

FERT Communication Driver

Driver for Serial Communication
with FERTRON Devices

Contents

INTRODUCTION2

GENERAL INFORMATION3

 DEVICE CHARACTERISTICS3

 LINK CHARACTERISTICS.....3

 DRIVER CHARACTERISTICS3

 CONFORMANCE TESTING4

INSTALLING THE DRIVER5

CONFIGURING THE DRIVER6

 SETTING THE COMMUNICATION PARAMETERS6

 CONFIGURING THE DRIVER WORKSHEETS8

 DEVICE CONFIGURATION 15

EXECUTING THE DRIVER 16

TROUBLESHOOTING 17

SAMPLE APPLICATION 19

REVISION HISTORY..... 20

Introduction

The FERT driver enables communication between Studio system and some of the FERTRON devices by using their Host proprietary protocol, in accordance with the characteristics covered in this document.

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

- **Introduction:** Provides an overview of the driver documentation.
- **General Information:** Provides information necessary to identify all the required components (hardware and software) necessary to implement the communication and global characteristics about the communication.
- **Installing the Driver:** Explains the procedures that must be followed to install the software and hardware required for the communication.
- **Configuring the Driver:** Provides the required information to configure the communication driver such as the different permutations for configuration and the driver's default values.
- **Executing the Driver:** Explains 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.
- **Sample Applications:** Provides a sample application for testing the driver configuration.
- **Revision History:** Provides a log of all the modifications done to the driver.



Notes:

- This document assumes that you have read the "Development Environment" chapter in the Studio *Technical Reference Manual*.
- This document also assumes that you are familiar with the Windows 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 communication with the FERT driver.

The information is organized into the following sections:

- Device Characteristics
- Link Characteristics
- Driver Characteristics

Device Characteristics

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

Manufacturer: Fertron

Compatible Equipment:

- PHC300 and PHC400 Series

Fertron PLC programmer software:

- FertSoft99

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

Link Characteristics

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

- **Device Communication Port:** RS485 port
- **Physical Protocol:** Serial (RS232/RS485)
- **Logic Protocol:** Fertron Proprietary Protocol
- **Device Runtime software:** None
- **Specific PC Board:** None
- **Adapters/Converters:** Fertron Insulator & Interface RS400 (RS232/RS485)
- **Cable Wiring:** Standard RS232 Cable (PC ↔ Adapter)

Driver Characteristics

The FERT driver is composed of the following files:

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

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

- Windows 9X
- Windows 2000
- Windows NT

Conformance Testing

The following hardware/software was used for conformance testing:

- **Driver Configuration:**
 - **PLC program:** FertTes
 - **COM Port:** COM2
 - **Baud Rate:** 57600
 - **Protocol:** Token Pass and Master/Slave
 - **Data Bits:** 8
 - **Stop Bits:** 1
 - **Parity:** Even

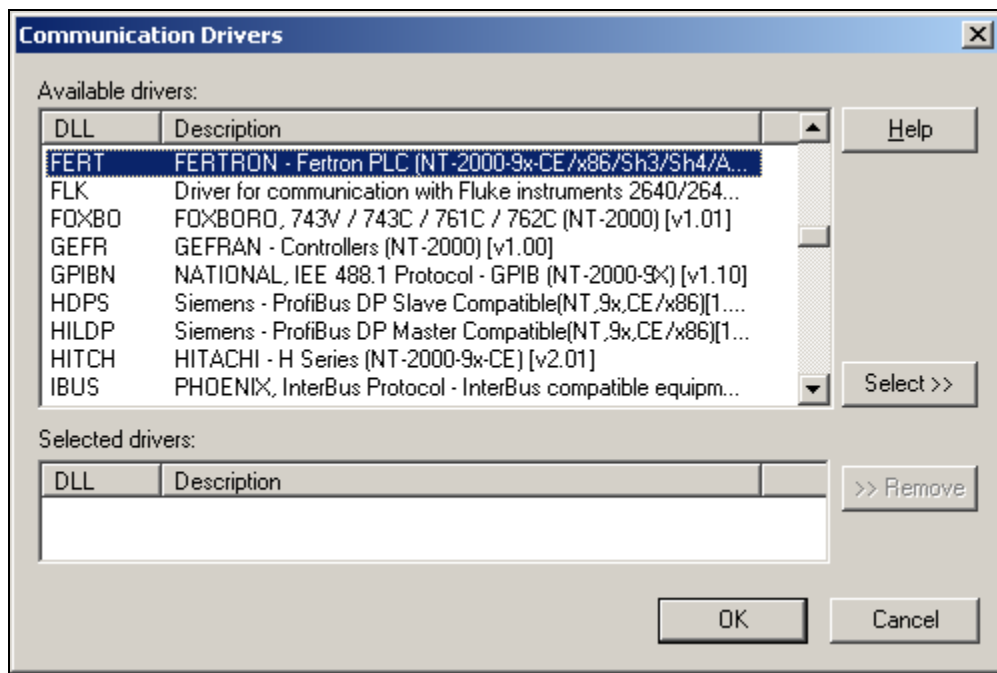
Driver Version	Studio Version	Operating System (development and runtime)	Equipment
1.01	3.01	Windows NT 4.0 + Service pack 6	Fertron PHC-400 (SCCPU and SC4AIO-FV)

