ALNET Communication Driver

Driver for Serial Communication
with Devices Using Alnet I Protocol

Contents

Introduction

The ALNET driver enables communication between the Studio system and devices using the Alnet 1 protocol communicating over serial, according to the specifications discussed in this document.

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

- **Introduction**: Provides an overview of the ALNET driver documentation.
- **General Information**: Provides information needed to identify all the required components (hardware and software) used to implement communication between Studio and the ALNET driver.
- **Installing the Driver**: Explains how to install the ALNET driver.
- **Configuring the Driver**: Explains how to configure the ALNET 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 ALNET 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 XP environment.
  If you are unfamiliar with Windows 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 ALNET driver and the Alnet 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**: Altus
- **Compatible Equipment**: AL1000, AL2000, AL3000 and QUARK series
- **Programming Software**: MasterTool
- **Device Runtime Software**: None

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**: COM2
- **COM**: COM2
- **Baud Rate**: 9600
- **Data Bits**: 8
- **Stop Bits**: 1
- **Parity**: Even

Driver Characteristics

The ALNET driver is composed of the following files:

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

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

- Windows XP/Vista/7/Server 2003/2008
- Windows CE

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

The ALNET driver supports the following registers:

Register Type	Length	Write	Read	Bit	Integer	Float	DWord
%A (Auxiliary)	1 Byte	●	●	●	●	–	–
%E (Octet: I/O)	1 Byte	●	●	●	●	–	–
%M (Memory)	1 Word	●	●	●	●	–	–
%D (Decimal)	2 Words	●	●	●	●	–	●
%TM (Memory Table)	1 Word	●	●	●	●	–	–
%TD (Decimal Table)	2 Words	●	●	●	●	–	●
%F (Float)	2 Words	●	●	●	●	●	–
%TF (Float Table)	2 Words	●	●	●	●	●	–

Conformance Testing

The following hardware/software was used for conformance testing:

- **Driver Configuration**:
  – **Protocol**: Alnet 1
- **Cable**: RS-232

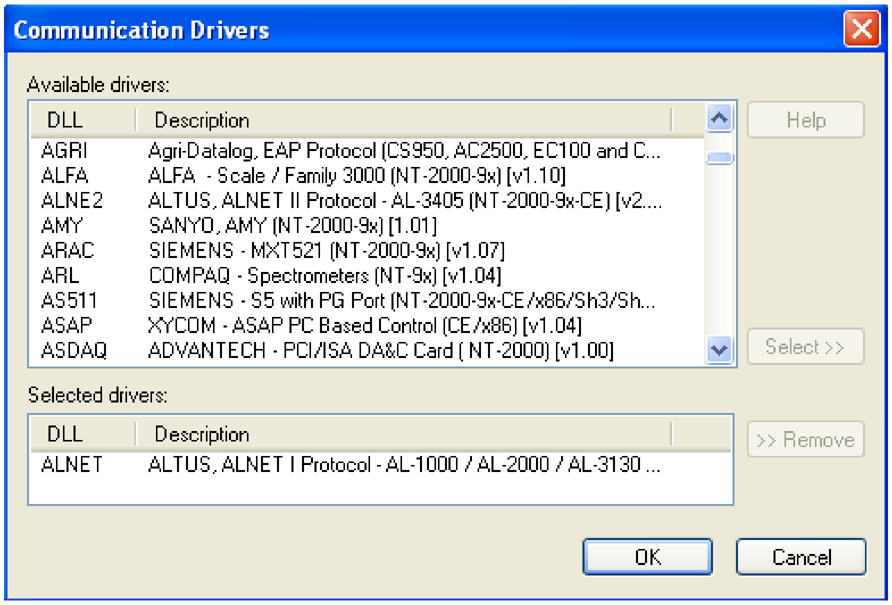
Driver Version	Studio Version	Operating System	Equipment
1.67	7.0	WinXP+SP3 WinCE v5	AL2000

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 **ALNET** driver from the *Available Drivers* list, and then click the **Select** button:



Available drivers:

