



AGILiGATE PROFIBUS - MODBUS

Technical Documentation



Parc d'activités Giraudeau
6, rue Auguste Perret
37000 TOURS
FRANCE

Tel: +33 (0)2 47 76 10 20

Fax: +33 (0)2 47 37 95 54

Email: info@agilicom.fr

Web: www.agilicom.fr

CONTENTS

1. GENERAL PRESENTATION OF AGILIGATE	4
1.1. HARDWARE PRESENTATION	4
1.2. COMPATIBILITY WITH OTHER FIELDBUSES	4
1.3. CONFIGURATION PRINCIPLE	4
1.4. SERIAL MODBUS	5
1.5. SERIAL OR USB DIAGNOSTIC PORT	5
1.6. PROFIBUS PORT	5
2. TECHNICAL SPECIFICATIONS.....	6
2.1. PHYSICAL & ELECTRICAL.....	6
2.2. MECHANICAL DATA.....	7
2.3. SUPPORTED MODBUS FUNCTIONS	7
3. HARDWARE DESCRIPTION.....	8
3.1. CONNECTORS	8
3.2. DESCRIPTION OF THE FRONT PANEL.....	9
3.2.1. LEDs	9
3.2.2. Selection of the PROFIBUS address	9
3.2.3. RS232/485 selection	10
3.2.4. Bus termination resistor	10
4. CONFIGURATION OF THE GATEWAY.....	11
4.1. PROFIBUS ADDRESS	11
4.2. PRINCIPLE FOR CONFIGURING AGILIGATE.....	11
4.3. SETUP OF THE SERIAL MODBUS.....	12
4.4. MODBUS MASTER MODE: DEFINITION OF THE SCENARIOS	13
4.5. MODBUS SLAVE MODE	15
5. MEMORY MAP	16
5.1. MODBUS MASTER MODE.....	16
5.2. SLAVE MODBUS MODE.....	17
6. CYCLE TIME OF AGILIGATE	18
7. DIAGNOSTIC AND HELP FOR SETUP	20
7.1. EXTENDED PROFIBUS DIAGNOSTIC	20
7.2. USING THE DIAGNOSTIC SERIAL INTERFACE.....	20
7.3. USING THE USB DIAGNOSTIC.....	22
APPENDICES	25
APPENDIX A: FORMAT OF MODBUS MESSAGES.....	26
APPENDIX B: LIST OF ERRORS SENT BY AGILIGATE.....	36

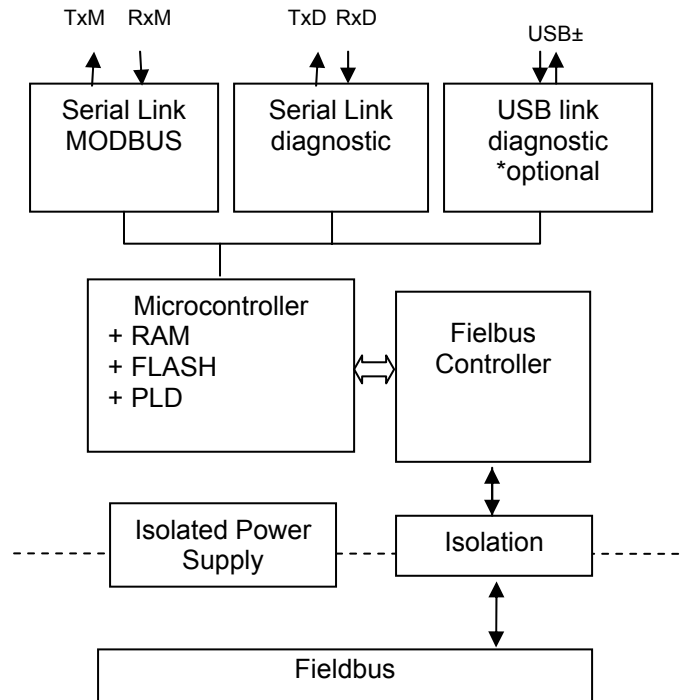
Modifications of the document		
Version	Date	Description
A	09-2004	- Creation.
B	01-2005	- Modification of AGILiCOM address.
C	01-2006	<ul style="list-style-type: none"> - AGILiGATE PROFIBUS v1.1. - GSD AGIL0834 v1.1. - The Modbus function 23 in slave mode is added. - Modifications of the parameters (§ 4.3 - §4.4) : <ul style="list-style-type: none"> ➔ The number of Modbus slave is no longer limited to 8. The slave address is defined in each scenario. ➔ Possibility to enable or disable the extended diagnostic. - Up grade of the USB diagnostic link (§7.3).
D	02-2006	<ul style="list-style-type: none"> - AGILiGATE PROFIBUS v1.2. - GSD AGIL0834 v1.2. - Modification of the maximum number of read and write MODBUS registers. - Improvement of the diagnostic link.
E	02-2006	<ul style="list-style-type: none"> - AGILiGATE PROFIBUS v1.2. - GSD AGIL0834 v1.3. - Up date of the table of errors. - Modifications of the schematics. - Add of technical data
F	03-2006	<ul style="list-style-type: none"> - AGILiGATE PROFIBUS v1.3. - GSD AGIL0834 v1.4. - Add of the error #15 "Module cfg error".

1. GENERAL PRESENTATION OF AGILiGATE

1.1. Hardware presentation

The range of AGILiGATE gateways brings to your devices the connectivity to fieldbuses. The product is based on a rail DIN mountable enclosure. This gateway allows the users to minimize development costs thanks to a fully integrated fieldbus technology. AGILiGATE communicates with its host application via a serial MODBUS link.

The following draw shows the different hardware function blocks of AGILiGATE:



Draw 1. : AGILiGATE function blocs

1.2. Compatibility with other fieldbuses

The AGILiGATE product line allows communication with the PROFIBUS, PROFINET, CANOPEN and DEVICENET fieldbuses and networks. All AGILiGATE gateways are identical regarding the connection. Only the connector for the fieldbus changes as it is fieldbus dependant.

1.3. Configuration principle

AGILiGATE PROFIBUS – MODBUS can be configured via any of the PROFIBUS network configuration tool. This intuitive method avoids the user to learn a new tool or a new configuration language. The gateway can be set up very quickly, increasing the productivity of the development. Any configuration mistake and wrong parameter found by the AGILiGATE are sent to the PROFIBUS master using the PROFIBUS extended diagnostic services.

1.4. Serial MODBUS

The serial MODBUS is used for data exchange. This simple protocol is a standard broadly deployed and easy to implement. The MODBUS specification is available at www.modbus.org.

1.5. Serial or USB Diagnostic port

The PROFIBUS extended diagnostic associated with a smart software tool able to clearly show errors in text message format is a powerful and efficient way to diagnose and trouble shoot a PROFIBUS device. If the PROFIBUS tool doesn't have this capability, the user still has the possibility to connect a simple ASCII terminal to the serial port in order to monitor the error messages appearing in clear text. This feature is also available on the USB port available as option (AG-P012).

1.6. PROFIBUS port

AGILiGATE is PROFIBUS pre-certified. It is fully compliant with the current IEC 61158 standard.

2. TECHNICAL SPECIFICATIONS

2.1. Physical & Electrical

PROFIBUS DP PORT	
Baud Rate	9.6kbps – 12 Mbps (Auto Baud)
Connector	SubD9 female
Bus Termination Resistor	No
Bus Address	2 hexadecimal rotary switches
Diagnostic	27 bytes of extended diagnostic, 6 LEDs for status
Input Bytes	0 – 244 adjustable
Output Bytes	0 – 244 adjustable
Max number of Inputs/Outputs	300
Isolation	1 kV
Technology	ASIC
Others	Sync, Freeze

SÉRIAL MODBUS PORT	
Configuration	With any PROFIBUS Master configuration tool (using the GSD file)
Bus Access	Master or Slave
Protocol	MODBUS RTU or ASCII
Max number of devices on the network	32
Distance	Maximum 1200m copper cable without repeater (depending on speed and cable quality)
Coding	NRZ (Non Return to Zero)
Transmission	Half Duplex, asynchronous
Cable	Shielded twisted pair
Supported Functions	3, 4, 5, 6, 7, 16, 23
Baud Rate	600, 1200, 2400, 4800, 9600, 19200, 38400, 57600 baud
Electrical Interface	RS232 or RS485
Number of slave accessible by the master.	20 MODBUS Slaves
Range of slave addresses	1 – 247
Number of accessible MODBUS registers	1 - 122 read registers 1 - 122 write registers
Max total number of accessible MODBUS registers	150
Number of messages	1 - 20 different MODBUS messages
Message trigger	Cyclic, On change, Once
Connector	Male open connector 3 contacts
Bus termination resistor	120 Ω configurable by switch

POWER SUPPLY	
Input Voltage	14 – 30V DC
Consumption	1,5 W
Polarity protection	Yes
Short cuts protection	Yes

2.2. Mechanical data

MECHANICAL	
Housing	Plastic box IP20 – rail DIN mountable
Dimensions	120 x 100 x 23 mm (L x W x H)
Weight	About 100g
Storage Temperature	-25°C...+70°C
Operating Temperature	0°C...+55°C
Humidity	Max. 80%

