address to all of the I-8KE4-MTCP and I-8KE8-MTCP. Please power off the I-8KE4/8-MTCP, short its “INIT” to “INIT * COM”, and then power it up.

Then connect a RS-232 cable from your PC ‘ s COM1 to the I-8KE4/8-MTCP ‘s COM1.

Then please run “7188xw.exe” on PC (“7188xw.exe” is burned in I-8000 CD-ROM or can be download at ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/minios7/utility/).

If your computer has no COM1/COM2 or you use other COM (like COM5) to link the I-8KE4/8-MTCP, you can change the “C number” in the first line of “7188xw.ini” file.

EX: Using computer’s COM5 to link to I-8KE4/8-MTCP

```
C1 B115200 P0 D8 S1
F
Xautoexec.bat Xisa7188e.exe
w25
```

```
C5 B115200 P0 D8 S1
F
Xautoexec.bat Xisa7188e.exe
w25
```

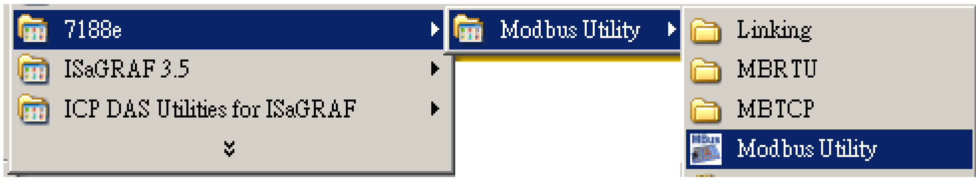
Press some <Enter> in the “7188xw.exe” windows , and then type “ip” to view the current IP setting. Then type in for example, “**ip 192.168.2.70**” to set an IP address to it.

To set mask address, please type in for example, “**mask 255.255.255.0**”

PLEASE make sure to remove the connection between the “INIT” and the “INIT * COM” pin on the I-8KE4/8-MTCP ‘s front panel after setting successfully. And then re-cycle its power. (Recommend to use the NS-205/ NS-208)

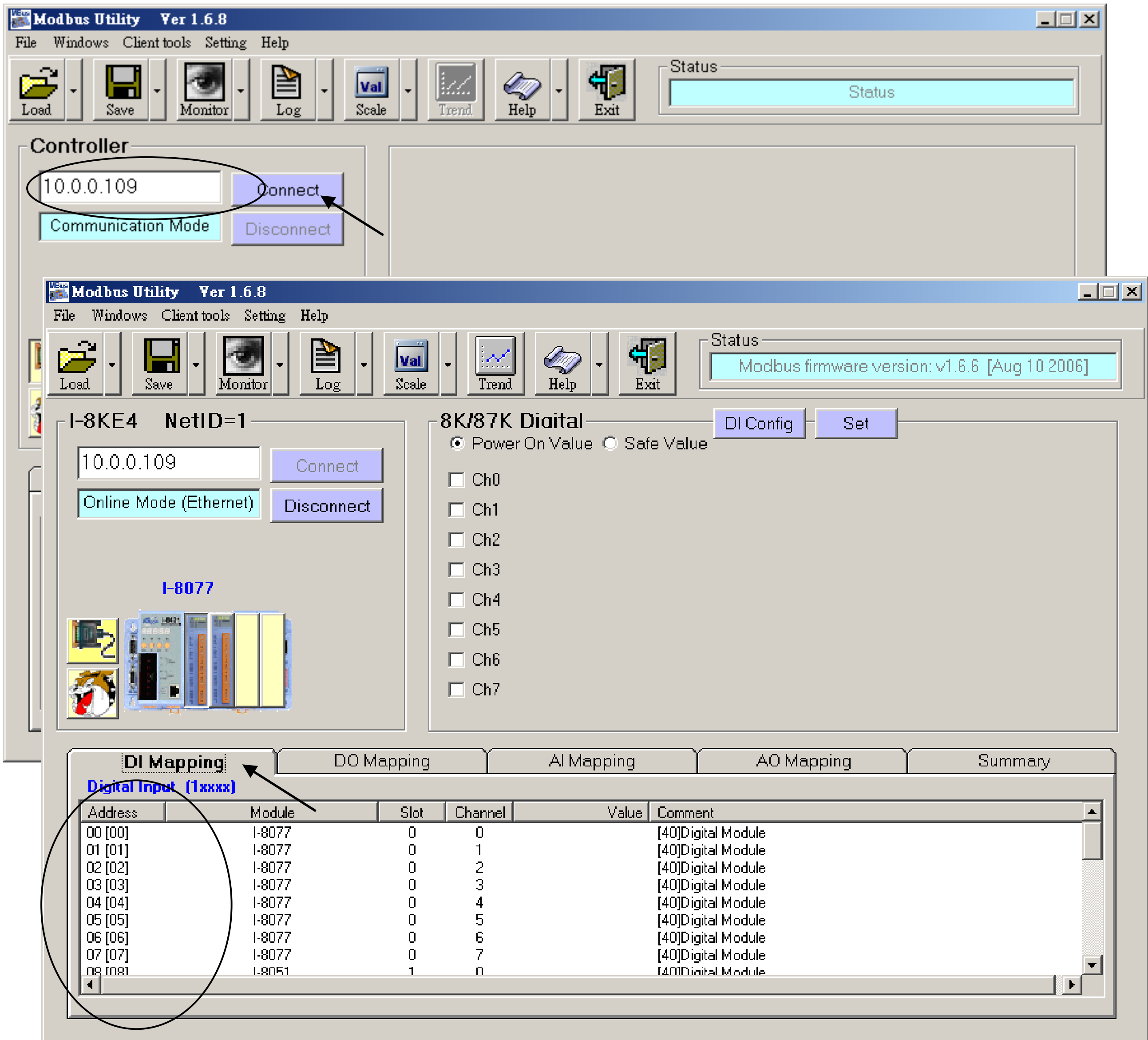
Step 2.

The second step is to configure all of the connecting I-8KE4/8-MTCP by running "Modbus utility". The Modbus utility is burned in the I-8000 CD-ROM or can be download at
http://www.icpdas.com/products/PAC/i-8000/modbus_web_download.htm



Important Noticed:
Every I-8KE4/8-MTCP with new plugged IO board should be configured at least once by "Modbus utility". **If the 2nd & 3rd Leds below the Five 7-Segment-Led is always blinking, it means this I-8KE4/8-MTCP is not configured well** by the "Modbus utility"

Enter the correct IP of the I-8KE4/8-MTCP on the Modbus utility, and then click on “Connect”. If the I-8KE4/8-MTCP is well connected, You will see the Modbus address assigned in the I-8KE4/8-MTCP. For example, D/I starting from 0 to ... , A/I starting from 0 to ...



