Getting started with Advantech Embedded Control System

www.advantech.com

http://www.e-automation.cc/

Table of Contents

1. Introduction

Advantech **E**mbedded **C**ontrol **S**ystem (**ECS**) is the latest addition to Advantech's eAutomation product family. ECS is a powerful product designed to offer real-time control functionality on a cost-effective embedded PC platform. The ECS software is based on KW-software's Multiprog and ProConOS (Programmable Controller Operating System) products that have been integrated and tested with selected Advantech eAutomation devices. **ECS** is also compatible with all of Advantech's ADAM I/O product lines.

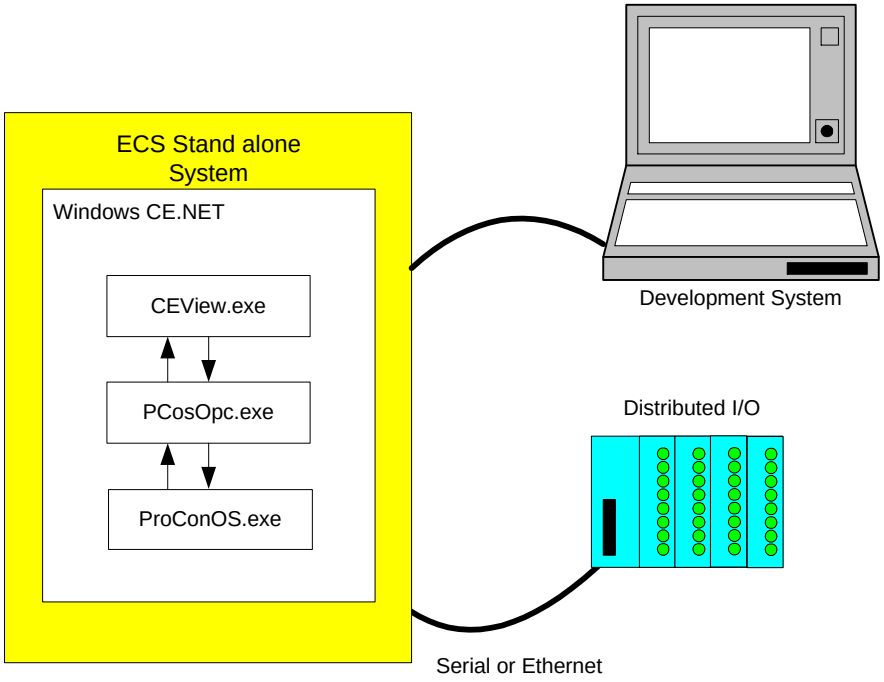
ECS is a modular design, offering the possibility to combine Advantech's AStudio web-enabled HMI software onto the same hardware device for a complete control and visualization solution in one package. The KW software can also be used without HMI software, or with a remote HMI using OPC data communication.

This document is intended to give the user some basic information to start using the Advantech **ECS**. This document is not intended to provide programming help for KW-Software's Multiprog or Advantech AStudio. Please refer to the online help files for these products for any information not covered in this document.

ECS Architecture – Integrated System

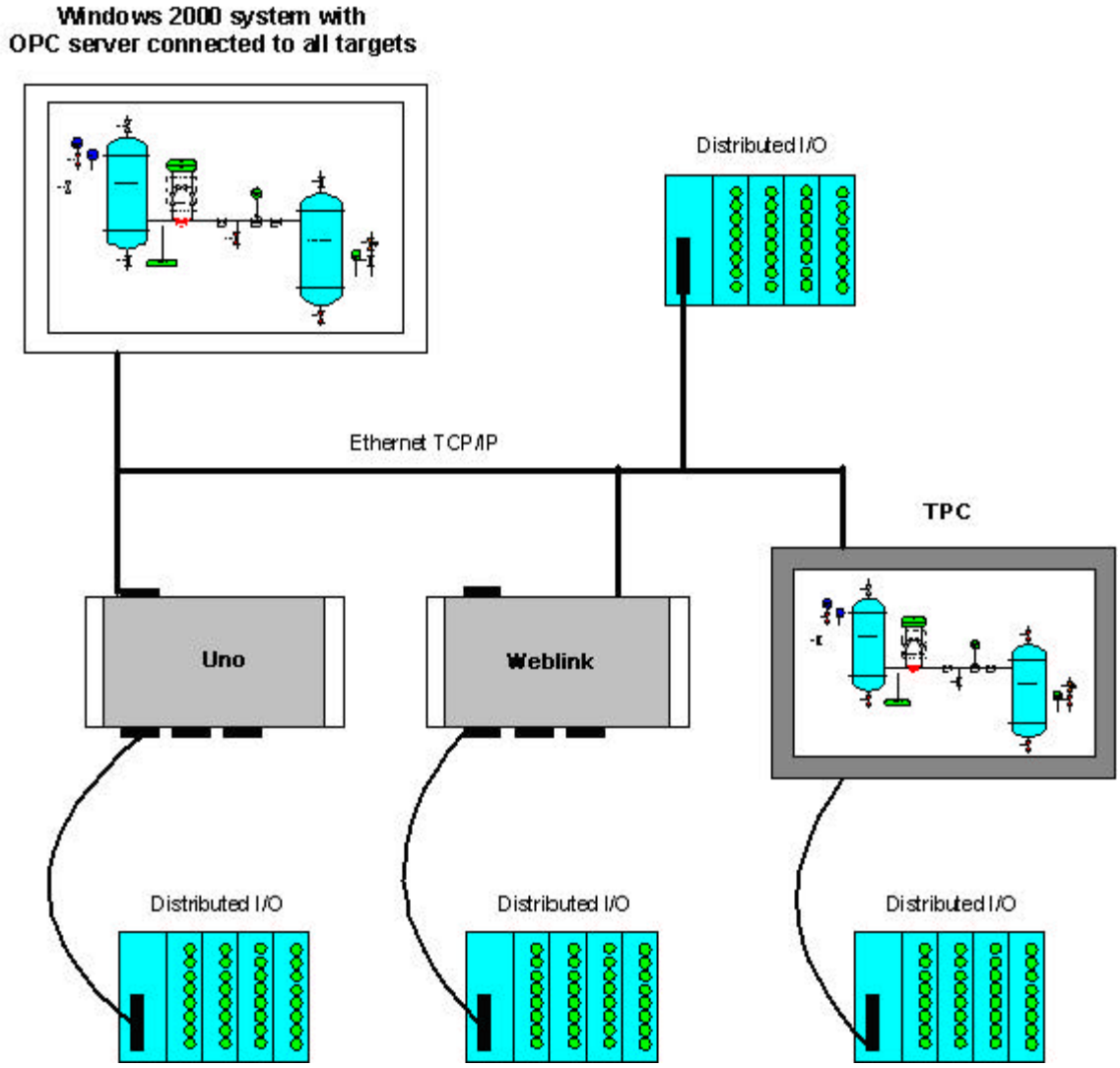
There are several possibilities when setting up **ECS**. It can be used as an integrated system, or as part of a distributed system.

The integrated system would be made up of an **ECS** Target connected to I/O. The **ECS** could display the HMI and do the controlling of the I/O, as well as serve images to third party machines with the Web Server. The target system would have AStudio reading and writing tags to the ProConOS engine through an OPC server. If HMI and web server functions are not required, they may be omitted.



ECS Architecture – Distributed System

ECS can also be part of a distributed system. It is possible for the Target system to only do the control and have the tags available on a master system, or both. This can be done by installing the OPC server on the master system and connecting to the targets with this master system. This leaves the target systems more CPU time to do control processing and not HMI processing and the master system is the web server.



2. Windows CE Target Systems

Getting Around Windows CE

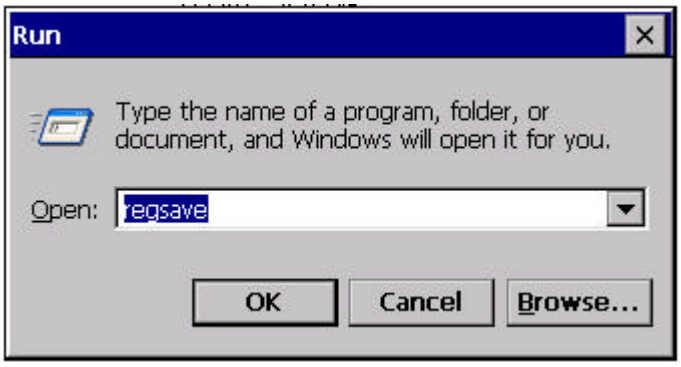
There are some things that will be helpful when using windows CE that are mentioned here.

Control Panel. Like windows desktop systems, there is a control panel that has some setup items. Some items that can be useful are:

1. **Input Panel** – this is configuration for the on screen keyboard.
2. **Network and Dial-up Connection**- this is where to set the IP address of the unit.
3. **PC Connection** – This is where the ActiveSync connection can be changed to a different COM port
4. **Touch screen** calibration can be found here also.

If any changes are made then there must be a **registry save** done or the changes will not be retained.

Save the registry. Select Start then Run and type Regsave in the box and click the OK button.



Some targets will give a confirmation that the registry has been saved.

Windows Directory

The windows directory IS NOT part of persistent storage. If something is saved into the windows directory, it will be gone upon restart. That is because the Windows directory and all directories that are not under Harddisk or Harddisk2 are in RAM only and are loaded to ram upon startup from the image.

Network services.

If the CE device is connected to a network and has an IP address that is in the network, the end user can browse other machines on the network. Simply open Windows explorer from the Start\Programs menu and type the network machine name in the address bar. An example would be \\mycomputer\c_drive . In this example the Windows CE device

would look for a computer on the network with the name mycomputer and the shared drive name of c_drive. If it makes the connection then the user login requirements will be asked for: User name, password and Domain name.

Com Port Defaults

Currently the COM ports are assigned in Windows CE as described in the chart below. The end user can alter these settings but a registry change may be needed to do so. For the latest information on these defaults visit the web page at
http://www.e-automation.cc/

	COM 1	COM 2	COM 3	COM 4
UNO/Weblink-2059	ActiveSync	Available	Available	Available/TouchScreen
TPC-642-SE	Available (unless changed by user, ActiveSync is defaulted to USB)	Available	Available	N/A
TPC-1260	Available/ActiveSync (ActiveSync does not run on Startup as with Uno/Weblink)	Available	Available	Available

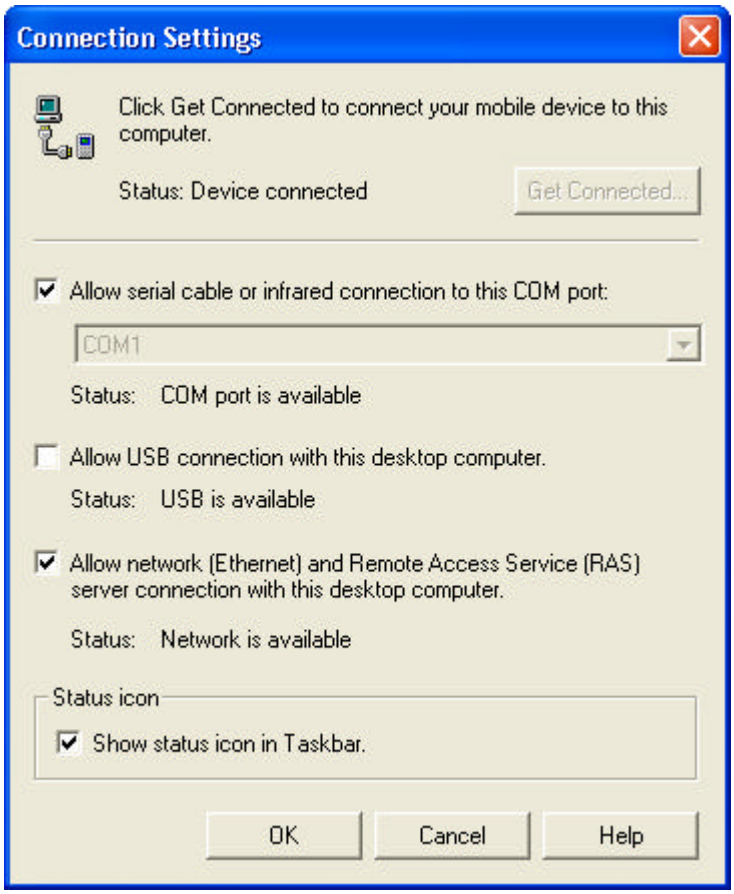
Using ActiveSync

As the chart says, ActiveSync is assigned to one of the COM ports. This does not mean that the COM port must be used for ActiveSync. This is just the default assignment. ActiveSync can be used to browse the target system and add files to the persistent storage. ActiveSync can be downloaded from the following location:
http://www.microsoft.com/mobile/pocketpc/downloads/default.asp

ActiveSync can be used for the following reasons:

Browse the Windows CE target
Download or upload files with the Windows CE target
Debug end user written applications
Use remote tools that come with Embedded VC++

When ActiveSync is installed on the development machine, it will use one COM port to connect with the target. The connection settings should be as shown in the picture.



To make the connection speed faster both the target and development applications should have Ethernet connections to the same network. This will also allow the target to browse the development machine if necessary. Once the settings are as show, the development machine is ready. There must also be a serial connection made with the cable provided with the target unit. This is a standard serial Crossover cable.

To make the COM port available for other applications, the checkbox for “Allow serial cable or infrared connection to this COM port” must be unchecked.

With ActiveSync started on the development machine and the cable connected, there must be an application started on the target machine called repllog.exe. This application automatically starts on the headless platforms. If no connection is found the application will time out and close. Other platforms may not provide this automatic startup.

