

**IDEC Communication Driver**

Driver for Serial communication with  
Devices using the IDEC MicroSmart  
Communication Protocol

**Contents**

**INTRODUCTION .....2**

**GENERAL INFORMATION .....3**

    DEVICE CHARACTERISTICS .....3

    LINK CHARACTERISTICS .....3

    DRIVER CHARACTERISTICS .....3

    CONFORMANCE TESTING .....4

**INSTALLING THE DRIVER .....5**

**CONFIGURING THE DEVICE .....6**

**CONFIGURING THE DRIVER .....6**

    SETTING THE COMMUNICATION PARAMETERS .....6

    CONFIGURING THE STANDARD DRIVER WORKSHEET .....8

**EXECUTING THE DRIVER ..... 13**

**TROUBLESHOOTING..... 14**

**SAMPLE APPLICATION..... 16**

**REVISION HISTORY ..... 17**

## Introduction

The IDEC driver enables communication between the Studio system and the IDEC MicroSmart PLC according to the specifications discussed in this document.

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

- **Introduction:** Provides an overview of the IDEC driver documentation.
- **General Information:** Provides information needed to identify all the required components (hardware and software) used to implement communication between Studio and the IDEC driver.
- **Installing the Driver:** Explains how to install the IDEC driver.
- **Configuring the Device:** Explains how to configure the MicroSmart device.
- **Configuring the Driver:** Explains how to configure the communication 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 driver configuration.
- **Revision History:** Provides a log of all modifications made to the driver and the documentation.



### Notes:

- This document assumes that you have read the “Development Environment” chapter in the product’s *Technical Reference Manual*.
- This document also assumes that you are familiar with the Windows NT/2000/XP environment. If you are unfamiliar with Windows NT/2000/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 serial communication between the Studio IDEC driver and devices using the MicroSmart protocol.

The information is organized into the following sections:

- Device Characteristics
- Link Characteristics
- Driver Characteristics

### Device Characteristics

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

- **Manufacturer:** IDEC
- **Compatible Equipment:**
  - MicroSmart PLC CPU Module
- **Programming Software:** WinLDR 4.32

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

### Link Characteristics

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

- **Device Communication Port:** Port 1 or Port 2 (RS232 port)
- **Physical Protocol:** EIA RS232
- **Logic Protocol:** MicroSmart Protocol
- **Device Runtime Software:** None
- **Specific PC Board:** None
- **Cable Wiring Scheme** IDEC FC2A-KC4C, FC2A-KP1C, FC4A-KC1C, FC4A-KC2C

### Driver Characteristics

The IDEC driver is composed of the following files:

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

#### **Notes:**

- All of the preceding files are installed in the /DRV subdirectory of the Studio installation directory.
- You must use Adobe Acrobat® Reader™ (provided on the Studio installation CD-ROM) to view the *IDEC.PDF* document.

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

- Windows 2000/XP
- Windows NT
- Windows CE (x86, ARM)

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

The IDEC driver supports the following addresses types:

Register Type	Length	Write	Read	Bit	Integer
Inputs (X or I)	1 Bit	•	•	•	–
Outputs (Y or Q)	1 Bit	•	•	•	–
Internal Relay (M)	1 Bit	•	•	•	–
Shift Register (R)	1 Bit	•	•	•	–
Timer (Preset Value) (T)	2 Bytes	•	•	•	•
Timer (Current Value) (t)	2 Bytes	•	•	•	•
Counter (Preset Value) (C)	2 Bytes	•	•	•	•
Counter (Current Value) (c)	2 Bytes	•	•	•	•
Data Register (D)	2 Bytes	•	•	•	•
Calendar - Clock (W)	2 Bytes	•	•	•	•

### **Conformance Testing**

The following hardware/software was used for conformance testing:

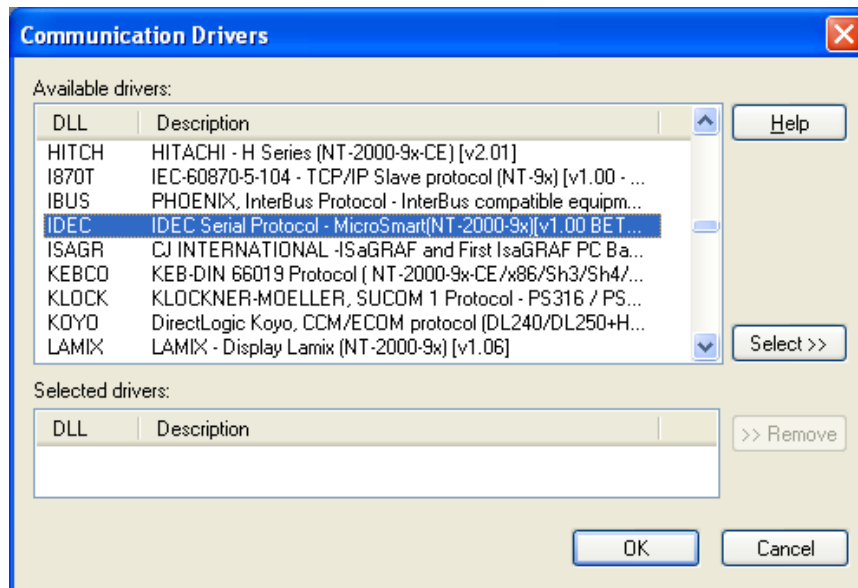
- **Equipment:** PLC IDEC MicroSmart PLC
- **Cable:** FC2A-KC4C REV.P1
- **Operating System (development):** Windows XP
- **Operating System (target):**
  - Windows XP/Vista/7
  - Windows CE
- **Studio Version:** 6.1 SP6
- **Driver Version:** 1.03

## 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 **IDEC** driver from the *Available Drivers* list (as shown in the following figure), and then click the **Select** button.



*Communication Drivers Dialog Box*

5. When the **IDEC** driver displays in the **Selected Drivers** list, click the **OK** button to close the dialog.



**Note:**

It is not necessary to install any other software on your computer to enable communication between Studio and the device. However, to download the custom program to your device you must install a WinLDR package. Consult the IDEC 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 Device

The PLC can be a network or stand-alone device. The default communication parameters are:


- **Baud Rate:** 9600
- **Data Bits:** 7
- **Stop Bits:** 1
- **Parity:** Even

## Configuring the Driver

After opening Studio and selecting the **IDEC** driver, you must configure the driver. Configuring the IDEC driver is done in two parts:

- Specifying communication parameters
- Defining communication tags and controls in the Communication tables or Driver Worksheet

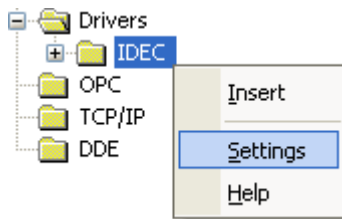
Worksheets are divided into two sections, a *Header* and a *Body*. The fields contained in these two sections are standard for all communications drivers — except the **Station**, **Header**, and **Address** fields, which are driver-specific. This document explains how to configure the **Station**, **Header**,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

 **Note:**  
For a detailed description of the Standard Driver Worksheets, and information about configuring the standard fields, review the product's *Technical Reference Manual*.

### Setting the Communication Parameters

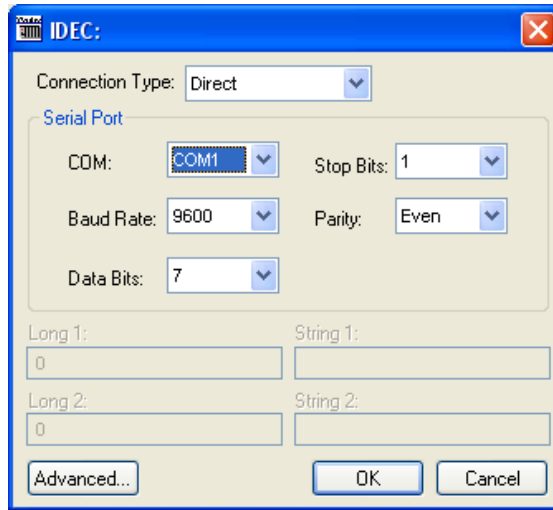
Use the following steps to configure the communication parameters, which are valid for all driver worksheets configured in the system:

1. From the Studio development environment, select the *Comm* tab located below the *Workspace*.
2. Click on the *Drivers* folder in the *Workspace* to expand the folder.
3. Right-click on the *IDEC* subfolder and when the pop-up menu displays (as shown in the following figure), select the **Settings** option.



