

PAC3K

Contents

PAC3K Driver	3
Driver specifications.....	4
Configuring the device's communication settings	7
Adding a communication driver to your project	8
Configuring the driver's communication settings.....	8
About driver worksheets	9
<i>Adding and configuring a Standard Driver Sheet.....</i>	<i>10</i>
<i>Configuring the Main Driver Sheet.....</i>	<i>16</i>
<i>Additional notes.....</i>	<i>19</i>
Checking the Driver Runtime task.....	20
Troubleshooting	20
Revision history.....	25

PAC3K Driver

PAC3K Driver for Communication with Automation Direct Programmable Automation Controller (PAC) 3000 devices (version 1.4, last revised 15 Oct 2015).

The PAC3K driver enables communication between the Studio system and remote devices using the Modbus Extended protocol, according to the specifications discussed in this document.

This document assumes that you have read the "Development Environment" section in the main Studio documentation.

This document also assumes that you are familiar with the Microsoft Windows XP/Vista/7 environment. If you are not familiar with Windows, then we suggest using the **Help and Support** feature (available from the Windows **Start** menu) as you work through this document.


Driver specifications

This section identifies all of the software and hardware components required to implement communication between the PAC3K driver in Studio and remote devices using the Modbus Extended protocol.

Driver files

The PAC3K driver package comprises the following files, which are automatically installed in the `Drv` folder of the Studio application directory:

- `PAC3K.DLL`: Compiled driver.
- `PAC3K.INI`: Internal driver file. *You must not modify this file.*
- `PAC3K.MSG`: Internal driver file defining error messages for the possible error codes. (These error codes are described in detail in the [Troubleshooting](#) section.) *You must not modify this file.*
- `PAC3K.PDF`: This document, which provides complete information about using the driver.

 **Note:** You must use a compatible PDF reader to view the `PAC3K.PDF` file. You can install Acrobat Reader from the Studio installation CD, or you can download it from [Adobe's website](#).

You can use the PAC3K driver on the following operating systems:

- Windows XP, Windows Server 2003
- Windows Vista/7/8, Windows Server 2008
- Windows Embedded Compact 5.x/6.x

Device specifications

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

- Manufacturer: Automation Direct
- Compatible Equipment: Programmable Automation Controller (PAC) 3000 with firmware above 1.1.12.14
- Programmer Software:

The PAC3K driver supports the following device registers:

Supported registers

System IDs	Description
C	Internal Bit
SBR	System Read only Bit
SBRW	System Read/Write Bit
US8	Unsigned 8 bits
S16	Signed Integer 16 bits
BCD16	BCD 16bits
US16	Unsigned Integer 16bits
SWR	System Read only Word
SWRW	System Read/Write Word
S32	Signed Integer 32bits

BCD32	BCD 32bits
F32	Float 32bits
STR	String
SSTR	System String
AR1C	Array 1D bit
AR1US8	Array 1D Unsigned 8bits
AR1S16	Array 1D Signed Integer 16bits
AR1US16	Array 1D Unsigned Integer 16bits
AR1BCD16	Array 1D BCD 16bits
AR1S32	Array 1D Signed Integer 32bits
AR1BCD32	Array 1D BCD 32bits
AR1F32	Array 1D Float 32bits
AR1STR	Array 1D String
AR2C	Array 2D Bit
AR2US8	Array 2D Unsigned 8bits
AR2S16	Array 2D Signed Integer 16bits
AR2US16	Array 2D Unsigned Integer 16bits
AR2BCD16	Array 2D BCD16bits
AR2S32	Array 2D Signed Integer 32bits
AR2BCD32	Array 2D BCD 32bits
AR2F32	Array 2D Float 32bits
AR2STR	Array 2D String
DI	Discrete In
DO	Discrete Out
MST	Module Status Bit
AIS32	Analog In Signed 32bits
AOS32	Analog Out Signed 32bits
AIF32	Analog In Float 32bits
AOF32	Analog Out Float 32bits

Network specifications

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

- Device Communication Port: 502
- Physical Protocol: TCP/IP
- Logic Protocol: Modbus Extended
- Device Runtime Software: None
- Specific PC Board: None
- Cable Wiring Scheme: None

Conformance testing

The following hardware/software was used for conformance testing:

- Driver Version: 1.4
- Studio Version: 8.0
- Operating System (development): Windows 7/8
- Operating System (target): Windows 7/8
- Equipment: P3-550 CPU
- Communication Settings:
- TCP/IP Port: 502
- Cable: Use specifications described in the "Network Specifications" section above.

