

PIDAT Communication Driver

Driver for TCP/IP Communication
with PI Software (OSI)

Contents

INTRODUCTION2

GENERAL INFORMATION.....3

 DEVICE SPECIFICATIONS.....3

 NETWORK SPECIFICATIONS.....3

 DRIVER CHARACTERISTICS3

 CONFORMANCE TESTING4

SELECTING THE DRIVER5

CONFIGURING THE DEVICE6

CONFIGURING THE DRIVER6

 CONFIGURING THE COMMUNICATION SETTINGS6

 CONFIGURING THE DRIVER WORKSHEETS6

EXECUTING THE DRIVER9

TROUBLESHOOTING 10

SAMPLE APPLICATION 12

REVISION HISTORY..... 13

Introduction

The PIDAT driver enables communication between the Studio system and PI Software over TCP/IP, according to the specifications discussed in this document.

This document will help you to select, configure and execute the PIDAT driver, and it is organized as follows:

- **Introduction:** This section, which provides an overview of the document.
- **General Information:** Identifies all of the hardware and software components required to implement communication between the Studio system and the target device.
- **Selecting the Driver:** Explains how to select the PIDAT driver in the Studio system.
- **Configuring the Device:** Describes how the target device must be configured to receive communication from the PIDAT driver.
- **Configuring the Driver:** Explains how to configure the PIDAT driver in the Studio system, including how to associate database tags with device registers.
- **Executing the Driver:** Explains how to execute the PIDAT driver during application runtime.
- **Troubleshooting:** Lists the most common errors for this driver, their probable causes, and basic procedures to resolve them.
- **Sample Application:** Explains how to use a sample application to test the PIDAT driver configuration
- **Revision History:** Provides a log of all changes made to the driver and this documentation.

Notes:

- This document assumes that you have read the “Development Environment” chapter in Studio’s *Technical Reference Manual*.
- This document also assumes that you are familiar with the Microsoft Windows NT/2000/XP environment. If you are not familiar with Windows, then we suggest using the **Help** feature (available from the Windows desktop **Start** menu) as you work through this guide.

General Information

This chapter identifies all of the hardware and software components required to implement communication between the PIDAT driver in Studio and a target device using the OSI's equipment protocol over TCP/IP.

The information is organized into the following sections:

- Device Specifications
- Network Specifications
- Driver Characteristics
- Conformance Testing

Device Specifications

To establish communication, your target device must meet the following specifications:

- **Manufacturer:** OSI (PI Software)

For a description of the device(s) used to test driver conformance, see "Conformance Testing" on the next page.

Network Specifications

To establish communication, your device network must meet the following specifications:

- **Device Communication Port:** Ethernet Port
- **Physical Protocol:** Ethernet/TCP-IP
- **Specific PC Board:** Any TCP/IP Adapter (Ethernet board)

Driver Characteristics

The PIDAT driver package consists of the following files, which are automatically installed in the **/DRV** subdirectory of Studio:

- **PIDAT.INI:** Internal driver file. *You must not modify this file.*
- **PIDAT.MSG:** Internal driver file containing error messages for each error code. *You must not modify this file.*
- **PIDAT.PDF:** This document, which provides detailed information about the PIDAT driver.
- **PIDAT.DLL:** Compiled driver.

 **Note:**

You must use Adobe Acrobat® Reader™ to view the **PIDAT.PDF** document. You can install Acrobat Reader from the Studio installation CD, or you can download it from Adobe's Web site.

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

- Windows NT/2000/XP
- Windows CE

For a description of the operating systems used to test driver conformance, see “Conformance Testing” below.

Conformance Testing

The following hardware/software was used for conformance testing:

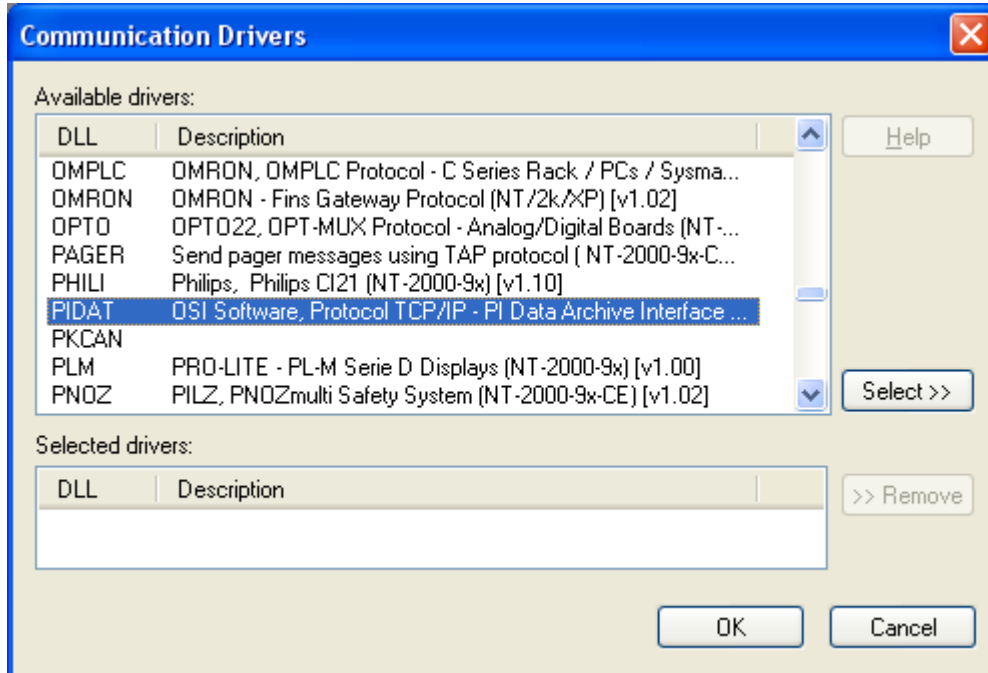
- **Driver Configuration:**
 - **Operation System (development):** Windows XP
 - **Operation System (target):** Windows XP / Windows CE v4.2
 - **Studio Version:** 6.1
 - **Driver Version :** 1.0
 - **Cable:** Ethernet Cable

Driver Version	Studio Version	Operating System (development)	Operating System (runtime)
2.21	6.1 + SP1	WinXP + SP2	<ul style="list-style-type: none">▪ WinXP + SP2▪ WinCEv5.00

Selecting the Driver

When you install Studio, all of the communication drivers are automatically installed in the `\DRV` subdirectory but they remain dormant until manually selected for specific applications. To select the PIDAT driver for your Studio application:

1. From the main menu bar, select **Insert** → **Driver** to open the *Communication Drivers* dialog.
2. Select the **PIDAT** driver from the *Available Drivers* list, and then click the **Select** button.



Communication Drivers Dialog

3. When the **PIDAT** driver is displayed in the **Selected Drivers** list, click the **OK** button to close the dialog. The driver is added to the *Drivers* folder, in the *Comm* tab of the Workspace.

Note:
It is necessary to install the PI software on your computer to enable communication between the host and the device. You need to install the PI-API32 API to accomplish the communication. (The driver uses four files: PI-API32.dll, PI-API32.lib, PI-LOG32.dll and PI-LOG32.lib).

Attention:
For safety reasons, you must take special precautions when installing any physical hardware. Please consult the manufacturer's documentation for specific instructions.

Configuring the Device

We recommend using the software’s suggested configuration.

The selected driver and the PI software should be both properly configured. All runtime communication is handled within your Studio application project.

Configuring the Driver

Once you have selected the PIDAT driver in Studio, you must properly configure it to communicate with your target device. First, you must set the driver’s communication settings to match the parameters set on the device. Then, you must build driver worksheets to associate database tags in your Studio application with the appropriate addresses (registers) on the device.

Configuring the Communication Settings

The communication settings are described in detail in the “Communication” chapter of the Studio *Technical Reference Manual*, and the same general procedures are used for all drivers. Please review those procedures before continuing.

Configuring the Driver Worksheets

Each selected driver includes a Main Driver Sheet and one or more Standard Driver Worksheets. The Main Driver Sheet is used to define tag/register associations and driver parameters that are in effect at all times, regardless of application behavior. In contrast, Standard Driver Worksheets can be inserted to define additional tag/register associations that are triggered by specific application behaviors.

The configuration of these worksheets is described in detail in the “Communication” chapter of the Studio *Technical Reference Manual*, and the same general procedures are used for all drivers. Please review those procedures before continuing.

For the purposes of this document, only PIDAT driver-specific parameters and procedures are discussed here.

MAIN DRIVER SHEET

The PIDAT driver only uses Standard Driver Sheet for a while.

