MPIAD Communication Driver

Driver for Serial Communication
with Devices Using MPI Green Cable Protocol

Contents

Introduction

The MPIAD driver enables communication between the Studio system and devices using the MPI Green Cable protocol communicating over RS232, according to the specifications discussed in this document.

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

- **Introduction**: Provides an overview of the MPIAD driver documentation.
- **General Information**: Provides information needed to identify all the required components (hardware and software) used to implement communication between Studio and the MPIAD driver.
- **Installing the Driver**: Explains how to install the MPIAD driver.
- **Configuring the Driver**: Explains how to configure the MPIAD 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 MPIAD 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 *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 MPIAD driver and the ADAM-8000 series PLC.

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**: Advantech
- **Compatible Equipment**: ADAM8000 series
- **Programmer Software**: WinPLC7

For a list of the devices used for conformance testing, see "Conformance Testing."

Link Characteristics

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

- **Device Communication Port**: MP2I
- **Physical Protocol**: RS-232
- **Logic Protocol**: MPI Green Cable
- **Device Runtime Software**: None
- **Specific PC Board**: None

Driver Characteristics

The MPIAD driver is composed of the following files:

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

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

- Windows 9x
- Windows 2000
- Windows NT

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

The MPIAD driver supports the following registers:

Register Type	Length	Write	Read	Bit	Integer	Float	DWord
M (Memory Bits)	1 Byte	•	•	•	•	•	•
I or E (Inputs)	1 Byte	•	–	•	•	•	•
Q or A (Outputs)	1 Byte	–	•	•	•	•	•
DB (Global Data)	1 Byte	•	•	•	•	•	•
C or Z (Counters)	2 Bytes	•	•		•		
T (Timers)	2 Bytes	•	•	–	•	–	–

Conformance Testing

The following hardware/software was used for conformance testing:

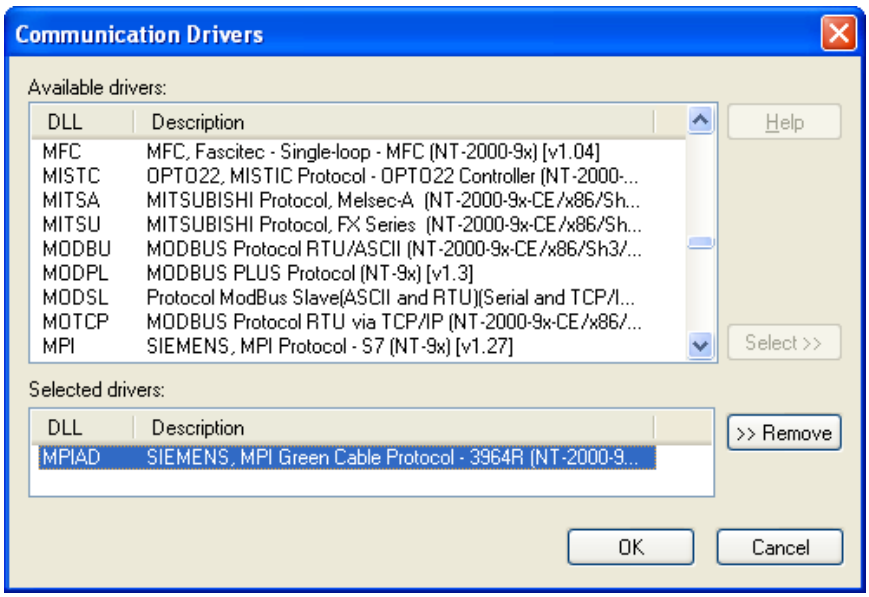
- **Equipment**: PLC ADAM-8222 CPU 208
- **Driver Configuration**:
  - **PLC Program:** WinPLC7
  - **Com Port**: 1
  - **Baud Rate**: 38400
  - **Data Bits**: 8
  - **Stop Bits**: 1
  - **Parity**: Odd
- **Cable**: MPI Green Cable
- **Operating System (*development*):** Windows XP
- **Operating System (*runtime*):** Windows XP
- **Studio Version**: 5.1
- **Driver Version**: 1.00

