WARNING

Installation, setting, inspection and maintenance operations must be performed only by qualified personnel in charge of it. Any operation must be carried out in compliance with the enforced regulations and legislation concerning personal safety and the use of adequate protective tools.

INTRODUCTION

The digital power factor controller PFC96Evo and PFC144Evo support the communication protocols Modbus® RTU, Modbus® ASCII and Modbus® TCP (only PFC144Evo) on the optical interface and the expansion modules:

- USB
- RS232
- RS485
- Ethernet (only PFC144Evo)

Using this function, it is possible to read the device status and to control the units through standard third party supervision software (SCADA) or through other devices fitted with Modbus® interface, such as PLCs and smart terminals.

PARAMETER SETTING

To configure the Modbus® protocol, enter the SETUP MENU and choose the ADV menu. Only one communication port can be configured.

ADVANCED MENU – SERIAL COMMUNICATION

PAR	FUNCTION	RANGE	DEFAULT
P49	Indirizzo nodo	1 ..255	1
P50	Velocità seriale	1200, 2400, 4800, 9600, 19200, 38400	9600
P51	Formato dati	8 bit Nessuna, 8 bit Dispari, 8 bit Pari, 7 bit Dispari, 7 bit Pari	8 bit Nessuna
P52	Stop bit	1, 2	1
P53	Protocollo	Modbus RTU, Modbus ASCII, Modbus TCP (solo PFC144Evo)	Modbus RTU

For the Ethernet expansion module (only PFC144Evo), additional parameters are available. Enter the SETUP MENU and select the FUN menu.

PAR	FUNCTION	RANGE	DEFAULT
F.01	IP address	000.000.000.000 - 255.255255.255	000.000.000.000
F.02	Subnet mask	000.000.000.000 - 255.255255.255	000.000.000.000
F.03	IP port	0 - 32000	1001
F.04	Client / Server	Client / Server	Server
F.05	Remote IP address	000.000.000.000 - 255.255255.255	000.000.000.000
F.06	Remote IP port	0 - 32000	1001
F.07	IP gateway address	000.000.000.000 - 255.255255.255	000.000.000.000

MODBUS® RTU PROTOCOL

If the Modbus® RTU protocol is selected, the communication message has the following structure:

T1 T2 T3	Address (8 bit)	Function (8 bit)	Data (N x 8 bit)	CRC (16 bit)	T1 T2 T3

- The Address field holds the serial address of the slave device to which the message is destined.
- The Function field holds the code of the function that must be executed by the slave.
- The Data field contains data sent to the slave or data received from the slave in response to a query. The maximum length for the data field is no.80 16-bit registers (160 bytes).
- The CRC field allows the master and slave devices to check the message integrity. If a message has been corrupted by electrical noise or interference, the CRC field allows the devices to recognize the error and thereby to ignore the message.
- The T1 T2 T3 sequence corresponds to the time during which data must not be exchanged on the communication bus to allow the connected devices to recognize the end of one message and the beginning of another. This time must be at least 3.5 times the time required to send one character.
  The controller measures the time that elapses from the reception of one character and the following one . If this time exceeds the time necessary to send 3.5 characters at the selected baudrate, then the next character will be considered as the first of a new message.

Modbus® functions

The available functions are:

03 = Read input register	Allows to read the controller measures.
04 = Read input register	Allows to read the controller measures.
06 = Preset single register	Allows to write the parameters
07 = Read exception	Allows to read the device status
10 = Preset multiple register	Allows to write several parameters
17 = Report slave ID	Allows to read information about the device.

