

UNITE Communication Driver

Driver for Serial Communication
with Devices using UNI-TELWAY Protocol

Contents

INTRODUCTION2

GENERAL INFORMATION.....3

 DEVICE CHARACTERISTICS3

 LINK CHARACTERISTICS3

 DRIVER CHARACTERISTICS3

 CONFORMANCE TESTING4

INSTALLING THE DRIVER5

CONFIGURING THE DRIVER6

 SETTING THE COMMUNICATION PARAMETERS6

 CONFIGURING THE DRIVER WORKSHEETS8

EXECUTING THE DRIVER 11

TROUBLESHOOTING 12

SAMPLE APPLICATION 13

REVISION HISTORY..... 14

Introduction

The UNITE driver enables communication between the Studio system and devices using the UNI-TELWAY protocol communicating over Serial, according to the specifications discussed in this document.

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

- **Introduction:** Provides an overview of the UNITE driver documentation.
- **General Information:** Provides information needed to identify all the required components (hardware and software) used to implement communication between Studio and the UNITE driver.
- **Installing the Driver:** Explains how to install the UNITE driver.
- **Configuring the Driver:** Explains how to configure the UNITE 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 UNITE 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 *Studio Users Guide and 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 UNITE driver and the UNI-TELWAY PLC.

The information is organized into the following sections:

- Device Characteristics
- Link Characteristics
- Driver Characteristics
- Conformance Testing

Device Characteristics

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

- **Manufacturer:** Modicon / Telemecanique / Schneider
- **Compatible Equipment:** TSX Micro
- **Programming Software:** PL7 Pro
- **Device Runtime Software:** None

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

Link Characteristics

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

- **Physical Protocol:** Serial
- **Logic Protocol:** UNI-TELWAY
- **Specific PC Board:** None

Driver Characteristics

The UNITE driver is composed of the following files:

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

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

- Windows NT/2K/XP
- Windows CE

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

The UNITE driver supports the following registers:

Register Type	Length	Write	Read	Bit	Integer	DWord
%M (Internal Bits)	1 Bit	•	•	•	–	–
%S (System Bits)	1 Bit	–	•	•	–	–
%MW (Internal Words)	1 Word	•	•	•	•	•
%KW (Constant Words)	1 Word	–	•	•	•	•

Conformance Testing

The following hardware/software was used for conformance testing:

- **Driver Configuration:**
 - **Baud Rate:** 9600
 - **Data Bits:** 8
 - **Stop Bits:** 1
 - **Parity:** Odd
 - **Station:** 1
 - **COM Port:** COM1
- **Cable:** TSXPCX 1031 PV:01 RL03

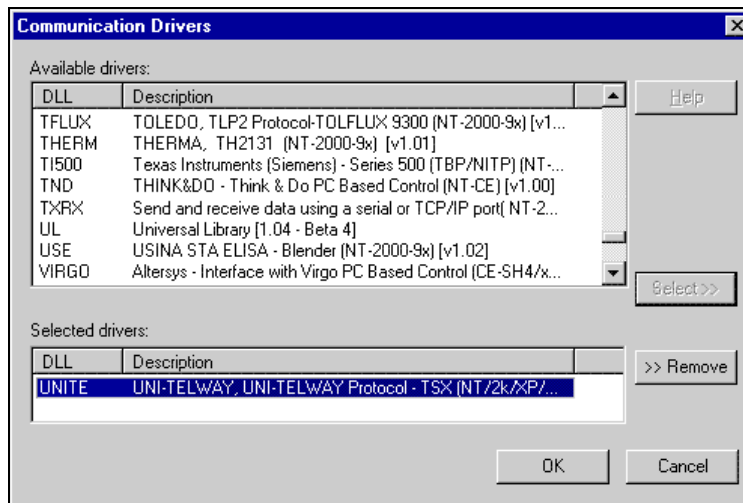
Driver Version	Studio Version	Operating System	Equipment
1.04	6.1	Windows XP	TSX 3722 V2.0

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.
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 **UNITE** driver from the *Available Drivers* list, and then click the **Select** button:



Communication Drivers Dialog

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



Note:

It is not necessary to install any other software on your computer to enable communication between the host and the device.



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 Driver

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

- Specifying communication parameters
- Defining tags and controls in the *STANDARD DRIVER SHEETS* (or Communication tables)

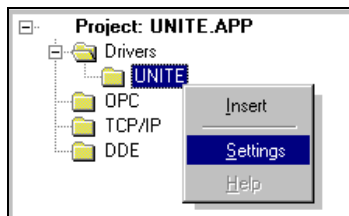
Worksheets are divided into two sections, a *Header* and a *Body*. The fields contained in these two sections are standard for all communications drivers—except the **Station**, **Header** and **Address** fields, which are driver-specific. This document explains how to configure the **Station**, **Header**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

Note:
For a detailed description of the Studio *STANDARD DRIVER SHEETS*, and information about configuring the standard fields, review the product's *Technical Reference Manual*.