Installing the Driver

When you install Studio version 3.0 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 **Fertron** 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 **FERT** driver displays in the **Selected Drivers** list, click the **OK** button to close the dialog.



Note:

It is not necessary to install any other software on your computer to enable communication between the host and the device; however, to download the custom program to the device, you must first install a copy of Fertron programmers' software (for example, FertSoft99). Please consult the Fertron documentation for the installation procedure.



Attention:

For safety reasons, you must use special precautions when installing the physical hardware. Consult the hardware manufacturer's documentation for specific instructions in this area.

Configuring the Driver

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

- Specifying communication parameters
- Defining tags and controls in the *STANDARD DRIVER SHEETS* (or Communication tables)

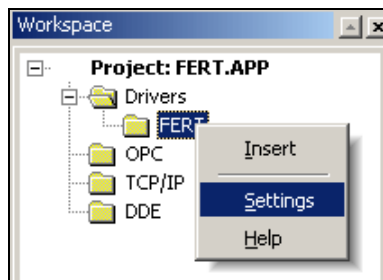
Worksheets are divided into two sections, a *Header* and a *Body*. The fields contained in these two sections are standard for all communications drivers — except the **Station**, **Header** and **Address** fields, which are driver-specific. This document explains how to configure the **Station**, **Header**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

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

Setting the Communication Parameters

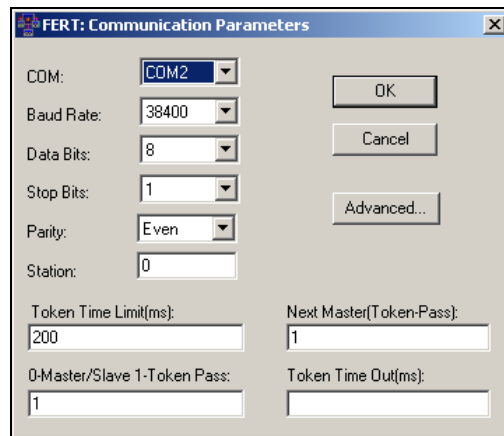
Use the following steps to configure the communication parameters, which are valid for all driver worksheets configured in the system:

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 FERT subfolder. When the pop-up menu displays (as shown in the following figure), select the **Settings** option.



Select Settings from the Pop-Up Menu

The *FERT: Communication Parameters* dialog displays (as follows).