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



Communication Drivers Dialog

5. When the **MPIAD** 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. However, to download the custom program to your device, you must install the WinPLC7 programmer software. Consult your WinPLC7 programmer software documentation for installation instructions

> **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 MPIAD driver, you must configure the driver. Configuring the MPIAD driver is done in two parts:

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

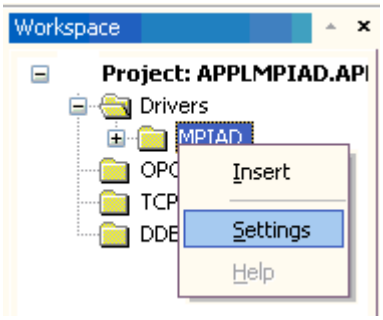
  Worksheets are divided into two sections, a *Header* and a *Body*. The fields contained in these two sections are standard for all communications drivers — except the **Station**, **Header**, and **Address** fields, which are driver-specific. This document explains how to configure the **Station**, **Header**,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

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

Setting the Communication Parameters

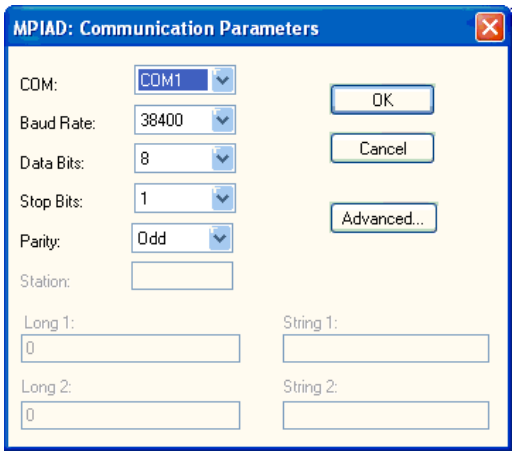
Use the following steps to configure the communication parameters, which are valid for all driver worksheets configured in the system:

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 *MPIAD* subfolder and when the pop-up menu displays, select the **Settings** option:



Select Settings from the Pop-Up Menu

The *MPIAD: Communications Parameters* dialog displays:



Communication Parameters Dialog

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

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

> **Notes:**
> - Do not change any of the other *Advanced* parameters at this time. You can consult the Studio *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, driver, and the host. You must be familiar with the DCE specifications before adjusting these configuration parameters.

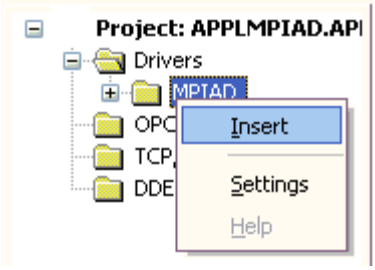
Configuring the Driver Worksheets

This section explains how to configure the *MAIN* and *STANDARD DRIVER SHEETs* (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 *Body* section.