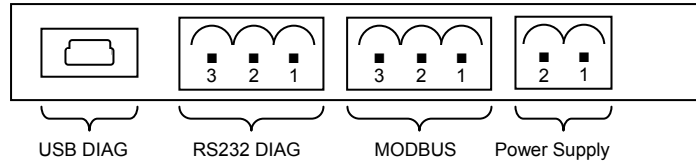
2.3. Supported MODBUS functions

Function code	Function	Description
3 (0x03)	Read multiple registers	Read from 1 to 50 MODBUS registers
4 (0x04)	Read multiple registers	Read from 1 to 50 MODBUS registers
5 (0x05)	Write Boolean	Set a Boolean to ON or to OFF
6 (0x06)	Write a register	Write one MODBUS register
7 (0x07)	Read Status	Read Status byte
16 (0x10)	Write multiple registers	Write from 1 to 50 MODBUS registers
23 (0x17)	Read/Write multiple registers	Chained Read and Write from 1 to 50 registers (only in slave mode)

Table 1 : Supported MODBUS functions

3. HARDWARE DESCRIPTION

3.1. Connectors



Draw 1: Connector on top of the enclosure

- Power supply connector:

	pin #	Name	Description
	1	VCC	DC Power 19-30 V
	2	GND	Ground

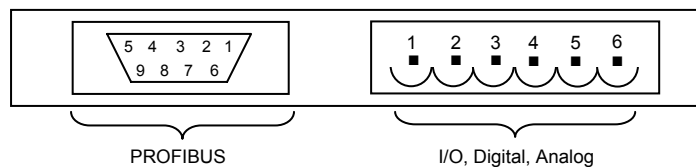
- MODBUS connector:

	pin #	Name	Description
	1	Rx/A	RS232: Receive (from Device to AGILiGATE). RS485: non-inverted line: A.
	2	Tx/B	RS232: Transmit (from AGILiGATE to device). RS485: inverted line: B.
	3	GND	Ground

Important : The shielding of the MODBUS cable must be connected to the ground, in order to guarantee a good EMC behaviour.

- RS232 DIAG connector:

	pin #	Name	Description
	1	Rx	Receive (from Device to AGILiGATE)
	2	Tx	Transmit (from AGILiGATE to device)
	3	GND	Ground



Draw 2: Connectors on bottom of the enclosure

- PROFIBUS connector:

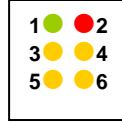
Pin #	Name	Description
3	B	PROFIBUS line B
5	GND_ISO	Isolated ground
6	5V_ISO	Isolated 5V DC
8	A	PROFIBUS line A

- I/O Connector (optional)

	Pin #	Name	Description
	1	AN+	Analog Input 4/20 mA or 0/10 V DC, depending on wiring
	2	AN-	Analog Input 0V signal
	3	E_TOR+	Isolated Digital Input.
	4	E_TOR-	Isolated Digital Input 0V.
	5	S_TOR	Digital Output (Contact relay)
	6	S_TOR	Digital Output (Contact relay)

3.2. Description of the front panel

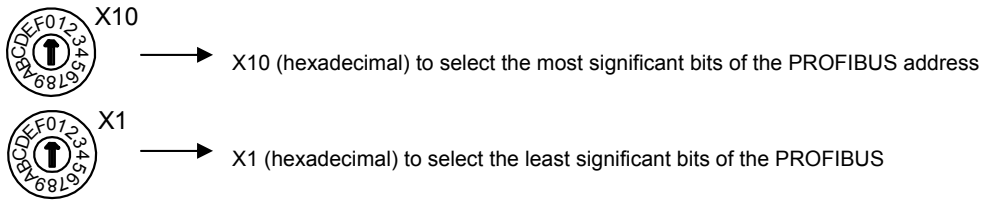
3.2.1. LEDs



LED #	Nom	Description
1	Power ON	Fix Green when the gateway is powered on.
2	Bus Fault	Fix Red if PROFIBUS is not in Data Exchange state.
3	Diagnostic	Fix Yellow in case of PROFIBUS Extended diagnostic.
4	RUN	Blinking at 2Hz when the gateway program works properly.
5	TxD MODBUS	Yellow when the gateway is sending a MODBUS message.
6	RxD MODBUS	Yellow when the gateway is receiving a MODBUS message.

3.2.2. Selection of the PROFIBUS address

Use the 2 rotary switches to select the PROFIBUS address of the AGILiGATE. (AGILiGATE default factory setting is with address 3_D):

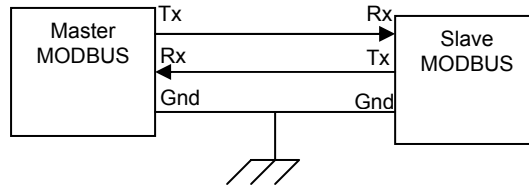


3.2.3. RS232/485 selection

Select the electrical interface mode of the MODBUS serial communication by using the DIP switch #1 (AGILiGATE factory setting is RS485):

Switch #1 : RS (RS232/485 mode)	
Position	Description
ON	RS232
OFF	RS485

RS232 Mode:

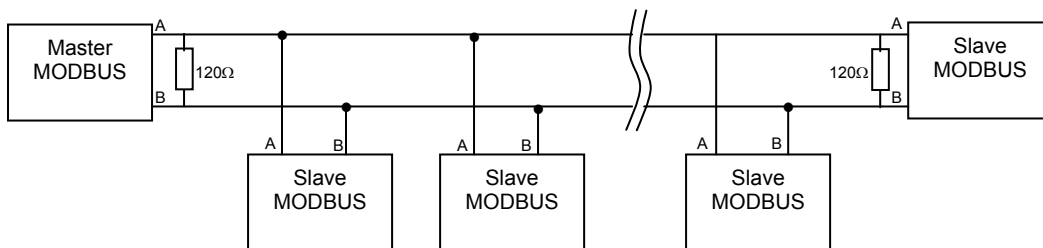


Draw 4 : MODBUS network in RS232 mode

This mode allows communication only between 2 MODBUS devices (point to point connection).

Important: With RS232 mode, please make sure that the RS485 termination resistor is disabled (DIP switch #2 OFF).

RS485 Mode:



Draw 5 : MODBUS RS485 network

This mode allows communications with multiple devices on a same bus. RS485 has better advantages compared to the RS232 (noise safe, network length up ...).

3.2.4. Bus termination resistor

RS485 requires a 120Ω resistor (bus terminator) connected at each side of the bus (see example in §1.4). To connect the built-in resistor, the DIP switch #2 must be at position ON. (AGILiGATE factory setting is termination disabled):

Switch #2 : RT (bus termination resistor)	
Position	Description
ON	120Ω termination connected
OFF	Disabled

4. CONFIGURATION OF THE GATEWAY

4.1. PROFIBUS address

The PROFIBUS address of the AGILiGATE gateway must be selected in the PROFIBUS network configuration tool. It must be set between 0 and $7E_H$ (126_D). AGILiGATE factory setting is at address 3_D .

The address can be changed with the 2 rotary switches on the front panel of the enclosure. During the power up, AGILiGATE reads the address from the rotary switches. If the address is lower than address 126_D , this address is taken into account by AGILiGATE. If the address from the rotary switches is upper than or equal to 126_D , then the address 126_D is taken into account.

If the address is changed while the gateway is already powered, you must recycle the power to have the new address taken into account.

4.2. Principle for configuring AGILiGATE

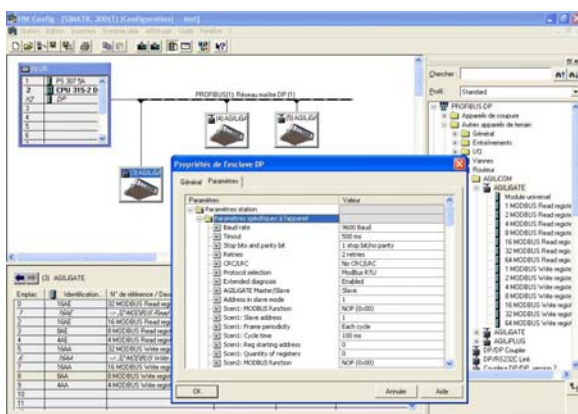
The configuration of the AGILiGATE PROFIBUS – MODBUS is done by using any PROFIBUS network configuration tool. It allows to:

- Setup the serial MODBUS port,
- Define if AGILiGATE is a MODBUS master or a MODBUS slave,
- Define with **scenarios**, the messages to generate for the data exchange between MODBUS registers and the PROFIBUS memory map.

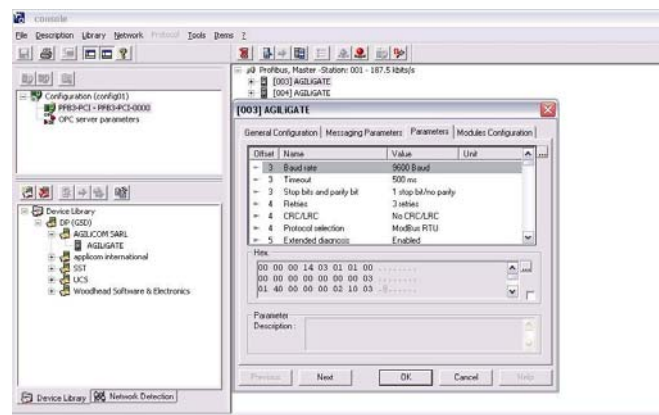
The MODBUS parameters are described in 155 bytes of the PROFIBUS message. The message is automatically sent by the PROFIBUS master during the initialization phase of the network.

