









**Communication Driver TI500**

Driver for serial communication with Texas Instruments - Series 500 using TBP or NITP protocols.


**Index**

-  **INTRODUCTION..... 2**
-  **GENERAL CHARACTERISTICS ..... 3**
  - 2.1 DEVICE CHARACTERISTICS ..... 3
  - 2.2 LINK CHARACTERISTICS ..... 3
  - 2.3 DRIVER CHARACTERISTICS ..... 3
  - 2.4 INFORMATION ABOUT THE CONFORMANCE TESTS ..... 5
-  **INSTALLATION ..... 7**
  - 3.1 INSTALLING THE DRIVER ..... 7
  - 3.2 OTHER SOFTWARE REQUIREMENTS ..... 7
-  **DRIVER CONFIGURATION ..... 8**
  - 4.1 SETTINGS - COMMUNICATION PARAMETERS ..... 8
  - 4.2 DRIVER WORKSHEET ..... 15
  - 4.3 STATION AND HEADER CONFIGURATION ..... 16
  - 4.4 ADDRESS CONFIGURATION ..... 17
  - 4.5 MAIN DRIVER SHEET (MDS) ..... 19
  - 4.6 DEVICE CONFIGURATION ..... 20
-  **EXECUTION ..... 22**
-  **TROUBLESHOOTING ..... 23**
-  **APPLICATION SAMPLE ..... 29**
-  **HISTORY OF VERSIONS ..... 30**

## Introduction

The TI500 driver enables communication between Studio system and some of the Texas Instruments devices using TDB, NITP and protocols, in accordance with the characteristics covered in this document.

This document contains 8 parts, as follows:

- **Introduction:** Provides an overview of the driver documentation.
  - **General characteristics:** Provides information necessary to identify all the required components (hardware and software) necessary to implement the communication and global characteristics about the communication.
  - **Installation:** Explains the procedures that must be followed to install the software and hardware required for the communication.
  - **Driver configuration:** Provides the required information to configure the communication driver such as the different permutations for configuration and its default values.
  - **Execution:** Explain the steps to test whether the driver was correctly installed and configured.
  - **Troubleshooting:** Supplies a list of the most common error codes for this protocol and the procedures to fix them.
  - **Application Sample:** Provides a sample application for testing the configuration the driver.
  - **History of versions:** Provides a log of all the modifications done in driver.
-  **Note:** This document presumes that the user has read the chapter *Driver Configuration* of the Studio's Technical reference manual.

## General Characteristics

### 2.1 Device Characteristics

- **Manufacturer:** Siemens (Texas Instruments)
- **Compatible Equipment:**
  - TI545
  - TI565
  - TI575
  - CTI 2500


 **Tip:** Refers to section 2.4 to see the Equipment used in the standard conformance tests for this driver.

### 2.2 Link Characteristics

- **Device communication port:** RS 232/Ethernet
- **Physical protocol:** RS232/Ethernet
- **Logic protocol:** CAMP (TCP, UDP), TBP (Serial), NITP(Serial)
- **Device Runtime software:** None
- **Specific PC Board:** None
- **Cable:** See manufacturer documentation.


### 2.3 Driver Characteristics

- **Operating System:**
  - Windows 7/8/10
  - Windows CE

 **Tip:** Please refer to section 2.4 to see the Operating System used in the conformance tests for this driver.

The driver is composed of the following files:

- **TI500.INI:** Internal file of the driver, it should not be modified by the user.
- **TI500.MSG:** Error messages for each error code. It should not be modified.
- **TI500.PDF:** Provides detailed documentation about the driver.
- **TI500.DLL:** Compiled driver.

 **Note:** All the files above must to be in the subdirectory /DRV of the Studio's installation directory.

▪ **Supported Registers:**

Register Type	Length	Write	Read	Bit	Integer	Float
V (Variable Memory)	2 Bytes	•	•	•	•	•
K (Constant Memory)	2 Bytes	•	•	•	•	•
TCP (Timer/Count Preset Memory)	2 Bytes	•	•	•	•	–
TCC (Timer/Count Count Memory)	2 Bytes	•	•	•	•	–
WX (Word Input Memory)	2 Bytes	–	•	•	•	•
WY (Word Output Memory)	2 Bytes	•	•	•	•	•
X (Discrete Input)	1 Bit	–	•	•	–	–
Y (Discrete Output)	1 Bit	•	•	•	–	–
C (Control relay)	1 Bit	•	•	•	–	–
STW (System Status)	2 Bytes	–	•	•	•	•
DSP (Drum Set Preset)	2 Bytes	•	•	•	•	•
DSC (Drum Set Current)	2 Bytes	•	•	•	•	•
DCC (DCC)	2 Bytes	•	•	•	•	•
DCP (Drum Count Preset)	2 Bytes	•	•	•	•	•

↳ **Tip:** The parameter **Write word bits (0=No/1=Yes)** from the **Communication Parameters** must be set to **1** when enable writing bits in word registers (V, K, TCP, TCC, WX and WY).

The following registers are implemented in the driver which are PID Control and Analog Alarm Data Elements. Note to use these registers they must be configured in the PLC using the PID Loop Directory and Analog Alarm Directory menus.

Register Type	Description
LPV	Loop Process Variable
LSP	Loop Setpoint Variable
LMN	Loop Output
LMX	Loop Bias
LERR	Loop Error
LKC	Loop Gain
LTD	Loop Rate
LTI	Loop Reset
LVF	Loop V-flags
LRSF	RAMP/SOAP flags
APV	Analog Alarm Process Variable
ASP	Analog Alarm Setpoint
AVF	Analog Alarm flags
LPVL	Loop Process Variable Low Limit
LPVH	Loop Process Variable High Limit
APVL	Analog Alarm Process Variable Low Limit
APVH	Analog Alarm Process Variable High Limit
LTS	Loop Sample Rate (seconds)
ATS	Analog Alarm Sample Rate (seconds)
LHA	Loop High Alarm Limit
LLA	Loop Low Alarm Limit
LODA	Loop Orange Deviation Alarm Limit
LYDA	Loop Yellow Deviation Alarm Limit
LSPL	Loop Setpoint Low Limit
LSPH	Loop Setpoint High Limit
LCFH	Most-significant word of Loop C-flags

