Communication Driver CUTL

Driver for serial communication with
Cutler-Hammer devices using RS-232

Index

1 Introduction

The CUTL driver enables communication between Studio system and some of the Cutler Hammer devices using their Host protocol by RS 232, in accordance with the characteristics covered in this document.

This document contains 8 parts, as follow:

- 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.

2 General Characteristics

2.1 Device Characteristics

- Manufacturer: Cutler Hammer

- Compatible Equipment
  - D50 – D300 Series

- Cutler Hammer PLC programmer software, for example: GPC5.

☞ Note: Please 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: RS232 port

- Physical protocol: RS232/RS485

- Logic protocol: Cutler Hammer Host Protocol

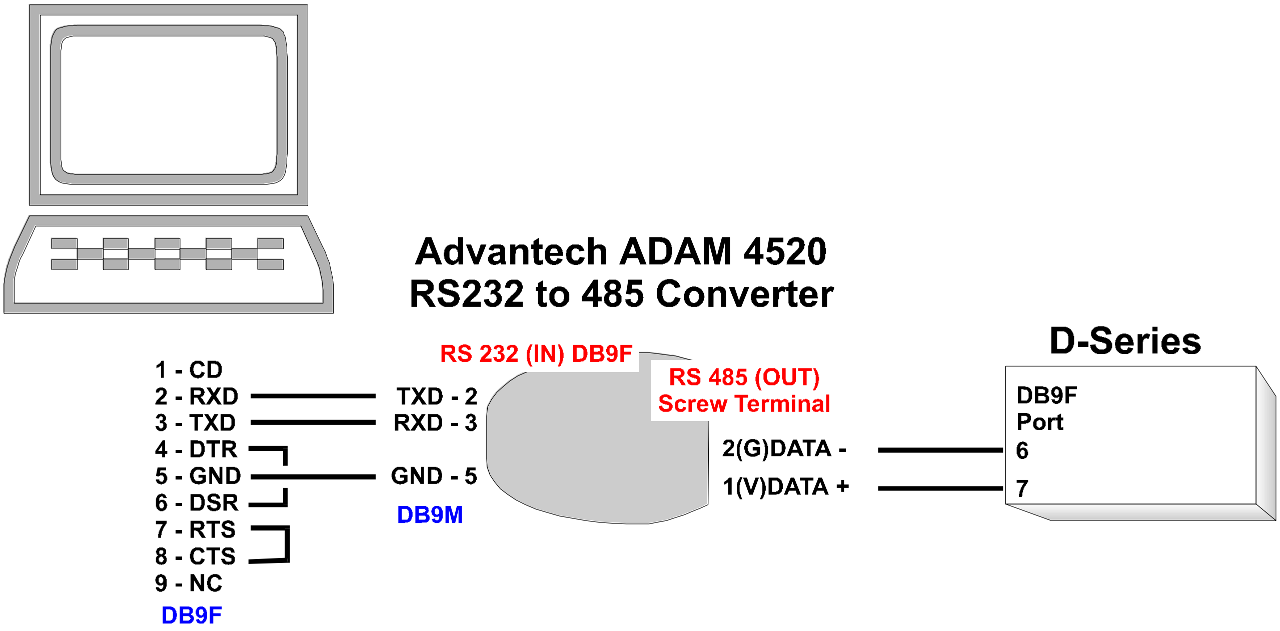
- Device Runtime software: None

- Specific PC Board: None

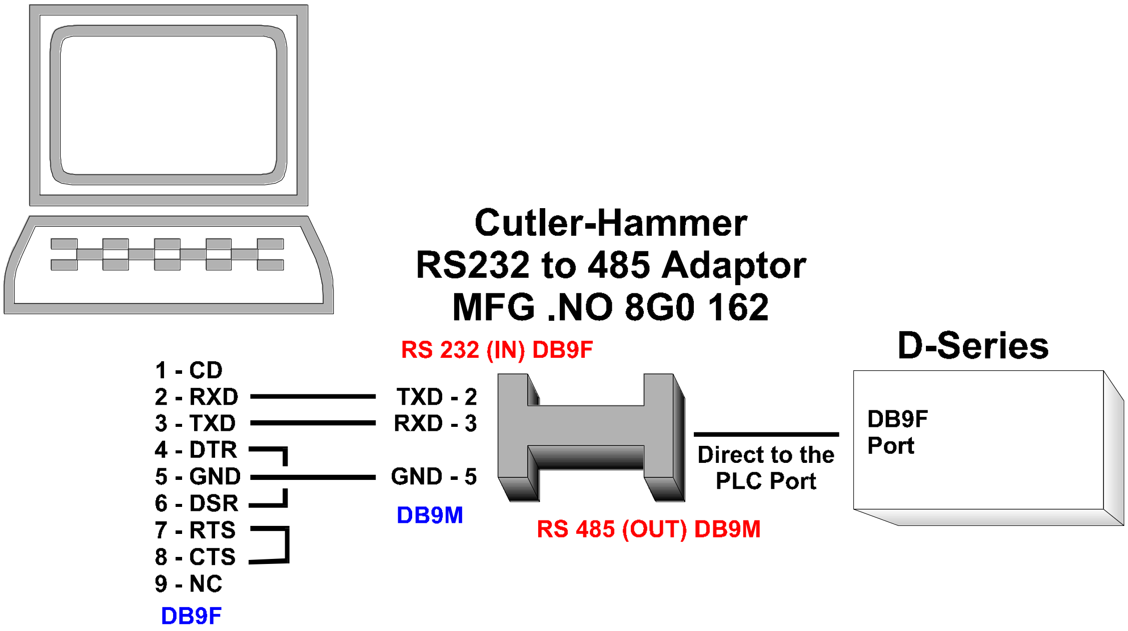
- Adapters / Converters: To run this communication under Windows NT, you can use the Cutler-Hammer MFG .NO 8G0162 RS323 to RS485 adapter, but, to run under Windows CE and Windows 95 we must use a converter that does not use the RTS / CTS control, like the Advantech ADAM-4520.