Usually, PROFIBUS network configuration tools allow an easy access to configure parameters, thanks to user friendly list boxes and menus. Draws 6 and 7 are screen captures of the SIEMENS:"STEP 7 HW Config" and of the Woodhead:"Console" PROFIBUS configuration tools. The first window allows to enter the parameters. This window is automatically built up with the information coming from the device GSD file. For each parameter, a menu allows to select the appropriated value required by the application.

The GSD file allowing to use the AGILiGATE PROFIBUS is named: **AGIL0834.GSD**.



Draw 6: Enter parameters in STEP7 HW Config



Draw 7: Enter parameters in the Woodhead Console

4.3. Setup of the serial MODBUS

Bytes 11 to 14 describe the physical data layer of the MODBUS serial link. The parameters are detailed in table 2:

# byte # bits	Designation	Description																																				
Byte 1-7	Standard parameters																																					
Byte 8-10	DPV1 Status																																					
Byte 11																																						
B0-2	<i>Baud Rate</i>	<table border="1"> <thead> <tr> <th>B2</th> <th>B1</th> <th>B0</th> <th>Bauds</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>600</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>1200</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>2400</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>4800</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>9600</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>19200</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>38400</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>57600</td></tr> </tbody> </table>	B2	B1	B0	Bauds	0	0	0	600	0	0	1	1200	0	1	0	2400	0	1	1	4800	1	0	0	9600	1	0	1	19200	1	1	0	38400	1	1	1	57600
B2	B1	B0	Bauds																																			
0	0	0	600																																			
0	0	1	1200																																			
0	1	0	2400																																			
0	1	1	4800																																			
1	0	0	9600																																			
1	0	1	19200																																			
1	1	0	38400																																			
1	1	1	57600																																			
B3-4	<i>Timeout (ms) (Receive mode)</i>	<table border="1"> <thead> <tr> <th>B4</th> <th>B3</th> <th>timeout</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>10</td></tr> <tr><td>0</td><td>1</td><td>100</td></tr> <tr><td>1</td><td>0</td><td>500</td></tr> <tr><td>1</td><td>1</td><td>1000</td></tr> </tbody> </table>	B4	B3	timeout	0	0	10	0	1	100	1	0	500	1	1	1000																					
B4	B3	timeout																																				
0	0	10																																				
0	1	100																																				
1	0	500																																				
1	1	1000																																				
B5-7	<i>Stop bits and Parity bit</i>	<table border="1"> <thead> <tr> <th>B7</th> <th>B6</th> <th>B5</th> <th>Function</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>no parity / 1stop</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>no parity / 2stops</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>even parity / 1stop</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>Réservé</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>odd parity / 1stop</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>Reserved</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>Reserved</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>Reserved</td></tr> </tbody> </table>	B7	B6	B5	Function	0	0	0	no parity / 1stop	0	0	1	no parity / 2stops	0	1	0	even parity / 1stop	0	1	1	Réservé	1	0	0	odd parity / 1stop	1	0	1	Reserved	1	1	0	Reserved	1	1	1	Reserved
B7	B6	B5	Function																																			
0	0	0	no parity / 1stop																																			
0	0	1	no parity / 2stops																																			
0	1	0	even parity / 1stop																																			
0	1	1	Réservé																																			
1	0	0	odd parity / 1stop																																			
1	0	1	Reserved																																			
1	1	0	Reserved																																			
1	1	1	Reserved																																			
Octet 12																																						
B0-1	<i>Retries :</i> Number of consecutive retries before a time out detection.	<table border="1"> <thead> <tr> <th>B1</th> <th>B0</th> <th>Essais</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>0</td><td>2</td></tr> <tr><td>1</td><td>1</td><td>3</td></tr> </tbody> </table>	B1	B0	Essais	0	0	0	0	1	1	1	0	2	1	1	3																					
B1	B0	Essais																																				
0	0	0																																				
0	1	1																																				
1	0	2																																				
1	1	3																																				
B2	<i>CRC/LRC</i>	1 = CRC/LRC enabled 0 = CRC/LRC disabled																																				
B3-4	<i>Protocol selection</i>	<table border="1"> <thead> <tr> <th>B4</th> <th>B3</th> <th>Protocol</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>RTU</td></tr> <tr><td>0</td><td>1</td><td>ASCII</td></tr> <tr><td>1</td><td>0</td><td>Reserved</td></tr> <tr><td>1</td><td>1</td><td>Reserved</td></tr> </tbody> </table>	B4	B3	Protocol	0	0	RTU	0	1	ASCII	1	0	Reserved	1	1	Reserved																					
B4	B3	Protocol																																				
0	0	RTU																																				
0	1	ASCII																																				
1	0	Reserved																																				
1	1	Reserved																																				
B5-7	Reserved	Reserved																																				

Byte 13 B0	<i>Extended Diagnosis :</i> From PFB	0 activated 1 deactivated
B1-3	Reserved	Reserved
B4	<i>AGILiGATE</i> <i>Master/Slave</i>	0 = master 1 = slave
B5-7	Reserved	Reserved
Octet 14	<i>Address in slave</i> <i>mode</i>	1 to 247 included
Octet 15-21	Reserved	Reserved
Octet 22-161	List of 20 scenarios	

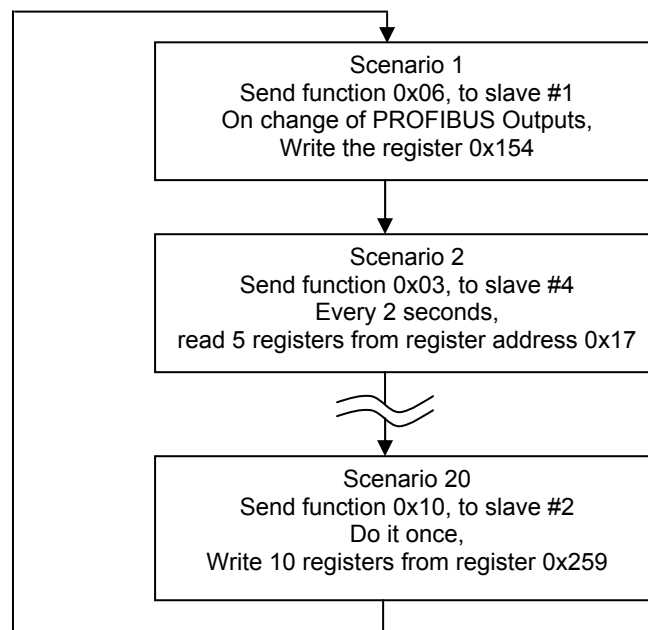
Table 2: Serial MODBUS parameter and possible values.

4.4. MODBUS Master mode: Definition of the scenarios

When AGILiGATE is configured as a MODBUS master, the MODBUS messages to be sent are described into a list of 20 **scenarios**. Each scenario describes:

- The MODBUS function (command) to send,
- The address of the device that receive the command,
- The event which will trigger the message,
- The address of the first register to read/write,
- The number of registers to read/write.

AGILiGATE engine does the polling of all defined scenarios, allowing the system to create and to send out all the appropriated MODBUS messages. This mechanism is described in the following example:



Draw 8: Scenarios polling mechanism

☞ The order of the scenarios doesn't define the order of the messages to be sent.

The 20 scenarios are coded into the bytes 22 to 161. 7 parameter bytes are enough to describe the n scenarios ($0 \leq n \leq 19$) :

Byte 22+n*7	<i>MODBUS Function</i>	0 to 255 included																																				
Byte 23+n*7 B0-3	<i>Slave Address</i>	0 to 247 included. 1 by default 0 → Broadcast																																				
B4-7	Reserved	Reserved																																				
Byte 24+n*7 B0-1	<i>Frame periodicity</i>	<table border="1"> <thead> <tr> <th>B1</th> <th>B0</th> <th>Trigger</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Each cycle</td> </tr> <tr> <td>0</td> <td>1</td> <td>on change</td> </tr> <tr> <td>1</td> <td>0</td> <td>once</td> </tr> <tr> <td>1</td> <td>1</td> <td>Reserved</td> </tr> </tbody> </table>	B1	B0	Trigger	0	0	Each cycle	0	1	on change	1	0	once	1	1	Reserved																					
B1	B0	Trigger																																				
0	0	Each cycle																																				
0	1	on change																																				
1	0	once																																				
1	1	Reserved																																				
B2-4	Reserved	Reserved																																				
B5-7	<i>Cycle time</i>	<table border="1"> <thead> <tr> <th>B7</th> <th>B6</th> <th>B5</th> <th>Timer</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>100 ms</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>500 ms</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>1s</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>5s</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>10s</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>30s</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>60s</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>Polling</td> </tr> </tbody> </table>	B7	B6	B5	Timer	0	0	0	100 ms	0	0	1	500 ms	0	1	0	1s	0	1	1	5s	1	0	0	10s	1	0	1	30s	1	1	0	60s	1	1	1	Polling
B7	B6	B5	Timer																																			
0	0	0	100 ms																																			
0	0	1	500 ms																																			
0	1	0	1s																																			
0	1	1	5s																																			
1	0	0	10s																																			
1	0	1	30s																																			
1	1	0	60s																																			
1	1	1	Polling																																			
Byte 25+n*7 to 26+n*7	@ <i>Reg MODBUS</i>	0 to 65535 included																																				
Byte 27+n*7 to 28+n*7	<i>Nb Registers</i>	0 to 50 included																																				

Table 3: Byte parameters to describe a scenario

- The parameter "Function" :

It defines the MODBUS function (command) to be sent. MODBUS functions supported by AGILiGATE are listed in appendix A.