LCFL	Least-significant word of Loop C-flags
LHHA	Loop High-High Alarm Limit
LLLA	Loop Low-Low Alarm Limit
LRCA	Loop Rate-of-Change Alarm Limit (Engineering units/minute)
LADB	Loop Alarm Deadband
AHA	Analog Alarm High Limit
ALA	Analog Alarm Low Alarm Limit
AODA	Analog Alarm Orange Deviation Alarm Limit
AYDA	Analog Alarm Yellow Deviation Alarm Limit
ASPL	Analog Alarm Setpoint Low Limit
ASPH	Analog Alarm Setpoint High Limit
ACFH	Most-significant word of Analog Alarm C-flags
ACFL	Least-significant word of Analog Alarm C-flags
AHHA	Analog Alarm High-High Alarm Limit
ALLA	Analog Alarm Low –Low Alarm Limit
ARCA	Analog Alarm Rate-of-Change Alarm Limit (engineering units/minute)
AADB	Analog Alarm Deadband
AERR	Analog Alarm Error
LDK	Loop Derivative Gain-limiting coefficient
LRSN	Loop RAMP/SOAK Step Number
LACK	Loop Alarm / Alarm Acknowledge flags
AACK	Analog Alarm / Alarm Acknowledge flags
LPET	Loop Peak Elapsed Time Value - Represents the elapsed time from when the process is scheduled until it completes execution (TI545, TI555, TI575)
APET	Analog Alarm Peak Elapsed Time Value - Represents the elapsed time from when the process is scheduled until it completes execution (TI545 , TI555 , TI575)
PPET	SF PGM Peak Elapsed Time Value - Represents the elapsed time from when the process is scheduled until it completes execution (TI545 , TI555 , TI575)

## 2.4 Information about the Conformance Tests

The following hardware/software was used for conformance testing:

### Driver Configuration:

1. Protocol: TBP (Serial)
  - Block Size: 7
  - Serial Encapsulation mode: None
  - Databits : 8
  - Parity : None
2. Protocol: NITP (Serial)
  - Block Size: 7
  - Serial Encapsulation mode: None
  - Databits : 7
  - Parity : Odd
3. Protocol: NITP
  - Block Size: 7
  - Serial Encapsulation mode: TCP/IP
4. Protocol : TCP (using CAMP)
  - Serial Encapsulation mode: None
5. Protocol : UDP (using CAMP)
  - Serial Encapsulation mode: None

Driver Version	Studio Version	Operating System	Equipment
1.20	8.1 + SP1	Windows XP + SP3 Windows CE 5.0 and 6.0 Windows 8.1	CTI2500-C100 CTI 2572-B module for UDP

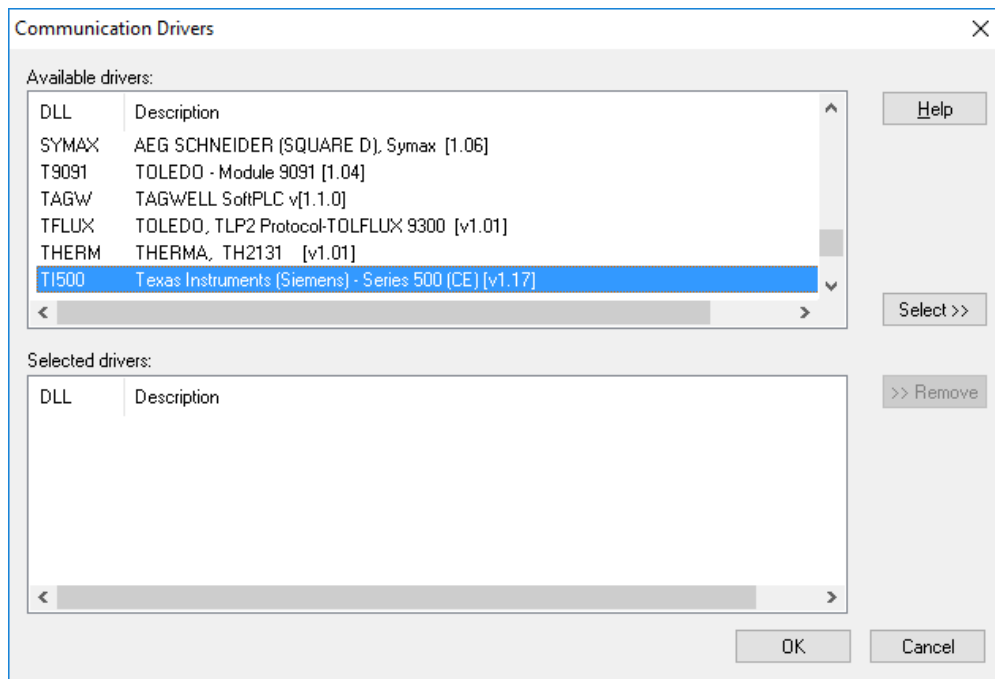
## Installation

### 3.1 Installing the Driver

When you install the Studio v3.0 or higher, the communication drivers are already installed. You need now to select the driver at the applications where it will be used.

The steps to select the driver inside an application are:

1. Execute the Studio and select the proper application.
2. Select the menu *Insert + Driver...*
3. In the column **Available Drivers**, select the **TI500 Driver** and push the button **Select>>** (the driver TI500 must appear in the column **Selected Drivers**).
4. Press **OK**.



### 3.2 Other software requirements

It is not necessary to install any other software in the PC to enable the communication between the Studio and the Device.

**⚠ Attention:** Special precautions must be taken when installing the physical hardware. Refer to the hardware manufacturer documentation for specific instructions in this area.

## Driver Configuration

After the driver is installed and selected in the Studio (see section 3.1), you should proceed to the driver configuration.

The driver configuration is two parts:

- The Settings or Communication parameters, it is only one configuration to the whole driver;
- The communication tables or Driver Worksheets, where the communication tags are defined. There are two types of communication tables: **Standard Tables** and **MAIN DRIVER SHEET**.

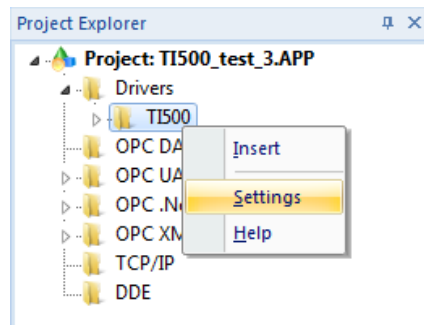
### 4.1 Settings - Communication Parameters

This driver supports three distinct protocols: CAMP, TBP and NITP. **CAMP is the preferred protocol for Ethernet communication**, while NITP and TBP are used for serial communication. There are basically three possible ways to configure the communication parameters:

- **Ethernet communication:** choose this option if the target device is connected through a TCP/IP or UDP/IP network and if it supports CAMP protocol.
- **Direct serial communication:** choose this option if the target device is directly connected to the computer through the serial port. For serial communication the available protocols are NIPT or TBP.
- **Serial encapsulation:** choose this option if the target device is connected through a TCP/IP or UDP/IP network and if it does not support CAMP protocol. In this case, the serial NITP protocol can be used if the serial communication is encapsulated in TCP or UDP.

