OPMMP Communication Driver

Driver for Ethernet Communication
with OPTO-22 Devices using OPTO-MMP protocol

Contents

Introduction

This document will help you to select, configure and execute the OPMMP 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 OPMMP driver in the Studio system.
- **Configuring the Driver**: Explains how to configure the OPMMP driver in the Studio system, including how to associate database tags with device registers.
- **Executing the Driver**: Explains how to execute the OPMMP 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 OPMMP 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 7/XP/Vista 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 OPMMP driver in Studio and remote devices.

The information is organized into the following sections:

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

Device Specifications

You can use this driver to communicate with any device using the OPTO-MMP protocol. (The devices used for conformance testing are listed on the next page.)

Network Specifications

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

- **Device Communication Port**: Ethernet
- **Physical Protocol**: TCP/IP
- **Logic Protocol**: OPTO-MMP
- **Device Runtime Software**: None
- **Specific PC Board**: None
- **Adapters/Converters**: None
- **Cable Wiring Scheme**: None

Driver Characteristics

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

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

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

- Windows 7/XP/Vista
- Windows CE 4.x, 5.x, 6.x

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:

For Ethernet Tests

- **TCP/IP Port**: 2001
- **Protocol**: OPTO-MMP
- **Station**: PLC IP Address
- **Cable**: Ethernet Cable

Driver Version	Studio Version	Operating System	Equipment
1.0	6.1+SP6	Windows XP SP3	SNAP PAC R1 Firmware vR8.5a

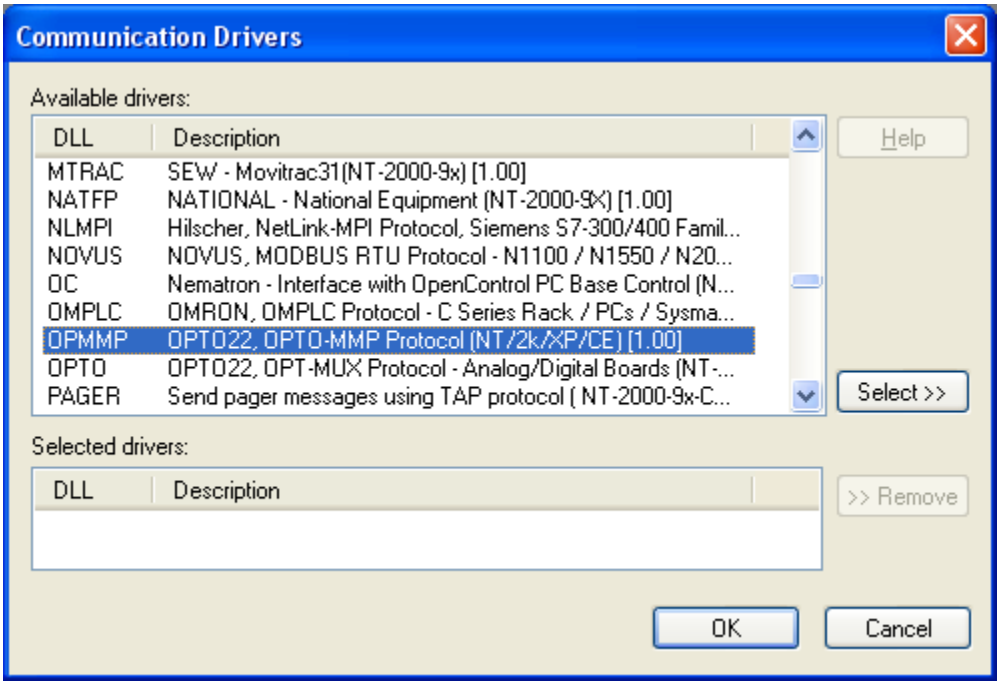
The OPMMP driver supports the following point types:

Header Information			Operations		Field Type			
Point Name	Register	Field	Write	Read	Boolean	Integer	Float	String
Analog Point	AMINFO	Value	•	•	–	–	•	–
		Counts	•	•	–	–	•	–
		Min	–	•	–	–	•	–
		Max	–	•	–	–	•	–
Analog Point Value	APVALUE		•	•	–	–	•	–
Analog Point Counts	APCOUNTS		•	•	–	–	•	–
Digital Point	DMINFO	State	•	•	•	–	–	–
		On-Latch	–	•	•	–	–	–
		Off-Latch	–	•	•	–	–	–
		Counter State	–	•	•	–	–	–
		Counter Data	–	•	–	•	–	–
Digital Point State	DPSTATE		•	•	•	–	–	–
Digital Point Counter Data	DPCDATA		–	•	–	•	–	–
Digital Point Read & Clear	DPRC	Counter	–	•	–	•	–	–
		On-Latch	–	•	•	–	–	–
		Off-Latch	–	•	•	–	–	–
Scratch Pad Bits	SPB		•	•	•	–	–	–
Scratch Pad Integers	SPI		•	•	–	•	–	–
Scratch Pad Floats	SPF		•	•	–	–	•	–
Scratch Pad Strings	SPS		•	•	–	–	–	•

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 OPMMP driver for your Studio application:

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



Communication Drivers Dialog

3. When the **OPMMP** 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.

> ➲ **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 Driver

Once you have selected the OPMMP driver in Studio, you must properly configure it to communicate with your target 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.

The OPMMP driver does not require any specific communication parameters.

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.

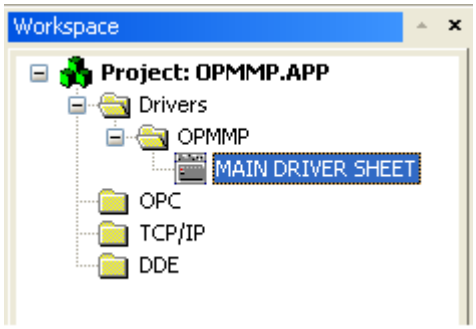
The configuration of these worksheets is described in detail in the "Communication" chapter of the Studio *Technical Reference Manual*, and the same general procedures are used for all drivers. Please review those procedures before continuing.

For the purposes of this document, only OPMMP driver-specific parameters and procedures will be discussed here.