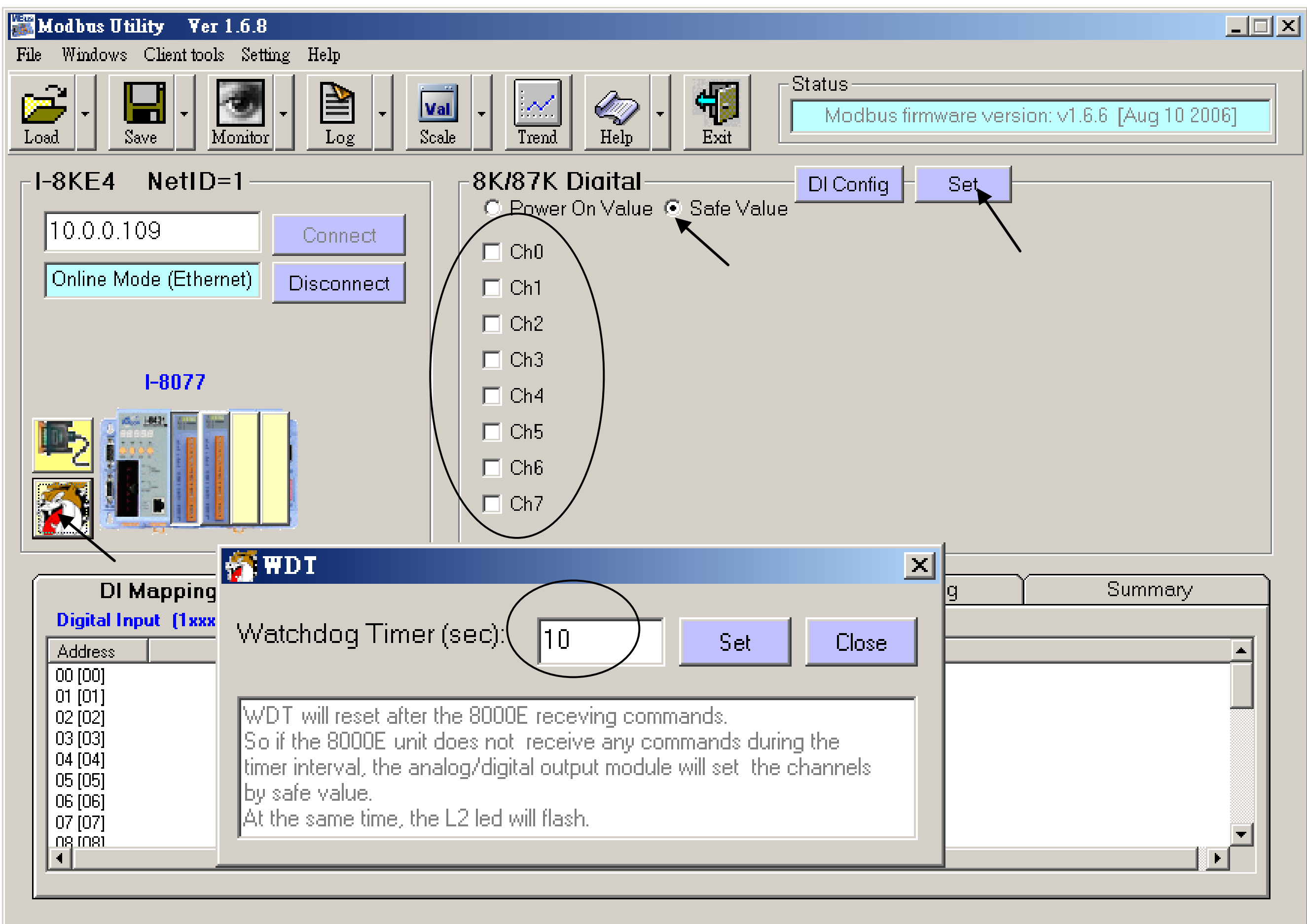
Address	Module	Slot	Channel	Value	Comment
00 [00]	I-8077	0	0		[40]Digital Module
01 [01]	I-8077	0	1		[40]Digital Module
02 [02]	I-8077	0	2		[40]Digital Module
03 [03]	I-8077	0	3		[40]Digital Module
04 [04]	I-8077	0	4		[40]Digital Module
05 [05]	I-8077	0	5		[40]Digital Module
06 [06]	I-8077	0	6		[40]Digital Module
07 [07]	I-8077	0	7		[40]Digital Module
08 [08]	I-8051	1	0		[40]Digital Module

There is a Watchdog setting for the I-8KE4/8-MTCP. The default value is “disable the watchdog” . You may enable it by assigning a “Watchdog timer” value , for example from 10 to 120 seconds. This will automatically set the Digital outputs and Analog outputs of the I-8KE4/8-MTCP to a pre-defined “Safe Value” when the Ethernet communication between the WinCon and the I-8KE4/8-MTCP is break.

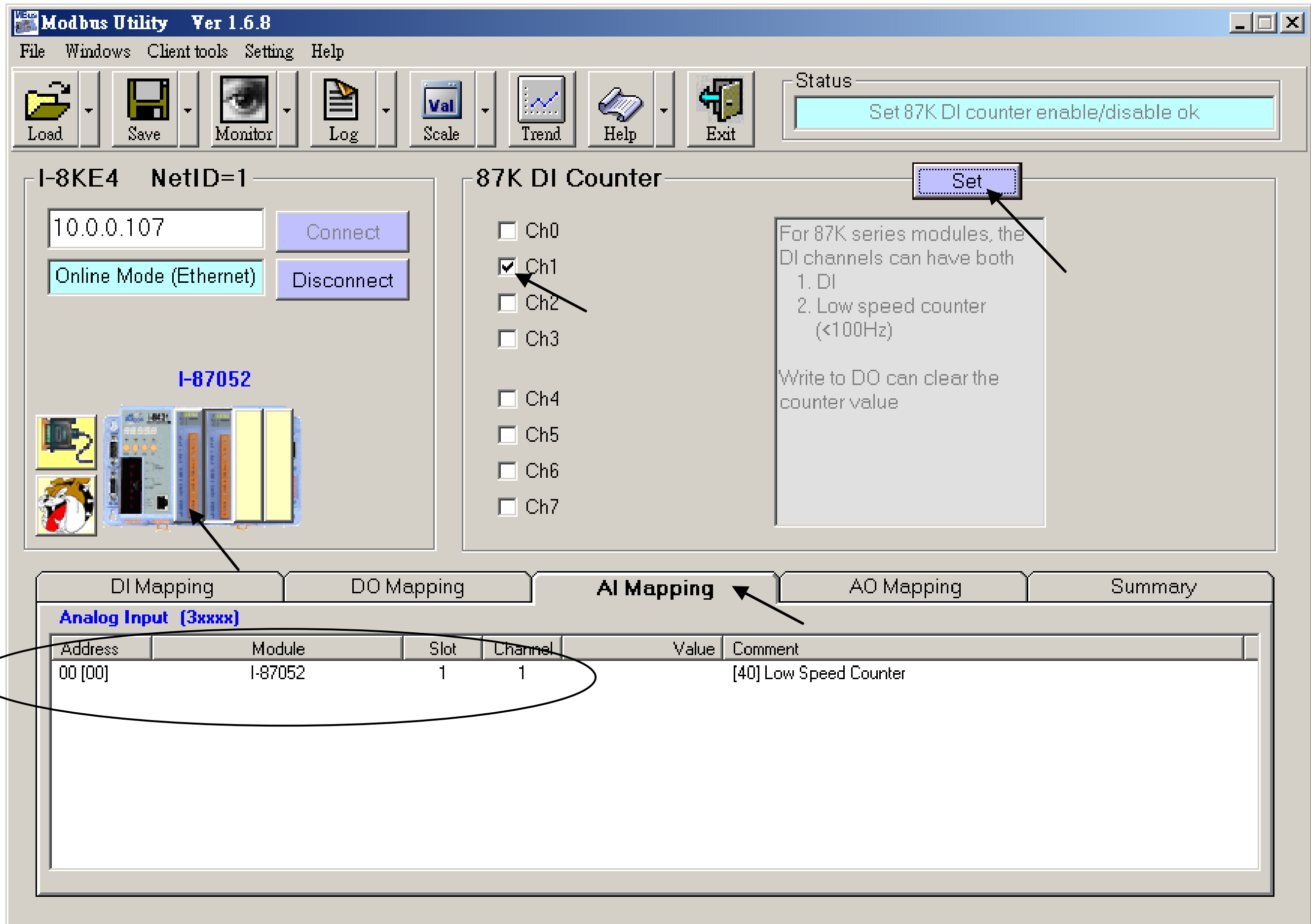
If you check the Ch0 to ... of the “Safe Value”, it means to set these Channels to have a safe value of “ON”.

If you un-check it, it means the related channel has safe value of “OFF”.

The “Safe value” function only works when you assign a value larger than zero to the “Watchdog timer” and the communication is break. Set 0 to the “Watchdog timer” is to disable the Watchdog function.



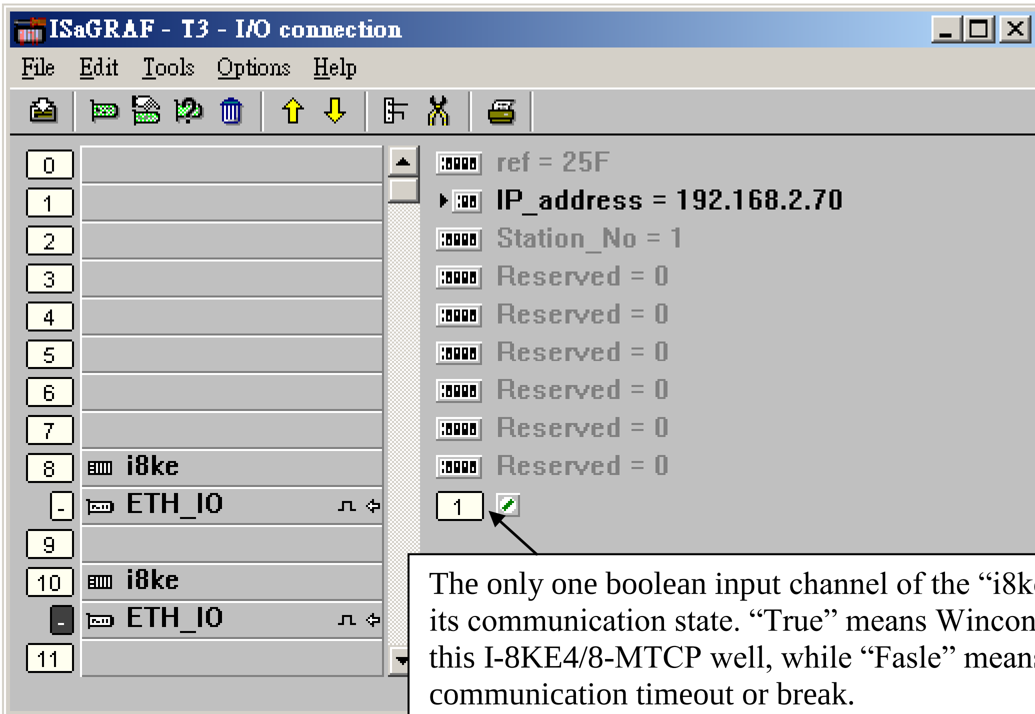
User may enable the “87K DI Counter” if you have plugged I-87K D/I module in the I-8KE4/8-MTCP. Every Digital Input channel of the I-87K module have a D/I Counter value. The max. rate can be accepted is 100 Hz . The DI Counter value is a 16-bits value (0 to 32767, and then from -32768 to -1, Hex is from 0000, 0001 ... to 7FFF , 8000 , 8001, ... to FFFF, then back to 0000, ...). The DI Counter value is a A/I value with a Modbus address.