Configuring the device's communication settings

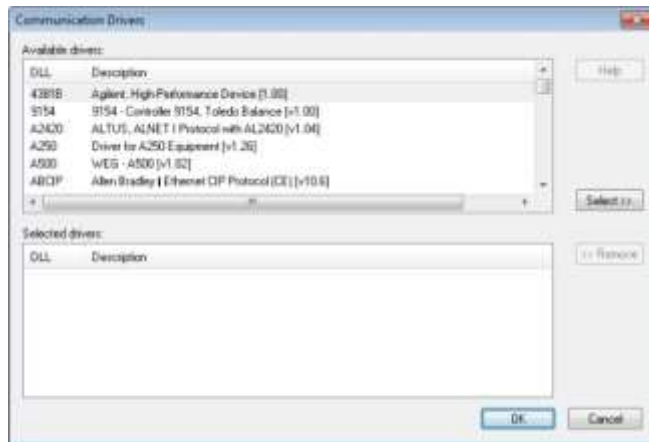
This section explains how to configure the communication settings for the remote device.

All communication settings are handled automatically by the PAC3K protocol and development software. You do not need to configure anything on the target device.

Adding a communication driver to your project

This section explains how to add a communication driver to your project.

1. On the **Insert** tab of the ribbon, in the **Communication** group, click **Add/Remove Driver**.
The *Communication Drivers* dialog is displayed.



Communication Drivers dialog

2. In the *Available drivers* list, click the communication driver that you want to add.
3. Click **Select**.
The driver is added to the *Selected drivers* list.
4. Click **OK**.
The *Communication Drivers* dialog is closed and the selected driver is inserted in the **Drivers** folder in the Project Explorer.

Configuring the driver's communication settings

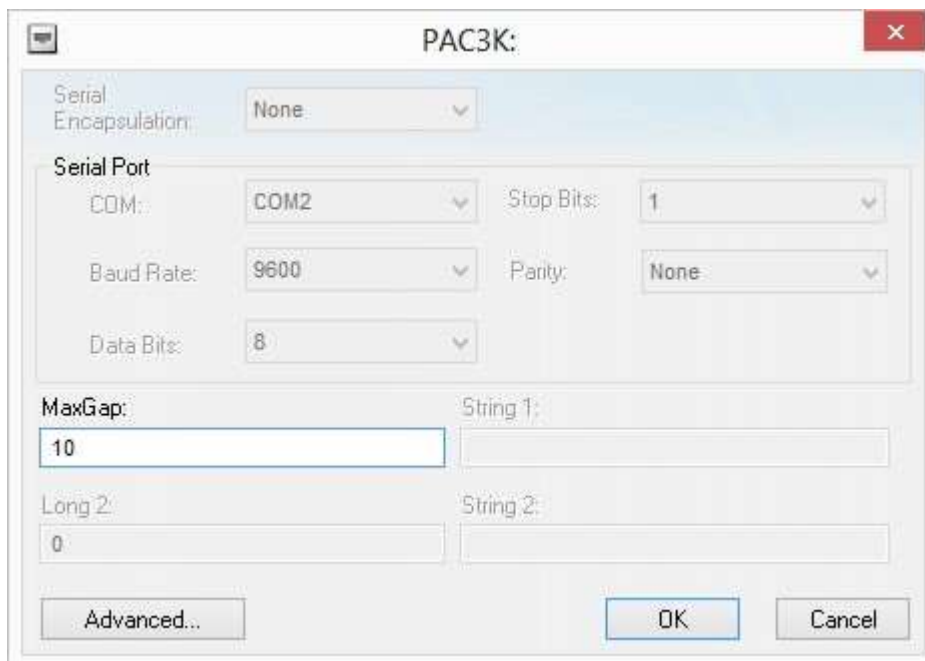
This section explains how to configure the communication settings for the driver.

You must add the communication driver to your project before you can configure its settings. For more information, see [Adding a communication driver to your project](#) on page 8.

The general procedure for configuring a driver's communication settings is the same for all drivers. However, the specific settings are different for each driver, depending on the options and protocols used by the target device.

To configure the communication settings:

1. In the **Comm** tab of the Project Explorer, expand the **Drivers** folder.
The folder contains the drivers that are currently enabled. If you do not see the driver that you want to configure, then you need to add it.
2. Right-click the driver that you want to configure, and then click **Settings** on the shortcut menu.
The *Communication Settings* dialog is displayed.



Communication Settings: PAC3K dialog

- Configure the remaining, driver-specific settings as needed.