MAIN DRIVER SHEET

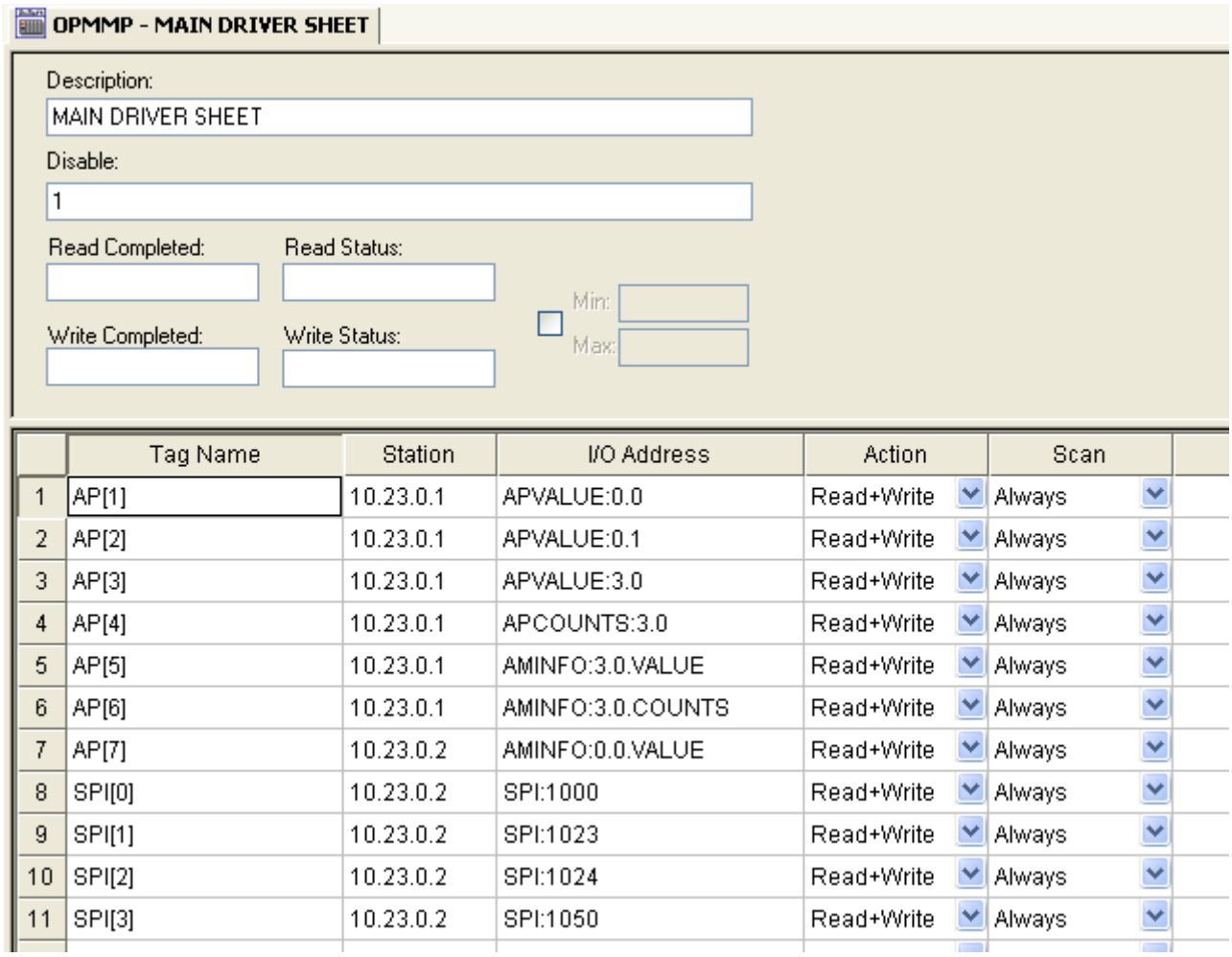
When you select the OPMMP driver and add it to your application, Studio automatically inserts the *Main Driver Sheet* in the *OPMMP* driver subfolder. To configure the Main Driver Sheet:

1. Select the *Comm* tab in the *Workspace* pane.
2. Open the *Drivers* folder, and then open the *OPMMP* subfolder:



Main Driver Sheet in the OPMMP Subfolder

3. Double-click on the **MAIN DRIVER SHEET** icon to open the following worksheet:



	Tag Name	Station	I/O Address	Action	Scan
1	AP[1]	10.23.0.1	APVALUE:0.0	Read+Write	Always
2	AP[2]	10.23.0.1	APVALUE:0.1	Read+Write	Always
3	AP[3]	10.23.0.1	APVALUE:3.0	Read+Write	Always
4	AP[4]	10.23.0.1	APCOUNTS:3.0	Read+Write	Always
5	AP[5]	10.23.0.1	AMINFO:3.0.VALUE	Read+Write	Always
6	AP[6]	10.23.0.1	AMINFO:3.0.COUNTS	Read+Write	Always
7	AP[7]	10.23.0.2	AMINFO:0.0.VALUE	Read+Write	Always
8	SPI[0]	10.23.0.2	SPI:1000	Read+Write	Always
9	SPI[1]	10.23.0.2	SPI:1023	Read+Write	Always
10	SPI[2]	10.23.0.2	SPI:1024	Read+Write	Always
11	SPI[3]	10.23.0.2	SPI:1050	Read+Write	Always

Main Driver Sheet

Most of the fields on this sheet are standard for all drivers; see the "Communication" chapter of the *Technical Reference Manual* for more information on configuring these fields. However, the **Station** and **I/O Address** fields use syntax that is specific to the OPMMP driver.

4. For each table row (i.e. each tag/register association), configure the **Station** and **I/O Address** fields as follows:

   - **Station** field

     - For *Ethernet*, use the following syntax:

       <IP Address>:<Opt Port Number>

     Example — **`192.168.0.52`** or **`192.168.0.52:2150`**

     Where:

     – ***<IP Address>*** is the IP address of the device on the Ethernet network.

