A2420 Communication Driver

Driver for communication with Altus devices
(ALNET II network) using the AL2420 Board

Contents

INTRODUCTION 2

GENERAL INFORMATION 3

DEVICE CHARACTERISTICS 3
LINK CHARACTERISTICS 3
DRIVER CHARACTERISTICS 3
CONFORMANCE TESTING 4

INSTALLING THE DRIVER 5

CONFIGURING THE DEVICE 6

CONFIGURING THE DRIVER 6

SETTING THE COMMUNICATION PARAMETERS 6
CONFIGURING THE STANDARD DRIVER WORKSHEET 8

EXECUTING THE DRIVER 13

TROUBLESHOOTING 14

SAMPLE APPLICATION 16

REVISION HISTORY 17

Introduction

The A2420 driver enables communication between the Studio system and a set of devices using the ALNET II protocol via the A2420 communication board, according to the specifications discussed in this document.

This document was designed to help you install, configure, and execute the A2420 driver to enable communication with these Altus devices. The information in this document is organized as follows:

- **Introduction**: Provides an overview of the A2420 driver documentation.
- **General Information**: Provides information needed to identify all the required components (hardware and software) used to implement communication between Studio and the A2420 driver.
- **Installing the Driver**: Explains how to install the A2420 driver.
- **Configuring the Device**: Explains how to configure the Altus device.
- **Configuring the Driver**: Explains how to configure the communication driver.
- **Executing the Driver**: Explains how to execute the driver to verify that you installed and configured the driver correctly.
- **Troubleshooting**: Lists the most common error codes for this protocol and explains how to fix these errors.
- **Sample Application**: Explains how to use a sample application to test the driver configuration.
- **Revision History**: Provides a log of all modifications made to the driver and the documentation.

> ✍ **Notes:**
>
> - This document assumes that you have read the “Development Environment” chapter in the product’s *Technical Reference Manual*.
> - This document also assumes that you are familiar with the Windows NT/2000/XP environment.
>   If you are unfamiliar with Windows NT/2000/XP, we suggest using the **Help** feature (available from the Windows desktop **Start** menu) as you work through this guide.

General Information

This chapter explains how to identify all the hardware and software components used to implement communication between the Studio A2420 driver and devices using the ALNET II protocol.

The information is organized into the following sections:

- Device Characteristics
- Link Characteristics
- Driver Characteristics

Device Characteristics

To establish communication, you must use devices with the following specifications:

- **Manufacturer**: Altus
- **Compatible Equipment**:
  - AL-2420 communication board + ALNET II compatible PLC (AL-2000 / AL-2002 / AL-2003 / QK2000)
- **Programming software:**
  - A2420 Gateway: AL2420 for NT programming software.
  - PLC: MasterTool® programming software

For a list of the devices used for conformance testing, see “Conformance Testing” on page 4.

Link Characteristics

To establish serial communication, you must use links with the following specifications:

- **Device Communication Port**: AL2420 communication board
- **Physical Protocol**: RS485
- **Logic Protocol**: ALNET II
- **Device Runtime Software**: AL2420 for WinNT
- **Specific PC Board**: AL2420 communication board
- **Communication cable**: AL2300 + AL2600 adapter for ALNET II network

> ➲ **Attention:**
> It is mandatory to install the AL2420 configuration software (provided by Altus) prior to executing the A2420 communication driver from Studio. The A2420 driver uses library files installed by the AL2420 configuration software.

Driver Characteristics

The A2420 driver is composed of the following files:

- **`A2420.INI`**: Internal driver file. *You must not modify this file*.
- **`A2420.MSG`**: Internal driver file containing error messages for each error code. *You must not modify this file*.
- **`A2420.PDF`**: Document providing detailed information about the A2420 driver.
- **`A2420.DLL`**: Compiled driver.

✍ **Notes:**

- All of the preceding files are installed in the **`/DRV`** subdirectory of the Studio installation directory.
- You must use Adobe Acrobat® Reader™ (provided on the Studio installation CD-ROM) to view the **`A2420.PDF`** document.