For instance, in order to read from the controller with serial address 01 the value of the temperature inside the panel (residing at location 14 (0E Hex), the message to send is the following:

01	04	00	2D	00	02	E0	08

where:
01= slave address
04 = Modbus® function 'Read input register'
00 2D = Address of the required register (temperature) decreased by one
00 02 = Number of registers to be read beginning from address 2D
EE 08 = CRC Checksum

The controller answer is the following:

01	04	04	00	00	00	1C	FA	4D

where:
01 = device address (Slave 01)
04 = Function requested by the master
04 = Number of bytes sent by the device
00 00 00 1C = Hex value of the temperature = 28 = 28°C
FA 4D = CRC checksum

Function 04: read input register

The Modbus® function 04 allows to read one or more consecutive registers from the slave memory. The address of each measure is given in tables 2 later on in this document.
As per Modbus® standard, the address in the query message must be decreased by one from the effective address reported in the table.
If the measure address is not included in the table or the number of requested registers exceeds the acceptable max number, the controller will return an error message (see error table).

Master query:

Slave address	08h
Function	04h
MSB address	00h
LSB address	0Fh
MSB register number	00h
LSB register number	08h
LSB CRC	C1h
MSB CRC	56h

In the above example, slave 08 is requested for 8 consecutive registers beginning with address 10h.
Thus, registers from 10h to 17h will be returned. As usual, the message ends with the CRC checksum.

Slave response:

address	08h
Function	04h
Byte number	10h
MSB register 10h	00h
LSB register 10h	00h
---	---
MSB register 17h	00h
LSB register 17h	00h
LSB CRC	5Eh
MSB CRC	83h

The response is always composed of the slave address, the function code requested by the master and the contents of the requested registers. The answer ends with the CRC.

Function 06: preset single register

This function allows to write in the registers. It can be used only with registers with address higher than 1000 Hex. For instance, it is possible to change setup parameters. If the value is not in the correct range, the 8BGA will answer with an error message. In the same way, if the parameter address is not recognised, the device will send an error response. The address and the valid range for each parameter are indicated in Tables 5, 6 and 7.

Master query:

Slave address	08h
Function	06h
MSB register address	2Fh
LSB register address	0Fh
MSB data	00h
LSB data	0Ah
LSB CRC	31h
MSB CRC	83h

Slave response:

The slave response is an echo to the query, which means that the slave sends back to the master the address and the new value of the variable.

Function 07: read exception status

This function allows to read the status of the automatic transfer switch.

Master query:

Slave address	08h
Function	07h
LSB CRC	41h
MSB CRC	B2h

The following table gives the meaning of the status byte sent by the controller as answer:

BIT	MEANING
0	Program memory checksum check
1	
2	
3	
4	
5	
6	
7	

Function 17: report slave ID

This function allows to identify the device type.

Master query:

Slave address	01h
Function	11h
LSB CRC	C0h
MSB CRC	2Ch

Slave response:

Slave address	01h
Function	11h
Byte counter	08h
Date 1 (Type) ❶	49h
Date 2 (software revision)	00h
Date 3 (hardware revision)	00h
Date 4 (parameter revision)	00h
Date 5 (device type) ❷	01h
Date 6 (reserved)	00h
Date 7 (reserved)	00h
LSB CRC	00h
MSB CRC	F7h

❶ 67 – 43h = not used
69 – 45h = PFC96Evo
73 – 49h = PFC144Evo

❷ 1 – 01h LGA controllers

Errors

In case the slave receives an incorrect message, it answers with a message composed by the queried function OR with 80 Hex, followed by an error code byte. In the following table the error codes sent by the slave to the master are listed:

table 1: error codes

CODE	ERROR
01	Invalid function
02	Invalid address
03	Parameter out of range
04	Function execution impossible
06	Slave busy, function momentarily not available

MODBUS® ASCII PROTOCOL

The Modbus® ASCII protocol is normally used in applications that require to communicate through modem.
Functions and addresses available are the same as the ones for the RTU version, but the transmitted characters are in ASCII and the message end is delimited by Carriage return/Line Feed instead of a transmission pause.
If the parameter P.53 as Modbus® ASCII protocol is selected, the communication message on the correspondent communication port has the following structure:

:	Address 2 chars	Function 2 chars	Data (N. chars)	LRC 2 chars	CR LF

- The Address field holds the serial address of the slave device to which the message is destined.
- The Function field holds the code of the function that must be executed by the slave.
- The Data field contains data sent to the slave or data received from the slave in response to a query. The maximum allowed length is of 8 consecutive registers.
- The LRC field allows the master and slave devices to check the message integrity. If a message has been corrupted by electrical noise or interference, the LRC field allows the devices to recognize the error and therefore ignore the message.
- The message always ends with CRLF control character (0D 0A).