DLL	Description
AGRI	Agri-Datalog, EAP Protocol (CS950, AC2500, EC100 and C...
ALFA	ALFA - Scale / Family 3000 (NT-2000-9x) [v1.10]
ALNE2	ALTUS, ALNET II Protocol - AL-3405 (NT-2000-9x-CE) [v2....
AMY	SANYO, AMY (NT-2000-9x) [1.01]
ARAC	SIEMENS - MXT521 (NT-2000-9x) [v1.07]
ARL	COMPAQ - Spectrometers (NT-9x) [v1.04]
AS511	SIEMENS - S5 with PG Port (NT-2000-9x-CE/x86/Sh3/Sh...
ASAP	XYCOM - ASAP PC Based Control (CE/x86) [v1.04]
ASDAQ	ADVANTECH - PCI/ISA DA&C Card (NT-2000) [v1.00]

Selected drivers:

DLL	Description
ALNET	ALTUS, ALNET I Protocol - AL-1000 / AL-2000 / AL-3130 ...

Communication Drivers Dialog

5. When the **ALNET** 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 MasterTool programmer software. Consult your MasterTool 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 ALNET driver, you must configure the driver. Configuring the ALNET driver is done in two parts:

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

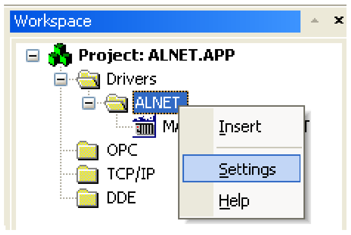
Worksheets are divided into two sections, a *Header* and a *Body*. The fields contained in these two sections are standard for all communications drivers — except the **Station**, **Header**, and **Address** fields, which are driver-specific. This document explains how to configure the **Station**, **Header**,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

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

Setting the Communication Parameters

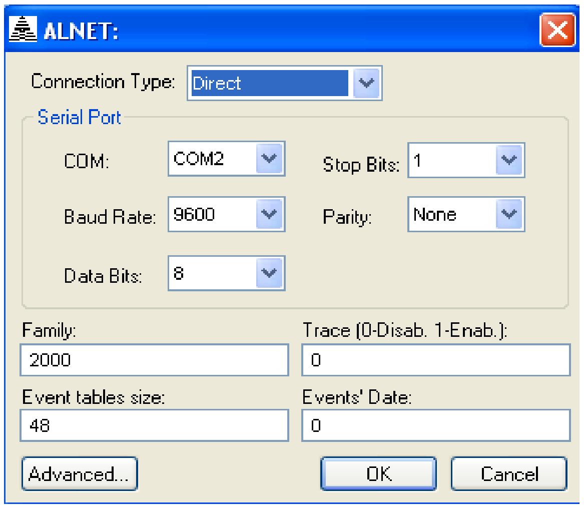
Use the following steps to configure the communication parameters, which are valid for all Driver Worksheets configured in the system:

1. From the Studio development environment, select the **Comm** tab located below the *Workspace* pane.
2. Click on the *Drivers* folder in the *Workspace* pane to expand the folder.
3. Right-click on the *ALNET* subfolder and when the pop-up menu displays, select the **Settings** option:



Select Settings from the Pop-Up Menu

The *ALNET: Communications Parameters* dialog displays:


ALNET:

Connection Type: Direct

Serial Port
COM: COM2
Stop Bits: 1
Baud Rate: 9600
Parity: None
Data Bits: 8

Family: 2000
Trace (0-Disab. 1-Enab.): 0
Event tables size: 48
Events' Date: 0

Communication Parameters Dialog

4. Specify the custom parameters as noted in the following table:

Parameters	Default Values	Valid Values	Description
Family	2000	<Family number>	Use 1000 for PLC AL1000 Use 2000 for PLC AL2000 and QUARK series
Event tables size	64	Integer value	Event table size inside of PLC program (number of bytes).
Event's Date	0	0 or 1	0 – Use the **PLC** information to generate the event's date. 1 – Use the **Computer** information to generate the event's date.
Trace **(0-Disab. 1-Enab.)**	0	0 or 1	0 – Disable debug information. 1 – Enable debug information. This information will be displayed in the LOGWIN.