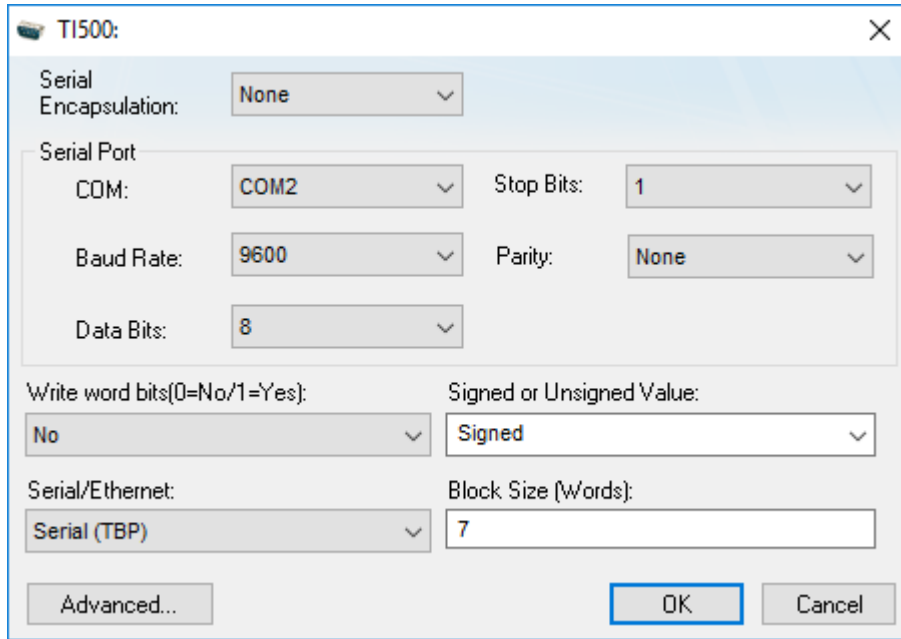
These parameters are valid for all driver worksheets configured in the system. To open the window for configuring the **Communication parameters**, follow these steps:

- 📁 In the **Workspace** of the Studio environment, select the **Comm** tab.
- 📁 Expand the folder **Drivers** and select the subfolder **TI500**.
- 📁 Right click on the **TI500** subfolder and select the option **Settings**.



On selecting the Settings, see dialog below for configuration:

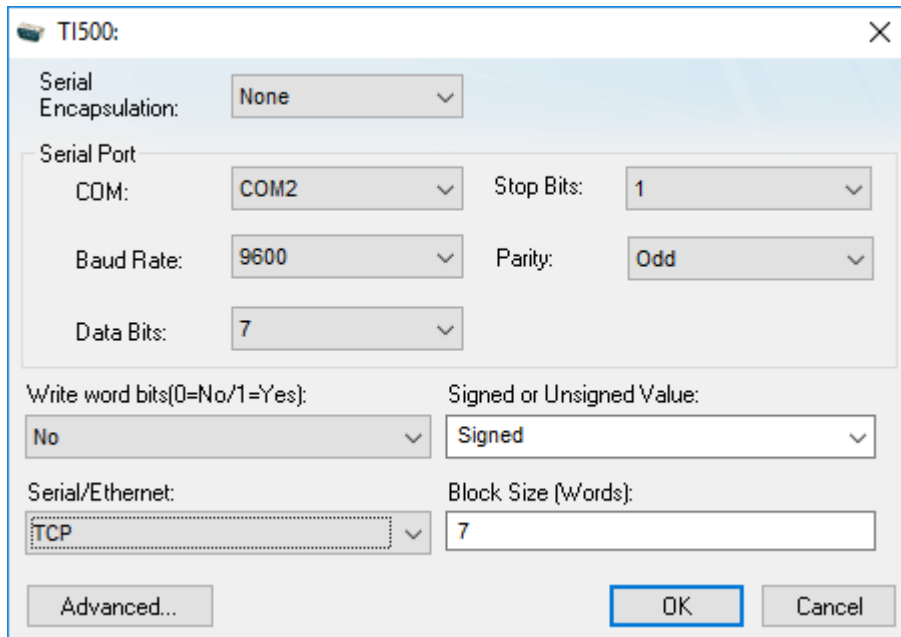




The parameters are configured differently depending on the choice for Ethernet communication, serial communication or serial encapsulation:

### 1. Ethernet communication

If the target device is connected through a TCP/IP or UDP/IP network and if it supports CAMP protocol, configure the following parameter:



a. Serial Encapsulation

In the Serial Encapsulation box, select the encapsulation mode “None”. It means Serial Encapsulation will be disabled. Leave all the fields inside the “Serial Port” box unchanged. They are not used in Ethernet communication.

b. Write word bits

An option to allow or disallow writing to individual bits of a Word. Select one of the following:

Option	Description
No	Do not allow writing to individual bits.
Yes	Allow writing to individual bits.

c. Serial/Ethernet

The specific transport protocol to be used. Select one of the following:

Option	Description
TCP	Common Ascii Messaging Protocol (CAMP) using TCP
UDP	Common Ascii Messaging Protocol (CAMP) using UDP

The other two options (NITP and TBP) should not be used for Ethernet communication.

d. Signed or Unsigned Value

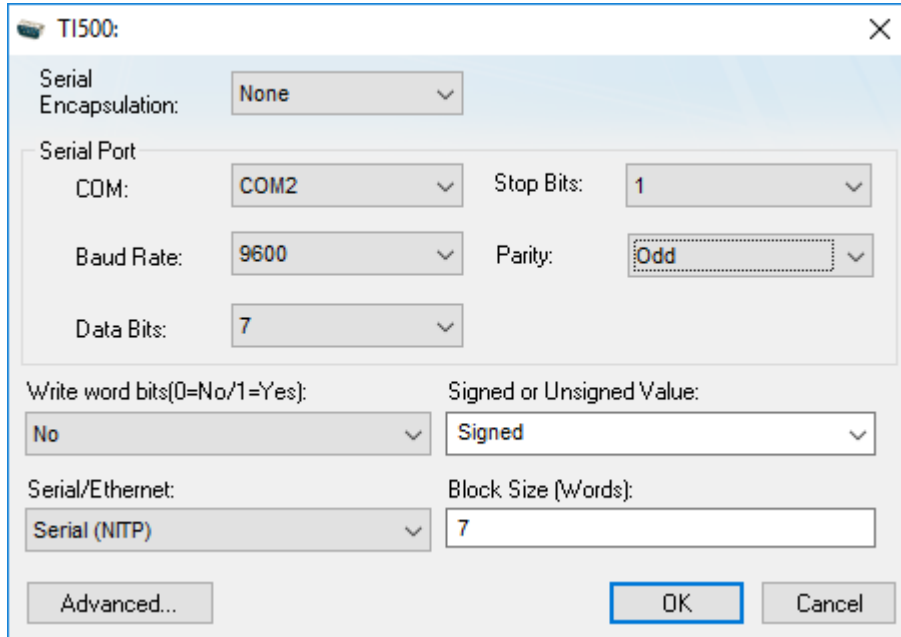
Whether register values are unsigned or signed by default. Select either Signed or Unsigned.

e. Max Block Size

Leave the default value as 7. This field is not used by CAMP protocol.

