User Manual

APAX-5520KW

Software Manual

ADVANTECH

eAutomation

Copyright

The documentation and the software included with this product are copyrighted 2009 by Advantech Co., Ltd. All rights are reserved. Advantech Co., Ltd. reserves the right to make improvements in the products described in this manual at any time without notice. No part of this manual may be reproduced, copied, translated or transmitted in any form or by any means without the prior written permission of Advantech Co., Ltd. Information provided in this manual is intended to be accurate and reliable. However, Advantech Co., Ltd. assumes no responsibility for its use, nor for any infringements of the rights of third parties, which may result from its use.

Acknowledgements

Intel and Pentium are trademarks of Intel Corporation.

Microsoft Windows and MS-DOS are registered trademarks of Microsoft Corp.

MULTIPROG and ProConOS are registered trademarks of KW-Software GmbH Lemgo (Germany)

All other product names or trademarks are properties of their respective owners.

Notes on the Manual

This is the Software Manual for the Advantech APAX-5520 product. This manual will help guide the end user through implementation and use of the software portion of this product.

What is covered in this manual:

This manual will give a general overview of the Windows CE operating system, most of the applications that are included with Windows CE as well as the applications added and/or created by Advantech Corporation in the Windows CE image. This manual will also cover installation and use of development and utility software that is needed. It will also reference optional software that can be used by the end user with the Windows CE Operating system. There is a section on programming tips for the MULTIPROG software, but this will not cover all MultiProg functions. This can be referenced from the online help from the MULTIPROG software.

What is not covered in this manual:

This manual will reference the hardware but does not contain hardware setup information, wiring information, electrical specifications or any detailed hardware information. Please refer to the hardware manual for this information.

For detailed MULTIPROG information or IEC-61131 programming, see the online help when using MULTIPROG.

For detailed ProConOS information, see the ProConOS user's manual.

Edition 1

September 2009

Contents

Chapter 1

Introduction

1.1 Overview

APAX-5520 is part of Advantech's Programmable Automation Controller series. APAX-5520 uses Windows CE.NET and its real-time capabilities along with controller software provides a soft real-time controller. The controlling portion of this product consists of two pieces of software, the development software (MULTIPROG) and the runtime software ProConOS. MULTIPROG will reside on the developer's computer, where the developer will create programs to download to the runtime software on the APAX-5520. The connection is a proprietary connection via TCP/IP.
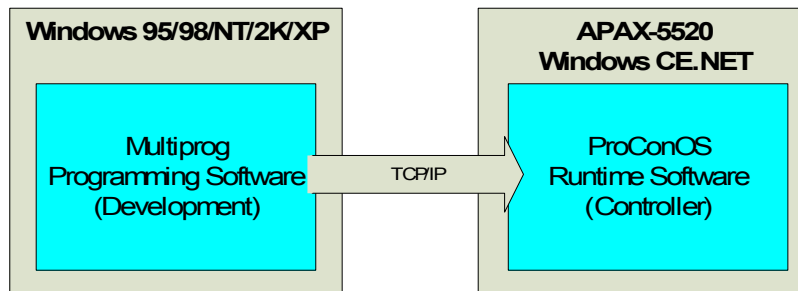
1.1.1 Development Software

MULTIPROG provides the tools to develop and download the project to the runtime. It also provides the developer with online capabilities for debugging and monitoring.

1.1.2 Controller Software

ProConOS stands for Programmable Controller Operating System. ProConOS is a product from KW Software that has been integrated with Advantech's APAX-5520 hardware. This is an executable program that uses the highest priority threads in Windows CE to perform real time execution. In the case of Advantech's implementation, Windows CE and ProConOS share processing time, each of which run at 1ms. WinCE is interrupted at the lowest level to insure real time performance from ProConOS. In depth discussion of ProConOS is provided in the **ProConOS Manual** provided with this product. It is highly recommended that the end user read this manual for a better understanding of ProConOS and its capabilities.

Note!

KW MultiProg and ProConOS are complaint with IEC 61131-3 to include programs written by the popular programming languages:

Text Languages:
1. Instruction List (IL)
2. Structure Text (ST)

Graphic Languages:
3. Function Block Diagram (FBD)
4. Ladder Diagram (LD)
5. Sequential Function Chart (SFC)

KW supports cross-language programming. For example, you can use Ladder Diagrams (LD) on the simple I/O module control, and use Function Block Diagrams (FBD) on process control for more advanced expressions, and use Sequential Function Chart (SFC) for system configuration in hybrid control system such as water treatment applications.

1.2 General Information

1.2.1 Software Installation

There are two main installations that are required for the APAX-5520 on the development computer, MULTIPROG and the Advantech MULTIPROG Add on. While the MULTIPROG installation will provide a working development environment, it will not work with Advantech products without first installing the "Advantech MULTIPROG Add on" that is provided on the CD. The software requirements to run MULTIPROG includes one of the following operating systems:

- Microsoft Windows 95
- Microsoft Windows 98
- Microsoft Windows NT 4.0 (SP6)
- Microsoft Windows 2000 (SP2)
- Microsoft Windows XP
- Microsoft Windows ME
- Microsoft Internet Explorer 4.02 or greater

1.2.2 ProConOS (Runtime)

ProConOS and all the support software come pre-installed on the APAX module. There is no configuration needed for ProConOS. ProConOS will automatically run when the APAX is started and contains a server that will listen on the TCP/IP network for a connection from MULTIPROG. The only configuration that has to be done to the APAX is to set the IP address in Windows CE.

Chapter 2

Windows CE.NET

2.1 WinCE Image

Advantech has engineered the Windows CE.NET embedded image exclusively for this hardware. It contains specific drivers for the APAX-5520 and is designed and licensed only for this hardware.

2.2 Modification of Standard Image

While the WinCE image is considered an embedded image, it is possible for the developer to add their own developed software to the image if done properly. This is possible through the Microsoft Visual Studio .NET programming environment. Users can create and deploy their own applications through this tool along with the libraries distributed by Advantech (see below for more information). Users can also make changes to an image and have that image deployed on subsequent purchased images through a Configure to Order specification (CTO). There may be a situation where a user needs modification of the standard image. Since the image is created by Advantech, this may be possible depending on the user requirements. A non reoccurring engineering fee (NRE) would most likely be required to create a custom image. Please check with your sales person for more information about the CTO and NRE services.

2.3 Connecting to the Device

2.3.1 DiagAnywhere

"DiagAnywhere", an abbreviation of "Diagnostic Anywhere", is a networking solution for remotely monitoring and controlling other Windows based devices. It is very similar to a remote desktop application with some additional features. Currently, "DiagAnywhere" includes the utility on client side, and the server on the other. The main technology is based on Microsoft .NET Framework for the client. For this reason, the PCs using this solution must have the Microsoft .NET Framework installed for Win32 platform. You can find the .NET Framework and DiagAnywhere client trial version on the CD that comes with the APAX-5520.

"DiagAnywhere" server can only run on Advantech's **TPC, UNO, AMAX, APAX** and **ADAM** Windows based devices. The supported platforms include Windows XP, Windows XPe and Windows CE.

However, the server can accept only one connection from the utility at a time, and other connection attempts will be rejected if there is a live connection. This server is set up to automatically start when Windows CE starts. APAX-5520 has built-in DiaAnywhere server and the server will launch automatically after the system boots. You can use DiagAnywhere client (The trial version is provided in the CD) to connect to the APAX-5520. There is no password by default. Then, you can remotely control APAX-5520 through Ethernet, including file transferring.

2.3.2 IP Address

The APAX-5520 will come with a default IP address set to 10.0.0.1 and 10.0.0.2. This IP address can be changed through DiagAnywhere to suit the users specific requirements. Refer to figure below. Double click the LAN port icon you want to change IP through Start>>Setting>>Control Panel>>Network and Dial-up Connection. You will see the configuration window as shown below. It is not recommended to use DHCP for the APAX-5520 because the project and other items connecting to the APAX-5520 will be programmed to specific IP address'.

Note! *You must save the registry after you update the IP address or your changes will be discarded upon reboot. This can be done from the start menu at "Start | Programs | Advantech | Registry Saver".*
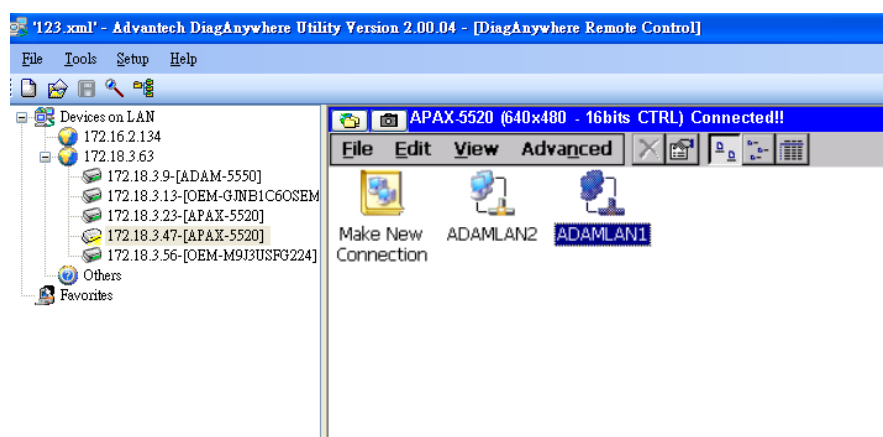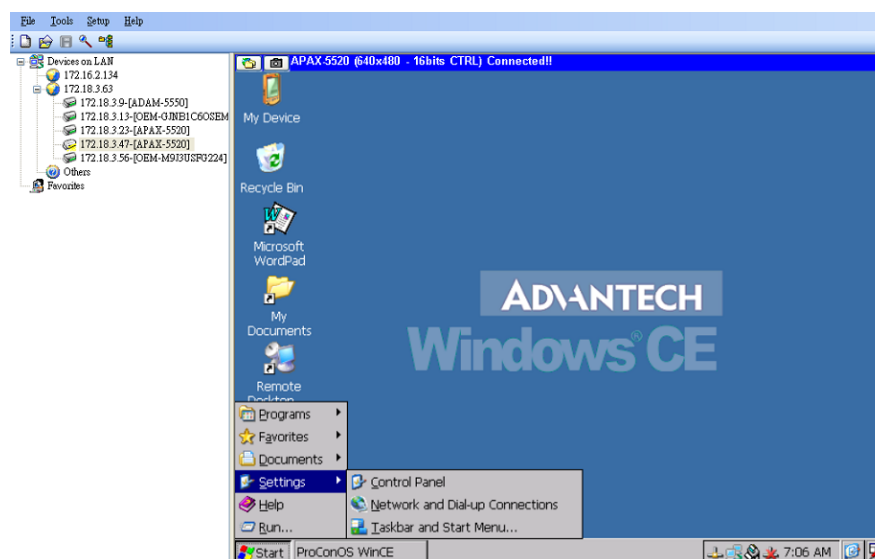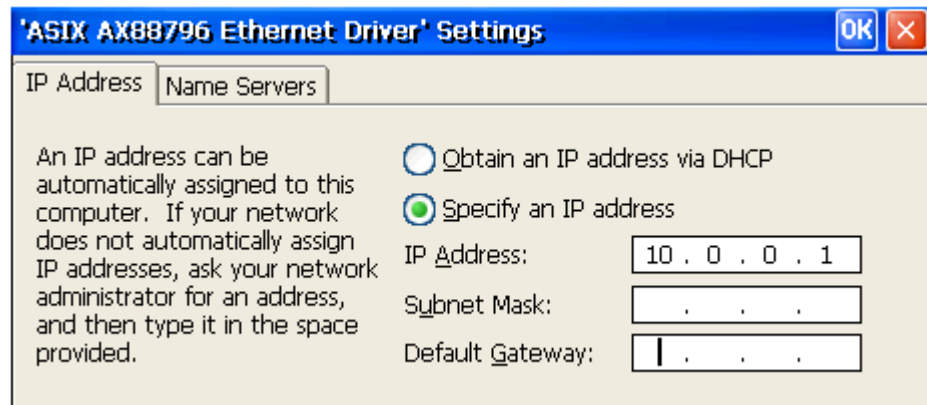
2.3.3 Connecting with MULTIPROG

When developing with Multiprog, the IP address must be set in the project and Pro-ConOS must be running to allow connection to the APAX . This will be covered further in the document.

2.4 WinCE Remote Tools

WinCE Remote tools are a set of Microsoft administration tools provided via web server on the APAX-5520. The remote tools are accessed by a web browser. The IP address of the APAX-5520 must be known in order to use the remote tools. It is important that you find your IP address either by setting a static address or getting the DHCP assigned address.

2.4.1 Remote Admin

Setup Administrator Password

The first time remote admin is connected to, the user must enter an administrator password. It is important that this step is done to keep the APAX-5520 protected. If the registry is lost or if the defaults are loaded, then this step must be done again.

Connect to the APAX-5520 via a web browser with the IP address that was previously set. Using the path xxx.xxx.xxx.xxx/RemoteAdmin

For the first time log on, the page will be redirected to the "DeviceConfig.htm" to set up the Admin password. Enter the Admin password and click the apply button.
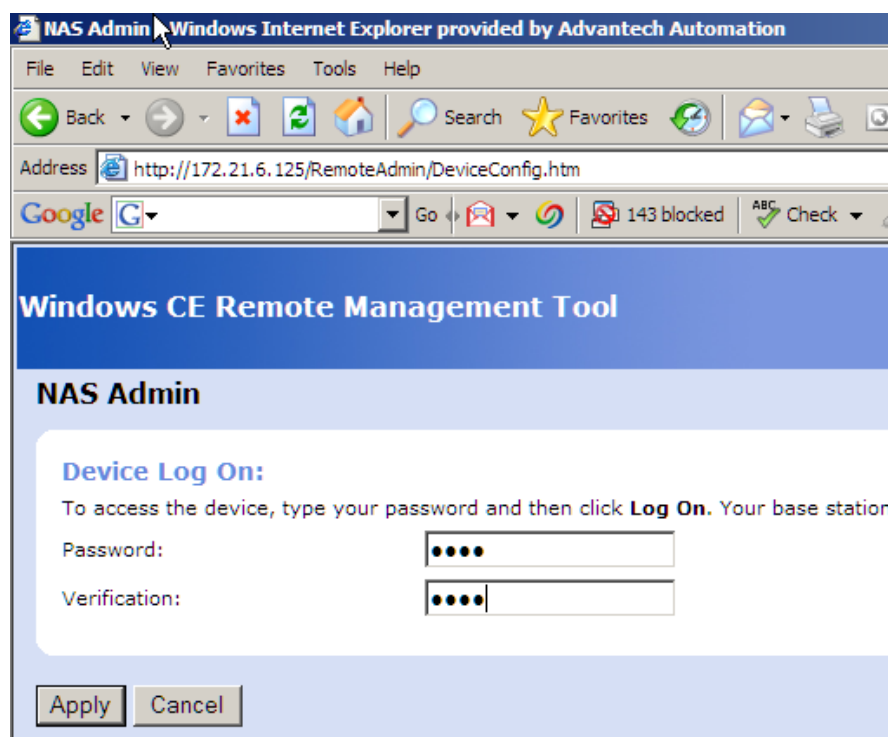
When the Apply button is clicked, the gateway will reset and the user is then prompted to log in with the new password.
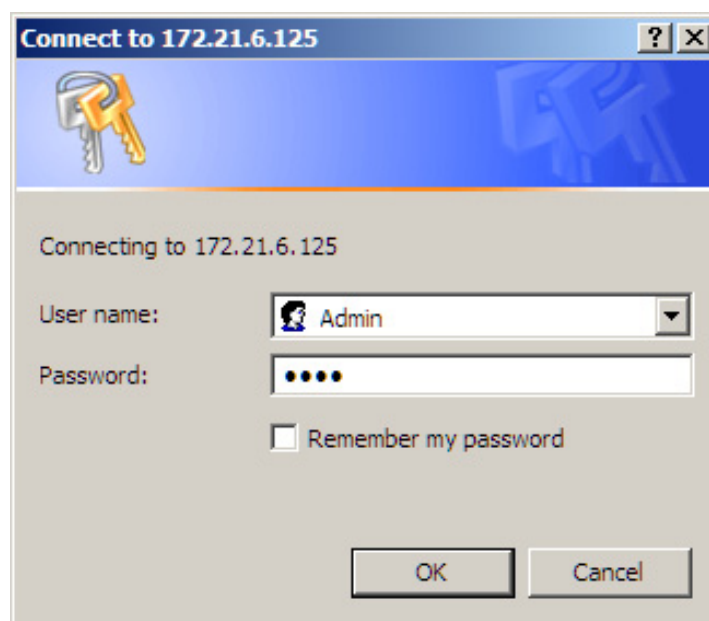
Once logged in, the user <u>must</u> change the device name. The device name box may have a sample such as APAX-5520. A suggestion is to change the name to "APAX-5520-1". Other controllers on the same network can have subsequent numbers or different names but all controller names on the same network must be unique.
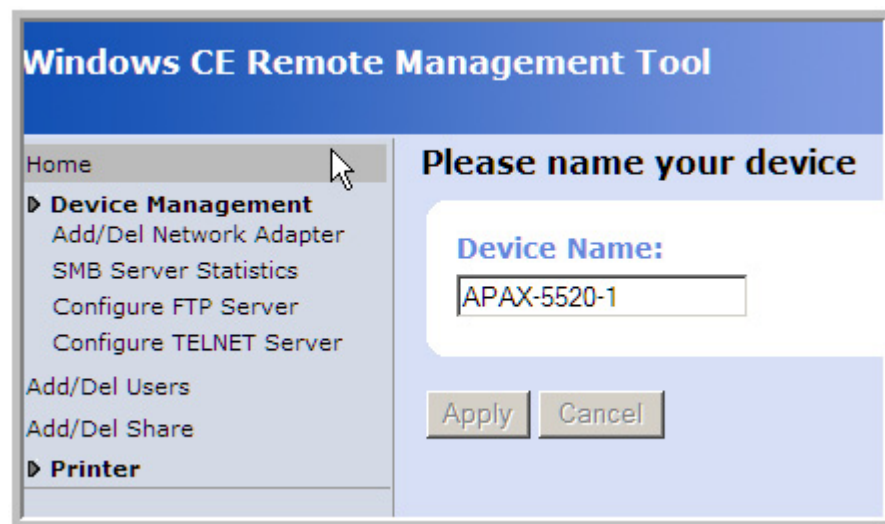
Once the device name is saved, the remote admin page will be displayed. From this page the following functions can be managed:

- Enable Network Adaptors for file share
- Configure FTP Server
- Configure TELNET Server
- Add/Delete Users
- Add/Delete file shares
- Add/Delete Printers

2.4.2 Remote Web Admin

Windows CE provides remote web server administration. This is located on a virtual root by typing in the address xxx.xxx.xxx.xxx/webadmin. The login and password will be the same for Remote Web Admin as the Remote Admin that was set in the previous section. The Web Server Administration (WebAdmin) page for the Windows CE Web Server enables you to remotely administer your Web server using your Web browser. Use WebAdmin to manage the accessibility, security, and file sharing features of your Web server, including the following tasks:

- Configure which files are shared and how they are accessed.
- Configure which users have access to which files.
- Configure the authentication protocols the Web server will use.
- View and configure the Web server log.
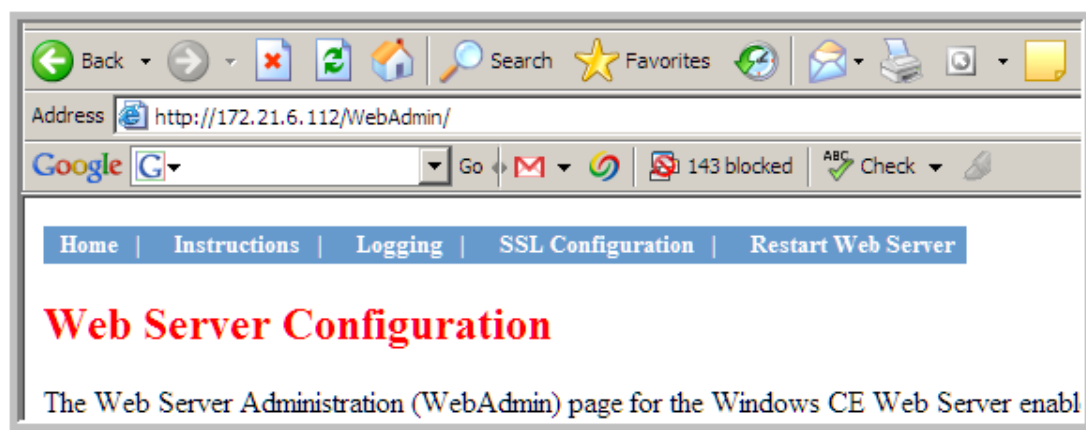
The web server configuration comes with its own instructions and help files. Please see these documents for further information.

2.4.3 Remote System Admin

Windows CE provides a remote system administration. This is located on a virtual root by typing in the address xxx.xxx.xxx.xxx/sysadmin. The login and password will be the same as the Remote Admin login and password. This interface includes the following tools:

- System Information viewer
- Process Management view and control
- File browser
- Registry editor

2.5 WinCE Applications

2.5.1 APAX.NET Utility

Advantech provides the APAX.NET utility which allows the developer/end user to interrogate the APAX bus, see connected modules and do simple testing of the I/O. This software can be helpful when checking wiring inputs prior to installing the runtime project. It is also able to detect and test other Advantech supported hardware for this product.