If repllog.exe does not automatically start on power up, the application must be started from the target device. From the target device select Start\Programs\ActiveSync.

3. Install Development Software

To use ProConOS**,** with an UNO or TPC target, the following software must be installed on the development machine:

Multiprog 3.3 Build 57 or higher
Advantech Multiprog Add on Pack

Multiprog must be installed, and the Advantech Multiprog Add on pack must be installed. Once this is done then Multiprog must be licensed. The Advantech Multiprog add on pack contains updates to Multiprog to allow it to connect and program the Advantech **ECS** target systems.

To use ProConOS with a WebLink/WebOIT target, the following software must be installed on the development machine:

Multiprog 3.3 Build 57 or higher
Advantech Multiprog Add on Pack
Advantech Studio
KW OPC Server version 2.0 or higher

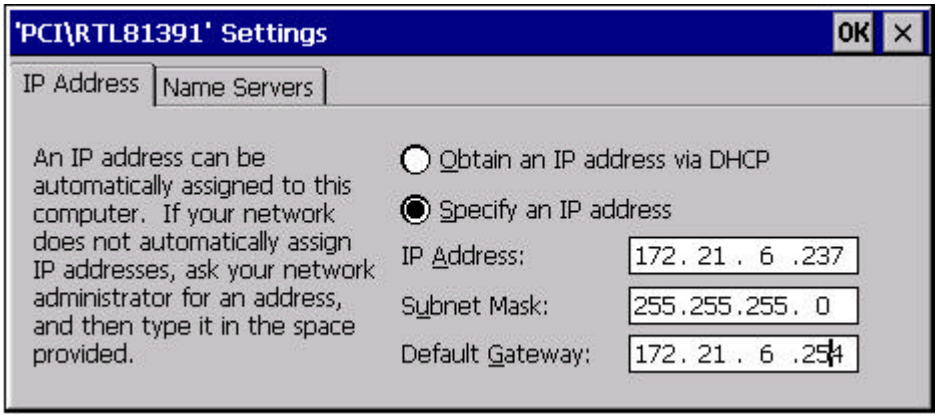
Please see the documentation on the AStudio CD for installation procedures for that software. The other software will be contained on the Advantech KW CD.

4. Power up the target system.

Upon power up, the target system will run a batch file that will automatically run the Repllog.exe program if the device is a headless device. This is designed to allow the headless device to connect to a development machine through ActiveSync without the use of a Keyboard or Mouse. In order for this to be accomplished the provided Null Modem cable must be connected and ActiveSync running on the development machine. Other items will also run from the batch file that will install registry settings if needed or other items from cab files.

5. Set the IP address for the target system.

To run the KW software, the target unit must have an IP address. DHCP can be used, but could cause problems if the IP address changes. This is because the IP address is part of the PLC application that is downloaded to the target. To set up the IP address on the Target unit, from the start menu select “settings/Control Panel/Network and Dial-up Connections. From there select the proper Network Adaptor. Specify an IP Address as the example shows below. This IP Address will also be used in your Multiprog application.



Once the IP address is set, run the regsave utility to save these settings to the registry.

6. Starting the ProConOS engine.

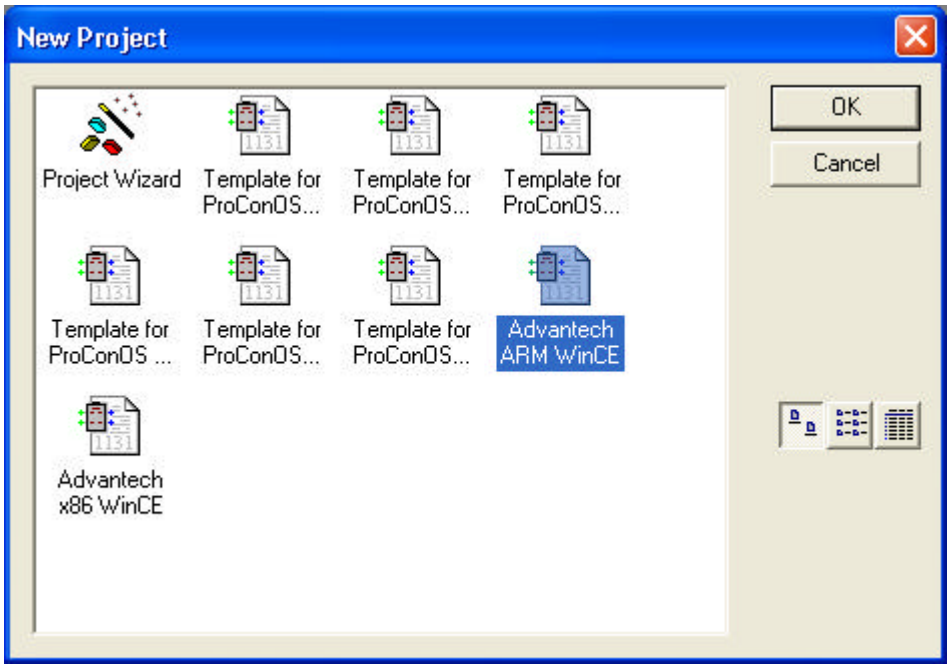
The ProConOS engine is located in the directory \harddisk\Proconos\proconos.exe. This executable is automatically started upon startup and cannot be shut down. It does not use much CPU time unless a user application is loaded. The CPU load is based on the size of the user application and amount of I/O configured.

7. Multiprog.

Creating a sample application with the demo driver:
NOTE: THE DEMO DRIVER IS CURRENTLY NOT AVAILABLE FOR THE ARM CPU TARGETS.

To create a sample application without connecting to the target, do the following steps.

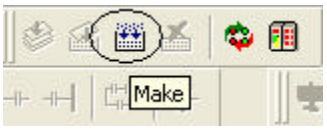
a. Open the Multiprog software and select “New Project” from the file menu.
b. Select the Advantech template that corresponds to your target hardware and press OK.



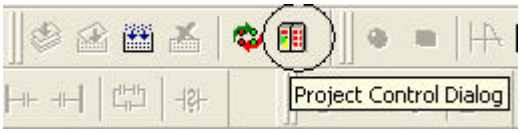
c. Open the Global Variables worksheet from the Project tree and make the following changes to map Inputs and Outputs:

	Name	Type	Usage	Description	Address
	⊟ **Default**				
	Inputs	INT	VAR_GLOBAL		%IW0
	Cycle_Count	INT	VAR_GLOBAL		%QW0

d. Select the “Make” Icon from the tool bar.

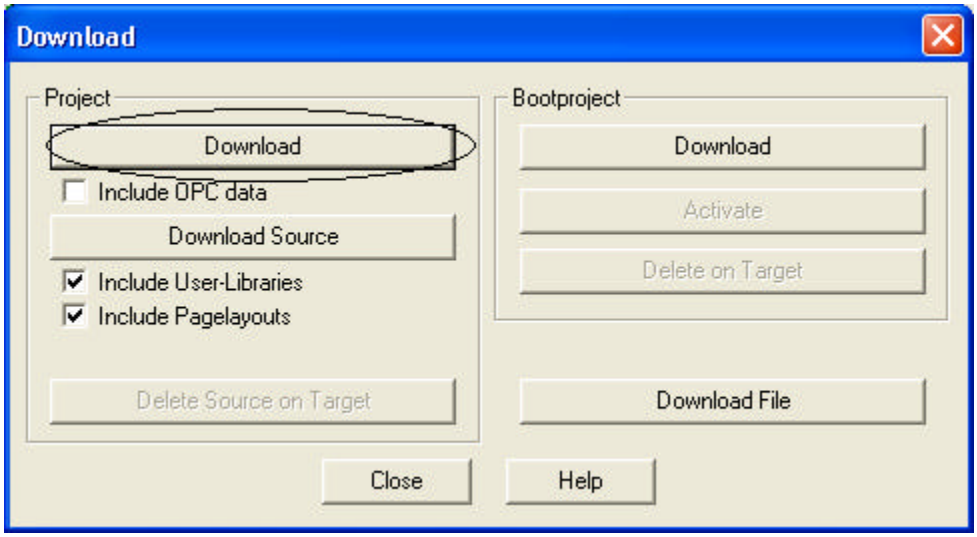


e. Once the build is complete, select the “Project Control Dialog” Icon from the tool bar. This will also start the simulator program.



f. Select download from the control dialog.

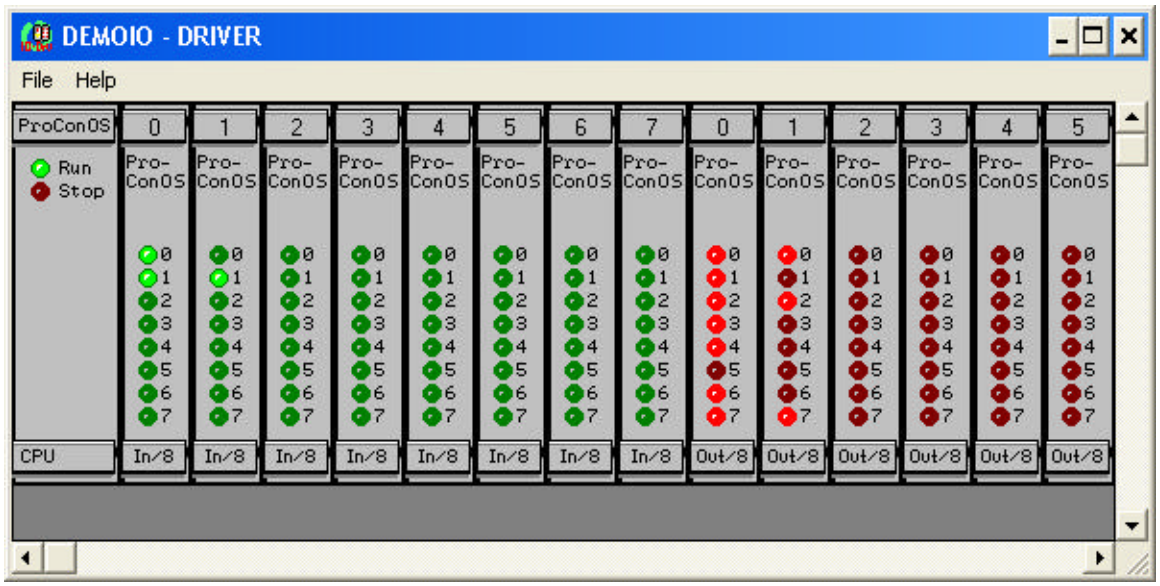
g. Select download in the project section



h. Now select “Cold” from the control dialog to start the simulator running.

i. Maximize the DEMOIO-Driver from the taskbar to see the simulated digital I/O.

j. The simulated Outputs can be seen, and, clicking on the LED's in the simulator can toggle the Inputs.



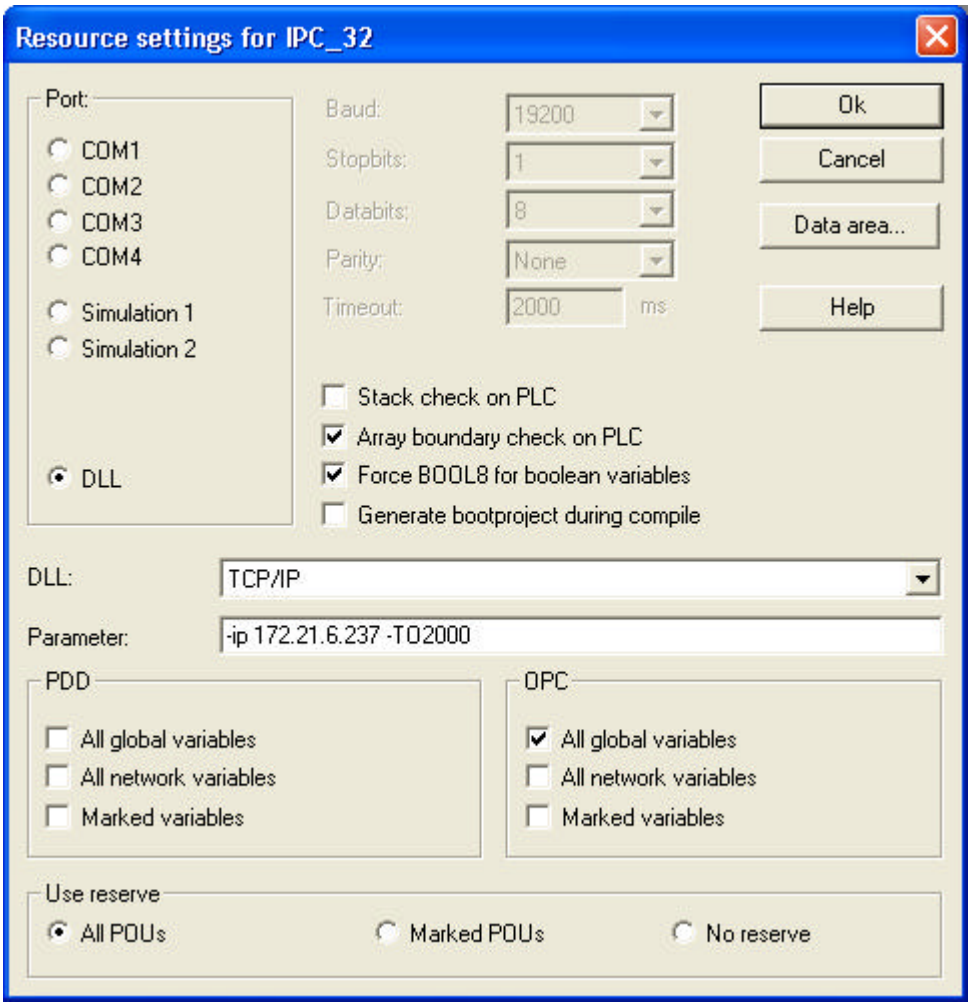
k. The I/O can be seen online by clicking the debug on/off Icon and selecting the global variable list.