**2. Serial communication**

If the target device is connected through a serial connection, configure the following parameters:



a. Serial Encapsulation

In the Serial Encapsulation box, select the encapsulation mode “None”. It means Serial Encapsulation will be disabled. In the “Serial Port” box, configure the direct serial communication settings:

- In the COM box, select the COM port to which the target device is connected.
- In the Baud Rate, Data Bits, Stop Bits, and Parity boxes, configure the serial communication settings to match the settings that have already been configured on the target device.

b. Write word bits

An option to allow or disallow writing to individual bits of a Word. Select one of the following:

Option	Description
No	Do not allow writing to individual bits.
Yes	Allow writing to individual bits.

c. Serial/Ethernet

The specific protocol to be used. Select one of the following:

Option	Description
Serial (TBP)	Transparent Byte Protocol
Serial (NITP)	Non-Intelligent Terminal Protocol

The other two options (TCP and UDP) should not be used for serial communication.

d. Signed or Unsigned Value

Whether register values are unsigned or signed by default. Select either Signed or Unsigned.

e. Max Block Size

The maximum number of bytes that will be requested from the device in a single read operation. You can type any value from 1 to 15, but the default value is 7.

### 3. Serial encapsulation

If the target device is connected through a TCP/IP or UDP/IP network and if it does not support CAMP protocol, configure the following parameters:

a. Serial Encapsulation

In the Serial Encapsulation box, select one of these options:

Option	Description
TCP/IP	Serial encapsulation over a TCP/IP network link.
UDP/IP	Serial encapsulation over a UPD/IP network link.

In the “Serial Port” box, configure the serial encapsulation settings:

- In the IP Address box, type the IP address of the target device.
- In the Port Number box, type the port number of the target device. In most of cases it should be 1505 or 4505.
- Leave “Server Mode” box unchecked.

b. Write word bits

An option to allow or disallow writing to individual bits of a Word. Select one of the following:

Option	Description
No	Do not allow writing to individual bits.
Yes	Allow writing to individual bits.

c. Serial/Ethernet

The specific protocol to be used. Select one of the following:

Option	Description
Serial (TBP)	Transparent Byte Protocol
Serial (NITP)	Non-Intelligent Terminal Protocol

The other two options (TCP and UDP) should not be used for serial encapsulation.

d. Signed or Unsigned Value

Whether register values are unsigned or signed by default. Select either Signed or Unsigned.

e. Max Block Size

The maximum number of bytes that will be requested from the device in a single read operation. You can type any value from 1 to 15, but the default value is 7.

## Advanced Settings

Click the “Advanced” button to open the Advanced Settings dialog box, which provides access to additional communication settings such as timeouts, retries, and buffer sizes. You might need to change these settings if the driver behaves unexpectedly during run time, but the default settings should work for most network configurations. For more information about these settings, see "Advanced Settings"

The screenshot shows the 'Advanced settings' dialog box with the following fields and options:




- Timeout (ms):** Start message: 1000, End message: 0, Interval between char: 500, Wait CTS: 100
- Handshake:** Control RTS: no, Verify CTS: no
- Simultaneous Requests:** Maximum: 1, Maximum per station: 1
- Protocol:**  Disable DTR,  Enable IR, Station: [empty], Retries: 0
- Buffers length (bytes):** Tx Buffer: 512, Rx Buffer: 512
- Buttons:** OK, Cancel

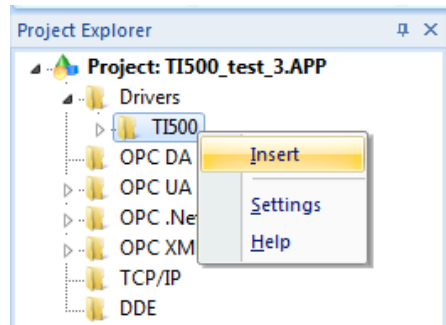
The Advanced setting parameters are explained at the Studio Technical Reference Manual, and you should keep the default values to all fields.


**Tip:** Usually, these parameters must be changed when using a DCE (Data Communication Equipment) - converter (232/485, for example), modem, etc - between the PC, driver and the host. It is necessary to know the characteristics of the DCE before adjusting these parameters.

## 4.2 Driver Worksheet

It is possible to configure many driver worksheets; each one will be composed of a Header and Body. To create a new driver worksheet, follow these steps:

-  In the **Workspace** of the Studio environment, select the table **Comm**.
-  Expand the folder **Drivers** and select the subfolder **TI500**.
-  Right click on the **TI500** subfolder and select the option **I**nsert.



 **Tip:** To optimize communication and ensure better performance for the system, it is important to tie the tags in different driver sheets according to the events that must trigger the communication of each group of tags and the periodicity for which each group of tags must be written or read. In addition, it is recommended to configure the addresses of communication in sequential blocks.

The screenshot shows the 'TI500001.DRV' configuration window. It has several input fields and a table at the bottom.

**Description:** V3405 - V3411  Increase priority

**Read Trigger:** RT[1] **Enable Read when Idle:**  **Read Completed:**  **Read Status:**

**Write Trigger:**  **Enable Write on Tag Change:**  **Write Completed:**  **Write Status:**

**Station:**  **Header:** V  Min:   Max:

	Tag Name	Address	Div	Add
1	Test3405	V:3405		
2	Test3406	V:3406		
3	Test3407	V:3407		
4	Test3408	V:3408		
5	Test3409	V:3409		
6	Test3410	V:3410		
7	Test3411	V:3411		

All entries at the Driver Worksheet, exception by the **Station**, **Header** and **Address** are standard to all communication drivers. You should refer to Studio Communication Driver documentation about the configuration of the standard fields. This document describes the Station, Header and Address fields, which are specific to each communication driver.