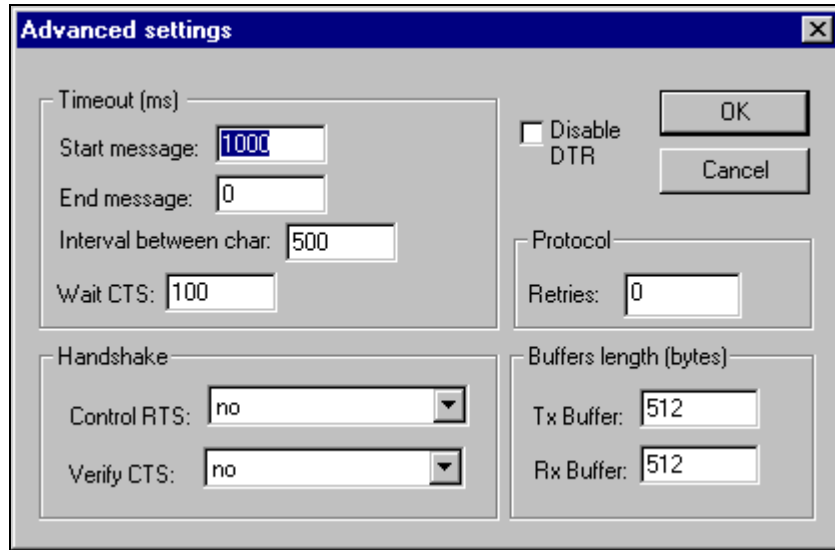
Communication Parameters Dialog

4. Use the following table to complete the fields on this dialog.

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

Parameter	Default Value	Valid Values	Description
COM	COM2	COM1 to COM8	Serial port of the PC used to communicate with the device
Baud Rate	57600	110 to 57600bps	Communication rate of data
Data Bits	8	5 to 8	Number of data bits used in the protocol
Stop Bits	1	1 or 2	Number of stop bits used in the protocol
Parity	Even	even, odd, none, space or mark	Parity of the protocol
Station	0	0 to 31	Computer ID number
0- Master/Slave 1- Token Pass	0	0 or 1	0 (Master/Slave communication) – The FERT driver is the only master in the Fertron devices network. 1 (Token Pass communication) – The Fertron device network has more than one master. The FERT driver is a master, but it never receives the data messages of another master (only the Token Pass). The Station field has the driver token identification.
Token Time(ms)	200	0 to 1000	Period that the computer keeps the token with it (used only with the Token Pass communication mode)
Next Master Station	0	0 to 31	Next device ID number, to which the token will be passed.
Token Time Out	–	200-10000	Time out for receiving token. If this field is left blank, the time out will be calculation will be based on the following expression: $1000 + (200 * Station)$

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



- Enter the following data in the **Control RTS** field.

Parameter	Default Value	Valid Values	Description
Control RTS	No	no, yes or yes + echo	Define a value to indicate whether if the handshake signal of RTS (Request to Send) is set before communication, and if there is an echo in the communication. Note that the wrong settings will impede the driver and deliver a Timeout waiting start a message error.

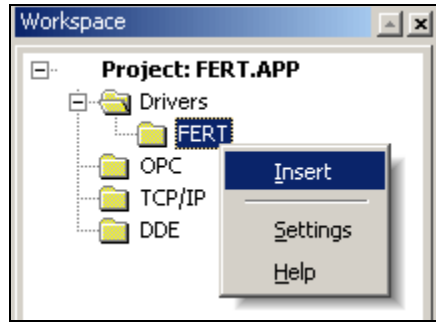
Notes:
 Do not change any of the other *Advanced* parameters at this time. You can consult the *Studio Technical Reference Manual* for information about configuring these parameters for future reference.

Configuring the Driver Worksheets

This section explains how to configure the *STANDARD DRIVER SHEETS* (or communication tables) to associate application tags with the device addresses. You can configure multiple Driver Worksheets — each of which is divided into a *Header* section and *Body* section.

- 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 *FERT* subfolder.

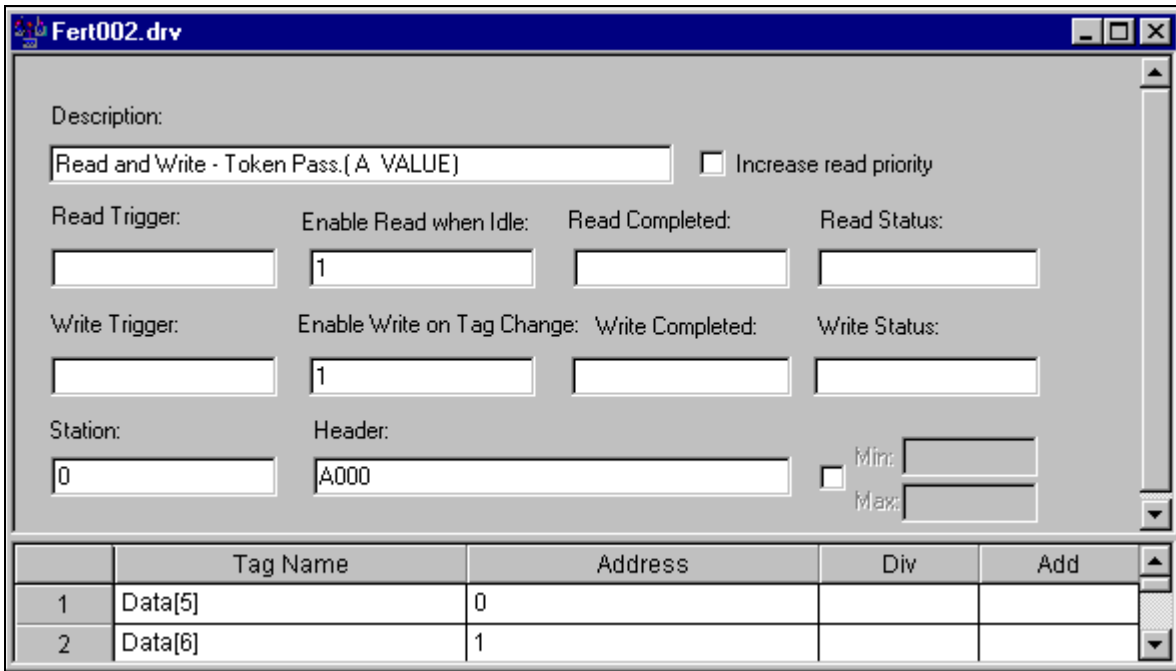
3. When the pop-up menu displays (as shown in the following figure), select the **Insert** option.