Name	Online value	Type	Usage	Description	Address
Default					
Inputs	515	INT	VAR_GLOBAL		%MW0
Cycle_Count	-20838	INT	VAR_GLOBAL		%QW0
System Variables					
PLCMODE_ON	FALSE	BOOL	VAR_GLOBAL	TRUE : current P...	%MX1.0.0
PLCMODE_RUN	TRUE	BOOL	VAR_GLOBAL	TRUE : current P...	%MX1.0.1
PLCMODE_STOP	FALSE	BOOL	VAR_GLOBAL	TRUE : current P...	%MX1.0.2
PLCMODE_HALT	FALSE	BOOL	VAR_GLOBAL	TRUE : current P...	%MX1.0.3
PLCDEBUG_BPSET	FALSE	BOOL	VAR_GLOBAL	TRUE : one or m...	%MX1.1.4
PLCDEBUG_FORCE	FALSE	BOOL	VAR_GLOBAL	TRUE : current P...	%MX1.2.0
PLCDEBUG_POWERFLOW	FALSE	BOOL	VAR_GLOBAL	TRUE : current P...	%MX1.2.3
PLC_TICKS_PER_SEC	100	INT	VAR_GLOBAL		%MW1.44
PLC_SYS_TICK_CNT	123476	DINT	VAR_GLOBAL		%MD1.52
PLC_TASK_AVAILABLE	16	INT	VAR_GLOBAL		%MW1.1000
PLC_TASK_DEFINED	1	INT	VAR_GLOBAL		%MW1.1002

Create a sample application for a live target:

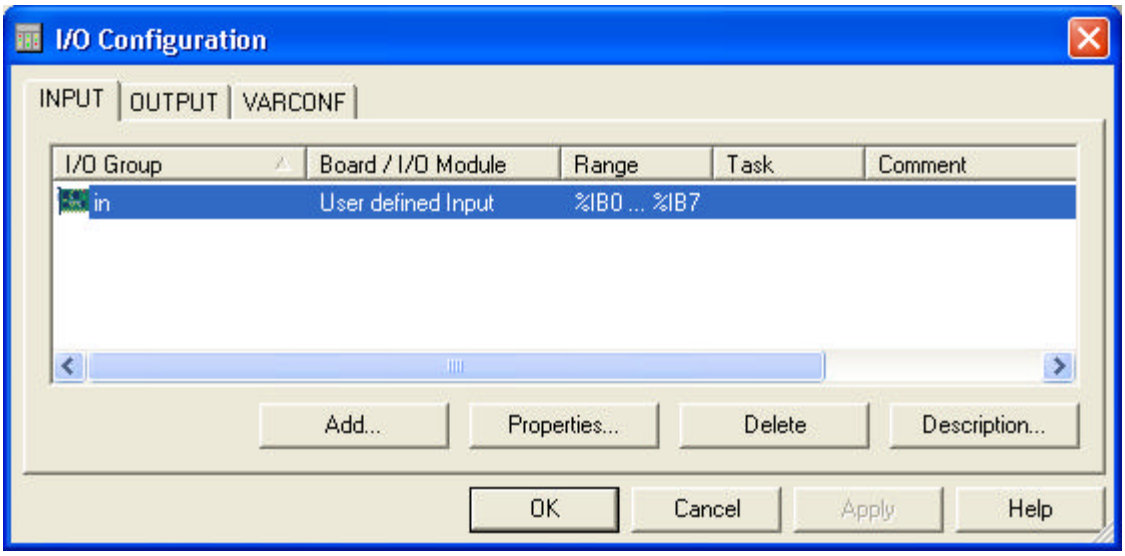
To make the above application for a live target some things must be changed.

a. The connection port must be changed to download to a different target other than the simulator. To do this right click on “resource” in the project tree, and select “settings”. This will bring up the resource settings dialog box which allows the user to input an IP address as shown in the picture below:

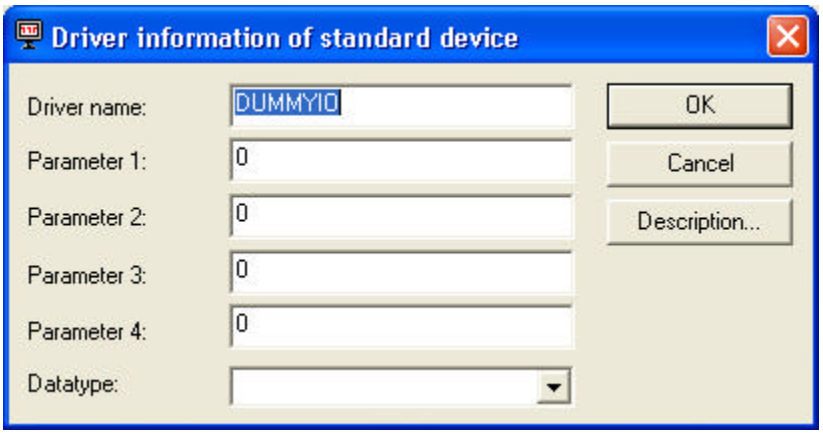


The connection now changes from Simulation to DLL and the parameter for the DLL is IP address with a Time out value. A connection can also be made with a COM port and a crossover cable. For Windows CE targets, altering the file \harddisk\proconos\proconos.ini will enable the port. This file contains a line that will allow using the COM port. The line has a semi-colon in front of it. Remove the semi-colon to enable that line:
;V24Port1=1,19200

b. The driver must also be changed. If there is currently no real I/O configured then the I/O can be changed from DEMOIO to DUMMYIO. This can be done by double clicking the I/O configuration from the project tree. This will bring up the I/O configuration dialog.



Click on the properties button to see that the User defined Input is selected. Now click on driver parameter and change the DEMOIO to DUMMYIO.



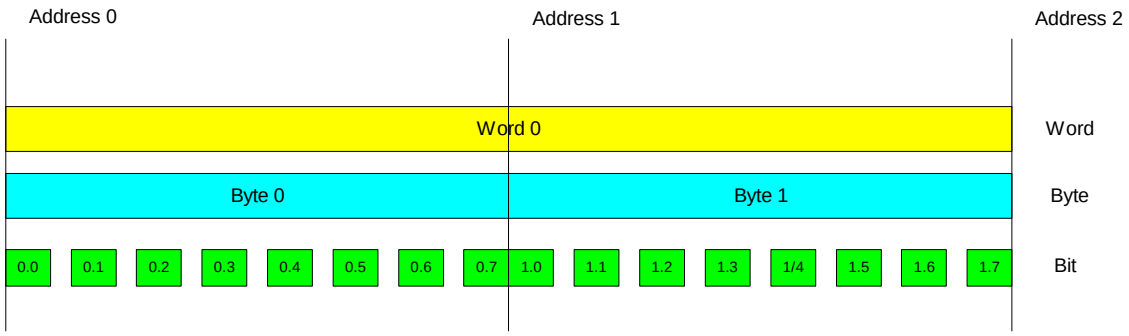
Now do the same to the output, rebuild and download to the live target.

If Advantech I/O is being used then see the section on Advantech Adam I/O or select the description button when the Advantech I/O is selected and the online help will be available.

Mapping I/O

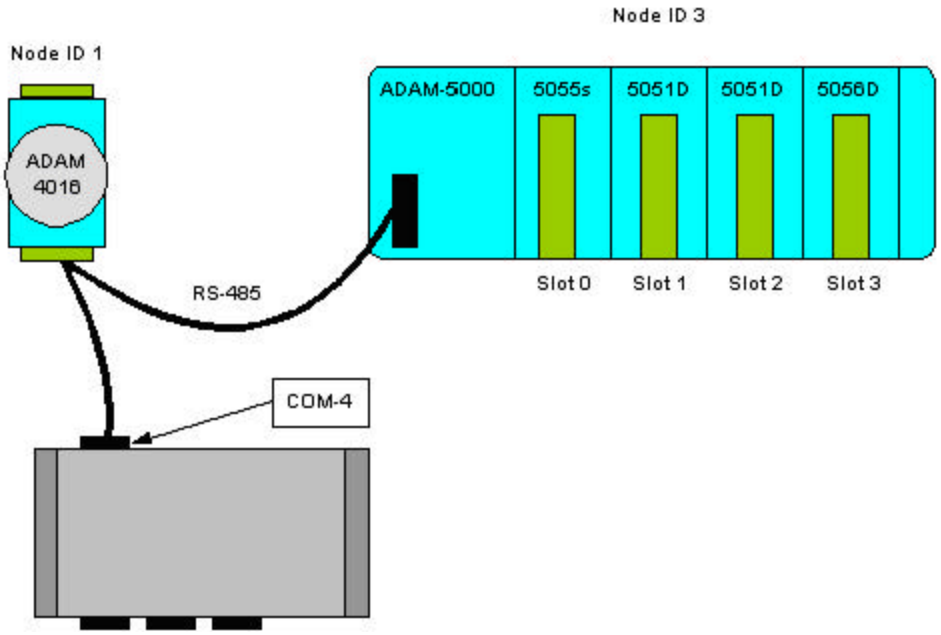
To understand how to map I/O, the I/O space of Proconos must first be understood. The I/O space addresses are based on Bytes. Therefore, whenever a mapped variable is referenced by an address, it is bases on the starting Byte address of the variable. Below is a simple layout of how some variable lengths overlay in the same I/O memory space.

There are separate I/O spaces provided for Inputs and Outputs. Therefore, each will start with zero.



To help illustrate mapping of variables, below is an example of how to map modules from an ADAM-485 network.
This network consists of:

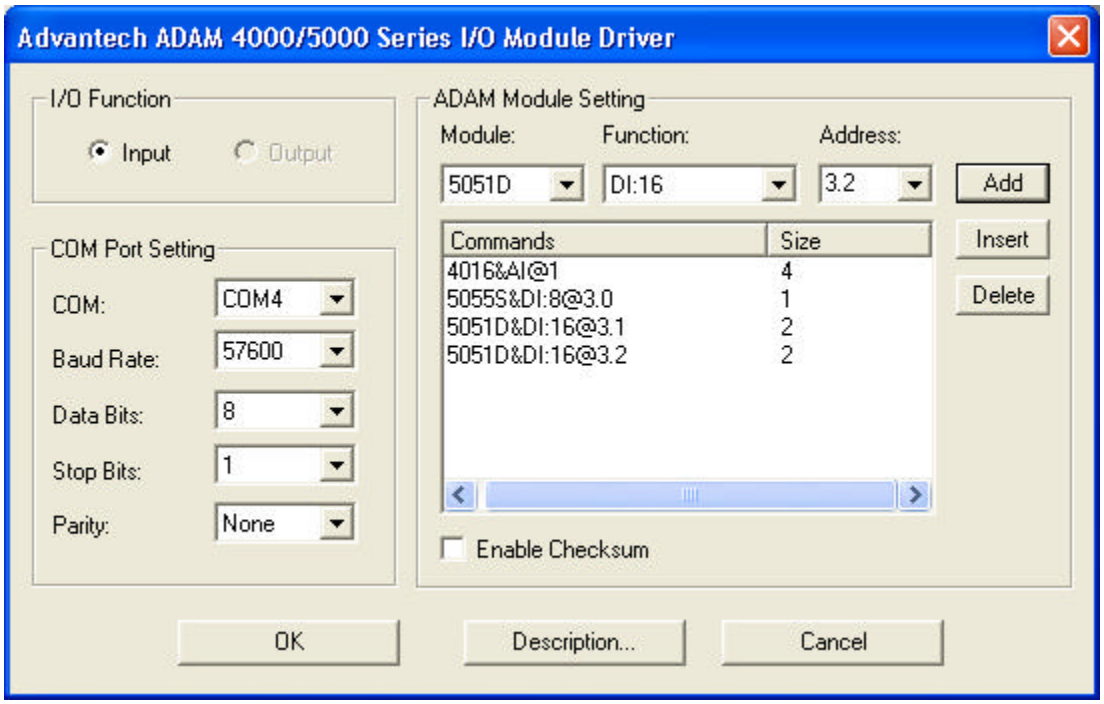
1 Analog Input module
3 Digital input modules
1 Digital output module.