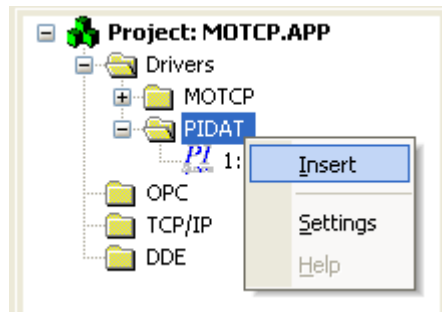
STANDARD DRIVER WORKSHEET

When you select the PIDAT driver and add it to your application, it has only a Main Driver Sheet by default (see previous section). However, you may insert additional Standard Driver Worksheets to define tag/register associations that are triggered by specific application behaviors. Doing this will optimize communication and improve system performance by ensuring that tags/registers are scanned only when necessary – that is, only when the application is performing an action that requires reading or writing to those specific tags/registers.

Note:
We recommend configuring device registers in sequential blocks in order to maximize performance.

To insert a new Standard Driver Worksheet:

1. In the *Comm* tab, open the *Drivers* folder and locate the *PIDAT* subfolder.
2. Right-click on the *PIDAT* subfolder, and then select **Insert** from the pop-up menu:



Inserting a New Worksheet

A new PIDAT driver worksheet is inserted into the *PIDAT* subfolder, and the worksheet is opened for configuration:

Header

Description: DRIVER STANDARD 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: LOGIN Min: Max:

Tag Name	Address	Div	Add
*			
*			
*			

Body

PIDAT Driver Worksheet

 **Note:**
Worksheets are numbered in order of creation, so the first worksheet is **PIDAT001.drv**.

Most of the fields on this worksheet are standard for all drivers; see the “Communication” chapter of the *Technical Reference Manual* for more information on configuring these fields. However, the **Header** and **Address** fields use syntax that is specific to the PIDAT driver.

3. Configure the **Header** fields as follows:

Specify the address of the first register of a block of registers on the target device. The addresses declared in the *Body* of the worksheet are simply offsets of this **Header** address. When Read/Write operations are executed for the entire worksheet (see **Read Trigger** and **Write Trigger** above), it scans the entire block of registers from the first address to the last.

The **Header** field uses the following syntax:

`<Type>(<server, user, password>)`

Example — `LOGIN(servername, john, abcde)`

Where:

- `<Type>` is the server name or the word “LOGIN”.
- `<server, user, password>` if the Type field have the string “LOGIN”, you should specify the server, user and password separated by commas. (optional)

You can also specify an indirect tag (e.g. `{header}`), but the tag that is referenced must follow the same syntax and contain a valid value.

4. Configure the **Address** field using the following syntax:

`<TagName>`

Examples — `INT1`

Where:

- `<TagName>`: The tag’s name on the PI Software.

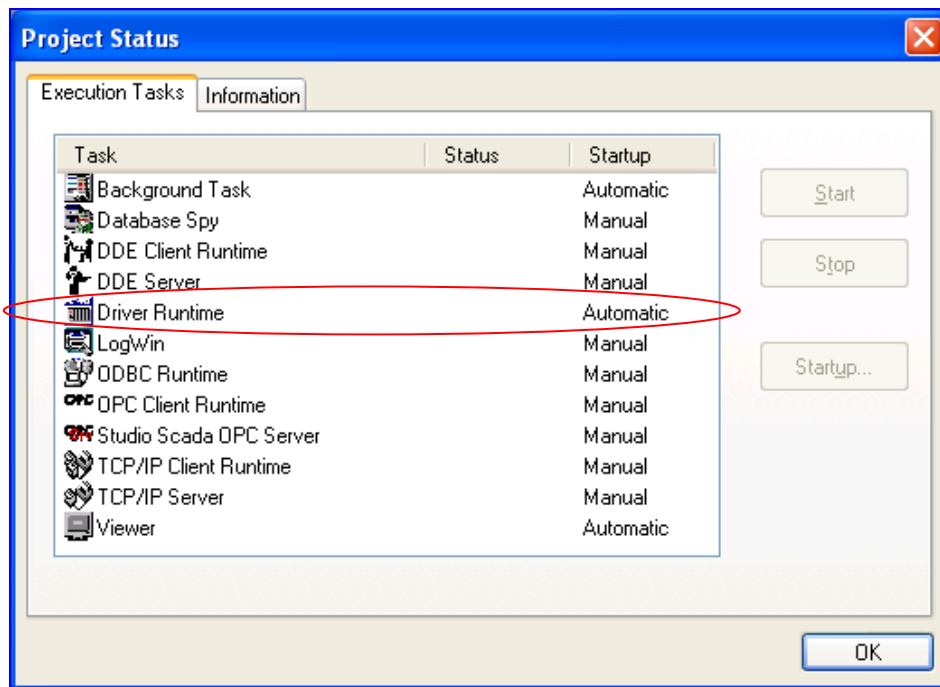
For more information about the device registers and addressing, please consult the manufacturer’s documentation.