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 *STANDARD DRIVER SHEET* displays (similar to the following figure).



In general, all parameters on the Driver Worksheet (except the **Station**, **Header** and **Address** fields) are standard for all communication drivers, but they will not be discussed in this document. For detailed information about configuring the standard parameters, consult the *Studio Technical Reference Manual*.

4. Use the following information to complete the **Station** and **Header** fields on this worksheet.
- **Station** field: Type the PLC Address (ID number).
 - **Header** field: Use the information in the following table to define the type of commands that will be read from or written to the device.
 - These commands must comply with the following syntax:
 - To CPU Mnemonic: **<Mnemonic Type><Initial Address>**
 - To Physical Address: **&<Initial Physical Address>**
 - To I/O Board Mnemonic (PHC 400 compatible): **None (Empty)**
 - 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 right syntax , for the field type (or Tag value) is described below:
 - Mnemonic type: Choose among **A, DI, DO, T** or **CT**.
 - Initial address: This is the first address of the selected operand (decimal value).
 - Initial physical address: This is the first physical memory access (hexadecimal value).
 - After you edit the **Header** field, Studio checks the syntax to determine if it is valid. If the syntax is incorrect, Studio automatically inserts the default value in the **Header** field.

Header Field Information			
Type	Sample of Syntax	Valid Range of Initial Address	Comment
A	A000	Depends on the CPU To SCCPU2: 000 to 255	<Mnemonic type><Initial Address> <Mnemonic type> : A <Initial Address> : 000 to 255 (decimal) Data Type: Word or Bit Physical Memory: F000h-F1FFh
DI	DI00	Depends on the CPU To SCCPU2: 00 to 31	<Mnemonic type><Initial Address> <Mnemonic type> : DI <Initial Address> : 00 to 31 (decimal) Data Type: Word or Bit Physical Memory: 00C0h-00FFh
DO	DO00	Depends on the CPU To SCCPU2: 00 to 31	<Mnemonic type><Initial Address> <Mnemonic type> : DO <Initial Address> : 00 to 31 (decimal) Data Type: Word or Bit Physical Memory: 0100h-013Fh
T	T000	Depends on the CPU To SCCPU2: 000 to 191	<Mnemonic type><Initial Address> <Mnemonic type> : T <Initial Address> : 000 to 191 (decimal) Data Type: Word or Bit Physical Memory: F200h-F37Fh

CT	CT00	Depends on the CPU To SCCPU2: 00 to 95	<Mnemonic type><Initial Address> <Mnemonic type>: CT <Initial Address>: 00 to 95 (decimal) Data Type: Word or Bit Physical Memory: F380h-F43Fh
&<Physical Address>	&E000	Depends on the CPU	&<Initial Address> <Initial Address>: 0000 to FFFF(hexa) Data Type: Float, Word or Bit
I/O board address		Depends on the I/O board	Keep the Header field blank. The Address field must have the I/O Board Mnemonics .


5. Use the following information to complete the worksheet body, including the **Address** field. The driver worksheet body allows you to associate each tag to its respective device address.

Tag Name field: Enter the tag from your application database. This tag will receive or send values from or to an address on the device.

Address field: Use the information in the Address Configuration Sample table to associate each tag to its respective device address. The **Address** must comply with the following syntax:

- CPU Mnemonic type:
 - <offset> (type: Word)
 - <offset>.<bit number> (type: Bit)
- Physical Address type:
 - <offset> (type: Word)
 - <offset>.<bit number> (type: Bit)
 - F<offset> (type: FLOAT)

The <offset> is a number that, when added to the <Initial Address> reference configured in the **Header** field, defines an address in the device.