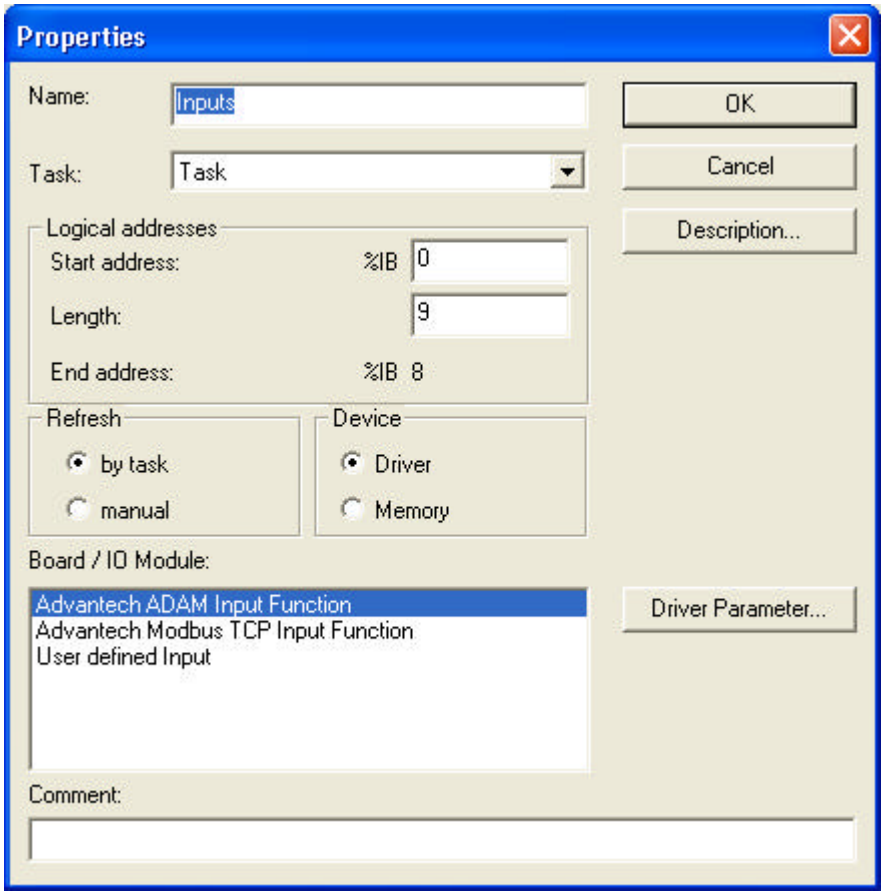
With this network, there is the following Modules configured:

Name	Node ID (address)	Type of I/O	Num of I/O	Num of Bytes	Data Type
Adam-4016	1	Analog Input	1	4	REAL
Adam-5055S	3.0	Digital In 8	8	1	BYTE
Adam-5051D	3.1	Digital In 16	16	2	WORD
Adam-5051D	3.2	Digital In 16	16	2	WORD
Adam-5056D	3.3	Digital Out 16	16	2	WORD

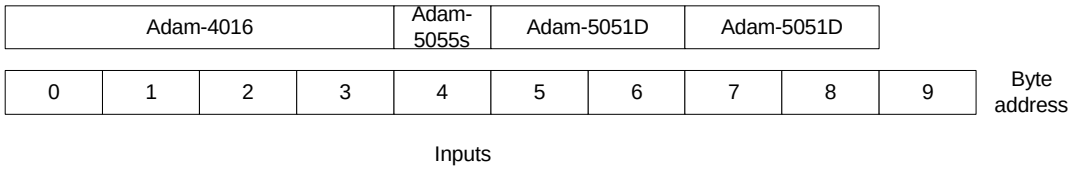
The mapping of the network to the I/O space is DIRECTLY RELATED to the order of appearance in the driver sheet. Below is an example of how the Inputs are set up in Multiprog.



When OK is pressed the dialog returns a byte Length of 9 (byte addresses 0-8).



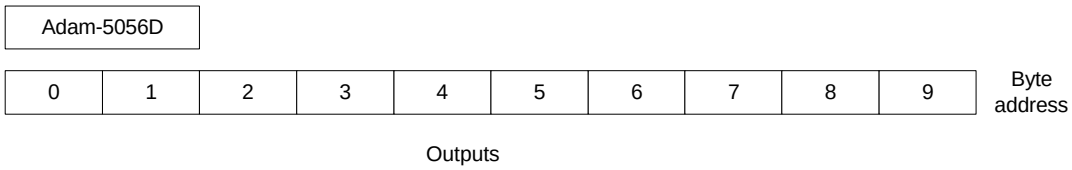
When matched to the I/O address, the I/O will line up like the example below.



When mapped as variables it will look similar to the example below.

Name	Type	Usage	Description	Address
Default				
Adam_4016_Addr1	REAL	VAR_GLOBAL	Analog Input	%ID0
Adam5055S_Add3_0	BYTE	VAR_GLOBAL	8 Digital Inputs	%IB4
Adam5051D_Add3_1	WORD	VAR_GLOBAL	16 Digital Inputs	%IW5
Adam5051D_Add3_2	WORD	VAR_GLOBAL	16 Digital Inputs	%IW7
Adam5056D_Add3_3	WORD	VAR_GLOBAL	16 Digital Outputs	%QW0

The Output I/O will appear as follows:



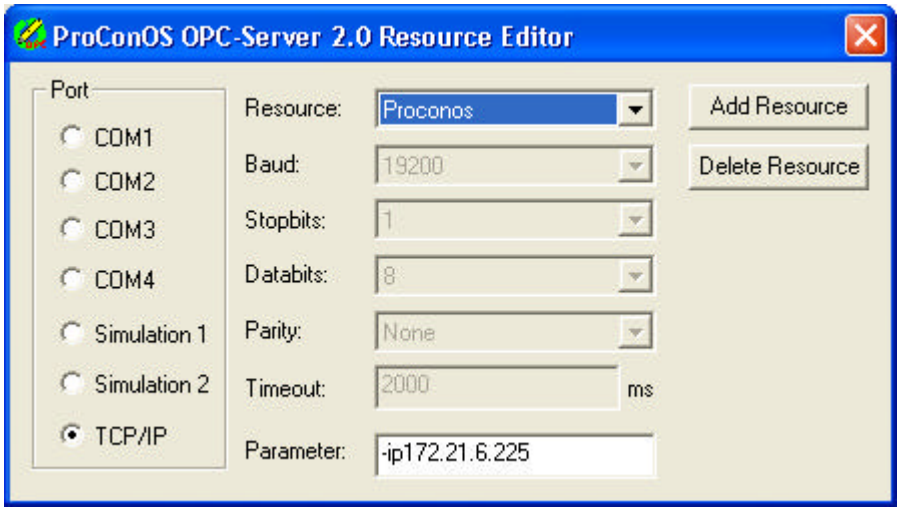
8. OPC server.

KW software provides an OPC server for communication between the target system and an OPC client (such as an HMI software). If OPC communication is desired, then, **the OPC server must run on the machine where the OPC communication is desired**. If the OPC client is on the target machine, then the OPC server for that target must be used. If the OPC client is on another machine than the target, then that is where the OPC server must reside. The server is set up the same for any of the supported platforms. Current supported platforms include Win32 systems such as Win2000 or XP, Windows CE.NET running on Strong-Arm or X86 platforms.

Development:

When developing an HMI application, there must be an OPC server installed on the development machine for the HMI client to connect to. KW provides an OPC server for the development machine that will run in full demo mode for 60 minutes. This is considered enough time for development and testing of the application.

a. To set up the OPC server, select Start\Programs\KW-Software\OPC Resource Editor .
b. Select Add Resource and give the resource name Proconos. This is the default resource name on the target system.
c. Check to see if there are any other resources and delete them if necessary. The OPC server will attempt to connect to ALL resources that are configured.
d. Put the IP address of the target system in the parameter box as shown. This will guide the OPC server to that IP address to get tags of the "running" target. If communications through a COM port is desired then set those settings as needed.
e. Once the Resource is configured then the client only needs to connect to the server PCOS.OPC.20.



Target:

The Advantech WinCE.NET targets that run KW will come with the resource already configured as "Proconos" and the IP address of 127.0.0.1. Additional resources can be configured in the same manor as above. If this is the desired setup then no further set up is necessary.

Connecting to the OPC server:

As mentioned above to connect to the server the client should connect to the name PCOS.OPC.20. **If the runtime is not running, or if there are no tags configured for the OPC server by Multiprog, then no tags will be available when connecting to the server.**

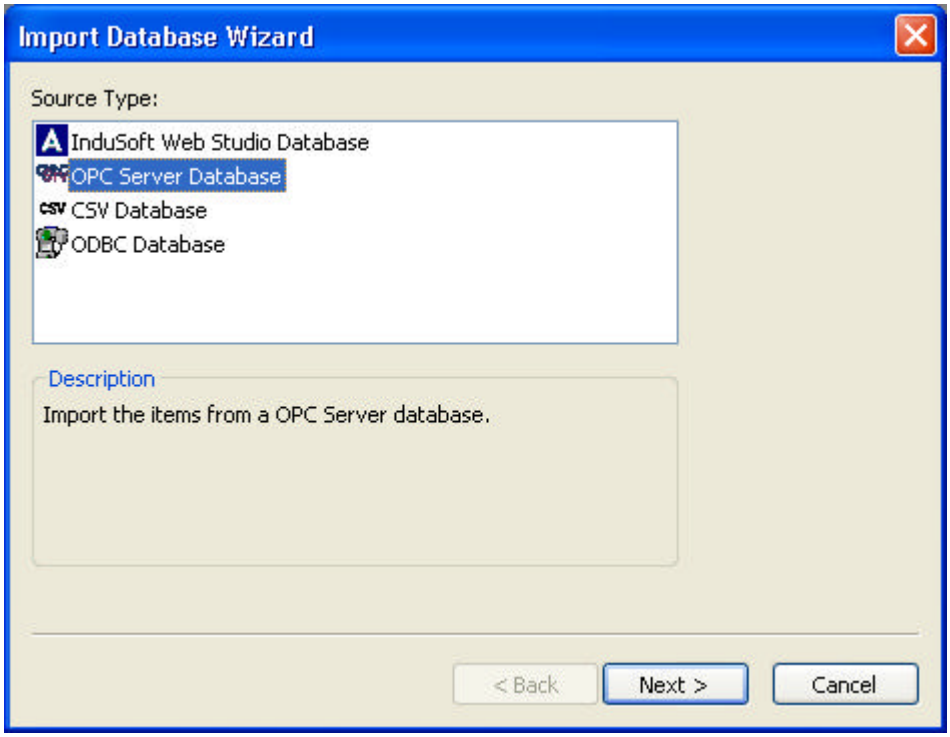
9. Advantech Studio

This example is done using Advantech Studio version 5.1. Advantech **ECS** will also work with older versions but some of the techniques mentioned here will not be available.

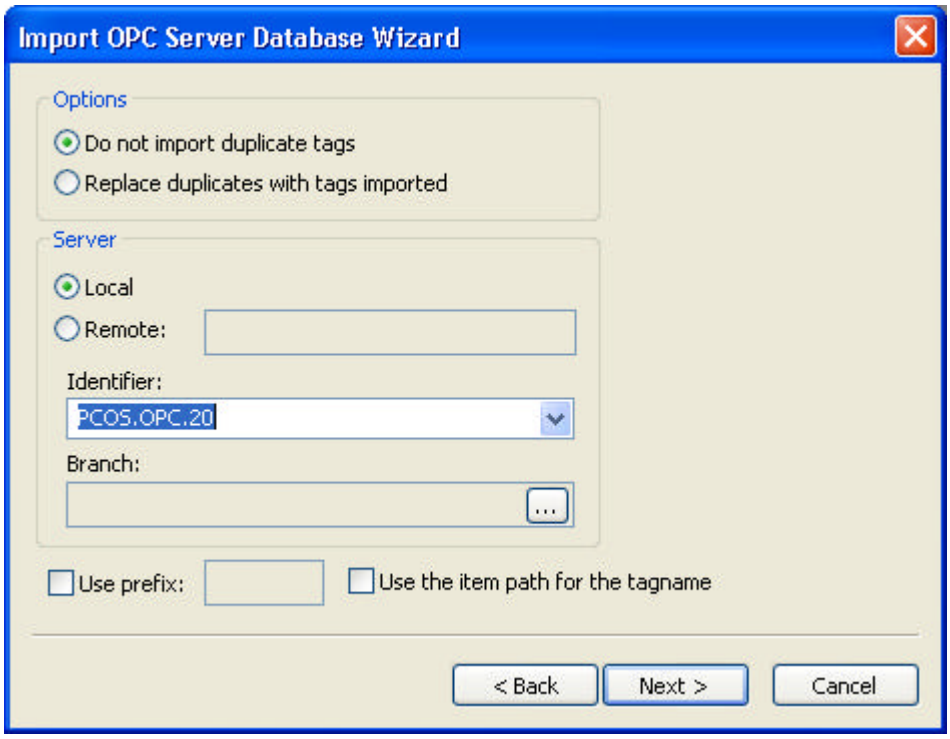
Create a Sample application.

The connection between AStudio and the ProConOS engine will be done with OPC connectivity. AStudio provides an easy way to import one or more tags at once, as well as being able to browse individual tags at a later time. It is assumed here that the end user has already configured the OPC server and done a test connection from the development machine to the target machine.

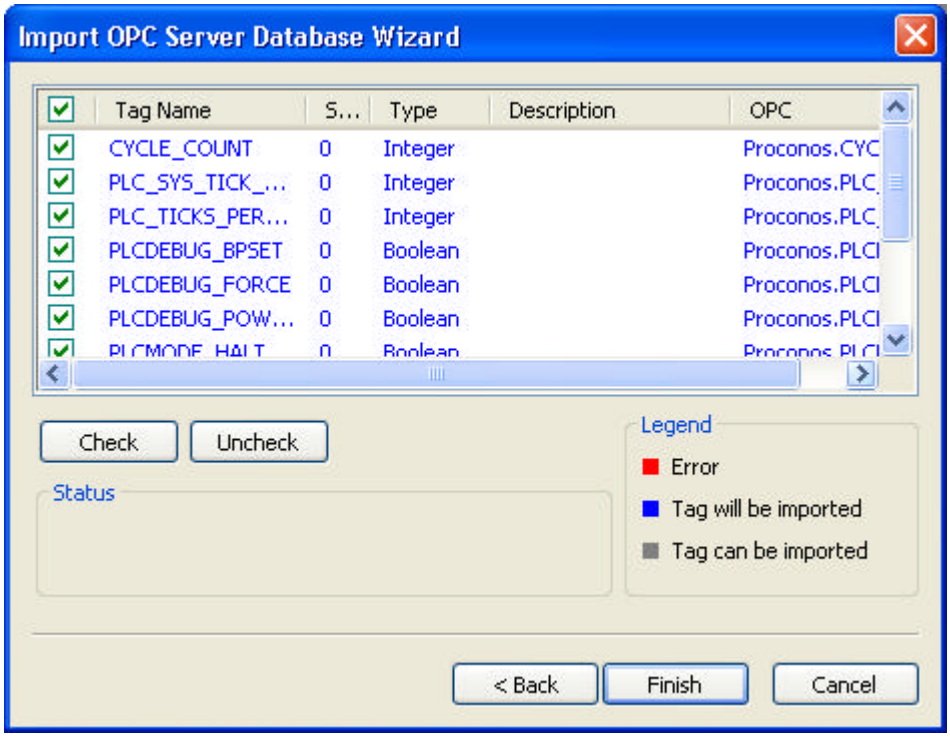
a. From the file menu select “Import”. From here select OPC Server Database and click next.