- The parameter "Slave Address" :

It allows to select the bus address of the MODBUS device. It must be equal to 0 for a broadcast.

- The parameter "Frame periodicity" :

It selects the trigger mode for sending out a MODBUS message (Frame). A message can be sent:

- ➔ "at Each cycle" (on timer) : the period must be defined.
- ➔ "On change" : In case of a MODBUS write function, the message is sent only if the value to be sent has changed.
- ➔ "Once" : The message is sent once just after the reception of the PROFIBUS configuration message, as soon as AGILiGATE switches to Data Exchange mode.

- The parameter "Cycle time" :

It is only applicable if "Frame periodicity" = "Each cycle". Otherwise it is ignored. It allows to select the period for sending out the messages. This parameter must be selected according to the baud rate, according to the number of scenarios to be sent and the length of each MODBUS message (cf. paragraph time cycle).

If the value "polling" is selected, the message is sent as possible, as soon as a total cycle of scenarios sending is completed.

- The parameter "@ reg. MODBUS " :

It allows to select the address of the first MODBUS register to read/write.

☞ **Warning, it is important to differentiate the address of a MODBUS register with the number of MODBUS register. Actually, the register 1 is at MODBUS address 0x0000. In our case, the parameter "@ reg. MODBUS " corresponds to the MODBUS register address.**

- The parameter " Nb registers " :

Depending on the MODBUS function, it allows to select the number of registers to read/write or it can be not used.

☞ **It is possible to read up to 50 registers in ones. Accordingly, the byte $27+n*7$ must necessarily be 0 and the byte $28+n*7$ must be below or equal to 50.**

Note: Functions 0 et 1 are not MODBUS functions.

The function 0 "NOP" is the default function. The scenario having "MODBUS function" = NOP defines the end of the valid scenarios. All scenarios after this one are ignored.

The function 1 "Suite paramètres" allows to add to 2 additional parameters to a function. Actually, some MODBUS functions require 2 additional parameters to be correctly described. This function is not used in the current version of AGILiGATE PROFIBUS.

4.5. MODBUS slave mode

When AGILiGATE is configured as a MODBUS slave, the 20 scenarios are ignored (Bytes 15 to 161 are ignored. They must all be set to zero). In addition, the parameters "*Timeout*" and "*Retries*" are also ignored. The byte 14 "*Address in slave mode*" allows to define the MODBUS address of AGILiGATE.

5. MEMORY MAP

The size of the PROFIBUS Input/Output messages is chosen in the configuration tool of the PROFIBUS network. AGILiGATE is modular. It is then possible to add several modules in order to get a sufficient number of bytes in the PROFIBUS Input/Output messages..

5.1. MODBUS master mode

Upon reception of the parameters, AGILiGATE checks the received scenarios and determines the number of MODBUS registers to be read / written.

Upon reception of the configuration, AGILiGATE compares the size of allocated PROFIBUS Inputs/Outputs with the number of MODBUS registers to be transferred. The system enter in "data exchange" mode only if the memory space allocated in the configuration is sufficient.

A MODBUS register is a WORD (2 bytes) and it is possible to read up to 122 registers (75 by default). So the Input message must contain between 0 and 244 bytes. Also, the Output message must contain 0 and 244 bytes. However, the total number of MODBUS registers must not exceed 150, which corresponds to 300 inputs/outputs bytes on PROFIBUS.

The PROFIBUS data are stored in the order defined by the scenarios, from offset 0 of the PROFIBUS input/output messages.

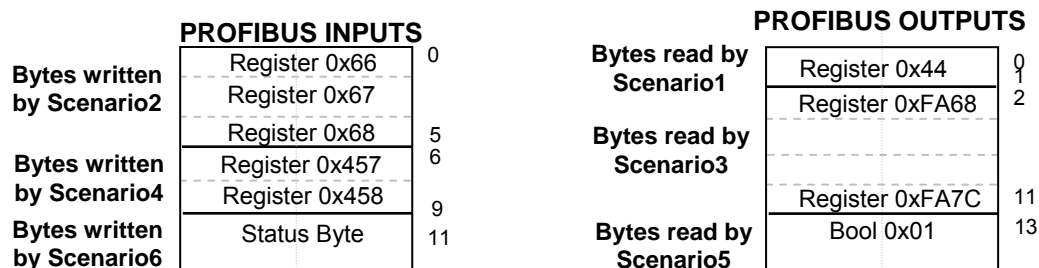
Example : If the following parameters were defined:

- Scenario 1: Function 0x06. Write register 0x44 (address 0x43)
- Scenario 2: Function 0x03. Read 3 registers from register 0x66 (address 0x65)
- Scenario 3: Function 0x10. Write 5 registers from register 0xFA68 (address 0xFA67)
- Scenario 4: Function 0x04. Read 2 registers from register 0x457 (address 0x456)
- Scenario 5: Function 0x05. Write 1 register (a Boolean with value 0x0000 or 0xFF00) from register 0x01 (address 0x00)
- Scenario 6: Function 0x07. Read Status register. This one is a byte and doesn't have any address.
- Scenario 7-20: Function 0x00. NOP

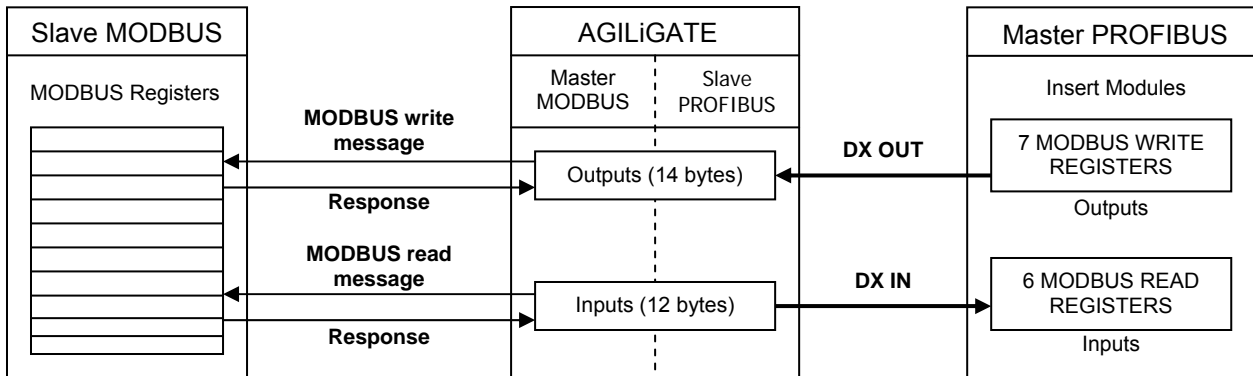
By choosing the following modules in the PROFIBUS tool, the configuration becomes valid:

- Module "4 MODBUS Read registers"
- + Module "2 MODBUS Read registers"
- + Module "4 MODBUS Write registers"
- + Module "2 MODBUS Write registers"
- + Module "1 MODBUS Write register"

The PROFIBUS memory map is as follow:



Draw 9 : Memory Map of the Slave PROFIBUS, according to the defined scenarios



Draw 10: Principle for operating in master MODBUS

5.2. Slave MODBUS mode

When AGILiGATE is in MODBUS slave mode, the size of the PROFIBUS memory map only depends on the configuration. The MODBUS addressing map is directly assigned to the PROFIBUS memory map.

A MODBUS register uses 2 bytes in the PROFIBUS message, the PROFIBUS memory map having a maximum of 244 input bytes and 244 output bytes, the MODBUS slave has a maximum of 122 registers for input and 122 registers for output. However, the maximum number of registers to be exchanged must not exceed 150, representing 300 input/output bytes on PROFIBUS. The number of accessible MODBUS registers is defined by the PROFIBUS configuration.

Read registers are accessible at MODBUS addresses 0 to 121.

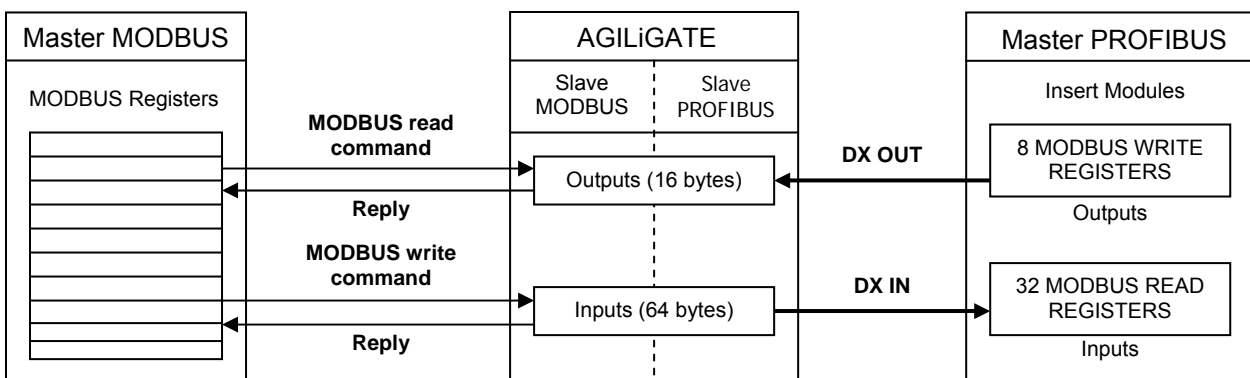
Write registers are accessible at MODBUS addresses 0 to 121.

