ABENI Communication Driver

Driver for Ethernet Communication with AB Devices
Using AB-1761-NET-ENI Gateway Interface

Contents

Introduction

The ABENI driver enables communication between the Studio system and some of the Allen-Bradley devices using the AB-1761-NET-ENI gateway interface, if these components comply with the characteristics described in this document.

This document was designed to help you install, configure and execute the ABENI driver to enable Ethernet communication with devices using the DF1 over EthernetIP protocol. The information in this document is organized as follows:

- **Introduction**: Provides an overview of the ABENI driver documentation.
- **General Characteristics**: Describes all of the required hardware and software components you need to implement Ethernet communication. This chapter also discusses global characteristics about the communication.
- **Installing the Driver**: Explains how to install the hardware and software components required for the ABENI driver.
- **Configuring the Driver**: Explains how to configure the communication driver, including the different permutations for configuration and the driver's default values.
- **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**: Provides a sample application, which you can use to test the 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 product's *Technical Reference Manual*.
- This document also assumes that you are familiar with the Windows NT/2000/XP/Vista environment. If you are unfamiliar with Windows NT/2000/XP/Vista, we suggest using the **Help** feature (available from the Windows desktop **Start** menu) as you work through this guide.

General Characteristics

This chapter explains how to identify all the hardware and software components used to implement communication between Studio's ABENI driver and Allen-Bradley devices using the AB-1761-NET-ENI gateway interface. In addition, this chapter provides information about the equipment used for conformance testing.

The information is organized into the following sections:

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

Device Characteristics

To establish TCP/IP Ethernet communication, you must use devices with the following specifications:

- **Manufacturer:** Allen-Bradley
- **Compatible Equipment:** PLC-5, SLC500, MicroLogix and AB 1761-NET-ENI Series A, B, C and D
- **Rockwell PLC Programmer Software:** RSLogix5, RSLogix500

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

Link Characteristics

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

- **Device Communication Port:** AB 1761-NET-ENI Ethernet port
- **Physical Protocol:** Ethernet/TCP/IP
- **PLC Error Check:** CRC
- **Logic Protocol:** DF1/Ethernet IP
- **Device Runtime Software:** None
- **Specific PC Board:** Any TCP/IP Adapter (Ethernet Board)

⇨ **Tip:**
Please refer to the "" section to see the equipment used in the standard conformance tests for this driver.

Driver Characteristics

The ABENI driver is composed of the following files:

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

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

- Windows NT/2000/XP
- Windows CE

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

The ABENI driver supports the following register types:

Register Type	Length	Write	Read	Bit	Word	Float	String	BCD
O (*Output*)	2 Bytes	●	●	●	●	–	●	●
I (*Input*)	2 Bytes	●	●	●	●	–	●	●
S (*Status*)	2 Bytes	–	●	●	●	–	●	●
B (*Binary*)	2 Bytes	●	●	●	●	–	●	●
T (*Timer*)	6 Bytes	●	●	–	●	–	–	–
C (*Counter*)	6 Bytes	●	●	–	●	–	–	–
R (*Control*)	6 Bytes	●	●	–	●	–	–	–
F *(Float)*	4 Bytes	●	●	–	–	●	–	–
N *(Integer File)*	2 Bytes	●	●	●	●	–	●	●
ST *(String File)*	N Bytes	●	●	–	–	–	●	–

Conformance Testing

The following hardware/software was used for conformance testing:

- **Configuration:**
  - IP: 192.168.23.190
  - PLCs: MicroLogix 1500 on Channels 0 and 1, with Error Check for CRC, SLC5/04 and PLC5
  - Protocol: DF1 – Ethernet/IP
  - AB 1761-NET-ENI Series A v1.02, B v2.31 and D3.21