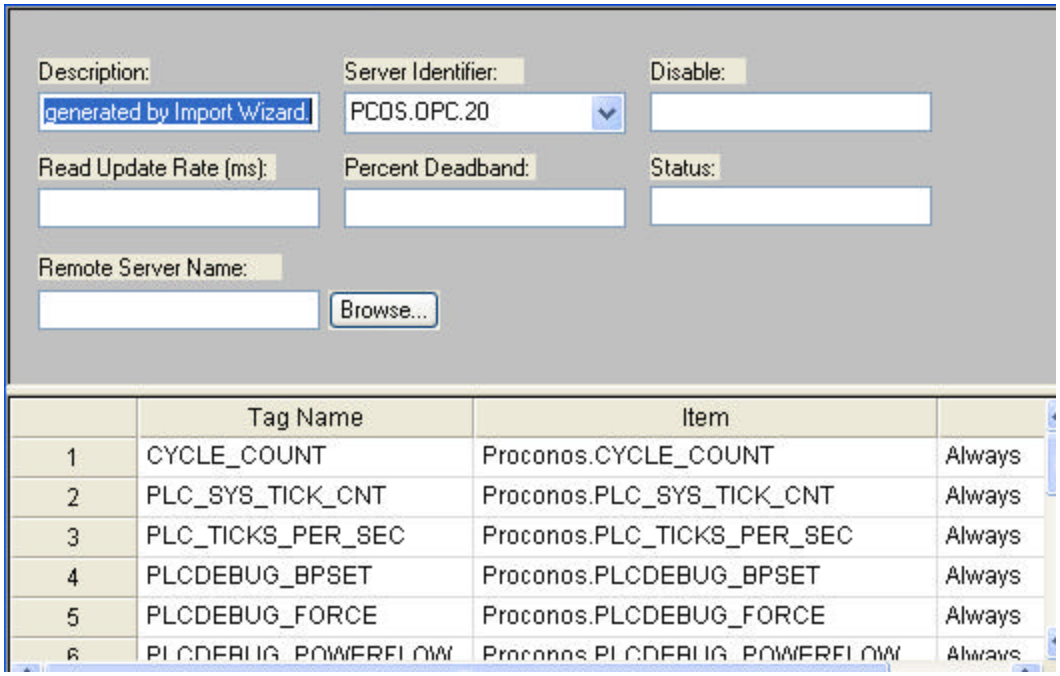
b. In the Identifier box select PCOS.OPC.20 and click next.



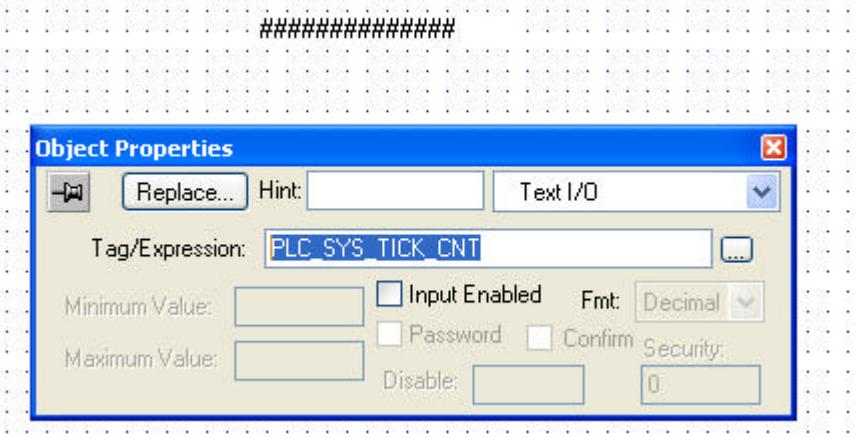
c. Now all of the tags have been read from the OPC server and are ready to be imported. Tags that are not required can be unchecked and they will not be imported.



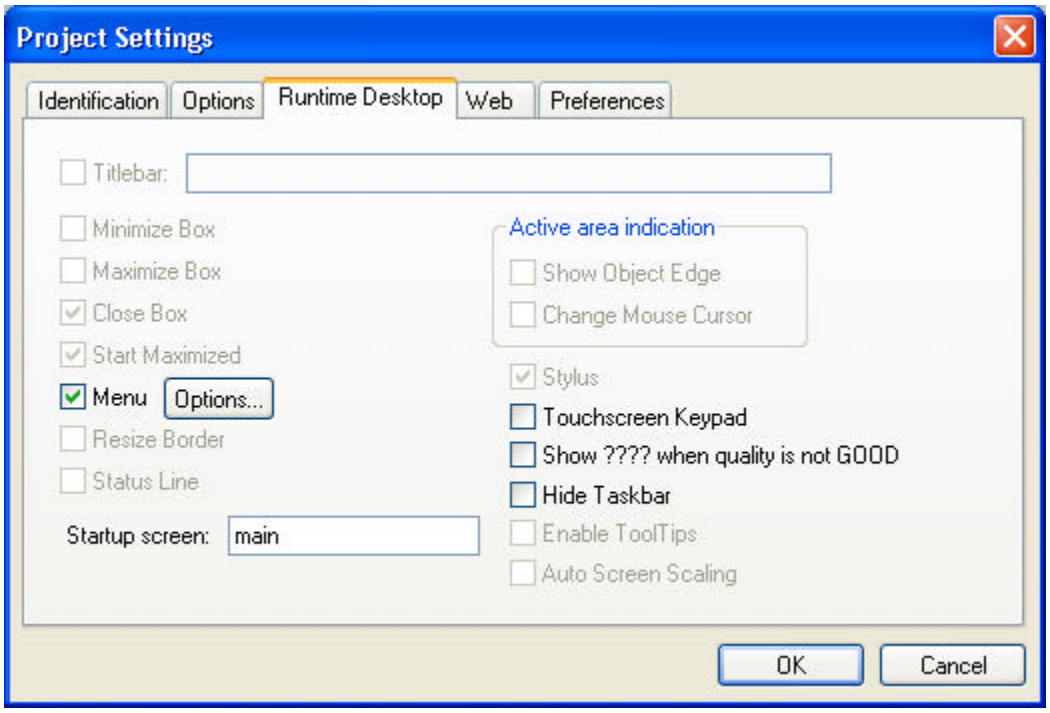
d. Once the import is complete, an OPC Driver sheet will be created and all selected tags will be mapped with matching tag names created in the “Application Tags” database. If more than one project is being imported a prefix can be added to the tag names.



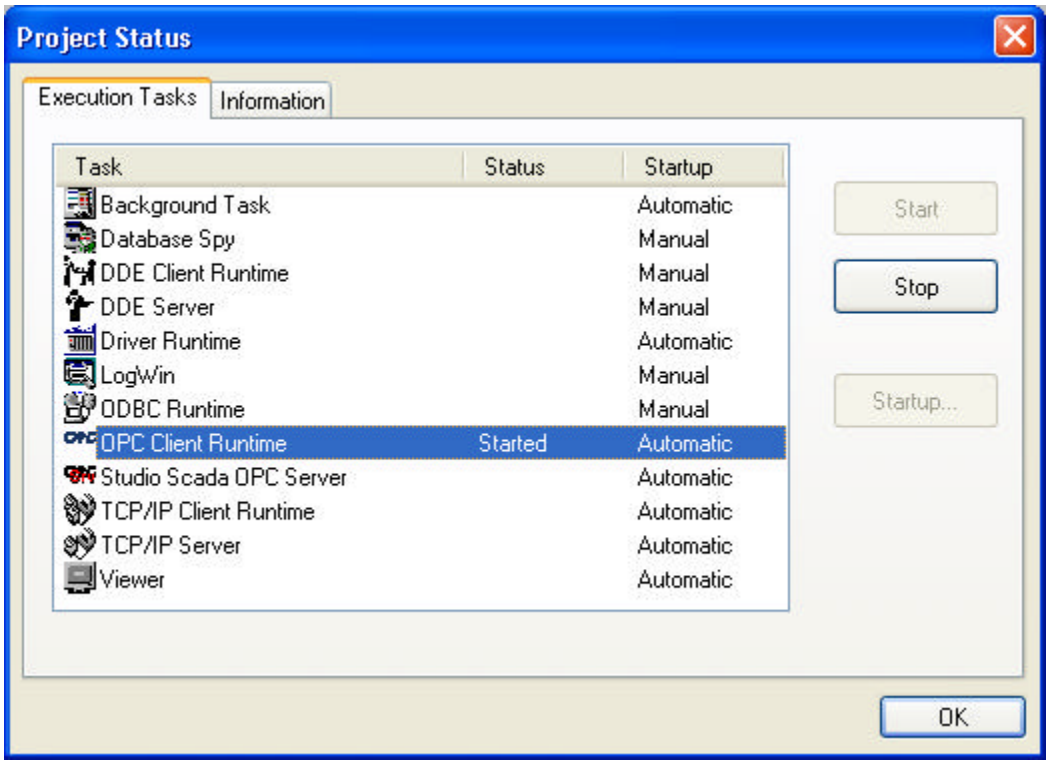
e. Now, create a screen, add a text item with the TextI/O property added to it and map this to the PLC_SYS_TICK_CNT variable. This will display the PLC tick counter.



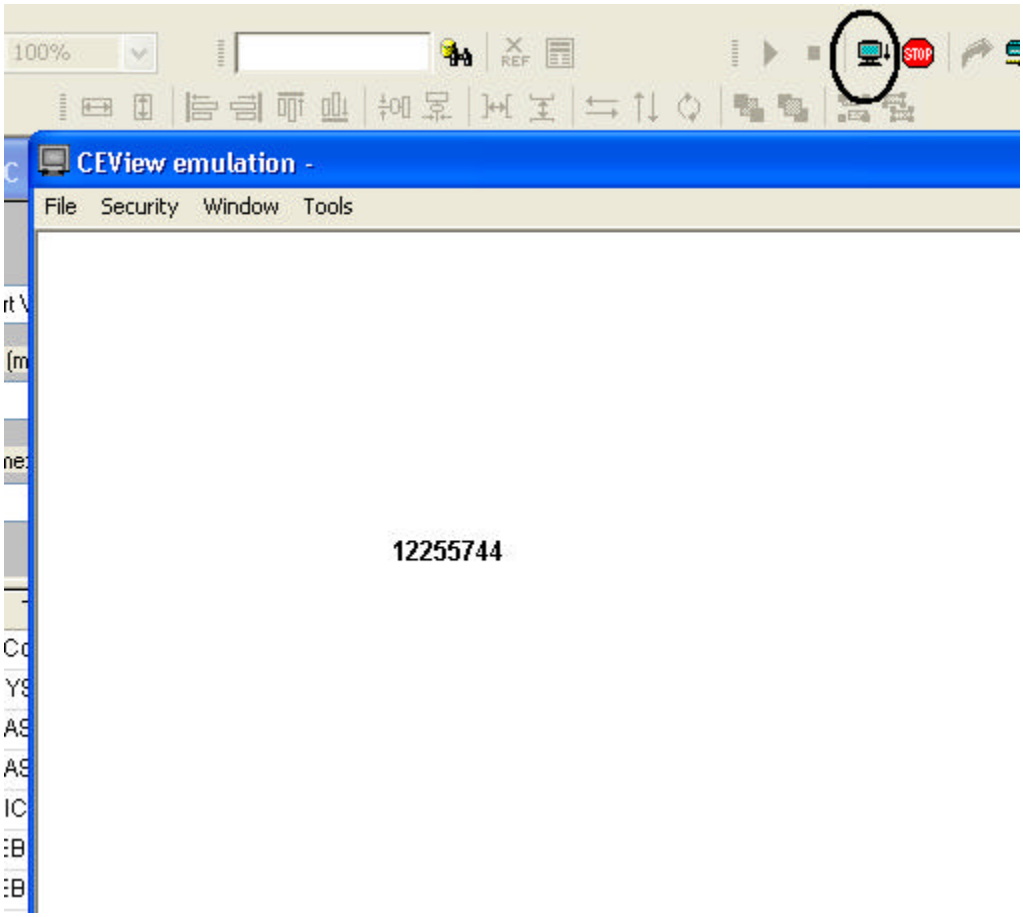
f. Close the newly created Screen and save it as main.scr.
Go to Project/Settings Runtime Desktop, and set the startup screen as main



g. To test this simple application from the development machine, the next step is to go to Project/Status, Select OPC Client Runtime and click the start button.