– ***<opt Port Number>*** optional TCP/IP Port number. If you do not configure this parameter, the default value of **2001** will be used

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

> ➲ **Attention:**
> You cannot leave the Station field **blank**

- **I/O Address** field — Specify the name or address of the associated device, using the following Syntax:

  <Type>:<Address>

  Where:

  – ***<Type>*** : Register type. Valid values are **APVALUE**, **APCOUNTS**, **AMINFO**, **DPSTATE**, **DPCDATA**, **DMINFO**, **DPRC**, **SPB**, **SPI**, **SPF**, **SPS**.

  – ***<Address>*** : Address of the device register.

  The valid addresses and the syntax for each of the register types are explained on the section below. However, the precautions on valid ranges and max block sizes are not applicable to the Main Driver Worksheet, once the groups are generated based on the addresses filled by the user. For the header **DPRC** where memory access implies on clearing it, the groups are generated taking also in consideration that they must be contiguous, to avoid erasing undesired areas.

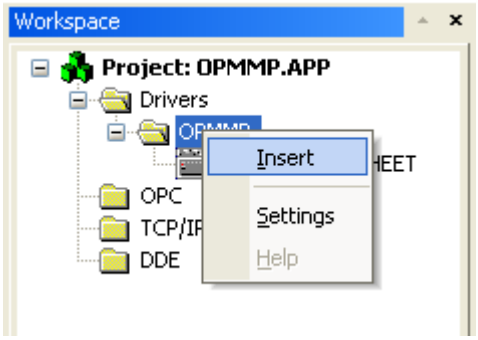
STANDARD DRIVER WORKSHEET

When you select the OPMMP 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.

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.

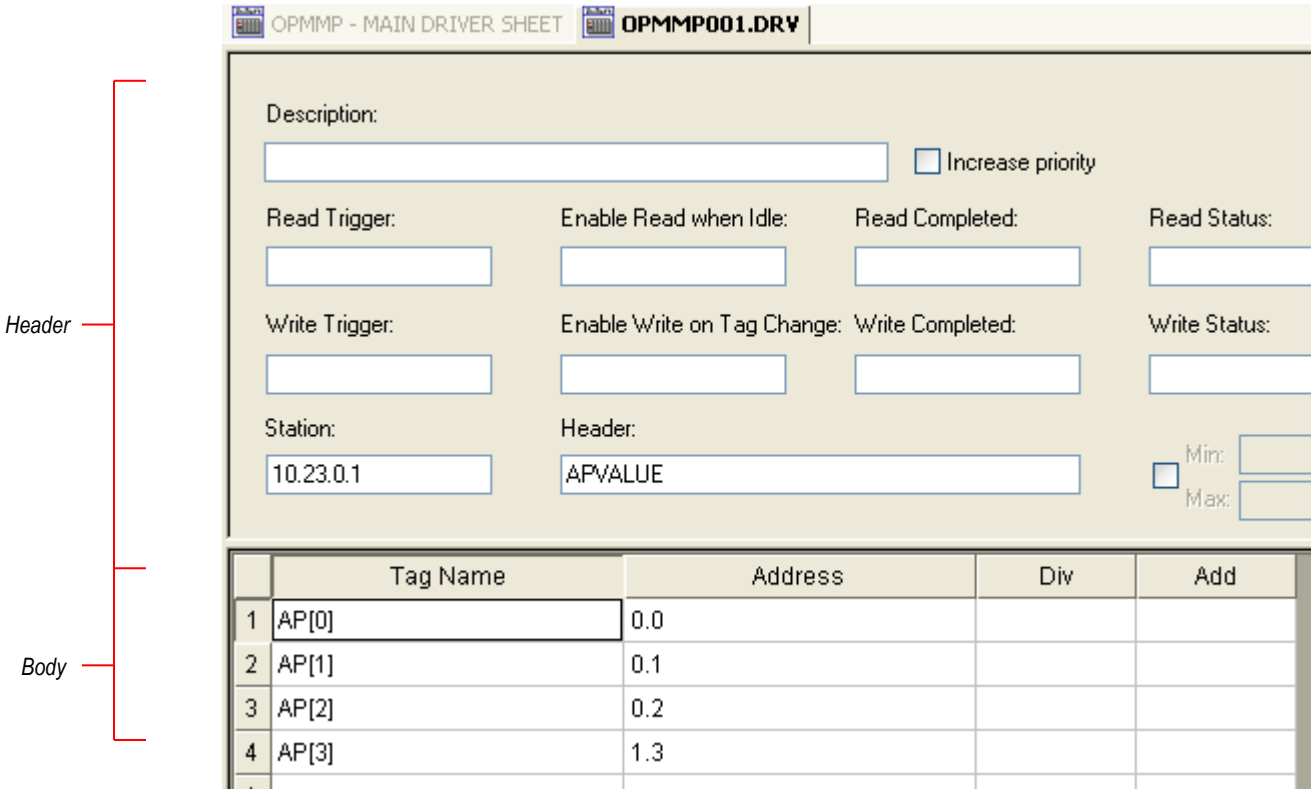
To insert a new driver worksheet:

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



Inserting a New Worksheet

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



OPMMP Driver Worksheet

> **Note:**
> Worksheets are numbered in order of creation, so the first worksheet is **`OPMMP001.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 **Header** and **Body** (as noted on the above picture) fields use syntax that is specific to the OPMMP driver.

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

- **Station** field

  - For *Ethernet*, use the following syntax:

    <IP Address>:<Opt Port Number>

  Example — **192.168.0.52** or **192.168.0.52:1:502**

  Where:

  – ***<IP Address>*** is the IP address of the device on the Ethernet network.

  – ***<opt Port Number>*** is the optional TCP/IP Port number. If you do not configure this parameter, the default value of **2001** will be used

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

  > ➲ **Attention:**
  > You cannot leave the Station field **blank**

- **Header** field: Specify the address of the first register of a block of registers on the target device. 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>

  Examples — **AMINFO**

  — **DPSTATE**

  Where:

  – ***<Type>*** is the register type (**APVALUE**, **APCOUNTS**, **AMINFO**, **DPSTATE**, **DPCDATA**, **DMINFO**, **DPRC**, **SPB**, **SPI**, **SPF**, **SPS**).

  After you edit the **Header** field, Studio checks the syntax to determine if it is valid. If the syntax is invalid, then Studio automatically inserts a default value of **APVALUE**.

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

  The following table lists all of the data types and address ranges that are valid for the OPMMP driver:

Type	Syntax	Comments
Analog Point Values	**APVALUE**	Physical Analog Input or Output. Reads/writes only the value information of points
Analog Point Counts	**APCOUNTS**	Physical Analog Input or Output. Reads/writes only the counts information of points.
Analog Point Module Info	**AMINFO**	Physical Analog Input or Output. Reads/writes the module information.
Digital Point States	**DPSTATE**	Physical Digital Input or Output. Reads/write only the state of points.
Digital Point Counter Data	**DPCDATA**	Physical Digital Input or Output. Reads only the counter data information of points.
Digital Point Module Info	**DMINFO**	Physical Digital Input or Output. Read/writes the module information.

Type	Syntax	Comments
Digital Point Read & Clear	**DPRC**	Physical Digital Input or Output. Reads and clear values of counters and on/off latches.
Scratch Pad Bits	**SPB**	Memory for transferring bits.
Scratch Pad Integers	**SPI**	Memory for transferring 32-bit integers.
Scratch Pad Floats	**SPF**	Memory for transferring 32-bit floats.
Scratch Pad Strings	**SPS**	Memory for transferring strings with at most 128 characters.

4. For each table row (i.e., each tag/register association), configure the **Address** field using the following syntax:

   For registers of type **AMINFO** and **DMINFO**, use the syntax:

   <ModuleNumber>.<PointNumber>.<Field>

   Where:

   – ***<ModuleNumber>***: the number of the module configured on the device.