You can use the A2420 driver on the following operating systems:

- Windows XP/2K/NP

✍ **Note:**

- The board manufacturer must provide the device driver (communication library) for the operating system where the board is installed.

For a list of the operating systems used for conformance testing, see the "Conformance Testing" section.

Conformance Testing

The following hardware/software was used for conformance testing:

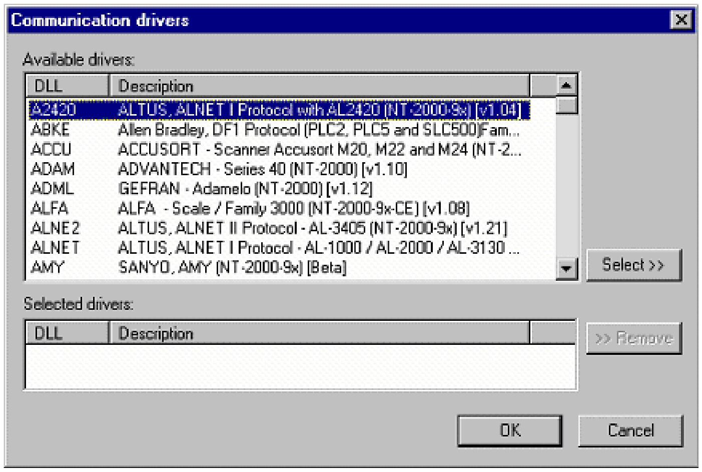
- **Equipment**: AL2420 communication board + AL2000 PLC
- **Configuration**:
- **Cable**: AL2300 + AL2600 adapter for ALNET II network
- **Operating System** (*development*): Windows 2000
- **Operating System** (*target*):
  - Windows 2000
- **Studio Version**: 6.0
- **Driver Version**: 1.04

Installing the Driver

When you install Studio version 5.1 or higher, all of the communication drivers are installed automatically. You must select the driver that is appropriate for the application you are using.

Perform the following steps to select the driver from within the application:

1. Open Studio from the **Start** menu or double-click the Studio shortcut icon on your desktop.
2. From the Studio main menu bar, select **File** → **Open Project** to open your application.
3. Select **Insert** → **Driver** from the main menu bar to open the *Communication drivers* dialog.
4. Select the **A2420** driver from the *Available Drivers* list, and then click the **Select** button.



DLL	Description
A2420	ALTUS, ALNET I Protocol with AL2420 (NT-2000-9x) [v1.04]
ABKE	Allen Bradley, DF1 Protocol (PLC2, PLC5 and SLC500)Fam...
ACCU	ACCUSORT - Scanner Accusort M20, M22 and M24 (NT-2...
ADAM	ADVANTECH - Series 40 (NT-2000) [v1.10]
ADML	GEFRAN - Adamelo (NT-2000) [v1.12]
ALFA	ALFA - Scale / Family 3000 (NT-2000-9x-CE) [v1.08]
ALNE2	ALTUS, ALNET II Protocol - AL-3405 (NT-2000-9x) [v1.21]
ALNET	ALTUS, ALNET I Protocol - AL-1000 / AL-2000 / AL-3130 ...
AMY	SANYO, AMY (NT-2000-9x) [Beta]

DLL	Description

Communication Drivers Dialog

5. When the **A2420** driver displays in the **Selected Drivers** list, click the **OK** button to close the dialog.

> **Note:**
> You must install the AL2420 board library supplied by Altus. This library is installed when you install Altus configuration software for AL-2420 board. After installing this library you must create a system variable named AL2420 with the same path of the file generate by the software configuration (AL2420.CFG). This variable must be create by the **Control Panel -> System -> Environment -> System Variables**.

> **Attention:**
> For safety reasons, you must use special precautions when installing the physical hardware. Consult the hardware manufacturer’s documentation for specific instructions in this area.

Configuring the Device

Because there are several brands of equipment that use the ALNET II protocol, it is impossible to define a standard device configuration.

