CYLON Communication Driver

Driver for Ethernet communication with Cylon devices (UCU and UC32.xx)

Contents

Introduction

The CYLON driver enables communication between the Studio system and some Cylon devices (UCU and UC32) through UC32.net device over Ethernet, according to the specifications discussed in this document.

This document will help you to select, configure and execute the CYLON 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 CYLON driver in the Studio system.
- **Configuring the Device**: Describes how the target device must be configured to receive communication from the CYLON driver.
- **Configuring the Driver**: Explains how to configure the CYLON driver in the Studio system, including how to associate database tags with device registers.
- **Executing the Driver**: Explains how to execute the CYLON 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 CYLON 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 Ethernet communication between the CYLON driver in Studio and a target Cylon devices.

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**: Cylon Controls Ltd
- **Compatible Equipment**:
  - UCU Series
  - UC32.xx Series

Network Specifications

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

- **Device Communication Port**: Ethernet Port on UC32.net
- **Physical Protocol**: Ethernet/TCP-IP
- **Specific PC Board**: Any TCP/IP Adapter (Ethernet board)
- **Device Runtime Software**: None

Driver Characteristics

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

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

Note:
You must use Adobe Acrobat® Reader™ to view the **`CYLON.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 CYLON driver on the following operating systems:

- Windows 2000/XP/Vista/7/8/Windows CE

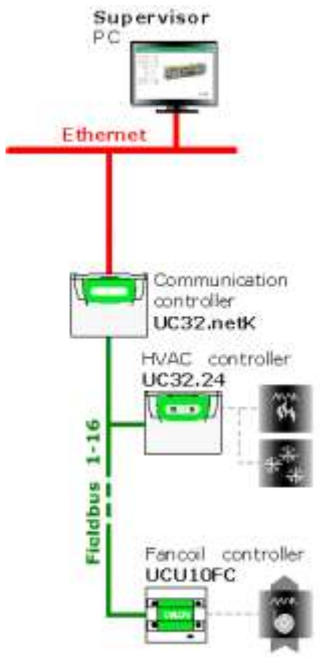
The CYLON driver supports the following registers:

Registers Type	Length	Write	Read
D (*Digital*)	1 bit	•	•
HD *(Hardware Digital)*	1 bit	•	•
A (*Analog*)	4 bytes (*Float*)	•	•
HA *(Hardware Analog)*	4 bytes (*Float*)	•	•

Conformance Testing

The following hardware/software was used for conformance testing:

- **Driver Configuration**:
  - **Cable**: Ethernet Cable
  - **Station field**: 10.168.23.97:1:UCU:1 and 10.168.23.97:1:UC32:2
  -

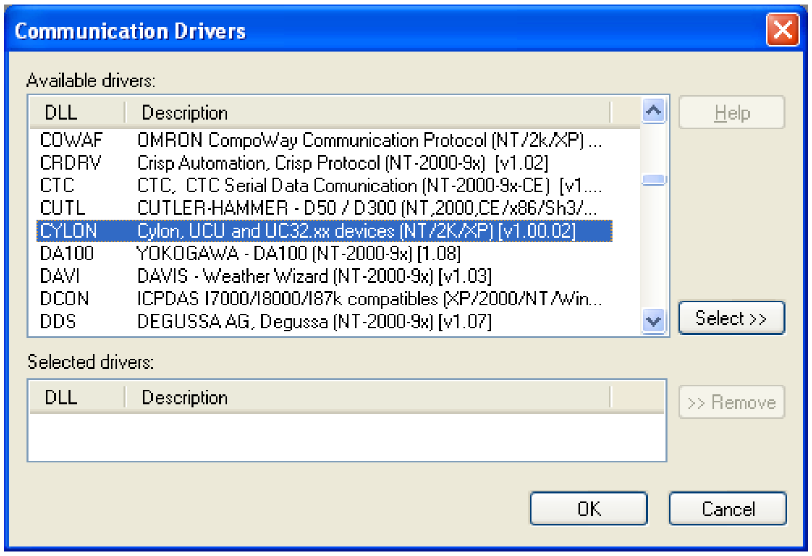


Driver Version	Studio Version	Operating System (development)	Operating System (runtime)	Equipment
2.00	V7.1 + SP3	WinXP + SP2/7/8	WinXP + SP2/7/8	- UC32.netK/Web - UC32.24 - UCU10FC