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

5. Click the **Advanced** button on the *Communication Parameters* dialog to open the *Advanced Settings* dialog and configure the settings that 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.

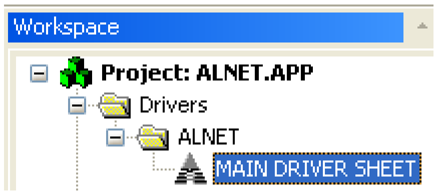
Configuring the Driver Worksheets

This section explains how to configure the *MAIN DRIVER SHEET* and *STANDARD DRIVER SHEETs* (or Communication tables) to associate application tags with the device addresses. You can configure multiple *Standard Driver Sheets* — each of which is divided into a *Header* section and *Body* section, and/or multiple independent lines on the *Main Driver Sheet*.

▪ Main Driver Sheet

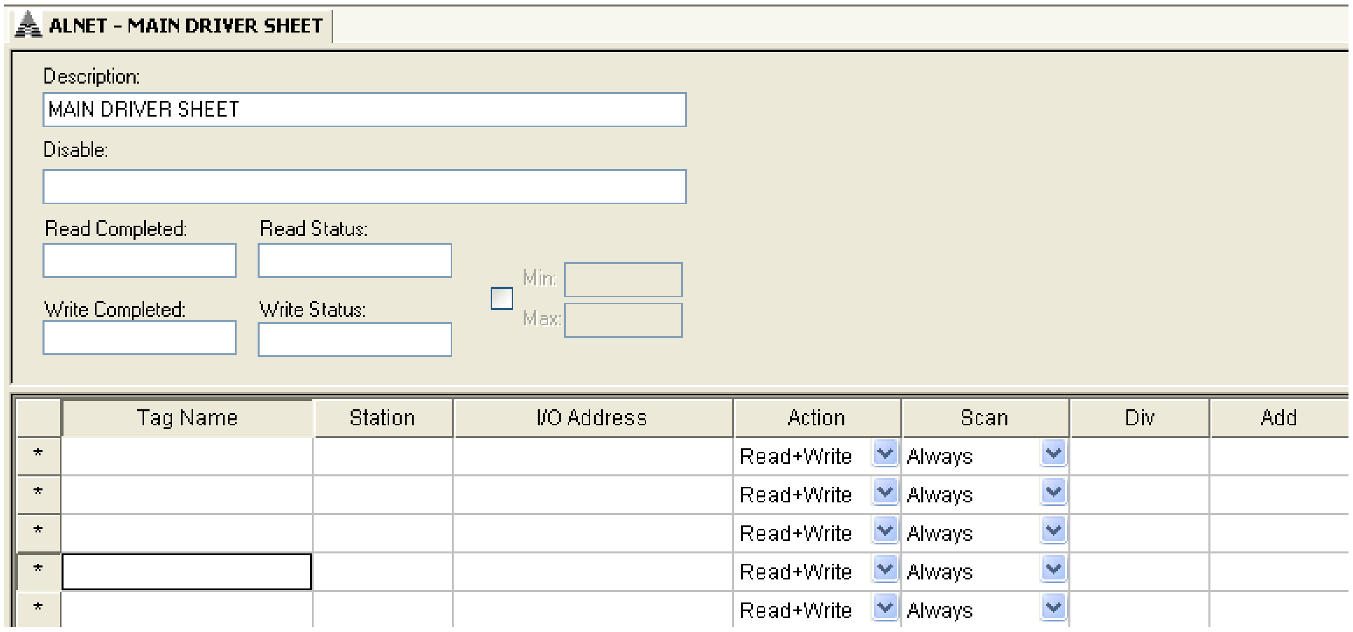
When you select the ALNET driver and add it to your application, Studio automatically inserts the *Main Driver Sheet* in the *ALNET* driver subfolder. To configure the Main Driver Sheet:

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



Main Driver Sheet in the ALNET Subfolder

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


ALNET - 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
*				Read+Write	Always		
*				Read+Write	Always		
*				Read+Write	Always		
*				Read+Write	Always		
*				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 ALNET 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 point-to-point connections, leave blank.
     - For networks, use the following syntax:

       `<node>:<sub network>`

     Example — **`10:3, 5:1`**

     Where:

     – ***`<node>`*** is the node identifier.