The Status register is accessible at MODBUS address 0 (Most significant byte only).

Example : The size of PROFIBUS input message is 64 bytes. The size of the PROFIBUS output message is 16 bytes. Then AGILiGATE has 32 write registers, accessible at MODBUS address 0 to 31, and 8 read registers from MODBUS address 0 to 7.

☞ **For AGILiGATE, the MODBUS functions 3 et 4 are identical. They address the same memory map. Also, the memory map used by the function 7 is the same as functions 3 et 4. The Status data must be written in the most significant byte of the register 0 (byte 0).**

If the MODBUS slave receives a command asking for non authorized action, an exception message is sent by AGILiGATE. These exception messages are detailed in the appendix A.



Draw 11: Principle for operating in slave MODBUS

6. CYCLE TIME OF AGILiGATE

MODBUS Master mode

When AGILiGATE is configured as MODBUS master, a parameter "Cycle time" is defined for each scenario. This parameter specifies the requested period for sending messages.

The cycle time is guaranteed only if it has been correctly adjusted, and without any timeout.

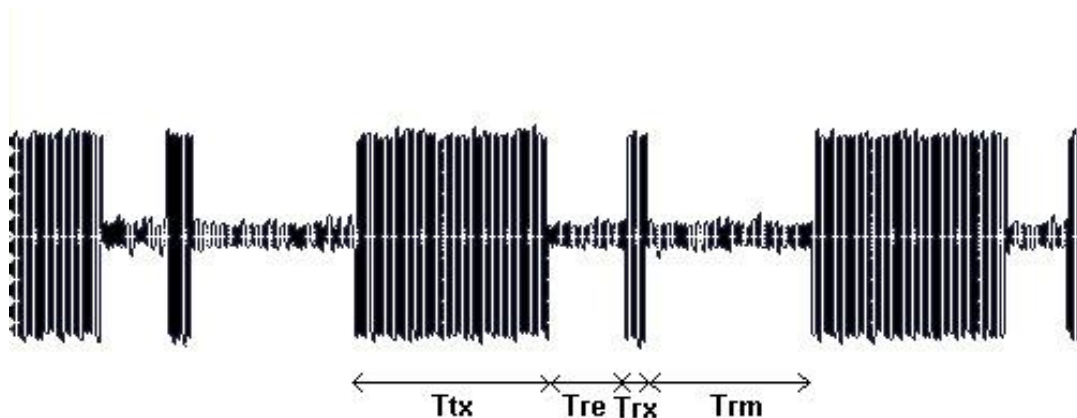
Example 1: If the parameter "Timeout" = 1000 ms, AGILiGATE is going to wait 1000 ms before detecting a timeout. The process of scenarios polling is then blocked during the timeout (here 1000 ms). If the next scenarios in the list have a cycle time of 100ms, the time can't be respected.

Example 2: The serial link is configured at 600 bauds, with 2 scenarios to be sent out (function 6, with a time cycle of 100ms). Each scenario sends an 8 bytes message, with an expected response of 8 bytes. So the send/receive time is 420ms. In addition of this time, we must add the response time of the Slave between the reception of the MODBUS command and the sending of the response. Finally, the time elapsed in each scenario is much more close to 500ms than 100ms. In this case the time cycle can't be respected.

For each scenario, it is recommended to calculate the communication time. By summing the communication of each scenario, we determine the minimum value of the cycle time to be applied to a scenario.

Generally, the following rule must be respected:

$$T_{comx} = T_{Tx} + T_{Re} + T_{Rx} + T_{Rm}$$



Choose: $T_{cy} > (T_{com1} + \dots + T_{com20})$

where:

T_{comx}	= time of communication for the scenario x
T_{Tx}	= transmit time for the command = (11 x Number of bytes to transmit) / Baud rate
T_{Rx}	= transmit time for the response = (11 x Baud rate) / Number of bytes to transmit
T_{Re}	= Response time of the Slave MODBUS
T_{Rm}	= Treatment time of the master AGILiGATE
T_{cy}	= Cycle time

☞ Each scenario has its own cycle time. The order that the messages are sent is not necessarily the order of the scenarios.

MODBUS Slave mode

The response time of AGILiGATE depends on the baud rate.

baud (bits/s)	RTU	ASCII
600	55	63
1200	30	40
2400	20	28
4800	13	22
9600	10	18
19200	10	17
38400	8	16
57600	9	16

Table 4: AGILiGATE response time as a slave (ms), function MODBUS 10h, 50 registers

baud (bits/s)	RTU	ASCII
600	51	55
1200	27	31
2400	17	20
4800	10	13
9600	9	11
19200	8	10
38400	6	9
57600	6	8

Table 5: AGILiGATE response time as a slave (ms), function MODBUS 10h, 5 registers

7. DIAGNOSTIC AND HELP FOR SETUP

7.1. Extended PROFIBUS Diagnostic

In addition to the 6 bytes for standard diagnostic, AGILiGATE manages 27 bytes of extended diagnostic. They allow to inform the PROFIBUS master on errors occurring at the slave level.

The extended diagnostic is sent just after the standard diagnostic. It is organized as follows:

- Bytes 8-14: 56 general error bits
 - Parameter error
 - Configuration error
 - Error on MODBUS reception
 - Timeout on MODBUS exchange
- Bytes 15-34 : 1 byte per scenario to report the error.

bytes 1 - 6						byte 7	bytes 8 -14	bytes 15 - 34			
Standard Diagnostic						Extended Diagnostic Length.	General errors	Error scénario1	Error scénario20
x	x	x	x	0x08	0x34	0x1C (28d)	56 bits	X	x	x	X

☞ **The meaning of each error is explained in annexe B. In normal operating mode, all bytes for extended diagnostic have null value (0).**

Example: The following error was detected in the scenario 5: The parameter "Cycle time" is impossible. The byte 19 is then set to value 108:

Err108: Parameter (Cycle time) impossible.

7.2. Using the Diagnostic serial interface

Information concerning the configuration, the parameters and the status of AGILiGATE are sent via the diagnostic serial port. These data can be monitored by any software able to manage ASCII messages sent via the serial port. In our example, the software is "Terminal.exe".

The connection of AGILiGATE to a PC is done as follows:

DB9 PC side	AGILiGATE connector
3 (Tx)	1 (Rx)
2 (Rx)	2 (Tx)
5 (gnd)	3 (gnd)

The diagnostic serial interface settings are:

- 9600 bps
- 1 start bit
- 8 bits data
- None
- 1 stop bit

The diagnostic serial interface can be used to facilitate the setup of AGILiGATE during the start-up. All the messages are listed in **annexe B**.

By sending the character 'e' or 'E' via the diagnostic serial interface, the gateway replies sending the header block and the PROFIBUS configuration.

At power on, the gateway sends out the following header:

```

////////// AGILiGATE PROFIBUS-MODBUS V1.3 //////////
18:33:16, Mar 20 2006

PROFIBUS address: 0x03 = 3
    
```

The parameters of the MODBUS serial interface are shown during the connection of the gateway to the PROFIBUS network. Then an array shows all parameters related to the scenarios. The following example shows 4 configured scenarios:

```

RS232, MASTER, 9600bps
1start, 1stop, no parity
No CRC/LRC check
MODBUS RTU
Ext. diag enabled, Retries B4 ext. diag = 02
    
```

scenario number	slave address	Modbus function	frame trigger	cycle time	register address	number of registers
1	1	3	cyclic	100 ms	0x0000	0x0032
2	1	16	cyclic	100 ms	0x0000	0x0023
3	1	3	cyclic	100 ms	0x0032	0x0032
4	1	3	cyclic	100 ms	0x0064	0x000F

```

PROFIBUS input bytes required for DX= 230
PROFIBUS output bytes required for DX= 70
    
```

If the MODBUS serial interface is properly connected, no more messages are shown. In the opposite case, a message shows the current error. In the following example, the MODBUS slaves defined into the scenarios 1, 2, 3 and 4 don't reply (timeout):

```

Err20 : MODBUS Timeout
Scenario 3, Error 118: Timeout after retries

Scenario 4, Error 118: Timeout after retries

Scenario 1, Error 118: Timeout after retries

Scenario 2, Error 118: Timeout after retries
    
```

Then, as soon as the communication with the slaves is back, the following message appears:

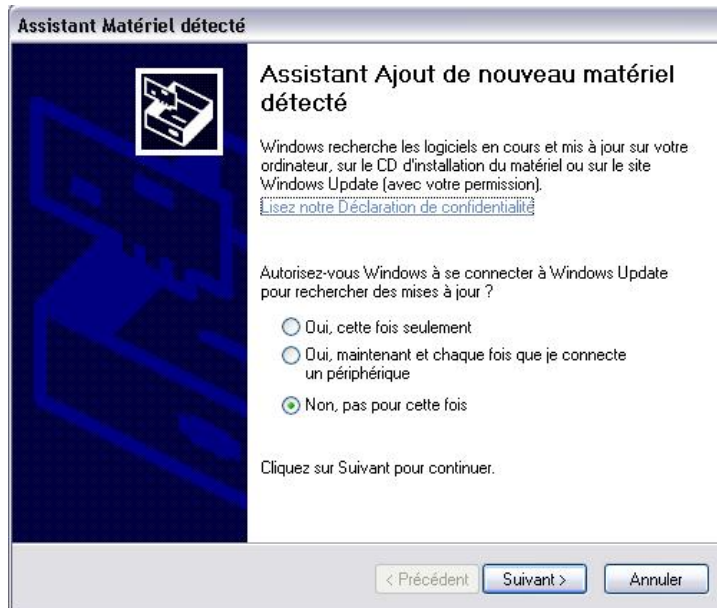
Back to normal operation

7.3. Using the USB diagnostic

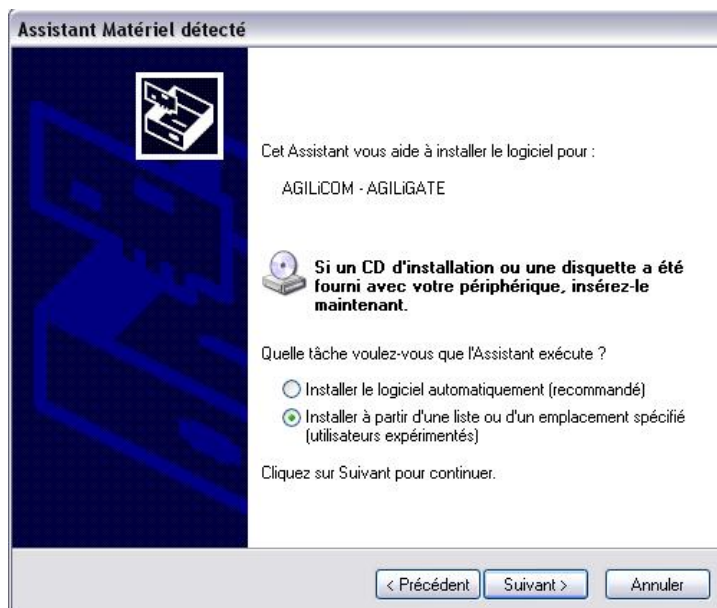
The USB diagnostic port is optional on AGILiGATE (AG-P012). When the driver is installed, a virtual COM port appears under Windows as soon as AGILiGATE is connected on the USB port. The operating mode and the setup are strictly identical to the Diagnostic serial interface.

To install the USB drivers for AGILiGATE :

- Plug AGILiGATE in the USB port of the PC. Windows detects a new device and the following window appears :



Check " **NO...** " then click on « next ».



Then select " **Install from a list...** " and click on " next ".

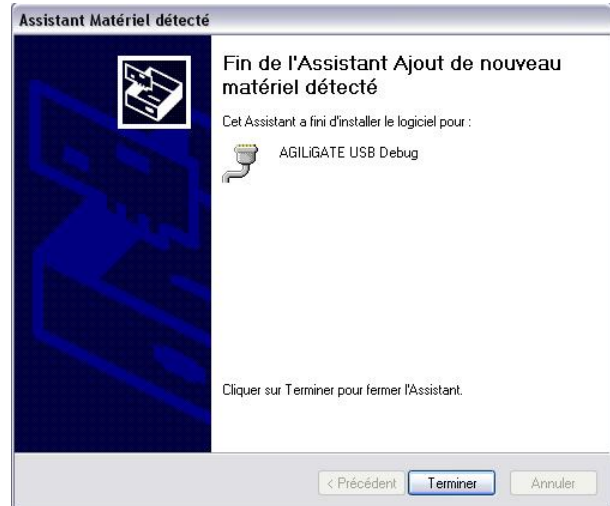


Check " **Inclure cet emplacement...** " then click on " browse ". Select the directory including the drivers, this one is located on the CD ROM provided with the AGILiGATE. The path appears in the list box. Click on " Next ".



Windows installs the drivers for the USB hardware. Click on " Finish " at the end of the installation.

- When the installation is completed, Windows detects a new peripheral and the window " Add new hardware Assistant" appears again. This new hardware corresponds to the virtual COM port. It is then necessary to execute again the procedure described above, keeping the path of the directory which contains the drivers.



Windows installs the drivers for the virtual COM port. Click on " Finish " at the end of the installation.

- The installation of the drivers is now completed. A new Serial port appears in the Device Manager. This one is accessible with a right click on "Desktop/Properties", then under the tab "Hardware" click on "Device Settings".



Windows automatically assigns a COM port ID depending on the current configuration of the other ports (here COM6). This can then change depending on your PC configuration. The user can continue to use his console as usual, but specifying the COM port ID assigned to AGILiGATE.

APPENDICES

APPENDICES	25
APPENDIX A: FORMAT OF MODBUS MESSAGES.....	26
APPENDIX B: LIST OF ERRORS SENT BY AGILIGATE.....	36

APPENDIX A: Format of MODBUS messages

The MODBUS functions 0x03, 0x04, 0x05, 0x06, 0x07, 0x10, 0x17 are supported by AGILiGATE. The format of each function is detailed using an example for MODBUS RTU and ASCII. The bytes for CRC error check (MODBUS RTU) or LRC (MODBUS ASCII) are mandatory. However, they are not treated if the parameter "CRC/LRC" is disabled (set to "No CRC/LRC").

In MODBUS RTU, data are 8 bit coded. In MODBUS ASCII, data are 7 bit coded.

Remind : It is important to differentiate the address of the MODBUS register and the MODBUS register # (number). Actually, the register 1 is at MODBUS physical address 0x0000. In a MODBUS message, it is the address of the register sent. When setting up a scenario, it is also the MODBUS register address that must be configured.

Function 3 (0x03)

This function allows to read MODBUS registers (Holding Registers). The broadcast mode is not supported. The number of registers to be read in a same message is limited to 50.

Command format:

Name of the field	Value to transmit	Bytes sent if RTU	Bytes sent if ASCII	
			Character	ASCII characters
Message header	-	-	":"	0x3A
Slave address	0x39	0x39	"39"	0x33, 0x39
Function code	0x03	0x03	"03"	0x30, 0x33
Most significant byte for address 1 st register	0x00	0x00	"00"	0x30, 0x30
Less significant byte for address 1st register	0x22	0x22	"22"	0x32, 0x32
Most significant byte for number of registers	0x00	0x00	"00"	0x30, 0x30
Less significant byte for number of registers	0x02	0x02	"02"	0x30, 0x32
Error check (CRC / LRC)	-	0x60 0xB9	"A0"	0x41, 0x30
End of message	-	-	CR LF	0xD, 0xA

Response format:

Name of the field	Value to be received	Bytes received if RTU	Bytes received if ASCII	
			Character	ASCII characters
Message header	-	-	":"	0x3A
Slave address	0x39	0x39	"39"	0x33, 0x39
Function code	0x03	0x03	"03"	0x30, 0x33
Number of data bytes	0x04	0x04	"04"	0x30, 0x34
MSB of the value 1st register	0x68	0x68	"68"	0x36, 0x38
LSB of the value 1st register	0x31	0x31	"31"	0x36, 0x38
MSB for the value 2 nd register	0x47	0x47	"47"	0x36, 0x38
LSB for the value 2 nd register	0x59	0x59	"59"	0x36, 0x38
Error check (CRC / LRC)	-	0xFD 0x95	"87"	0x38, 0x37
End of message	-	-	CR LF	0xD, 0xA

Function 4 (0x04)

This function allows to read the MODBUS registers (Input Registers). The broadcast is not supported. The number of registers to be read in the same message is limited to 50.

Command format:

Name of the field	Value to transmit	Bytes sent if RTU	Bytes sent if ASCII	
			character	ASCII characters
Message header	-	-	":"	0x3A
Slave address	0x39	0x39	"39"	0x33, 0x39
Function code	0x04	0x04	"04"	0x30, 0x34
MSB for address 1 st register	0x00	0x00	"00"	0x30, 0x30
LSB for address 1 ^{er} register	0x22	0x22	"22"	0x32, 0x32
MSB for the number of registers	0x00	0x00	"00"	0x30, 0x30
LSB for the number of registers	0x03	0x03	"03"	0x30, 0x33
Error check (CRC / LRC)	-	0x14 0xB9	"9E"	0x39, 0x45
End of message	-	-	CR LF	0xD, 0xA

Response format:

Name of the field	Value to be received	Bytes received if RTU	Bytes received if ASCII	
			Character	ASCII characters
Frame header	-	-	":"	0x3A
Slave address	0x39	0x39	"39"	0x33, 0x39
Function code	0x04	0x04	"04"	0x30, 0x34
Number of data bytes	0x06	0x06	"06"	0x30, 0x36
MSB of value 1 st register	0x68	0x68	"68"	0x36, 0x38
LSB of value 1 st register	0x31	0x31	"31"	0x33, 0x31
MSB of value 2 nd register	0x47	0x47	"47"	0x34, 0x37
LSB of value 2 nd register	0x59	0x59	"59"	0x35, 0x39
MSB of value 32 nd register	0x00	0x00	"00"	0x30, 0x30
MSB of value 32 nd register	0x00	0x00	"00"	0x30, 0x30
Error check (CRC / LRC)	-	0xE2 0xD9	"84"	0x38, 0x34
End of Message	-	-	CR LF	0xD, 0xA

Function 5 (0x05)

This function allows to write a Boolean (coil) having ON or OFF status. The broadcast is supported. This boolean can have the value 0x0000 (OFF) or 0xFF00 (ON).