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 CYLON driver for your Studio application:

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



Available drivers:

DLL	Description
COWAF	OMRON CompoWay Communication Protocol (NT/2k/XP) ...
CRDRV	Crisp Automation, Crisp Protocol (NT-2000-9x) [v1.02]
CTC	CTC, CTC Serial Data Comunication (NT-2000-9x-CE) [v1....
CUTL	CUTLER-HAMMER - D50 / D300 (NT,2000,CE/x86/Sh3/...
CYLON	Cylon, UCU and UC32.xx devices (NT/2K/XP) [v1.00.02]
DA100	YOKOGAWA - DA100 (NT-2000-9x) [1.08]
DAVI	DAVIS - Weather Wizard (NT-2000-9x) [v1.03]
DCON	ICPDAS I7000/I8000/I87k compatibles (XP/2000/NT/Win...
DDS	DEGUSSA AG, Degussa (NT-2000-9x) [v1.07]

Selected drivers:

DLL	Description

Communication Drivers Dialog

3. When the **CYLON** 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 not necessary to install any other software on your computer to enable communication between Studio and your target device. However, this communication can only be used by the Studio application; it cannot be used to download control logic to the device. To download control logic to a Cylon device, you must also use the Cylon programming software. For more information, please consult the documentation provided by the device manufacturer.

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

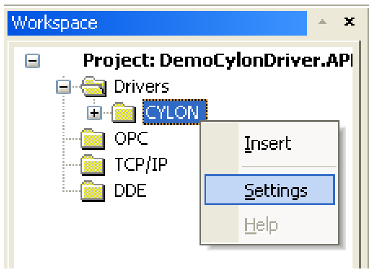
Once you have selected the CYLON driver in Studio, you must properly configure it to communicate with your target device. First, you must set the driver's communication settings to match the parameters set on the device. Then, you must build driver worksheets to associate database tags in your Studio 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.

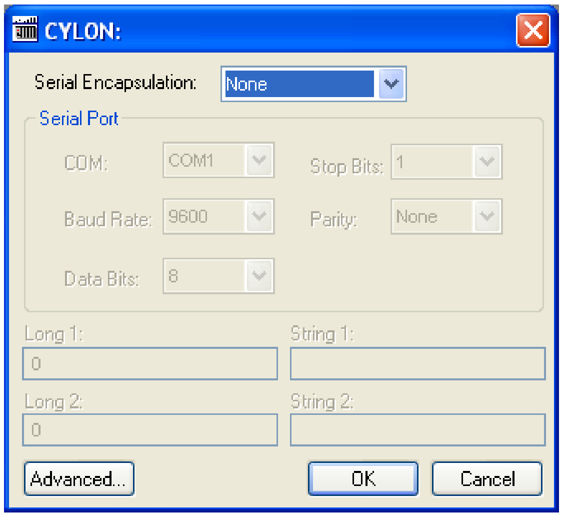
For the purposes of this document, only CYLON driver-specific settings and procedures will be discussed here. To configure the communication settings for the CYLON driver:

1. In the *Workspace* pane, select the *Comm* tab and then expand the *Drivers* folder. The CYLON driver is listed here as a subfolder.
2. Right-click on the *CYLON* subfolder and then select the **Settings** option from the pop-up menu:



Select Settings from the Pop-Up Menu

The *CYLON: Communication Settings* dialog is displayed:



Serial Encapsulation: None
Serial Port
COM: COM1
Stop Bits: 1
Baud Rate: 9600
Parity: None
Data Bits: 8
Long 1: 0
String 1:
Long 2: 0
String 2:

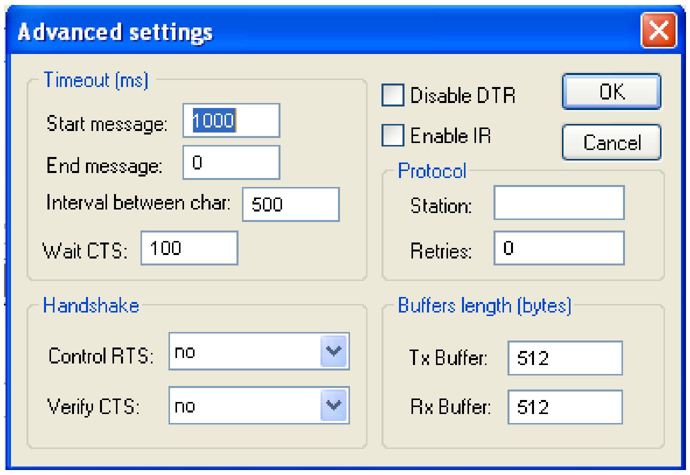
CYLON: Communication Settings Dialog

3. In the *Communication Settings* dialog, configure the driver settings to enable communication with your target device. To ensure error-free communication, the driver settings must *exactly match* the corresponding settings on the device. Please consult the manufacturer's documentation for instructions how to configure the device and for complete descriptions of the settings.

   Depending on your circumstances, you may need to configure the driver *before* you have configured your target device. If this is the case, then take note of the driver settings and have them ready when you later configure the device.

   ➲ **Attention:**
   For safety reasons, you **must** take special precautions when connecting and configuring new equipment. Please consult the manufacturer's documentation for specific instructions.

4. If you are using a Data Communication Equipment (DCE) converter (e.g., 232/485) between your PC and your target device, then you must also adjust the **Control RTS** (Request to Send) setting to account for the converter. In the *Communication Settings* dialog, click the **Advanced** button to open the *Advanced Settings* dialog:



Start message: 1000
End message: 0
Interval between char: 500
Wait CTS: 100
Disable DTR: unchecked
Enable IR: unchecked
Station:
Retries: 0
Control RTS: no
Verify CTS: no
Tx Buffer: 512
Rx Buffer: 512

Advanced Settings Dialog

When the dialog is displayed, configure the **Control RTS** setting using the following information:

Setting	Default	Values	Description
Control RTS	no	no	Do not set the RTS (Request to Send) handshake signal. IMPORTANT: If you are using Windows 95/98 or Windows CE with the correct RS232/RS485 adapter (i.e. without RTS control), then you must select this option.
		yes	Set the RTS (Request to Send) handshake signal before communication. IMPORTANT: If you are using Windows NT and the Cutler-Hammer RS232/RS485 adapter, then you must select this option.
		yes+echo	Set the RTS (Request to Send) handshake signal before communication, and echo the signal received from the target device.

➲ **Attention**:

If you incorrectly configure the **Control RTS** setting, then runtime communication will fail and the driver will generate a 1005 error. See “Troubleshooting” for more information.

You do not need to change any other advanced settings at this time. You can consult the Studio *Technical Reference Manual* later for more information about configuring these settings.

5. Click **OK** to close the *Advanced Settings* dialog, and then click **OK** to close the *Communication Settings* dialog.

Configuring the Driver Worksheets

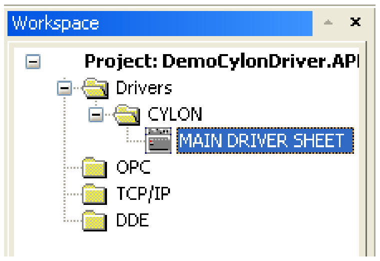
A selected driver includes one or more driver worksheets, which are used to associate database tags in Studio with registers on the target device. Each worksheet is triggered by specific application behavior, so that the tags / registers defined on that worksheet are scanned only when necessary – that is, only when the application is doing something that requires reading from or writing to those specific tags / registers. Doing this optimizes communication and improves system performance.

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.