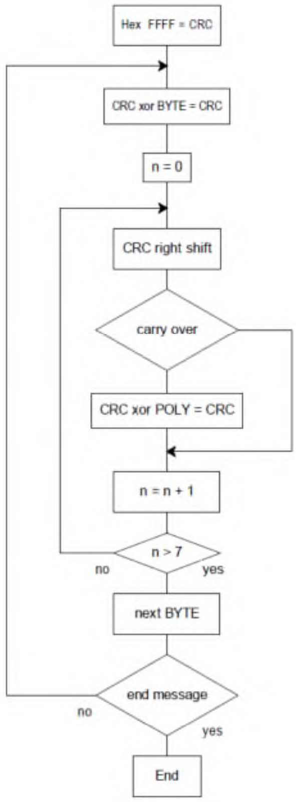
For instance, in order to read from the slave with serial address 08 the equivalent voltage residing at location 4 (04 Hex), the message to send is the following:

:	08	04	00	03	00	02	56	CRLF

where:
: = ASCII 3Ah message start delimiter
08 = slave address
04 = Modbus® function 'Read input register'
00 03 = Address of the required register (voltage) decreased by one
00 02 = Number of registers to be read beginning from address 04
56= LRC Checksum
CRLF = ASCII 0Dh 0Ah = Message end delimiter
The response is the following:

:	08	04	04	00	00	01	3D	9B	CRLF

where:
: = ASCII 3Ah message start delimiter
08 = device address (Slave 08)
04 = Function requested by the master
04 = Number of bytes sent by the multimeter
00 00 01 3D = measured voltage Hex value (317V)
9B = LRC checksum
CRLF = ASCII 0Dh 0Ah = Message end delimiter



CRC CALCULATION (CHECKSUM for RTU)

Example of CRC calculation:

Frame = 0207h

CRC initialization	1111	1111	1111	1111	
Load the first byte			0000	0010	
Execute xor with the first Byte of the frame	1111	1111	1111	1101	
Execute 1st right shift	0111	1111	1111	1110	**1**
Carry=1,load polynomial	1010	0000	0000	0001	
Execute xor with the polynomial	1101	1111	1111	1111	
Execute 2nd right shift	0110	1111	1111	1111	**1**
Carry=1,load polynomial	1010	0000	0000	0001	
Execute xor with the polynomial	1100	1111	1111	1110	
Execute 3rd right shift	0110	0111	1111	1111	**0**
Execute 4th right shift	0011	0011	1111	1111	**1**
Carry=1,load polynomial	1010	0000	0000	0001	
Execute xor with the polynomial	1001	0011	1111	1110	
Execute 5th right shift	0100	1001	1111	1111	**0**
Execute 6th right shift	0010	0100	1111	1111	**1**
Carry=1,load polynomial	1010	0000	0000	0001	
Execute xor with the polynomial	1000	0100	1111	1110	
Execute 7th right shift	0100	0010	0111	1111	**0**
Execute 8th right shift	0010	0001	0011	1111	**1**
Carry=1,load polynomial	1010	0000	0000	0001	
Load the second byte of the frame			0000	0111	
Execute xor with the Second byte of the frame	1000	0001	0011	1001	
Execute 1st right shift	0100	0000	1001	1100	**1**
Carry=1,load polynomial	1010	0000	0000	0001	
Execute xor with the polynomial	1110	0000	1001	1101	
Execute 2nd right shift	0111	0000	0100	1110	**1**
Carry=1,load polynomial	1010	0000	0000	0001	
Execute xor with the polynomial	1101	0000	0100	1111	
Execute 3rd right shift	0110	1000	0010	0111	**1**
Carry=1,load polynomial	1010	0000	0000	0001	
Execute xor with the polynomial	1100	1000	0010	0110	
Execute 4th right shift	0110	0100	0001	0011	**0**
Execute 5th right shift	0010	0100	0000	1001	**1**
Carry=1,load polynomial	1010	0000	0000	0001	
Execute xor with the polynomial	1001	0010	0000	1000	
Execute 6th right shift	0100	1001	0000	0100	**0**
Execute 7th right shift	0010	0100	1000	0010	**0**
Execute 8th right shift	0001	0010	0100	0001	**0**
CRC Result	**0001 0001**	**0010**		**0100**	
	12h	**41h**			