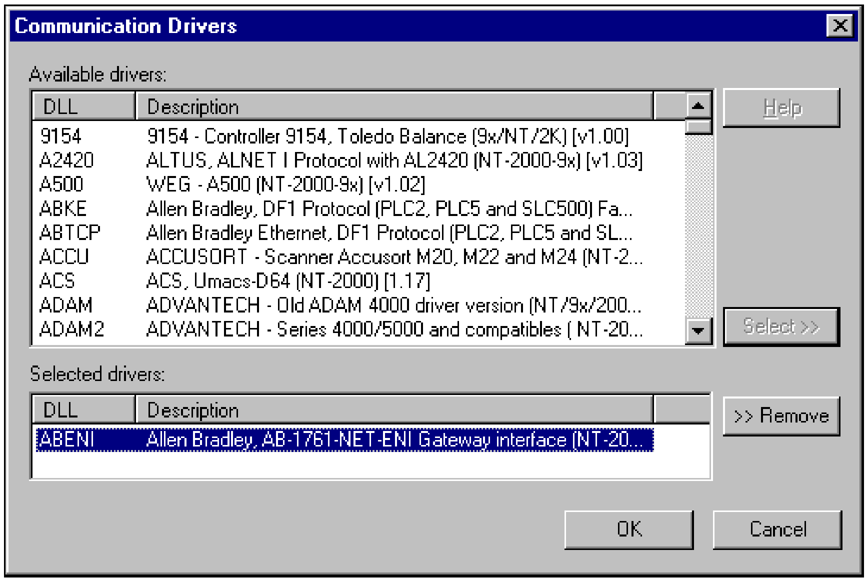
Driver Version	Studio Version	Operating System – Development	Operating System – Target	Equipment
1.09	6.1 + SP5	Windows XP + SP3	Windows XP + SP3	PLC5, SLC5/04 MicroLogix1500
1.09	6.1 + SP5	Windows XP + SP3	WinCE 4.2 ArmV4 WinCE 5.0 x86	PLC5, SLC5/04 MicroLogix1500

Installing the Driver

When you install the Studio v5.1 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, or double-click the **Studio** shortcut icon from the desktop.
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 **ABENI** driver from the *Available Drivers* list, and then click the **Select** button:



Available drivers:

DLL	Description
9154	9154 - Controller 9154, Toledo Balance (9x/NT/2K) [v1.00]
A2420	ALTUS, ALNET I Protocol with AL2420 (NT-2000-9x) [v1.03]
A500	WEG - A500 (NT-2000-9x) [v1.02]
ABKE	Allen Bradley, DF1 Protocol (PLC2, PLC5 and SLC500) Fa...
ABTCP	Allen Bradley Ethernet, DF1 Protocol (PLC2, PLC5 and SL...
ACCU	ACCUSORT - Scanner Accusort M20, M22 and M24 (NT-2...
ACS	ACS, Umacs-D64 (NT-2000) [1.17]
ADAM	ADVANTECH - Old ADAM 4000 driver version (NT/9x/200...
ADAM2	ADVANTECH - Series 4000/5000 and compatibles (NT-20...

Selected drivers:

DLL	Description
ABENI	Allen Bradley, AB-1761-NET-ENI Gateway interface (NT-20...

Communication Drivers Dialog

5. When the **ABENI** 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 Studio and the Host. However, to download the custom program to the device, you must install one of the Rockwell programmer software packages (such as RSLogix). Consult the Rockwell 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 **ABENI** driver, you must configure the driver. Configuring the driver is done in two parts:

- Specifying settings or communication parameters (there is only one configuration for the whole driver).
- Defining communication tags and controls in the *Communication* tables or *Driver* worksheets. There are two types of communication tables: STANDARD TABLES and the MAIN DRIVER SHEET (MDS).

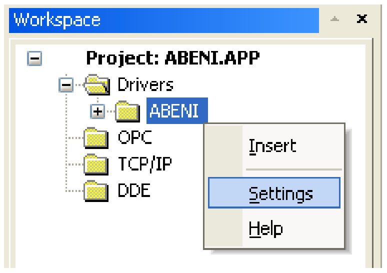
  Worksheets are divided into two sections, a *Header* and a *Body*. The fields contained in these two sections are standard for all communications drivers — except the **Station**, **Header** and **Address** fields, which are driver-specific. This document explains how to configure the **Station**, **Header**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

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

Setting the Communication Parameters

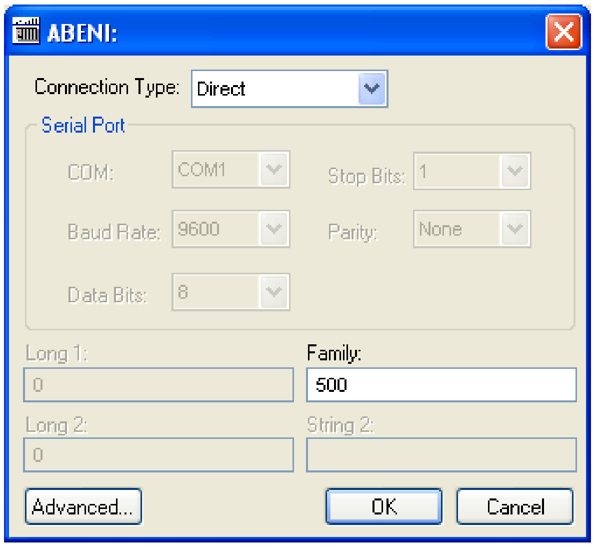
When you set the communication parameters, they are valid for all *Driver* worksheets configured in the system. Use the following steps to configure the communication parameters for the driver:

1. From the *Studio* application screen, click the **Comm** tab located below the *Workspace* pane.
2. From the *Workspace* pane, expand the *Drivers* folder.
3. Right-click on the *ABENI* subfolder. When the pop-up menu displays, select the **Settings** option:



Select Settings from the Pop-Up Menu

The *Communication Parameters* dialog displays:



Connection Type: Direct

Serial Port

COM: COM1

Stop Bits: 1

Baud Rate: 9600

Parity: None

Data Bits: 8

Long 1: 0

Family: 500

Long 2: 0

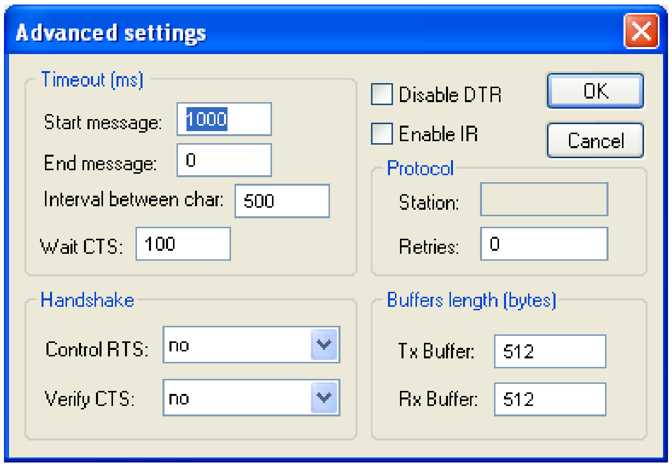
String 2:

Communication Parameters Dialog

4. You must configure the following parameters:

Parameter	Default Value	Valid Values	Description
Family	SLC500	2 5 or 5:0 5:1 500	AB PLC family (communication with the Studio's driver) **2** - Driver ABENI uses "unprotected read/write" DF1 command to communicate with PLC2 family. **5 or 5:0** - Driver ABENI uses "typed read/write" DF1 command to communicate with PLC5 family. The address for any data type is decimal. **5:1** - Driver ABENI uses "typed read/write" DF1 command to communicate with PLC5 family. The address for I and O data types is octal. The address for the remaining data types is decimal. **500** - Driver ABENI uses "protected typed logical read/write" DF1 command to communicate with SLC500 family.

5. Click the **Advanced** button on the *Communication Parameters* dialog to open the *Advanced Settings* dialog:



Timeout (ms)
Start message: 1000
End message: 0
Interval between char: 500
Wait CTS: 100