- Cable Wiring: We can see it on the following picture bellow. The same cable can be used whether a Cutler Hammer Converter is used or another one, like the ADAM.

PC with Windows 95/NT or Windows CE Unit



PC with Windows NT



2.3 Driver Characteristics

- Operating System:
  - Windows 9x
  - Windows 2000
  - Windows NT
  - Windows CE

☞ Note: 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:

- CUTL.INI: Internal file of the driver, it should not be modified by the user.
- CUTL.MSG: This file contains the error messages for each error code. It is an internal file of the driver, the user should not modify it.
- CUTL.PDF: This document provides detailed documentation about the driver.
- CUTL.DLL: This is the compiled library for the driver.

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

2.4 Information about conformance testing

- Equipment: Cutler-Hammer PLC D50 Model CR14

Configuration:

PLC program: StudioTest
Baud Rate: 9600
Protocol: Cutler Hammer Host Protocol
Communication Asynchronous Half-Duplex
Data Bits: 8
Stop Bits: 1
Parity: None
COM port: COM1

- Cable: According link specification, section 2.2.
- Operating System (development): Windows NT 4.0 + Service pack 4
- Operating System (target): Windows NT 4.0 + Service Pack 4; Windows CE v2.11
- Studio Version: 3.0
- Driver version: 2.00