Command format:

Name of the field	Value to transmit	Bytes sent if RTU	Bytes sent if ASCII	
			character	ASCII characters
Message header	-	-	":"	0x3A
Slave address	0x01	0x01	"01"	0x30, 0x31
Function code	0x05	0x05	"05"	0x30, 0x35
MSB for address of the register	0x00	0x00	"00"	0x30, 0x30
LSB for address of the register	0x22	0x22	"22"	0x32, 0x32
MSB for value of the register	0xFF	0xFF	"FF"	0x46, 0x46
LSB for value of the register	0x00	0x00	"00"	0x30, 0x30
Error check (CRC / LRC)	-	0x2C 0x30	"D9"	0x44,0x39
End of message	-	-	CR LF	0xD, 0xA

Response format:

Name of the field	Value to be received	Bytes received if RTU	Bytes received if ASCII	
			character	ASCII characters
Message header	-	-	":"	0x3A
Slave address	0x01	0x01	"01"	0x30, 0x31
Function code	0x05	0x05	"05"	0x30, 0x35
MSB for the address of the register	0x00	0x00	"00"	0x30, 0x30
MSB for the address of the register	0x22	0x22	"22"	0x32, 0x32
MSB for value of the register	0xFF	0xFF	"FF"	0x46, 0x46
LSB for value of the register	0x00	0x00	"00"	0x30, 0x30
Error check (CRC / LRC)	-	0x2C 0x30	"D9"	0x44, 0x39
End of message	-	-	CR LF	0xD, 0xA

Function 6 (0x06)

This function allows to write one MODBUS register (Holding Registers). The broadcast is supported.

Command format:

Name of the field	Value to transmit	Bytes sent if RTU	Bytes sent if ASCII	
			character	ASCII characters
Message header	-	-	":"	0x3A
Slave address	0x39	0x39	"39"	0x33, 0x39
Function code	0x06	0x06	"06"	0x30, 0x36
MSB for the address of the register	0x00	0x00	"00"	0x30, 0x30
LSB for the address of the register	0x22	0x22	"22"	0x32, 0x32
MSB for value of the register	0x00	0x00	"00"	0x30, 0x30
LSB for value of the register	0x56	0x56	"56"	0x35, 0x36
Error check (CRC / LRC)	-	0xAD 0x46	"49"	0x34, 0x39
End of message	-	-	CR LF	0xD, 0xA

Response format:

Name of the field	Value to be received	Bytes received if RTU	Bytes received if ASCII	
			character	ASCII characters
Message header	-	-	":"	0x3A
Slave address	0x39	0x03	"39"	0x33, 0x39
Function code	0x06	0x06	"06"	0x30, 0x36
MSB for the address of the register 0x0	0x00	0x00	"00"	0x30, 0x30
LSB for the address of the register 0x0	0x22	0x22	"22"	0x32, 0x32
MSB for value of the register	0x00	0x00	"00"	0x30, 0x30
LSB for value of the register	0x56	0x56	"56"	0x35, 0x36
Error check (CRC / LRC)	-	0xAD 0x46	"49"	0x34, 0x39
End of message	-	-	CR LF	0xD, 0xA

Function 7 (0x07)

This function allows to read a Status byte. This register is not located at a particular address as it is unique. The broadcast is not supported.

Command format:

Name of the field	Value to transmit	Bytes sent if RTU	Bytes sent if ASCII	
			character	ASCII characters
Message header	-	-	":"	0x3A
Slave address	0x39	0x39	"39"	0x33, 0x39
Function code	0x07	0x07	"07"	0x30, 0x37
Error check (CRC / LRC)	-	0x52 0x22	"C0"	0x43, 0x30
End of message	-	-	CR LF	0xD, 0xA

Response format:

Name of the field	Value to be received	Bytes received if RTU	Bytes received if ASCII	
			character	ASCII characters
Message header	-	-	":"	0x3A
Slave address	0x39	0x39	"39"	0x33, 0x39
Function code	0x07	0x07	"07"	0x30, 0x37
Value of the status	0x14	0x14	"14"	0x31, 0x34
Error check (CRC / LRC)	-	0xA3 0xF2	"AC"	0x41, 0x43
End of message	-	-	CR LF	0xD, 0xA

Function 16 (0x10)

This function allows to write MODBUS registers (Holding Registers). The broadcast is supported. The number of registers to be written in the same command is limited to 50.

Command format:

Name of the field	Value to transmit	Bytes sent if RTU	Bytes sent if ASCII	
			character	ASCII characters
Message header	-	-	":"	0x3A
Slave address	0x39	0x39	"39"	0x33, 0x39
Function code	0x10	0x10	"10"	0x31, 0x30
MSB for address of the 1 st register	0x00	0x00	"00"	0x30, 0x30
LSB for address of the 1 st register	0x22	0x22	"22"	0x32, 0x32
MSB for number of registers	0x00	0x00	"00"	0x30, 0x30
LSB for number of registers	0x02	0x02	"02"	0x30, 0x32
Number of data bytes	0x04	0x04	"04"	0x30, 0x34
MSB for value of register 0x52	0x00	0x00	"00"	0x30, 0x30
LSB for value of register 0x52	0x56	0x56	"56"	0x35, 0x36
MSB for value of register 0x53	0x00	0x00	"00"	0x30, 0x30
LSB for value of register 0x53	0x57	0x57	"57"	0x35, 0x37
Error check (CRC / LRC)	-	0x04 0xE0	"E2"	0x45, 0x32
End of message	-	-	CR LF	0xD, 0xA

Response format:

Name of the field	Value to be received	Bytes received if RTU	Bytes received if ASCII	
			character	ASCII characters
Message header	-	-	":"	0x3A
Slave address	0x39	0x39	"39"	0x33, 0x39
Function code	0x10	0x10	"10"	0x31, 0x30
MSB for address of the 1 st register	0x00	0x00	"00"	0x30, 0x30
LSB for address of the 1 st register	0x22	0x22	"22"	0x32, 0x32
MSB for number of registers	0x00	0x00	"00"	0x30, 0x30
LSB for number of registers	0x02	0x02	"02"	0x30, 0x32
Error check (CRC / LRC)	-	0xE5, 0x7A	"93"	0x39, 0x33
End of message	-	-	CR LF	0xD, 0xA

Function 23 (0x17) in slave mode

This function allows to read and write with a single command the MODBUS registers (Holding Registers). The broadcast is not supported. The number of registers to be read is limited to 50. The number of registers to be written is limited to 50. This function is only available in Slave mode.

Command format:

Name of the field	Value to transmit	Bytes sent if RTU	Byte sent if ASCII	
			character	ASCII characters
Message header	-	-	":"	0x3A
Slave address	0x39	0x39	"39"	0x33, 0x39
Function code	0x17	0x17	"17"	0x31, 0x37
MSB for address of the 1 st register in reading	0x00	0x00	"00"	0x30, 0x30
LSB for address of the 1 st register in read	0x22	0x22	"22"	0x32, 0x32
MSB for number of registers in read	0x00	0x00	"00"	0x30, 0x30
LSB for number of registers in read	0x02	0x02	"02"	0x30, 0x32
MSB for address of the 1 st register in write	0x00	0x00	"00"	0x30, 0x30
LSB for address of the 1 st register in write	0x56	0x56	"56"	0x35, 0x36
MSB for number of registers in write	0x00	0x00	"00"	0x30, 0x30
LSB for number of registers in write	0x57	0x57	"57"	0x35, 0x37
Number of data bytes	0x04	0x04	"04"	0x30, 0x34
MSB for value of register 0x52	0x00	0x00	"00"	0x30, 0x30
LSB for value of register 0x52	0x56	0x56	"56"	0x35, 0x36
MSB for value of register 0x53	0x00	0x00	"00"	0x30, 0x30
LSB for value of register 0x53	0x57	0x57	"57"	0x35, 0x37
Error check (CRC / LRC)	-	0x04 0xE0	"E2"	0x45, 0x32
End of message	-	-	CR LF	0xD, 0xA

Response format:

Name of the field	Value to be received	Bytes received if RTU	Bytes received if ASCII	
			Character	ASCII characters
Message header	-	-	":"	0x3A
Slave address	0x39	0x39	"39"	0x33, 0x39
Function code	0x17	0x17	"17"	0x31, 0x37
Number of data bytes	0x04	0x04	"04"	0x30, 0x34
MSB for value of the 1 st register	0x68	0x68	"68"	0x36, 0x38
LSB for value of the 1 st register	0x31	0x31	"31"	0x33, 0x31
MSB for value of the 2 nd register	0x47	0x47	"47"	0x34, 0x37
LSB for value of the 2 nd register	0x59	0x59	"59"	0x35, 0x39
Error check (CRC / LRC)	-	0xFD 0x95	"87"	0x38, 0x37

Exception Message

An exception message is built by the MODBUS slave each time the master asks to perform a non authorized action/command. AGILiGATE PROFIBUS MODBUS manages the following exceptions :

Exception code	Name	Description
01	ILLEGAL FUNCTION	Sent by the slave each time the master sends an unknown command. AGILiGATE MODBUS only support the functions (commands) 3, 4, 5, 6, 7, 16, 23.
02	ILLEGAL DATA ADDRESS	<p>Sent by the slave each time the master tries to access to a non accessible register. For example, AGILiGATE has got a configuration with 10 read registers (address 0 to 9), and 10 write registers (address 0 to 9). An error is returned when the master tries to access to register 20.</p> <p>This error also appears with the function 3 when for example the master tries to read 20 registers from register 1 (address 0).</p>
03	ILLEGAL DATA VALUE	This exception is sent if the master tries to write a data which is non acceptable for the slave. For example, with the function 5, the 2 only possible values are either 0x0000 or 0xFF00. For any other value, the exception code 3 is returned.