The installation file is contained in the CD and on our website at: *http://www.advantech.com* in the download area under the support page.
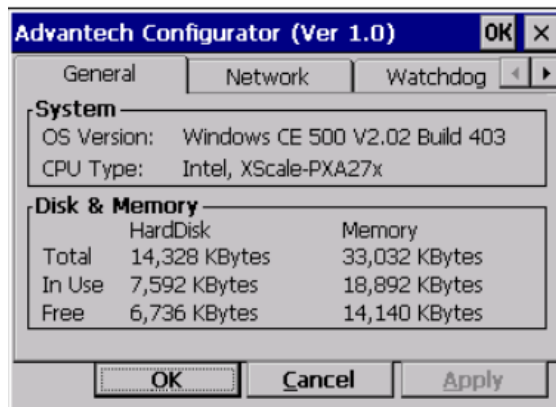
Detailed operation for APAX.NET utility can be found in Appendix B.
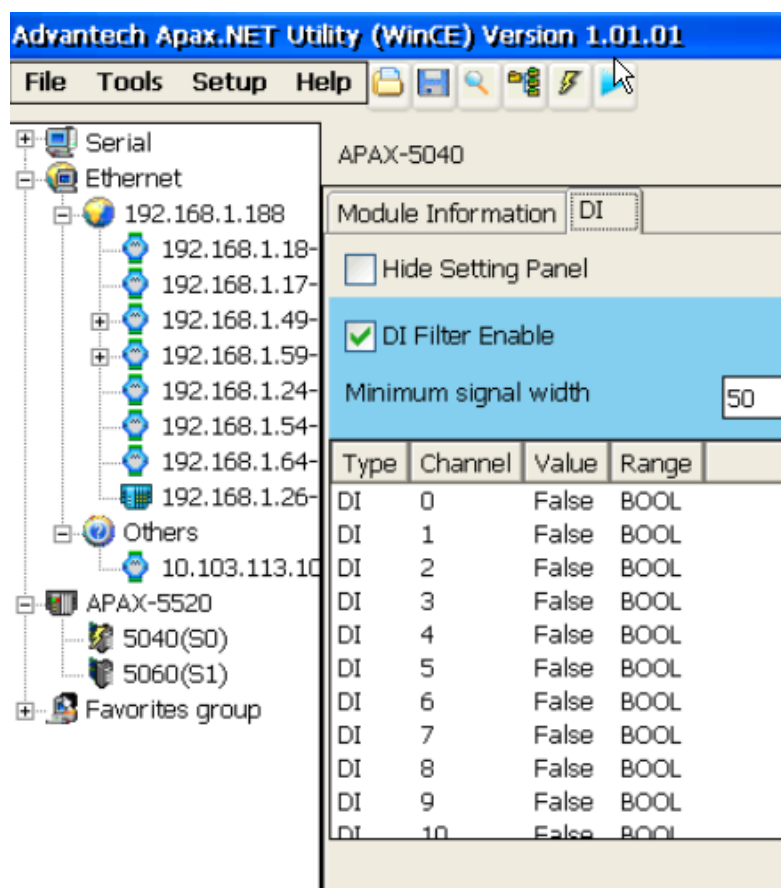
2.5.2 Advantech Configuration Utility

Advantech provides a tool called the Configuration Utility which can be accessed from the start menu through Start >> Programs >> Advantech >> Configuration Utility.. This tool provides the following items:

■ General: System and disk information is available here.

■ Network: The two LAN port information (such as MAC address, IP address, Subnet Mask, etc) is available here. If you configure the LAN port as DHCP, click the **Renew** button to get another ID. Click the **Ping** button to ping another device in the same network. Click the **Advanced** button for further information such as DHCP server or DNS server.

■ Watchdog: APAX-5520 offers built-in watchdog timer. It will continuously check the system and automatically reset the system if the system fails. Choose the periodical checking time for watchdog timer by the **Response Time** combo box and then enable the watchdog timer by the **Enable** button. Here, you also can test the watchdog timer.

■ Miscellaneous: You can define which program application should execute automatically when system boot-up by including it in the Startup Program. Use **Add** and **Delete** buttons to decide which programs become startup programs. There are other configuration for system such as Register, Web Server Root and FTP Server root. Click the **Reboot** button can help to reboot the system without power-off the system.
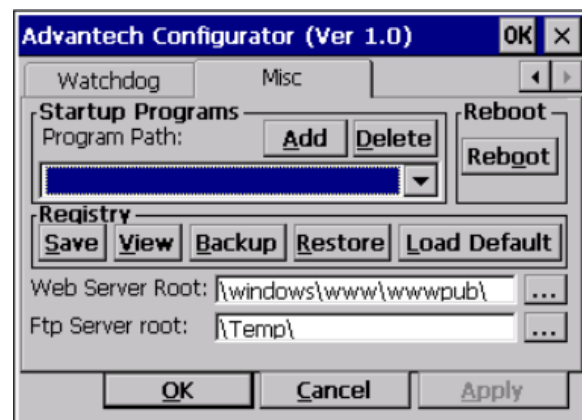
2.5.3 Advantech Version InformationTool

Advantech provides a simple reporting tool that will provide necessary version information for the Windows CE operating system as well as any post OS Build installations from Advantech. This is an important tool for determining what versions of Advantech Added software are on the APAX-5520 and may help during troubleshooting. Launch the Version Information Tool by selecting Start >> Programs >> Advantech >> Version Information.
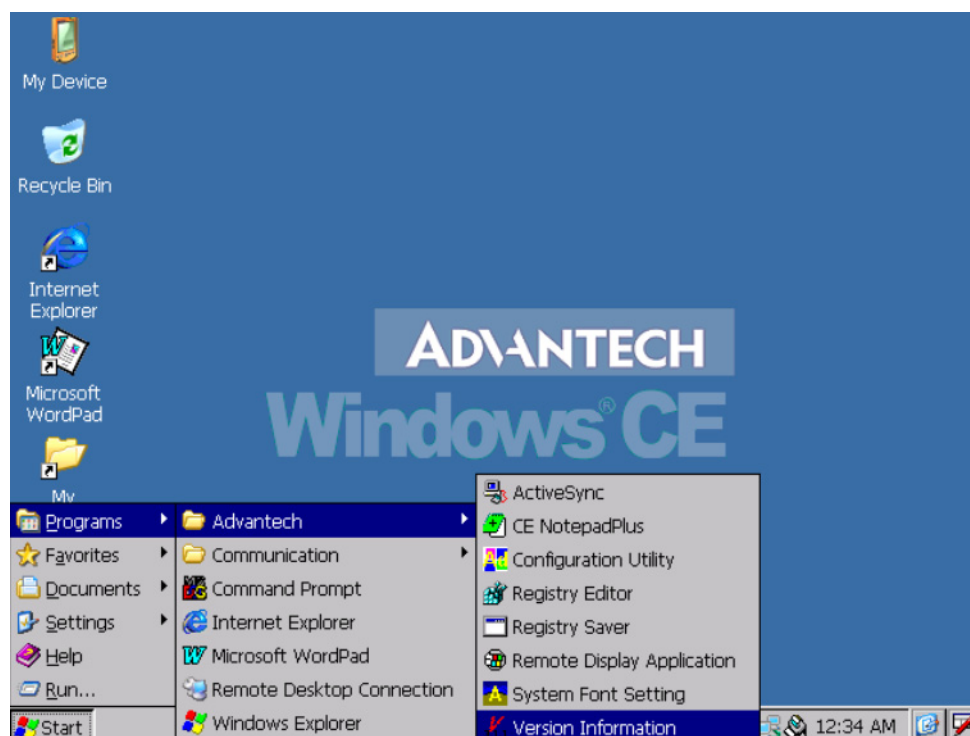
2.5.4 DiagAnywhere Server

APAX-5520 provides the DiagAnywhere Server to allow a connection from the DiagAnywhere client. The application is automatically started. If you choose not to use this program, you can disable the startup by using the Configuration Utility to remove it from startup. (Refer to Section 2.5.2)
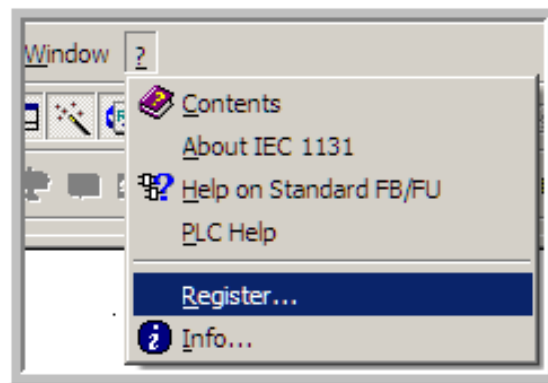
Chapter 3

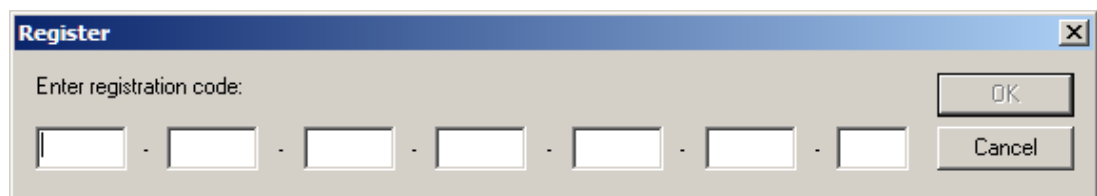## Programming

3.1 Programming with MULTIPROG

3.1.1 Licensing the Software

Multiprog will work for a limited time without licensing the development software. Once installed, you must enter the supplied license code as follows:

Select "?" from the menu bar and then "Register".

Enter the license key.


3.1.2 Quick Start

This quick start section is exactly that. It will show you how to create a very simple project, without drivers, download and run the project while seeing live variables in remote debug mode.

3.1.3 Notes on the Quick Start

By using this quick start, you will be able to create a simple application using a template provided by Advantech for the APAX-5520 hardware. This application will create one integer variable (tag) and continuously add the value 1 to it each time the PLC engine scan's. The template comes with a "default" task, which runs as a background task. You will remove this task and replace it with a task that runs at a specified scan rate of 100ms. Once this is done, you will download and run the program and see the live data update. This quick start should only take a few minutes to complete.

Create a Simple Project

1. Set the IP address for as shown in section "Connecting to Device".
2. Open Multiprog, Select New Project, Select APAX-5520KW from the list.
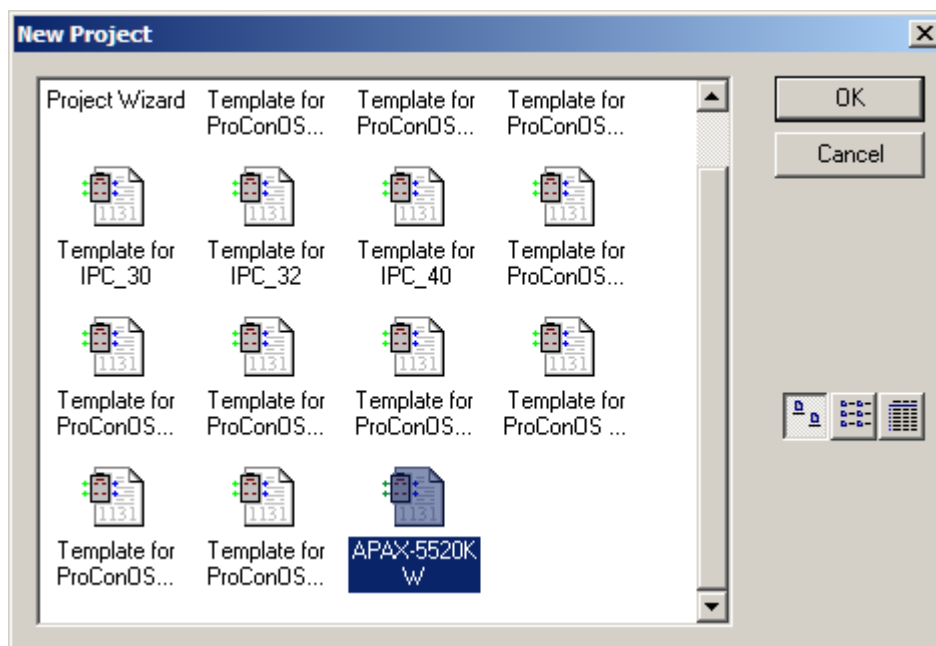
3. Right click "Resource" in the project tree and select settings.
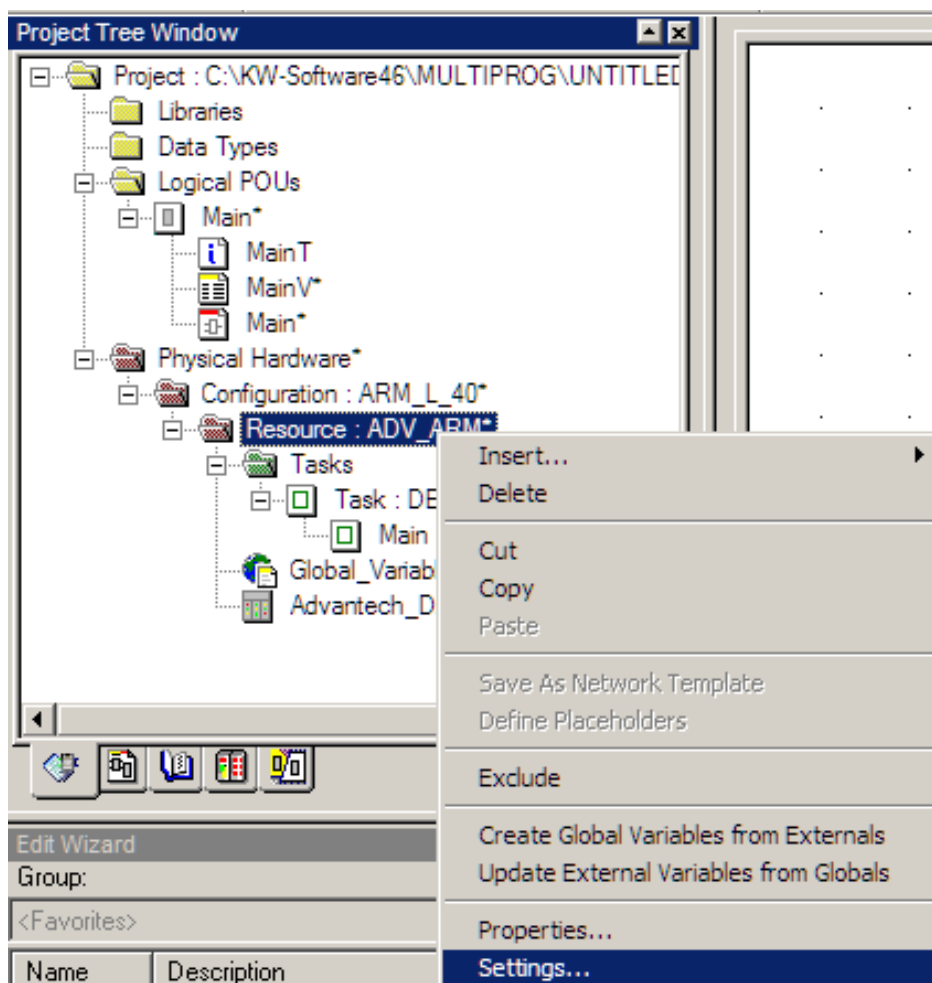
4. Type in the IP address of the APAX-5520 as shown below. If this is disabled then you must license the MULTIPROG software.
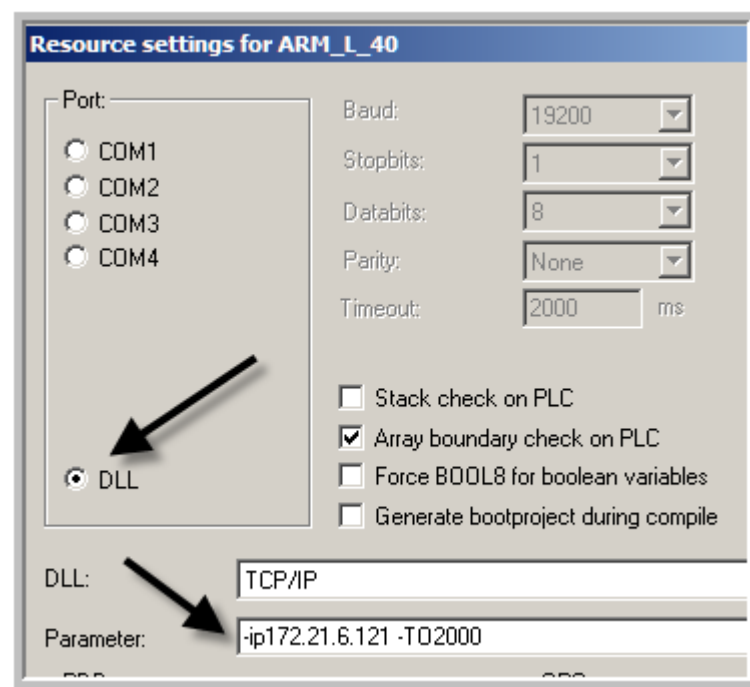
5. Select the "Untitled" POU worksheet from the project tree by double click. Then click into the workspace to the right.
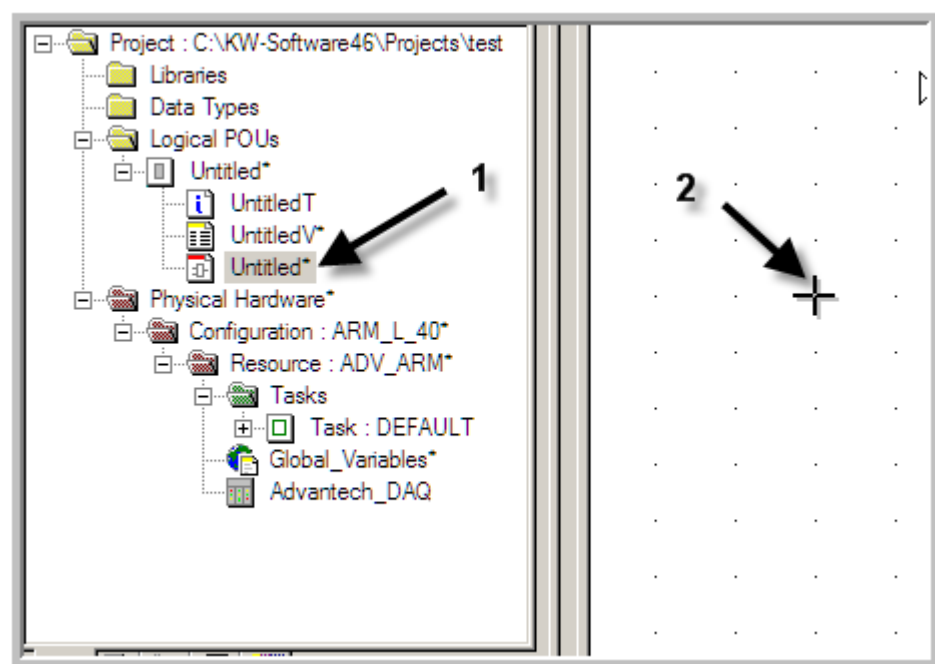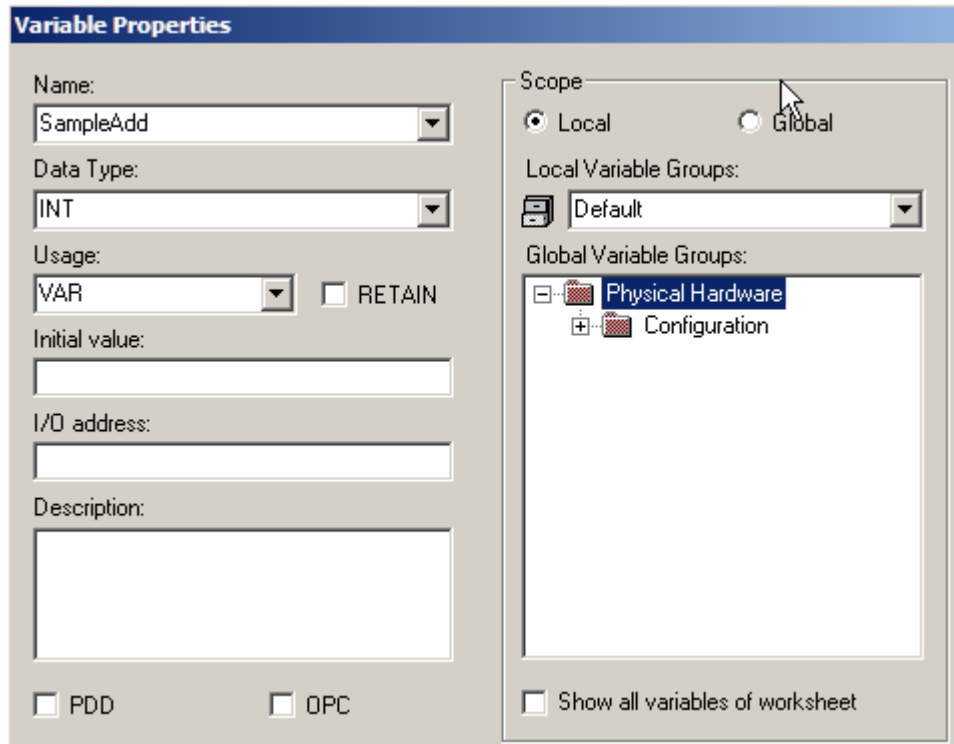
6. When you click into the workspace, it will enable the Edit Wizard available. If the Edit wizard is not shown, then you can enable it from the view menu. Double click the Add function and it will place this function in the workspace.
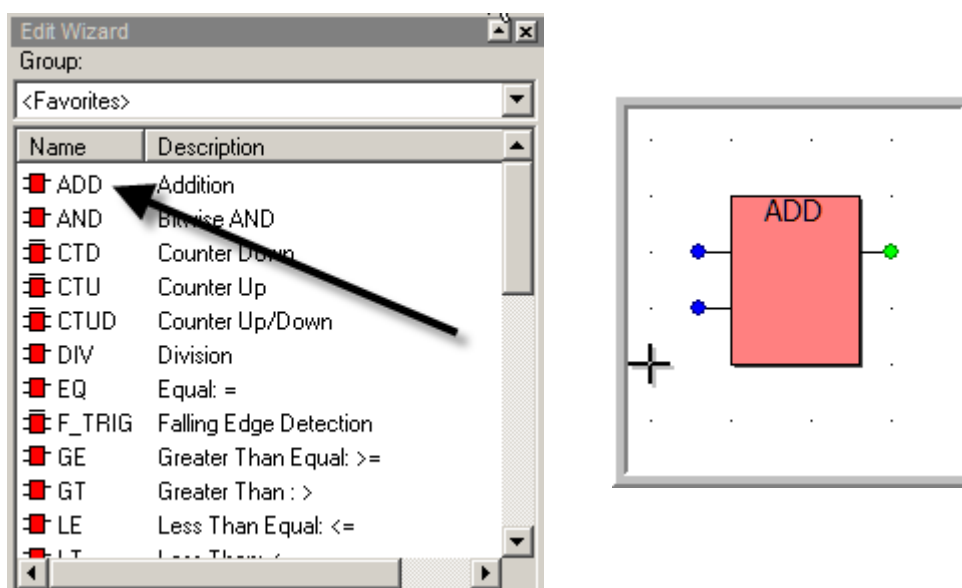
7. Create a variable that will demonstrate the add function in use. Right click in the program workspace and select "Variable".
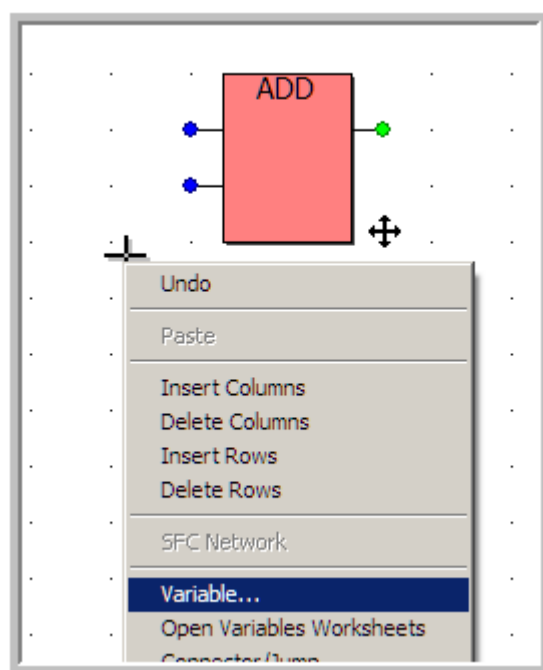
8. Specify a variable called SampleAdd. This variable will be a local integer.