☐ Disable DTR
☐ Enable IR

Protocol
Station:
Retries: 0

Handshake
Control RTS: no
Verify CTS: no

Buffers length (bytes)
Tx Buffer: 512
Rx Buffer: 512

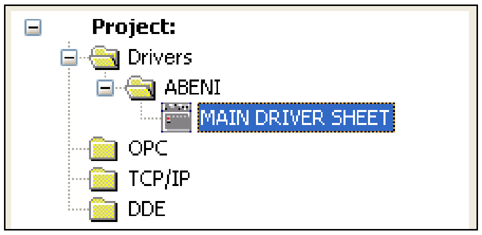
Advanced Settings Dialog

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 MAIN DRIVER SHEET

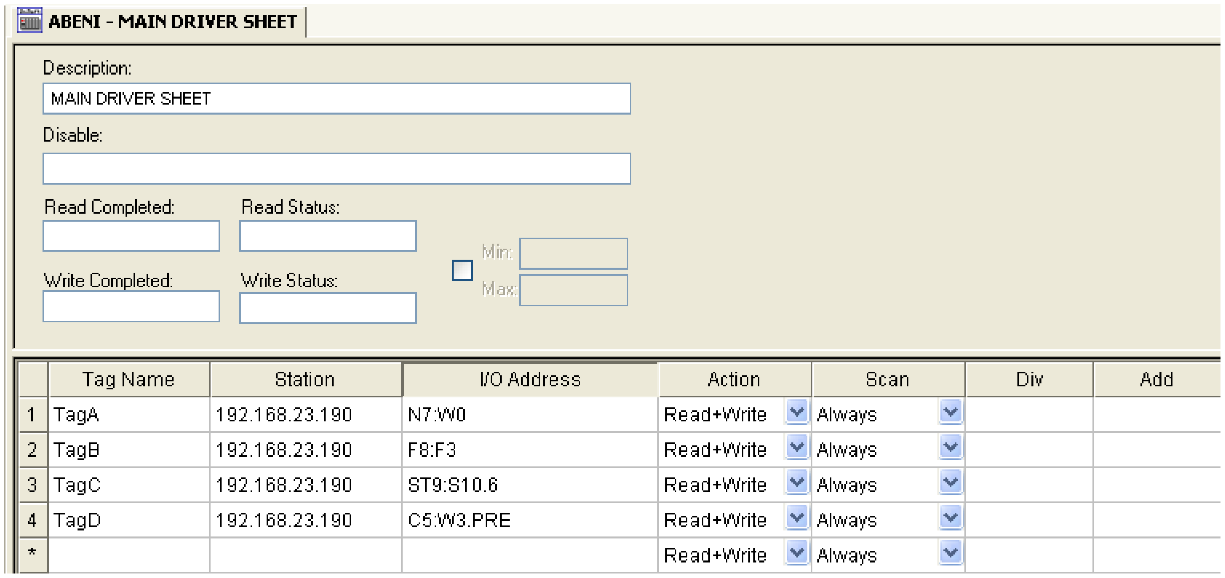
When you add the ABENI driver to your application, the program automatically adds the MAIN DRIVER SHEET (*MDS*) to the **ABENI** driver folder (refer to the following figure).



Select the Main Driver Sheet

The MDS provides a simple way for you to associate Studio tags to addresses in the PLC. Most MDS entries are standard for any driver. For detailed information about configuring these standard entries, refer to the *Studio Technical Reference Manual*.

1. Double-click on the **MAIN DRIVER SHEET** icon to open the following Worksheet:



ABENI - MAIN DRIVER SHEET

Description: MAIN DRIVER SHEET

Disable:

Read Completed: Read Status:

Write Completed: Write Status:

Min: Max:

	Tag Name	Station	I/O Address	Action	Scan	Div	Add
1	TagA	192.168.23.190	N7:W0	Read+Write	Always		
2	TagB	192.168.23.190	F8:F3	Read+Write	Always		
3	TagC	192.168.23.190	ST9:S10.6	Read+Write	Always		
4	TagD	192.168.23.190	C5:W3.PRE	Read+Write	Always		
*				Read+Write	Always		

Main Driver Sheet

2. Complete the following fields on this worksheet, being sure to comply with the following syntax:

   - **Station:** PLC Address (IP address) e.g.: `192.168.2.99`
   - **I/O Address:** Address of each register from the PLC. You must use the following syntax:
     – For **Input** and **Outputs**: ***`<Type>:<SlotNumber>.<Format><OctetNumber>/<Bit>`***
     (for example: `O:1.W2/4`)

- For **Status**: ***<Type>*:*<Format><Address>/<Bit>***
  (for example: S:`W1/2`)
- For **Binary** and **Integer**: ***<Type><TypeGroup>*:*<Format><Address>/<Bit>***
  (for example: `N7:W150/2`)
- For **Float**: ***<Type><TypeGroup>:<Format><Address>*** (for example: F8:`F40`)
- For **Timer**, **Counter** and **Control**: ***<Type><TypeGroup>:<Format><Address>.<Element>***
  (for example: `T4:W0.DN`)
- For **String**: ***<Type><TypeGroup>:<Format><Address>.<Number of Bytes>*** (for example: `ST15:S0.50`)

  Where**:**
  - **Type:** Register Type: **O** (Output), **I** (Input), **S** (Status), **B** (Binary), **N** (Integer), **T** (Timer), **C** (Counter), **R** (Control), **F** (Float), **ST** (String).
  - **SlotNumber**: The I/O card slot number.
  - **TypeGroup**: The Group number of the register type configured.
  - **Format** Type