Configuring the Driver

After opening Studio and selecting the **A2420** driver, you must configure the driver. Configuring the A2420 driver is done in two parts:

- Specifying communication parameters (only one configuration needed).
- Defining communication tags and controls in the Communication tables or *Driver* worksheets (Standard and Main Driver Worksheets).

  Worksheets are divided into two sections, a *Header* and a *Body*. The fields contained in these two sections are standard for all communications drivers — except the **Station**, **Header**, and **Address** fields, which are driver-specific. This document explains how to configure the **Station**, **Header**, and **Address** fields only.

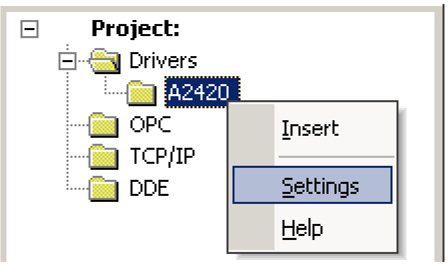
> ✍ **Note:**
> For a detailed description of the Studio Standard and Main Driver Worksheets, and information about configuring the standard fields, review the product's *Technical Reference Manual*.

Setting the Communication Parameters

Use the following steps to configure the communication parameters, which are valid for all driver worksheets configured in the system):

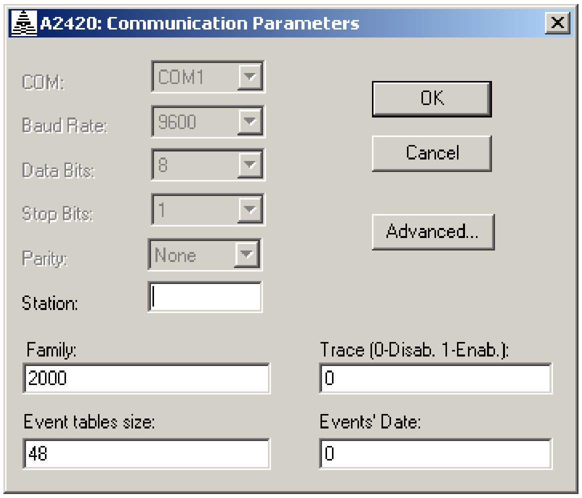
From the Studio development environment, select the **Comm** tab located below the *Workspace*.

2. Click on the *Drivers* folder in the *Workspace* to expand the folder.
3. Right-click on the *A2420* subfolder and when the pop-up menu displays, select the **Settings** option:



Select Settings from the Pop-Up Menu

The *A2420: Communications Parameters* dialog displays:


A2420: Communication Parameters

COM: COM1
Baud Rate: 9600
Data Bits: 8
Stop Bits: 1
Parity: None
Station:
Family: 2000
Trace (0-Disab. 1-Enab.): 0
Event tables size: 48
Events' Date: 0

A2420: Communication Parameters Dialog

4. Specify the parameters as noted in the following table:

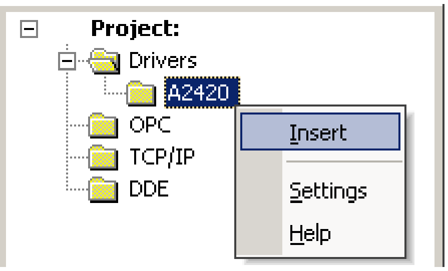
Parameters	Default Values	Valid Values	Description
Station	0	0	Not required for this driver
Family	2000	1000 2000	▪ 1000: Compatible with the AL1000 PLC Family ▪ 2000: Compatible with the AL2000 PLC Family
Event tables size	48	Numeric value	Number of bytes received by the AL3130 event board when a event occurs. This setting is used only when the driver is configured to read events from the AL3130 board
Trace (0-Disab. 1-Enab.)	0	0 or 1	When this field is enabled (value = 1), additional debugging messages are sent to the LogWin module.
Events' Date	0	0 or 1	When enabled (value = 1), the current date is concatenated to the event values.