Note:
Every I-8KE4/8-MTCP with new plugged IO board should be configured at least once by “Modbus utility”. **If the 2nd & 3rd Leds below the Five 7-Segment-Led is always blinking, it means this I-8KE4/8-MTCP is not configured well** by the “Modbus utility”

Step 3.

Please connect “i8ke” I/O complex equipment in your ISaGRAF project. Please enter the IP address of the related I-8KE4-MTCP or I-8KE8-MTCP. If the WinCon has connected more than one I-8KE4/8-MTCP, you should connect more “i8ke” as below.



The only one boolean input channel of the “i8ke” indicates its communication state. “True” means Wincon connecting this I-8KE4/8-MTCP well, while “Fasle” means communication timeout or break.

If the i8ke , i8ke_b, i8ke_n , i8ke_f , i8ke_b_a , i8ke_n_a , i8ke_f_a is not found in your ISaGRAF, please visit http://www.icpdas.com/products/PAC/i-8000/isagraf.htm to download “ICP DAS Utilities For ISaGRAF.zip” to install it again to the ISaGRAF workbench.

Step 4.

Please map ISaGRAF internal variables to the related Modbus I/O address of the I-8KE4/8-MTCP by using below functions, please refer to Appendix A.4.

I8KE_B	Set Boolean variable as an I-8KE4/8-MTCP Ethernet I/O
I8KE_B_A	Set Boolean 'Variable Array' as several I-8KE4/8-MTCP Ethernet I/O(s)
I8KE_F	Set REAL variable as an I-8KE4/8-MTCP Ethernet I/O
I8KE_F_A	Set REAL 'Variable Array' as several I-8KE4/8-MTCP Ethernet I/O(s)
I8KE_N	Set Integer 'Variable Array' as an I-8KE4/8-MTCP Ethernet I/O (s)
I8KE_N_A	Set Integer 'Variable Array' as several I-8KE4/8-MTCP Ethernet I/O(s)

Example program: “Wdemo_30” & “Wdemo_31” at W-8xx7 CD-ROM : \napdos\isagraf\wincon\demo\ or ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/demo/

Chapter 23. Connecting the Fast FRnet Remote I/O

µPAC-7186EG (since its driver version 1.06), WinCon-8xx7 (since its driver version 3.42), iPAC-8447/8847 (since its driver version 1.01), VP-25W7/23W7 (since its driver version 1.02), WP-8xx7, XP-8xx7-Atom-CE6 and XP-8xx7-CE6 (since its released driver version 1.01) support the FRnet Digital I/O.

I-8xx7 (40MHz), I-8437-80, I-8837-80 and I-7188EG/XG don't support the FRnet I/O.

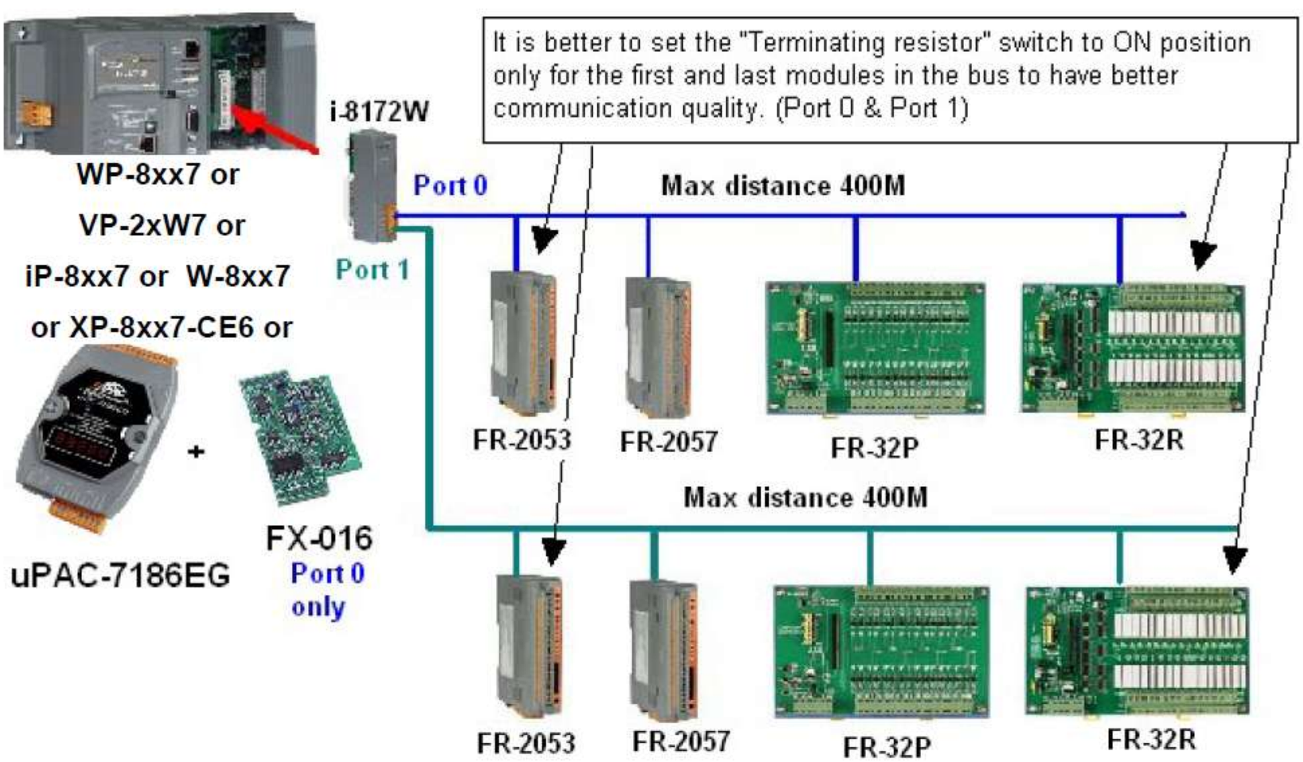
WP-8xx7, VP-2xW7, iPAC-8xx7, XP-8xx7-Atom-CE6, XP-8xx7-CE6 and W-8xx7 required an **I-8712W** to connect to the FRnet I/O.

µPAC-7186EG required a FX-016 (x-board) to connect to the FRnet I/O.

Please visit below web site for more information.