MAIN DRIVER SHEET

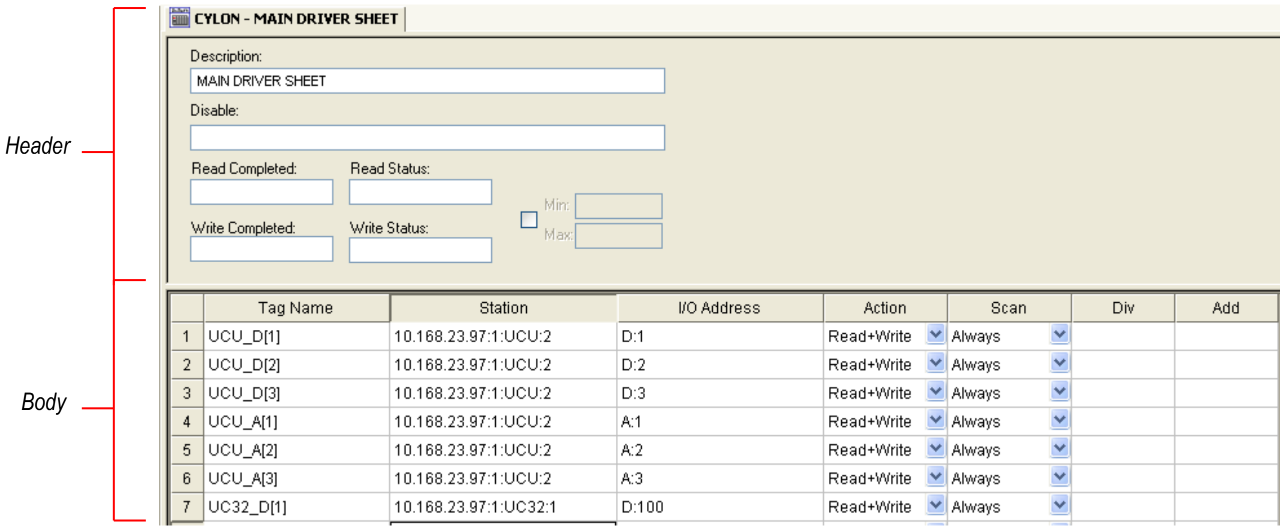
When you select the CYLON driver and add it to your application, Studio automatically inserts the *Main Driver Sheet* in the *CYLON* driver subfolder. To configure the Main Driver Sheet:

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



Main Driver Sheet in the CYLON Subfolder

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



Description: MAIN DRIVER SHEET
Disable:
Read Completed:
Read Status:
Write Completed:
Write Status:

	Tag Name	Station	I/O Address	Action	Scan	Div	Add
1	UCU_D[1]	10.168.23.97:1:UCU:2	D:1	Read+Write	Always		
2	UCU_D[2]	10.168.23.97:1:UCU:2	D:2	Read+Write	Always		
3	UCU_D[3]	10.168.23.97:1:UCU:2	D:3	Read+Write	Always		
4	UCU_A[1]	10.168.23.97:1:UCU:2	A:1	Read+Write	Always		
5	UCU_A[2]	10.168.23.97:1:UCU:2	A:2	Read+Write	Always		
6	UCU_A[3]	10.168.23.97:1:UCU:2	A:3	Read+Write	Always		
7	UC32_D[1]	10.168.23.97:1:UC32:1	D:100	Read+Write	Always		

Opening the 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 CYLON driver.

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

   - **Station** field: Identify the target device, using the following syntax:

     <IP Address>*:*<UC32.Net Address>:<Controller Type>:<Controller Address>

     Example — **`10.168.23.73:1:UC32:2`**

     Where:

     <IP Address> is the UC32.net IP address on the Ethernet network.

     <UC32.net Address> is the UC32.net Address.

     <Controller Type> is the Controller Type on the Fieldbus network. The options for this parameter are: UCU or UC32.

     <Controller Address>**is the Controller Address on the FieldBus network.**

     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 must use a non-zero value in the **Station** field, and you cannot leave the field blank.

   - **I/O Address:** Specify the address of the associated device register.

     <Register Type>*:*<Address>

     Examples — **`A:1, A:10, D:2, D:5`**

Where:

- ***<Register Type>*** is the register type, which must be D (*Digitals*), HD *(Hardware Digitals)*, A *(Analog)* and HA *(Hardware Analog).*

– ***<Address>*** is the specific address of the register on the device.

> **Note:**
> - See the supported registers in the Driver Characteristics session.

For examples of how register addresses are composed using the values in the **I/O Address** field, see the following table:

Register Address in the Device	I/O Address
A10	A:10
A100	A:100
A7168	A:7168
D5	D:5
D200	D:200
D7191	D:7191
D7168	HD:1
D7191	HD:24
A7169	HA:2
A7170	HA:3

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

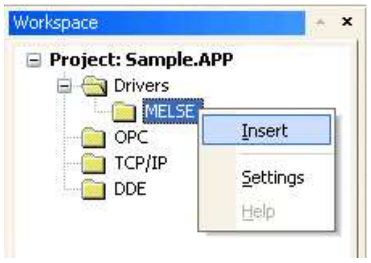
STANDARD DRIVER WORKSHEET

When you select the CYLON 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.