Driver-specific communication settings

Setting	Default Value	Possible Values	Description
MaxGap	10	1 to 999999	It is set to specify the maximum difference between the addresses that is allowed to read in each block when reading in multiple blocks.

- Click **OK**.

The settings are saved and the *Communication Settings* dialog is closed.

About driver worksheets

Like the other parts of your project, communication with remote devices is controlled by worksheets. This section explains how to add worksheets to your project and then configure them to associate project tags with device registers.

Each selected driver includes a Main Driver Sheet (MDS) and one or more Standard Driver Sheets (SDS). The Main Driver Sheet is used to define tag/register associations and driver parameters that are in effect at all times, regardless of project behavior. In contrast, Standard Driver Sheets can be inserted to define tag/register associations that are triggered by specific project behaviors.

The configuration of these worksheets is described in detail in the "Communication" chapter of the *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 PAC3K driver-specific parameters and procedures are discussed here.

Adding and configuring a Standard Driver Sheet

By default, a communication driver does not include any Standard Driver Sheets. This section explains how to add a Standard Driver Sheet to your project and then configure it.

The PAC3K driver must be added to the project before you can configure any of its worksheets. For more information, see [Adding a communication driver to your project](#) on page 8.

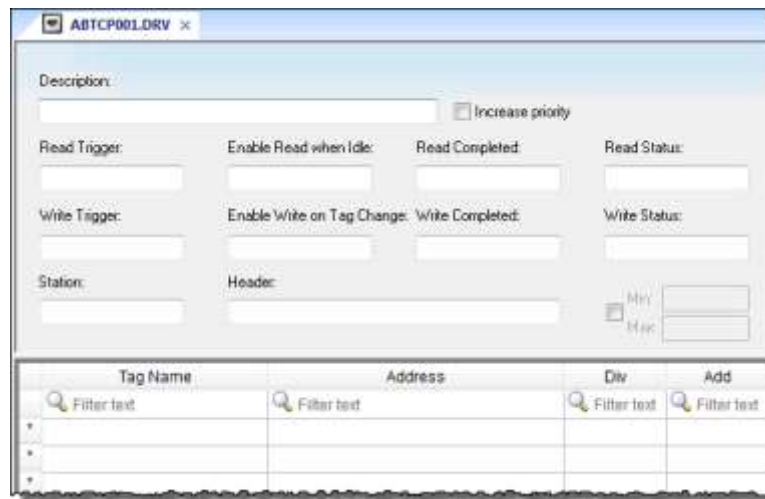
Standard Driver Sheets can be inserted to define additional tag/register associations that are triggered by specific project behaviors.

Note: Most of the settings on this worksheet are standard for all drivers; for more information about configuring these settings, see the “Communication” chapter of the *Technical Reference Manual*. The **Station** and **I/O Address** fields, however, use syntax that is specific to the PAC3K driver.

1. Do one of the following.

- On the **Insert** tab of the ribbon, in the **Communication** group, click **Driver Sheet** and then select **PAC3K** from the list.
- In the **Comm** tab of the Project Explorer, right-click the **PAC3K** folder and click **Insert** on the shortcut menu.

A new PAC3K driver worksheet is inserted into the **PAC3K** folder, and then it is automatically opened for configuring.



Standard Driver Sheet

Note: Worksheets are numbered in order of creation, so the first worksheet is PAC3K001.drv.

2. Configure the Station and Header fields as described below.

Station

Specify the IP address of the target PAC device.

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

Note: You cannot leave the **Station** field blank.

Header

Specify one of the following register types:

C

To access the Internal Bit

SBR

To access the System Read only Bit

SBRW

To access the System Read/Write Bit

US8

To access the Unsigned 8 bits

S16

To access the Signed Integer 16 bits

BCD16

To access the BCD 16bits

US16

To access the Unsigned Integer 16bits

SWR

To access the System Read only Word

SWRW

To access the System Read/Write Word

S32

To access the Signed Integer 32bits

BCD32

To access the BCD 32bits

F32

To access the Float 32bits

STR

To access the String

SSTR

To access the System String

AR1C.ID

To access the Array 1D bit

AR1US8.ID

To access the Array 1D Unsigned 8bits

AR1S16.ID

To access the Array 1D Signed Integer 16bits

AR1US16.ID

To access the Array 1D Unsigned Integer 16bits

AR1BCD16.ID

To access the Array 1D BCD 16bits

AR1S32.ID

To access the Array 1D Signed Integer 32bits

AR1BCD32.ID

To access the Array 1D BCD 32bits

AR1F32.ID

To access the Array 1D Float 32bits

AR1STR.ID

To access Array 1D String

AR2C.ID

To access Array 2D Bit

AR2US8.ID

To access Array 2D Unsigned 8bits

AR2S16.ID

To access Array 2D Signed Integer 16bits

AR2US16.ID

To access Array 2D Unsigned Integer 16bits

AR2BCD16.ID

To access Array 2D BCD16bits

AR2S32.ID

To access Array 2D Signed Integer 32bits

AR2BCD32.ID

To access Array 2D BCD 32bits

AR2F32.ID

To access Array 2D Float 32bits

AR2STR.ID

To access Array 2D String

DI

To access Discrete In

DO

To access Discrete Out

MST

To access Module Status Bit

AIS32

To access Analog In Signed 32bits

AOS32

To access Analog Out Signed 32bits

AIF32

To access Analog In Float 32bits

AOF32

To access Analog Out Float 32bits

...where...

ID

The ID number of the array to be read.

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

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

3. For each tag/register association that you want to create, insert a row in the worksheet body and then configure the row's fields as described below.

Tag Name

Type the name of the project tag.

Address

Specify the address of the associated device register.

For C, SBR, SBRW, F32 system ID types, use the following syntax:

ID

...where...

ID

The ID portion of the specific System ID to be read.

For US8, S16, BCD16, US16, SWR, SWRW, S32, BCD32 system ID types, use the following syntax:

ID.Bit

...where...

ID

The ID portion of the specific System ID to be read.

Bit

Read a specific bit of the System ID.

This is an optional parameter. If no bit is specified, then the entire register will be read.

For AR1C, AR1F32 array system ID types, use the following syntax:

Index

...where...

Index

The index position of the specific element of the System ID array to be read.

For AR1US8, AR1S16, AR1US16, AR1BCD16, AR1S32, AR1BCD32 array system ID types, use the following syntax:

Index.Bit

...where...

Index

The index position of the specific element of the System ID array to be read.

Bit

Read a specific bit of the System ID array element.

This is an optional parameter. If no bit is specified, then the entire register will be read.

For STR, SSTR string system ID types, use the following syntax:

ID.Length

...where

... ***ID***

The ID of the specific System ID string to be read.

Length

The length of the string to be read.

For AR1STR array, use the following syntax:

Index.Length

...where...

Index

The index position of the specific element of the System ID array to be read.

Length

The length of the string to be read.

For AR2C, AR2F32 array system ID types, use the following syntax:

Index1.Index2

...where...

Index1

The first dimension index of the specific element of the System ID array to be read.

Index2

The second dimension index of the specific element of the System ID array to be read.

For AR2US8, AR2S16, AR2US16, AR2BCD16, AR2S32, AR2BCD32 array system ID types, use the following syntax:

Index1.Index2.Bit

...where...

Index1

The first dimension index of the specific element of the System ID array to be read.

Index2

The second dimension index of the specific element of the System ID array to be read.

Bit

Read a specific bit of the System ID array element.

This is an optional parameter. If no bit is specified, then the entire register will be read.

For AR2STR array, use the following syntax:

Index1 . Index2 . Length

...where...

Index1

The first dimension index of the specific element of the System ID array to be read.

Index2

The second dimension index of the specific element of the System ID array to be read.

Length

The length of the string to be read.

For DI, DO, MST, AIS32, AOS32, AIF32, AOF32, use the following syntax:

Group . Base . Slot . Bit/Channel/Module

...where...

Group

The group number (00 = Local Group, 01 – 99 = Remote system).

Base


The base number.

Slot

The slot number.

Bit/Channel/Module

The DI/DO bit, AI/AO channel or module information number.

 **Note:** Each Standard Driver Sheet can have up to 4096 rows. However, the **Read Trigger**, **Enable Read When Idle**, and **Write Trigger** commands attempt to communicate the entire block of addresses that is configured in the sheet, so if the block of addresses is larger than the maximum block size that is supported by the driver protocol, then you will receive a communication error (e.g., "invalid block size") during run time. Therefore, the maximum block size imposes a practical limit on the number of rows in the sheet.

For examples of how device registers are specified using **Header** and **Address**, see the following table.

Examples of Header and Address fields in Standard Driver Sheet

System ID	Header	Address
US8-000250	US8	250
AR1S16-000010[15]	AR1S16.10	15

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

4. Save and close the worksheet.

Configuring the Main Driver Sheet

When you add the PAC3K driver to your project, the Main Driver Sheet is automatically included in the **PAC3K** folder in the Project Explorer. This section describes how to configure the worksheet.