    W to treat the values as words
    B to treat the values as BCDs
    F to treat the values as Floats (double words)
    S to treat the values as Strings

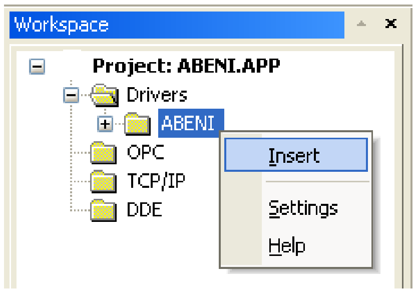
  - **OctetNumber**: The Octet number of the I/O card configured.
  - **Address**: The Address of the Group configured.
  - **Number of Bytes**: The maximum size of STRING data type.
  - **Bit**: The bit numbers (from 0 to 15) from the word address (optional parameter).
  - **Element**: The element type for Timers, Counters and Controls (refer to the table in the "Configuring the Address Field" section).

Configuring the *STANDARD DRIVER WORKSHEET*

This section explains how to configure a *Standard Driver Worksheet* (or Communication table) to define communication tags. You can configure multiple *Driver* worksheets, each of which is divided into a *Header* and *Body*.

Use the following steps to create a new *Standard Driver Worksheet*:

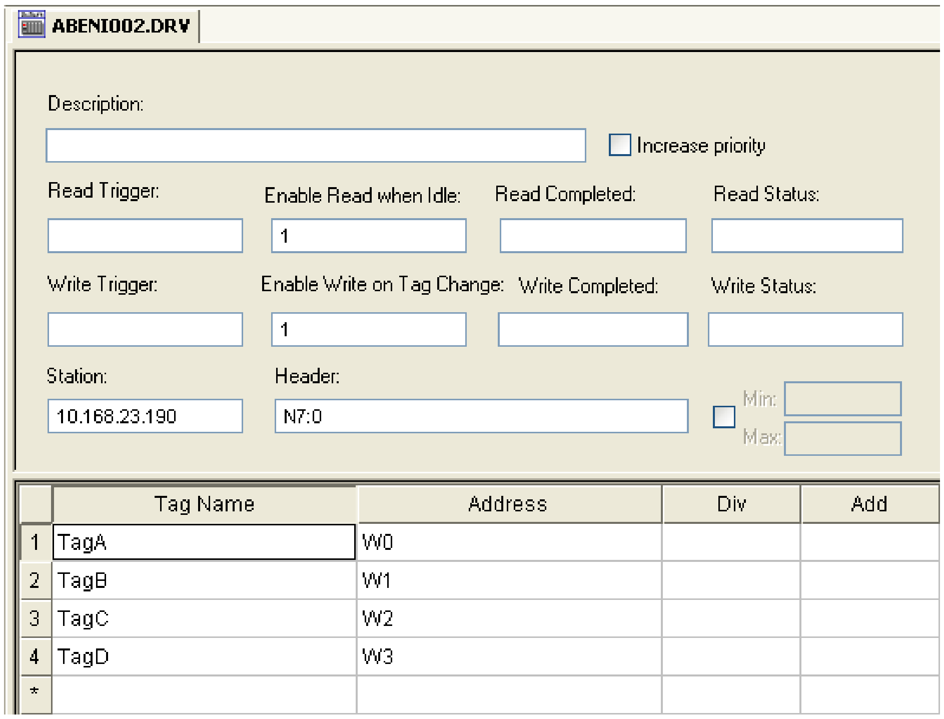
1. From the *Studio* application screen, select the **Comm** tab, located below the *Workspace* pane.
2. In the *Workspace* pane, expand the *Drivers* folder and right-click the *ABENI* subfolder.
3. When the pop-up menu displays, select the **Insert** option:



Inserting a New Worksheet

> ⇨ **Tip:**
> To optimize communication and ensure better performance for the system, it is important to tie the tags in different driver sheets together according to the events that trigger communication between each group of tags and the period for which each group of tags must be written or read. In addition, we recommend configuring the communication addresses into sequential blocks.

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



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:	10.168.23.190
Header:	N7:0

	Tag Name	Address	Div	Add
1	TagA	W0		
2	TagB	W1		
3	TagC	W2		
4	TagD	W3		
*				

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: This field complies with the following syntax:
     <IP address> `e.g.: 192.168.2.99`
     Where:
     – **IP Address**: PLC IP Address in the TCP/IP network.