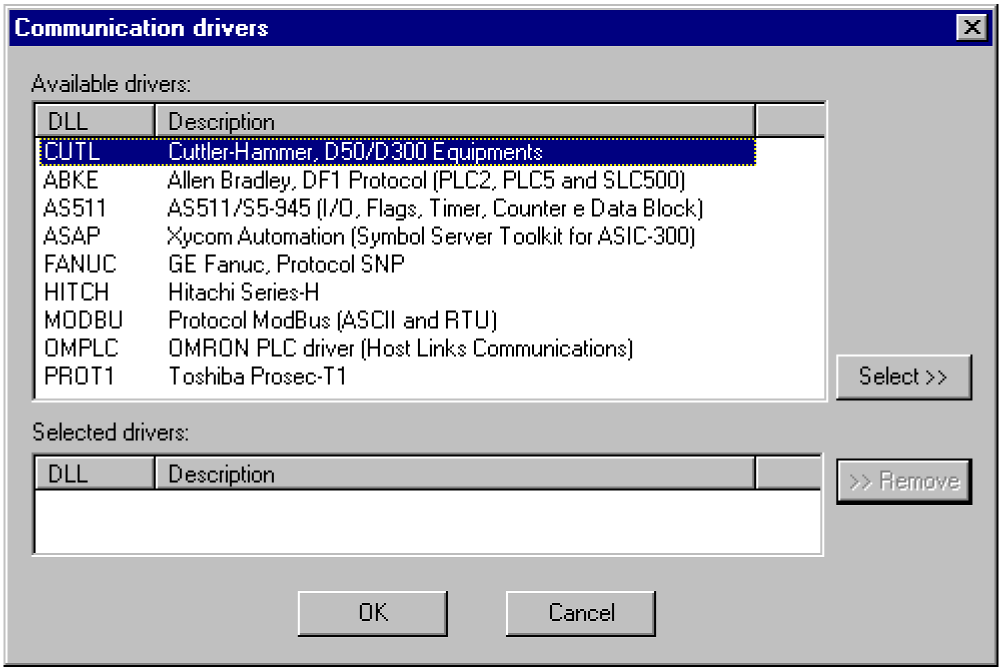
3 Installation

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 **CUTL Driver** and push the button **ADD>>>** (the driver CUTL must appear in the column **Selected Drivers**).
4. Press **OK**.



Communication drivers

Available drivers:

DLL	Description
CUTL	Cuttler-Hammer, D50/D300 Equipments
ABKE	Allen Bradley, DF1 Protocol (PLC2, PLC5 and SLC500)
AS511	AS511/S5-945 (I/O, Flags, Timer, Counter e Data Block)
ASAP	Xycom Automation (Symbol Server Toolkit for ASIC-300)
FANUC	GE Fanuc, Protocol SNP
HITCH	Hitachi Series-H
MODBU	Protocol ModBus (ASCII and RTU)
OMPLC	OMRON PLC driver (Host Links Communications)
PROT1	Toshiba Prosec-T1

Selected drivers:

DLL	Description

3.2 *Other software requirements*

It is not necessary to install any other software in the PC to enable the communication between the host and the Device. However, to download the custom program to the device, it is necessary to install one of the Cutler Hammer programmer software, for example, GPC5. Please see the GPC5 documentation about the procedure to install their software.

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

4 Driver Configuration

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

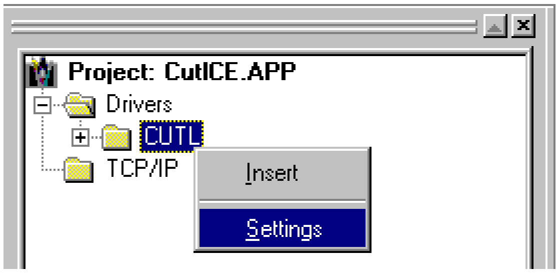
The driver configuration is two parts:

The Settings or Communication parameters, it is only one configuration to the whole driver, them you have the communication tables or Driver Worksheets, where the communication tags are defined.

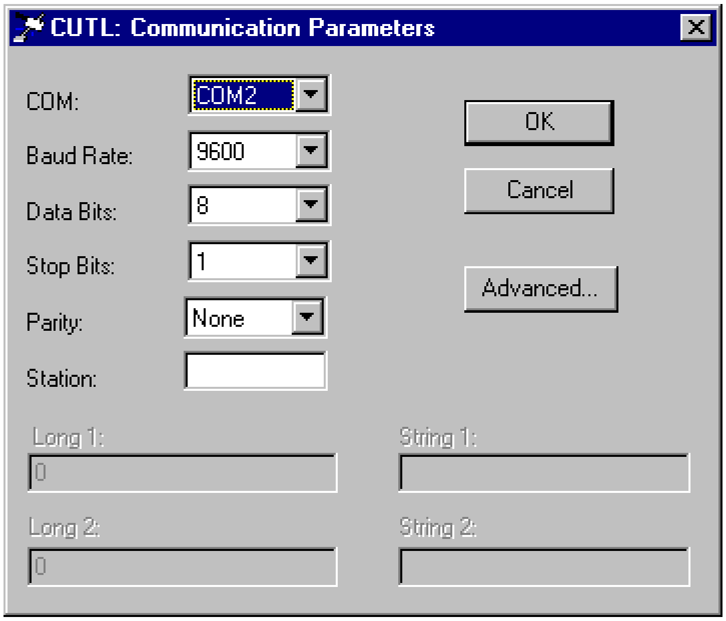
4.1 Settings - Communication Parameters

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

1. In the **Workspace** of the Studio environment, select the **Comm** table.
2. Expand the folder **Drivers** and select the subfolder **CUTL**.
3. Right click on the **CUTL** subfolder and select the option **Settings**.