     – ***`<sub network>`*** is the subnetwork identifier. This parameter is optional. If it is not specified, the value 0 (zero) will be used as default.

     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.

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

     `%<Type>:<Address>`

     `Or`

     `%<TM, TD or TF><AddressReference>:<Address>`

     Where:

     – ***`<Type>`*** : Register type. Valid values are **`A`**, **`E`**, **`M`**, **`D`**, **`F`**, **`STA`**.

       - For tables (registers of type **`TM`**, **`TD`** and **`TF)`**, the address reference must be supplied, as a decimal unsigned number, indicating the table number.
       - Offsets are not supported on the Main Driver Sheet.

     – ***`<Address>`*** : Address of the device register. Follows the same syntax explained on the Standard Driver Sheet section below for the **Address** field.

     Examples:
     – **`%A:0`**
     – **`%A:1.5`**
     – **`%D:4b2`**
     – **`%D:3`**
     – **`%TM0010:4`**
     – **`%TM0010:4b1`**

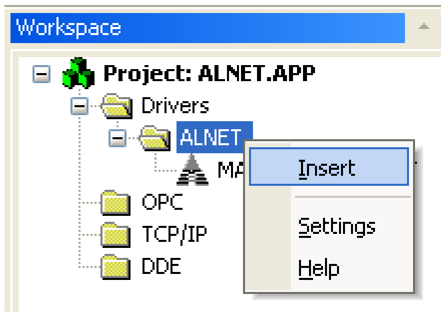
➲ **Attention:**

- Unsolicited messages, F-ARQ headers and event addresses are not supported on the main driver sheet.
- Offsets for the valid headers, including tables, are not supported and are eliminated on validation.

▪ STANDARD DRIVER SHEET

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

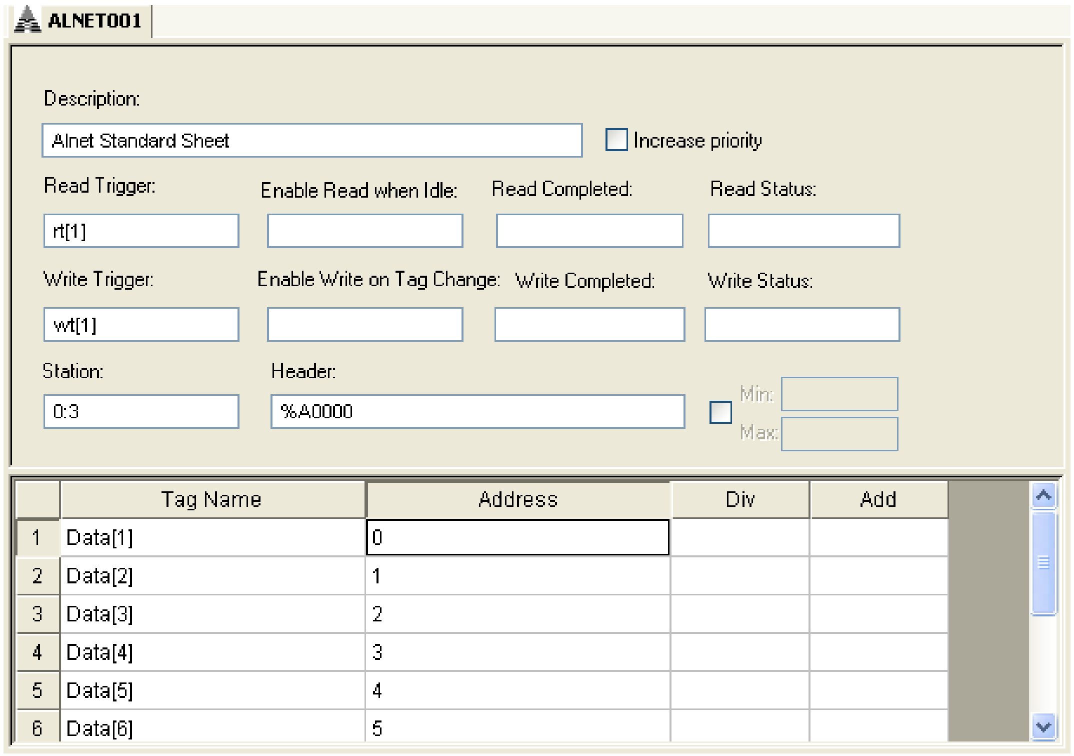
1. From the Studio development environment, select the **Comm** tab, located below the *Workspace* pane.
2. In the *Workspace* pane, expand the *Drivers* folder and right-click the *ALNET* subfolder.
3. When the pop-up menu displays, select the **Insert** option:



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):



ALNET001

Description: Alnet Standard Sheet
Increase priority: ☐

Read Trigger:	Enable Read when Idle:	Read Completed:	Read Status:
rt[1]			

Write Trigger:	Enable Write on Tag Change:	Write Completed:	Write Status:
wt[1]			

Station: 0:3
Header: %A0000
☐ Min: Max:

	Tag Name	Address	Div	Add
1	Data[1]	0		
2	Data[2]	1		
3	Data[3]	2		
4	Data[4]	3		
5	Data[5]	4		
6	Data[6]	5		

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: Specify the device using the following syntax:

`<node>:<sub network>`

Where:

- **Node** is the node identifier.
- **Sub network** is the subnetwork identifier. This parameter is optional. If it is not specified, the value 0 (zero) will be used as default.

– **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:

`%<Type><AddressReference>` (for example: **`%A0010`**)

or

`%<TM, TD or TF><AddressReference>[TablePositionReference]` (for example: **`%TM0010[5]`***)*

Where:

* **Type** is the register type (**A**, **E**, **M**, **D**, **TM**, **TD**, **F**, **TF**).

* **AddressReference** is the initial address (reference) of the configured type.

* **TablePositionReference** is the initial position (reference) of the configured table.

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.

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

Header Field Information			
Data Types	**Sample Syntax**	**Valid Range of Initial Addresses per Worksheet**	**Comments**
A	`%A0000`	Varies according to the equipment	Auxiliary: Read and write integer values. The operand is a BYTE data type.
E	`%E0000`	Varies according to the equipment	Octet: Read and write integer values in the inputs or outputs image. The operand is a BYTE data type.
M	`%M0000`	Varies according to the equipment	Memory: Read and write integer values. The operand is a WORD data type.
D	`%D0000`	Varies according to the equipment	Decimal: Read and write BCD values. The operand is a DWORD data type.
TM	`%TM0000`	Varies according to the equipment	Memory Table: Read and write integer values. The operand is a WORD data type.
TD	`%TD0000`	Varies according to the equipment	Decimal Table: Read and write BCD values. The operand is a DWORD data type.
TF	`%TF0000`	Varies according to the equipment	Memory Table: Read and write float values. The operand is a FLOAT data type.
F	`%F0000`	Varies according to the equipment	Decimal Table: Read and write float values. The operand is a FLOAT data type.
STA	`STA`	0-59	Status codes of the PLC.

– **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:

***<AddressOffset>*[type][number]** (for example: **10**, **20.1**, **40n0**)

Where:

* **AddressOffset** is a parameter added to the **AddressReference** parameter (configured in the **Header** field) to compose the group address configured in the **Header** field. When the **TablePositionReference** parameter is configured in the header, the **AddressOffset** is added to compose the group address.

* **Type** (*optional parameter used to define the type of operator piece*).

Where:

.: bit (zero-based index of the bit of the operand)
n: nibble
b: byte (to memory and decimal operand type)
w: word (only to decimal operand type)
h: bit of second word (only to decimal operand type)

* **Number** (*optional parameter used with **type** parameter*) is the number of operand piece to be read from or written to the device. This number is on hexadecimal format.