9. Once the variable is created, connect it to the Add function.
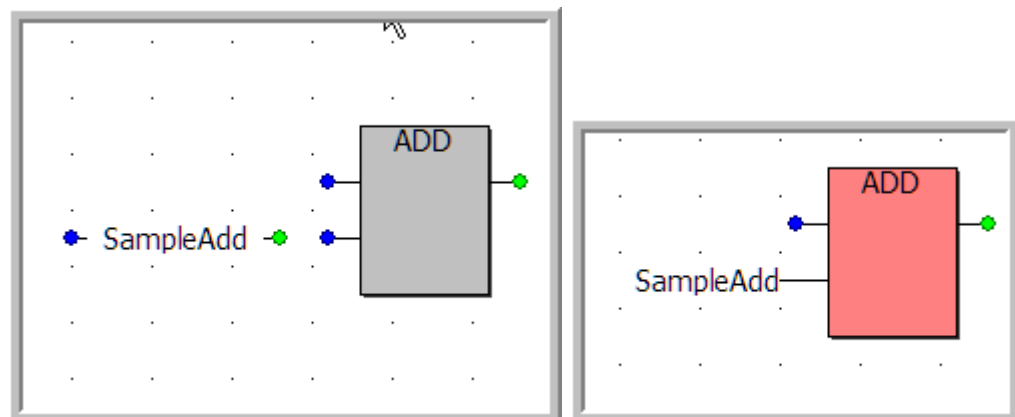
10. Right click "SampleAdd" and select copy. Right click in the workspace and select paste to connect the variable to the output side of the Add function.
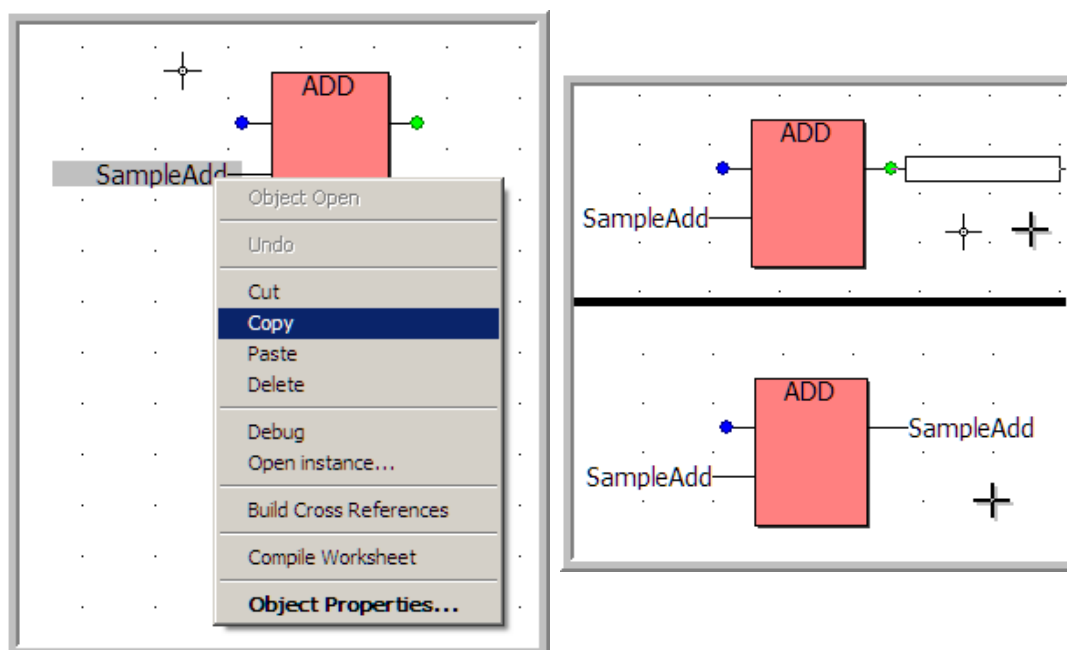
11. Insert the constant value of 1 on the upper input of the Add function. Select the blue input dot on the Add function, right click and select Variable. This will cause 1 to be added to SampleAdd each scan, and put the results into SampleAdd. This will increment the value by 1 each PLC scan.
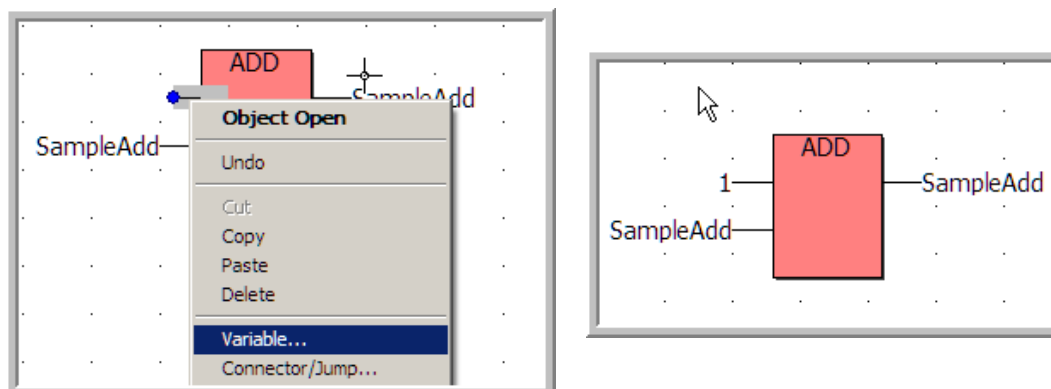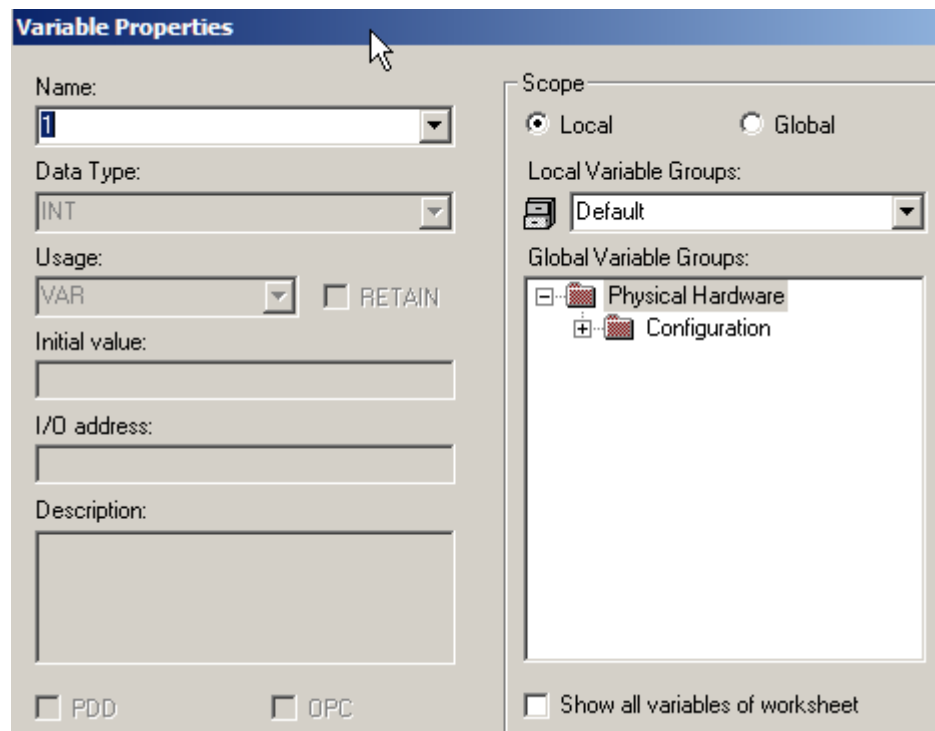
12. Right click the default task in the project tree and select Delete. This will remove the default task as described in the notes at the beginning of the quick start section.
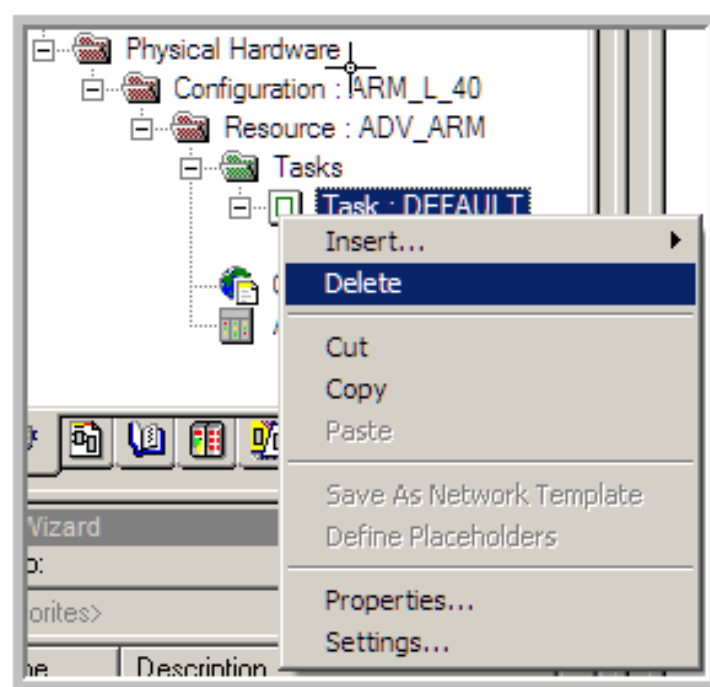
13. Right click "Tasks" and select insert Task. Name the task "Main" and make it "Cyclic".
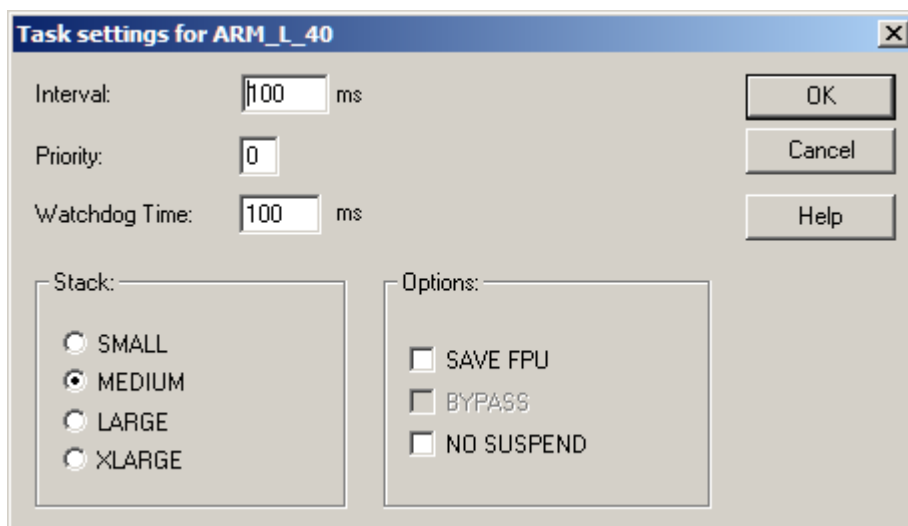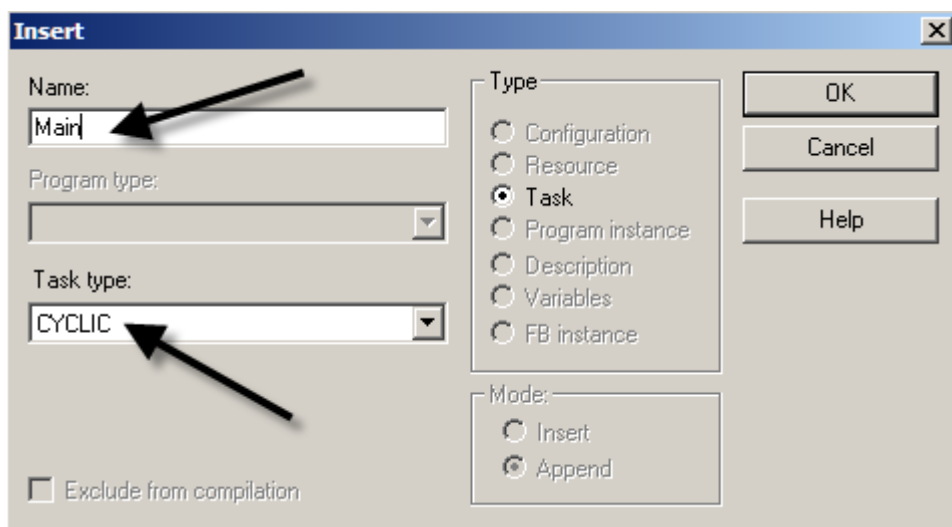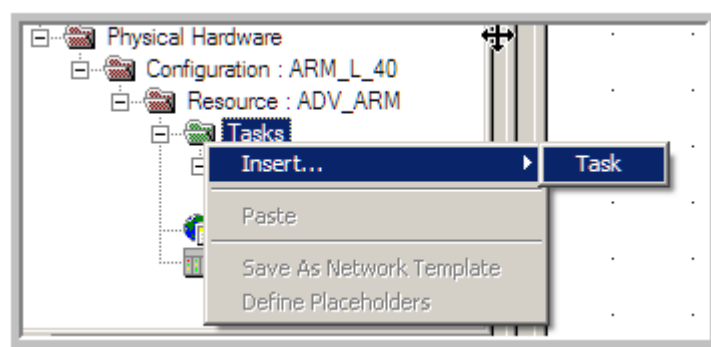
14. Now that you have a cyclic task that will run every 100ms, you need to assign code that will run under that task. Right click the task and select "Insert | Program Instance". Give the program instance a name and select the OK button.
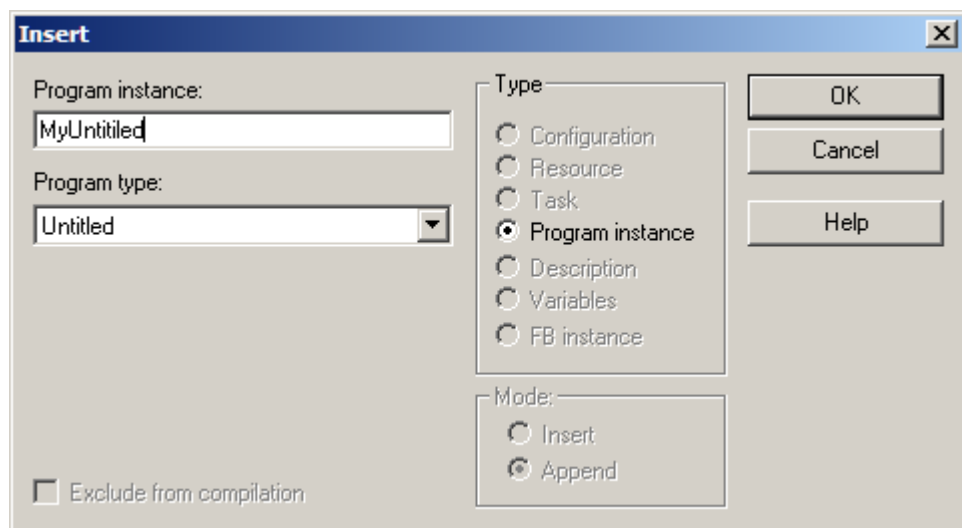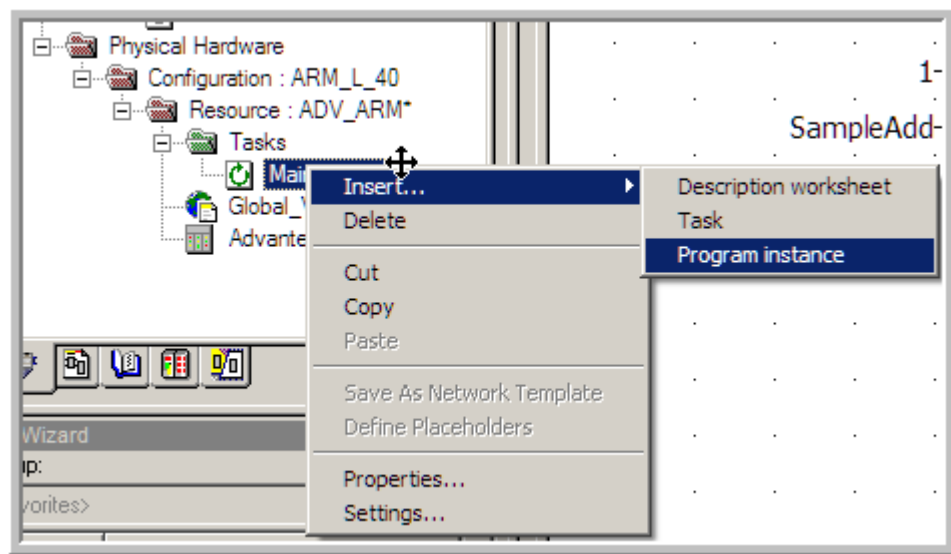
15. Build the project. Select "Build | Make" from the menu. Verify there are no errors in the message window.
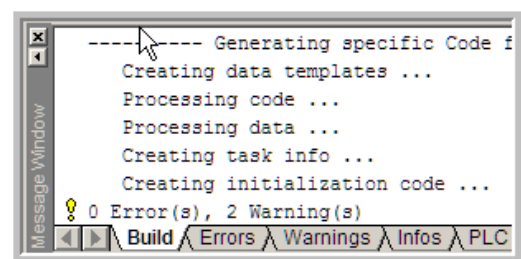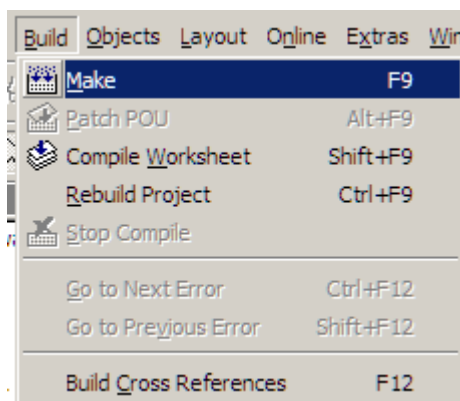
16. Select the Project Control Dialog from the menu. If there is a connection established, you will be able to download the project. If there is not a connection established, you must resolve this problem before continuing.
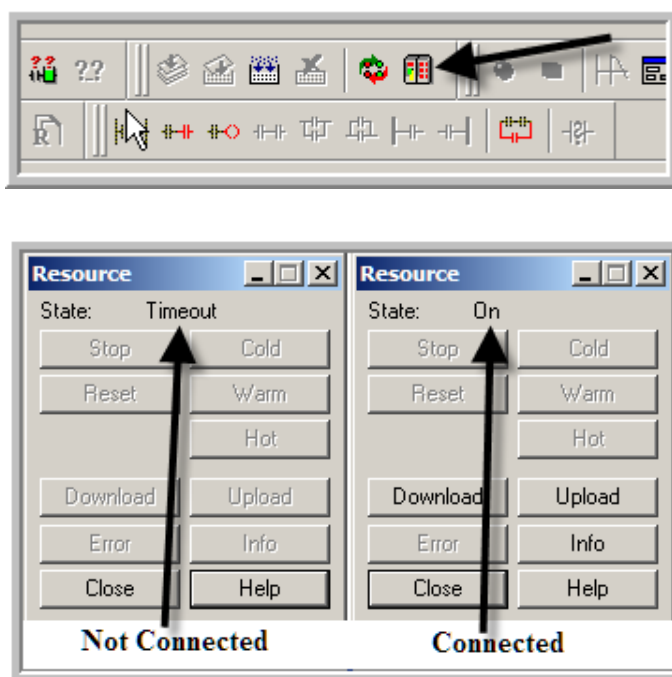
If not connected, check the following:
- IP address set in the project matches the target system.
- ProConOS running on target system.
- Target system and Development system are on the same network.
- Check network cables.
- Ping target system from development system.
- Verify OS Firewall allows outgoing connection for Multiprog.