   – ***<PointNumber>*** : the number of the point of the corresponding module

   – ***<Field>***: the field associated with the point to be accessed. For the valid fields of each header, see the following table:

Header	Field	Meaning
AMINFO	VALUE	Value in scaled units of the analog point
AMINFO	COUNTS	Value in counts of the analog point
AMINFO	MIN	Minimum value in scaled units
AMINFO	MAX	Maximum value in scaled units
DMINFO	STATE	State (On/Off) of the digital point
DMINFO	ON-LATCH	
DMINFO	OFF-LATCH	
DMINFO	COUNTER-STATE	State (On/Off) of the counter
DMINFO	COUNTER-DATA	Counts of the digital point

   Examples:

   – **0.1.VALUE** – The value in scaled units of the analog point 1 in module 0

   – **1.1.STATE** – The state of the digital point 1 in module 1

   – **1.1.COUNTER-STATE** – The state of the counter of the digital point 1 in module 1

   – **3.1.COUNTS** – The value in counts of the analog point 1 in module 3

   For registers **APVALUE**, **APCOUNTS**, **DPSTATE** and **DPCDATA**, use the syntax:

   <ModuleNumber>.<PointNumber>

Examples:

- **0.1** – The value/counts/state/counter data of point 1 in module 0
- **1.1** – The value/counts/state/counter data of point 1 in module 1
- **3.1** – The value/counts/state/counter data of point 1 in module 3

For the other registers, use the syntax:

<Number>

Where:

- ***<Number>***: the number of the register associated with the register in memory

Examples:

- For a header **SPI/SPF**:
  - 1 – The integer/float stored in position 1
  - 10239 – The integer/float stored in position 10239
- For a header **SPB**:
  - 1 – The bit in position 1
  - 63 – The bit in position 63
- For a header **SPS**:
  - 1 – The string in position 1
  - 63 – The string in position 63

Attention:

The protocol does not support reading certain ranges of addresses by using a single command. For instance, it is not possible to read the **SPI** addresses 1023 and 1024 in a single block.

For this reason, you cannot have these addresses configured in the same standard driver sheet. If you try such configuration the driver will return error code **2**, with the message **Invalid address range** indicating that you need to split the addresses into different driver groups.

Notice that the main driver sheet does not have this problem once it will break the addresses into different groups automatically.

The table below lists all the ranges that cannot coexist in the same standard driver sheet:

Valid ranges for types SPI and SPD when using Standard Driver Sheet:

Header Field	Address Field
SPI	0000~1023
SPI	1024~10239
SPF	0000~1023
SPF	1024~10239

Also, the types **SPB** and **SPS** are limited on range 0~63 on its address field.

You must not configure a range of addresses greater than the maximum block size (data buffer length)

supported by the target device. The block size for the different registers are:

Header Type	Max Block Size (number of registers)
APVALUE, APCOUNTS, DPCDATA, SPI and SPF	508
DPSTATE	64
SPB	64
SPS	15

The headers **APVALUE**, **APCOUNTS**, **DPCDATA** and **DPSTATE** are organized with data sequential on memory. That allows reading more information with one round-trip to the PLC. However, these banks allow only up to 3 points per module. When necessary, for the analog points, to read more than 3 points of a single module, the user must use the **AMINFO** header.

Since the **AMINFO** and **DMINFO** headers are organized differently than the others in memory, theirs block size restrictions are based on the number of points and modules. For the **AMINFO** header, its limitation is on 1 module per sheet with up to 31 points. This occurs due to reserved areas on device's memory for future growth. The **DMINFO** header limitation is on 31 sequential points, where these are ordered first on module number, then on point, where each module has at most 3 points. That is, the addresses for points `1.3` and `2.0` are sequential on memory, as well as for `1.2` and `1.3`. The **DPRC** header also has reserved memory areas between points. For this header, the limitation is on 84 sequential points. However, the user must consider that each module has reserved space for 64 points. That is, a group with points `0.1` and `1.0` is valid, and accounts for a read of 64 (63+1) points, whereas reading `0.1` and `2.1` is invalid, as it accounts for a read of 129 (63+64+2) points.

Configuring sheets with more registers than advised here yields an **Invalid block size** error.