I-8172 / I-8172W and FRnet I/O: http://www.icpdas.com/products/Remote_IO/frnet/frnet_list.htm
µPAC-7186EG + FX-016 (x-board) : Please contact your local agent or service@icpdas.com
(FX-016 doesn't support RoHS yet)
ISaGRAF driver: http://www.icpdas.com/products/PAC/i-8000/isagraf-link.htm
ISaGRAF PAC : http://www.icpdas.com/products/PAC/i-8000/isagraf.htm

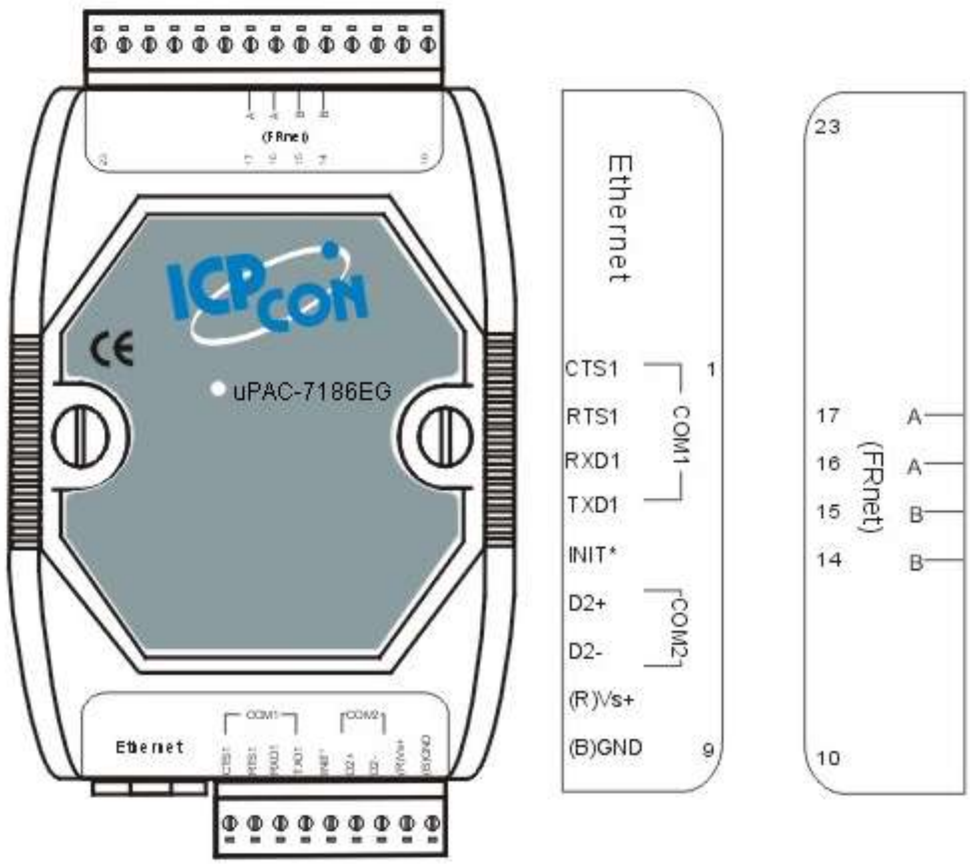
For the information to control the FRNET A/I and A/O modules, please refer to http://www.icpdas.com/faq/isagraf.htm > FAQ-154.



Pin assignment of the uPAC-7186EG + FX-016:

Please plug the FX-016 (x-board) into the uPAC-7186EG’s slot 0 (Remove the front shell of the uPAC-7186EG , you will find where the slot 0 is) .

FX-016 supports only one FRnet port. The signal is A and B. (There is no A2, B2 for FX-016)



23.1: Introduction of the FRnet I/O

Important:

2. For FRnet I/O board:
WP-8xx7, VP-25W7/23W7, iPAC-8xx7, XP-8xx7-Atom-CE6 and XP-8xx7-CE6 supports only I-8172W, no support I-8172. WinCon-8xx7 supports I-8172W and I-8172. uPAC-7186EG supports only FX-016 (x-board), no support I-8172W and I-8172.

3. Every FRnet Output module has a 'RESET' or called 'HOLD' dip on its Dip switch or a special Jumper. User may set it to 'ON' position (or enable it), this will reset the output channels to OFF state when the communication is broken between the I-8172W (or uPAC-7186EG + FX-016) and the FRnet D/O module. For example, set 8th Dip to ON of FR-2057 means enable it.

4. The communication state of D/I modules can be detected in the 8-Ch. D/I of "I-8172" (For uPAC-7186EG+FX-016, it is "Frnet86") in the IO connection window. However FRnet Output module doesn't support communication detection.

5. WinPAC-8xx7 supports max. 8 pcs. of I-8172W in its Slot 0 through 7 , while XP-8xx7-CE6 and W-8xx7 support max. 7 pcs. of I-8172W in slot 1 thru. 7, iPAC-8xx7 supports only max. 4 pcs. of I-8172W in its Slot 0 through 7, VP-2xW7 supports max. 3 pcs. of I-8172W in its Slot 0 through 2 and uPAC-7186EG supports only one FX-016 in its slot 0.

Advantage of FRnet I/O:
Fast I/O scan, it is about 3 ms / per FRnet I/O scan. (This depends on your program's PLC scan time, for ex, if the ISaGRAF PLC program scan time is about 15 ms, then the scan time for all will be 15 ms, not 3 ms).

Below is the approximate PLC scan time of an ISaGRAF project which runs only the FRnet setup code (without other codes):

	i8172w x 1	i8172w x 2	i8172w x 3	i8172w x 4	i8172w (5~8)
WP-8xx7 XP-8xx7-CE6 VP-2xW7	3 ms 3 ms (3 ms)	4 ms 4 ms (4 ms)	4 ms 4 ms (4 ms)	4 ms 4 ms (-)	8 ms 8 ms (-)
iP-8xx7	3 ms	6 ms	9 ms	12 ms	-

	FX-016 x 1
µPAC-7186EG	3 ms

WP-8xx7, VP-25W7/23W7, W-8xx7, XP-8xx7-CE6 or iPAC-8x47 plus I-8172W boards (or uPAC-7186EG plus FX-016) can connect to FRnet I/O modules, for example, FR-2053, FR-2057, FR-2054, FR-32P, FR-32R listed in http://www.icpdas.com/products/Remote_IO/frnet/frnet_list.htm.

One I-8172W board has two FRnet ports, ID is port 0 & port 1 (uPAC-7186EG + FX-016 supports only port 0). Each FRnet port can connect up to 8 FRnet D/O “Module Address” and up to 8 D/I “Module Address”. It is very important. The “Module Address” for D/O modules can only be set as 0 to 7, while D/I “Module Address” can only be set as 8 to 15.

For normal usage the “Module Address” settings for D/O and D/I must be set as different. In special case, it allows setting D/O as the same “Module Address” but it just repeats the number of D/O, the D/O channels with the same “Module Address” has the same output value. So, each FRnet port can connect more than 8 FRnet D/O “Module Address” and up to 8 D/I “Module Address”. (The D/I “Module Address” cannot be the same)

The max. I/O channel number for one FRnet “Module Address” is 16. That means one I-8172W can connect max. 2 (ports) x 8 x 16 = 256 ch. of digital output plus max. 2 x 8 x 16 = 256 ch. of digital input (uPAC-7186EG + FX-016 supports max. 128-ch. D/I plus 128-ch. D/O). You may plug up to 8 pcs. of I-8172W (max. 2048-ch. D/I plus 2048-ch. D/O) in the WinPAC-8847 depends on your application. (Max. 4 pcs. of I-8172W can plug in the iPAC-8447 / 8847, it supports max. 1024-ch. D/I plus 1024-ch. D/O).

Note:
ISaGRAF 3.x can program FRnet I/O by using “I-8172” (for iP-8xx7, WP-8xx7 , W-8xx7) or “Frnet86” I/O complex equipment (for uPAC-7186EG) & “fr_16di”, “fr_16do” & “fr_b_a” functions.

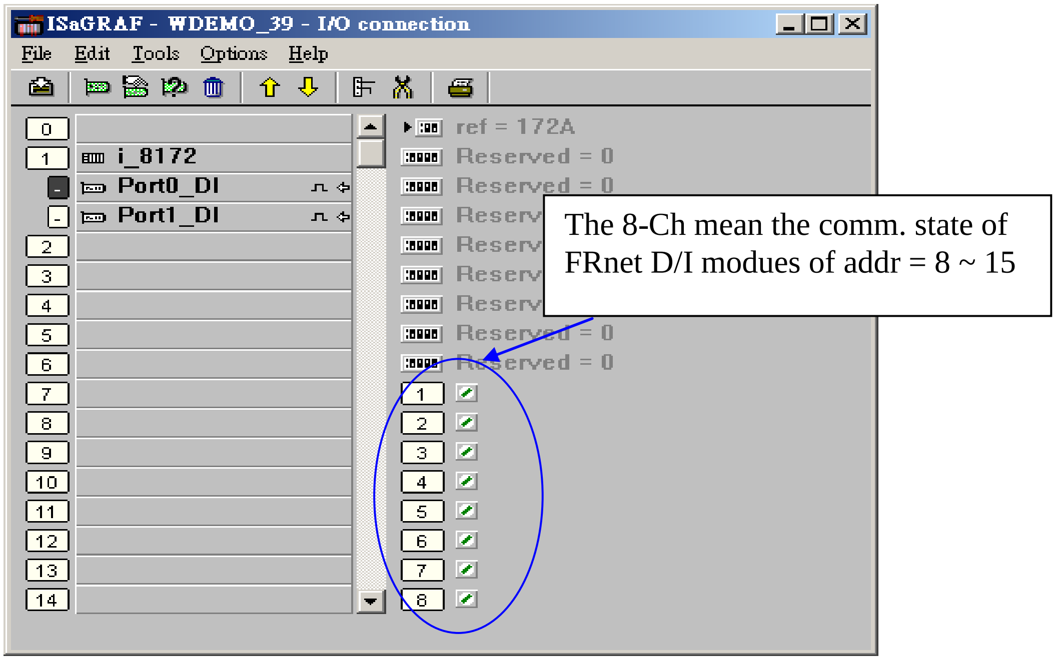
If your ISaGRAF doesn't support them, please visit http://www.icpdas.com/products/PAC/i-8000/isagraf.htm > Driver to download “ICP DAS Utilities For ISaGRAF.zip” to install it again to the ISaGRAF workbench.