The PAC3K driver must be added to the project before you can configure any of its worksheets. For more information, see [Adding a communication driver to your project](#) on page 8.

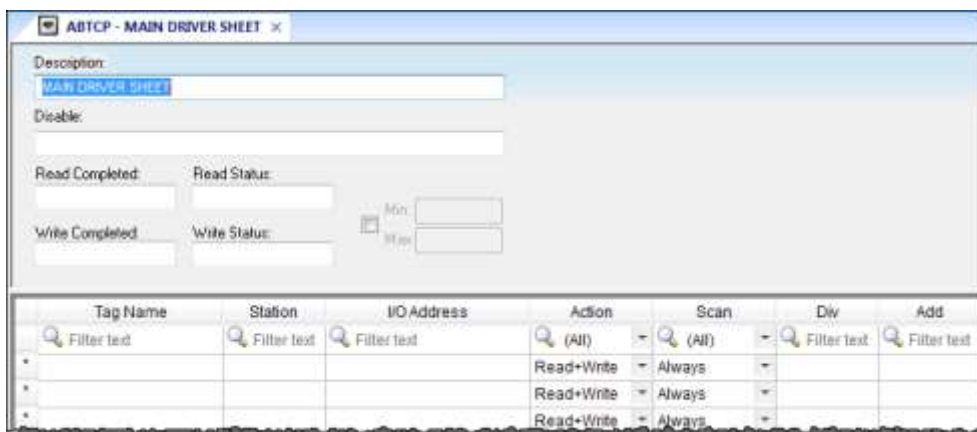
The Main Driver Sheet is used to define tag/register associations and driver parameters that are in effect at all times, regardless of project behavior. The worksheet is continuously processed during project runtime.

Note: Most of the settings on this worksheet are standard for all drivers; for more information about configuring these settings, see the “Communication” chapter of the *Technical Reference Manual*. The **Station** and **I/O Address** fields, however, use syntax that is specific to the PAC3K driver.

1. Do one of the following.

- On the **Insert** tab of the ribbon, in the **Communication** group, click **Main Driver Sheet** and then select **PAC3K** from the list.
- In the **Comm** tab of the Project Explorer, expand the **PAC3K** folder and then double-click **MAIN DRIVER SHEET**.

The Main Driver Sheet is displayed.



Main Driver Sheet

2. For each tag/register association that you want to create, insert a row in the worksheet body and then configure the row's fields as described below.

Tag Name

Type the name of the project tag.

Station

Specify the IP address and optionally the path (absolute or relative to the application's folder) to the exported tags database in CSV format of the target PAC device, separated by a semi-colon.
Example: 192.168.0.115;..\resources\P3000.csv



Note: You cannot leave the **Station** field blank.