Examples:

* **10** – The operand at position 10

* **20.1** – The bit 1 (zero-based index) of operand at position 20

* **40n0** – The nibble 0 of operand at position 40

➲ **Attentions:**

The bit write commands will write the zero value in the other bits of the operand.

Address Configuration Sample		
Device Address	**Header Field**	**Address Field**
%A0001	%A0001	0
%A0010	%A0000	10
%S0000	%E0000	0
%M0020	%M0010	10
%M0020	%M0000	20
%M0020	%M0020	0
%D0015	%D0002	13
%TM0000 position 0	%TM0000	0
%TM0005 position 3	%TM0005[3]	0
%TM0005 position 3	%TM0005	3
%TD0002 position 5	%TD0002[3]	2
%A0005 bit 0	%A0005	0.0
%M0003 nibble 1	%M0000	3n1
%D0007 word 0	%D0005	2w0
%D0007 word 1	%D0005	2w1
%F0007	%F0000	7
%TF0001	%TF0001	0

➲ **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.

Device Configuration

- **Solicited Communication**
  The solicited communication is executed when a read or a write command is executed in the Studio. The only configuration in the device is:
  - Configuration of node and sub network.
  - Creation of operands to read and write data value. The operands creation must be using the MasterTool programmer software.

- **Unsolicited Communication**
  The unsolicited communication is executed when a read or a write command is executed in the PLC. The configuration in the device is:
  - Configuration of node and sub network.
  - Creation of operands to read and write data value. The operands creation must be using the MasterTool programmer software.
  - Must be created ECR block in the PLC program where you configure the PLC node and the subnetwork must be every 64.

Reading Files from the PLC

This section explains how to read F-ARQ modules from the PLC. These modules are used to store large amounts of data on the PLC, and are read using a special configuration of header, tag and address. The file read from the PLC is stored locally on a format appropriate to be processed through a recipe file by Studio.

F-ARQ files can be read only by Standard Driver Sheets. On these sheets, the header must be configured using the following syntax:

<*File Name*>.<Module Name>:<Registers>:<Fields>:<Bytes per Register>

Example:

F-ARQ.035:100:9:2

The destination of the file on the computer file system is configured on the first and only row on the Address field, like: **C:\input.txt**

The status of the operation is then stored on the tag configured for the row, which should have Integer type. Errors are associated with problems opening and/or creating the file, such as incorrect paths, permission level inadequate, etc., and have values differing from zero. Studio does not create directories and does not validate the address typed.

Unsolicited Messages

Any message that arrives without being requested is considered an unsolicited message. These messages are processed by the sheet with the same station number and header sent on the message. If no sheet applies for this rule, the message is discarded. Only Standard Driver Sheets are considered for searching for an appropriate station and header. Thus, stations and headers configured on the Main Driver Sheet are not considered for this kind of message.

➲ **Attention:**
You must configure on station the address of the receiver, that is, the address where to the PLC sends the message.

Events

Studio is able to process received messages as events, using a special configuration of station, header and address, valid only for Standard Driver Sheets.

- Station:
  - Address of the PLC where the event table is stored (for pooling), or
  - Address of the system that will receive the event table (as unsolicited message)
- Header:
  - The event table is stored as a memory table, with type **`%TM`**
  - The header must be set to the memory table to store or to send the event table, without offset
  - E.g.: **`%TM0001`** is valid, while **`%TM0001[10]`** is not
- Address:
  - The address must follow a special syntax: **`<Rack>:<Octet>.<Bit>`**
  - Where Octet ranges from 0 to 3 and Bit from 0 to 7

If the BGTASK is open and the event tags are configured on the Alarm module, the Driver will set these alarms with corresponding timestamps. These configurations are also valid for Unsolicited Messages.

Reading the Status of the PLC

A special header is used to read the status of the PLC. The header **`STA`** describes this special meaning, not supporting offset, and using a number ranging from 0-59 as the address. Each number reads a different status, explained on the table below. Some of the values returned are codes indicating values to be interpreted using different tables, noted when applicable.