17. Download project by selecting download from the Control Dialog shown above.
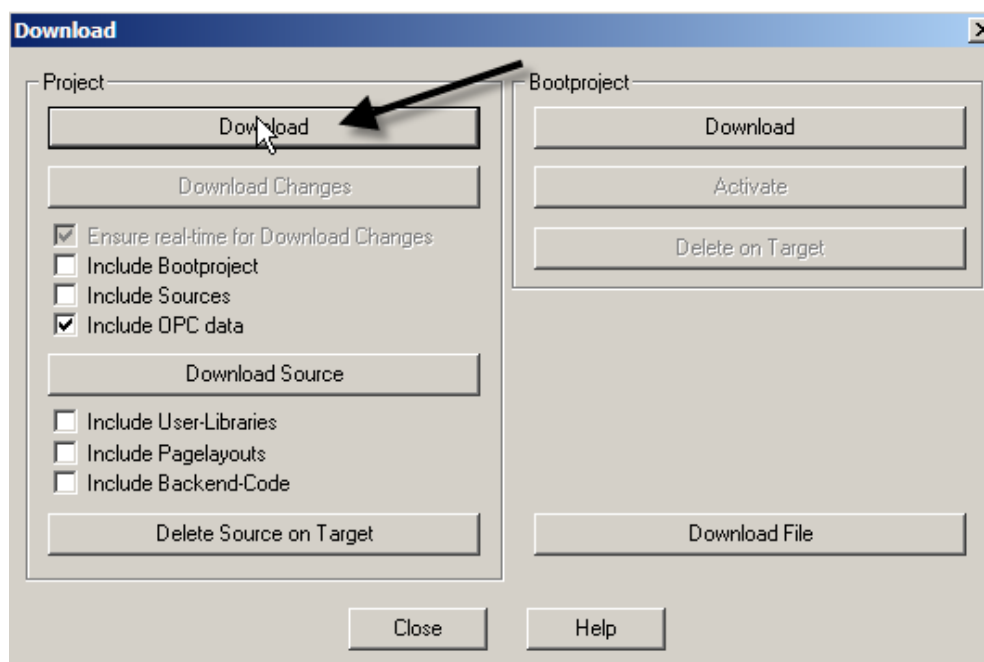
18. Run the project by selecting "Cold" which will trigger a cold start of the system. Cold start re-initializes all variables.
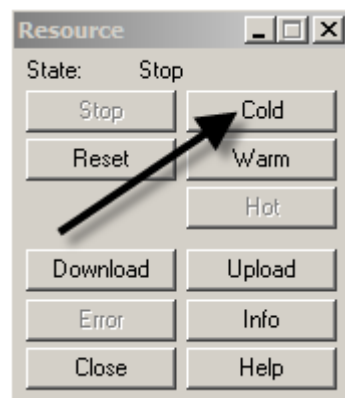
19. Select the online Icon which will allow live debugging of the program. You will now see data updating in the project workspace.
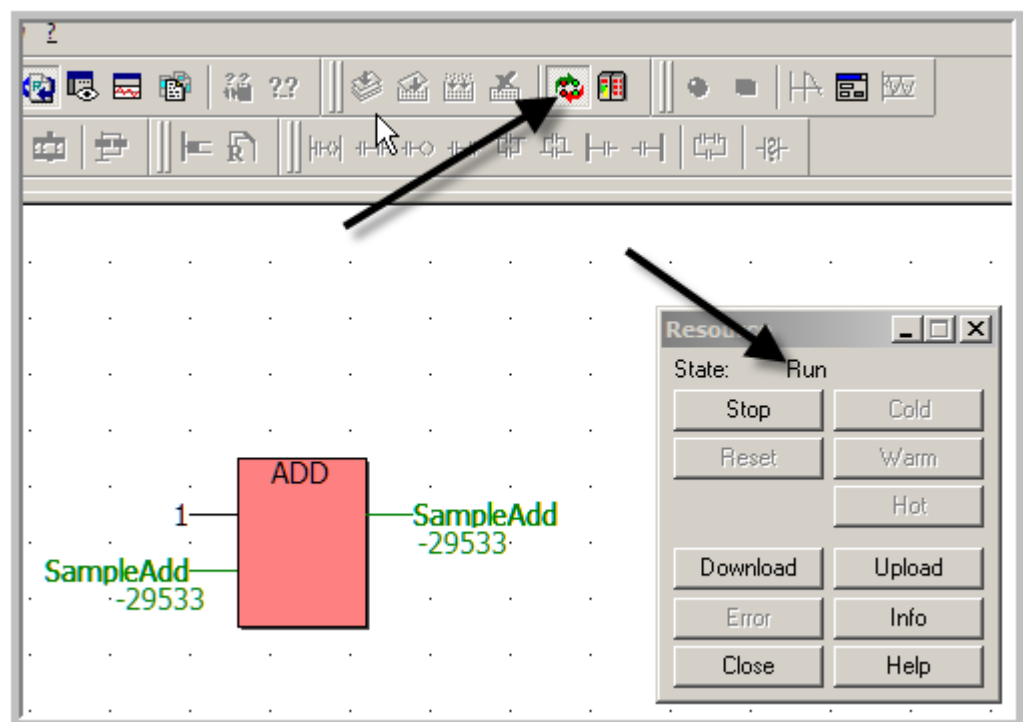
3.2 MULTIPROG Advantech Driver Interface

Please read this section carefully as it may save you time and effort when developing your project.

When you set up your communication driver interface, you will be adding driver entries to the driver I/O configuration. Each Driver entry will set up the driver for one input or output module, or one input or output group for distributed I/O. Each driver entry will also be the connection for the variables (tags) for that I/O entry. There will be one driver entry for each I/O module or distributed I/O group. This section will describe each type of I/O group and explain the relationship with the variables.

3.2.1 Digital Input

Select the digital input board. If you double click, it will append to the end of the list. Once added to the list you must configure the board.
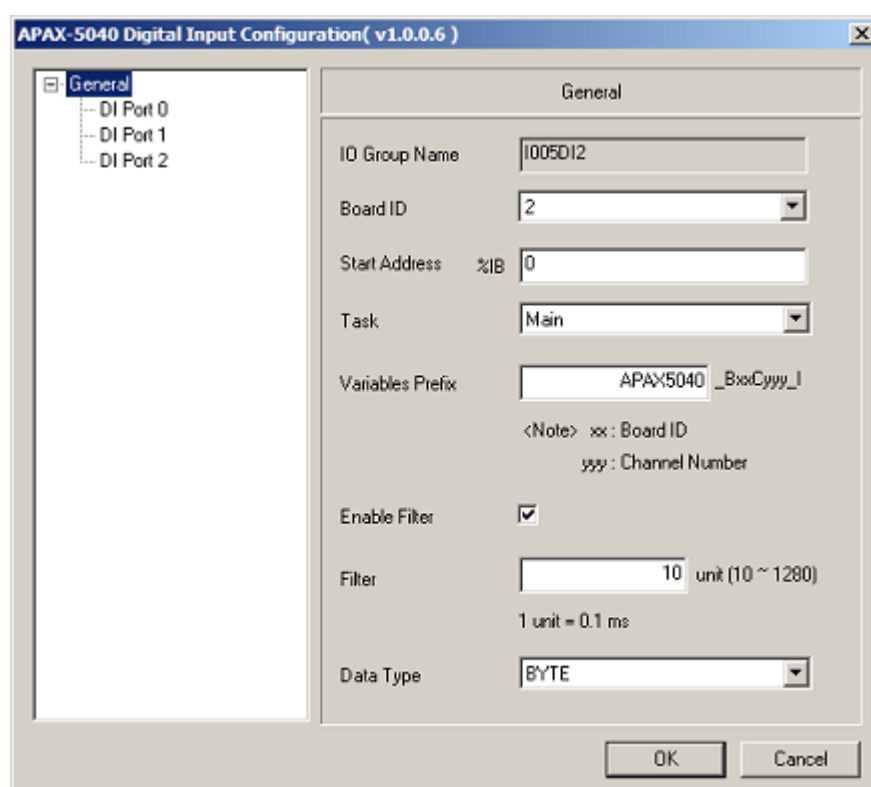
IO Group Name: Each I/O group has a unique name and this cannot be changed by the user.

Board ID: This is the ID set on the front of the APAX-5520 module.

Start Address %IB: This is the starting physical address of the variables for this I/O group. The board shown above has 24 Digital inputs. This will require either 24 Boolean addresses or 3 Byte addresses. The dialog will automatically suggest the next available address.

Task: This is the task that this I/O group will be controlled by. Each time this task runs it will read these inputs first before performing other items controlled by this task.

Variables Prefix: When the variables are auto created, they will have this text prior to the Board ID, channel number and type of variable.

Enable Filter: This enables the high frequency change of state filter for the digital input signal.

Filter: This defines the minimum acceptable signal width time. This will be the same setting for both low pass and high pass filter settings. The signal must remain for this length of time for a change of state to occur.

Data Type: This is the data type that will be created when the variables are created. If BOOL is selected, there will be 24 BOOLS created and addressed. If BYTE is selected, there will be three BYTE'S created and addressed.

Port: Each port is made up of 8 DI's. If reverse logic is needed for an input, you can individually control this by selecting the port and checking the box for each DI that you want inverted in the KW Logic.

3.2.2 Analog Input

IO Group Name: Each I/O group has a unique name and this cannot be changed by the user.

Board ID: This is the ID set on the front of the APAX-5520 module.

Start Address %IB: This is the starting physical address of the variables for this I/O group. The dialog will automatically suggest the next available address. The size of the group will depend on how the channels are configured. See channel configuration below for more information.

Task: This is the task that this I/O group will be controlled by. Each time this task runs it will read these inputs first before performing other items controlled by this task.

Variables Prefix: When the variables are auto created, they will have this text prior to the Board ID, channel number and type of variable.

Channel Configuration

Each Analog Input channel can be individually configured to a different range and return type. If all the channels will be configured the same, click the Apply to All button to configure all the channels to the currently selected channel.

Range: The range selections will be different depending on what Analog Input board is selected.

Return type: If Raw Data is selected, the return type for that channel will be type WORD which is 16 bit in size.

Note!   *Even though the Raw Data selection is 16 Bit in size, the automatic Variable declaration will reserve 32 bits for each AI point in case the user wants to make a change to Eng. Unit at some future point. This will keep the user from having to re-align mapped data.*

If you want a pre-scaled return value then select Eng. Unit and provide the minimum and maximum values to scale the return value. Eng. Unit return type for that channel will by type REAL which is 32 bit in size.

3.2.3 Modbus TCP Input

IO Group Name: Each I/O group has a unique name and this cannot be changed by the user.

Modbus Command: Four different functions are available for input. Two will read inputs and two will read outputs.

Slave ID: Slave ID of Modbus Module. This is generally "1" but sometimes can be something other than "1". One example would be when using a MODBUS/TCP to RTU gateway.

Slave IP: IP address of the slave module.

Data Type: This is the data type of what is being read by the driver. Coil status will only offer BOOL and BYTE. Register status will provide several types that must be decided by the user.

Modbus Start Address: Starting address of I/O to be read on Modbus device. Consult your hardware manual for this address.
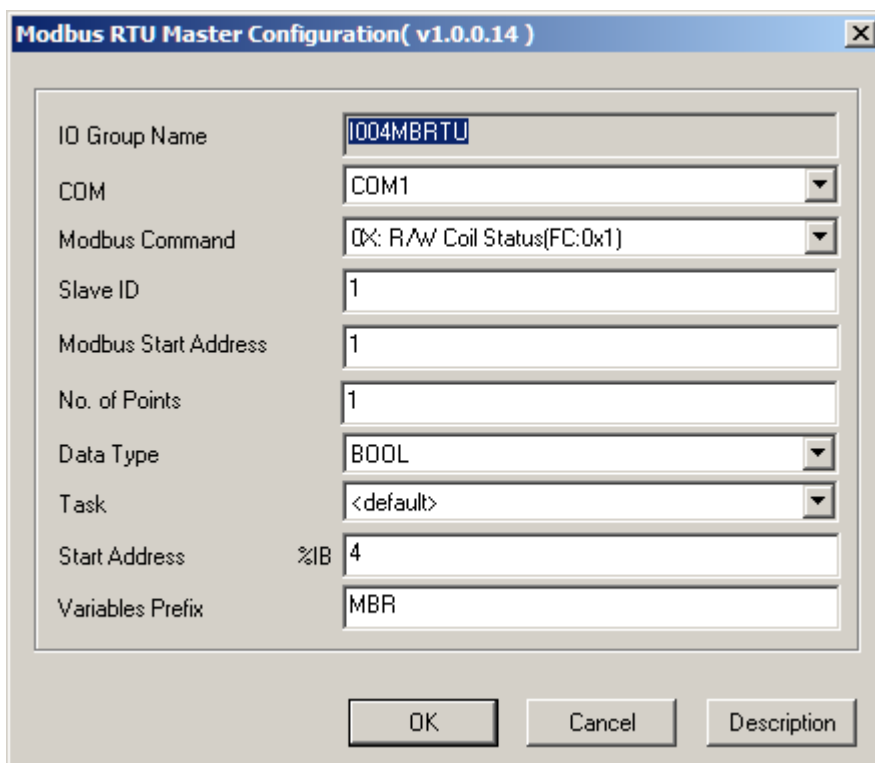
No. of Points: Number of Modbus points being read by this I/O group.

Task: This is the task that this I/O group will be controlled by. Each time this task runs it will read these inputs first before performing other items controlled by this task.
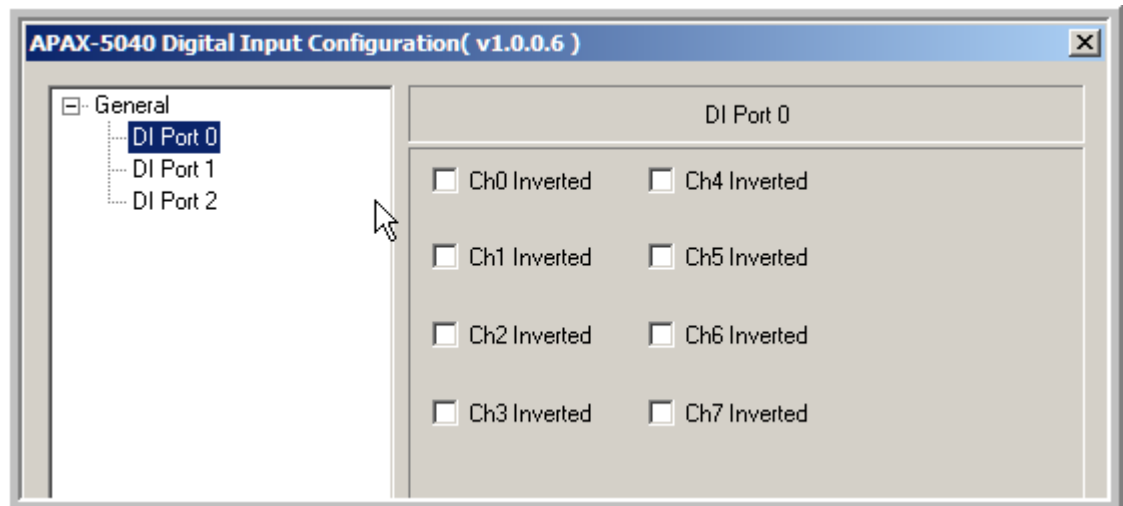
Start Address %IB: This is the starting physical address of the variables for this I/O group in the KW software. The size of the group will depend on the Data type and number of Points. The dialog will automatically suggest the next available address.

Variables Prefix: When the variables are auto created, they will have this text prior to the rest of the variable name.

3.2.4 Modbus/RTU Input

IO Group Name: Each I/O group has a unique name and this cannot be changed by the user.

COM: COM port for Modbus RTU.

Modbus Command: Four different functions are available for input. Two will read inputs and two will read outputs.

Slave ID: Slave ID of Modbus Module

Modbus Start Address: Starting address of I/O to be read on Modbus device. Consult your hardware manual for this address.

No. of Points: Number of Modbus points being read by this I/O group.

Data Type: This is the data type of what is being read by the driver. Coil status will only offer BOOL and BYTE. Register status will provide several types that must be decided by the user.

Task: This is the task that this I/O group will be controlled by. Each time this task runs it will read these inputs first before performing other items controlled by this task.

Start Address %IB: This is the starting physical address of the variables for this I/O group. The size of the group will depend on the Data type and number of Points. The dialog will automatically suggest the next available address.

Variables Prefix: When the variables are auto created, they will have this text prior to the rest of the variable name.

3.2.5 Digital Output

IO Group Name: Each I/O group has a unique name and this cannot be changed by the user.

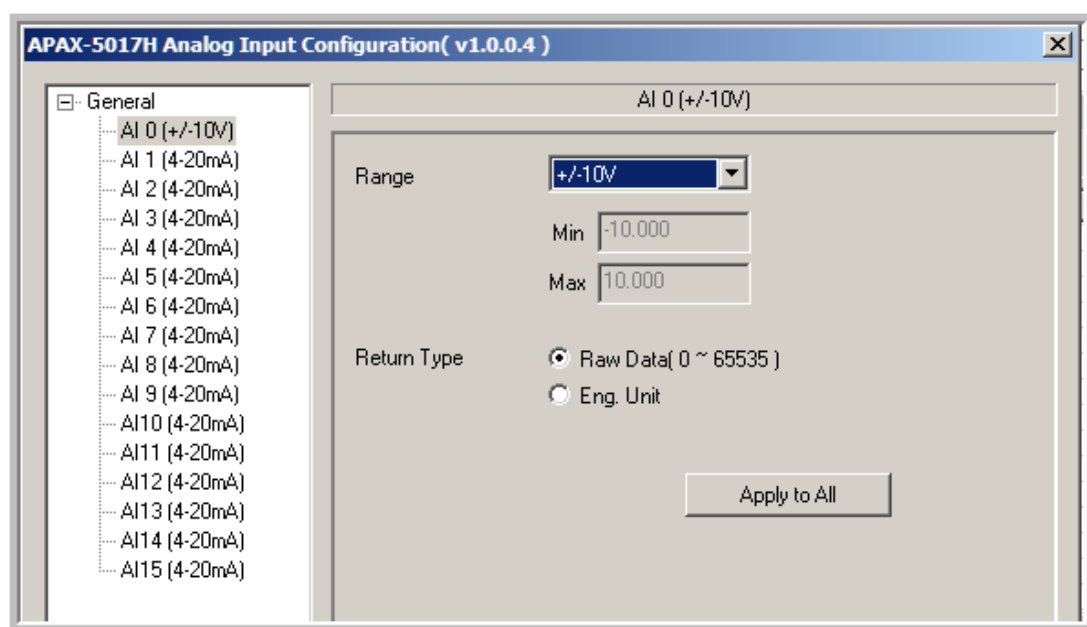**Board ID**: This is the ID set on the front of the APAX-5520 I/O module.

Start Address %QB: This is the starting physical address of the variables for this I/O group. The board shown above has 24 Digital outputs. This will require either 24 Boolean addresses or 3 Byte addresses. The dialog will automatically suggest the next available address.

Task: This is the task that this I/O group will be controlled by. Each time this task runs it will write these outputs first after performing other items controlled by this task.

Variables Prefix: When the variables are auto created, they will have this text prior to the Board ID, channel number and type of variable.