When selecting the Settings, there is the following dialog to configure:



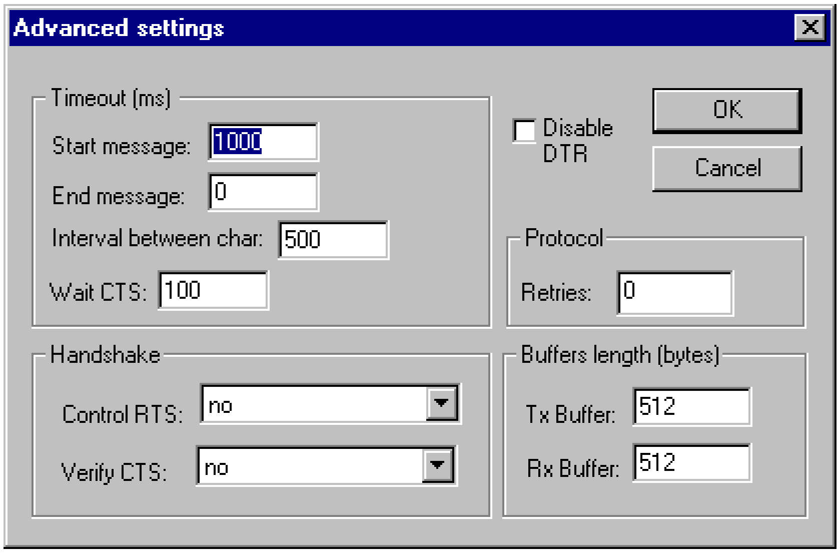
COM: COM2
Baud Rate: 9600
Data Bits: 8
Stop Bits: 1
Parity: None
Station:
Long 1: 0
String 1:
Long 2: 0
String 2:

Parameter	Default Value	Valid values	Description

COM	COM2	COM1 to COM8	Serial port of the PC used to communication with the device
Baud Rate	9600	110 to 57600bps	Communication rate of data
Data Bits	8	5 to 8	Number of data bits used in the protocol
Stop Bits	1	1or 2	Number of stop bits used in the protocol
Parity	None	even, odd, none, space or mark	Parity of the protocol
Station	0	0	Not used for the driver

☞ Note: These Parameters must be just the same as the configured on the Cutler-Hammer device. The Cutler-Hammer D50 Programmable Controller Manual indicates the following serial communication settings: Baud Rate: 9600, Data bits: 8, Stop bits: 1, Parity: None

By clicking on the button **Advanced…** in the window **Communication Parameters,** you open additional communication parameters.



Advanced settings

Timeout (ms)

Start message: 1000

End message: 0

Interval between char: 500

Wait CTS: 100

□ Disable DTR

Protocol

Retries: 0

Handshake

Control RTS: no

Verify CTS: no

Buffers length (bytes)

Tx Buffer: 512

Rx Buffer: 512

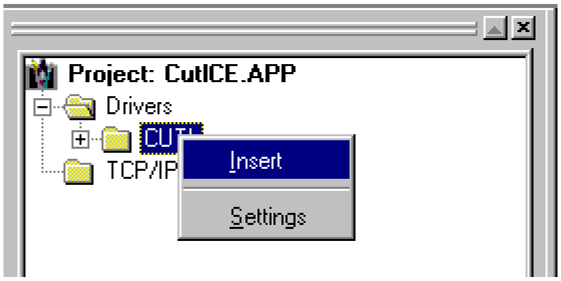
The Advanced setting parameters are explained at the Studio Technical Reference Manual, and you should keep the default values to all fields. Only the field described at the next table should be configured:

Parameter	Default Value	Valid values	Description
Control RTS	No	no, yes or yes + echo	Define if the handshake signal of RTS (Request to Send) is set before communication and if there is an echo in the communication. If we are using Windows 95 or CE with the right RS 232 – RS 485 Converter (without RTS Control), choose the option "**no**". But, if we are under Windows NT and if the Cutler Hammer RS232 – 485 adapter is used, the choice must be "**yes**". **Important**: Wrong settings on this field will not let the driver work, having the Timeout waiting start a message error.