Setting the Communication Parameters

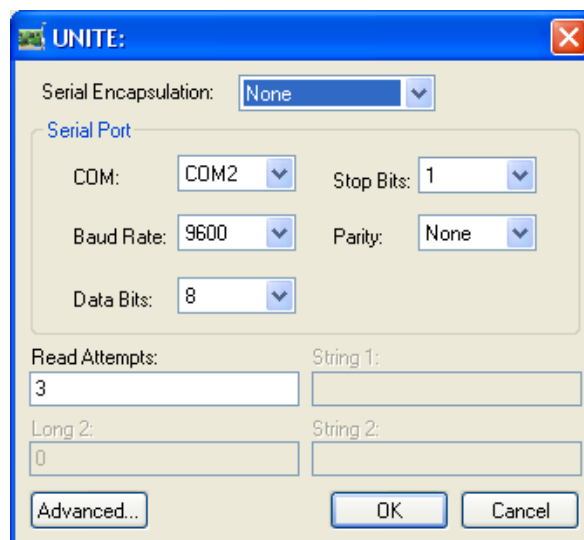
Use the following steps to configure the communication parameters, which are valid for all *Driver* worksheets configured in the system:

1. 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 UNITE subfolder. When the pop-up menu displays, select the **Settings** option:



Select Settings from the Pop-Up Menu


The *UNITE: Communication Parameters* dialog displays:




Communication Parameters Dialog

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

Parameters	Default Values	Valid Values	Description
Station	–	1 – 255	Slave Address
Read Attempts	3	1 – 10	Total number of attempts to communicate with the device if a timeout happens.

 **Note:**
The device must be configured with *exactly the same* parameters that you configured in the *UNITE Communication Parameters* dialog.

5. Click the **Advanced** button on the *Communication Parameters* dialog to open the *Advanced Settings* dialog and configure the settings that are necessary.

 **Notes:**

- Do not change any of the other *Advanced* parameters at this time. You can consult the *Studio Users Guide and Technical Reference Manual* for information about configuring these parameters for future reference.
- Generally, you must change the *Advanced* parameter settings if you are using a DCE (Data Communication Equipment) converter (232/485 for example), modem, and so forth between the PC, the driver and the host. You must be familiar with the DCE specifications before adjusting these configuration parameters.

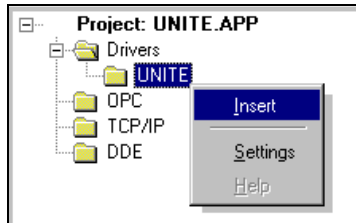
Configuring the Driver Worksheets

This section explains how to configure the *STANDARD DRIVER SHEET*s (or Communication tables) to associate application tags with the device addresses. You can configure multiple *Driver* worksheets—each of which is divided into a *Header* section and a *Body* section.

Configuring the *STANDARD DRIVER SHEET*

Use the following steps to create a new *STANDARD DRIVER SHEET*:

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 *UNITE* 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 *STANDARD DRIVER SHEET* displays (similar to the following figure):

Description: Increase priority

Read Trigger: Enable Read when Idle: Read Completed: Read Status:

Write Trigger: Enable Write on Tag Change: Write Completed: Write Status:

Station: Header: Min:
 Max:

	Tag Name	Address	Div	Add
1	M[1]	0		
2	M[2]	5		
3	M[3]	10		
4	M[4]	15		
5	M[5]	20		
6				
7				

STANDARD DRIVER SHEET

In general, all parameters on the *Driver* worksheet (except the **Station**, **Header** and **Address** fields) are standard for all communication drivers, but they will not be discussed in this document. For detailed information about configuring the standard parameters, consult the *Studio Users Guide and Technical Reference Manual*.

4. Use the following information to complete the **Station**, **Header**, and **Address** fields on this worksheet:
 - **Header** field: Use the information in the following table to define the type of variables that will be read from or written to the device, and a reference to the initial address. The default value is **%M:0**.
 These variables must comply with the following syntax:
%<Type>:<AddressReference> (for example: **MW:10**)

Where:

- **<Type>** is the register type (M, S, MW and KW)
- **<AddressReference>** is the initial address (reference) of the configured type.

After you edit the **Header** field, Studio checks the syntax to determine if it is valid. If the syntax is incorrect, Studio automatically inserts the default value (**%M:0**) in the **Header** field.

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; otherwise, 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 UNITE driver.

Header Field Information			
Data Types	Sample Syntax	Valid Range of Initial Addresses per Worksheet	Comments
M	%M:0	0 ... 4095	Internal Bits
S	%S:0	0 ... 63	System Bits
MW	%MW:0	0 ... 61	Internal Words
KW	%KW:0	0 ... 61	Constant Word

- **Address** field: Use this field to associate each tag to its respective device address.
 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:

[DataFormat]<AddressOffset>.[Bit] (for example, **10, DW20, 10.5**)

Where:

- **[DataFormat]** (optional parameter used for integer values only). If you do not specify this parameter, Studio inserts a word value. Valid value is **DW** (DWord).
- **<AddressOffset>** is a parameter added to the **AddressReference** parameter (configured in the **Header** field) to compose the group address configured in the **Header** field.
- **[Bit]** (optional parameter used for MW and KW types only) is the bit number to be read from or written to the device.