Executing the Driver

By default, Studio will automatically execute your selected communication driver(s) during application runtime. However, you may verify your application’s runtime execution settings by checking the *Project Status* dialog.

To verify that the the communication driver(s) will execute correctly:

1. From the main menu bar, select **Project** → **Status**. The *Project Status* dialog displays:



Project Status Dialog

2. Verify that the *Driver Runtime* task is set to **Automatic**.
 - If the setting is correct, then proceed to step 3 below.
 - If the **Driver Runtime** task is set to **Manual**, then select the task and click the **Startup** button to toggle the task’s *Startup* mode to **Automatic**.
3. Click **OK** to close the *Project Status* dialog.
4. Start the application to run the driver.

Troubleshooting

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 Station	Wrong IP Address	You must specify the IP Address using the appropriate syntax—four fields, with up to three characters in each field, separated by periods.
2	Invalid Header field	Invalid tag value in the Header field	Specify a valid tag value in the Header field.
3	Invalid Address field	Invalid Address	<ul style="list-style-type: none"> ▪ Check the initial address in the <i>Driver</i> worksheet. ▪ Check the Holding register in the <i>Driver</i> worksheet with bit configuration. This parameter cannot execute write triggers—it executes “Write on Tag Change” only. ▪ Retype the address in the <i>Driver</i> worksheet.
4	Invalid Block size	Offset is greater than the maximum allowed. The maximum offset is usually 64.	Specify a valid offset or create a new <i>Driver</i> worksheet.
5	Protocol Error	The driver is not following the protocol standards.	Check the protocol documentation and the buffers sent to the devices.
6	Checksum Error	The driver is receiving buffers with wrong sizes.	Check the protocol documentation and the buffers received from the devices.
8	Cannot Read API Version	The driver is not reading the data from the Library API.	Check the Software and the files localization
9	Error on open driver	The driver doesn't start the execution.	Check the driver and the Studio version.
10	Error on try to get PI Point of the API	The driver doesn't get the PI Point.	Check the address values. Check if the API is set correctly.
11	Error on try to read tags from PI Server	Wrong Address.	Use “Telnet” or “Ping” tools to check your network configuration and try to find the PLC with the computer on which you are running Studio.
12	Error on try to write tags from PI Server	Wrong Address.	Use “Telnet” or “Ping” tools to check your network configuration and try to find the PLC with the computer on which you are running Studio.
13	Error on try to connect to the PI Server	Wrong TCP/IP network configuration	Use “Telnet” or “Ping” tools to check your network configuration, and try to find the PLC with the computer on which you are running Studio.

If you are unable to establish communication between Studio and the PI software, then try instead to establish communication using the software. Quite often, communication is interrupted by a hardware or cable problem or by a device configuration error. If you can successfully communicate using the programming software, then recheck the driver's communication settings in Studio.

To test communication between Studio and the device, 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

A sample application that employs the PIDAT driver is provided on the Studio installation CD. We strongly recommend that you use this sample application to test the driver *before* you develop your own applications, for the following reasons:

- To better understand the information and instructions provided in this document;
- To verify that your driver configuration is working satisfactorily with the target device; and
- To ensure that the all of hardware used in the test (i.e. the device, adapter, cable, and PC) is functioning safely and correctly.

 **Note:**

The following instructions assume that you are familiar with developing project applications in Studio. If you are not, then please review the relevant chapters of the Studio *Technical Reference Manual* before proceeding.

To use the sample application:

1. Configure the device's communication settings according to the manufacturer's documentation.
2. Run Studio.
3. From the main menu bar, select **File → Open Project**.
4. Insert the Studio installation CD and browse it to find the sample application. It should be located in the directory `\COMMUNICATION EXAMPLES\PIDAT`.
5. Select and open the sample application.
6. Configure and test the driver, as described in the rest of this document.

When you have thoroughly tested the driver with your target device, you may proceed with developing your own Studio application projects.

 **Tip:**

You can use the sample application screen as the maintenance screen for your own applications.

Revision History

Doc. Revision	Driver Version	Author	Date	Description of changes
A	1.00	Plinio M. Santana	Oct/03/2007	Document created.
B	1.00	Andre Bastos	Mar/31/2010	Changed the documentation only. No modifications in the driver