Or refer to Appendix A.2 of the ISaGRAF User's Manual to restore - c-function: “fr_b_a” and c-function-block: “ fr_16di”, “fr_16do” and IO complex-equipment: “i_8172”, “frnet86” into your ISaGRAF Workbench.

Below is a demo program show you how to program FRnet I/O. This ISaGRAF example program can be download at www.icpdas.com > FAQ > Software > ISaGRAF > 082 or visit ftp://ftp.icpdas.com/pub/cd/winpac-8xx7/napdos/isagraf/wp-8xx7/demo/ to download “wpdmo_70.pia” (For WP-8xx7, iP-8xx7)

23.2: Programing the FRnet I/O

Step 1: Connecting I-8172 in the related slot in the IO connection windows (if using uPAC-7186EG + FX-016, please connect “frnet86”). WinPAC-8xx7's slot No. is from 0 to 7 (max. 8 pcs). iPAC-8447 / 8847's slot No. is from Slot 0 to 7 (max. 3 pcs). The uPAC-7186EG can connect “frnet86” on slot 0.



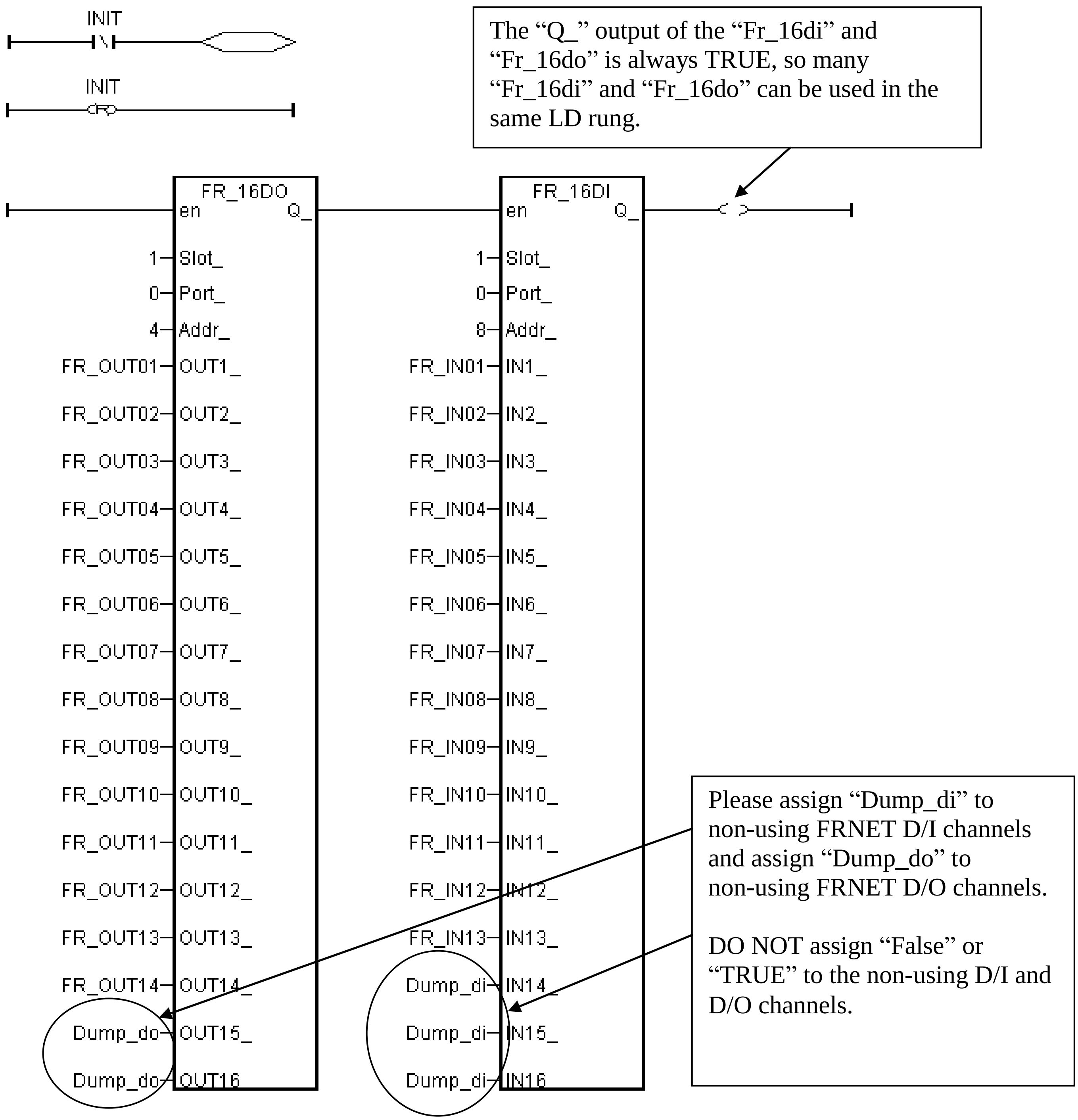
Step 2: Declaring ISaGRAF variable

Name	Type	Attribute	Description
INIT	Boolean	**Internal**	**Init as True**
Dump_di	Boolean	**Internal**	boolean variable for non-using FRnet D/I channel
Dump_do	Boolean	**Internal**	boolean variable for non-using FRnet D/O channel
FR_IN01 ~ FR_IN16	Boolean	**Internal**	Will map to 16-chanel FRnet DI channels
FR_OUT01 ~ FR_OUT16	Boolean	**Internal**	Will map to 16-chanel FRnet DO channels

Step 3: writing LD program to map ISaGRAF boolean variable as FRnet I/O

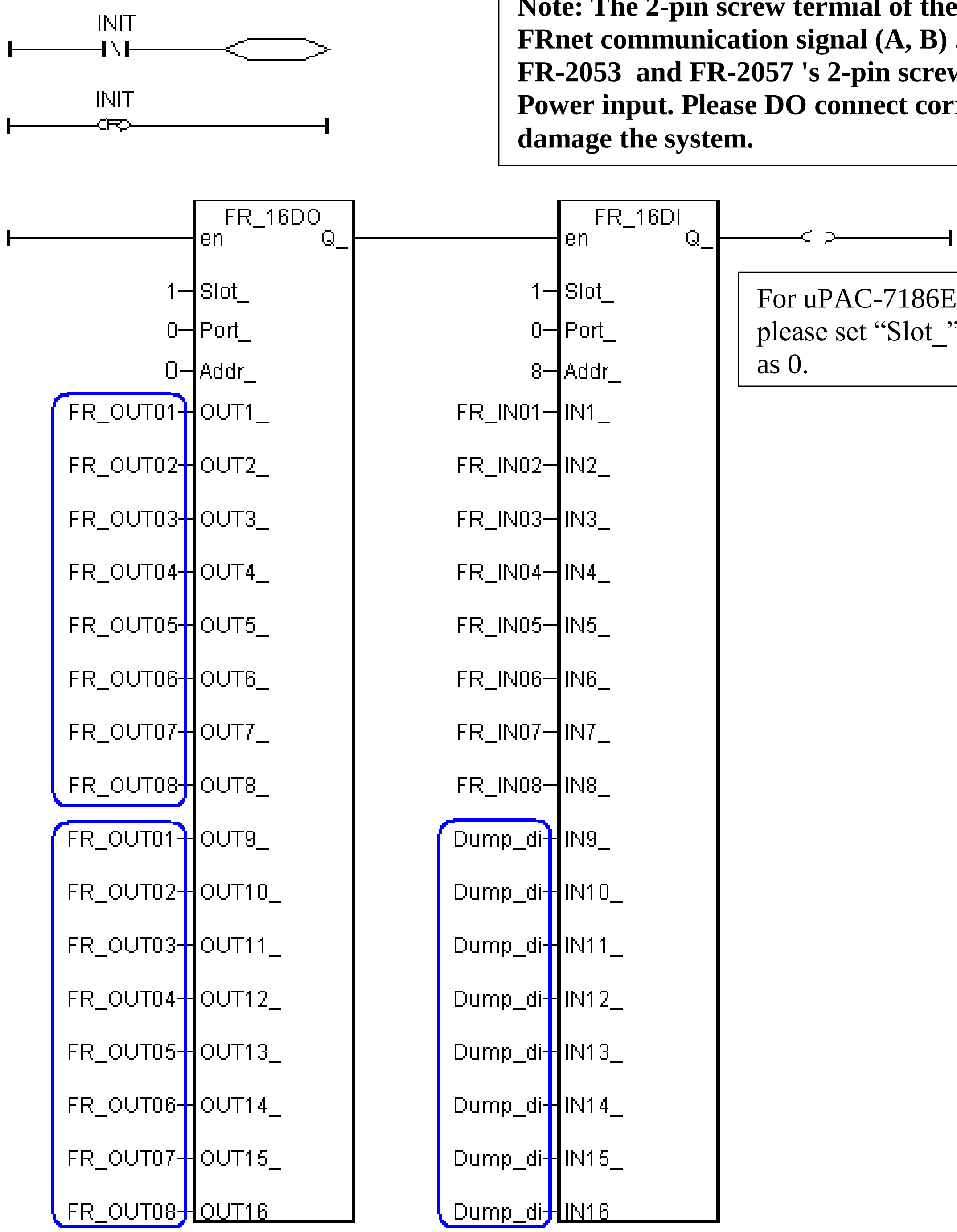
(* INIT should be declared with an initial value of TRUE. The below code can only run once in the first PLC scan cycle, please don't use "Fr_16di" and "Fr_16do" in other PLC scan cycles. The "Fr_16DI" and "Fr_16DO do not support array vareable, but you can use "FR_B_A" in the section 23.3 "*)