Configuring the Standard Driver Worksheet

This section explains how to configure a *Standard Driver Worksheet* (or Communication table) to associate application tags with the PLC addresses. You can configure multiple *Driver* worksheets — each of which is divided into a *Header* section and *Body* section.

Use the following steps to create a new Standard Driver Worksheet:

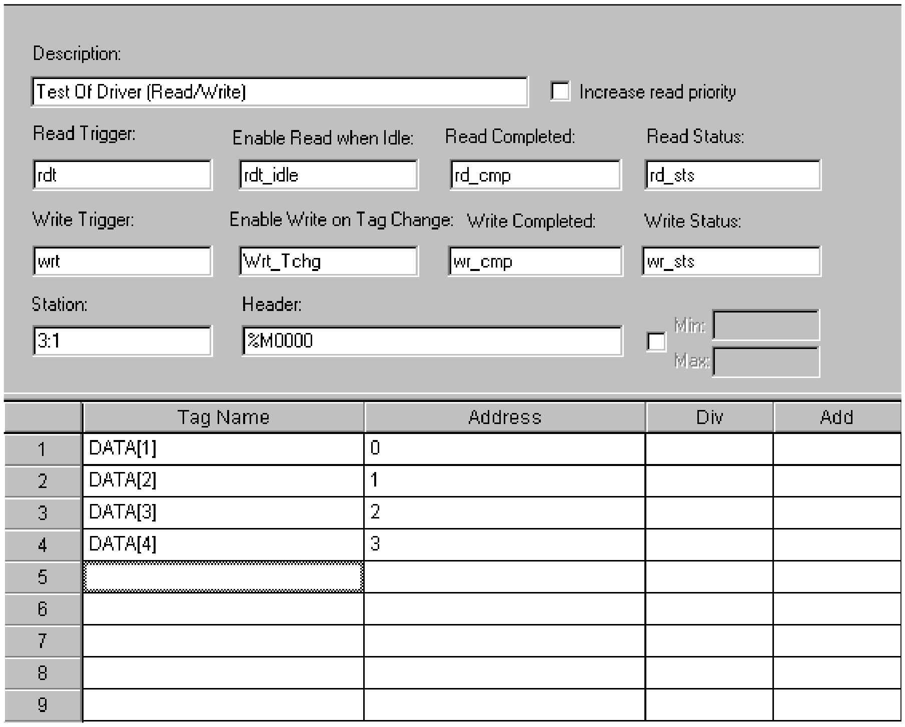
1. From the Studio development environment, select the **Comm** tab, located below the *Workspace* pane.
2. In the *Workspace* pane, expand the *Drivers* folder and right-click the *A2420* subfolder.
3. When the pop-up menu displays, select the **Insert** option:



Inserting a New Worksheet

> **Note:**
> To optimize communication and ensure better system performance, you must tie the tags in different driver worksheets to the events that trigger communication between each tag group and the period in which each tag group must be read or written. Also, we recommend configuring the communication addresses in sequential blocks to improve performance.

The *<drivername>.drv* dialog displays (similar to the following figure):



Description: Test Of Driver (Read/Write)

□ Increase read priority

Read Trigger: rdt	Enable Read when Idle: rdt_idle	Read Completed: rd_cmp	Read Status: rd_sts
Write Trigger: wrt	Enable Write on Tag Change: Wrt_Tchg	Write Completed: wr_cmp	Write Status: wr_sts
Station: 3:1	Header: %M0000		□ Min: Max:

	Tag Name	Address	Div	Add
1	DATA[1]	0		
2	DATA[2]	1		
3	DATA[3]	2		
4	DATA[4]	3		
5				
6				
7				
8				
9				

A2420 Driver Worksheet

4. Use the following information to complete the **Station**, **Header**, and **Address** fields on this worksheet:

- **Station** field: This field must comply with the following syntax:

  <Node>:<SubNetwork> (for example: 5:2)

  Where:

  - **Node** is the ID number of the PLC.

  - **SubNetwork** is the number of the sub-network where the PLC is connected.