Data Type: This is the data type that will be created when the variables are created. If BOOL is selected, there will be 24 BOOLS created and addressed. If BYTE is selected, there will be three BYTE'S created and addressed.

3.2.6 Analog Output

IO Group Name: Each I/O group has a unique name and this cannot be changed by the user.

Board ID: This is the ID set on the front of the APAX-5520 module.

Start Address %QB: This is the starting physical address of the variables for this I/O group. The dialog will automatically suggest the next available address. The size of the group will depend on how the channels are configured. See channel configuration below for more information.

Task: This is the task that this I/O group will be controlled by. Each time this task runs it will write these outputs after performing other items controlled by this task.

Variables Prefix: When the variables are auto created, they will have this text prior to the Board ID, channel number and type of variable.

Channel Configuration

Each Analog Input channel can be individually configured to a different range and return type. If all the channels will be configured the same, click the Apply to All button to configure all the channels to the currently selected channel.

Range: The range selections will be different depending on what Analog Output board is selected.

Return type: If Raw Data is selected, the output type for that channel will be type WORD which is 16 bit in size.

NOTE: Even though the Raw Data selection is 16 Bit in size, the automatic Variable declaration will reserve 32 bits for each AO point in case the user wants to make a change to Eng. Unit at some future point. This will keep the user from having to re-align mapped data.

If you want a pre-scaled return value then select Eng. Unit and provide the minimum and maximum values to scale the return value. Eng. Unit return type for that channel will by type REAL which is 32 bit in size.

3.2.7 Modbus TCP Output

IO Group Name: Each I/O group has a unique name and this cannot be changed by the user.

Modbus Command: Two functions are available for output. Write coil or write register.

Slave ID: Slave ID of Modbus Module. This is generally "1" but sometimes can be something other than "1". One example would be when using a MODBUS/TCP to RTU gateway where each module has a different Slave ID.

Slave IP: IP address of the slave module.

Data Type: This is the data type of what is being written by the driver. Coil write will only offer BOOL and BYTE. Register write will provide several types depending on the register.

Modbus Start Address: Starting address of I/O to be written on Modbus device. Consult your hardware manual for this address.

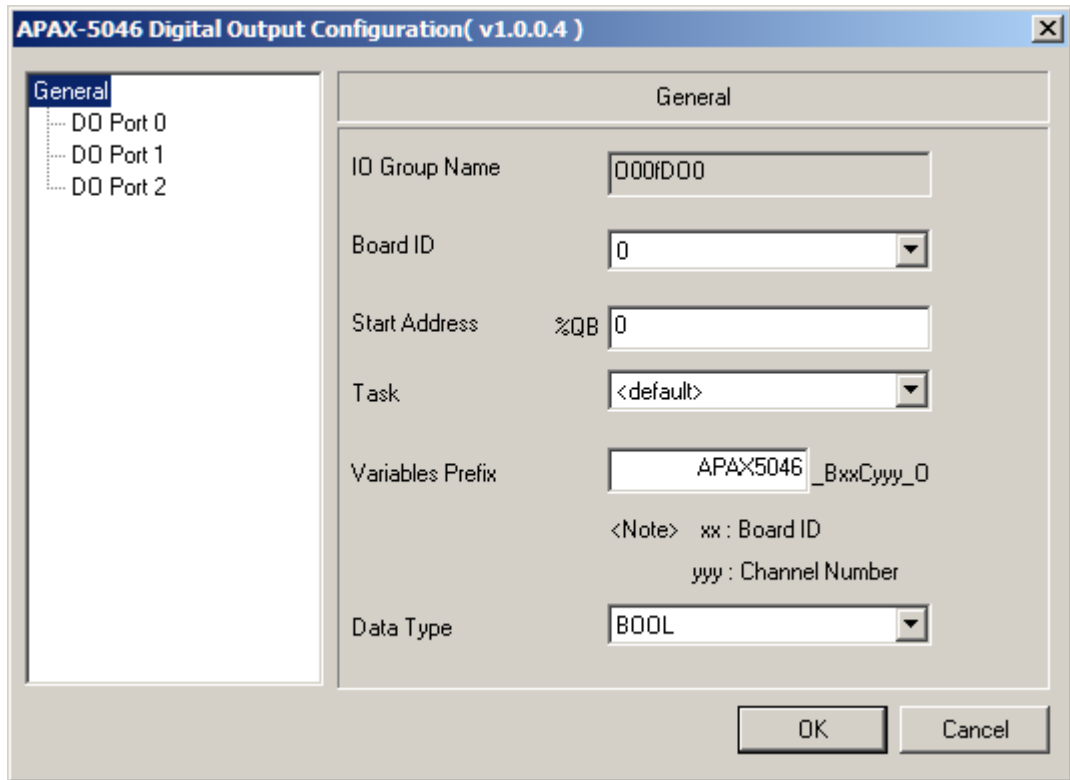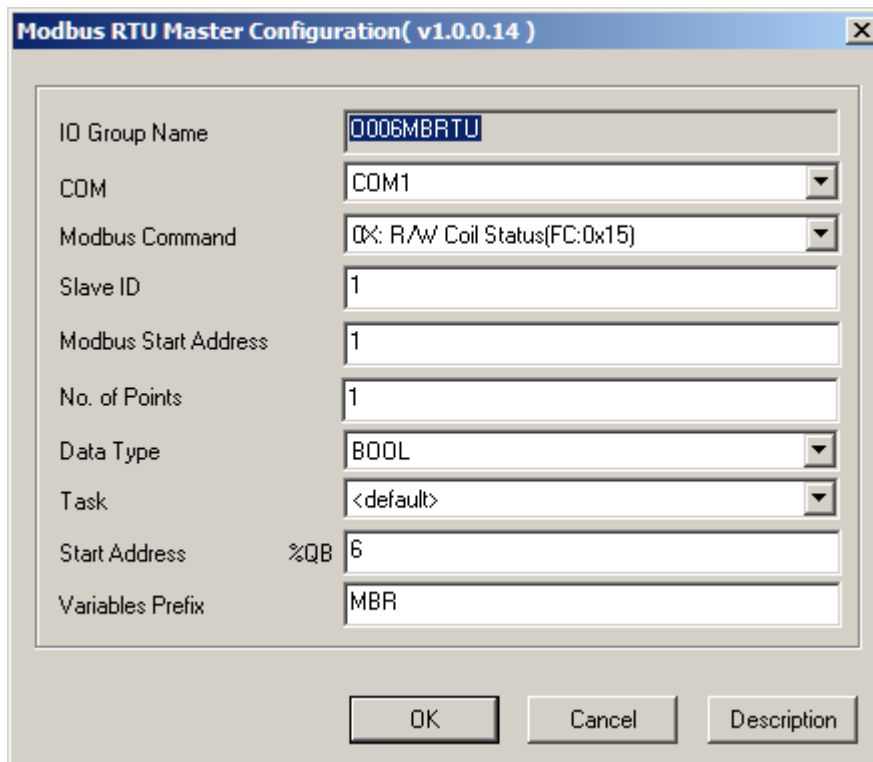**No. of Points:** Number of Modbus points being written by this I/O group.

Task: This is the task that this I/O group will be controlled by. Each time this task runs it will write these outputs after performing other items controlled by this task.

Start Address %QB: This is the starting physical address of the variables for this I/O group in the KW software. The size of the group will depend on the Data type and number of Points. The dialog will automatically suggest the next available address.

Variables Prefix: When the variables are auto created, they will have this text prior to the rest of the variable name.

3.2.8 Modbus/RTU Output

IO Group Name: Each I/O group has a unique name and this cannot be changed by the user.

COM: COM port for Modbus RTU.

Modbus Command: 2 functions are available for output. Write coils and write registers.

Slave ID: Slave ID of Modbus Module

Modbus Start Address: Starting address of I/O to be written on Modbus device. Consult your hardware manual for this address.

No. of Points: Number of Modbus points being written by this I/O group.

Data Type: This is the data type of what is being written by the driver. Coil write will only offer BOOL and BYTE. Register write will provide several types.

Task: This is the task that this I/O group will be controlled by. Each time this task runs it will write these outputs after performing other items controlled by this task.

Start Address %QB: This is the starting physical address of the variables for this I/O group. The size of the group will depend on the Data type and number of Points. The dialog will automatically suggest the next available address.

Variables Prefix: When the variables are auto created, they will have this text prior to the rest of the variable name.

3.2.9 Modbus/TCP Client General Settings
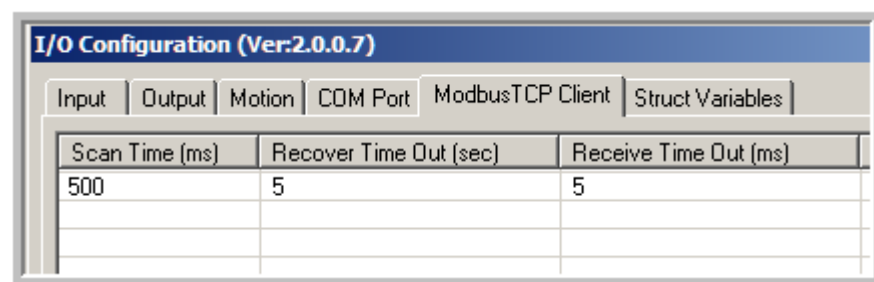
Scan Time: The execution of the Modbus TCP communication happens in its own thread. This thread will send to all slaves and wait for all responses. The scan time is the interval between start of current thread execution and the start of the next thread execution.

Recover Time Out: Interval between reconnect attempts for a disconnected slave.

Receive Time Out: This value can only be multiples of 5. The maximum number of retries are "Receive time out"/5. If "Receive Time Out" is 15 then The maximum number of retries are 3. Once the maximum retries are reached, then the slave goes to a disconnected state and is handled by the recovery timeout.

3.2.10 Variables and Auto Declaration

Once the I/O configuration is set, Advantech has added a way to automatically declare variables. This gives the developer an easy method and good starting point for declaring variables. There is a button on the I/O configuration to do this called "Create All Input Variables". There is also one for the Output variables. This button will automatically create and map the variables associated with configured I/O. The Create All Input Variables button will create a variable based on the following example:

Example Variable: APAX5040_B02C014_I

APAX5040 = Name of I/O Module

B02 = Board ID 2

C014 = Channel 14, in this case it would be input 14 starting from zero.

I = Input

In this example, we have configured two APAX5040 modules.

APAX5040, Board ID 2, Data Type BOOL, Start Address %IX0.0

APAX5040, Board ID 5, Data Type BYTE, Start Address %IB4This will create two Global Variable groups, one group for each I/O configuration. Since we have created two of the same board with different DATA TYPE, you can see that the variables are mapped differently. The first group has been mapped as individual bits. The second group has been mapped as a packed byte.

The variable names are automatically created, but these names can be changed to whatever is more relevant to the programmer. From the example above there have been some changes as shown in the following example.

Name	Type	Usage	Description	Address
⊟ In_APAX5040_2_10				
Emergency_Stop_PB	BOOL	VAR_GLOBAL	[Board ID2] APAX5040 DI channel 0	%IX0.0
OverTravelLimitSwitch	BOOL	VAR_GLOBAL	[Board ID2] APAX5040 DI channel 1	%IX0.1
AutomaticModePB	BOOL	VAR_GLOBAL	[Board ID2] APAX5040 DI channel 2	%IX0.2
ManualModePB	BOOL	VAR_GLOBAL	[Board ID2] APAX5040 DI channel 3	%IX0.3

Important note: if the programmer decides to make changes to the driver such as "Start Address" or "Board ID", the driver interface will automatically remove the mapped Variable group from being deleted, associated with it. Before making name changes, make sure you have your I/O entry set up properly. If you would like to make changes to the Start address or Board ID, but save your Variable Group, you must change the name of the variable group first. Once this is done, you must also change the mapping of the saved variable group to match the updated variable group settings.

Name	Type	Usage	Description	Address
⊟ MySavedGroup				
Emergency_Stop_PB	BOOL	VAR_GLOBAL	[Board ID2] APAX5040 DI channel 0	%IX0.0
OverTravelLimitSwitch	BOOL	VAR_GLOBAL	[Board ID2] APAX5040 DI channel 1	%IX0.1
AutomaticModePB	BOOL	VAR_GLOBAL	[Board ID2] APAX5040 DI channel 2	%IX0.2
ManualModePB	BOOL	VAR_GLOBAL	[Board ID2] APAX5040 DI channel 3	%IX0.3

3.2.11 Direct Mapping Examples

Example Mapping: %IW7

% = Directly Mapped variable

I = Physical Input

W = Word Size

7 = Starting at Byte 7 and ending AFTER byte 8. The next available address would be 9.

Example Mapping: %QX4.6

% = Directly Mapped variable

Q = Physical Input

X = Bit Size

4.6 = BYTE 4 Bit 6. Since Bytes are 8 bits, the bits are 0-7

Location prefix	Description
I	Physical input
Q	Physical output
M	Physical address in the PLC memory

Size prefix	Description
X	Single bit size (only with data type BOOL)
None	Single bit size
B	Byte size (8 bits)
W	Word size (16 bits)
D	Double word size (32 bits)
L	Long word size (64 bits)

3.2.12 Modbus Slave Operation (Server)

Advantech has provided an interface to monitor and control tags in the APAX-5520. This interface is accessible via Modbus/TCP as well as Modbus/RTU The APAX-5520 can be treated as a MODBUS Slave. There are four different memory sections offered to the user. Three of them require no configuration. The fourth section is user configurable. These sections include the following:

Inputs - SCADA/HMI software can directly access the Input points of the APAX-5520 for monitoring.

Outputs - SCADA/HMI software can directly access the Output points of the APAX-5520 for read only access. Direct control of the APAX-5520 outputs is not allowed. Since the Output control is done via the controller software, direct control of Outputs could cause problems as well as hidden programming problems and is therefore restricted.

Shared Memory Area - SCADA/HMI software can directly access user configured memory mapped variables of the APAX-5520 for Read/Write. The user configures this area within the KW software and then decides on usage and access of the area.

Special Function Area – Reserved for future use.

Modbus/RTU Setup

Before the user can use the Modbus/RTU server, the Com Port must be setup. This is done via the I/O Configuration on the COM port tab. The port settings are fixed at the following:

Baud rate=57600, Data bits=8, Stop bits=1, Parity=no parity, Protocol=Modbus.

Once the port is set up as slave, it cannot be used as Master through the driver setup section. This can cause errors if one port is set to both master and slave.
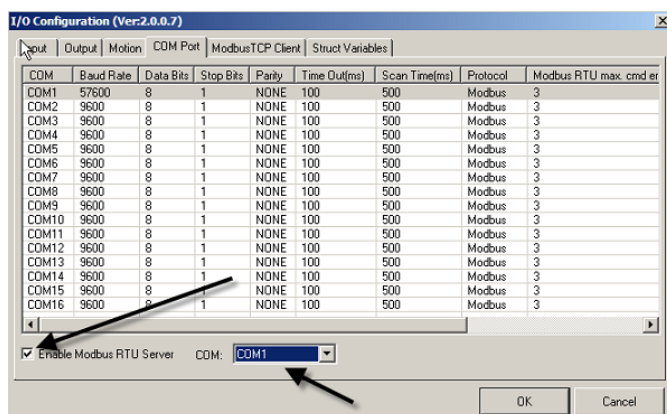
Inputs

In the mapped input area, the first 1000 "Mapped" input WORDS (%IB0 - %IB1999) OR the first 1000 BOOLS (%IX0.0- %IX124.7) are available via Modbus as READ ONLY. All input variables that are mapped to this area are readable via standard MODBUS functions.

Inputs can be read via 1x function or 3x function depending on user needs. It is possible to mix the data types. This area stores:

MODBUS Input Registers 3x0001 ~ 3x1000 (This function returns word values)

MODBUS Input Status 1x0001 ~ 1x1000 (This function returns bit values)

Examples of Modbus addresses

KW Variable Address	KW Variable Size	Modbus command	Description
%IW0	Word – 16 bit	3x0001 1 data point	Read Input bytes 0-1
%IW4	Word – 16bit	3x0003 1 data point	Read Input bytes 4-5
%IW2-9	4 Words	3x0002 4 data points	Read Input bytes 2-9
%IX0.0	BOOL – 1bit	1x0001 1 data point	Read Input bit 0.0
%IX1.1	BOOL – 1bit	1x0010 1 data point	Read Input bit 1.1
%IX1.0 - %IX3.7	24 BOOLS -24bits	1x0009 24 data points	Read Input bits 1.0 – 3.7

KW BOOL Inputs and MODBUS 1x Function Mapping Alignment							
%IB0							
%IX0.0	%IX0.1	%IX0.2	%IX0.3	%IX0.4	%IX0.5	%IX0.6	%IX0.7
1x0001	1x0002	1x0003	1x0004	1x0005	1x0006	1x0007	1x0008
%IB1							
%IX1.0	%IX1.1	%IX1.2	%IX1.3	%IX1.4	%IX1.5	%IX1.6	%IX1.7
1x0009	1x0010	1x00011	1x0012	1x0013	1x0014	1x0015	1x0016

KW WORD Inputs and MODBUS 3x Function Mapping Alignment									
%IB0	%IB1	%IB2	%IB3	%IB4	%IB5	%IB6	%IB7	%IB8	%IB9
%IW0		%IW2		%IW4		%IW6		%IW8	
3x0001		3x0002		3x0003		3x0004		3x0005	
%IB10	%IB11	%IB12	%IB13	%IB14	%IB15	%IB16	%IB17	%IB18	%IB19
%IW10		%IW12		%IW14		%IW16		%IW18	
3x0006		3x0007		3x0008		3x0009		3x0010	

Note! *The most efficient way to use MODBUS communication is to pack as many of the same kinds of data together, and read as many as possible in one communication command.*

Outputs

In the mapped Output area, the first 2000 "Mapped" Output bytes (%QB0 - %IQB1999) OR the first 1000 BOOLS (%QX0.0- %QX124.7) are available via MOD-BUS as READ ONLY. All Output variables that are mapped to this area are readable via standard MODBUS functions.

Outputs can be read via 0x function or 4x function depending on user needs. It is possible to mix the data types. This area stores:

MODBUS Input Registers 4x0001 ~ 4x1000 (This function returns word values)

MODBUS Input Status 0x0001 ~ 0x1000 (This function returns bit values)

Examples of MODBUS addresses

KW Variable Address	KW Variable Size	MODBUS command	Description
%QW0	Word – 16 bit	4x0001 1 data point	Read Output bytes 0-1
%QW4	Word – 16bit	4x0003 1 data point	Read Output bytes 4-5
%QW2-9	4 Words	4x0002 4 data points	Read Output bytes 2-9
%QX0.0	BOOL – 1bit	0x0001 1 data point	Read Output bit 0.0
%QX1.1	BOOL – 1bit	0x0010 1 data point	Read Output bit 1.1
%QX1.0 - %QX3.7	24 BOOLS -24bits	0x0009 24 data points	Read Output bits 1.0 – 3.7

KW BOOL Outputs and MODBUS 0x Function Mapping Alignment							
%QB0							
%QX0.0	%QX0.1	%QX0.2	%QX0.3	%QX0.4	%QX0.5	%QX0.6	%QX0.7
0x0001	0x0002	0x0003	0x0004	0x0005	0x0006	0x0007	0x0008
%QB1							
%QX1.0	%QX1.1	%QX1.2	%QX1.3	%QX1.4	%QX1.5	%QX1.6	%QX1.7
0x0009	0x0010	0x00011	0x0012	0x0013	0x0014	0x0015	0x0016

KW WORD Outputs and MODBUS 4x Function Mapping Alignment									
%QB0	%QB1	%QB2	%QB3	%QB4	%QB5	%QB6	%QB7	%QB8	%QB9
%QW0		%QW2		%QW4		%QW6		%QW8	
4x0001		4x0002		4x0003		4x0004		4x0005	
%QB10	%QB11	%QB12	%QB13	%QB14	%QB15	%QB16	%QB17	%QB18	%QB19
%QW10		%QW12		%QW14		%QW16		%QW18	
4x0006		4x0007		4x0008		4x0009		4x0010	

Shared Memory Area

The APAX-5520 Controller reserves approximately 128K Bytes memory space for general Modbus use. This shared memory block can store user's data and exchange the data through Modbus TCP and RTU protocol with a SCADA/HMI. The units in Modbus 4X registers are Word size (16 bit), so there are a total of 64K Words available. The memory can be accessed as read or write with four different Modbus functions as shown below:

MODBUS Coil Status 0x2001~0x65536

MODBUS Input Status 1x2001~1x65536

MODBUS Input Registers 3x2001~3x65536

MODBUS Holding Registers 4x2001~4x65536

It is possible to mix the use of this area, using some for read and some for write, as well as mixing of Coil and registers.

Since this area is fully configurable by the developer, it is up to them to keep track of what values are in what position for MODBUS reads and writes.

Since the units of MW3.0 are BYTES, users need to map the I/O Address and Modbus Address as follows.