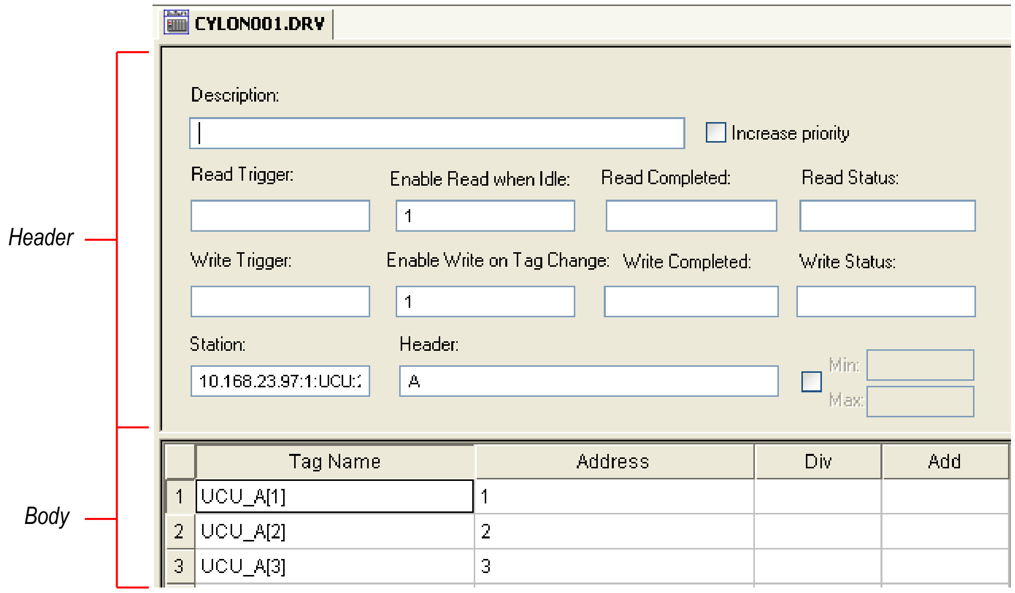
To insert a new driver worksheet:

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



Inserting a New Worksheet

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



Description:	
Read Trigger:	
Enable Read when Idle:	1
Read Completed:	
Read Status:	
Write Trigger:	
Enable Write on Tag Change:	1
Write Completed:	
Write Status:	
Station:	10.168.23.97:1:UCU::
Header:	A
Min:	
Max:	

	Tag Name	Address	Div	Add
1	UCU_A[1]	1		
2	UCU_A[2]	2		
3	UCU_A[3]	3		

CYLON Driver Worksheet

Note:
Worksheets are numbered in order of creation, so the first worksheet is **`CYLON001.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 **Station**, **Header**, and **Address** fields use syntax that is specific to the CYLON driver.

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

   - **Station** field: Identify the target device, using the following syntax:

     `<IP Address>:<UC32.Net Address>:<Controller Type>:<Controller Address>`

     Example — **`10.168.23.73:1:UC32:2`**

     Where:

     `<IP Address>` is the UC32.net IP address on the Ethernet network.

     `<UC32.net Address>` is the UC32.net Address.

     `<Controller Type>` is the Controller Type on the Fieldbus network. The options for this parameter are: UCU or UC32.

     `<Controller Address>` **`is the Controller Address on the FieldBus network.`**

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.

- **Header** field: Specify the address of the first register in 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:

  `<Register Type>`

  Examples — **`A; D; HA; HD`**

  Where:

  – <***`Register Type`***> is the register type, which must be D (*Digitals*), HD *(Hardware Digitals)*, A *(Analog)* and HA *(Hardware Analog).*

  After you edit the **Header** field, Studio checks the syntax to determine if it is valid. If no value is entered, then the default type is **`A`**.

  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.

➲ **Attention:**

- You cannot leave the **Station** and **Header** fields blank; you must specify some values.

Note:

- See the supported registers in the Driver Characteristics session.

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

   <Address>

   Examples — `5, 10`

   Where:

   – ***<Address>*** is the specific address of the register on the device.