Note: The byte 41h is sent first(even if it is the LSB), then12h is sent.

LRC CALCULATION (CHECKSUM for ASCII)

Example of LRC calculation:

Address	01		00000001
Function	04		00000100
Start address hi.	00		00000000
Start address lo.	00		00000000
Number of registers	08		00001000
		Sum	00001101
	1. complement		11110010
		+ 1	0000001
	2. complement		11110101
LRC result			
			F5

table 2: measured supplied by serial communication protocol (to be used with functions 03 e 04)

ADDRESS	WORDS	MEASURE	UNIT	FORMAT
02h	2	Cos phi ❶		Unsigned long
04h	2	Voltage	V	Unsigned long
06h	2	Current	A / 1000	Unsigned long
08h	2	Delta kvar ❷	kvar	Signed long
0Ah	2	Weekly average power factor	/100	Unsigned long
0Ch	1	Capacitor overload current	%	Unsigned integer
0Eh	2	Panel temperature ❷	°C	Signed long
10h	2	Output status		Unsigned long
12h	2	Phase angle offset	° / 4	Unsigned long
14h	2	Power factor		Signed long
16h	2	Error bit status ❸		Unsigned long

ADDRESS	WORDS	MEASURE	UNIT	FORMAT
18h	2	Max voltage	V	Unsigned long
1Ah	2	Max current	A / 100	Unsigned long
1Ch	2	Max capacitor overload	%	Unsigned long
1Eh	2	Max temperature	°C	Unsigned long
200h	2	Step 1 insertion time	min	Unsigned integer

21Ah	2	Step 14 insertion time	min	Unsigned integer
2020h	2	Step 1 no. of insertions		Unsigned integer

202Dh	2	Step 14 no. of insertions		Unsigned integer
2040h	2	Step 1 residual power	kvar	Unsigned integer

204Dh	2	Step 14 residual power	kvar	Unsigned integer
2008h	1	THDV	% / 10	Unsigned integer
2080h	1	2nd order voltage harmonic	% / 10	Unsigned integer

208Dh	1	15th order voltage harmonic	% / 10	Unsigned integer
2009h	1	THDI	% / 10	Unsigned integer
20A0h	1	2nd order current harmonic	% / 10	Unsigned integer

20ADh	1	15th order current harmonic	% / 10	Unsigned integer
2007h	1	Hours to maintenance	h	Signed integer

❶ Bit 31 indicates the sign (0=positive; 1=negative), while bit 30 indicates induttivo/capacitivo load (0=ind; 1=cap)

❷ Bit 31 indicates the sign (0=positive; 1=negative)

❸ By reading the words starting at address 16h, 32 bits will be returned with the meaning as per table 3:

table 3: codifica allarmi

BIT #	ALARM
0	A01 undercompensation
1	A02 overcopmpensation
2	A03 low current
3	A04 high current
4	A05 low voltage
5	A06 high voltagea
6	A07 capacitor overload
7	A08 overtemperature
8	A09 microinterruption
9	A10 voltage THD too high
10	A11 current THD too high
11	A12 maintenance required
12	A13 step failure
13 - 31	Free bits

table 4: Commands (to be used with function 06)