   - The **Header** parameter must comply with the following syntax:
     – For **Input** and **Output**: ***<Type>:<SlotNumber>.<AddressReference>*** (for example: `O:1.0`).
     – For **Status**:
     <Type>:<AddressReference>
     (for example: `S:0`).
     – For **Binary**, **Integer**, **Float, String, Timer**, **Counter** and **Control**:
     <Type><TypeGroup>:<AddressReference>
     (for example: `N7:0`).

Where:

– **Type:** Register Type: **O**=Output, **I**=Input, **S**=Status, **B**=Binary, **N**=Integer; **T**=Timer, **C**=Counter, **R**=Control, **F**=Float, **ST**=String
– **SlotNumber**: The I/O Card Slot Number.
– **TypeGroup:** The Group Number of the Register Type you configured.
– **AddressReference:** The Initial Address (Reference) of the group you configured.

After you edit the **Header** field, the system checks the syntax. If the syntax is invalid, the system automatically inserts the default value (**N7:0**) into the **Header** field.

If you type a Tag string between curly brackets **{Tag}** into this field, you must ensure that the Tag value and syntax are both correct, or an **Invalid Header** error will result.

The following table describes the proper syntax for both the field type and the Tag value:

Information about the Header Parameter			
Type	**Sample of Syntax**	**Valid Range of Initial Address**	**Comments**
Output	O:0.0	Varies according to the equipment	Physical outputs: Where "O" means output. The first digit after the colon defines the slot number. If there is more than one word in the same slot, the first digit following the dot is the word number.
Input	I:0.0	Varies according to the equipment	Physical inputs: Where "I" means input. The first digit after the colon defines the slot number. If there is more than one word in the same slot, the first digit following the dot is the word number.
Status	S:0	Varies according to the equipment	Reads the status words
Binary	B3:0	0 to 255	Reads and Writes the Binary Operator
Integer	N7:0	0 to 255	Reads and Writes the Integer addresses
Timer	T4:0	0 to 255	Reads and Writes the Timer addresses
Counter	C5:0	0 to 255	Reads and Writes the Counter addresses
Control	R6:0	0 to 255	Reads and Writes the Control addresses
Float	F8:0	0 to 255	Reads and Writes the Float addresses
String	ST9:0	0 to 255	Reads and Writes the String addresses

- **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:

  For **Input** and **Outputs**: **<Format>*<OctetNumber>*/*<Bit>*** (for example: **W0/3**)

  For **Status**, **Binary** and **Integer**: **<Format>*<AddressOffset>*/*<Bit>*** (for example: **W10/12**)

  For **Float**: ***<Format><Address>*** (for example: F40)
  For **Timer**, **Counter** and **Control**: **<Format>*<AddressOffset>*.*<Element>*** (for example: **W2.PRE**)

  For **String**: **<Format>*<AddressOffset>*.*<Number of Bytes>*** (for example: **S0.50**)

  Where:

  - **Format** Type (optional parameter)
    W to treat the values as words
    B to treat the values as BCDs
    F to treat the values as Floats (double words)

S to treat the values as Strings

– `OctetNumber`: The Octet Number of the I/O card you configured in the **Header** field.
– `AddressOffset`: Add this parameter to the `AddressReference` (configured in the **Header** field) to compose the address of the Group you configured in the **Header** field.
– `Bit`: The Bit Number (from 0 to 15) from the word address (optional parameter).
– **Number of Bytes**: The maximum size of STRING data type
– `Element`: The Element Type for **Timers**, **Counters** and **Controls** according to the following table:

Register	Elements																	
	DN	PRE	ACC	EN	TT	UA	UN	OV	CD	CU	FD	IN	UL	ER	EM	EU	LEN	POS
Timer	•	•	•	•	•	–	–	–	–	–	–	–	–	–	–	–	–	–
Counter	•	•	•	–	–	•	•	•	•	•	–	–	–	–	–	–	–	–
Control	•	–	–	•	–	–	–	–	–	–	•	•	•	•	•	•	•	•

➲ **Attentions:**

- You can use the `Bit Writing` function only when the **Write on tag change** driver tag is enabled. This also means that you cannot use the **Write trigger** tag for the `Bit Writing` function. The same rule applies to Timers, Counters and Controls registers.
- The **Format B** option (BCD) applies to the first 12 bits only. You can view the last 4 bits in the **Quality** field. For example, when the **TAG_N7[0]** tag is read from the **N7:0** address, the last 4 bits of this address are written to the **TAG_N7[0]->Quality** field.