- **Header** field: This field must comply with the following syntax:

  %<Type><InitAddress> (for example: %M0010)

  Where:

  - **Type** is the register type, according to the list below:

  A = Auxiliary

  E = Input

  S = Output

  R = Physical Address

  M = Memory

  D = Decimal Memory

  TM = Memory Table

  TD = Decimal Memory Table

  - **InitAddress** is the initial address used as reference on the worksheet. It must be composed with 4 digits.

> **Note:**
> When configuring registers TM or TD, it is possible to specify the initial table position on the Header, using the syntax %<Type><TableNumber>[<InitPosition>] (for example: %TM10[5]). If the initial table position (<InitPosition>) is not specified in the header, the position 0 (zero) is used as the initial reference for the worksheet.

Also, you can type a tag string in brackets **`{Tag}`** into the **Header** field, but you must be certain that the tag's value is correct and that you are using the correct syntax, or you will get an **`Invalid Header`** error.

The following table lists all of the data types and address ranges that are valid for the **Header** field:

Register Type	Example of syntax for the Header field	Range of values supported for the initial address
%A	%A0000	0 to 95 0 to 255 (AL2003 only)
%E	%E0000	0 to 9999
%S	%S0000	0 to 9999

%R	%R0000	0 to 95 0 to 255 (AL2003 only)
%M	%M0000	0 to 9999
%D	%D0000	0 to 9999
%TM	%TM0000[0]	0 to 9999
%TD	%TD0000[0]	0 to 9999

Note:

The number of %A registers is fixed for each PLC. The number of %E, %S and %R registers is configured by the user, according to the physical I/O cards connected to the CPU. The number of %M, %TM, %D, %TD is defined by the PLC programmer, according to the amount of memory available on the PLC.

After specifying the **Station** and **Header** parameters, proceed to the next section for instructions on how to configure the **Address** field.

CONFIGURING THE ADDRESS FIELD

The body of the *Driver* worksheet allows you to associate each tag to its respective address in the device. Type the tag from your application database into the **Tag Name** column. This tag will receive values from or send values to an address on the device.

The address must comply with the following syntax:

<AddressOffset><SubSet><SubSetNumber>

Where:

- **AddressOffeset** is the number that when added to the **initial address** configured in the **Header** field defines an address in the device.

- **SubSet** is an optional parameter. When configured, it indicates a part of the register that is been addressed:

. = bit

n = nibble

b = byte

w = word (for %D and %TD register types only)

h = bit of high word (for %D and %TD register types only)

- **SubSetNumber** is the number of the subset specified for a specific address.

Attention:

- In the *Driver* worksheet, the start address is defined by the sum of the Header (from the **Header** field) and the lowest **`<AddressOffset>`** parameter (in the **Address** field).
- You must not configure a range of addresses greater than the maximum block size (*data buffer length*) supported by each PLC within the same *Driver* worksheet. The maximum data

buffer length for this driver is 128 bytes per *Standard Driver Worksheet*.

Address Configuration Sample		
Device Address	**Device Address**	**Address Field**
%A0000	%A0000	0
%A0010	%A0000	10
%A0010	%A0010	0
%A0000(bit 0)	%A0000	0.0
%M0001	%M0001	0
%M0005	%M0000	5
%M0010(byte 0)	%M0005	5b0
%D0000	%D0000	0
%D0025	%D0025	0
%D0024	%D0000	25
%D0007(word 1)	%D0000	7w1
%E0010	%R0000 or %S0000 or %E0000	10
%E0010(bit 6)	%R0005 or %S0005 or %E0005	5.6
%S0020	%R0005 or %S0005 or %E0005	15
%S0020(bit 7)	%R0005 or %S0005 or %E0005	15.7
%TM0000 (position 0)	%TM0000	0
%TM0000 (position 5)	%TM0000[5]	0
%TM0000 (position 5)	%TM0000	5
%TM0010 (position 8, bit 3)	%TM0010	8.3
%TD0000 (position 0)	%TD0000	0
%TD0000 (position 5)	%TD0000[5]	0
%TD0000 (position 5)	%TD0000	5
%TD0010 (position 8, bit 3)	%TD0010	8.3