For Bool data type: (0x and 1x functions)

	I/O Address	MODBUS ADDRESS	Length
Data 1	%MX3.0.0	0x2001	1 Bit
Data 2	%MX3.0.1	0x2002	1 Bit
Data 3	%MX3.0.2	0x2003	1 Bit

For Byte and Word data type: (3x and 4x functions)

	I/O Address	MODBUS ADDRESS	Length
Data 1	%MW3.0	4x2001	2 Bytes
Data 2	%MW3.2	4x2002	2 Bytes
Data 3	%MW3.4	4x2003	2 Bytes

For Dword and Real data type: (3x and 4x functions)

	I/O Address	MODBUS ADDRESS	Length
Data 1	%MD3.0	4x2001 + 4x2002	4 Bytes
Data 2	%MD3.4	4x2003 + 4x2004	4 Bytes
Data 3	%MD3.8	4x2005 + 4x2006	4 Bytes

3.2.13 MULTIPROG and ProConOS Highlights and Tips

To help the end user, Advantech has added this section to show some of the more important features of the software and some items that may not be explicitly covered in the MULTIPROG online help. It is also intended to cover some basics of Pro-ConOS. It should reduce the time needed to come up to speed on using this programming environment. It is not intended to be the only resource, but an additional resource to help the end user get started and understand MULTIPROG. For more information on MULTIPROG, see the online help. For more information on Pro-ConOS, see the ProConOS user's manual.

3.2.14 Project Tree Overview

This section is a quick overview of what each part of the project tree is used for. More detailed explanations are below and in the online help for MULTIPROG.
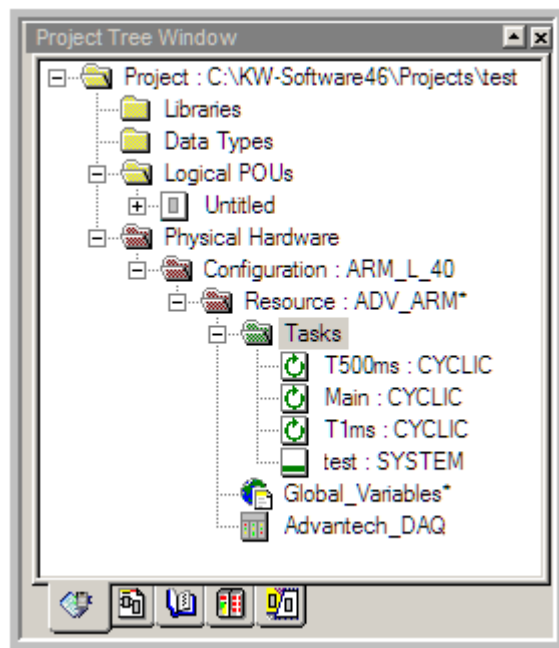
Libraries: This is where you can add libraries to your project. Libraries can include Firmware libraries or User libraries. User libraries can be other Projects. So, the user can develop a project with just a collection of Function blocks, save it as a project and add it as a library at a later time.

Data Types: This section is where user data types are defined. This includes Arrays, Structs and special string types.

Logical POUs: Program **O**rganization **U**nits can be Programs, Functions and Function blocks. Code does not execute from this section. Programs are added to tasks, Functions and Function blocks are added to programs.

Configuration: One project can have multiple Configurations. This means one project can share Libraries, Data Types and Logical POUs for two different hardware platforms.

Resource: One configuration can have multiple Resources. One example is that one project could connect and download to two APAX5520s on the same network.

Tasks: Each cyclic task is a separate thread in WinCE and can execute at its own priority and scan rate. See below for a more detailed description of tasks.

Global_Variables: This is where all global variables are declared. Each POU can have local variables as well as global variables.
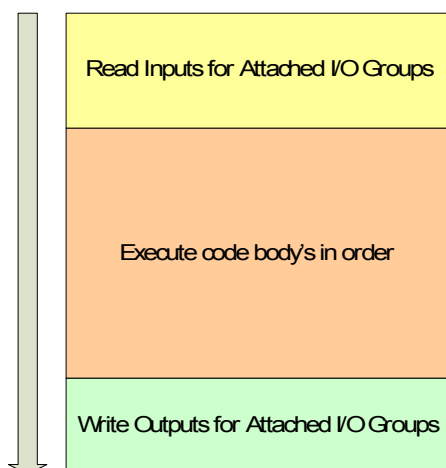
Advantech_DAQ: This is the I/O configuration or hardware driver interface.

3.2.15 Tasks

Tasks are at the heart of ProConOS and deserve some explanation.

Task functions

Tasks have three functions in this order: Read inputs, execute code, write outputs.

Tasks have Input groups, Output groups and Programs attached to them. The tasks must be created before the Programs and I/O groups are created. This allows them to be associated.

There are three types of tasks implemented on the APAX-5520:

Cyclic tasks

Cyclic tasks run on a timed basis set by the developer. The task is set up in the Pro-ConOS Scheduler to start each time the interval is started. Once the task runs, it is suspended until the next time interval.
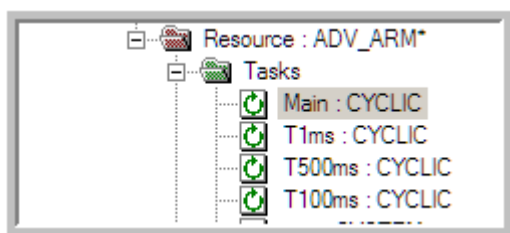
System tasks

System tasks run based on a system event such as "Divide by Zero", "Watchdog error" or "Warm Start". These tasks will run once when the event occurs.

Default task

There is only one Default task available and it runs in the background when CPU time is available. This is considered a background task.

Task implementation

Below is a sample of task implementation.

Main – a cyclic task called Main with a scan rate of 200ms

T1ms – a cyclic task called T1ms with a scan rate of 1ms

T500ms – a cyclic task called T500ms with a scan rate of 500ms

T100ms – a cyclic task called T100ms with a scan rate of 100ms

The order of tasks in the task list is NOT important unless two tasks have the same priority and interval. In that case the first one in the list will be executed first. When adding a cyclic task, you give it an interval time (scan rate), a watchdog time and a priority.

Watchdog time

If the task takes longer than the watchdog time to complete, then the system watch-dog error will occur. The watchdog time should be set to the same as the interval (scan time).

Priority

Tasks can be set at any of 16 priority levels. 0 is the highest priority and 15 is the low-est priority. If a low priority task is running and a higher priority task is scheduled to run, the higher priority task will interrupt the lower priority task to execute. If both tasks are the same priority level, then the second task will wait.

3.2.16 Edit Wizard

The Edit wizard must be used for implementing functions and function blocks. To add a function block in any language, click in the workspace first, then double click the function to be added. Once the function or function block is added, then the user must connect variables to the function or function block. Below are examples of add function in the FBD language and ST language.
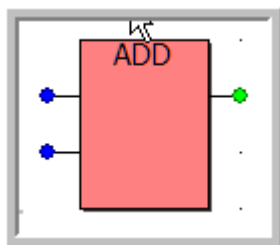
Function Block Diagram. Variables or other functions must be connected to the rails.

Structured Text. Comments must be replaced with variables.

3.2.17 Data Types

When implementing a Data type, it is best to use the Edit Wizard to start the process. Below is an example of an Array:

Once it is defined, you can then implement this data type. It will show up in the list of data types when you declare the variable:
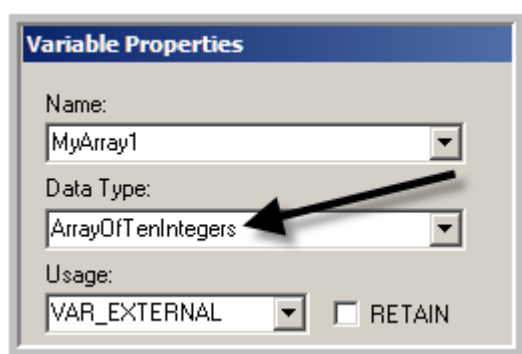
NOTE: to see online values of Arrays or Structs, you must add the implemented Array or Struct to the watch window and view it there. The value won't show up in the local or global variable list.

3.2.18 Double Mapping Variables

One important concept is Double Mapping of physically located variables. This allows the end user to accomplish some other tasks. For example we have a 5060 Digital output board. When mapped as BOOL, it will create 16 variables. In the example below an additional WORD variable has been added and mapped to the same memory location.

⊟ **Out_APAX5060_1_16**				
APAX5060_B01C000_O	BOOL	VAR_GLOBAL	[Slot1]APAX5060 Relay xhannel 0	%QX4.0
APAX5060_B01C001_O	BOOL	VAR_GLOBAL	[Slot1]APAX5060 Relay xhannel 1	%QX4.1
APAX5060_B01C002_O	BOOL	VAR_GLOBAL	[Slot1]APAX5060 Relay xhannel 2	%QX4.2
APAX5060_B01C003_O	BOOL	VAR_GLOBAL	[Slot1]APAX5060 Relay xhannel 3	%QX4.3
APAX5060_B01C004_O	BOOL	VAR_GLOBAL	[Slot1]APAX5060 Relay xhanne 4	%QX4.4
APAX5060_B01C005_O	BOOL	VAR_GLOBAL	[Slot1]APAX5060 Relay xhannel 5	%QX4.5
APAX5060_B01C006_O	BOOL	VAR_GLOBAL	[Slot1]APAX5060 Relay xhannel 6	%QX4.6
APAX5060_B01C007_O	BOOL	VAR_GLOBAL	[Slot1]APAX5060 Relay xhannel 7	%QX4.7
APAX5060_B01C008_O	BOOL	VAR_GLOBAL	[Slot1]APAX5060 Relay xhannel 8	%QX5.0
APAX5060_B01C009_O	BOOL	VAR_GLOBAL	[Slot1]APAX5060 Relay xhannel 9	%QX5.1
APAX5060_B01C010_O	BOOL	VAR_GLOBAL	[Slot1]APAX5060 Relay xhannel 10	%QX5.2
APAX5060_B01C011_O	BOOL	VAR_GLOBAL	[Slot1]APAX5060 Relay xhannel 11	%QX5.3
APAX5060_B01_ALL	WORD	VAR_GLOBAL	All Outputs for Board 1	%QW4

Same Address

This allows control of all outputs on this board with one variable. If the user wants to turn on three outputs, they can use three commands, one for each BOOL variable, or control the whole board with one command as shown below.

```
APAX5060_B01C000_O := 1;(* Enable Output 0 *)
APAX5060_B01C003_O := 1;(* Enable Output 3 *)
APAX5060_B01C005_O := 1;(* Enable Output 5 *)
APAX5060_B01_ALL := 41;(* Enable Outputs 0,3,5 Disable all others on B01
```

This is also convenient for monitoring multiple inputs with one variable, reducing code size. Double mapping is also an important concept for Memory variables. This allows multiple memory mapped variables to be combined into one larger memory mapped variable.

Users can also have two different variables mapped to the same address. This can cause some confusion and use more memory, but this is possible.

|				
APAX5060_B01C000_O	BOOL	VAR_GLOBAL	[Slot1]APAX5060 Relay xhannel 0	%QX4.0
WarningLight	BOOL	VAR_GLOBAL		%QX4.0

Same Address

3.2.19 Memory Variables

Memory variables are global variables that have an address assigned to them. When a global variable is declared, its physical location in memory is decided by the compiler. This means that the only way to reference this variable is by name. If the user wants to reference a variable by location as well as by name then memory variables are used.

PhysicallyLocatedVars				
MemBool0	BOOL	VAR_GLOBAL	Bit Zero	%MX0.0
MemBool1	BOOL	VAR_GLOBAL	Bit One	%MX0.1
MemBool2	BOOL	VAR_GLOBAL	Bit two	%MX0.2
MemByte0	BYTE	VAR_GLOBAL	Byte Zero	%MB0

When using Memory Variables, you must verify that the automatic declaration is done. To do this go to Resource Settings and select the Data Area button and check the box "Declare user memory automatically".

☑ Declare user memory automatically
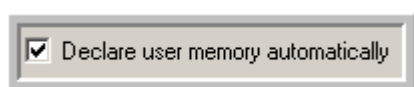
3.2.20 Retentive Variables

There is a checkbox in the Variable worksheet for local variables as well as global variables that says "Retain". This will ensure that on a hot or warm boot, that the variable will have the same value as when it was shut down. A cold restart will re-initialize ALL variables including the retained variables.
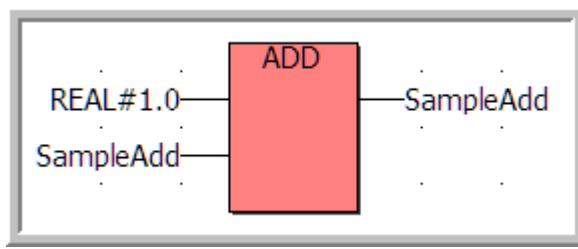
3.2.21 Literals

Sometimes the user will find that they want to put a constant number in place of a variable. This is easy to accomplish, but there is something to note. If the constant is anything other than an INT, then it must be explicitly defined when used. Some examples are below:
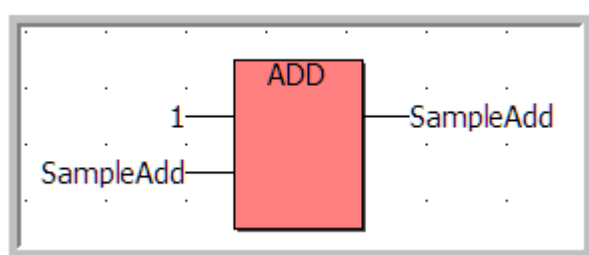
Add Integer number 1.
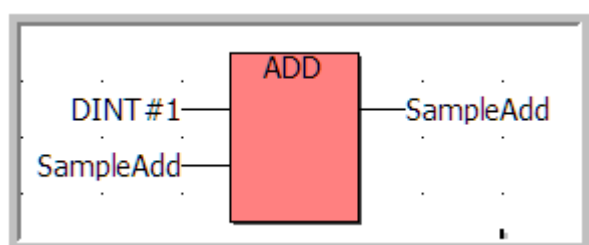
Add Double Integer number 1. Defined explicitly as DINT#1.

Add Real number 1.0. Defined explicitly as REAL#1.0.

3.2.22 Advantech Firmware Function Blocks

Advantech has provided some firmware function block libraries. For help on these functions, you can select help by right clicking on the graphical representation of the function block or find the help file located in the same directory as the firmware function library. This will be located on your hard disk where KW Multiprog has been installed at: \MULTIPROG\PLC\FW_LIB

Appendix A

Version & Firmware Information

A.1 Version Information

A.1.1 ProConOS

There are two ways to find out the version of ProConOS.

The version is displayed when connected with the MULTIPROG development software. Select the Project Control dialog and click the Info button for the "Firmware" version. This is referring to the Advantech build of ProConOS. The PLC version is referring to the KW version number.
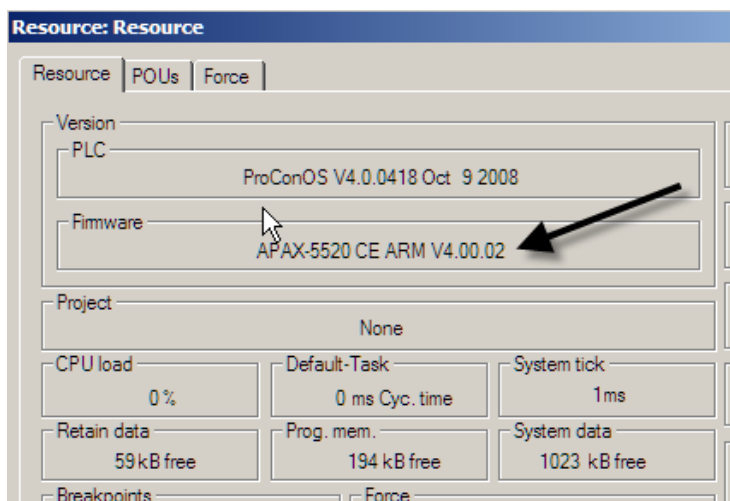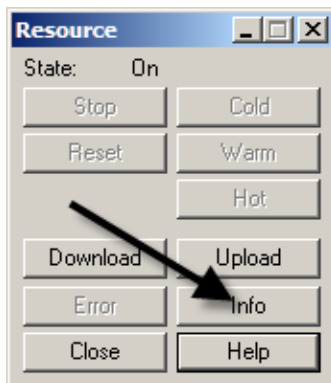
Use the Advantech Version tool that is supplied with the Windows CE. This will supply the version number that is written to the registry when ProConOS is installed in the OS.
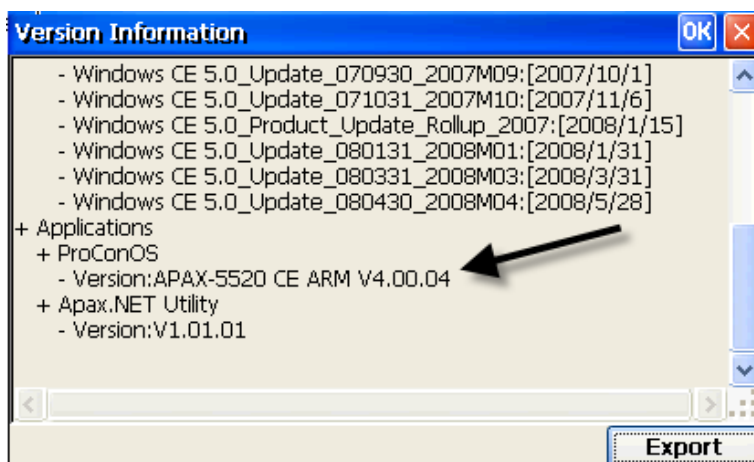
A.2 Firmware

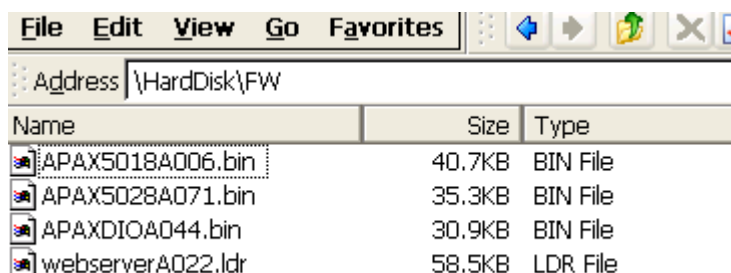The term firmware here is referring to driver firmware that is added to the ProConOS software. Updates can be found via the Advantech web site. These are other drivers that can change independently of ProConOS. These files contain the version number in their file name. The files can be found and updated through explorer in the WinCE OS. They can be found at \HardDisk\FW. Some examples appear below.

File	Edit	View	Go	Favorites

Address \HardDisk\FW

Name	Size	Type
APAX5018A006.bin	40.7KB	BIN File
APAX5028A071.bin	35.3KB	BIN File
APAXDIOA044.bin	30.9KB	BIN File
webserverA022.ldr	58.5KB	LDR File

Appendix B

APAX.NET Utility Operation

B.1 APAX.NET Utility General Window

After you install the APAX.NET utility, you can launch it through Start>>Programs>>Advantech>>ApaxNET Utility. Refer to Section 2.5.1 for installation information. Or you can click the shortcut in the HarDisk folder under My Device.
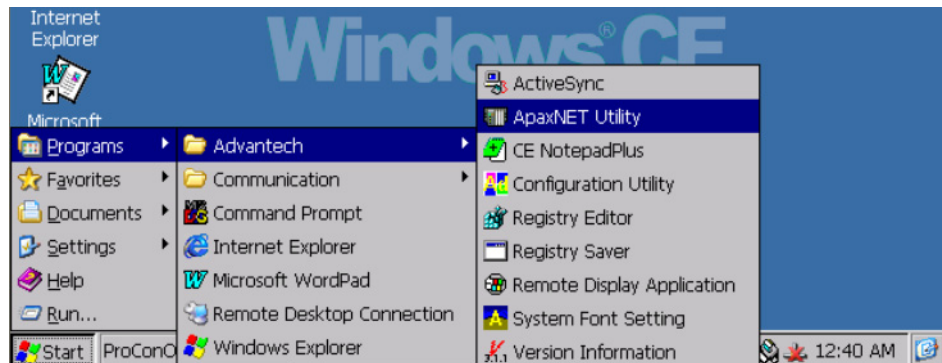

Note! *It may take around 25 ~ 30 second to launch the utility.*

Warning! *We strongly suggest to close APAX.NET utility after you complete your configuration to release system memory for other applications.*

After you launch the utility, you should see the operation window as figure below. Except APXA-5000 I/O modules, other devices such as ADAM-4000, ADAM-5000 and ADAM-6000 modules can also be searched and configured in this utility.
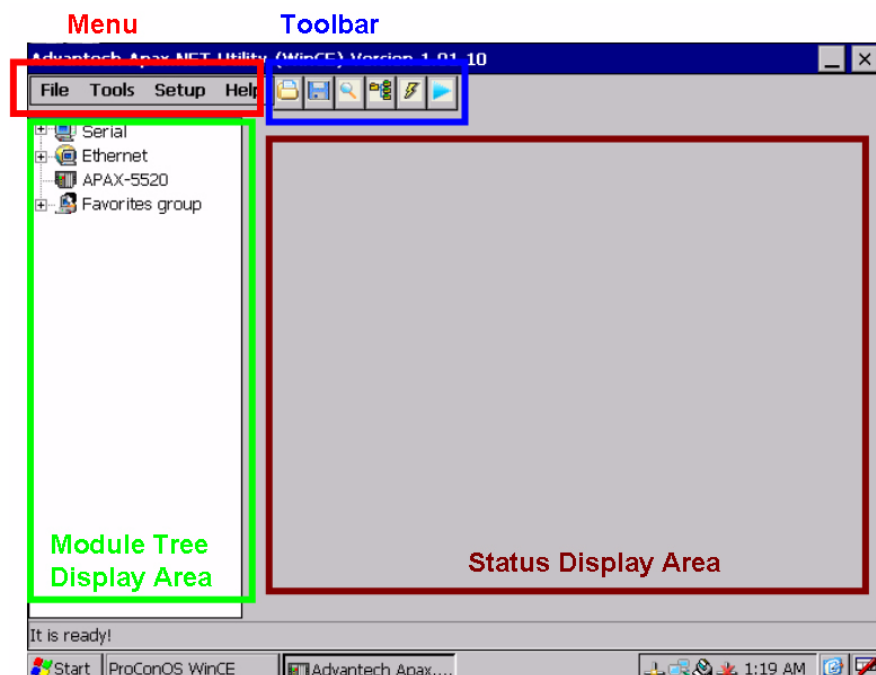
The operation window consists of four areas --- the **Menu**, the **Toolbar**, the **Module Tree Display Area** and the **Status Display Area**.