Sample of Addressing Configuration		
Address on the Device	Header Field	Address Field
I:0/7	I:0.0	0/7
I:0/10	I:0.0	0/10
I:0/17	I:0.0	0/17
I:3/4	I:0.0	3/4
I:3/4	I:0.3	0/4
O:0/7	O:0.0	0/7
O:0/10	O:0.0	0/10
O:0/17	O:0.0	0/17
O:3/4	O:0.0	3/4
O:3/4	O:0.3	0/4
S:0/5	S:0	0/5
S:10/7	S:0	10/7
S:10/7	S:10	0/7
B3:0/5	B3:0	0/5
B3:10/7	B3:0	10/7
B3:10/7	B3:10	0/7

Sample of Addressing Configuration		
Address on the Device	**Header Field**	**Address Field**
N7:0	N7:0	0
N7:0/10	N7:0	0/10
N7:50	N7:20	30
T4:0/ACC	T4:0	0/ACC
T4:0/PRE	T4:0	0/PRE
T15:0/EN	T15:0	0/EN
T15:0/ACC	T15:0	0/ACC
T15:1/ACC	T15:0	1/ACC
C5:0/ACC	C5:0	0/ACC
C5:1/PRE	C5:0	1/PRE
C20:15/UA	C20:10	5/UA
R6:0/LEN	R6:0	0/LEN
R6:0/POS	R6:0	0/POS
R6:1/POS	R6:0	1/POS
F8:0	F8:0	F0
F8:5	F8:5	F0
F8:5	F8:0	F5
ST9:10	ST9:0	S10.12
ST9:3	ST9:3	S0.10

➲ **Attention:**

You are not permitted to configure a range of addresses greater than the maximum block size (data buffer length) supported by each PLC (as follows) in the same worksheet:

- For the SLC device family in the Read commands:
  Maximum data buffer length is 236 bytes (SLC/03 and SLC/04) and 82 bytes (SLC5/01 and SLC5/02).
- For the SLC device family in the Write commands:
  Maximum data buffer length is 234 bytes (SLC/03 and SLC/04) and 82 bytes (SLC5/01 and SLC5/02)
- For the PLC5 device family in the Read commands:
  Maximum data buffer length is 234 bytes

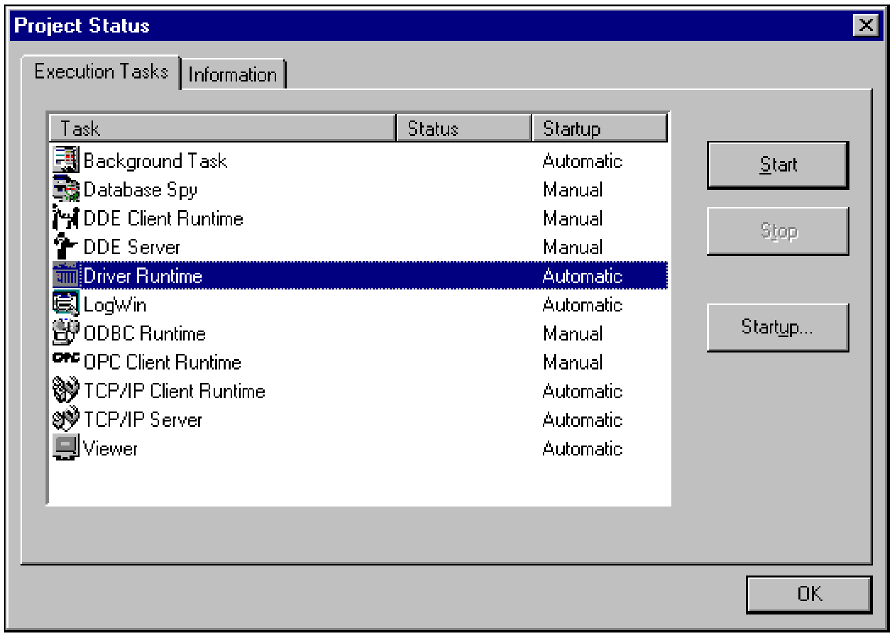
Executing the Driver

When you add the driver to a project, the system sets it automatically. This means the driver is ready to execute when you start-up the Runtime Environment.

To verify that the driver is enabled and will start correctly:

1. Select **Project** from the main menu bar, and then select the **Status...** option from the menu to verify the **Driver Runtime** task.