 **Note:**

- If the **F** character is configured before the <offset>, the data type will be a float value (4 bytes).
- If the <offset> you configure is over the word register limit, it will return invalid read values.

The <bit number> is optional. When it is used, it specifies a bit, always from 0 to 15.

- I/O Mnemonic type:
 - This configuration can only be used when the **Header** field is empty.
 - The **Address** must be configured with mnemonics compatible with PCH 400 boards.
 - The valid mnemonics and the physical addresses are in the following table:

ADR_HEX	MNEMON	ADR_HEX	MNEMON	ADR_HEX	MNEMON
E000	SP1	E020	TFT1	E040	TFT5
E004	KP1	E024	GAI1	E044	GAI5
E008	RTM1	E028	TYP1	E048	TYP5
E00C	DTM1	E02C	TAL1	E04C	TAL5
E010	SF1%	E030	AL1%	E050	AL5%
E014	ADP1	E034	AH1%	E054	AH5%

E018	ZRP1	E038	ZRI1	E058	ZRI5
E01C	MXP1	EO3C	MXI1	E05C	MXI5
ADR_HEX	MNEMON	ADR_HEX	MNEMON	ADR_HEX	MNEMON
E060	FCN1	E080	FCN5	E0A0	CR1%
E064	TLF1	E084	GAF5	E0A4	CT1%
EO68	THF1	E088	GBF5	E0A8	ZAF5
E06C	GAO1	E08C	GCF5	E0AC	MAF5
E070	GIN1	E090	GDF5	E0B0	ZBF5
E074	BI1%	E094	GEF5	E0B4	MBF5
E078	FMT1	E098	GFF5	E0B8	ZRF5
E07C	FUT1	E09C	GGF5	E0BC	MXF5
ADR_HEX	MNEMON	ADR_HEX	MNEMON	ADR_HEX	MNEMON
E0C0	X11%	E0E0	Y11%	E100	L/R1
E0C4	X12%	E0E4	Y12%	E104	D/R1
E0C8	X13%	E0E8	Y13%	E108	STP1
E0CC	X14%	E0EC	Y14%	E10C	SQR1
E0DO	X15%	E0F0	Y15%	E110	TOT1
E0D4	X16%	E0F4	Y16%	E114	CLR1
E0D8	X17%	E0F8	Y17%	E118	KYA1
E0DC	X18%	E0FC	Y18%	E11C	KYB1

ADR_HEX	MNEMON	ADR_HEX	MNEMON	ADR_HEX	MNEMON
E120	SP2	E140	TFT2	E160	TFT6
E124	KP2	E144	GAI2	E164	GAI6
E128	RTM2	E148	TYP2	E168	TYP6
E12C	DTM2	E14C	TAL2	E16C	TAL6
E130	SF2%	E150	AL2%	E170	AL6%
E134	ADP2	E154	AH2%	E174	AH6%
E138	ZRP2	E158	ZRI2	E178	ZRI6
E13C	MXP2	E15C	MXI2	E17C	MXI6
ADR_HEX	MNEMON	ADR_HEX	MNEMON	ADR_HEX	MNEMON
E180	FCN2	E1A0	FCN6	E1C0	CR2%
E184	TLF2	E1A4	GAF6	E1C4	CT2%
E188	THF2	E1A8	GBF6	E1C8	ZAF6
E18C	GAO2	E1AC	GCF6	E1CC	MAF6
E190	GIN2	E1B0	GDF6	E1D0	ZBF6
E194	BI2%	E1B4	GEF6	E1D4	MBF6
E198	FMT2	E1B8	GFF6	E1D8	ZRF6
E19C	FUT2	E1BC	GGF6	E1DC	MXF6
ADR_HEX	MNEMON	ADR_HEX	MNEMON	ADR_HEX	MNEMON
E1E0	X21%	E200	Y21%	E220	L/R2
E1E4	X22%	E204	Y22%	E224	D/R2
E1E8	X23%	E208	Y23%	E228	STP2
E1EC	X24%	E20C	Y24%	E22C	SQR2
E1FO	X25%	E210	Y25%	E230	TOT2
E1F4	X26%	E214	Y26%	E234	CLR2
E1F8	X27%	E218	Y27%	E238	KYA2
E1FC	X28%	E21C	Y28%	E23C	KYB2
ADR_HEX	MNEMON	ADR_HEX	MNEMON	ADR_HEX	MNEMON
E240	SP3	E260	TFT3	E280	TFT7
E244	KP3	E264	GAI3	E284	GAI7
E248	RTM3	E268	TYP3	E288	TYP7
E24C	DTM3	E26C	TAL3	E28C	TAL7