Example1: FR-2053 (16 IN) and FR-2057 (16 OUT)



Example 2: FR-2054 (8 IN and 8 OUT)

Note: The 2-pin screw termial of the FR-2054 is the FRnet communication signal (A, B) . However the FR-2053 and FR-2057 's 2-pin screw is the I/O Power input. Please DO connect correctly or it will damage the system.

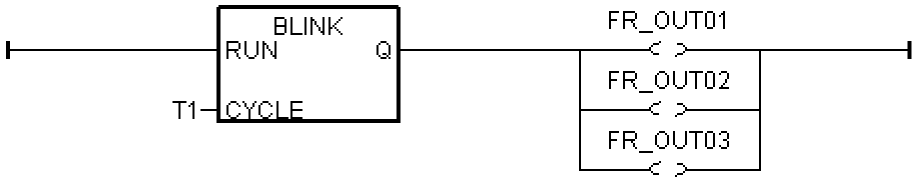


For uPAC-7186EG + FX-016, please set “Slot_” and “Port_” as 0.

If the connected module is the FR-2054 (8 IN and 8 OUT), its output code is a little different. Please connect the same variable names of the 1st through 8th position to the 9th through 16th position of the “FR_16DO” function block.

The FR-2054 's Dip switch ADDR setting is to set the DO module address, its DI module address will be auto-configured as “DO ADDR + 8”. For ex, setting 3 as ON, Dip 1 and 2 as OFF means DO ADDR is 4 , then DI ADDR will become 12. The above figure is for the FR-2054 with its ADDR setting as 0 (Dip 1, 2 & 3 are all OFF) , so its DI ADDR will be 8.

Step 4: Write a Ladder program to blink output of FR_OUT01 to 03



Step 5: How to test ?

1. Please plug I-8172W into slot 1 of the WinPAC-8xx7 (The WinPAC's left-most slot is 0).
2. Please connect the I-8172W's Port 0 in this demo to one FR-2053 (16-Ch. DI) and one FR-2057 (16-Ch. DO).

   The FR-2053's ADDR = 8 (DIP Switch 4 = ON, other dips 1,2,3, ,5,6,7,8 is OFF)
   The FR-2057's ADDR = 4 (DIP Switch 3 = ON, other dips 1,2, ,4,5,6,7,8 is OFF)

   Connecting the I-8172W's Port 0-A to the FR-2053's termial "A" and then connect to the FR-2057's termial "A". Connecting the I-8172W's Port 0-B to the FR-2053's termial "B" and then connect to the FR-2057's termial "B".

Note:

FRnet DO module, for example FR-2057 can only set module ADDR as 0 to 7
FRnet DI module, for example FR-2053 can only set module ADDR as 8 to 15

Then after you download this ISaGRAF project into WinPAC-8xx7 or iPAC-8x47, you will see the FR-2057's DO1 to 3 is blinking in the period of 0.5 second.

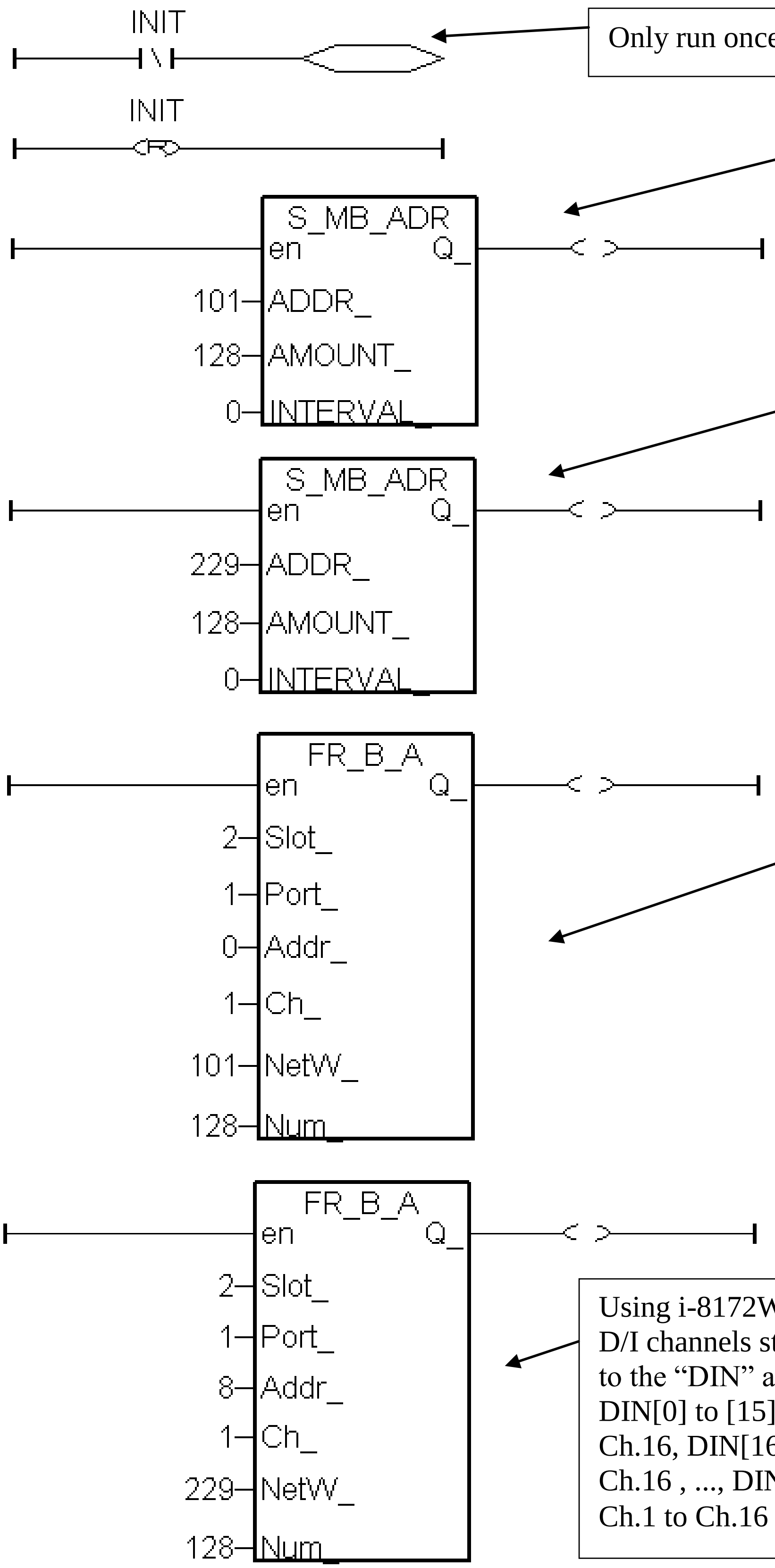
This ISaGRAF example program is "Wdemo_070.pia" which can be download at:
WP-8xx7 CD-ROM: \napdos\isagraf\wp-8xx7\demo or
ftp://ftp.icpdas.com/pub/cd/wincon_isagraf/napdos/isagraf/wincon/demo/ or
www.icpdas.com > FAQ > Software > ISaGRAF > 082