B.1.1 Menu

The menu at the top of the operation window contains:

- The **File** menu
1. **Open Favorite Group** - You can configure your favorite group and save the configuration into one file. Using this option, you can load your configuration file for favorite group.
2. **Save Favorite Group** - You can configure your favorite group and save the configuration into one file. Using this option, you can save your favorite group into one configuration file.
3. **Auto-Initial Group** - If you want to have the same favorite group configuration when you exit APAX.NET utility and launch it again, you need to check this option.
4. **Exit** - Exit APAX.NET Utility.

- The **Tools** menu
1. **Search** - Search if there are any remote I/O modules connected. For I/O modules communicated by serial (such as ADAM-4000 modules), click the **COM1** item (COM 2 is an internal COM port) under **Serial** item in the **Module Tree Display Area** first before you click this button. For I/O modules communicated by Ethernet (such as ADAM-6000 modules), click the **Ethernet** item in the **Module Tree Display Area** first before you click this button.
2. **Add Devices to Group** - You can add any I/O modules to your favorite group by this option. You need to select the device you want to add in the **Module Tree Display Area** (it will be described below) first, and then select this option to add.
3. **Terminal for Command Testing** - ADAM modules support ASCII commands and Modbus as communication protocol. You can launch the terminal to communicate with remote module by these two kinds of protocols directly. Refer to ADAM-4000, ADAM-5000 and ADAM-6000 manual for ASCII and Modbus command.
4. **Monitor Stream/ Event Data** - ADAM-6000 modules support Data Stream function. You can use this to configure related setting for the connected ADAM-6000 modules connected. Refer to ADAM-6000 manual for more detail.

- The **Setup** menu
1. **Favorite Group** - You can configure your favorite group including add one new device (only for remote device), modify or delete one current device, sort current devices and diagnose connection to one device.
2. **Refresh COM and LAN node** - APAX.NET utility will refresh the serial and LAN network connection situation.
3. **ShowTreeView** - Check this option to display the **Module Tree Display Area** or not.
4. **Add COM Port Tree Nodes** - This option is used to add serial COM ports in APAX.NET Utility.
5. **Delete the COM Port** - This option is used to delete serial COM ports in APAX.NET Utility.

- The **Help** menu
1. **Check Up-to-Date on the Web** - Choose this option, it will automatically connect to Advantech download website. You can download the latest utility there.
2. **About Apax.NET Utility** - Choose this option, you can see version of APAX.NET Utility installed on your computer.

B.1.2 Toolbar

The six buttons on the toolbar represent the six commonly used items from the **Menus**. Refer to figure below for the definition of each button:
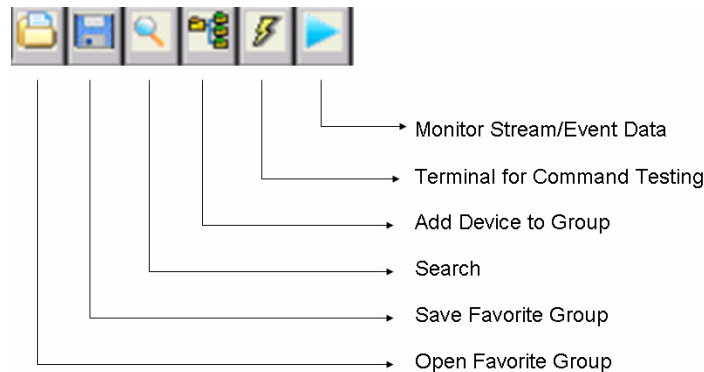
B.1.3 Module Tree Display Area

APAX.NET Utility is one complete software tool that all APAX and ADAM I/O module can be configure and operated in this utility. The **Module Tree Display Area** is on the left part of the utility operation window. There are four categories in the **Module Tree Display Area**:

■ **Serial**

All serial remote I/O Modules connected to APAX-5520 will be listed in this category. You also can configure COM port parameter (such as baud rate, parity, stop bit, etc.) here.

■ **Ethernet**

All Ethernet remote I/O Modules connected to APAX-5520 will be listed in this category.

■ **APAX-5520**

All APAX-5000 local I/O modules in the same system will be listed in this category. Simply click this item all related modules will be displayed automatically.

■ **Favorite Group**

You can define which devices listed in **Serial** or **Ethernet** categories above into your personal favorite group. This will make you easier to find your interested modules. Click on the **ADAM device** item under **Favorite group** item, and select **Favorite >> New** in Setup menu to create a new group. After you create your own group, click on your group and select **Favorite >> New** in **Setup** menu to add any remote devices into your group. You can also select **Diagnose connection** to check the communication.

B.1.4 Status Display Area

Status Display Area, on the right part of utility operation window, is the main screen for operation. When you select different items in **Modules Tree Display Area**, **Status Display Area** will change dependently. You can do all configurations and tests on this area.

B.2 General Configuration

If you click the **APAX-5520** item in the **Module Tree Display Area**, the **Status Display Area** should looks similar to the figure below:
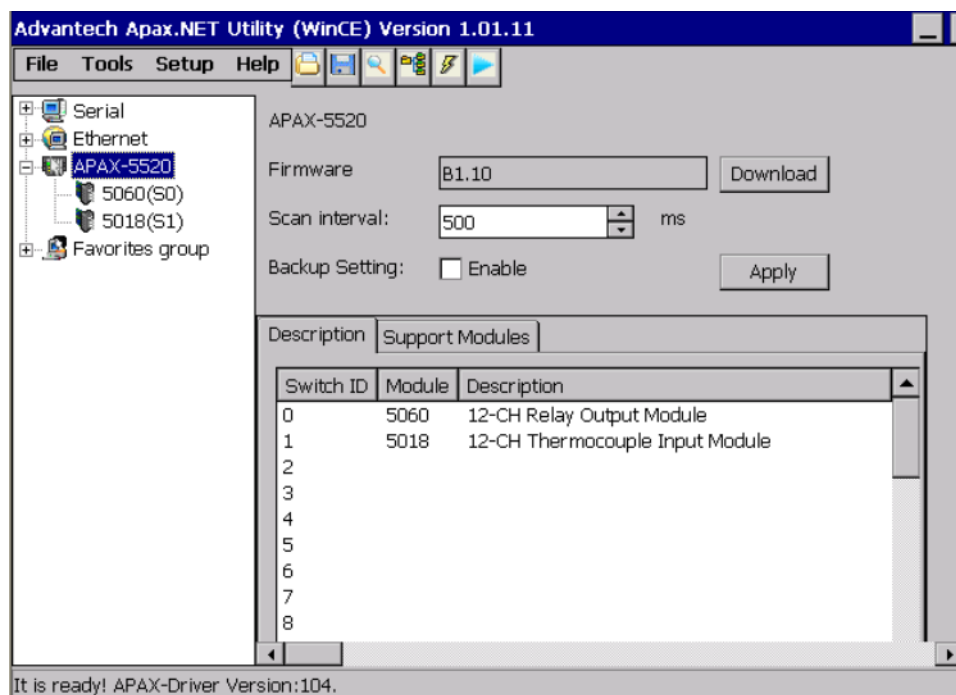
All I/O modules with its ID number are listed in the **Description** tab in the **Module Tree Display Area** (the left tab) and **Description** tab on **Status Display Area** (the right tab). You can see all I/O modules supported by APAX-5520 by the **Support Modules** tab on **Status Display Area**. The **Backup Setting** check box is used to enable or disable APAX-5520 backup function. Refer to Appendix C for more detail about backup functionality.

B.3 I/O Modules Configuration

When you click any I/O module in the **Module Tree Display Area**, the **Status Display Area** at the right side will automatically change to show the module's information. There will be two tabs displayed: **Module Information** and **I/O Information**. (Refer to the figure below)
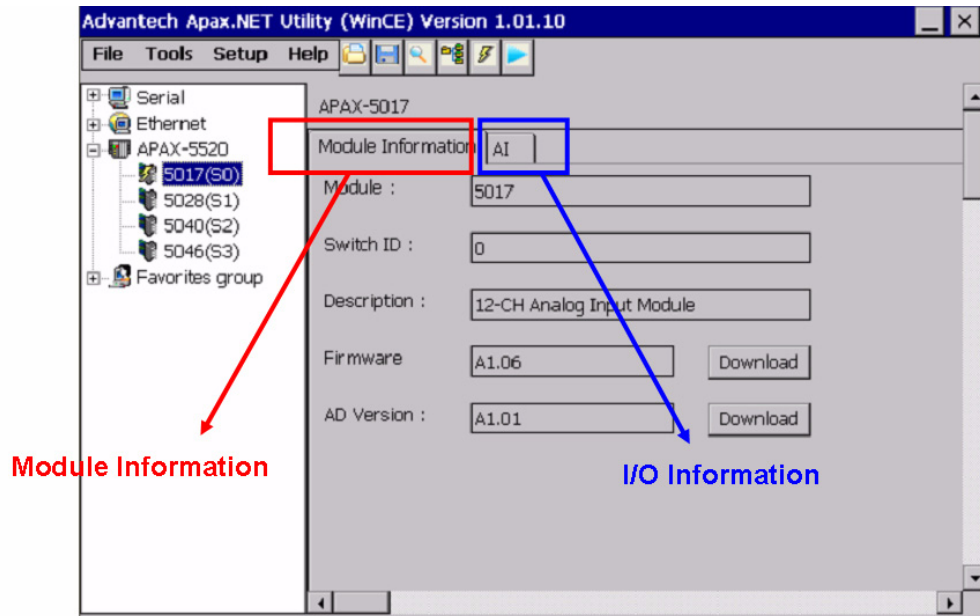
On the **Module Information** tab, information such as module name, switch ID, module description, and firmware version is displayed. You also can update related firmware to the specific module by the **Download** button.

On the **I/O Information** tab, you can write or read all channels' status and perform related configuration and calibration. Refer to sections below for more detail.

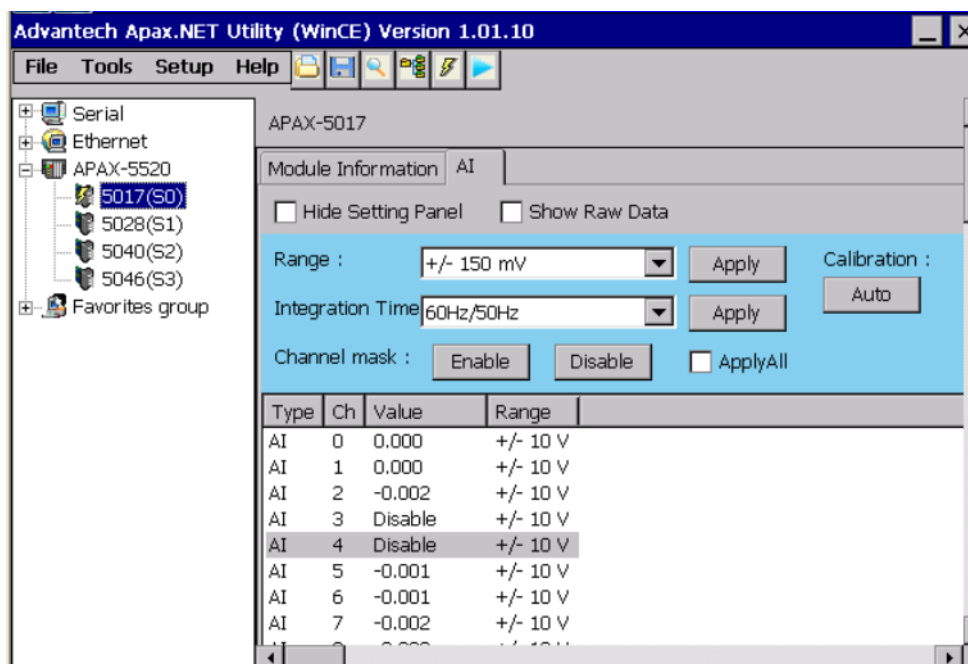## B.3.1 Analog Input Modules
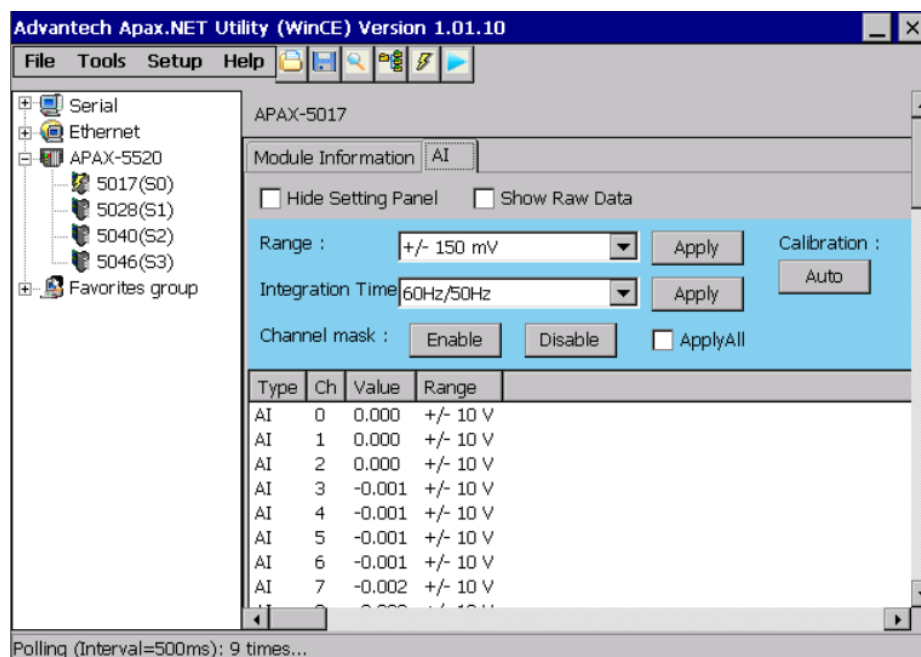
There are two parts for the **I/O Information** tab of APAX-5000 AI module. At the bottom is the **Channel Status** Area. You can see all channels' type, value, and range. Above the **Channel Status** Area is the **Setting Panel** Area. If you don't want see the **Setting Panel** Area, you can click the **Hide Setting Panel** check box to hide the **Setting Panel** Area. If you want to see the raw data (presented in Hexadecimal format) from the input channels, click the **Show Raw Data** check box.

If you want to configure specific input channels' range or integration time, select the channels in the **Channel Status** Area. Choose appropriate range and integration time by the **Range** and **Integration Time** combo boxes in the **Setting Panel** Area and then click the **Apply** button to save the configuration. If you want to save the same range setting for all channels, click the **ApplyAll** check box before you click the **Apply** button.

> **Note!**   In order to remove the noise from the power supply, APAX AI modules feature built-in filter. Filters are used to remove noise generated from environment. The integration time is used to configure the filter frequency.

You can define specific channels reading or not by the **Enable** and **Disable** button. Refer to figure below, channel 3 and channel 4 are disabled that no data will be read.

By clicking on the **Auto** button, you can perform auto calibration to the AI module. The module will automatically calibrate itself.
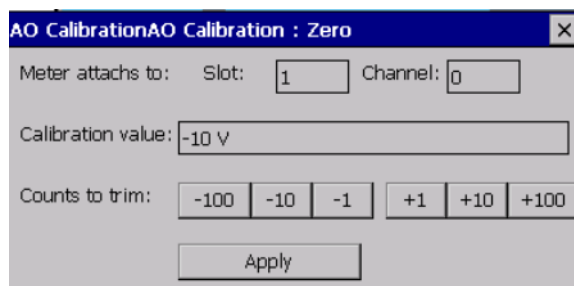
B.3.2 Analog Output Module
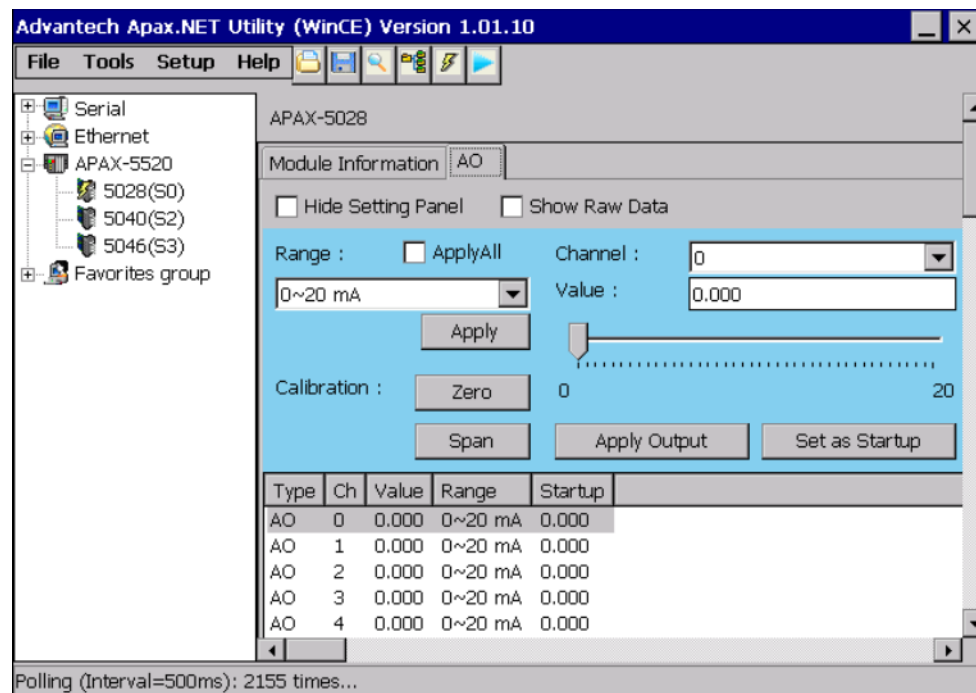
There are two parts for the **I/O Information** tab of APAX-5000 AO module. At the bottom is the **Channel Status** Area. You can see all channels' type, value, range and startup value (the initial value when the AO module is power-on). Above the **Channel Status** Area is the **Setting Panel** Area. If you don't want see the **Setting Panel** Area, you can click the **Hide Setting Panel** check box to hide the **Setting Panel** Area. If you want to see the raw data (presented in Hexadecimal format) from the output channels, click the **Show Raw Data** check box.
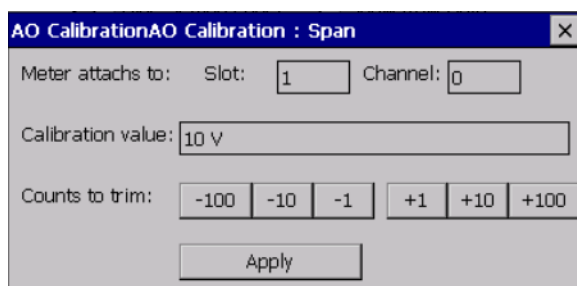
If you want to configure specific output channels' range, select the channels in the **Channel Status** Area. Choose appropriate range by the **Range** combo box in the **Setting Panel** Area and then click the **Apply** button to save the configuration. If you want to save the same range setting for all channels, click the **ApplyAll** check box before you click the **Apply** button.

If you want to change specific output channel' output value, select that channel by clicking the channel in the **Channel Status** Area or choosing it from **Channel** combo box in the **Setting Panel** Area. Then define the output value by the **Value** text box or the horizontal slide below in the **Setting Panel** Area. Then, click the **Apply** button to save the configuration. You can see the channel output value changed in the **Channel Status** Area. Similarly, you can save the value in the **Value** text box to become the startup value by the **Set as Startup** button. And you also can see the startup value changed in the **Channel Status** Area.

By clicking the **Span** button and **Zero** button, you can perform span calibration and zero calibration, separately. When you click the **Zero** button, you will see a dialog popping-up as figure below. The specific channel will generate output signal using the minimum value within range which is shown in the **Calibration Value** text box. Connect that channel to an external accurate instrument and measure the output signal. Using the **Counts to trim** buttons to adjust until the output value real matches the value in the **Calibration Value** text box. Then click the **Apply** button to save the calibration configuration.