E250	SF3%	E270	AL3%	E290	AL7%
E254	ADP3	E274	AH3%	E294	AH7%
E258	ZRP3	E278	ZRI3	E298	ZRI7
E25C	MXP3	E27C	MXI3	E29C	MXI7
ADR_HEX	MNEMON	ADR_HEX	MNEMON	ADR_HEX	MNEMON
E2A0	FCN3	E2C0	FCN7	E2E0	CR3%
E2A4	TLF3	E2C4	GAF7	E2E4	CT3%
E2A8	THF3	E2C8	GBF7	E2E8	ZAF7
E2AC	GAO3	E2CC	GCF7	E2EC	MAF7
E2B0	GIN3	E2D0	GDF7	E2F0	ZBF7
E2B4	BI3%	E2D4	GEF7	E2F4	MBF7
E2B8	FMT3	E2D8	GFF7	E2F8	ZRF7
E2BC	FUT3	E2DC	GGF7	E2FC	MXF7
ADR_HEX	MNEMON	ADR_HEX	MNEMON	ADR_HEX	MNEMON
E300	X31%	E320	Y31%	E340	L/R3
E304	X32%	E324	Y32%	E344	D/R3
E308	X33%	E328	Y33%	E348	STP3
E30C	X34%	E32C	Y34%	E34C	SQR3
E310	X35%	E330	Y35%	E350	TOT3
E314	X36%	E334	Y36%	E354	CLR3
E318	X37%	E338	Y37%	E358	KYA3
E31C	X38%	E33C	Y38%	E35C	KYB3

ADR_HEX	MNEMON	ADR_HEX	MNEMON	ADR_HEX	MNEMON
E360	SP4	E380	TFT4	E3A0	TFT8
E364	KP4	E384	GAI4	E3A4	GAI8
E368	RTM4	E388	TYP4	E3A8	TYP8
E36C	DTM4	E38C	TAL4	E3AC	TAL8
E370	SF4%	E390	AL4%	E3B0	AL8%
E374	ADP4	E394	AH4%	E3B4	AH8%
E378	ZRP4	E398	ZRI4	E3B8	ZRI8
E37C	MXP4	E39C	MXI4	E3BC	MXI8
ADR_HEX	MNEMON	ADR_HEX	MNEMON	ADR_HEX	MNEMON
E3C0	FCN4	E3E0	FCN8	E400	CR4%
E3C4	TLF4	E3E4	GAF8	E404	CT4%
E3C8	THF4	E3E8	GBF8	E408	ZAF8
E3CC	GAO4	E3EC	GCF8	E40C	MAF8
E3D0	GIN4	E3F0	GDF8	E410	ZBF8
E3D4	BI4%	E3F4	GEF8	E414	MBF8
E3D8	FMT4	E3F8	GFF8	E418	ZRF8
E3DC	FUT4	E3FC	GGF8	E41C	MXF8
ADR_HEX	MNEMON	ADR_HEX	MNEMON	ADR_HEX	MNEMON
E420	X41%	E440	Y41%	E460	L/R4
E424	X42%	E444	Y42%	E464	D/R4
E428	X43%	E448	Y43%	E468	STP4
E42C	X44%	E44C	Y44%	E46C	SQR4
E430	X45%	E450	Y45%	E470	TOT4
E434	X46%	E454	Y46%	E474	CLR4
E438	X47%	E458	Y47%	E478	KYA4
E43C	X48%	E45C	Y48%	E47C	KYB4

ADR_HEX	MNEMON	ADR_HEX	MNEMON	ADR_HEX	MNEMON
ED00	PV1	ED20	SP1A	ED40	AIN1
ED04	PV2	ED24	SP2A	ED44	AIN2
ED08	PV3	ED28	SP3A	ED48	AIN3
ED0C	PV4	ED2C	SP4A	ED4C	AIN4
ED10	MV1%	ED30	SPR1	ED50	AIN5
ED14	MV2%	ED34	SPR2	ED54	AIN6
ED18	MV3%	ED38	SPR3	ED58	AIN7
ED1C	MV4%	ED3C	SPR4	ED5C	AIN8
ADR_HEX	MNEMON				
ED60	AOU1				
ED64	AOU2				
ED68	AOU3				
ED6C	AOU4				
ED70	TOT1				
ED74	TOT2				
ED78	TOT3				
ED7C	TOT4				