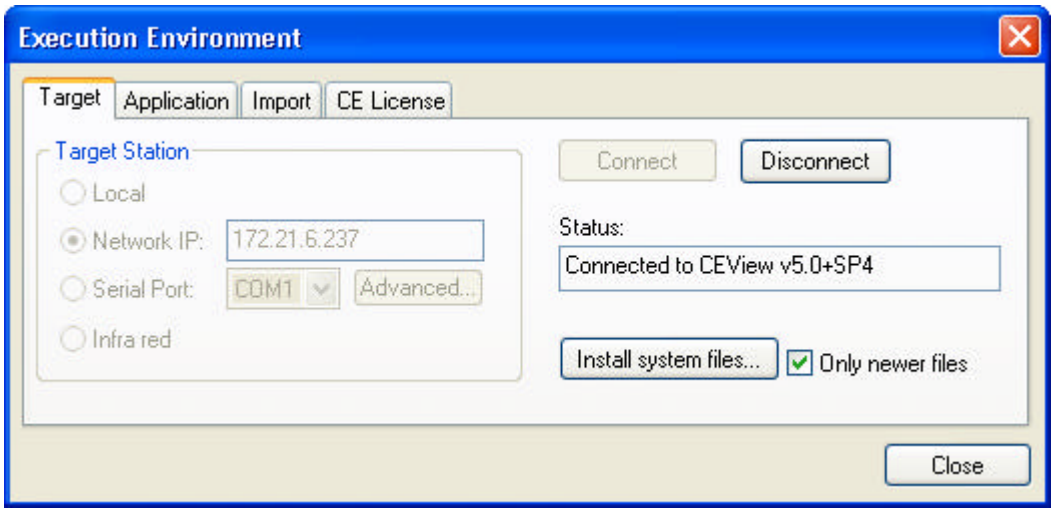
h. This should automatically start up the KW OPC server installed on the development machine that was tested and configured in the previous chapter. Now select the run Icon from the tool bar to run the application locally.



A value should appear on the emulation screen and display the counting PLC system tick.

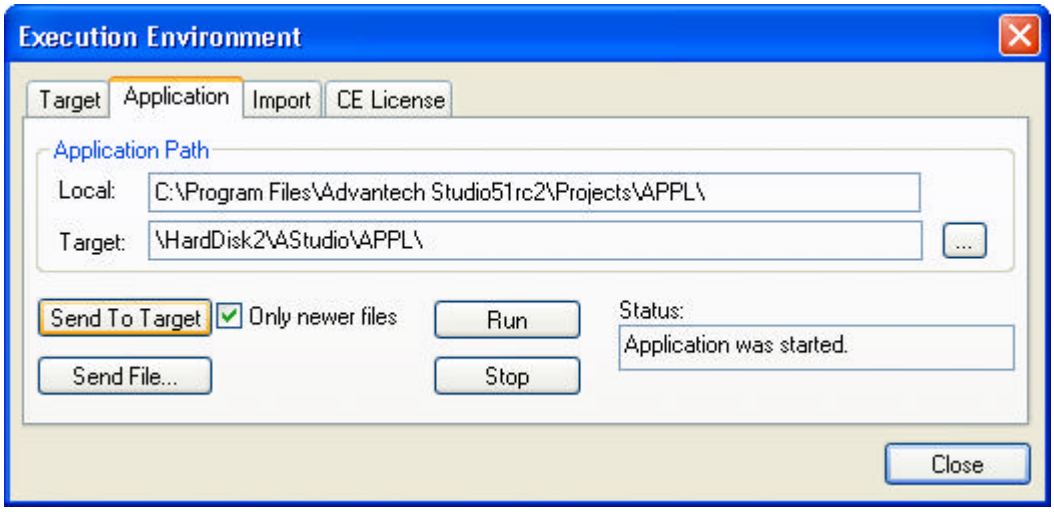
i. The next step is to run the application on the remote target. The CEView emulation must be stopped first. Click the stop sign icon, located next to the “Run Application” icon shown in the picture above.

j. Now select Project/Execution environment to prepare to connect to the remote machine. Change the Target Station to Network IP: and insert the IP address of the target. Now click the connect button. This will connect to the remote target. If it does not connect, make sure that the IP address is correct and that the remote agent is running on the target.

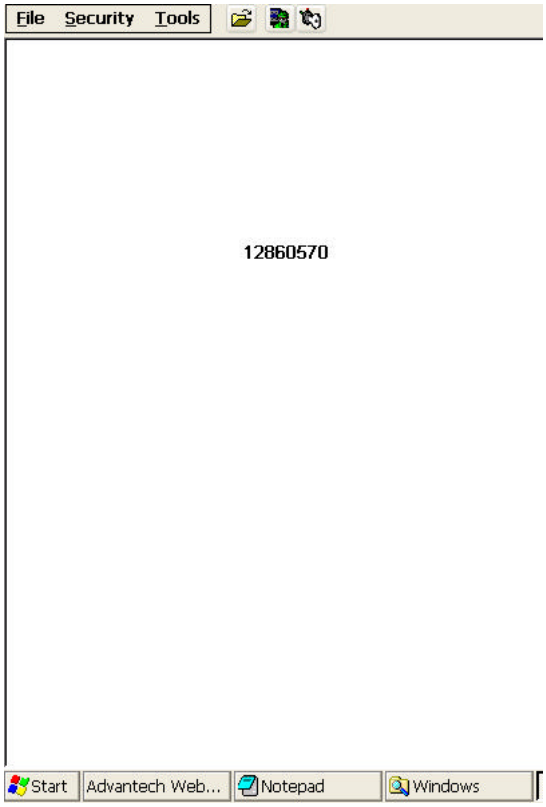


k. If it does not connect, make sure that the IP address is correct and that the remote agent is running on the target.

l. After successfully connecting to the target, select the application tab and click the Stop button to stop the application that may be running. Next, click the “Send To Target” button to send the application to the target and then click the run button.



m. Now AStudio should be running and displaying the PLC Tick count on the target system.

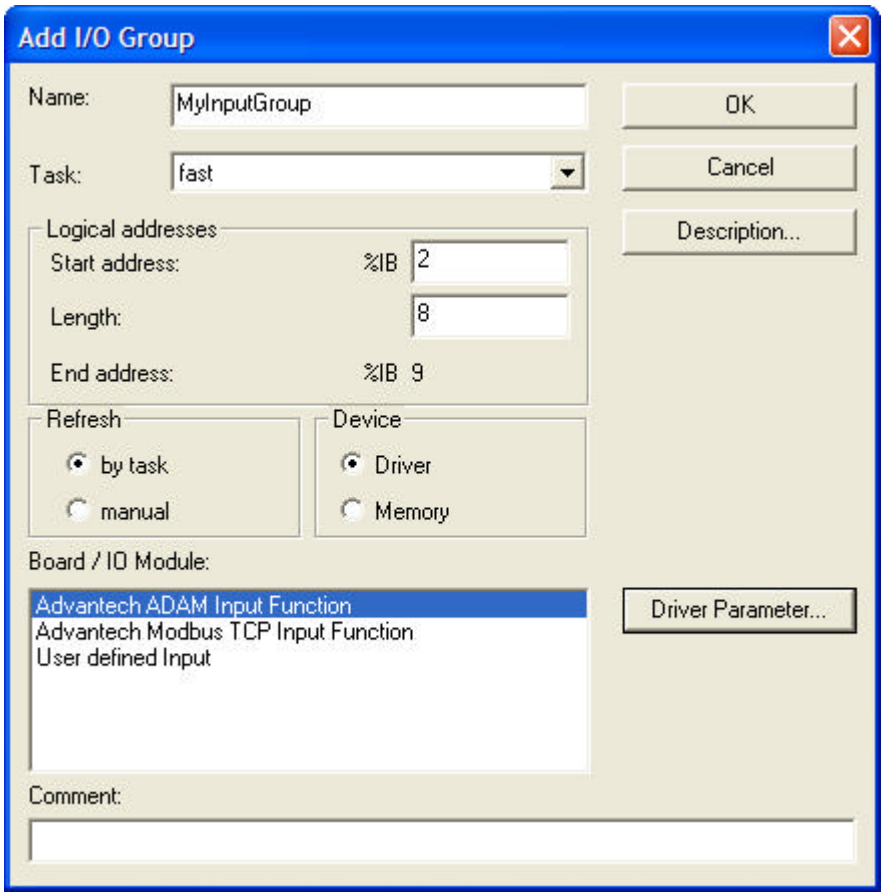


For further help with AStudio, see the AStudio online documentation.

10. Advantech Adam I/O

Adam-485 Serial driver

To use the ADAM Serial Driver select “Advantech Adam Input Function” Or “Advantech Adam Output Function” from the driver selection box depending on whether or not inputs or outputs are being configured. Select the task to drive the I/O group.



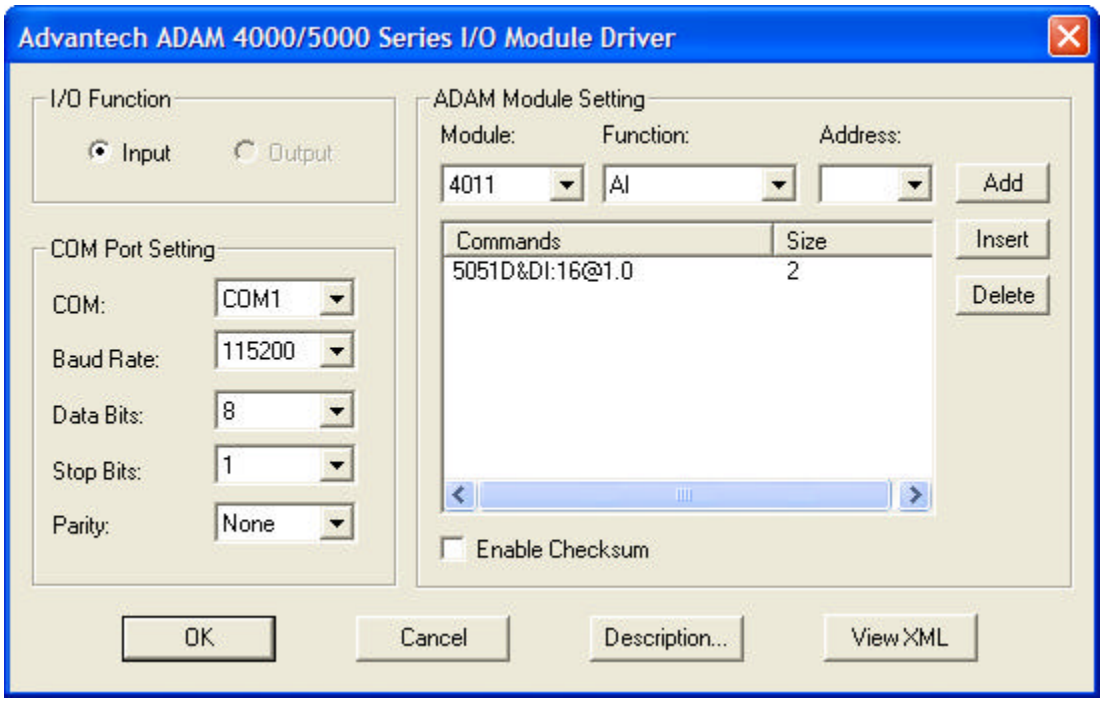
NOTE: This is a very important, the same task for both input and output I/O groups must be used, AND, only one Input and one Output I/O group per Com Port can be configured.

If this method is not followed, it is a possibility that a higher priority task will interrupt a lower priority task and therefore interrupt communication and cause loss of data.

Driver dialog

Once the configuration dialog box is open, simply select a Module, the function to implement, the input Address, and click the Add button.
A complete function list matrix is provided below.

Since the order of the commands is important, there is an insert button supplied. Highlight the command to insert in front of and click the insert button to add a command. The Add button will put the command at the bottom of the list.



Module address

For Adam 4000 Modules the Address is <module address>.<I/O point>
For Adam 5000 Modules the Address is <module address>.<slot>.<I/O point>

Note: I/O Points AND slot numbers start with zero.

Some examples would be:

Module	Function	Module Addr.	I/O Point	Command
4011	DI	2	Only one input	4011&DI@2
4050	DO:8	12	All 8 input points	4050&DO:8@12
4024	AO	22	Second output	4024&AO@22.1
5017H	AI:8	30	First slot, all 8 inputs	5017H&AI:8@30.0
5024	AO	5	Last slot, second output	5024&AO@5.3.1
5000	WD	4	NONE	5000&WD@4

Mapping I/O to Variables

Once done configuring the modules it is important to understand the correlation between the order of the configured modules and the mapping of the corresponding variables.

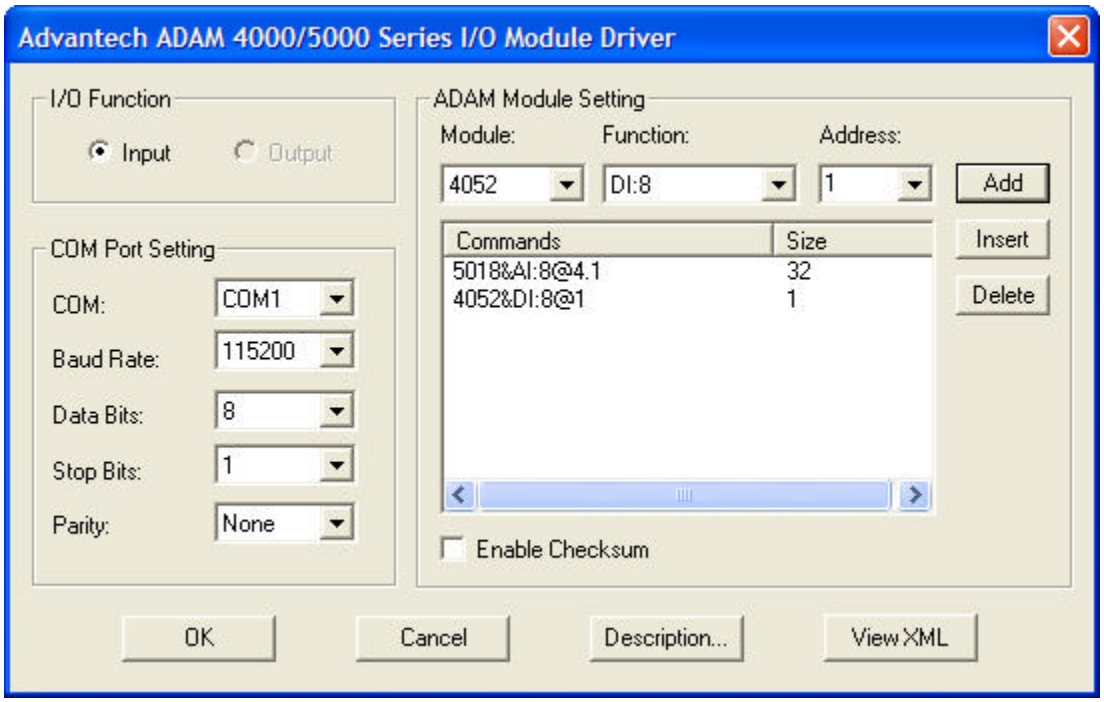
Each command will use a specified number of bytes in the global variable list and it is up to the user to map the variables properly. The number of bytes is listed in the configuration dialog, and is also shown on the function support matrix below.