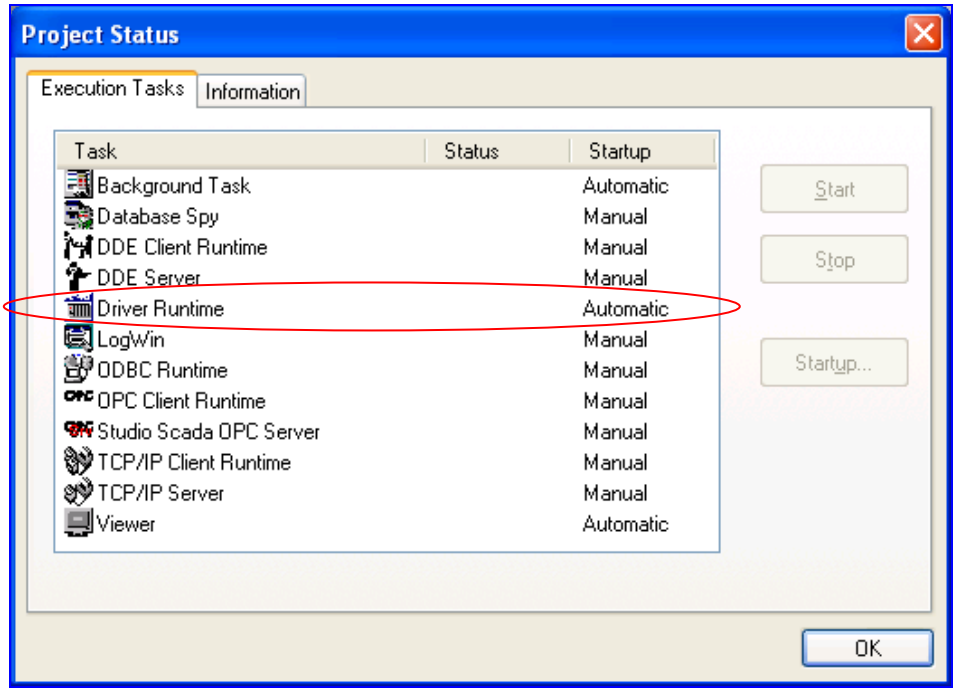
The user must also pay attention to the fact that not all fields of headers **AMINFO** and **DMINFO** are writable, as well as **DPCDATA** and **DPCR** are also not. As shown before, for header **AP** only the `VALUE` and `COUNTS` field may be written as for header **DP** only the `STATE` field may be written. The **DPCR** fields' are automatically zeroed when a read is performed on them. The user must pay attention that the standard driver sheet does not separate reads of fields that are not contiguous, and that those unused addresses are also read and cleared, and its values are discarded.

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 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

If the OPMMP driver fails to communicate with the target device, then the database tag(s) that you configured for the **Read Status** or **Write Status** fields of the Standard Driver Sheet will receive an error code. Use this error code and the following table to identify what kind of failure occurred. Error codes above 100 are associated with device errors.