Configuring the *STANDARD DRIVER SHEET*

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

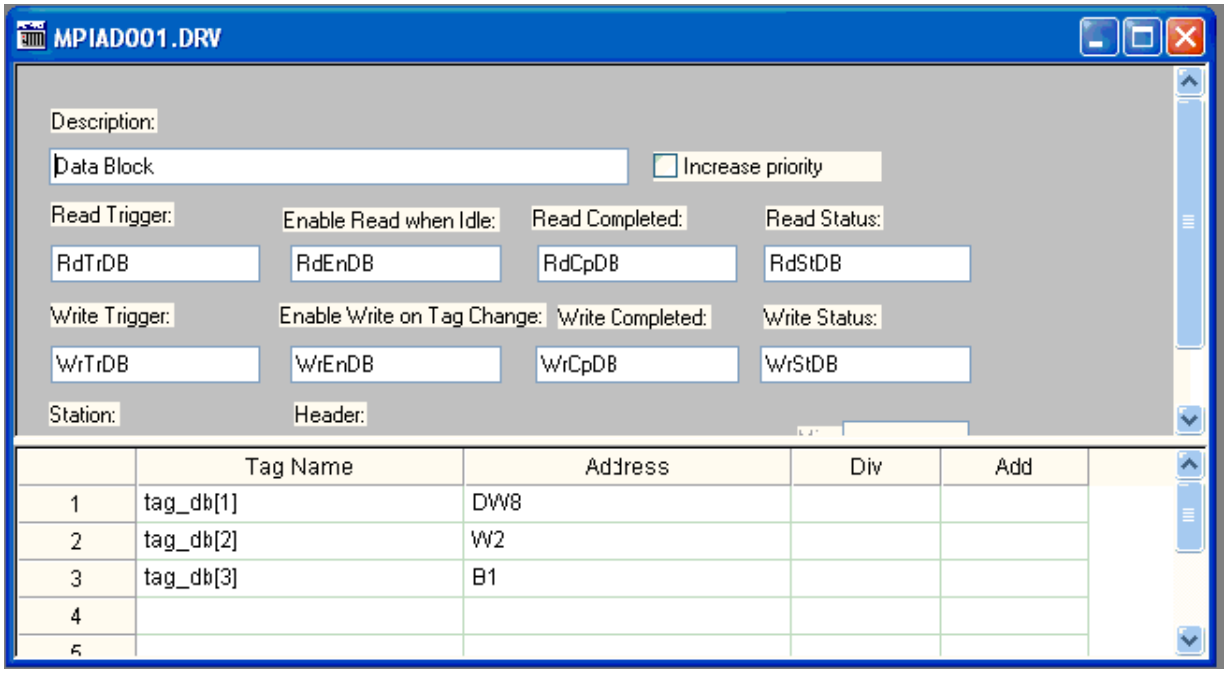
1. From the Studio development environment, select the **Comm** tab, located below the *Workspace* pane.
1. In the *Workspace* pane, expand the *Drivers* folder and right-click the <*Driver Name*> subfolder.
2. When the pop-up menu displays, select the **Insert** option:



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):



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 Technical Reference Manual*.

4. Use the following information to complete the **Station**, **Header**, and **Address** fields on this worksheet.
   - **Station** field: Not used for this driver.
   - **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. These variables must comply with the following syntax:
     - For Flags, Timers, Counters, Inputs, and Outputs:
       `<Type>:<AddressReference>` (for example: **`M:1`**)
     - For Data-Blocks:
       `<Type><TypeGroup>:<AddressReference>` (for example: **`DB2:1`**)

Where:

– **Type** is the register type. (M, I or E, Q or A, DB, C or Z and T)

– **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 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, or you will get an Invalid Header error.

– **Address** field: Use the information in the next table 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:

<Format><AddressOffset>.[Bit] (for example: **`B10`**, **`DW20`**, **`W40`**, **`B10.5`**)

Where:

– ***<Format>*** defines how Studio treats the value read or written from/to the device (**B**=Byte, **W**=Word, **DW**=Dword, **F**=Float); ***<Format>*** is not necessary for types **C** and **T**.

– ***<AddressOffset>*** is a parameter added to the **AddressReference** parameter (configured in the **Header** field) to compose the group address configured in the **Header** field.