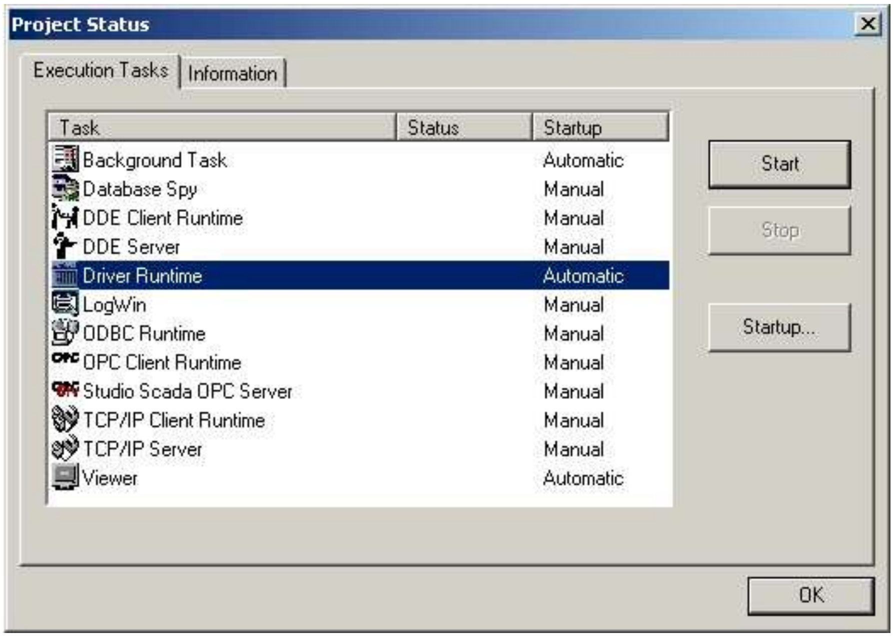
Executing the Driver

After adding the A2420 driver to a project, Studio sets the project to execute the driver automatically when you start the run-time environment.

To verify that the driver run-time task is enabled and will start correctly, perform the following steps:

Select **Project** → **Status** from the main menu bar.

The *Project Status* dialog displays:



Task	Status	Startup
Background Task		Automatic
Database Spy		Manual
DDE Client Runtime		Manual
DDE Server		Manual
Driver Runtime		Automatic
LogWin		Manual
ODBC Runtime		Manual
OPC Client Runtime		Manual
Studio Scada OPC Server		Manual
TCP/IP Client Runtime		Manual
TCP/IP Server		Manual
Viewer		Automatic

Project Status Dialog

2. Verify that the *Driver Runtime* task is set to **Automatic.**
   - If the setting is correct, click **OK** to close the dialog.
   - If the **Driver Runtime** task is set to **Manual**, select the **Driver Runtime** line. When the **Startup** button becomes active, click the button to toggle the *Startup* mode to **Automatic.**
3. Click **OK** to close the *Project Status* dialog.
4. Start the application to run the driver.

Troubleshooting

If the A2420 driver fails to communicate with the device, the tag you configured for the **Read Status** or **Write Status** fields will receive an error code. Use this error code and the following table to identify what kind of failure occurred.