4.2 Driver Worksheet

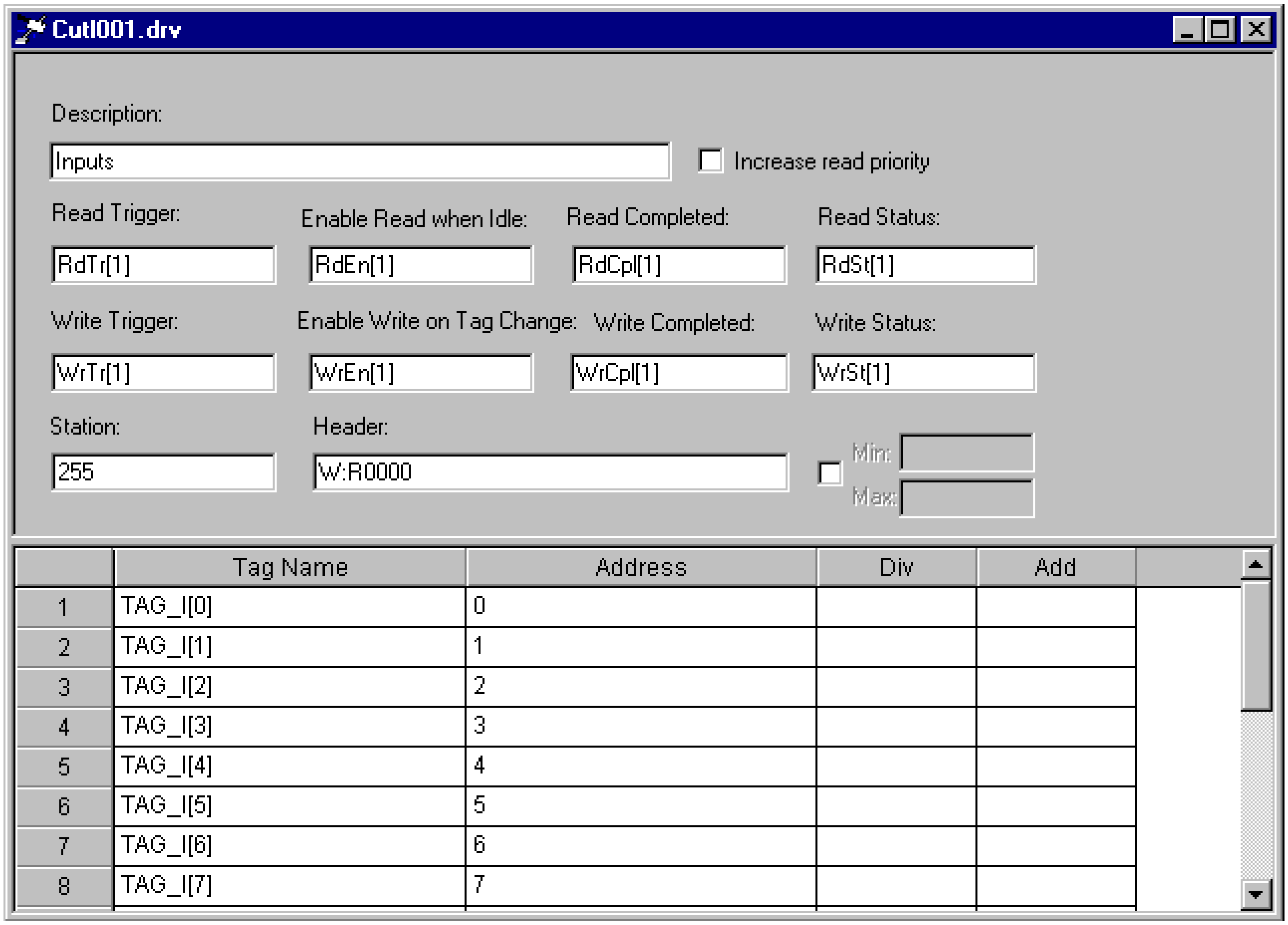
It is possible to configure many driver worksheets, each one will be composed of a Header and Body. To create a new driver worksheet, follow these steps:

1. In the **Workspace** of the Studio environment, select the table **Comm**.
2. Expand the folder **Drivers** and select the subfolder **CUTL**.
3. Right click on the **CUTL** subfolder and select the option **Insert**.



☞ Note: 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.

When creating a communication table, you have the following window:



Description: Inputs

☐ Increase read priority

Read Trigger:	Enable Read when Idle:	Read Completed:	Read Status:
RdTr[1]	RdEn[1]	RdCpl[1]	RdSt[1]
Write Trigger:	Enable Write on Tag Change:	Write Completed:	Write Status:
WrTr[1]	WrEn[1]	WrCpl[1]	WrSt[1]

Station: 255

Header: W:R0000

☐ Min: Max:

	Tag Name	Address	Div	Add
1	TAG_I[0]	0		
2	TAG_I[1]	1		
3	TAG_I[2]	2		
4	TAG_I[3]	3		
5	TAG_I[4]	4		
6	TAG_I[5]	5		
7	TAG_I[6]	6		
8	TAG_I[7]	7		

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	-	1 to 31, or 255	The PLC's Address. It can also be used the value of 255, that means direct point to point communication with any station.
Header	W:W0000	Vide next table	Defines the type of variable to be read or written from or to the device and the reference of the initial address.