I/O Address

C: ID

To access the Internal Bit

SBR: ID

To access the System Read only Bit

SBRW: ID

To access the System Read/Write Bit

US8: ID.Bit

To access the Unsigned 8 bits

S16: ID.Bit

To access the Signed Integer 16 bits

BCD16: ID.Bit

To access the BCD 16bits

US16: ID.Bit

To access the Unsigned Integer 16bits

SWR: ID.Bit

To access the System Read only Word

SWRW: ID.Bit

To access the System Read/Write Word

S32: ID.Bit

To access the Signed Integer 32bits

BCD32: ID.Bit

To access the BCD 32bits

F32: ID

To access the Float 32bits

STR: ID.Length

To access the String

SSTR: ID.Length

To access the System String

AR1C.ID: Index.Bit

To access the Array 1D bit

AR1US8.ID: Index.Bit

To access the Array 1D Unsigned 8bits

AR1S16.ID: Index.Bit

To access the Array 1D Signed Integer 16bits

AR1US16.ID: Index.Bit

To access the Array 1D Unsigned Integer 16bits

AR1BCD16.ID: Index.Bit

To access the Array 1D BCD 16bits

AR1S32.ID: Index.Bit

To access the Array 1D Signed Integer 32bits

AR1BCD32.ID:Index.Bit

To access the Array 1D BCD 32bits

AR1F32.ID:Index

To access the Array 1D Float 32bits

AR1STR.ID:Index.Length

To access Array 1D String

AR2C.ID:Index1.Index2

To access Array 2D Bit

AR2F32.ID:Index1.Index2

To access Array 2D Float 32bits

AR2US8.ID:Index1.Index2.Bit

To access Array 2D Unsigned 8bits

AR2S16.ID:Index1.Index2.Bit

To access Array 2D Signed Integer 16bits

AR2US16.ID:Index1.Index2.Bit

To access Array 2D Unsigned Integer 16bits

AR2BCD16.ID:Index1.Index2.Bit

To access Array 2D BCD16bits

AR2S32.ID:Index1.Index2.Bit

To access Array 2D Signed Integer 32bits

AR2BCD32.ID:Index1.Index2.Bit

To access Array 2D BCD 32bits

AR2STR.ID:Index1.Index2.Length

To access Array 2D String

DI:Group.Base.Slot.Bit

To access Discrete In

DO:Group.Base.Slot.Bit

To access Discrete Out

MST:Group.Base.Slot.Module

To access Module Status Bit

AIS32:Group.Base.Slot.Channel

To access Analog In Signed 32bits

AOS32:Group.Base.Slot.Channel

To access Analog Out Signed 32bits

AIF32:Group.Base.Slot.Channel

To access Analog In Float 32bits

AOF32:Group.Base.Slot.Channel

To access Analog Out Float 32bits

TAG:TagName

To access device addresses using tag names

...where...

ID

The ID portion of the specific System ID to be read.

Index

The index position of the specific element of the System ID array to be read.

Index1

The first dimension index of the specific element of the System ID array to be read.

Index2

The second dimension index of the specific element of the System ID array to be read.

Length

The length of the string to be read.

Bit

Read a specific bit of the System ID.

This is an optional parameter. If no bit is specified, then the entire register will be read.

Group

The group number (00 = Local Group, 01 – 99 = Remote system).

Base

The base number.

Slot

The slot number.

Bit


The DI/DO bit number.

Channel

The AI/AO channel number.

Module

The module information number.

 **Note:** The Main Driver Sheet can have up to 32767 rows. If you need to configure more than 32767 communication addresses, then either configure additional Standard Driver Sheets or create additional instances of the driver.

3. Save and close the worksheet.

Additional notes

Additional notes about the PAC3K driver.

Block sizes

Block sizes depend on the register type. The protocol accepts messages of up to 240 bytes for read block commands and of up to 200 bytes or 25 blocks for read multiple blocks. For most of the headers this will give a fixed amount of registers, but for strings it is calculated on demand, based on the length of the strings. The following table shows the block size for each header.

Header	Maximum Block Size
C, SBR, SBRW, AR1C	1920
US8, AR1US8	240
S16, BCD16, US16, SWR, SWRW, AR1S16, AR1US16, AR1BCD16	120
S32, BCD32, F32, AR1S32, AR1BCD32, AR1F32	60
AR2C, AR2US8, AR2S16, AR2US16, AR2BCD16, AR2S32, AR2BCD32, AR2F32, DI, DO, MST, AIS32, AOS32, AIF32, AOF32	25
STR, SSTR, AR1STR, AR2STR	Depends on string lengths configured on the PLC.

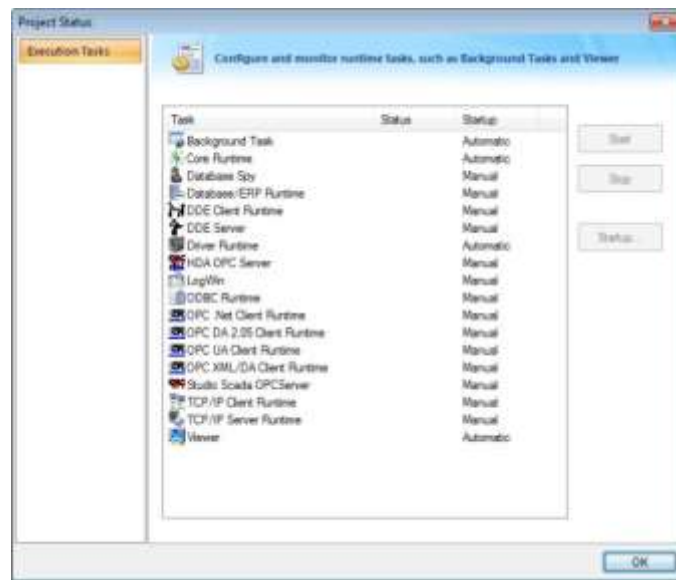
Checking the Driver Runtime task

This section describes how to check the status of the Driver Runtime task in the list of execution tasks.

The Driver Runtime task handles communication with remote devices and the processing of the driver worksheets. By default, the task is configured to start up automatically when the project is run, but you can check it for yourself.