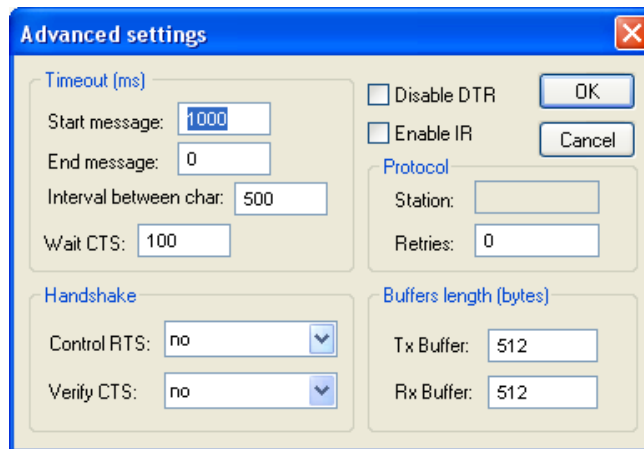
Select Settings from the Pop-Up Menu

The IDEC: Communications Parameters dialog displays (as follows).



**IDEC: Communication Parameters Dialog**


4. Click the **Advanced** button on the *Communication Parameters* dialog box to open the *Advanced Settings* dialog (shown in the following figure)



**Advanced settings Dialog**

5. Use the following table to specify the **Control RTS** (*Request to Send*) parameter:

Parameter	Default Value	Valid Values	Description
Control RTS	No	no yes yes + echo	Specify this parameter if the RTS handshake signal is set before communication and if there is an echo in the communication.  <b>Important:</b> Setting this parameter incorrectly prevents the driver from working, and generates a <i>Timeout, waiting to start a message</i> error message.

 **Notes:**

- Do not change any of the other default Advanced settings 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 these parameters only if you are using a DCE (Data Communication Equipment) converter (such as 232/485), modem, and so forth between your computer, the driver, and the host. However, before adjusting the Advanced communication parameters, you must be familiar with the characteristics of the DCE.

### Configuring the Standard Driver Worksheet

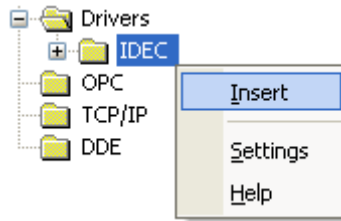
This section explains how to configure a *Standard Driver Worksheet* (or communication table) to associate application tags with the PLC addresses. You can configure multiple Driver Worksheets — each of which is divided into a *Header* section and *Body* section.

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

- From the Studio development environment, select the *Comm* tab, located below the *Workspace* pane.
- In the *Workspace* pane, expand the *Drivers* folder and right-click the *IDEC* subfolder.



3. When the pop-up menu displays (as shown in the following figure), 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 <drivername>.drv dialog box displays (similar to the following figure).

**IDEC001.DRV**

Description: Y  Increase priority

Read Trigger: Rt  Enable Read when Idle:  Read Completed:  Read Status:

Write Trigger:  Enable Write on Tag Change: 1  Write Completed:  Write Status:

Station: 0  Header: Y0   Min:   
 Max:

	Tag Name	Address	Div	Add
1	Y[0]	0		
2	Y[1]	1		
3	Y[1 0]	10		
4	Y[1 1]	11		
5	Y[1 7]	17		
6	Y[20]	20		
7	Y[21]	21		

**IDEC Driver Worksheet**

4. Use the following information to complete the **Station**, **Header**, and **Address** fields on this worksheet.

- **Station** field: Use this field to specify the PLC ID of the device. The value for the point-to-point communication needs to be 255. If you want to communicate with more than one device simultaneously via TCP/IP, set this field with the following syntax:

**<IP> : <Port Number> | <Device ID>** (For example: 192.168.1.23:5201|0)

- **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: X0)

Where:

- **Type** is the PLC Tag type. Type one of the following: X, Y, M, R, T, t, C, c, D or W
- **AddressReference** is the initial address (reference) of the register type you configured

After you edit the **Header** parameter, the system checks that the syntax is valid. If the syntax is invalid, Studio automatically inserts the default value (*D0*) into the **Header** field.

Also, 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 **Header** field.

Header Field Information			
Data Types	Sample Syntax	Valid Range of Initial Addresses	Comments
I or X	X0 or I0	0 – 30	Input Bits
Q or Y	Y0 or Q0	0 – 30	Output Bits
M	M0	0 – 127 or 800 – 807	Internal Relays – Bits
R	R0	0 – 127	Shift Register – Bits
T	T0	0 – 99	Timer preset values – Words
t	t0	0 – 99	Timer current values – Words
C	C0	0 – 99	Counter preset values – Words
c	c0	0 – 99	Counter current values – Words
D	D0	0000 – 1299, 2000 – 7999, 8000– 8199	Data Register
W	W0	0 – 6	Calendar – Clock

- **Address** field: Use the information provided in the following table 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**> (for example, 10) or <**AddressOffset**>.**[Bit]** (For example: 10.3)

Where:

- **AddressOffset** is a parameter added to the **AddressReference** parameter (configured in the **Header** field) to compose the address to be read from or written to the device.
- **Bit** is the bit number (from 0 – 15) of the Word address. This parameter is *optional* and can be combined with X, Y, R, T, t, C, c, D or W **Header** configuration.