The **Header** field defines the type of variables that will be read or written from or to the device. It complies with the syntax: <option>:<type of word register><initial address reference>. After editing the field **Header**, the system will check if it is valid or not. If the syntax were incorrect, the default value (W:0) will be automatically placed in this field.

You can type Tag between curly brackets into this field, but be sure that the Tag's value is correct, with the right syntax, or you will get the Invalid Header error. The right syntax, both for the field typing and Tag value is described bellow:

- **Option**: choose between W (word) or B (bit). For example, if you want to read the word R0011, one of the ways is to type on the header W:R0010. But if you want to read the bits of this word, you can type B:R01100, where B means *bit*, and the next is the bit address.
- **Type of word register**: it can be R (Inputs/Outputs), L (Link Relay), K (Retentive Internal Relay), M (Non retentive Internal Relay), F (Special Relay), W (Word Register), S (Timer/Counter set value), P (Timer/Counter present value) or T (Status)
- **Initial address**: As the communication driver establishes serial communications using messages blocks, this driver has been developed to use an initial address and a block size. So, all you need to do to type a right device's variable address is to type here the initial address and on the *Address* cells (Driver Configuration's Body), just type the offset regarding this initial address. The tables below have a lot of examples how to get the same variable's value using different headers.

There's another addressing way that is using the Word absolute address. If you want to use it, just type W:<Word Absolute address>. For example, the address W0001 has the absolute address 513. You can get this value using W:W0001 or W:513. The absolute address must be on decimal base.

Important: If you use the option *W (Word)*, the initial address must be the right word register address, for example W:R0015. But if you choose the option *B (bit)*, the initial address must be the bit address, for example B:R01600. The last two digits (00) must be between 0 or 15, because it is the bit address of the word refereed on the 3rd and 4th digits (16).

☞ Note: The bit writing is possible only using the option *B (bit)* on the header.

Information regarding the parameter "Header"			
Type	**Sample of syntax**	**Valid range of initial Address**	**Comment**
Input / Output	W:R0001 or B:R00100	Depends of the CPU. On the D50, for example is from R0000 to R0014 (R00000 to R01415), and on the D300 from R0000 to R0127 (R00000 to R12700)	Physical Inputs. "R" means Input / Output; the four next digits are the word number. If it is bit address, the three first digits after the colon defines the word's number and the two last are the bit address. Important: reading Inputs and Outputs on the same driver configuration worksheet will provide invalid out put values. Read the notes after this table.
Link Relay (D300 only)	W:L0022 or B:L02200	In the D300 it is from L0000 to L0063 (L00000 to L06315)	Link Relay. It is possible to read or write these bits or words.
Non retentive Internal relay	W:M0030 or B:M03001	Depends of the CPU On the D50, for example is from M0000 to M0031 (M00000 to M03115)	Non retentive internal relay. It is possible to read or write these bits or words.
Retentive Internal relay	W:K0015 or B:K01501	Depends of the CPU On the D50, for example is from K0000 to K0015 (K00000 to K01515)	Retentive internal relay. It is possible to read or write these bits or words.
Internal Flag (Special Relay) D300 only	W:F0011 or B:F01100	On the D300 it is from F0000 up to F0015 (F00000 up to F01515	Special Relay. Use these bits or words to read only. Writing Special Relay can generate unexpected CPU behavior.
Word Register	W:W0000	Depends of the CPU. On the D50, for example is from W0000 to W0255	You can read and write only the whole word. The Header option B (bit) is not allowed to this operator. If we want to read a bit, the only possible is to type the header with the W (Word) option and on the Address cell type a dot followed by the bit number. For example,