1. On the **Home** tab of the ribbon, in either the **Local Management** or the **Remote Management** group (depending on where you project server will be running), click **Tasks**.

The *Project Status* dialog is displayed.



Project Status dialog

2. Verify that the **Driver Runtime** task is set to **Automatic**.
 - If the setting is correct, then proceed to the next step.
 - If the **Driver Runtime** task is set to **Manual**, select the task and then click **Startup** to change the task to **Automatic**.
3. Click **OK** to close the *Project Status* dialog.

Troubleshooting

This section lists the most common errors for this driver, their probable causes, and basic procedures to resolve them.

Checking status codes


If the PAC3K driver fails to communicate with the target device, then the database tag(s) that you configured for the **Read Status** and **Write Status** fields of the driver sheets will receive a status code. Use this status code and the following tables to identify what kind of failure occurred and how it might be resolved.

Status codes for the driver

Status Code	Description	Possible Causes	Procedure To Solve
0	OK	Communication without problems	None required
1	Error sending buffer	<ul style="list-style-type: none"> If using TCP/IP, the connection with the slave device might have been reset 	<ul style="list-style-type: none"> Check the switches and network connection for failures
2	Invalid station on slave response	The response sent by the slave does not match the request made by the driver	<ul style="list-style-type: none"> Check for multiple masters on the network. Check the time out configuration settings (see section "Configuring the communication settings") and increase the time out.
3	Invalid CheckSum	<ul style="list-style-type: none"> Electrical interference on your connection 	<ul style="list-style-type: none"> Check the physical connections on your network and verify if it is within the recommended physical layer specifications.
4	Invalid sequence number	The time out configured in the driver settings is too short	Check the time out configuration settings (see section "Configuring the communication settings") and increase the time out.
7	Invalid function code on slave response	The slave replied with a function code that does not match the request	<ul style="list-style-type: none"> Check for multiple masters on the network. Check the time out configuration settings (see section "Configuring the communication settings") and increase the time out.
8	Slave returned exception code	The slave could not comply with the driver request	Enable the protocol analyzer to see more details and the exception code returned, the message will help you to identify wrong settings (e.g.: unsupported address range) or list possible errors in the PLC
9	Invalid tag name	The driver could not find a tag name in the CSV file	Check if the tags being used in the application match the CSV file
10	Bit write not supported	The driver does not support writing to bits	Use Studio functionalities to change the bit and write the entire element
11	Invalid Max Gap.	MaxGap value should be in the range from 1 to 999999	Set MaxGap with value in the range 1 to 999999
101	Illegal function code	Function code used by driver is not supported by device	Check with device manufacturer to support the function code
102	Illegal data address	Data address that is accessed by driver is non-existent	Change the address in the driver sheet to address, which is present in the device
103	Illegal data value	Data value that is read from the address which is non-existent	Change the address in the driver sheet to address, which is present in the device
104	Slave device failure	Unrecoverable error occurred in the device, performing the requested action	Device should be restarted, make sure that the requested action is valid.
105	Acknowledge	Device taking more time to process request	Device will respond with appropriate response once processing is complete

Status Code	Description	Possible Causes	Procedure To Solve
106	Slave device busy	Device is busy processing other request/ command	Request for read or write once the device is free
108	Memory parity error	Parity error detected in device while reading the address	Device requires a service
110	Gateway path unavailable	Gateway is misconfigured or overloaded	Check that gateway that is used for internal communication from input port to output port is properly configured
111	Gateway target device failed to respond	Device is not present in the network	Check if the device is present

Common status codes

Status Code	Description	Possible Causes	Procedure To Solve
0	OK	Communicating without error.	None required.
-15	Timeout waiting for message to start	<ul style="list-style-type: none"> Disconnected cables. PLC is turned off, in stop mode, or in error mode. Wrong station number. Wrong parity (for serial communication). Wrong RTS/CTS configuration (for serial communication). 	<ul style="list-style-type: none"> Check cable wiring. Check the PLC mode — it must be RUN. Check the station number. Increase the timeout in the driver's advanced settings. Check the RTS/CTS configuration (for serial communication).
-33	Invalid driver configuration file	The driver configuration file (<i>drivername.INI</i>) is missing or corrupt.	Reinstall the driver.
-34	Invalid address	The specified address is invalid or out of range.	Check the supported range of addresses described in this document, and then correct the address.
-35	Driver API not initialized	The driver library was not initialized by the driver.	Contact technical support.
-36	Invalid data type	The specified data type is invalid or out of range.	Check the supported data types described in this document, and then correct the data type.
-37	Invalid header	The specified header in the driver worksheet is invalid or out of range.	Check the supported range of headers described in this document, and then correct the header.
-38	Invalid station	The specified station in the driver worksheet is invalid or out of range.	Check the supported station formats and parameters described in this document, and then correct the station.
-39	Invalid block size	Worksheet is configured with a range of addresses greater than the maximum block size.	<p>Check the maximum block size number of registers described in this document, and then configure your driver worksheet to stay within that limit. Keep in mind that you can create additional worksheets.</p> <p> Note: If you receive this error from a Main Driver Sheet or Tag Integration configuration, please contact Technical Support.</p>