### 4.3 Station and Header configuration

Parameter	Default Value	Valid values	Description
Station	-	<p>Case 0: None for <b>Serial</b> Communication</p> <p>Case 1: For <i>Serial encapsulation: TCP/IP or UDP/IP</i> when <i>Serial/ Ethernet</i> setting is set to <i>Serial(NITP)</i> <b>&lt;IP Address&gt;:&lt;Port Number&gt; </b> (e.g.: 192.168.0.5:1505 )</p> <p>Case 2: For <i>Serial/Ethernet</i> field set as <i>TCP</i> or <i>UDP</i></p>	<p>Case 0: Station field is only used if you are using <b>Ethernet communications</b>.</p> <p>Case 1: - If the IP address is different than what is configured on the <b>Communication Settings</b>. In that case, you have to follow the below syntax:  <b>&lt;IP Address&gt;:&lt;Port Number&gt; </b> (e.g.: 192.168.0.5:1505 )</p> <p>Please notice the <b>pipe</b> character “   ” at the end, after the port number. You must place the <b>pipe</b> character otherwise the communication will fail. More details about the Serial Encapsulation over TCP/IP can be found on the Studio Technical Reference manual</p> <p>- Else if the station is the same as what is configured on the Communication settings the user can leave the station field on the driver sheet blank.</p> <p>Case 2: <b>The station cannot be set in the Communication Settings</b>. The station has to be set in the driver sheets with the syntax : &lt;IP Address&gt;:&lt;Port Number&gt; Example : 192.168.0.5:1505 In this case there is <b>no</b> ‘ ’ bar in the station.</p>
Header	-	V, K, TCP, TCC, WX, WY, X, Y, C, DCC,DSP, LSP, LPV, APV etc.	<p>Can be one of the following types:</p> <ul style="list-style-type: none"> <li>• V=Variable Memory ;</li> <li>• K=Constant Memory ;</li> <li>• TCP=Timer/Count Preset Memory ;</li> <li>• TCC=Timer/Count Count Memory ;</li> <li>• WX=Word Input Memory ;</li> <li>• WY=Word Output Memory ;</li> <li>• X=Discrete Input ;</li> <li>• Y=Discrete Output ;</li> <li>• C=Control relay;</li> <li>• STW=System Status</li> <li>• DCC=Drum Current Count</li> <li>• DCP=Drum Count Preset</li> <li>• DSP=Drum Set Preset</li> <li>• DSC=Drum Step Current</li> <li>• PID and Analog Alarm Data Elements (see Page 4)</li> </ul>




### 4.4 Address Configuration


The body of the driver worksheet allows you to associate each tag to its respective address in the device. In the column **Tag Name**, you must type the tag from your application database. This tag will receive or send values from or to an address on the device. The address cells complies to the following syntax:

- To digital registers (X ; Y and C):  
**<Type>:<Address>** (e.g.: X10)
- To analog registers (V, K, TPC, TCC, DCC, DSP, DSC, STW, WX and WY and PID Loop or Analog Alarm registers like APV, LSP, etc.)  
**<Type><Address>** (e.g.: V10)  
**<Type><Address><Format>** (e.g.: V10F)  
**<Type><Address>.<Bit>** (e.g.: V10.2)  
**<Type><Address><Format><String Length>** (e.g.: V10ST2, V10STS2)

where

- **Type**: Register type (V=Variable Memory ; K=Constant Memory ; TCP=Timer/Count Preset Memory ; TCC=Timer/Count Count Memory ; WX=Word Input Memory ; WY=Word Output Memory ; X=Discrete Input ; Y=Discrete Output ; C=Control relay, LSP=Loop Setpoint, DCC=Drum Count Current, APV= Alarm process variable etc.);
- **Address**: Address of the device register;
- **Format**: Data format. It can be **F** for Float or **D** for Double Word, **ST** for strings, **STS** for string swap.
- **Bit**: bit number (from 0 up to 15) from the word address. It's an optional parameter.
- **String Length** : length of the string when using ST or STS


 **Tip**: The parameter **Write word bits (0=No/1=Yes)** from the **Communication Parameters** must be set to **1** when enable writing bits in word registers (V, K, TCP, TCC, WX and WY).


 **Tip**: The suffix ST or STS for strings can be added and accessed for any of the analog registers. The String length parameter is required to be specified in the address to be able to successfully get values as strings.

- To drum register (DCP):  
**<Type><Address>.<Step>** (e.g.: DCP1.12)
  - **Type**: Register type (DCP = Drum Count Present)
  - **Address** : Address of the device register
  - **Step**: Step is a parameter of the drum count preset and it is a value between 1 to 16

Sample of Addressing Configuration	
Address on the Device	Address Field
X00001	X1
Y00100	Y100
C00010	C10
WX00020.15 (bit 15)	WX20.15
WY00030	WX30
WY00030/ WY00031 (Float)	WY30F
V100-V103 (String)	V100ST3
K100-K120 (String with swap)	K100STS20
WY00040 (Double)	WY40D
DCC00001	DCC1

DSP00001.12	DSP1.12
DSC00002 (float)	DSC2F
DCP00003.3 (step= 3)	DCP3.3
STW00001	STW1

 **Tip:** The maximum string length that can be specified depends on the register being used and the size of the message.

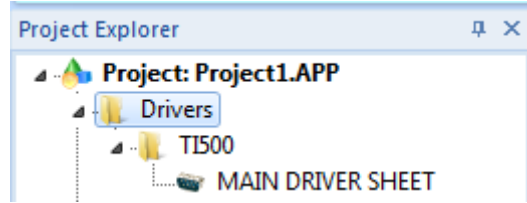
 **Tip:** To use the PID Loop and Analog Alarm Data Elements, they must be configured in the PLC by using the PID Loop Directory and the Analog Alarm Directory menus.

 **Note:**

1. String datatypes ST and STS are not supported by when using protocols TBP or NITP. This means strings cannot be used when using Direct Serial or when using UDP/IP or TCP/IP with Serial Encapsulation. They only work with using Direct Ethernet protocol CAMP (TCP and UDP)
2. Drum memory registers DCC, DSC, DCP are not supported by when using protocols TBP or NITP. This means they cannot be used when using Direct Serial or when using UDP/IP or TCP/IP with Serial Encapsulation. They only work with using Direct Ethernet protocol CAMP (TCP and UDP).

### 4.5 Main Driver Sheet (MDS)

When the driver is inserted in the application, the MAIN DRIVER SHEET is automatically added to the driver folder.



The MAIN DRIVER SHEET provides a simple way to associate Studio tags to addresses in the PLC. Most of the MAIN DRIVER SHEET entries are standard for any driver. Refer to Studio Technical Reference Manual about the configuration of the standard fields. The fields which require specific syntax for this driver are described below:

**TI500 - 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	Test3405		V3405	Read+Write	Always		
2	Test3406		V3406	Read+Write	Always		
3	Test3407		V3407	Read+Write	Always		
4	Test3408		V3408	Read+Write	Always		
5	Test3409		V3409	Read+Write	Always		
6	Test3410		V3410	Read+Write	Always		
7	Test3411		V3411	Read+Write	Always		

- **Station:** Station field syntax depends on how the communication parameters are configured:
  - **Serial communication:** “Serial/Ethernet” field is configured as “Serial(NITP)” or “Serial(TBO)” and Serial Encapsulation fields is set as “None”.  
Leave station field blank.
  - **Ethernet communication:** if “Serial/Ethernet” field is configured as TCP or UDP.  
Syntax : <IP Address>: <Port number> (for example : 198.6.3.78:1505)
  - **Serial Encapsulation:** Serial/Ethernet” field is configured as “Serial(NITP)” or “Serial(TBO)” and Serial Encapsulation fields is set as as TCP/IP or UDP/IP.  
Syntax : <IP Address>:<Port number>| (for example : 198.2.68.5:1505|)

For Serial encapsulation, the station field is optional. If station is blank, the default IP address and port configured in the communication parameters will be used.