			Header = W0000, and address = 0.5 (bit number 5 from the Word register number 0). It is not possible to write bits on this operator. The greatest possible offset is 126.
Timer Counter Set value	W:S0000	Depends of the CPU. On the D50, for example is from SV0000 to SV0255	Here you get the Timer / Counter set value. You can read and write only the whole word. The Header option B (bit) is not allowed to this operator. If we want to read a bit, the only possible is to type the header with the W (Word) option and on the Address cell type a dot followed by the bit number. For example, Header = S0000, and address = 0.5 (bit number 5 from the Timer/Counter number 0). It is not possible to write bits on this operator. The greatest possible offset is 126.
Timer Counter Present value	W:P0000	Depends of the CPU. On the D50, for example is from PV0000 to PV0255	Here you get the Timer / Counter present value. You can read and write only the whole word. The Header option B (bit) is not allowed to this operator. If we want to read a bit, the only possible is to type the header with the W (Word) option and on the Address cell type a dot followed by the bit number. For example, Header = P0000, and address = 0.5 (bit number 5 from the Word register number 0). It is not possible to write bits on this operator. The greatest possible offset is 126.
Status	W:T0000	Depends of the CPU. On the D50, for example is from SR0000 to SR0255	Here is an important operator's type. Remember that you can change the PLC address and stop the communication if you change the SR000 value! You can read and write only the whole word. The Header option B (bit) is not allowed to this operator. If we want to read a bit, the only possible is to type the header with the W (Word) option and on the Address cell type a dot followed by the bit number. For example, Header = T0000, and address = 0.5 (bit number 5 from the Word register number 0). It is not possible to write bits on this operator. The greatest possible offset is 126.

☞ Note: Always creates two different driver worksheets to read Inputs and Outputs, even they have similar addresses. On the D50, for example, read and write from the addresses R0000 to R0014 to the inputs, and in another worksheet from R0015 to R0029. There's no way to set a read or write status error value, but the read values will be inconsistent.

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:

<offset>.<bit number>

offset: the **offset** is a number that when added to the **initial address reference** configured in the **Header** field defines an address in the device. If you're using the header W option (Word), it's the Word Offset. If you're using the B (bit) option, it's the bit offset. In both cases, the highest allowed offset is 126.

Important: if you configure an offset that reaches above the word register limit, it will be got invalid read values. For example, if you configure to read from K0000 to K0020, the read values will be inconsistent because you have only from K0000 to K0015. The read status still Ok, but the values are invalid.

bit: The **bit** is optional and when used it specifies a bit, always from 0 to 15 even in the I/Os, of the address configured.

Sample of Addressing Configuration		
Address on the Device	**Header Field**	**Address Field**
R0000	W:R0000	0
R0015	W:R0000	15
R0015	W:15	0
R0015	W:0	15
R01500	B:R01500	0
R01500	W:R0000	15.0
R01500	W:R0015	0.0
R01510	W:R0015	0.10
R01510	B:R01500	10
R01510	B:R01510	0
M0020	W:M0000	20
M0020	W:M0020	0
K0010	W:K0010	0
K0010	W:K0000	10
W0010	W:W0010	0
W0010	W:W0000	10
W0010	W:522	0
W0010	W:512	10
SV000	W:S0000	0
SV050	W:S0000	50
SV050	W:S0050	0
SV050	W:2560	50
SV050	W:2610	0
PV000	W:S0000	0
PV050	W:P0000	50
PV050	W:P0050	0
PV050	W:2816	50
PV050	W:2866	0
SR000	W:T0000	0
SR050	W:T0000	50
SR050	W:T0050	0
SR050	W:3072	50
SR050	W:3122	0

☞ Note: In the previous table there are several ways to set the same variable on the device because of the variable's number defined by the sum of the **initial address reference** defined in the field header and the **offset** defined in the Address Field.