Status Codes (Header STA)		
Address	**Status**	**Range of Values**
0	CPU Model.	See Table
1	Version Number of current program.	-
2	Revision and minor version of current program. Must be read as two nibbles.	-
3	Digital outputs disabled.	0-1
4	PLC is compressing RAM	0-1
5	PLC is forcing relays	0-1
6	PLC is copying a module from EPROM to RAM	0-1
7	PLC is on test mode	0-1
8	PLC is on cyclic mode	0-1
9	PLC is on programming mode	0-1
10	PLC is on run mode	0-1
11	Error Codes. See manufacturer documentation.	-
12	Free RAM on bank 1 in bytes.	-
13	Free RAM on bank 2 in bytes (if available).	-
14	RAM Status. Existing banks for current program. (Bank 0)	-
15	RAM Status. Existing banks for current program. (Bank 1)	-
16	RAM Status. Existing banks for current program. (Bank 2)	-

Status Codes (Header STA)		
Address	**Status**	**Range of Values**
17	RAM Status. Existing banks for current program. (Bank 3)	-
18	RAM Status. Existing banks for current program. (Bank 4)	-
19	RAM Status. Existing banks for current program. (Bank 5)	-
20	RAM Status. Existing banks for current program. (Bank 6)	-
21	RAM Status. Existing banks for current program. (Bank 7)	-
22	Instantaneous scan time in milliseconds.	-
23	Average scan time in milliseconds.	-
24	Maximal scan time in milliseconds.	-
25	Minimal scan time in milliseconds.	-
26	Module e-018 call time period in milliseconds. 255 if not available.	-
27	Module e-019 call time period in milliseconds. 255 if not available.	-
28	Reserved	-
29	Maximal program execution time.	See Table
30	RAM is not compressed.	0-1
31	Protection Level.	0-3
32	Erasing EPROM.	0-1
33	I/O modules exchange energized	0-1
34	RAM free space. (Bank 8)	-
35	RAM free space. (Bank 7)	-
36	RAM free space. (Bank 6)	-
37	RAM free space. (Bank 5)	-
38	RAM free space. (Bank 4)	-
39	RAM free space. (Bank 3)	-
40	EPROM Status. Availability. (Bank 1)	0-1
41	EPROM Status. Availability. (Bank 2)	0-1
42	EPROM Status. Availability. (Bank 3)	0-1
43	EPROM Status. Availability. (Bank 4)	0-1
44	EPROM Status. Availability. (Bank 5)	0-1
45	EPROM Status. Availability. (Bank 6)	0-1
46	EPROM Status. Availability. (Bank 7)	0-1
47	EPROM Status. Availability. (Bank 8)	0-1
48	EPROM Free space in bytes. (Bank 8)	-
49	EPROM Free space in bytes. (Bank 7)	-
50	EPROM Free space in bytes. (Bank 6)	-
51	EPROM Free space in bytes. (Bank 5)	-
52	EPROM Free space in bytes. (Bank 4)	-
53	EPROM Free space in bytes. (Bank 3)	-
54	EPROM Free space in bytes. (Bank 2)	-
55	EPROM Free space in bytes. (Bank 1)	-

Status Codes (Header STA)		
Address	**Status**	**Range of Values**
56	Message 2. Indicates message 2 code.	-
57	Message 3. Indicates message 3 code.	-
58	Message 4. Indicates message 4 code.	-
59	ASCII Message. Must be read on a string tag.	-

Value	CPU Model
0	AL-3003
1	AL-3004
32	AL-2000
33	AL-2002
34	QK2000
64	AL-600
80	QK800
81	QK801
128	AL-2400
129	AL-2401
136	QK2400
137	QK2401
143	AL-2400/S

Value	Maximal program execution time (ms)
0	100
1	200
2	300
3	400
4	500
5	600
6	700
7	800