   For examples of how register addresses are composed using the values in the **Header** and **Address** fields, see the following table:

Register Address in the Device	Header Field	Address Field
A10	**A**	**10**
A100	**A**	**100**
A7168	**A**	**7168**
D5	**D**	**5**
D200	**D**	**200**
D7191	**D**	**7191**
D7168	**HD**	**1**
D7191	**HD**	**24**
A7169	**HA**	**2**
A7170	**HA**	**3**

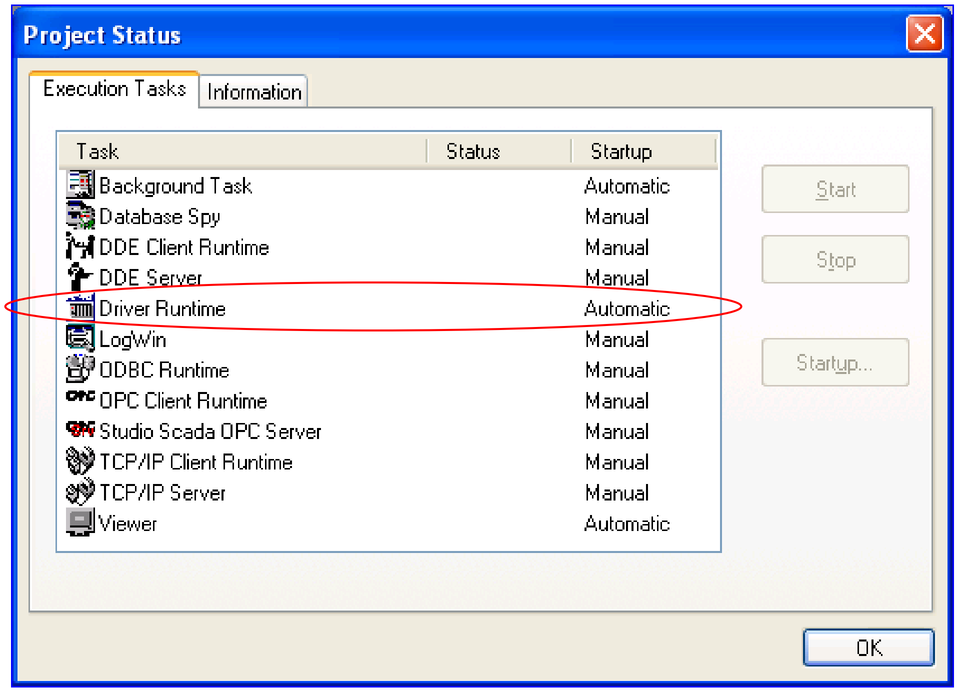
For more information about registers and addressing, please consult the manufacturer’s documentation.

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:



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, 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 CYLON 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 Main Driver Sheet 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
1	ERROR CONNECT	IP address does not match with UC32.net	Check the right IP Address
2	INVALID READ	The device responded with an error code.	Check if the registers and address configured exist into the PLC.
3	INVALID IP ADDRESS	IP Address invalid	Check the IP Address on Station field
4	INVALID CONTROLLER	Controller type is invalid	Check the Controller Type on Station field
5	INVALID UNET ADDRESS	UC32.Net Address is invalid	Check the UC32.Net Address on Station field
6	INVALID CONTROLLER ADDRESS	Controller Address is invalid	Check the Controller Address on Station field
7	INVALID WRITE	The device responded with an error code.	Check if the registers and address configured exist into the PLC.
8	INVALID COMMAND	Trying to send a command to wrong controller	Check the Controller Type on Station field
9	INVALID HEADER	Trying to communicate with a wrong Header.	Check the registers type.
1005	Timeout waiting start a message.	▪ 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 (it must be RUN) ▪ Check the station number. ▪ Check the right configuration. Review the Communication Parameters section for valid RTS/CTS configurations.

⇨ **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 **Serial Communication,** right-click in the *Output* window and select the desired options from the pop-up menu.

You can also use the *LogWin* module (**Tools** → **LogWin**) to establish an event log on a remote unit that runs Windows CE. The log is saved on the unit in the `celog.txt` file, which can be downloaded later.

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. 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

There was not an official sample application available for this driver by the timer that this document was written.

Revision History

Doc. Revision	Driver Version	Author	Date	Description of changes
A	1.00	Eric Vigiani	Mar-20-2008	First driver version.
B	2.00	Paulo Balbino	May-08-2014	Added CE support