Format of the exception message:

This exception message is an example received in response to the function 6. The exception happened because the master tried to access to a non authorized register.

The MODBUS function codes are coded on 7 bits (code 1 to 127). The 8th bit is reserved, it is used to signal an exception message when it is set to 1.

Name of the field	Value to be received	Bytes received if RTU	Bytes received if ASCII	
			character	ASCII character
Message header	-	-	":"	0x3A
Slave address	0x03	0x03	"03"	0x30, 0x33
Function code + 0x80	0x86	0x86	"86"	0x38, 0x36
Exception code	0x02	0x02	"02"	0x30, 0x32
Error check (CRC / LRC)	-	0x31 0x62	"75"	0x37, 0x35
End of message	-	-	CR LF	0xD, 0xA

APPENDIX B: List of errors sent by AGILiGATE

Description of general errors:

The general errors are located in the bytes 8 à 14 of the extended diagnostic (1 bit per error).

Byte	Byte 8		Byte 9		Byte 10		Byte 11		Byte 12		Byte 13		Byte 14	
Bit	7	0	7	0	7	0	7	0	7	0	7	0	7	0
Error	1	8	9	16	17	24	25	32	33	40	41	48	49	56

# err	Error Message	Explanation
1	Bytes 0-2 must be empty	Error on parameter message. The first 3 bytes must be set to 0 (mandatory).
2	Baud rate impossible	Error on parameter message. The value of the parameter "Baud rate" is wrong.
3	Timeout impossible	Error on parameter message. The value of the parameter "Timeout" is wrong.
4	Stop bit impossible	Error on parameter message. The values of the parameter "Stop bit" is wrong.
5	Parity bit impossible	Error on parameter message. The value of the parameter "Parity bit" is wrong.
6	Retries impossible	Error on parameter message. The value of the parameter "Retries" is wrong.
7	CRC check impossible	Error on parameter message. The value of the parameter "CRC check" is wrong.
8	Protocol impossible	Error on parameter message. The value of the parameter "Protocol" is wrong.
13	Slave address = 0	Error on parameter message. The value of the parameter "Address in slave mode" is wrong. The address of a MODBUS slave can't be 0 (reserved for broadcast).
14	Slave address > 247	Error on parameter message. The value of the parameter "Address in slave mode" is wrong. The address of a MODBUS slave can't be higher than 247.
15	Module cfg error	Error on configuration message. One or several modules are unknown.
16	PFB message > DxxBufLen	Error on configuration message. The size of inputs can't be higher than 244 bytes. The size of outputs can't be higher than 244 bytes.
17	PFB message < Required	Error on configuration message. The received configuration doesn't allow to map all MODBUS registers into the PROFIBUS memory map. You must add modules with the PROFIBUS configuration tool.
18	CFG message is too long	Error on configuration message. The number of configuration bytes is greater than 16. You must delete modules with the PROFIBUS configuration tool.
19	Error MODBUS reception	Error on receipt of the last MODBUS message. This general error is detailed in the byte of the scenario on which the error occurred (cf. error 111-116).
20	MODBUS Timeout	Timeout on the MODBUS network. One slave didn't reply after the number of retries ("n Retries").
21	Error 2 nd stop bit	Un character was received on the MODBUS link but it didn't have the expected 2 nd stop bit. Despite of AGILiGATE is set up with 2 stop bits.
22	Parity error	One character was received on the MODBUS link but the parity bit is wrong. AGILiGATE is set up to check parity.

# err	Error Message	Explanation
23	Received message non ASCII	AGILiGATE is in ASCII but the received message doesn't start with the character ':'.
24	Too many chars received	More than 255 characters received in the same message on the serial link. This can show a crash of the remote device.
25	"passe trame" reception err	The number of bytes received in response to a scenario "passe trame" is not the one expected. This general error is detailed in the byte of the scenario on which the error occurred (cf. error 111-116).
28	Exception message	A MODBUS exception message was received.

Description of errors specific to the scenarios:

The error specific to the scenarios are located in the bytes 15 to 34 of the extended diagnostic (1 byte per scenario).

# err	Error Message	Explanation
100 (0x64)	# READ regs < 1	Error on scenario parameter message. The number of MODBUS registers to be read must be >= 1.
101 (0x65)	# READ regs > 50	Error on scenario parameter message. The number of MODBUS registers to be read must be <= 50.
102 (0x66)	Frame period error	Error on scenario parameter message. The MODBUS functions 3, 4 et 7 can't be sent on change. The field "On Change" is only allowed with the writing functions 5, 6 et 16.
103 (0x67)	# WRITE regs < 1	Error on scenario parameter message. The number of MODBUS registers to be written must be >= 1.
104 (0x68)	# WRITE regs > 50	Error on scenario parameter message. The number of MODBUS registers to be written must be <= 50.
105 (0x69)	MODBUS Function error	Error on scenario parameter message. The value of the parameter "MODBUS Function" is wrong.
106 (0x6A)	Slave address error	Error on scenario parameter message. The value of the parameter "Slave address" is wrong.
107 (0x6B)	Frame periodicity error	Error on scenario parameter message. The value of the parameter "Frame periodicity" is wrong.
108 (0x6C)	Cycle time error	Error on scenario parameter message. The value of the parameter "Cycle time" is wrong.
109 (0x6D)	#WRITE regs > DoutBufLen	Error on scenario parameter message. The total number of MODBUS registers to be written must be <= total number of "MODBUS WRITE REGISTERS" insert. You must delete this scenario as well as the following or you must write less registers in the previous scenarios.
110 (0x6E)	# READ regs > DinBufLen	Error on scenario parameter message. The number of MODBUS registers to be read must be <= total number of MODBUS "MODBUS READ REGISTERS". You must delete this scenario as well as the followings or read less registers in the previous scenarios.
111 (0x6F)	CRC error	Error when receiving the last MODBUS message. The received CRC in the message doesn't match the calculated CRC. The message is corrupted and then not taken into account.
112 (0x70)	MODBUS address error	Error when receiving the last MODBUS message. The received message was send by a non expected slave.
113 (0x71)	MODBUS function error	Error when receiving the last MODBUS message. The receive message doesn't match the expected function MODBUS.

# err	Error Message	Explanation
114 (0x72)	Bytes number error	Error when receiving the last MODBUS message. The message received doesn't have the appropriated number of data bytes.
115 (0x73)	Response length error	Error when receiving the last MODBUS message. The message received doesn't have the appropriate number of bytes.
116 (0x74)	Register value error	Error when receiving the last MODBUS message. The message received doesn't have the expected register value.
117 (0x75)	Timeout	The slave didn't respond to the MODBUS command even after several consecutive sent of the same message (parameter "retries"). This error is acknowledged as soon as the slave has responded.
118 (0x76)	Timeout after retries	The slave didn't respond to the MODBUS acyclic command even after several consecutive sent of the same message (parameter "retries"). This error is acknowledged as soon as the slave has responded.
119 (0x77)	Broadcast impossible	Broadcast is not available with the read functions of MODBUS. It is possible with the write functions.
120 (0x78)	"Passe frame" input byte number >100	Error on scenario parameter message. The function "passe frame" allows to send up to 100 bytes on the serial link.
121 (0x79)	"Passe frame" output byte number >100	Error on scenario parameter message. The function "passe frame" allows to receive up to 100 bytes on the serial link.
122 (0x7A)	"Passe frame" input byte number = 0	Error on scenario parameter message. The function "passe frame" must send at least 1 byte on the serial link.
123 (0x7B)	Too many bytes received on "passe frame" reply	A number of bytes greater than expected according to the parameters were received on the serial link. The parameter may be wrong or the slave didn't send the expected response.
124 (0x7C)	Not enough bytes received on "passe frame" reply	A number of bytes less important than expected according to the parameters were received on the serial link. The parameter may be wrong or the slave didn't send the expected response.
125 (0x7D)	Illegal data value	A MODBUS exception message was received. An impossible value was received by the slave.
126 (0x7E)	Illegal address	A MODBUS exception message was received. A command was done addressing a MODBUS register which doesn't exist.
127 (0x7F)	Illegal function	A MODBUS exception message was received. A non-supported MODBUS function was sent.