When you click the **Span** button, you will see a dialog popping-up as figure below. The specific channel will generate output signal using the maximum value within range which is shown in the **Calibration Value** text box. Connect that channel to an external accurate instrument and measure the output signal. Using the **Counts to trim** buttons to adjust until the output value real matches the value in the **Calibration Value** text box. Then click the **Apply** button to save the calibration configuration.
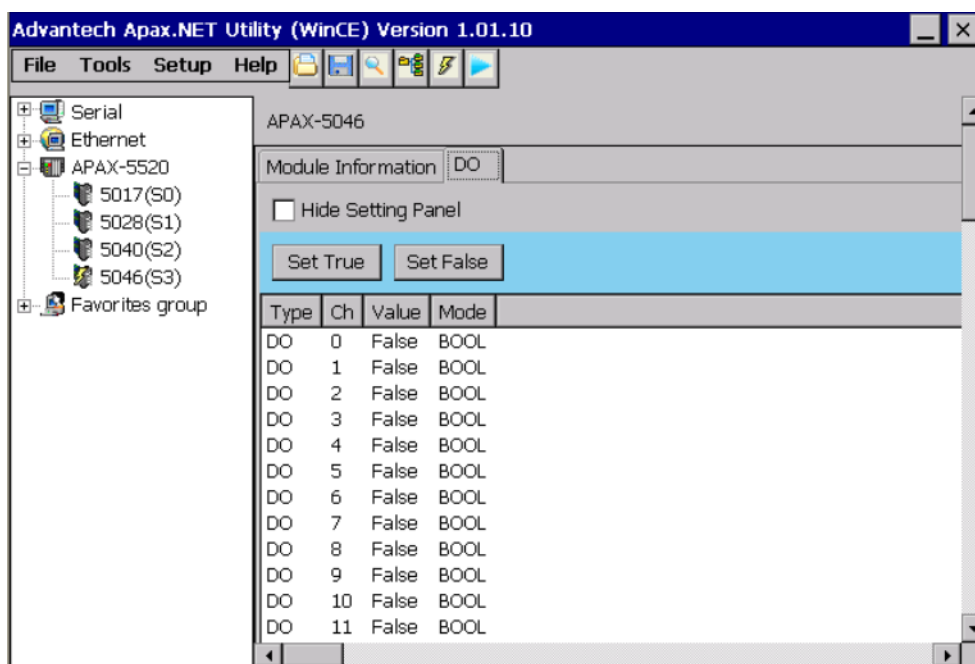

B.3.3 Digital Input Module
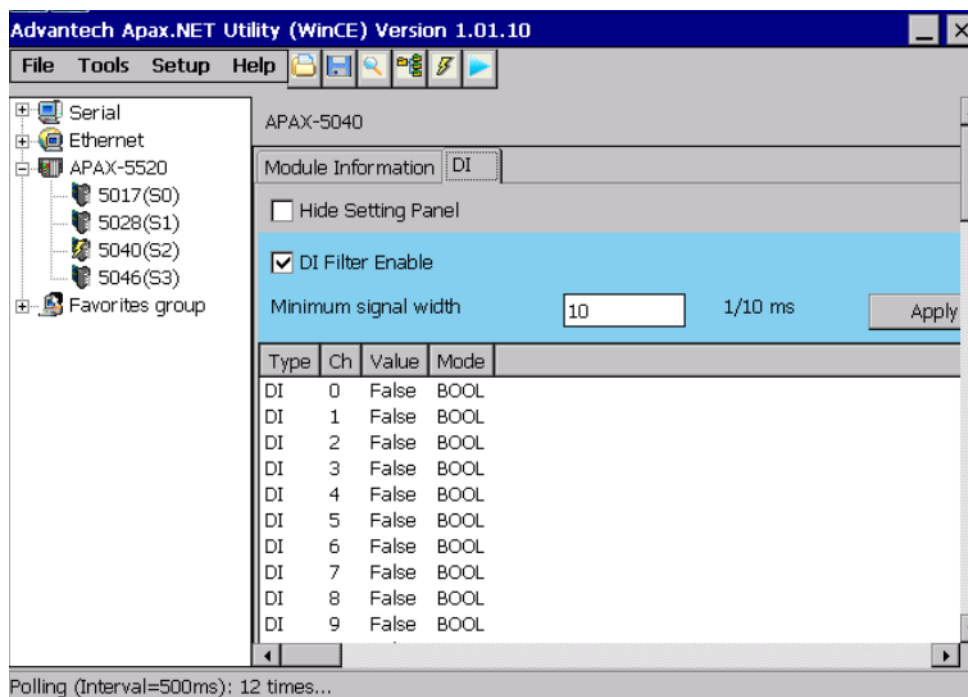
There are two parts for the **I/O Information** tab of APAX-5000 DI module. At the bottom is the **Channel Status** Area. You can see all channels' type, value, and mode. Above the **Channel Status** Area is the **Setting Panel** Area. If you don't want see the **Setting Panel** Area, you can click the **Hide Setting Panel** check box to hide the **Setting Panel** Area.

If you want to configure specific output channels' filter status or minimum acceptable pulse width, select the channels in the **Channel Status** Area. Click the **DI Filter Enable** check box in the **Setting Panel** Area to enable filter for that channel. Type the appropriate value (unit: 0.1 ms) into the **Minimum signal width** text box to configure acceptable minimum pulse width in the **Setting Panel** Area. After you complete the configuration, click the **Apply** button to save the configuration.

B.3.4 Digital Output Module

There are two parts for the **I/O Information** tab of APAX-5000 DO module. At the bottom is the **Channel Status** Area. You can see all channels' type, value, and mode. Above the **Channel Status** Area is the **Setting Panel** Area. If you don't want see the **Setting Panel** Area, you can click the **Hide Setting Panel** check box to hide the **Setting Panel** Area.

If you want to change specific output channels' output value, select those channels by clicking the channel in the **Channel Status** Area. Then define the output value by the **Set True** button or **Set False** button in the **Setting Panel** Area. Then, click the **Apply** button to save the configuration. You can see the channel output value changed in the **Channel Status** Area.

Appendix C

**System Backup Functionality**

C.1 Introduction

APAX-5000 series delivers system backup functionality. To leverage this functionality, two CPU modules (controllers), with the same control program, are installed in one system. After both controllers' backup function is enabled, the APAX-5000 system will automatically delegate one of the two controllers as the master controller.

The master controller will run the control program to execute the control process, while another controller (the backup controller) is put on standby. The master controller will periodically send living message to the backup controller. If the backup controller dose not receive living message from master controller over 500 milliseconds, it will automatically become master controller and take the control responsibility and restarts the control process execution. The maximum operation time for the backup controller to become master controller (the take over time) won't be greater than 1.5 second.

Changing master controller means there is something wrong for the previous master controller. Therefore, engineers can check or change the previous master controller with a new one and enable it to have backup functionality, becoming a second backup controller. Then if the new master controller fails again, the second backup controller will automatically take the control responsibility.

This mechanism ensures the control system will continuously run the control process. And the system won't be stopped even if controller fails.
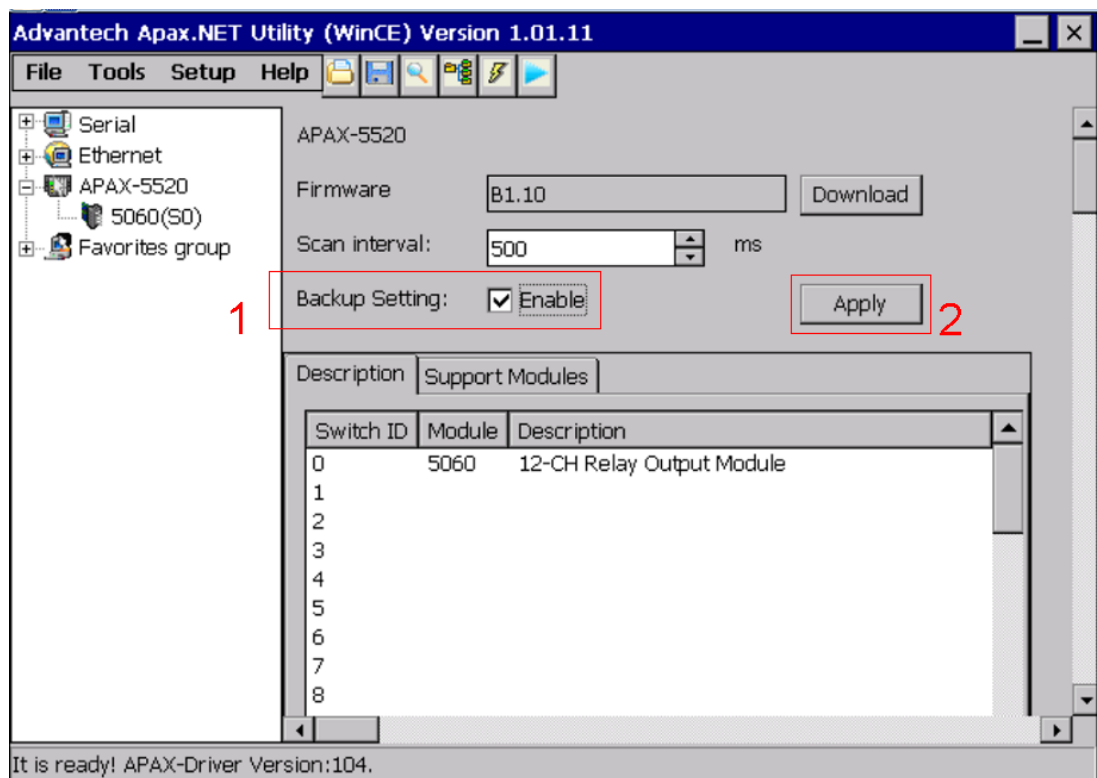
C.2 Configuration
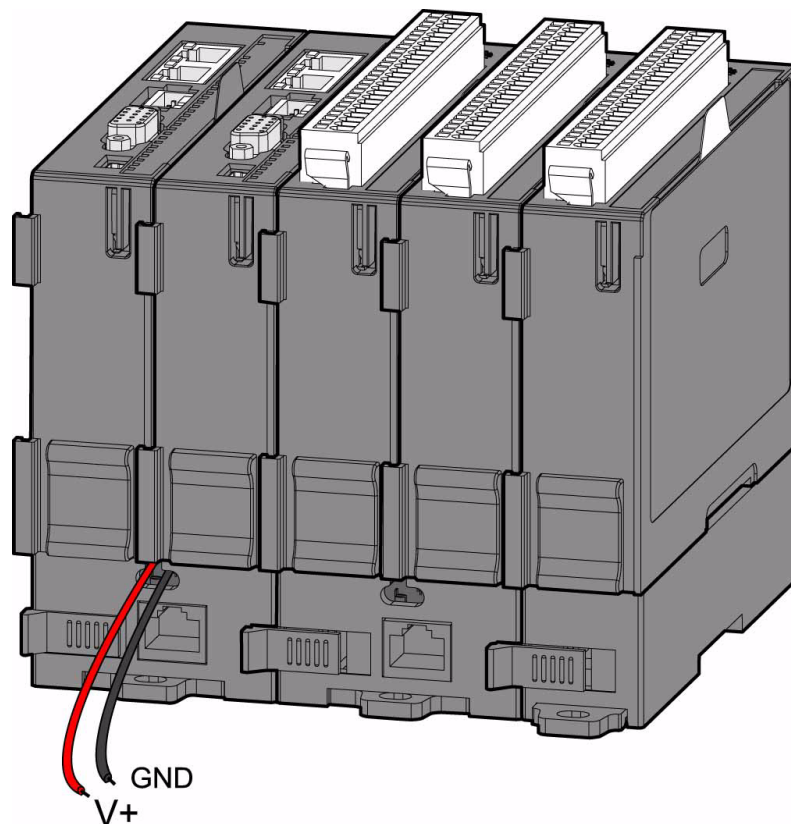
As you can see in the figure above, two APAX-5520 CPU modules are installed in one system. APAX-5000 series will automatically decide which one is the master controller.

Warning! *The controller ID of the APAX-5520 module is auto-identified by the location where the module is inserted on the APAX-5002 backplane (Slot 1 or 2). Thus, be sure NOT to insert two APAX-5520 modules in the same location on two backplanes. For example, if you insert one APAX-5520 on slot 1 of one APAX-5002 backplane and insert the second on slot 1 of another APAX-5002 backplane in the same system, APAX-5000 series cannot distinguish the two APAX-5520 modules.*
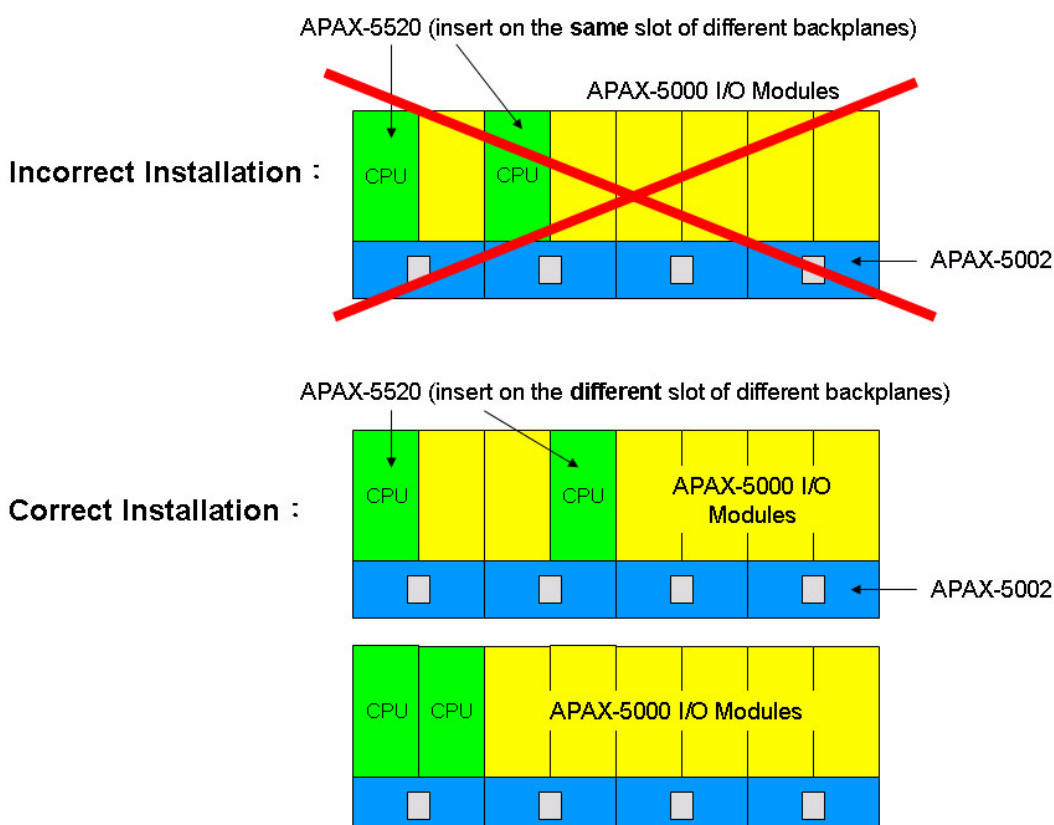
Backup functionality needs to be enabled for both the two APAX-5520 modules, in the APAX utility. Refer to figure below. Click the **Backup Setting** check box in **Setting Panel** Area and then click the **Apply** button to enable backup functionality for APAX-5520.


Note! *After applying the configuration for the backup system, remember to power cycle the whole system to run the backup functionality.*

C.3 Programming Backup in KW

After you enable backup functionality by utility, you can leverage the backup function-ality into your KW program. Before you start your programming, you need add the Backup I/O driver into the project for later use. Follow the procedure below to add the Backup I/O driver:

In the **Input** tab of I/O configuration window, select the **BackupSys** component and click the **Apply** button to add into project.
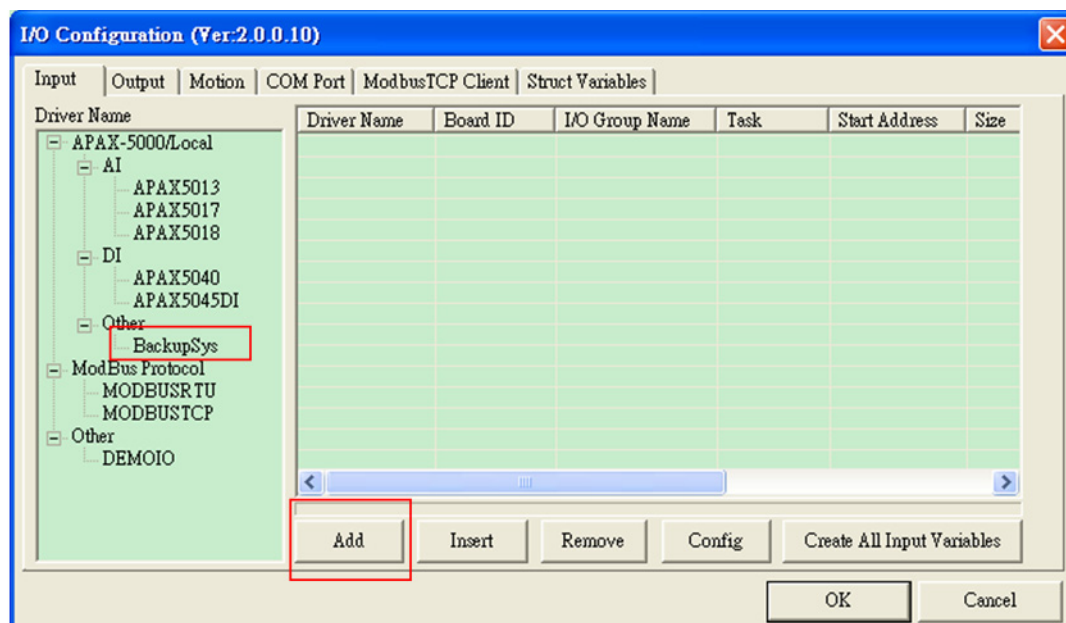

Then one pop-up window as shown by figure below will appear. You need to select one task by the **Task** combo box for the **BackupSys** component to be added. As described in Section C.1, the master controller will send living message to backup controller periodically and the backup controller will automatically become master controller if it does not receive living message within 500 ms. Therefore, remember to select one task with cyclic type and less than 500 ms interval time.
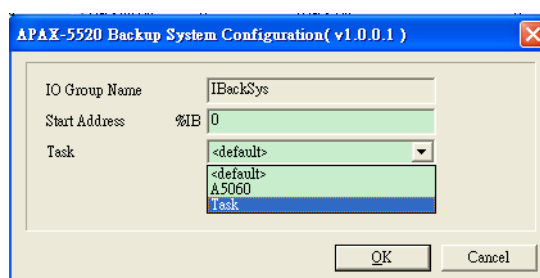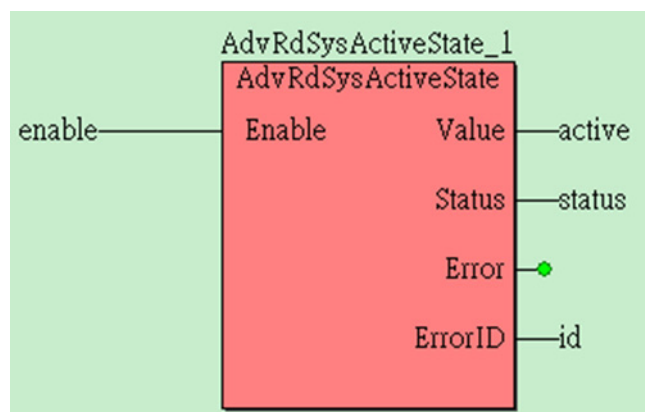

After you select the task, click the **OK** button to finish the configuration.

Change the configuration window to **Outpu**t tab. Run the same procedures to add the **BackupSys** component into the project.
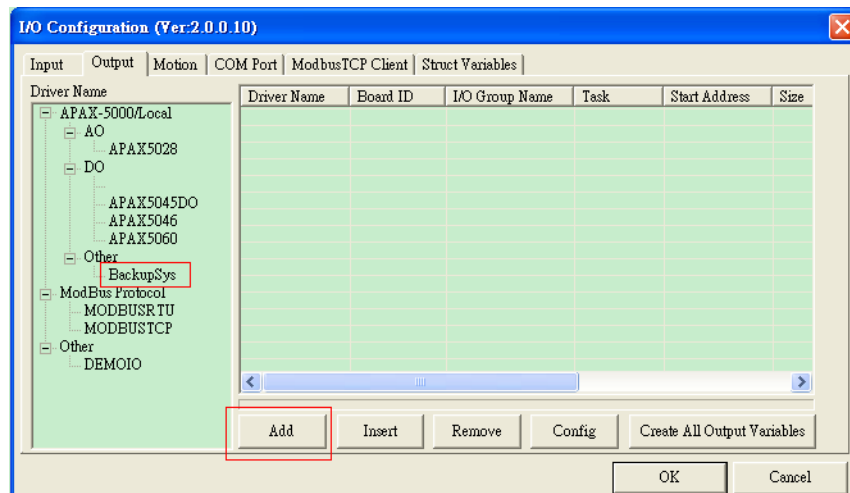
Then you start to write your control program in KW MultiProg. To leverage backup function, use the function block **"AdvRdSysActiveState"** (Refer to the figure below). As described in Section C.1, APAX-5000 series will automatically assign the master controller and backup controller. You can use this function block to know if the controller is master controller currently, by the parameter **Value**. If the **Value** responses "**True**", it means the controller is master controller. If the **Value** responses "**False**", it means the controller is backup controller. So your control program should use this parameter to decide if the controller should execute the control or simply be put standby.

ADVANTECH

*e*Automation