- **I/O Address:** Address of each register from the PLC. The syntax used in this field is described below:

i. To digital registers (X ; Y and C):

**<Type>:<Address>** (e.g.: X10)

ii. To analog registers (V, K, TPC, TCC, DCC, DSP, DSC, STW, WX and WY and PID Loop or Analog Alarm registers like APV, LSP, etc.)

**<Type><Address>** (e.g.: V10)

**<Type><Address><Format>** (e.g.: V10F)

**<Type><Address>.<Bit>** (e.g.: V10.2)

**<Type><Address><Format><String Length>** (e.g.: V10ST2, V10STS2)

where


- **Type:** Register type (V=Variable Memory ; K=Constant Memory ; TCP=Timer/Count Preset Memory ; TCC=Timer/Count Count Memory ; WX=Word Input Memory ; WY=Word Output Memory ; X=Discrete Input ; Y=Discrete Output ; C=Control relay, LSP=Loop Setpoint, DCC=Drum Count Current, APV= Alarm process variable etc.);


- **Address:** Address of the device register;

- **Format:** Data format. It can be **F** for Float or **D** for Double Word, **ST** for strings, **STS** for string swap.

- **Bit:** bit number (from 0 up to 15) from the word address. It's an optional parameter.

- **String Length :** length of the string when using ST or STS

 **Tip:** The parameter **Write word bits (0=No/1=Yes)** from the **Communication Parameters** must be set to **1** when enable writing bits in word registers (V, K, TCP, TCC, WX and WY).

 **Tip:** The suffix ST or STS for strings can be added and accessed for any of the analog registers. The String length parameter is required to be specified in the address to be able to successfully get values as strings.

iii. To drum register (DCP):

**<Type><Address>.<Step>** (e.g.: DCP1.12)

- **Type:** Register type (DCP = Drum Count Present)

- **Address :** Address of the device register

- **Step:** Step is a parameter of the drum count preset and it is a value between 1 to 16

 **Note:**

1. String datatypes ST and STS are not supported by when using protocols TBP or NITP. This means strings cannot be used when using Direct Serial or when using UDP/IP or TCP/IP with Serial Encapsulation. They only work with using Direct Ethernet protocol CAMP (TCP and UDP)

2. Drum memory registers DCC, DSC, DCP are not supported by when using protocols TBP or NITP. This means they cannot be used when using Direct Serial or when using UDP/IP or TCP/IP with Serial Encapsulation. They only work with using Direct Ethernet protocol CAMP (TCP and UDP)

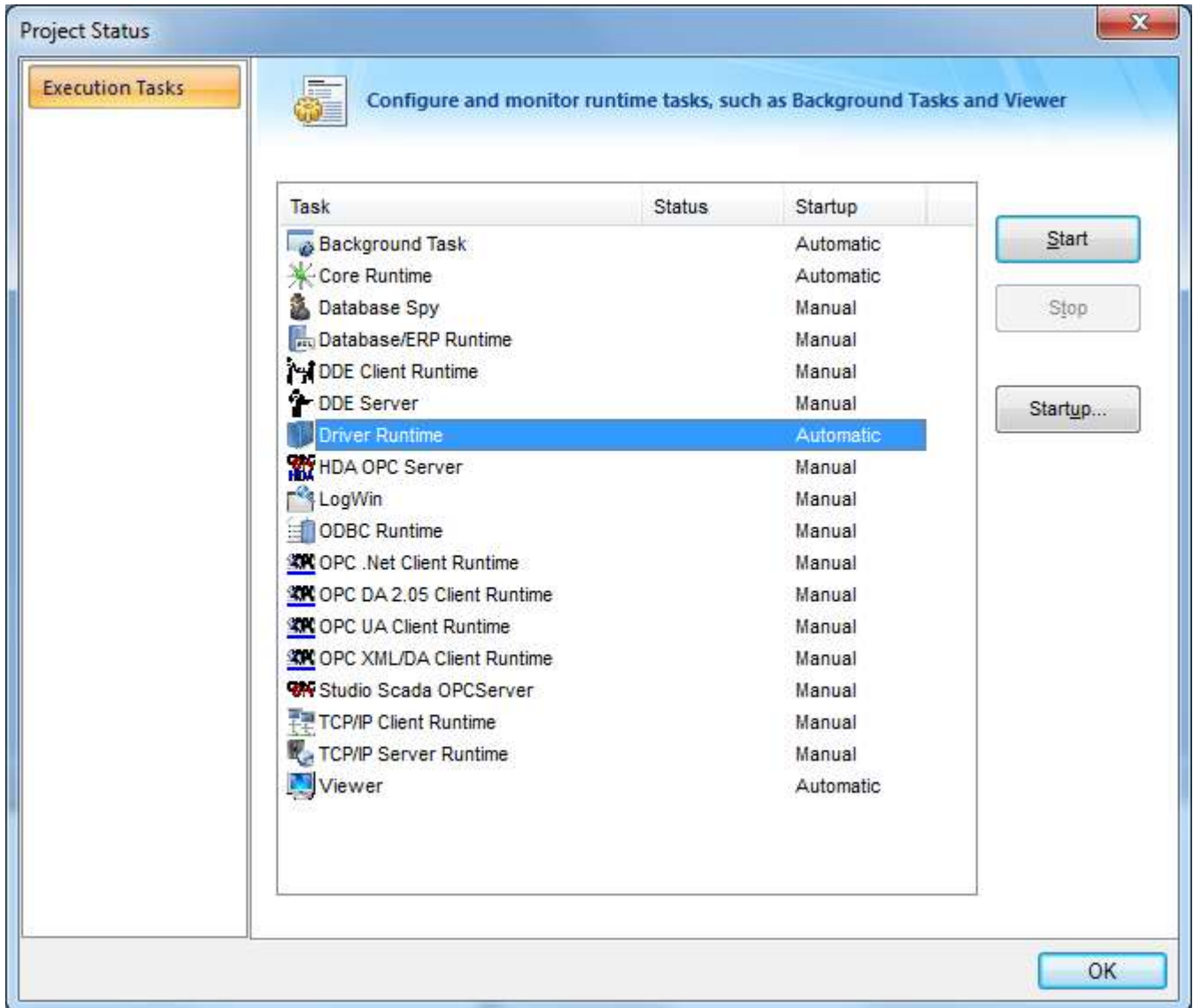
## 4.6 Device Configuration

We can choose several ways to configure the Driver Settings.