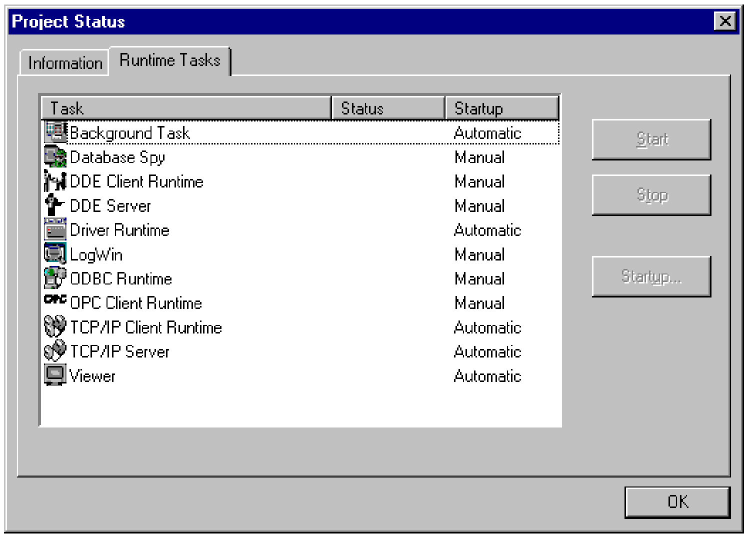
Executing the Driver

After adding the ALNET 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:



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
TCP/IP Client Runtime		Automatic
TCP/IP Server		Automatic
Viewer		Automatic

Project Status Dialog

2. 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**.
3. Click **OK** to close the *Project Status* dialog.
4. Start the application to run the driver.

Troubleshooting

If the ALNET 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.
5	Invalid Block size	Offset is greater than the maximum allowed. The maximum offset is usually 64.	Specify a valid offset or create a new Driver Worksheet.
6	Invalid Station	Invalid Station	Specify a valid station in the Driver Worksheet.
7	Invalid Header field	Invalid tag value in the Header field.	Specify a valid tag value in the Header field.
8	Invalid Address field	Invalid Address.	▪ Check the initial address in the Driver Worksheet. ▪ Check the Holding register in the Driver Worksheet with bit configuration. This parameter cannot execute write triggers—it executes "Write on Tag Change" only. ▪ Retype the address in the Driver Worksheet.
45	Unsolicited Message Error	The driver can not find anyone driver worksheet to treat the received unsolicited message.	Configure any Driver Worksheet compatible with the received unsolicited message. Look for Header of unsolicited message received in the LogWin module.
46	Checksum Error	The calculated checksum differs from the received one from the PLC	Contact Studio's support.
100	Timeout Start Message	▪ Disconnected cables. ▪ PLC is turned off, in stop mode, or in error mode. ▪ Wrong station number. ▪ Wrong RTS/CTS control settings.	▪ Check cable wiring. ▪ Check the PLC state – it must be RUN. ▪ Check the station number. ▪ Check the configuration. See *Studio Technical Reference Manual* for information about valid RTS/CTS configurations.
230	Invalid Command	Invalid Command	Specify a valid command.
241	Invalid Event Date	Invalid Event Date	Specify a valid Event Date (0 – use the PLC information to generate the event's date, 1 – use the computer information to generate the event's date).
242	Invalid Event Size	Invalid Event Size	Specify a valid Event Size (Event table size inside of PLC program – number of bytes).
243	Invalid PLC Family	Invalid PLC Family	Specify a valid PLC Family (1000 for PLC AL1000 or 2000 for PLC AL2000 and QUARK series).
244	Invalid Trace	Invalid Trace	Specify a valid Trace (0 – Disable, 1 - Enable).

⇨ **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 **Protocol Analyser** 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 Remote runtime system (e.g. a Windows CE target), you can use the Remote LogWin of Studio to get the communications log 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

There is no Sample Application available for this driver

Revision History

Doc. Revision	Driver Version	Author	Date	Description of changes
A	1.65	Diego Barros	Jan/3/2006	Implemented read/write of Float and Table Float registers (%F and %TF)
B	1.66	André Körbes	May/25/2010	-Included documentation for F-ARQ, Status, Events, Unsolicited Messages and implemented Main Driver Sheet -Improved documentation on driver configuration
C	1.67	André Körbes	Aug/04/2011	Fixed unsolicited messages on driver sheets that do not start with address 0