Address Configuration Sample		
Device Address	Header Field	Address Field
%M0	%M:0	0
%M5	%M:5	0
%M10	%M:5	5
%S0	%S:0	0
%S15	%S:15	0
%S20	%S:10	10
%MW1	%MW:0	1
%MW3	%MW:3	0
%MW10	%MW:5	5
%KW2	%KW:0	2
%KW6	%KW:6	0
%KW10	%KW:5	5
%MD0	%MW:0	DW0
%MD5	%MW:5	DW0
%MD10	%MW:5	DW5
%KD1	%KW:1	DW0
%KD5	%KW:5	DW0
%KD20	%KW:10	DW10

⇒ Attention:

You must not configure a range of addresses greater than the maximum block size (date buffer length) supported by each PLC within the same worksheet. The maximum data buffer length for this driver is 1024 bytes per *STANDARD DRIVER SHEET*.

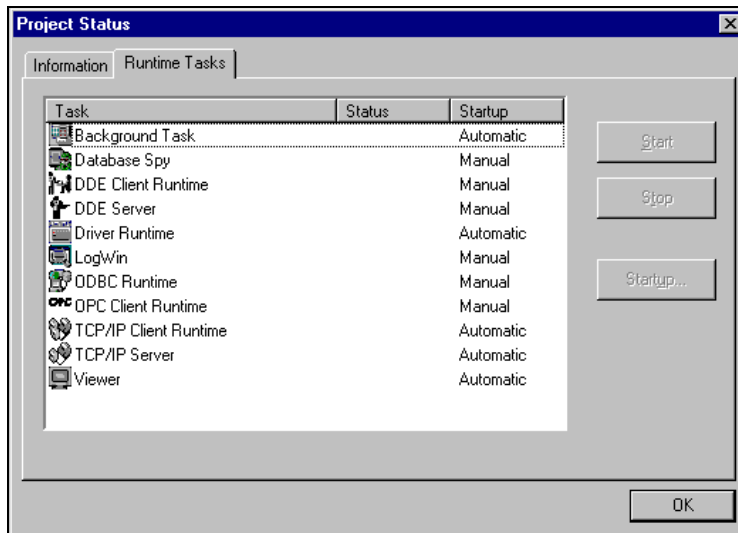
Executing the Driver

After adding the UNITE 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:

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

The *Project Status* dialog displays:



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 UNITE 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	None required
1	INVALID READ	<ul style="list-style-type: none"> ▪ Out of limits address ▪ Data Type invalid 	<ul style="list-style-type: none"> ▪ Verify that the address is valid (not out of limits). ▪ Verify that the Data type is valid on the PLC.
2	INVALID WRITE	Writing attempt in the System Bits (%S) or Constant Words (%KW)	You cannot write in these Data Types.
-15	Timeout Start Message	<ul style="list-style-type: none"> ▪ Disconnected cables ▪ PLC is turned off, in stop mode, or in error mode ▪ Wrong station number ▪ Wrong RTS/CTS control settings 	<ul style="list-style-type: none"> ▪ Check cable wiring. ▪ Check the PLC state – it must be RUN. ▪ Check the station number. ▪ Check the configuration. See the <i>Studio Users Guide and Technical Reference Manual</i> for information about valid RTS/CTS configurations.
-17	Timeout between rx char	<ul style="list-style-type: none"> ▪ PLC in stop mode or in error mode ▪ Wrong station number ▪ Wrong parity ▪ Wrong RTS/CTS configuration settings 	<ul style="list-style-type: none"> ▪ Check the PLC state – it must be RUN. ▪ Check the station number. ▪ Check the configuration. ▪ Check the configuration. See the <i>Studio Users Guide and Technical Reference Manual</i> for information about valid RTS/CTS configurations.

⇒ **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 use the Remote LogWin of Studio (**Tools** → **Remote LogWin**) to get the log events from the target unit remotely.

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**.
- **Studio Version**: To find this information, select **Help** → **About**.
- **Driver Version**: To find this information, read the full description of the driver on the *Communication Drivers* dialog.
- **Communication Log**: Displays in the Studio *Output* window (or *LogWin* window) when the driver is running. Be sure to enable the **Field Read Commands**, **Field Write Commands**, and **Serial Communication** for the *LogWin* window.
- **Device Model and Boards**: Consult the hardware manufacturer's documentation for this information.

Sample Application

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

- To better understand the information provided in each 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 sample application 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.

 **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
A	1.00	Eric Vigiani	Oct/15/2003	Initial version
B	1.01	Eric Vigiani	Apr/29/2004	Modified driver to read values properly when running under WinCE
C	1.02	Leandro G. Coeli	Mar/15/2005	Fixed bugs in communication
D	1.03	Rafael R. Fernandes	Apr/26/2007	Enhancement for working with several devices in the same network.
E	1.04	Fellipe Peternella	Sep/10/2010	Improved driver performance and added parameter Read Attempts.