Address Configuration Sample		
Device Address	Header Field	Address Field
I0000	X0 or I0	0
I0001	X0 or I0	1
I0010	X0 or I0	10
I0013	X0 or I0	13
Q0000	Y0 or Q0	0
Q0001	Y0 or Q0	1
Q0010	Y0 or Q0	10
Q0030	Y0 or Q0	30
Q0133	Y130 or Q130	3
Q0133	Y0 or Q0	133
M0000	M0	0.0
M0157	M0	157
M0120	M110	10
T000 (Preset)	T0	0
T050 (Preset)	T0	50
T050 (Preset)	T50	0
T050 (Preset)	T30	20
T000 (Current)	T0	0
T050 (Current)	T0	50
T050 (Current)	T50	0
T050 (Current)	T30	20
C000 (Preset)	C0	0
C050 (Preset)	C0	50
C050 (Preset)	C50	0
C050 (Preset)	C30	20
C000 (Current)	C0	0
C050 (Current)	C0	50
C050 (Current)	C50	0
C050 (Current)	C30	20
R0000	R0	0

Address Configuration Sample		
Device Address	Header Field	Address Field
R0007	R0	7
R0008	R0	8
R0016	R0	16
R0032	R0	32
D0000	D0	0
D1000	D1000	0
D1022	D1000	22

 **Note:**  
 The **W (Calendar / Clock)** type has the following address meanings:

Operand Number	DATA
0	Year
1	Month
2	Day
3	Day of week
4	Hour
5	Minute
6	Second

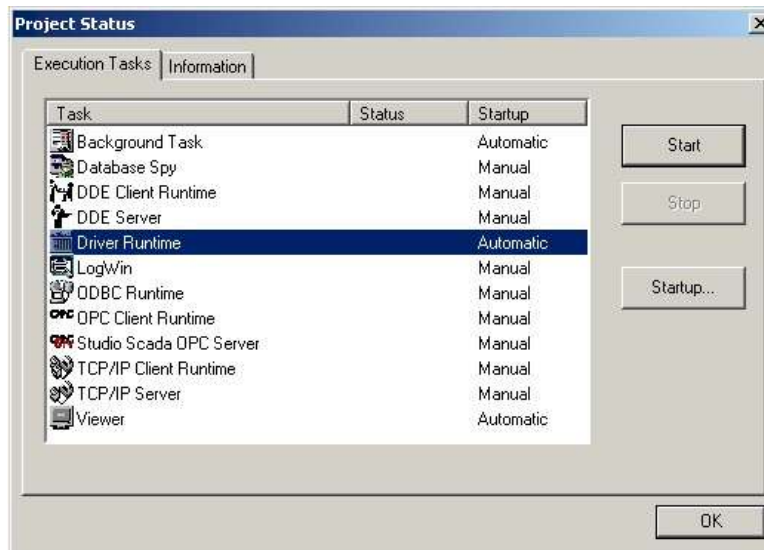
## Executing the Driver

After adding the IDEC 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, as follows.