23.3: Using "FR_B_A" Function to Reduce the Program Size

Some application may use many FRnet I/O channels in the WP-8xx7 or iPAC-8x47. If using "FR_16DI" and "FR_16DO" in this kind of program, the size will become complicated and large. To low down the program size, user may use "FR_B_A" and Boolean variable array in the program.
(uPAC-7186EG + FX-016 also supports FR_B_A)

Variable declaraction sample in the following page:
INIT ia as Internal / Boolean and with an **initial value at TRUE**
DOUT is as Boolean variable array, dimension is 128, that is DOUT[0..127]. Please assign its **Network addr as 101** (65h)
DIN is as Boolean variable array, dimension is 128, that is DIN[0..127] ,Please assign its **Network addr as 229** (E5h)

(Please refer to www.icpdas.com > FAQ > Software > ISaGRAF > 039 for more about Variable Array)



Only run once in the first PLC scan

Assign DOUT[0..127] 's Network addr as 101 to 228 (Please make sure DOUT is declared as array , DIM = 128 and with a Network address number 101 in the dictionary window)

Assign DIN[0..127] 's Network addr as 229 to 356 (Please make sure DIN is declared as array , DIM = 128 and with a Network address number 229 in the dictionary window)
DO NOT wire two or more S_MB_ADR and FR_B_A in one row, because they may not return TRUE

Using i-8172W 's Port 1 in slot 2 , map 128 FRNET D/O channels starting from Module Addr 0, Channel 1 to the “DOUT” array variable. This sample will be DOUT[0] to [15] mapping as Module Addr 0 's Ch.1 to Ch.16, DOUT[16] to [31] as Module Addr 1 's Ch.1 to Ch.16 , ..., DOUT[112] to [127] as Module Addr 7 's Ch.1 to Ch.16 .

For uPAC-7186EG + FX-016, please set “Slot_” and “Port_” as 0

Using i-8172W 's Port 1 in slot 2 , map 128 FRNET D/I channels starting from Module Addr 8 , Channel 1 to the “DIN” array variable. This sample will be DIN[0] to [15] mapping as Module Addr 8 's Ch.1 to Ch.16, DIN[16] to [31] as Module Addr 9 's Ch.1 to Ch.16 , ..., DIN[112] to [127] as Module Addr 16 's Ch.1 to Ch.16 .

Chapter 24. Using "COM" Functions TO Read/Write the RS-232/422/485 Port

ICP DAS ISaGRAF controllers support below Serial COM Port (RS-232/422/485) protocols:

Modbus RTU Slave	Refer to Chapter 4 of the ISaGRAF user's manual & respective getting started manual
I-7000 and I-87xxx RS-485 I/O	Refer to Chapter 6 of the ISaGRAF user's manual
Modbus RTU Master (M-7000)	Refer to Chapter 8 and 21 of the ISaGRAF user's manual
Modbus ASCII Master	Refer to Chapter 8 of the ISaGRAF user's manual
SMS : Short Message Service	Refer to Chapter 17 of the ISaGRAF user's manual

User can apply below COM functions to operate other protocols or 3rd party protocols. (Please refer to Appendix A.4 of the ISaGRAF user's manual for description of these COM functions)

COMOPEN	Open Serial COM Port (without "Flow control" parameter)
COMOPEN2	Open Serial COM Port (with "Flow control" parameter, not for I-8xx7)
COMREADY	Test if any byte come in
COMARY_R	Read all bytes which already come in to a byte array
COMARY_W	Write many bytes in a byte array to COM Port
COMREAD	Read one bytes (Please call "COMREADY" to test first, if there is data, then "COMREAD" can be called)
COMCLEAR	Clear all received bytes in the receiving buffer
COMARY_NW	Write one signed long Integer to COM Port, format is Binary, 4-byte
COMARY_WW	Write one signed Word to COM Port, format is Binary, 2-byte
COMSTR_W	Write one string to COM Port
COMWRITE	Write one byte to COM Port
COMCLOSE	Close Serial COM Port

Note:

1. The default shipping of I-8xx7 controller has set its COM1 and COM2 (COM2: RS-485 is only for I-8417/8817) as Modbus RTU Slave Port. User can choose to switch off the COM1: Modbus RTU Slave function to become a freely used COM port by the above listed COM functions. (Please refer to Appendix C.1). To use I-8xx7 's COM5 to COM20, Please refer to Chapter 1.8 to install I-8112/8114/8142/8144 serial expansion boards.

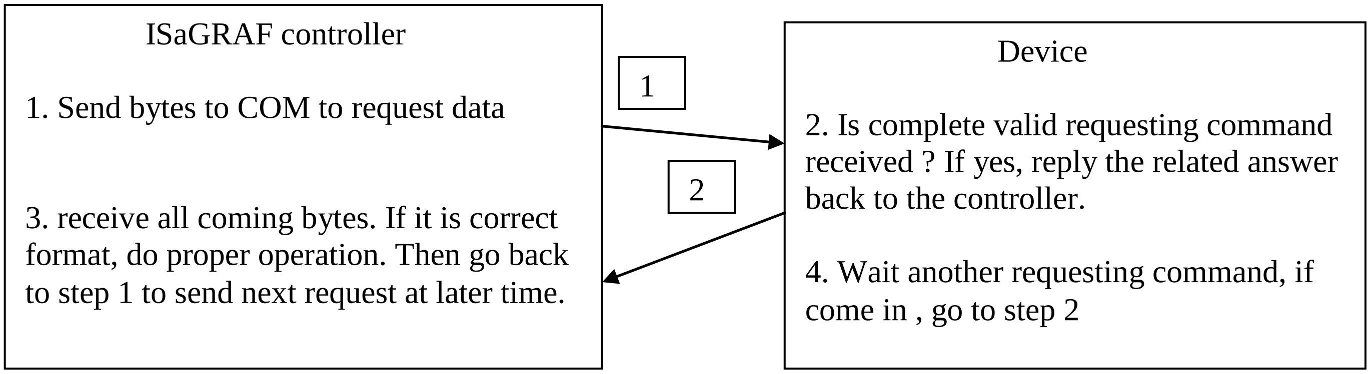
2. W-8xx7 / 8xx6 's COM2 / COM3 can be switched ON as a Modbus RTU Slave Port. Or Switch Off for freely used. (Refer to Appendix A.2 of its Getting Started manual delivered with the hardware). To use WinCon's COM5 to COM14 at I-8112/8114/8142/8144 serial expansion boards, please refer to Appendix E of the "Getting Started:WinCon ISaGRAF PAC" manual.

3. COM1 of I-7188EG / µPAC-7186EG/ µPAC-5xx7 is set as Modbus RTU Slave port when shipping. User may switch it OFF to freely use it by COM port functions. (Please refer to Section 3.6 of its "Getting Started Manual" delivered with its hardware). However I-7188XG 's COM1 can not be switch OFF, it is always Modbus RTU Slave port. If user want to use COM3 to COM8 of I-7188EG/XG and 7186EG, please plug one extra X-5xx expansion I/O board inside it . http://www.icpdas.com/products/PAC/i-o_expansion/x_list.htm

The following will introduce the most common methods of communication.