1. – Protocol: TBP (Serial)
  - Block Size: 7
  - Serial Encapsulation mode: None
  - Databits : 8
  - Parity : None
2. – Protocol: NITP (Serial)
  - Block Size: 7
  - Serial Encapsulation mode: None
  - Databits : 7
  - Parity : Odd
3. – Protocol: NITP
  - Block Size: 7
  - Serial Encapsulation mode: TCP/IP
4. – Protocol : TCP (using CAMP)
  - Serial Encapsulation mode: None
  - Block Size: 7
5. – Protocol : UDP (using CAMP)
  - Serial Encapsulation mode: None

## Execution

When installing the driver, it is automatically selected to execute when you start-up the Runtime Environment. To verify if the driver is correctly enabled to start, use the option **Home + Tasks** and verify if the Driver Runtime task startup is mode is set to **Automatic**



## Troubleshooting

After each attempt to communicate using this driver, the tag configured in the field **Read Status** or **Write Status** will receive the error code regarding the kind of failure that occurred. The error messages are:

Error Code	Description	Possible causes	Procedure to solve
0	OK	Communication without problems	
1	Protocol Error	Driver Settings does not match with the PLC Settings	Check PLC Settings and configure Driver Settings to match with it
2	Error Invalid Command	Driver Settings does not match with the PLC Settings	Check PLC Settings and configure Driver Settings to match with it
3	Invalid Response	Driver Settings does not match with the PLC Settings	Check PLC Settings and configure Driver Settings to match with it
4	Invalid Block Size	The PLC model do not accept the amount of bytes requested or sent	Change the Worksheets dividing the data in more Worksheets
5	Invalid Header	An tag with invalid type has been inserted into the Header field	Check the tag value and put on it a valid type
6	Invalid Address	An tag with invalid type has been inserted into the Address field	Check the tag value and put on it a valid type
9	Word But Write is not enabled	Attempt to write to a bit without enabling it on the Write Bit on the driver settings	Configure the <i>Write Bit on Word</i> parameter on the driver settings
15	Invalid Entry	Invalid entry or entries in the driver settings	Check Write word bits or Protocol (in the driver settings) and make sure they are each set to either 0 or 1, according to the descriptions provided in this documentation.
16	Configured block size is not supported by PLC	The configured driver sheet is requesting more registers than what is supported by the CPU model	Configure the Parameter "Block Size" on the Driver Settings window to a value supported by that CPU. The default value is 7
17	Invalid entry for block size	The specified maximum block size is invalid.	Set Max Block Size (in the channel settings) to an integer value from 1 to 64.
18	Invalid block size	The specified maximum block size is invalid.	Set Max Block Size( in the channel settings) to an integer value from 1 to 64.
-15	Timeout waiting start a message.	<ul style="list-style-type: none"> <li>- Disconnected cables</li> <li>- PLC turned off, or in Stop or error mode</li> <li>- Wrong Station number</li> <li>- Wrong RTS/CTS control settings.</li> </ul>	<ul style="list-style-type: none"> <li>- Check the cable wiring</li> <li>- Check the PLC state. It must be RUN</li> <li>- Check the station number.</li> <li>- Check the right configuration. See on the section 2.2 the different RTS/CTS valid configurations.</li> </ul>
-17	Timeout between rx char.	<ul style="list-style-type: none"> <li>- PLC in stop or error mode</li> <li>- Wrong station number</li> <li>- Wrong parity</li> </ul>	<ul style="list-style-type: none"> <li>- Check the cable wiring</li> <li>- Check the PLC state. It must be RUN</li> <li>- Check the station number.</li> </ul>

		- Wrong RTS/CTS configuration settings	- Check the right configuration. See on the section 2.2 the different RTS/CTS valid configurations.
--	--	--	---

**List of Error Codes for the protocol sent by the PLC (NITP and CAMP):**

This group of error codes is returned by the remote PLC when an error is encountered processing a task code

Error Code	Description
129 [01]	Reset Current Transaction
130 [02]	Address out of Range (other than ladder logic)
131 [03]	Requested data not found
132 [04]	Illegal Task Code Request
133 [05]	Request exceeds available memory
134 [06]	Diagnostic fail on power up
135 [07]	Fatal error detected
136 [08]	Keylock/password protection error
137 [09]	Incorrect amount of data sent with request
138 [0A]	Illegal request in current operational mode
139 [0B]	Network was not deleted
140 [0C]	Attempted write operation did not verify
141 [0D]	Illegal number of ASCII characters received
142 [0E]	Illegal request when running from EEPROM or flash
143 [0F]	Data not inserted
144 [10]	Data not written
145 [11]	Illegal data sent with command
146 [12]	Invalid operation with NIM local/remote mode (obsolete)



147;[13]	The store and forward buffer is busy
148 [14]	No response from special function module
149 [15]	Illegal instruction found in program memory (may include memory address)
150 [16]	Attempted to write to protected variable (e.g. TCC, TCP)
151 [17]	No response from PLC (e.g. single scan not performed)
152 [18]	Requested memory size exceeds total available memory
153 [19]	Requested memory size is not multiple of block allocation size
154 [1A]	Requested memory size is less than minimum defined value
155 [1B]	Requested memory size is larger than maximum defined value
156 [1C]	PLC busy – cannot complete requested operation
157 [1D]	Communications error in HOLD mode – Transition to Run not allowed
158 [1E]	Port Lockout is active
159 [1F]	Attempting to delete active program via reconfiguration
160 [20]	Program load in progress or invalidated
161 [21]	I/O configuration error –too many points
162 [22]	I/O configuration error – attempt to assign Output point to multiple applications
191 [3F]	Bus error detected
192 [40]	Operating system error detected
193 [41]	Invalid control block type
194 [42]	Control block number out of range
195 [43]	Control block does not exist
196 [44]	Control Block already exists
197 [46]	Offset out of range
198 [47]	Arithmetic error detected while writing Loop or Loop Alarm parameters

199 [48]	Invalid SF program type
200 [49]	Instruction number or RAMP/SOAK step number out of range
201 [4A]	Attempt to access an integer variable as a real
202 [4B]	Attempt to access a real variable as an integer
203 [4C]	Trask code buffer overflow – too much data requested
204 [2D]	Control block size error (cannot exceed 32767 bytes)
208 [50]	Task code request buffer too large
209 [51]	Invalid SF statement size
210 [52]	Invalid return value
211 [53]	Attempt to execute a cyclic statement in a non-cyclic SF program
212 [54]	Control block is disabled
213 [55]	Control block is not disabled
214 [56]	Attempt to perform a FSTR_OUT SF statement on an empty FIFO
215 [57]	Attempt to perform a FSTR_INT SF statement on a full FIFO
216 [58]	Stack overflow while evaluating a MATH, IF-THEN, or IMATH statement
217 [59]	Maximum SF subroutine nesting level exceeded (maximum = 4)
218 [1A]	Arithmetic Overflow
219 [5B]	Invalid operator in and IF, MATH, or IMATH expression
220 [5C]	S memory overflow
221 [5D]	Attempt to divide by 0
224 [60]	Invalid data type code
225 [61]	RAMP/SOAK step type mismatch


**List of Error Codes for the protocol sent by the PLC (CAMP):**

This group of error codes is returned by the remote PLC when an error is encountered processing a task code