As the example shows below, the order of functions are directly correlated to the I/O address space. The example shows two input modules configured.

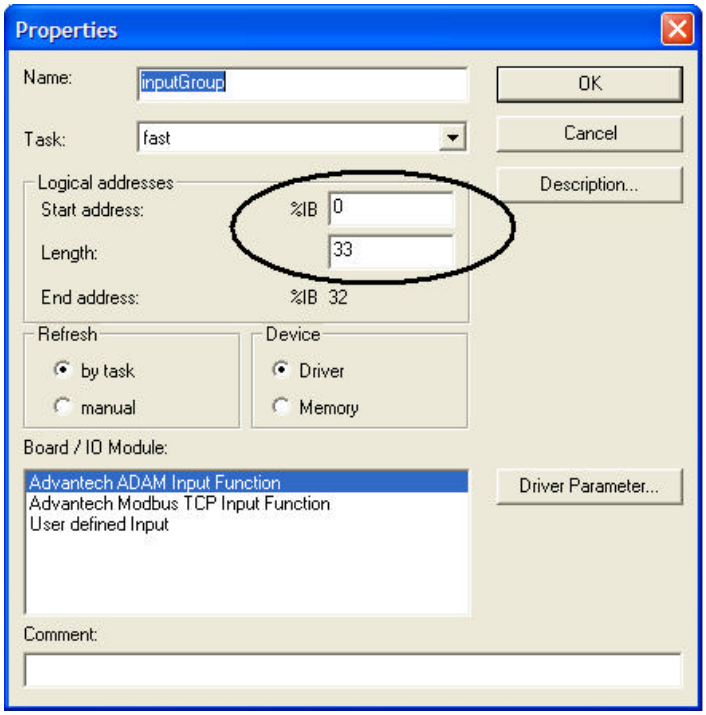
Variable In MWT	Address	Variable type	Function	Number of Bytes
AnalogIn0	ID0	REAL	5018&AI:8@4.1	32
AnalogIn1	ID4	REAL		
AnalogIn2	ID8	REAL		
AnalogIn3	ID12	REAL		
AnalogIn4	ID16	REAL		
AnalogIn5	ID20	REAL		
AnalogIn6	ID24	REAL		
AnalogIn7	ID28	REAL		
All8DigitalInputs	IB32	BYTE	4052&DI:8@1	1
DigitalInput0	IX32.0	BOOL		
DigitalInput1	IX32.1	BOOL		
DigitalInput2	IX32.2	BOOL		
DigitalInput3	IX32.3	BOOL		
DigitalInput4	IX32.4	BOOL		
DigitalInput5	IX32.5	BOOL		
DigitalInput6	IX32.6	BOOL		
DigitalInput7	IX32.7	BOOL		

The address of the ADAM module has no affect on the address of the variables. It is up to the location of the command in the command list. This example also shows how variables can be double mapped to the same I/O space. In the example one can look at all digitals in one variable or as individual bool's

The following images show how this setup will look in Multiprog.



Notice that the total number of bytes has been returned to the length. The end address is calculated from the start address and the length. The user can change the start address to whatever desired, this allows group mapping. For example this could have started at 100.



This is how the variables for this setup appear in the global variable list.

	Name	Type	Usage	Description	Address	Init	Retain	PDD	OPC
	⊟ **ADAM Input modules**								
	AnalogIn0	REAL	VAR_GLOBAL		%ID0		☐	☐	☐
	AnalogIn1	REAL	VAR_GLOBAL		%ID4		☐	☐	☐
	AnalogIn2	REAL	VAR_GLOBAL		%ID8		☐	☐	☐
	AnalogIn3	REAL	VAR_GLOBAL		%ID12		☐	☐	☐
	AnalogIn4	REAL	VAR_GLOBAL		%ID16		☐	☐	☐
	AnalogIn5	REAL	VAR_GLOBAL		%ID20		☐	☐	☐
	AnalogIn6	REAL	VAR_GLOBAL		%ID24		☐	☐	☐
	AnalogIn7	REAL	VAR_GLOBAL		%ID28		☐	☐	☐
	All8Digitals	BYTE	VAR_GLOBAL		%IB32		☐	☐	☐
	DigitalInput0	BOOL	VAR_GLOBAL		%IB32.0		☐	☐	☐
	DigitalInput1	BOOL	VAR_GLOBAL		%IB32.1		☐	☐	☐
	DigitalInput2	BOOL	VAR_GLOBAL		%IB32.2		☐	☐	☐
	DigitalInput3	BOOL	VAR_GLOBAL		%IB32.3		☐	☐	☐
	DigitalInput4	BOOL	VAR_GLOBAL		%IB32.4		☐	☐	☐
	DigitalInput5	BOOL	VAR_GLOBAL		%IB32.5		☐	☐	☐
	DigitalInput6	BOOL	VAR_GLOBAL		%IB32.6		☐	☐	☐
	DigitalInput7	BOOL	VAR_GLOBAL		%IB32.7		☐	☐	☐

Also, the variable names can just as easily be named “PressureValve1” “Oil_Level_Tank_5” etc. Variable names can be up to 31 characters in length.

This example shows input variables being mapped to the global input address space. The output address space is a separate address space and also starts at zero.

Supported function list

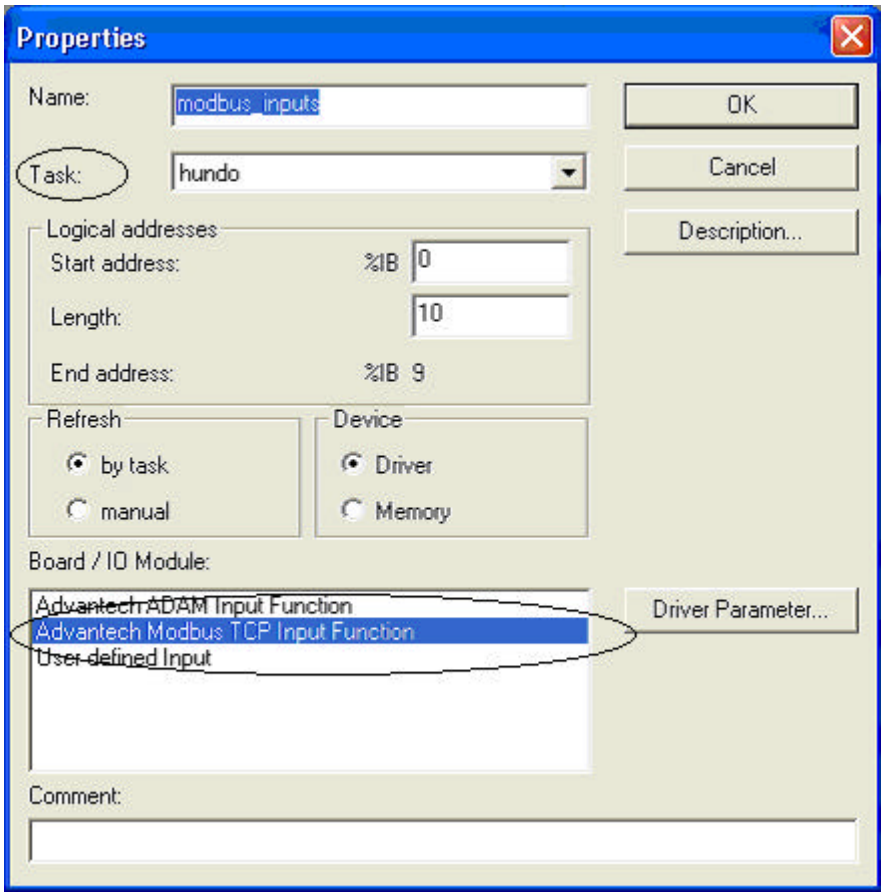
ADAM-485 driver function list																
	Input Functions							Output Functions								
Number Of Bytes	4	12	32	1	1	2	4	1	1	1	2	4	1	1	1
Module	AI	AI:3	AI:8	DI	DI:8	DI:16	CNT:R	DO	DO:2	DO:8	DO:16	AO	CNT:CLR	CNT:ON	CNT:OFF
4011/4011D	x			x			x	x					x	x	x
4012	x			x			x	x					x	x	x
4013	x														
4016	x						x	x	x				x	x	x
4017			x												
4018			x												
4018M			x												
4019			x												
4021												x			
4024												x			
4050					x					x					
4051						x									
4052					x										
4053						x									
4055					x					x					
4060										x					
4080/4080D							x						x	x	x
5013		x													
5017/5017H			x												
5018			x												
5024												x			
5050						x					x				
5051 , D, S						x									
5052					x										
5055S					x					x					
5056 , D, S, SO											x				
5060										x					
5068										x					
5080							x						x	x	x

Modbus /TCP driver

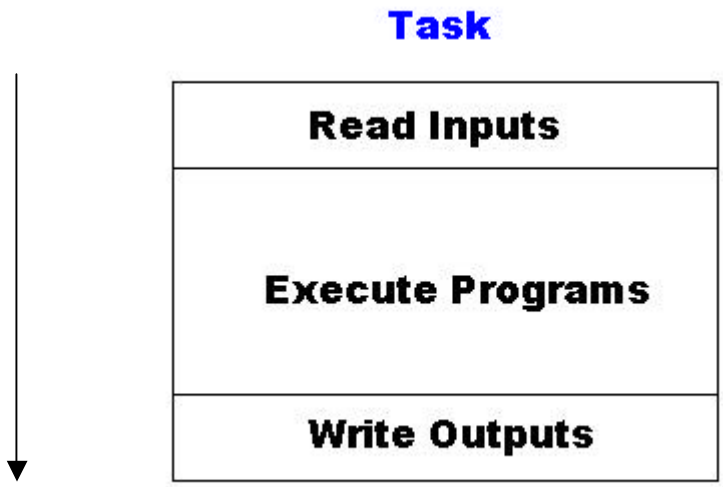
The Modbus/TCP driver can use Advantech Modbus/TCP I/O but will also support third party Modbus/TCP I/O.

Add an I/O Group

To use the Modbus/TCP driver select, “Advantech Modbus TCP Input Function” or “Advantech Modbus TCP Output Function” from the Board / IO Module from the driver selection box.



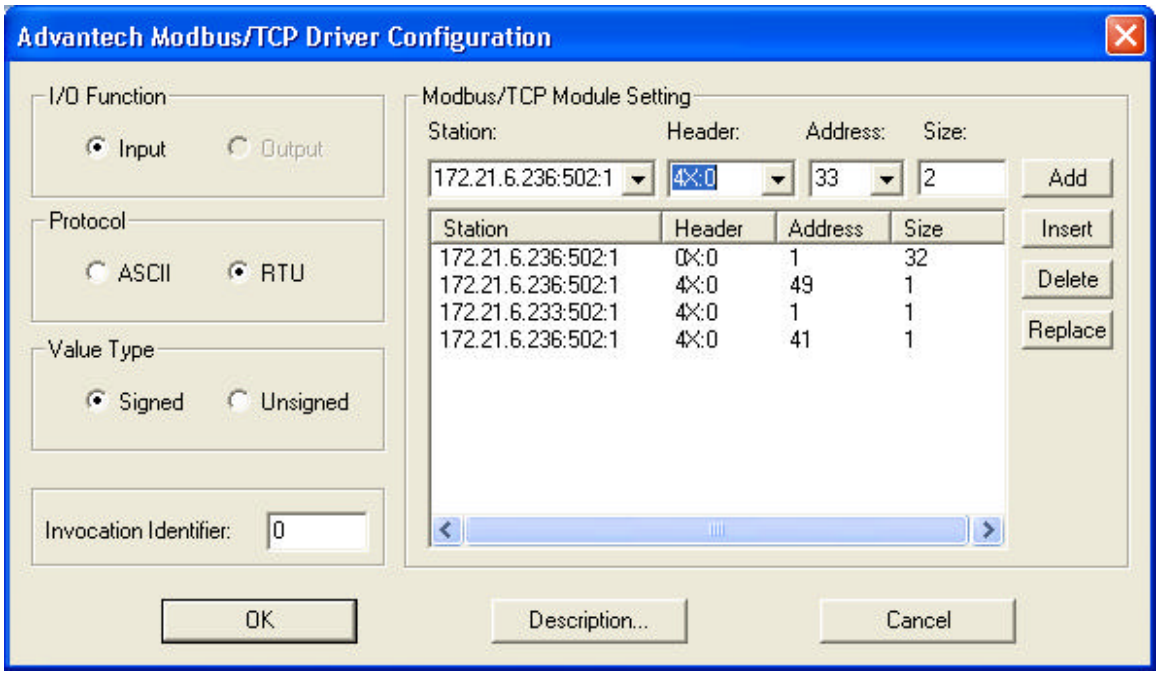
Select the task that will control the I/O exchange. The task selected will determine the frequency that the inputs and/or outputs are read and written. A task executes top down as show below.