ADDRESS	WORDS	COMMAND	FORMAT
2FF0h	1	Commands list ❶	Unsigned integer
3000h	1	Operational mode change ❷	Unsigned integer
3001h	1	Device reset ❸	Unsigned integer
3005h	1	Step activation ❹	Unsigned integer
3006h	1	Step de-activatio ❹	Unsigned integer
3007h	1	Keyboard lock ON/OFF ❺	Unsigned integer

❶ The following table shows the functions associated to the value to be written at the address 2FF0h. It is possible to activate several functions at the same time.

table 5:

VALUE	FUNCTION
00h	C01 reset maintenance service interval
01h	C02 reset step operation counter
02h	C03 reset original powers in step adjusting
03h	C04 reset the step operation hour counter
04h	C05 reset maximum peak values
05h	C06 reset weekly total power factor history
06h	C07 reset parameters to factory default
07h	C08 create a backup copy of the user settings
08h	C09 reload parameters using the user backup copy

❷ The following table indicates the values to be written at address 3000h to achieve the corresponding functions:

VALUE	FUNCTION
0	Switch from MAN to AUT and vice-versa
1	Switch to MAN mode
2	Switch to AUT mode

❸ By writing 01 at the indicated address, the corresponding function will be activated.

❹ Write in the corresponding register the number of the step to be activated/de-activated. An attempt to activate a step while the reconnection time is running will be ignored.

❺ By writing 01 at the indicated address the keyboard will be locked. By writing 00, it will be unlocked.

PARAMETER SETTING

By using the Modbus® protocol, the menu parameters can be accessed. In order to understand correctly the correspondence between the numeric value and the selected function and/or the unit of measure, please see the 8BGA operating manual.

Procedure for reading parameters

1. Write the value of the parameter to be read by using ***function 6*** at address ***5002h***. ❶
2. Perform ***function 4*** at the address ***5003h***, with a number of registers appropriate to the length of the parameter (see table).
3. To read the next parameter, repeat step 2, otherwise perform step 1.

Procedure for writing parameters

1. Write the value of the menu to be changed by using ***function 6*** at address ***5002h***. ❶
2. Perform ***function 16*** at address ***5003h***, with a number of registers appropriate to the length of the parameter
3. To write the next parameter, repeat step 2, otherwise perform step 1. If no additional parameters need to be written, go to step 4.
4. To make the changes to setup parameters effective, it is necessary to store the values in the EEPROM using the dedicated command. Write 5 by using ***function 6*** at address ***2F03h.***

TYPE OF PARAMETER	NUMBER OF REGISTER
Numeric value ≤ 65535 (for example, P.30)	1 register (2 byte)
Numeric value > 65535 (for example, P.6)	2 registers (4 byte)

❶ It is possible to read the parameter stored at the address ***5002h*** by using ***function 4***.

Example

Set the value of parameter P.01 to 250

Step 1: P.01 parameter setup

MASTER Function = 6
Address = 5002hH (5002h – 0001h = 5001h)
Value = 1 (01hH)

01	06	50	01	00	01	08	CA

controller Function = 6
Address = 5002h (500h – 0001h = 5001h)
Value = 1 (01hH)

01	06	50	01	00	01	08	CA

Step 2: value 250 setup

MASTER Function = 16 (10h)
Address = 5003h (5003h – 0001h = 5002h)
Register no. = 2 (02h)
Byte no. = 4 (04h)
Value = 250 (000000FAh)

01	10	50	02	00	02	04	00	00	00	FA	0E	36

controller Function = 16 (10h)
Address = 5003h (5004h – 0001h = 5003h)
Value = 250 (000000FAh)

01	10	50	02	00	02	F1	08

Step 3: save and reboot

MASTER Function = 6 (06h)

Address = 2F03h (2F03h – 0001h = 2F02h)
Value = 1 (01H)

01	6	2F	02	00	04	21	1D

controller No answer

SERVICE

Please refer to:

ITALFARAD Caring for your energy
by ORTEA S.p.A. Via dei Chiosi 21 20873 Cavenago B.za (MB) – Italia
Tel. +39 02 9591 7800
italfarad@italfarad.com