Status Code	Description	Possible Causes	Procedure To Solve
-40	Invalid bit write	Writing to a bit using the attempted action is not supported.	<ul style="list-style-type: none"> Writing to a bit using Write Trigger is not supported in some drivers. Modify the driver worksheet to use Write On Tag Change. The bit is read-only.
-42	Invalid bit number	The bit number specified in the address is invalid. The limit for the bit number depends on the registry type.	Check the addresses to see if there are bit numbers configured outside the valid range for the registry.
-43	Invalid byte number	The byte number specified in the address is invalid. The limit for the byte number depends on the registry type.	Check the addresses to see if there are byte numbers configured outside the valid range for the registry.
-44	Invalid byte write	Writing to a byte using the attempted action is not supported.	The byte is read-only or inaccessible.
-45	Invalid string size	The string is more than 1024 characters.	Modify the addresses that have string data type to be less than 1024 characters.
-56	Invalid connection handle	The connection is no longer valid.	Please contact Technical Support.
-57	Message could not be sent	The socket was unable to send the TCP or UDP message.	<ul style="list-style-type: none"> Check the station IP address and port number. Confirm that the device is active and accessible. Try to ping the address.
-58	TCP/IP could not send all bytes	The TCP/IP stack was not able to send all bytes to destination.	<ul style="list-style-type: none"> Check the station IP address, port number and/or ID number. Confirm that the device is active and accessible. Try to ping the address.
-60	Error to establish TCP/IP connection	Error while establishing a TCP/IP connection with the slave device. Possibly incorrect IP address or port number in the specified station.	<ul style="list-style-type: none"> Check the station IP address, port number and/or ID number. Confirm that the device is active and accessible. Try to ping the address.
-61	TCP/IP socket error	The TCP/IP connection has been closed by the device.	Confirm that the device is active and accessible. Try to ping the address.
-62	UDP/IP receive call returned error	The UDP socket is in error.	<ul style="list-style-type: none"> Check the station IP address, port number and/or ID number. Confirm that the device is active and accessible. Try to ping the address.
-63	UDP/IP error initializing	The UDP socket initialization failed.	Confirm that the operating system supports UDP sockets.
-64	UDP/IP receive call returned error	The UDP socket is in error.	<ul style="list-style-type: none"> Check the station IP address, port number and/or ID number. Confirm that the device is active and accessible. Try to ping the address.

-65	UDP/IP bind error, port number may already be in use	The driver was not able to bind the UDP port.	<ul style="list-style-type: none">• Check the port number used by the driver.• Check for other programs that might be bound to the UDP port.
-----	--	---	---

Monitoring device communications

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 to establish an event log on a remote unit that runs Windows Embedded. 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.

Contacting Technical Support

If you must contact Technical Support, please have the following information ready:

- **Operating System** and **Project Information**: To find this information, click **Support** in the **Help** tab of the ribbon.
- **Driver Version** and **Communication Log**: Displays in the *Output* window (LogWin module) when the driver is enabled and the project is running is running.
- **Device Model** and **Boards**: Consult the hardware manufacturer's documentation for this information.

Revision history

This section provides a log of all changes made to the driver.

Revision history

Driver Version	Revision Date	Description of Changes	Author
1.0	31 May 2013	<ul style="list-style-type: none">Initial release of driver.	André Körbes
1.1	12 Sep 2013	<ul style="list-style-type: none">Updated status codes of read and write operations.Fixed lock on writing.	Charan Manjunath P
1.2	27 Jan 2014	<ul style="list-style-type: none">Updated with MaxGap parameter in driver settings.Updated with status codes for invalid entry for MaxGap value.	Charan Manjunath P
1.3	16 Oct 2014	<ul style="list-style-type: none">Fixed invalid block size error when reading analog datatypes.	Felipe Andrade
1.4	11 Nov 2015	<ul style="list-style-type: none">Fixed invalid block size error when reading string tags.	Anushree Phanse