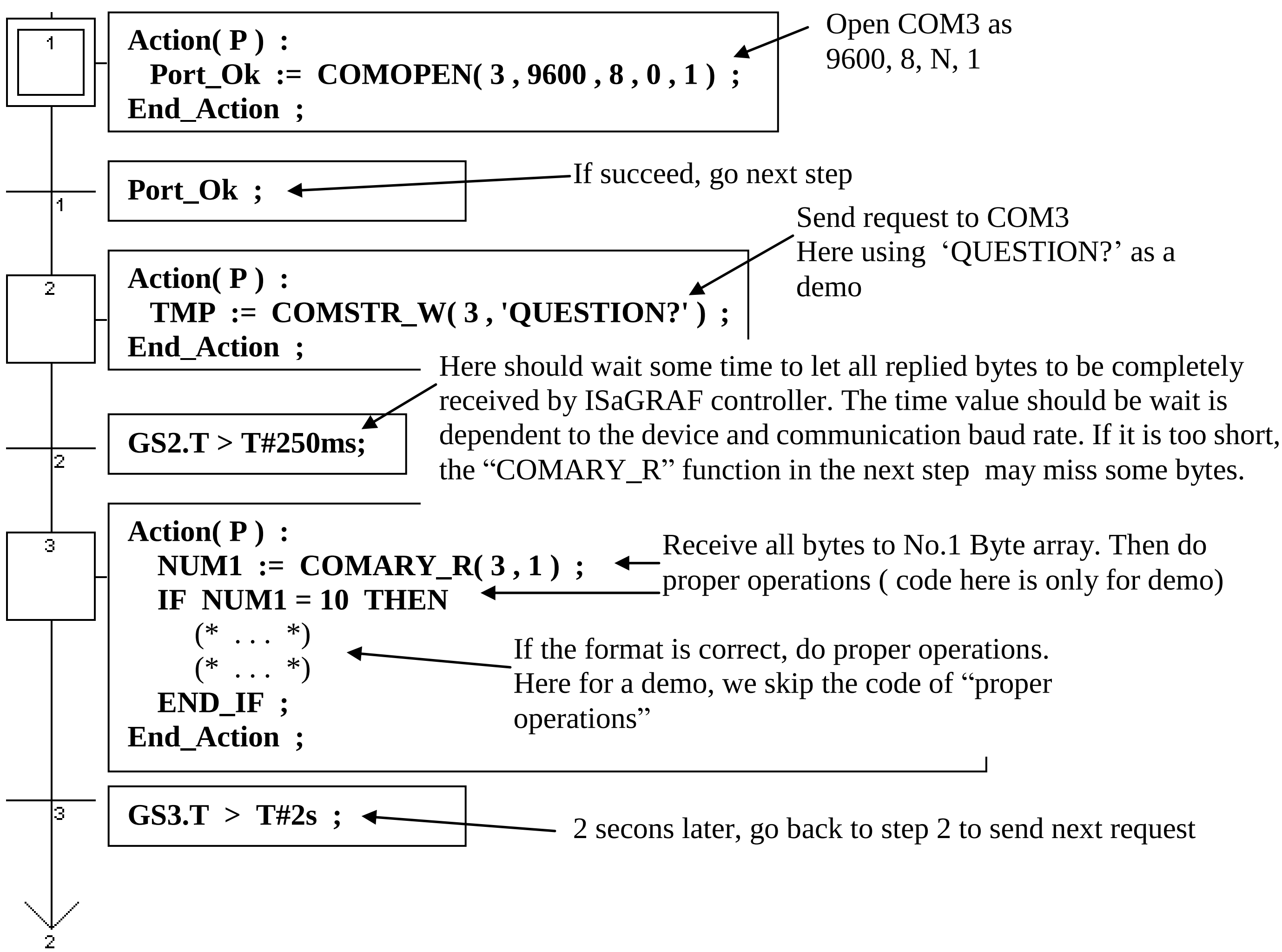
24.1: Cntroller send 1 request and get 1 reply from device

Below figure lists the most common RS-232 / 422 / 485 application.



User can use the below code or similiar code to do it.
Below example will send a string “QUESTION?” to device via COM3 , then waiting device to reply the related answer. And then 2 seconds later, send next same question to device , …

SFC program: (“Port_OK” & “TMP” is Boolean Internal , “NUM1” is Integer Internal)



24.2: Controller just wait data from the remote device

This kind of application is very common in the store. Like the device of "Bar code reader", when it reads bar code on the product, it will send the related data to the controller via RS-232 / 422 / 485. The controller just receive it, not necessary to send any byte to device. (Please visit www.icpdas.com – FAQ – Software_ISaGRAF-066 for demo program and more information.)

ISaGRAF controller

2. If receive correct data, do proper action.

3. Then wait next data .

1

Device

1. When event happens, send data to controller

ST program:

```
IF  INIT  THEN
   INIT  :=  FALSE ;
   Port_Ok  :=  COMOPEN( 3 , 9600 , 8 , 0 , 1 ) ;
   T1  :=  T#0s ;
   STEP  :=  0 ;
END_IF  ;

IF  Port_Ok = False  THEN
   Return ;
END_IF ;

CASE  STEP  OF

 0 :   IF  COMREADY(3)  THEN
          STEP := 1 ;
          T1 := T#0s ;
          Tstart(T1) ;
       END_IF ;

 1 :   IF  T1 > T#250ms  THEN
          Tstop(T1) ;
          T1 := T#0s ;
          STEP := 0 ;
          NUM1 := COMARY_R(3, 1) ;

          IF  NUM1=10  THEN
              (* … *)
          END_IF ;
      END_IF ;

END_CASE ;
```

Variable declaration
"INIT" is Boolean Internal, init as TRUE
"Port_Ok" & "TMP" is Boolean Internal
"T1" is Timer Internal
"STEP" & "NUM1" is Integer Internal

Open COM3 as 9600, 8, N,1

If open fail, just quit this ST program

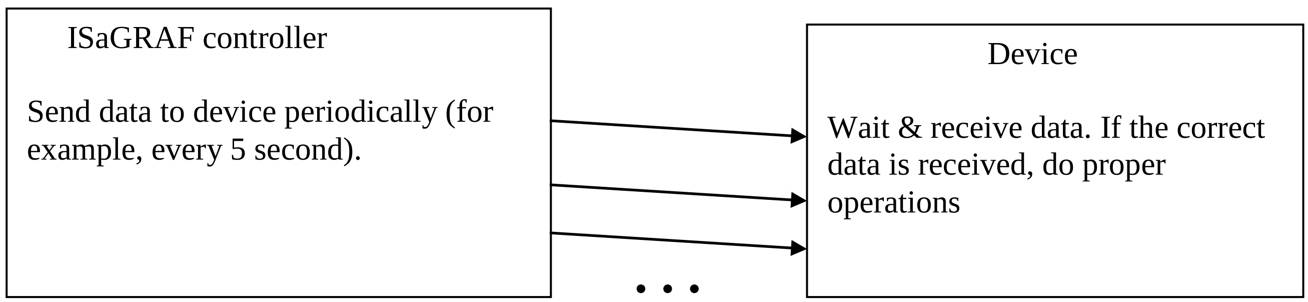
STEP=0 means "waiting state", it should check if any byte come in. If "COMREADY" returns TRUE, it means there is at least one byte come in. Then set "STEP" to 1 and start timer "T1" to tick.

STEP=1 means "data is coming", when "T1" tick to the specified waiting time, call "COMARY_R" to receive all bytes to No.1 byte array (this will make sure all bytes from the "device" all well received). This waiting time is dependent with the "Device" and the communication baud rate. Here we use 0.25 second as a demo. (If setting too short than your device should have, some bytes may be missing) Remember to set "STEP" to 0 to wait next data.

Do proper operation if receive correct data.
The code listed here is only for demo. (It is depend with your application, we just assume the correct data has 10 bytes here.)

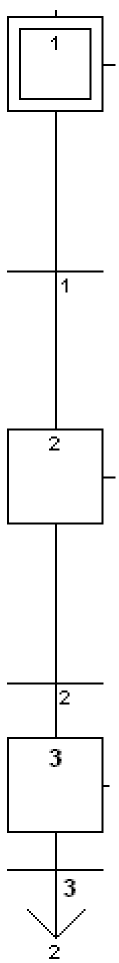
24.3: Report data to remote device periodically

If the ISaGRAF controller need to send data to other device or PC every a short time by using its RS-232/422/485 COM Port, just like below:



User can use the similar program as below.

SFC program: (Please declare “TMP” & “Port_Ok” as Boolean Internal)




```
Action( P ) :
  Port_Ok := COMOPEN( 3, 9600, 8, 0, 1 ) ;
End_Action ;
```

Open COM3 as 9600, 8, N, 1

```
Port_Ok ;
```

If succeed, go next step

```
Action( P ) :

   (* Prepare all bytes in No.1 byte array *)
   (* … *)

   TMP := COMARY_W( 3 , 1 , 20 ) ;
End_Action ;
```

The data to be sent depends on your application. Please prepare them and call COMARY_W or COMSTR_W to send it

Here using “COMARY_W” to send 20 bytes in No.1 byte array to COM3. Or “COMSTR_W” may be used for sending string (string length is max. 255 bytes) , there should be no 0 value in the string. (because 0 is string-end byte, however character ‘0’ is Ok in the string).

```
GS2.T > T#5s ;
```

5 seconds later, first jump to an empty step3, then go to step 2 to send next data

(SFC program will operate error if jumping to the same step after it is finished. So we should jump to an empty step first. The ISaGRAF compiler generate no erro if not jumping to an empty step first. But its result will be error.)

24.4: Controller send data when event happens

Please refer to Chapter 11.3.5