<table>
<tr><th>Error Code</th><th>Description</th><th>Possible Causes</th><th>Procedure to Solve</th></tr>
<tr><td>0</td><td>OK</td><td>N/A</td><td>N/A</td></tr>
<tr><td>1</td><td>Read-only memory</td><td>Attempt to write to read-only fields of headers AP or DP</td><td>- Ensure the application does not tries to write to the device tags associated with read-only fields</td></tr>
<tr><td>2</td><td>Invalid address range</td><td>Attempt to read a worksheet with a range of addresses for headers SPI or SPF that is not supported</td><td>- Check for the valid range of addresses that cannot coexist on the same worksheet. Refer to the table below for fixing it. For more information check the Standard Driver Worksheet section of this document.
<table><tr><th>Header Field</th><th>Address Field</th></tr><tr><td>SPI</td><td>0000~1023</td></tr><tr><td>SPI</td><td>1024~10239</td></tr><tr><td>SPF</td><td>0000~1023</td></tr><tr><td>SPF</td><td>1024~10239</td></tr></table></td></tr>
<tr><td>3</td><td>Unable to read from PLC</td><td>PLC is not responsive for read commands.</td><td>- Check the configurations of the device for error codes.
- Check physical connections.
- Please contact support to report and solve the issue</td></tr>
<tr><td>101</td><td>Undefined Command</td><td>Unrecognized command sent to device</td><td>- Check the error code E101 for the device
- Please contact support to report and solve the issue</td></tr>
<tr><td>102</td><td>Invalid point type</td><td>Attempt to read a digital point as analog or the contrary.</td><td>- Check device configurations for modules
- Check the error code E102 for the device
- Please contact support to report and solve the issue</td></tr>
<tr><td>103</td><td>Invalid float</td><td>Unrecognized float representation</td><td>- Check the error code E103 for the device
- Please contact support to report and solve the issue</td></tr>
<tr><td>104</td><td>Powerup clear expected</td><td>Attempt to read/write from memory without sending a powerup clear command</td><td>- Check that the device is responsive with manufacturer memory inspection tool
- Check the error code E104 for the device
- Please contact support to report and solve the issue</td></tr>
<tr><td>105</td><td>Invalid memory address or invalid data for memory address</td><td>Attempt to read/write from unrecognized memory address.</td><td>- Check the addresses and headers for worksheets.
- Check the error code E105 for the device
- Please contact support to report and solve the issue</td></tr>
<tr><td>106</td><td>Invalid command length</td><td>A command too large was sent.</td><td>- Check the error code E106 for the device
- Please contact support to report and solve the issue</td></tr>
<tr><td>108</td><td>Busy</td><td>The PLC is busy.</td><td>- Check that the device is responsive
- Check the error code E108 for the device</td></tr>
<tr><td>114</td><td>Feature is not yet implemented</td><td>Attempt to access reserved memory areas, or unavailable at the moment.</td><td>- Check the error code E114 for the device</td></tr>
<tr><td>115</td><td>Communications watchdog timeout</td><td>Timeout on the communications.</td><td>- Check the configurations for the communications watchdog
- Check physical connections</td></tr>
</table>

			- Check the error code E115 for the device - Please contact support to report and solve the issue
-15	Timeout waiting start a message	PLC is not responsive.	- Check cable connections - Check Station field for wrong IP or port addresses
-38	Invalid station	Invalid configuration in the Station field, such as leaving it blank or configuring an invalid IP Address	- Check the Station field on your driver sheet. Make sure that it is not blank and the IP address is complete and valid.
-39	Invalid block size	Standard Driver Sheet with an offset between the lowest and the highest address higher than the allowed amount for that device register type	- Check your driver sheet to make sure you respect the block size limitations, as described in the Standard Driver Worksheet section of this document.
-37	Invalid header	This error happens when you configure a {Tag} between curly brackets on the Header field and the value of this {Tag} does not comply with the Header syntax	- Change the Tag value in order to comply with the Header Syntax.
-42	Invalid bit number	This error happens when you configure the SPS register and does not use the S data type, whilst using a second parameter to inform the length of the string	- Check your driver worksheet Address configuration, to make sure that for headers SPS the data type S is used for every address.
-60	Connection Error	Error to establish a TCP/IP connection with the Slave device. Possibly wrong IP Address or Port Number in the Station field	- Check the IP Address, port and ID number in the Station field - Try to *ping* the IP address that you configured in the Station field

⇨ **Tip:**

You can monitor communication status by establishing an event log in Studio's *Output* window (*LogWin* module). To establish a log for **Field Read Commands**, **Field Write Commands** and **Protocol Analyzer,** right-click in the *Output* window and select the desired options from the pop-up menu.

You can also use the *Remote LogWin* module (**Tools** → **LogWin**) to establish an event log on a remote unit that runs Windows CE

If you are unable to establish communication between Studio and the target device, then try instead to establish communication using the device's own programming software (PAC Control or PAC Manager with Inspection Tool). 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.

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

- **Operating System and Project Information** (type and version): To find this information, select **Help** → **Support Information**.
- **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 OPMMP 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\OPMMP`**.
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	André Körbes	Aug. 18, 2010	Initial version
B	1.1	Lourenço Teodoro	Jul. 29, 2015	Fixed issue to reestablish TCP/IP connection