4.5 Device Configuration

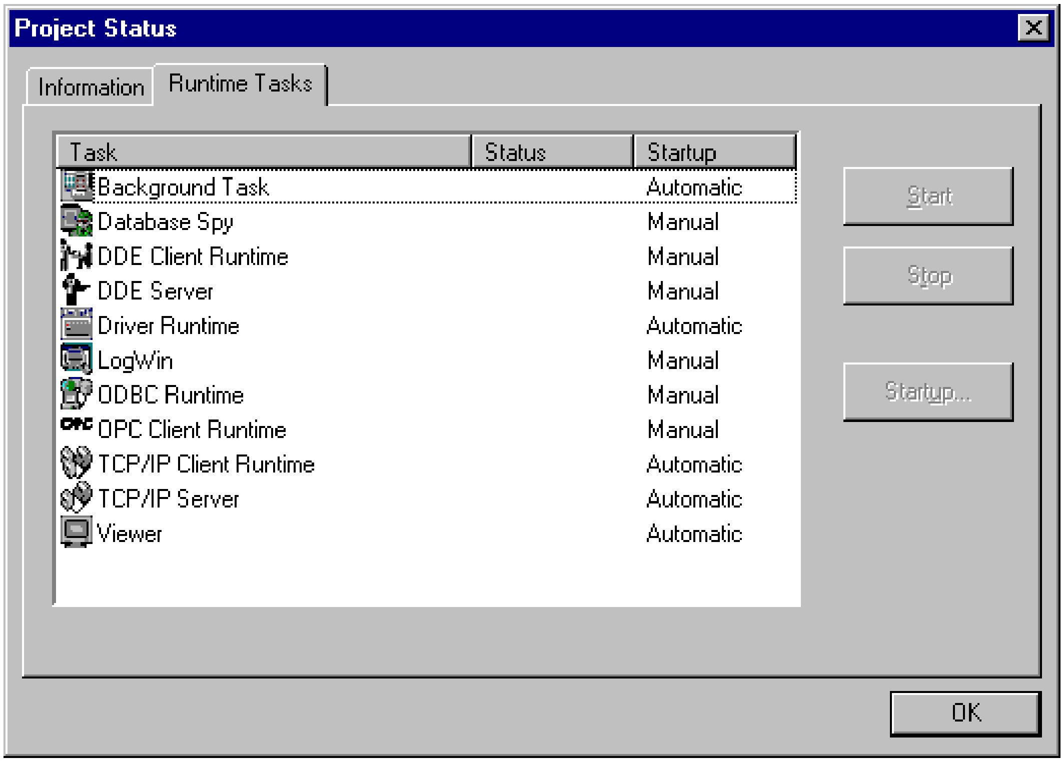
The Cutler-Hammer D50 Programmable Controller Manual indicates the following serial communication settings:

Baud Rate: 9600
Data bits: 8
Stop bits: 1
Parity: None

Besides that, it is required to select the PLC station number, using the PLC programming tools.

5 Execution

- When installing the driver, it is automatically selected to execute when you start-up the Runtime Environment. To verify the if the driver is correctly enabled to start, use the menu option **Project** + **Status…,** and verify the task Driver Runtime



Task	Status	Startup
Background Task		Automatic
Database Spy		Manual
DDE Client Runtime		Manual
DDE Server		Manual
Driver Runtime		Automatic
LogWin		Manual
ODBC Runtime		Manual
OPC Client Runtime		Manual
TCP/IP Client Runtime		Automatic
TCP/IP Server		Automatic
Viewer		Automatic

6 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	-
4	Block Size Error	Address offset greater then 126	Check on the driver configuration's worksheet that is getting this error if the configured offset s is greater then 126. If so, correct it.
5	Invalid Header	An invalid Header has been typed or the tag that is inside this field has an invalid configuration.	Type a valid Header either on the header field or on the tag value. A lot of different valid headers are shown on the section 4.2
6	Checksum Error	It's a protocol error	Check the serial communication configuration. Verify if the settings on the Communication Parameters and on the device are the same.
-15	Timeout waiting start a message.	- Disconnected cables - PLC turned off, or in Stop or error mode - Wrong Station number - Wrong RTS/CTS control settings.	- Check the cable wiring - Check the PLC state. It must be RUN - Check the station number. - Check the right configuration. See on the section 2.2 the different RTS/CTS valid configurations.
-17	Timeout between rx char.	- PLC in stop or error mode - Wrong station number - Wrong parity - Wrong RTS/CTS configuration settings	- Check the cable wiring - Check the PLC state. It must be RUN - Check the station number. - Check the right configuration. See on the section 2.2 the different RTS/CTS valid configurations.

☞ Note: The results of the communication may be verified in the **output** Window of the Studio's environment. 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, you should first of all establish the communication between the PLC Programming Tool and the PLC. Very frequently the communication it 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.
When testing the communication with the Studio, you should first use the application sample described at item 7, 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

7 Application Sample

The Studio contains 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 is in the directory: /COMMUNICATION EXAMPLES/<Driver Name>

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

- Configure the device communication parameters using manufacturer programmer software.
- Open the application /COMMUNICATION EXAMPLES/<Driver Name>
- Execute the application
- To display the following screen with some information about the communication, please execute the Viewer module in the Studio.

☞ Note: The application for testing may be used like a maintenance screen for the custom application.

8 History of Versions

Version	By	Date	Description of changes
2.01	Roberto V. Junior	26-Jan-2000	▪ Fixed bug related with write functions (locking error).
2.00	Roberto V. Junior	30-sep-1999	▪ First driver version ▪ Driver available for Windows CE