*Project Status Dialog*

5. 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 IDEC 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.
100	Invalid Station	Wrong or invalid PLC Address typed in the "Station" field	Check the PLC address and type it correctly in the "Station" field. It needs to be between 0 and 31, or 255 for a single node station.
101	Invalid Address, address must have bit	The address type requires a bit (Inputs, Outputs, Internal Relays)	If it is an address type Input, Output or Internal Relays, you need to specify the bit number besides the Byte number, i.e. 0.1, 0.7, 15.4, etc...
102	Block size, protocol limitation	The worksheet has as offset greater then 100 to word and 200 to bits	Split this worksheet in 2 or more, and type offsets less then 100(words) or 200(bits).
103	Couldn't write bit with WriteGroup	The Address you are trying to write to on the PLC is a bit, and the "Write Trigger" could not execute it	Use the Write on Tag change to Write Bits.
104	Receive answer of wrong station	Unexpected message arrive from a PLC different then the one that was supposed to be.	Use the Field "Read" or "Write" complete in the Driver Worksheet to take the control of the PLC readings and writings, avoiding to call 2 or more PLCs at the same time.
105	Protocol error	Wrong Communication Parameter settings	Check the Communication Parameters on the Driver Settings Window. It must match exactly with the PLC configuration.
106	Checksum error	Wrong Communication Parameter settings	Check the Communication Parameters on the Driver Settings Window. It must match exactly with the PLC configuration. This error occurs very often when there is a wrong parity or Data Bits number configured.
107	PLC protected against write/read	The PLC configuration is protected against Read/Write	Check your PLC configuration to make sure that it is allowing the communication.
108	Invalid data range designated	This data type does not have the range configured on the Driver Worksheet	Check whether the range you are trying to reach matches with the ones described on the table "Header Field Information" present on the chapter "Configuring the Standard Driver Worksheet."
109	Invalid Preset value change	Preset value change is attempted with preset value designated by data register	Check your PLC configuration to make sure that it allows changing the preset value or it is tied to a Data Register.
110	Invalid value written to calendar/clock	You are probably trying to send a value such as a day greater then 31, a month greater then 12, etc... to the PLC	Check whether the Month/Day order is compatible in your application with the PLC. Check whether the values you are trying to write are consistent.

Error Code	Description	Possible Causes	Procedure to Solve
111	Invalid data other than 0(30h)-9(39h) or A(41h)-F(46h)	The data you are trying to send to the PLC are different than the Hexadecimal values	Check the consistent of the values that you are trying to send to the PLC.
-15	Timeout Start Message	<ul style="list-style-type: none"> <li>• Disconnected Cables</li> <li>• PLC is turned off, in stop mode, or in error mode</li> <li>• Wrong station number</li> </ul>	<ul style="list-style-type: none"> <li>• Check cable wiring.</li> <li>• Check the PLC state – it must be RUN.</li> <li>• Check the station address.</li> </ul>
-17	Timeout between rx char	<ul style="list-style-type: none"> <li>• PLC in stop mode or in error mode</li> <li>• Wrong station number</li> </ul>	<ul style="list-style-type: none"> <li>• Check cable wiring.</li> <li>• Check the PLC state – it must be RUN.</li> <li>• Check the station address.</li> </ul>

➔ **Tip:**  
 You can verify communication status using the Studio development environment *Output* window (*LogWin* module). To establish an event log for **Field Read Commands**, **Field Write Commands**, and **Serial Communication**, right-click in the *Output* window. When the pop-up menu displays, select the option to set the log events. If you are testing a Windows CE target, you can enable the log at the unit (**Tools** → **Logwin**) and verify the `ce1og.txt` file created at the target unit.

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**.
- **Project Information**: To find this information, select **Project** → **Status**.
- **Driver Version** and **Communication Log**: Displays in the Studio *Output* window when the driver is running.
- **Device Model** and **Boards**: Consult the hardware manufacturer's documentation for this information.

## Sample Application

You will find a sample application in the /COMMUNICATION EXAMPLES/IDEC directory on our installation CD. We strongly recommend that you use this sample application to test the IDEC driver before configuring your own customized application, for the following reasons:

- To better understand the information provided in the section of this document.
- To verify that your configuration is working satisfactorily.
- To certify that the hardware used in the test (device, adapter, cable, and PC) is working satisfactorily before you start configuring your own, customized applications.



**Note:**

This application sample is not available for all drivers.

Use the following procedure to perform the test:

1. Configure the device's communication parameters using the manufacturer's documentation.
2. Open and execute the sample application.
3. Execute the *Viewer* module in Studio to display information about the driver communication.



**Tip:**

You can use the sample application screen as the maintenance screen for your custom applications.



## Revision History

Doc. Revision	Driver Version	Author	Date	Description of Changes
A	1.00	Fabio Komura	May/21 <sup>st</sup> /2003	▪ Driver Released
B	1.01	Eric Vigiani	Oct/04 <sup>th</sup> /2006	▪ Fixed the memory leak
C	1.02	Eric Vigiani	Jul/15 <sup>th</sup> /2008	▪ Fixed problem with writing
D	1.03	Eric Vigiani	Jan/5 <sup>th</sup> /2009	▪ Fixed problem with writing to wrong address
E	1.5	Joel Nascimento	Jul/25 <sup>th</sup> /2011	▪ Modified the driver to support communication with more than one device simultaneously via TCP/IP