Error Code	Description	Possible Causes	Procedure to Solve
0	OK	Communication without problems	-
4	Load library error.	Error loading library of AL2420 board.	▪ Install the AL2420 library. ▪ Configure the AL2420 board using its configuration software and generate the AL2420.cfg file ▪ Create the system variable for the AL2420 board by the Control Panel.
5	Block Size Error	Address offset greater then : 255 for %A, %E, %S or %R. 128 for %M or %TM 64 for %D or %TD	Modify the configuration of the Standard Driver Sheet and make sure that the difference between the highest address and the lowest address configured in the worksheet is not greater than 128 bytes.
6	Initialize board error.	Driver can not initialize board.	Check the path for the configuration file in the variables system.
7	Invalid Header	An invalid Header has been typed or the tag that is inside this field has an invalid configuration.	Type a valid Header on the header field.
8	Invalid Address	An invalid Address has been typed.	Type a valid Address on the address field.
9	Send message error.	▪ Invalid station. ▪ Invalid board configuration (Altus configuration software).	▪ Check the PLC node and sub-network configured in the Station field. ▪ Check the board configuration with its configuration software and execute the self-test from this software.
10	Timeout waiting message.	▪ Invalid station. ▪ Invalid Header. ▪ Invalid Address. ▪ Invalid configuration of board (Altus configuration software).	▪ Type a valid Header on the header field. ▪ Type a valid Address on the address field. ▪ Check the PLC node and sub-network configured in the Station field. ▪ Check the board configuration with its configuration software and execute the self-test from this software.
45	Unsolicited message error.	There is not any driver worksheet configured to treat unsolicited messages	Create a driver worksheet with the Header and Station of the unsolicited message.
240	File error.	File to receive data from F-ARQ could not be created.	Check path and name configured for the file that must store the data received from the F-ARQ function.
-15	Timeout waiting start a message.	▪ Disconnected cables ▪ PLC turned off, or in Stop or	▪ Check the cable wiring ▪ Check the PLC state.

		error mode ▪ Wrong Station number ▪ Wrong RTS/CTS control settings	▪ Check the station number. ▪ Check the Communication Parameters settings.
-17	Timeout between rx char.	▪ Disconnected cables ▪ PLC turned off, or in Stop or error mode ▪ Wrong Station number ▪ Wrong RTS/CTS control settings	▪ Check the cable wiring ▪ Check the PLC state. ▪ Check the station number. ▪ Check the Communication Parameters settings.

⇨ **Tip:**

You can verify communication status using the Studio development environment *Output* window (*LogWin* module). To establish an event log for **Field Read Commands**, **Field Write Commands**, and **Serial Communication** right-click in the *Output* window. When the pop-up menu displays, select the option to set the log events. If you are testing a Windows CE target, you can enable the log at the unit (**Tools** → **LogWin**) and verify the **`celog.txt`** file created at the target unit.

If you are unable to establish communication with the PLC, try to establish communication between the PLC Programming Tool and the PLC. Quite frequently, communication is not possible because you have a hardware or cable problem, or a PLC configuration error. After successfully establishing communication between the device's Programming Tool and the PLC, you can retest the supervisory driver.

To test communication with Studio, we recommend using the sample application provided rather than your new application.

If you must contact us for technical support, please have the following information available:

- **Operating System** (type and version): To find this information, select **Tools** → **System Information**.
- **Project Information**: To find this information, select **Project** → **Status.**
- **Driver Version** and **Communication Log**: Displays in the Studio *Output* window when the driver is running.
- **Device Model** and **Boards**: Consult the hardware manufacturer's documentation for this information.

Sample Application

You will find a sample application in the /COMMUNICATION EXAMPLES/A2420 directory. We strongly recommend that you use this sample application to test the A2420 driver before configuring your own customized application, for the following reasons:

- To better understand the information provided in the section of this document.
- To verify that your configuration is working satisfactorily.
- To certify that the hardware used in the test (device, adapter, cable, and PC) is working satisfactorily before you start configuring your own, customized applications.

> ✍ **Note:**
> This application sample is not available for all drivers.

Use the following procedure to perform the test:

1. Configure the device's communication parameters using the manufacturer's documentation.
2. Open and execute the sample application.
3. Execute the *Viewer* module in Studio to display information about the driver communication.

> ⇨ **Tip:**
> You can use the sample application screen as the maintenance screen for your custom applications.

Revision History

Doc. Revision	Driver Version	Author	Date	Description of Changes
-	1.01	Roberto Vigiani	Jan/2000	Initial driver version
-	1.03	Roberto Vigiani	Ago/2001	Updated for UNICODE compliance.
A	1.04	Fabio Komura	Feb/2004	Modified internal algorithm (register and unregister myClass).