Address Configuration Sample		
Address on the Device	Header Field	Address Field
DI000	DI000	0
DI005	DI000	5
DI005	DI005	0
DO001	DO001	0
DO0012	DO010	2
A000	A000	0
A0020	A010	10
T100	T100	0
T101	T100	1
CT50	CT25	25
A000(Physical Address - CPU)	&F000	0
A001(Physical Address - CPU)	&F002	0
A001(Physical Address - CPU)	&F000	2
SP1 (I/O Board)		SP1
KP1 (I/O Board)		KP1
TOT2		TOT2
SP2 (Physical Address – I/O Board)	&E120	F0
KP2 (Physical Address – I/O Board)	&E120	F4
KP2 (Physical Address – I/O Board)	&E124	F0
DI000 (Bit 0)	DI000	0.0
DI000 (Bit 1)	DI000	0.1
DI000 (Bit 5)	DI000	0.5
DO05 (Bit 0)	DO005	0.0
DO05 (Bit 0)	DO000	5.0
DO08 (Bit 10)	DO003	5.10
CT000 (Bit 2) (Physical Address – CPU)	&F380	0.2
T003 (Bit 8) (Physical Address – CPU)	&F206	0.8
T003 (Bit 8) (Physical Address – CPU)	&F200	6.8
T000 (Bit 1) (Physical Address – CPU)	&F200	0.1



Note:

There are several ways to set the same variable on the device because the variable's number is defined by the sum of the <Initial Address> reference entered in the **Header** field, and the <offset> entered in the **Address** field.

Device Configuration

- The Fertron Programmable Controller Manual uses the following serial communication settings:
 - **Baud Rate:** 57600
 - **Data Bits:** 8
 - **Stop Bits:** 1
 - **Parity:** Even
- The PLC station number must be selected with the PLC programming tools.
- There are two jumps of communication parameters in the board:
 - **JP1** – Baud rate communication (57600 or 38400)
 - **JP2** - Token Pass ON/OFF



Note:

The FERT driver must be configured with *exactly the same* parameters that you configured in the *Communication Parameters* dialog.

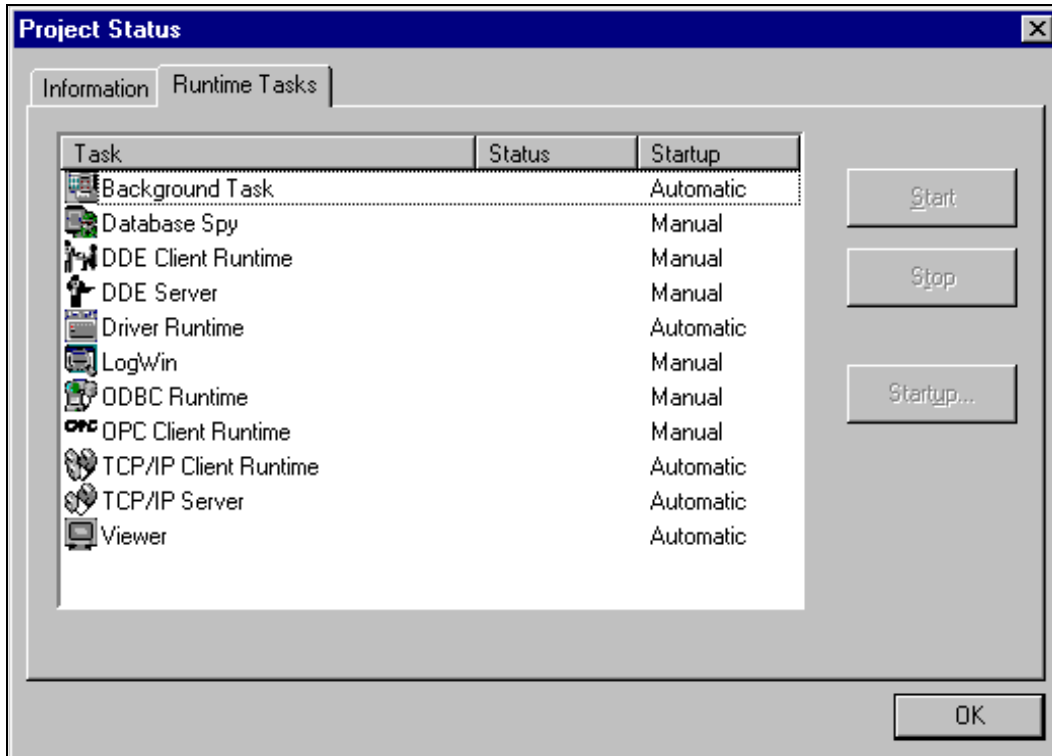
Executing the Driver

After adding the FERT 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 box displays, as follows.