   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		Automatic
ODBC Runtime		Manual
OPC Client Runtime		Manual
TCP/IP Client Runtime		Automatic
TCP/IP Server		Automatic
Viewer		Automatic

Project Status Dialog

2. Click the **Driver runtime** line. The **Startup...** button becomes active.
3. Click the **Startup...** button to toggle between **Automatic** and **Manual** Startup modes.
4. Verify that the **Driver Runtime** task is selected (highlighted), and then click **OK** to close the dialog.

Troubleshooting

If the ABENI driver fails to communicate with the device, the tag you configured for the **Read Status** or **Write Status** fields receives an error message. This error message contains an error code, which you can use to identify the type of failure that occurred.

The following table describes all of the error codes:

Error Code	Description	Possible Causes	Procedure to Solve
0	OK	Communication without problems	None required
1	Invalid Block Size	Offset specified for the Driver Configuration worksheet is too big and the message cannot be framed	Change offsets or create a new worksheet.
2	Invalid Address	Wrong Format type specified for **Address** field	Type the correct Format type.
3	Invalid Header	Wrong Data type specified for **Header** field	Type the correct Data Type.
4	Invalid Station	Wrong value specified for **Station** field	Type the correct value on the **Station** field.
5	Invalid Command	Wrong address configured	Type the correct address.
24	Invalid Family	Wrong family configured in Communication Parameters	Type a valid family.
-15	Timeout Waiting to Start a Message	▪ Disconnected cables ▪ PLC turned off, in Stop or Error mode ▪ Wrong Station number ▪ Wrong RTS/CTS control settings	▪ Check cable wiring. ▪ Check PLC state. It must be RUN. ▪ Check station number. ▪ See “Link Characteristics” section for valid RTS/CTS configurations. Check required configuration.
-17	Timeout Between rx char	▪ PLC in stop or error mode ▪ Wrong station number ▪ Wrong parity ▪ Wrong RTS/CTS configuration settings	▪ Check cable wiring. ▪ Check PLC state. It must be RUN. ▪ Check station number. ▪ See “Link Characteristics” section for valid RTS/CTS configurations. Check required configuration.

⇨ **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 enable the log at the unit (**Tools**/**LogWin**) and verify the `celog.txt` file created at the target unit.

If you are unable to establish communication with the PLC, you must first 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 you successfully establish communication between the PLC Programming Software 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.**
- **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

You will find a sample application for drivers in the **`/COMMUNICATION EXAMPLES/ABENI`** directory. We strongly recommend that you check for 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 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 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	Eric Vigiani	Sep/12/2003	▪ First driver version
B	1.01	Eric Vigiani	Jan/19/2004	▪ Fixed problem with the Station field
C	1.02	Fabio H.Y.Komura	Jul/15/2004	▪ Fixed problems with Octal (PLC5) ▪ Fixed bug when "Write with Header" offset is different from 0 (zero) ▪ Implemented the Main Driver Sheet (MDS)
D	1.03	Fabio H.Y.Komura	Oct/29/2004	▪ Fixed bug when "Read with Header" offset is different from 0 (zero) ▪ Implemented communication with multiple Station IPs.
E	1.04	Fabio H.Y.Komura	Mar/30/2006	▪ Implemented String Header.
F	1.05	Eric Vigiani	Oct/04/2006	▪ Fixed problem with String and Float header.
G	1.06	Plínio M. Santana	Mar/07/2007	▪ Fixed problem with Address formats and negative values for Timers, Counters and Controls.
H	1.06	Plínio M. Santana	May/11/2007	▪ Document revised.
I	1.06	Jonathan C. Romanus	May/18/2007	▪ Modified Sample Addresses Configuration Table.
J	1.07	Eric Vigiani	Nov/19/2007	▪ Modified the driver to work with the device series A, B, C and D.
K	1.07	Andre Bastos	Dec/23/2008	▪ Removed support to PLC5 from documentation
L	1.09	Fellipe Peternella	Mar/20/2009	▪ Added support to PLC5 ▪ Fixed problem with writing to Timers, Counters and Registers
M	1.10	André Körbes Fellipe Peternella Joel Nascimento Paulo Balbino	May/25/2010	▪ Fixed several bugs with BCD, address calculation and error checking.
N	1.11	André Körbes	Oct/01/2010	▪ Fixed bugs with address calculation and improved driver stability.