Error Code	Description
30 [6E]	NITP Protocol error
35 [73]	Bad or missing delimiter
36 [74]	Bad clock check character
37 [75]	Invalid Type
38 [76]	Invalid Data Character
39 [77]	Odd number of characters
40 [78]	Invalid device code
48 [80]	Invalid error character
49 [81]	No words to write
50 [82]	Invalid word count
51 [83]	Memory Address = 0
52 [84]	Write Unsuccessful
53 [85]	Invalid command code
63 [8F]	Invalid number of words
64 [90]	Unsupported address class or device class
65 [91]	Request Too Large
67 [93]	CAMP maximum response exceeded
68 [94]	Maximum number of task codes per message
69 [95]	Invalid Task Code Character Count
92 [AC]	Memory Read Error
93 [AD]	Memory Write Error

119 [C7]

Message Queue Fill

 **Tip:** The communication status can be verified by the **output** Window of the Studio's environment or by the **LogWin** module. To set a log of events for **Field Read Commands**, **Field Write Commands** and **Serial Communication** click with the right button of the mouse on the output window and chose the option setting to select these log events. When testing under a Windows CE target, you can enable the log at the unit (Tools/Logwin) and verify the file celog.txt created at the target unit.

When you are not able to establish the communication with the PLC, first of all establish the communication between the PLC Programming Tool and the PLC. Very frequently the communication is not possible due to a hardware or cable problem, or due an error or lack of configuration at the PLC. Only after the communication between the PLC Programming Software and the PLC is working fine, you can test again the supervisory driver.

When testing the communication with the Studio, you should first use the application sample described at item 7 (if it's available), instead of the new application that you are creating.

If is required to contact technical support, please have the following information available:

- Operating System (type and version): To find this information use the Tools/System Information option
- Project information: It is displayed using the option Project/Status from the Studio menu
- Driver version and communication log: Available from Studio Output when running the driver
- Device model and boards: please refer to hardware manufacture's documentation

## Application Sample

Studio provides a configured project to test the driver. It is strongly recommended to do some tests with this application before beginning the configuration of the customized project, for the follow reasons:


- To understand better the information covered in section 4 of this document.
- To verify that your configuration is working.
- To certify that the hardware used in the test (device + adapter + cable + PC) is in working conditions before beginning the configuration of the applications.

 **Note:** The Application Sample is not available for all drivers.

The Studio application can be found in the Studio installation CD or DVD, under /Examples/Communication/TI500.zip.

To perform the test, you need to follow these steps:

- Configure the device communication parameters using manufacturer programmer software.
- Open the application /Examples/Communication/TI500.zip
- Execute the application
- To display the following screen with some information about the communication, please execute the Viewer module in the Studio.

 **Tip:** The application for testing may be used like a maintenance screen for the custom application.

## History of Versions

Version	By	Date	Description of changes
1.00	Lourenco	28-Sep-2000	<ul style="list-style-type: none"> <li>First Driver Version</li> </ul>
1.01	Lourenco	30-Out-2000	<ul style="list-style-type: none"> <li>Added an option to enable/disable write to word bits;</li> <li>Made compatible with Main Driver Worksheet</li> </ul>
1.02	Lourenco	12-Jan-2001	<ul style="list-style-type: none"> <li>Fixed bug when reading addresses higher than 1024</li> </ul>
1.03	Lourenco	16-Apr-2001	<ul style="list-style-type: none"> <li>Fixed memory leaking problem</li> </ul>
1.04	Lourenco	03-Jul-2001	<ul style="list-style-type: none"> <li>Implemented double word and float point data types</li> </ul>
1.05	Lourenco	21-Dec-2001	<ul style="list-style-type: none"> <li>Fixed bug when writing digital points with the Main Driver Sheet</li> </ul>
1.06	Lourenco	01-Jul-2002	<ul style="list-style-type: none"> <li>Implemented the NITP protocol</li> </ul>
1.07	Lourenco	08-Oct-2002	<ul style="list-style-type: none"> <li>Fixed leak of memory problem</li> </ul>
1.08	Plínio M. Santana	Apr-02-2007	<ul style="list-style-type: none"> <li>Fixed writing bits, operand DSC and page numbers.</li> </ul>
1.09	André Körbes	Sep-12-2011	<ul style="list-style-type: none"> <li>Fixed values assignment</li> </ul>
1.10	Charan Manjunath	Jul-11-2013	<ul style="list-style-type: none"> <li>Added the capability of configuring the Block Size</li> <li>Added support to multiple PLCs when using Ethernet Encapsulation</li> <li>Resolved Writing Issues in NITP mode</li> <li>Resolved error handling when the PLC returns a invalid block size message</li> </ul>
1.11	Charan Manjunath	Aug-28-2013	<ul style="list-style-type: none"> <li>Resolved issue when trying to write in NITP mode with more than 1 PLC using TCP Encapsulation</li> </ul>
1.12	Anoop R Anushree Phanse	Dec-10-2015	<ul style="list-style-type: none"> <li>Fixed issue of block size error when reading float or double values</li> </ul>
1.13	Blesson Thomas	Apr-25-2016	<ul style="list-style-type: none"> <li>Added support for PID loops and Analog Alarms (like LPV, LSP, APV etc)</li> <li>Updated settings dialog to match serial settings</li> </ul>
1.14	Anushree Phanse	Sep-26-2016	<ul style="list-style-type: none"> <li>Updated Driver version for OI server release, no change in the driver.</li> </ul>
1.15	Anushree Phanse	Feb-09-2017	<ul style="list-style-type: none"> <li>Added support to C, X and Y registers higher than 4096.</li> <li>Fixed the issue of writing to consecutive C, X and Y registers.</li> </ul>
1.16	Anushree Phanse	Apr-05-2017	<ul style="list-style-type: none"> <li>Improved and added log messages with error codes that are received from the PLC</li> <li>Improved creation of groups by the main driver sheet.</li> <li>Fixed issue when response timeouts caused wrong reads for C, Y and Y registers.</li> </ul>
1.17	Anushree Phanse	Aug-30-2017	<ul style="list-style-type: none"> <li>Added new error codes for CAMP, updated driver documentation to include information about driver settings for NITP and TBP when using serial.</li> <li>Added support to CAMP</li> <li>Added support to using strings when using CAMP</li> </ul>

			<ul style="list-style-type: none"> <li>▪ Fixed bug when reading some addresses using NITP</li> <li>▪ Added support for UDP/IP</li> </ul>
1.18	Anushree Phanse	Feb-07-2018	<ul style="list-style-type: none"> <li>▪ Improved calculation of blocks</li> <li>▪ Improved validation of messages</li> <li>▪ Improved message requests</li> </ul>
1.19	Anushree Phanse	Mar-16-2018	<ul style="list-style-type: none"> <li>▪ Fixed issue for CAMP protocol when handling float values</li> </ul>
1.20	Anushree Phanse	Jun-28-2018	<ul style="list-style-type: none"> <li>▪ Improved address validation and invalid response validation.</li> </ul>