NOTE: This is very important, the same task for both input and output I/O groups must be used, AND, only have one Input and one Output I/O group per Modbus/TCP driver implementation can be configured.

If this method is not followed, it is a possibility that a higher priority task will interrupt a lower priority task and therefore interrupt communication and cause loss of data.

Driver dialog



Station: The station value is the IPAddress:Port:DeviceID.

Example: 172.21.6.236:502:1
IPAddress: 172.21.6.236
Port: 502
Device ID: 1

The **Header** is the I/O Type:
Register: 4X:0
Coil : 0X:0

Address: This is the starting address on the module.

Size: This is the number of **I/O POINTS**.
For 6 coils it will be a 6.
For 6 registers it will be a 6 also.

Mapping I/O to Variables

Since the ProConOS Engine works only with BYTE size data, it is important to understand the next concept in order to use the Modbus/TCP driver properly.

REGITSERS:
When configuring Register type I/O, it is very straightforward. One Register value uses 2 BYTES.
1 I/O point = 2 BYTES.

COILS:

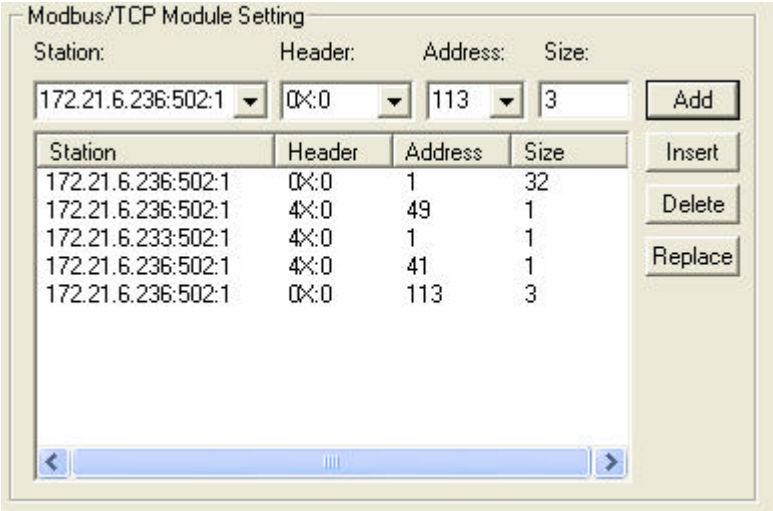
When configuring Coil type I/O (digital I/O), It takes 8 I/O points to use 1 byte.
8 I/O Points = 1 BYTE

Therefore, when a number of Coils less than 8 are used, it still takes one byte. The remaining BITS will be ignored. If the number of Coils is greater than 8 then it will be the number of coils/8. If there is a remainder then one additional byte will be used.

Examples:

16 Coils = 2 BYTES
17 Coils = 3 BYTES
23 Coils = 3 BYTES
24 Coils = 3 BYTES
25 Coils = 4 BYTES

The **Driver Configuration Dialog** will automatically return the proper amount of BYTES. It is up to the end user to map the I/O properly. The example below shows mixed Register and Coils, with the corresponding Mapped variables. The order of appearance is directly related to how they will map to variables in Multiprog.



This example above will return 11 BYTES, (32/8 coils, + 3 registers X 2 and one more byte for the 3 coils at the end.

The configured I/O will start at address 0 with a length of 11 as mentioned above. Next is an example of how to map the variables.

Variable Worksheet:

Name	Type	Usage	Description	Address
Default				
Adam_5051DSlot0	UINT	VAR_GLOBAL	First 16 Coils, 8 are double mapped below as BOOL	%MW0
Adam_5051DSlot1	UINT	VAR_GLOBAL	Coils 17 to 32	%MW2
Adam_5018_TC0	UINT	VAR_GLOBAL	First Register Value	%MW4
A6017	UINT	VAR_GLOBAL	Second Register Value	%MW6
A5017	UINT	VAR_GLOBAL	Third Register Value	%MW8
Digital_inputs	USINT	VAR_GLOBAL	This is a short integer to hold the last three digital values	%IB10
Adam_5051DSlot0Bit0	BOOL	VAR_GLOBAL	First Coil	%MW0.0
Adam_5051DSlot0Bit1	BOOL	VAR_GLOBAL	Second Coil	%MW0.1
Adam_5051DSlot0Bit2	BOOL	VAR_GLOBAL	Third Coil	%MW0.2
Adam_5051DSlot0Bit3	BOOL	VAR_GLOBAL	Fourth Coil	%MW0.3
Adam_5051DSlot0Bit4	BOOL	VAR_GLOBAL	Fifth Coil	%MW0.4
Adam_5051DSlot0Bit5	BOOL	VAR_GLOBAL	Sixth Coil	%MW0.5
Adam_5051DSlot0Bit6	BOOL	VAR_GLOBAL	Seventh Coil	%MW0.6
Adam_5051DSlot0Bit7	BOOL	VAR_GLOBAL	Eighth Coil	%MW0.7

This example shows the inputs in order of the way they are configured. Since the Coils appear first in the Driver Configuration Dialog, then they must also be mapped to the first 4 BYTES (0-3). They do not have to appear in order on the variable worksheet but the address must be this way.
This example also shows the first bank of digital values double mapped as an UNSIGNED INTEGER and as BOOL. This way the programmer can look at a bank of digitals at once or individually.

For more information on variables see the Multiprog Help files.

11. Other Items.

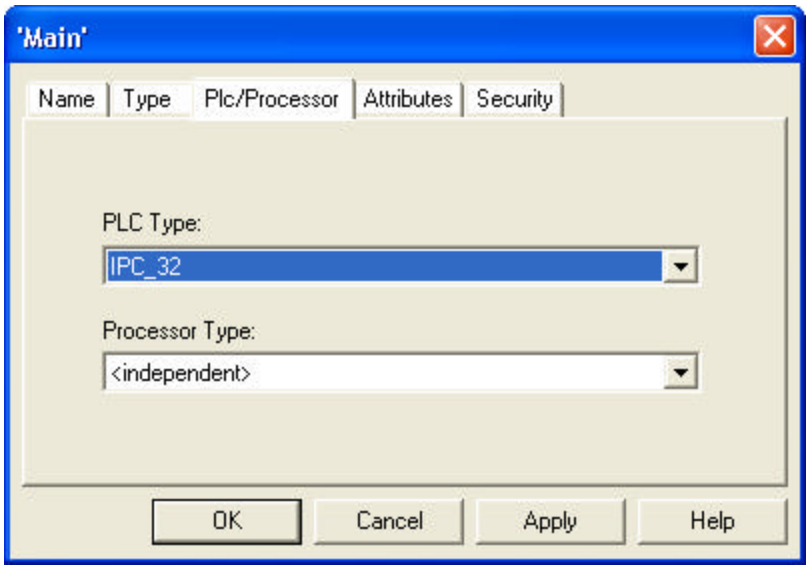
Proconos Retained data

Since Advantech **ECS** targets do not have battery backed-up RAM, the retained data is retained on the persistent storage (compact flash card) in the same directory as the ProConOS engine.

To use this feature the end user must mark the tag names they want retained in the Variable worksheets where they are declared as in the example below. Local and global variables can be retained.

Name	Type	Usage	Description	Address	Init	Retain
Default						
Adam_4016_Addr1	REAL	VAR_GLOBAL	Analog Input	%ID0		☐
Adam5055S_Add3_0	BYTE	VAR_GLOBAL	8 Digital Inputs	%IB4		☐
Adam5051D_Add3_1	WORD	VAR_GLOBAL	16 Digital Inputs	%IW5		☐
Adam5051D_Add3_2	WORD	VAR_GLOBAL	16 Digital Inputs	%IW7		☐
Adam5056D_Add3_3	WORD	VAR_GLOBAL	16 Digital Outputs	%QW0		☐
Setpoint1	DINT	VAR_GLOBAL				☑
Setpoint2	INT	VAR_GLOBAL				☑

Marking the tags is not enough to have the data retained. There is a function that also must be called that will force the retained file to be written. The developer must be careful to not have excess calls to this function as it can shorten the life of the Compact Flash card. The Function Call is “WRITE_RETAIN” and can only be called from a program that is declared with a PLC Type of IPC_32 as shown in the example “properties dialog” of the “Main” program below. This Property can be declared at the time of creating the program or later by right click and “settings” on the Program name in the Project tree.



The function call will start the write when a rising edge is detected on the input of the function. Once the write is finished, it will not attempt another write until another rising edge is detected. For more information on this function see the online help for function blocks within Multiprog.

Adam-5000/485 communication watchdog implementation

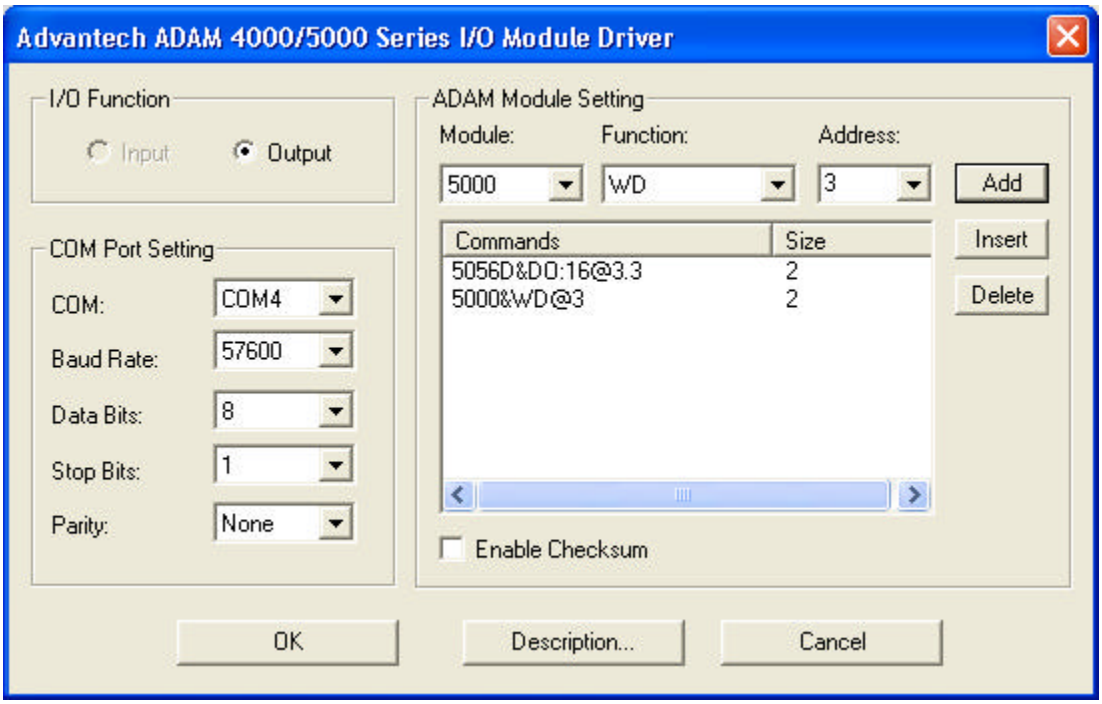
1) The Communication watchdog uses a new firmware for the Adam-5000. **You must load this firmware to the Adam-5000 unit before this function will be supported.** The command is supported as follows:

ADAM-5000/485 Command Set

Command Syntax	Command Response Syntax	Command Description	Command Example	Command response Example
$AAXnnnn(cr) nnnn: 0000 ~ 9999 unit: 0.1sec	**!AA(cr)**	1.Set communication WDT value form 0000 ~ 9999 (if value is 0000 the communication WDT function will be disable) 2.Whan the Init pin connect to ground or Adam-5000/485 address is “0”, The WDT function will be disable	$05X0030(cr)	!05(cr)

2) If this command is implemented in KW, It is written once to the unit upon startup to implement the watchdog and to set the timeout. It is also written again upon shutdown to disable the watchdog timer.

3) Add an output command to the I/O configuration as shown below:



4) Since this command uses two bytes, implement a variable and set the initial value as your timeout value as shown below:

Name	Type	Usage	Description	Address	Init	Retain
Default						
Adam_4016_Addr1	REAL	VAR_GLOBAL	Analog Input	%ID0		☐
Adam5055S_Add3_0	BYTE	VAR_GLOBAL	8 Digital Inputs	%IB4		☐
Adam5051D_Add3_1	WORD	VAR_GLOBAL	16 Digital Inputs	%IW5		☐
Adam5051D_Add3_2	WORD	VAR_GLOBAL	16 Digital Inputs	%IW7		☐
Adam5056D_Add3_3	WORD	VAR_GLOBAL	16 Digital Outputs	%QW0		☐
Adam5000WD_Add3	UINT	VAR_GLOBAL	Comm Watchdog	%QW2	20	☐
Setpoint1	DINT	VAR_GLOBAL				☑
Setpoint2	INT	VAR_GLOBAL				☑

This picture shows the “CommWDA5000_1” variable mapped, and with an initial value of 20. The value 20 = 2 seconds and zero tenths of seconds

Revisions

Author	Changes	Date
Walt Medlock	Created	12/15/2003