Sample Address Configuration		
Address on the Device	**Header Field**	**Address Field**
M (Word 5 = Byte 5 / Byte 6)	M:0	W5
	M:5	W0
	M:3	W2
M (Byte 5)	M:0	B5
	M:5	B0
	M:1	B4
M (Byte 6)	M:0	B6
	M:6	B0
	M:3	B3
M (String 3, size 10)	M:0	S3:10
	M:3	S0:10
	M:1	S2:10
O (Word 2 = Byte 2 / Byte 3)	O:0	W2
	O:2	W0
	O:1	W1
I (Byte 2)	I:0	B2
	I:2	B0
	I:1	B1
DB5 (Byte 3)	DB5:0	B3
	DB5:3	B0
	DB5:2	B1
DB5 (Word 7 = Byte 7 / Byte 8)	DB5:0	W7
	DB5:7	W0
	DB5:4	W3

Sample Address Configuration		
Address on the Device	**Header Field**	**Address Field**
C (3)	C:2	1
	C:0	3
	C:3	0
T(10)	T:2	8
	T:5	5
	T:0	10

➲ **Attention:**

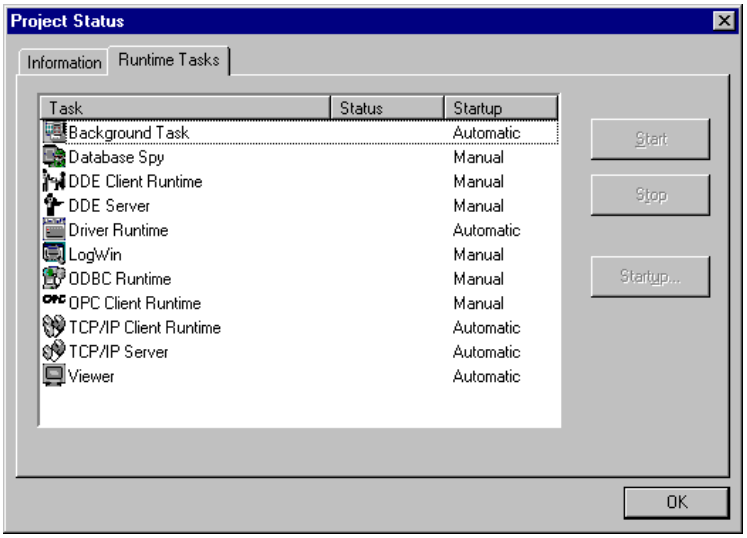
You must not configure a range of addresses greater than the maximum block size (date buffer length) supported by each PLC within the same worksheet. The maximum data buffer length for this driver is 246 bytes per *STANDARD DRIVER SHEET*.

Executing the Driver

After adding the MPIAD 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

3. 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**.
4. Click **OK** to close the *Project Status* dialog.
5. Start the application to run the driver.

Troubleshooting

If the MPIAD 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.
10	Block size error	Specified a block size larger than what the protocol can read/write	Change the driver configuration to read less than 246 bytes.
11	Checksum error	Protocol error	Contact your Studio technical support representative.
12	Create line protocol error	Error in the communication when trying to start communicating with the device.	Check the equipment state.
13	Create connection error	Error in the communication when trying to connect with the device.	Check the communication parameters for the device and the Studio software.
14	Close connection error	Error in the communication when trying to close the connection with the device.	Contact your Studio technical support representative.
15	Close line protocol error	Error in the communication when trying to close the line protocol with the device.	Contact your Studio technical support representative.
16	Driver Sheet Empty	Trying to read an empty driver sheet.	Fill the driver sheet.
17	DLE RX error	Protocol Error.	Contact your Studio technical support representative.
18	DLE STX RX error	Protocol Error	Contact your Studio technical support representative.
19	ACK RX error	Protocol Error	Contact your Studio technical support representative.
20	Dados RX error	Protocol Error	Contact your Studio technical support representative.
21	Protocol error	Protocol Error	Contact your Studio technical support representative.
22	Invalid Address	Invalid Address	▪ Check the initial address in the Driver Worksheet. ▪ Check the Holding register in the Driver Worksheet with bit configuration. ▪ Retype the address in the Driver Worksheet.

⇨ **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/<*Driver Name*>** directory. We strongly recommend that you check if there is a sample application for this driver and use it to test the 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.

> **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 RC	Lidiane A. Moreira	Jul/23/2003	First Version
B	1.00	Lidiane A. Moreira	Oct/03/2003	Add information about timers and counters