Project Status Dialog Box

2. Verify that the *Driver Runtime* task is set to **Automatic**.
 - If the setting is correct, click **OK** to close the dialog box.
 - 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 FERT 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 the type of failure.

Error Code	Description (*)	Possible Causes	Procedure to Solve
0	OK	Communication without problems	N/A
4	Token use timeout	The FERT driver keeps the token longer than the stated Token Time(ms) value	Increase the Token Time(ms) value.
5	Block size error	The Address offset is greater than the buffer limit (128 words).	Check the worksheet of the driver that is getting this error. If the configured offset is greater then 128 words, correct it.
7	Error invalid address	An invalid Address has been typed.	Enter a valid Address in the Address field.
9	Error memory	The FERT driver can not allocate memory to buffers.	Check the free memory of the computer.
46	Checksum error	Protocol error	Check the serial communication configuration. Verify that the settings on the Communication Parameters and on the device are the same.
230	Error invalid command	Bit write in the wrong operand	Modify the operand type or data type to use bit write.
-15	Timeout waiting start a message	<ul style="list-style-type: none"> - Disconnected cables - PLC turned off, or in Stop or error mode - Wrong station number - Wrong RTS/CTS control settings. 	<ul style="list-style-type: none"> - Check the cable wiring. - Check the PLC state. It must be RUN. - Check the station number. - Check the configuration. See Control RTS on page 8 for valid values.
-17	Timeout between rx char	<ul style="list-style-type: none"> - PLC in stop or error mode - Wrong station number - Wrong parity - Wrong RTS/CTS configuration settings 	<ul style="list-style-type: none"> - Check the cable wiring. - Check the PLC state. It must be RUN. - Check the station number. - Check the configuration. See Control RTS on page 8 for valid values.

⇒ **Tip:**

You can verify communication status with 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 use Studio's Remote LogWin (**Tools** → **Remote Logwin**) to get the log events from the target unit remotely.

If you are unable to establish communication with the PLC, try to establish communication between the PLC Programming Tool and the PLC. Quite frequently, communication is not possible because you have a hardware or cable problem, or a PLC configuration error. After successfully establishing communication between the device's Programming Tool and the PLC, you can retest the supervisory driver.

To test communication with Studio, we recommend using the sample application provided rather than your new application.

If you must contact us for technical support, please have the following information available:

- **Operating System** (type and version): To find this information, select **Tools** → **System Information**.
- **Studio version**: To find this information, select **Help** → **About**.
- **Driver Version**: To find this information, read the full description of the driver on the *Communication Drivers* dialog box.
- **Communication Log**: Displays in the Studio *Output* window (or *LogWin* window) when the driver is running. Be sure to enable the **Field Read Commands**, **Field Write Commands** and **Serial Communication** for the LogWin window.
- **Device Model** and **Boards**: Consult the hardware manufacturer's documentation for this information.

Sample Application

You will find a sample application for drivers in the `/COMMUNICATION EXAMPLES/FERT` directory. We strongly recommend that you check for a sample application for this driver and use it to test the driver before configuring your own, customized application, for the following reasons:

- To better understand the information provided in each 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.

 **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	Luis Fernando Espinosa	30 July 1999	<ul style="list-style-type: none">Initial version
B	1.01	Roberto V. Junior	04 April 2000	<ul style="list-style-type: none">Fixed bugs related to the Master/Slave communication (checksum)Implemented the Token Pass communication modeImplemented the mnemonics for Header and Address fields.
C	1.03	José Lourenço Teodoro	04 April 2000	<ul style="list-style-type: none">Implemented token time out optionFixed problems with token exchange
D	1.04	Roberto V. Junior	08 Dec 2004	<ul style="list-style-type: none">Included Treat to Token when Master/SlaveFixed problems with token exchange