PANASONIC FIRE ALARM SOLUTIONS

TECHNICAL DESCRIPTION

MODBUS PROTOCOL IN WEB-SERVER / GATEWAY

DOCUMENT INFORMATION

DOCUMENT NAME:	TECHNICAL DESCRIPTION MODBUS PROTOCOL IN WEB-SERVER / GATEWAY
DOCUMENT NUMBER:	MEW02061
DATE OF ISSUE:	2016-11-07
REV:	4
DATE OF REVISION:	2020-11-17

Panasonic Fire & Security Europe AB
Jungmansgatan 12
SE-211 11 Malmö
Sweden
Tel: +46 (0)40 697 70 00
Internet: www.panasonic-fire-security.com

TABLE OF CONTENTS

1 INTRODUCTION 3
1.1 REFERENCE DOCUMENTS 3
2 ABBREVIATIONS 3
3 GENERAL DESCRIPTION 4
4 INTERFACES 5
4.1 HARDWARE INTERFACE 5
4.1.1 MODBUS SERIAL 5
4.1.2 MODBUS TCP/IP 5
4.2 SOFTWARE INTERFACE 5
4.2.1 PROTOCOL VERSION 1 5
4.2.2 PROTOCOL VERSION 2 7
4.2.3 LIMITATIONS 10
5 TECHNICAL DATA 11

1 INTRODUCTION

This document describes the subset of Modbus that Panasonic Fire & Security Europe AB provides for the EBL-system.

The document is intended for third parts that need Modbus to interface our fire alarm system. It is also the specification for the internal implementation.

1.1 REFERENCE DOCUMENTS

[1] Company document: Technical Description of the Web-server II, MEW01930.
[2] Company document: Technical Description of the Gateway, MEW02670
[3] Other Document: PI–MBUS–300 Rev. J, “Modicon Modbus Protocol Reference Guide”
[4] Other Document: MODBUS Messaging on TCP/IP Implementation Guide V1.0b

2 ABBREVIATIONS

CU / c.i.e.	Control Unit	Control and Indicating Equipment
The application	The application in the web-server / gateway	
Web-server	PFSEU AB product Web-server II 1598	
ADU	Application Data Unit	
PDU	Protocol Data Unit	

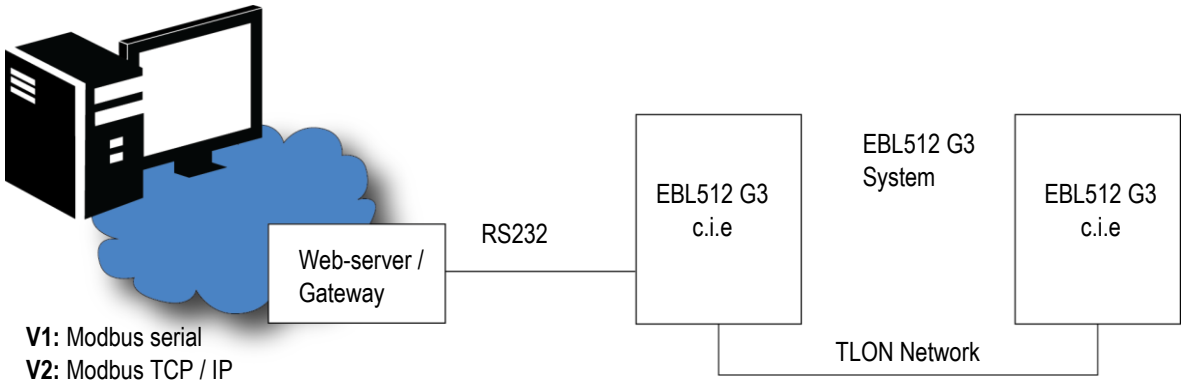
3 GENERAL DESCRIPTION

Using the web-server or gateway with Modbus, it is possible to interface a system of EBL512, EBL128, and EBL512 G3 control panels.

For newer systems, the protocol is installed via EBLWin.
For older systems, the protocol is installed via Modbus config tool.

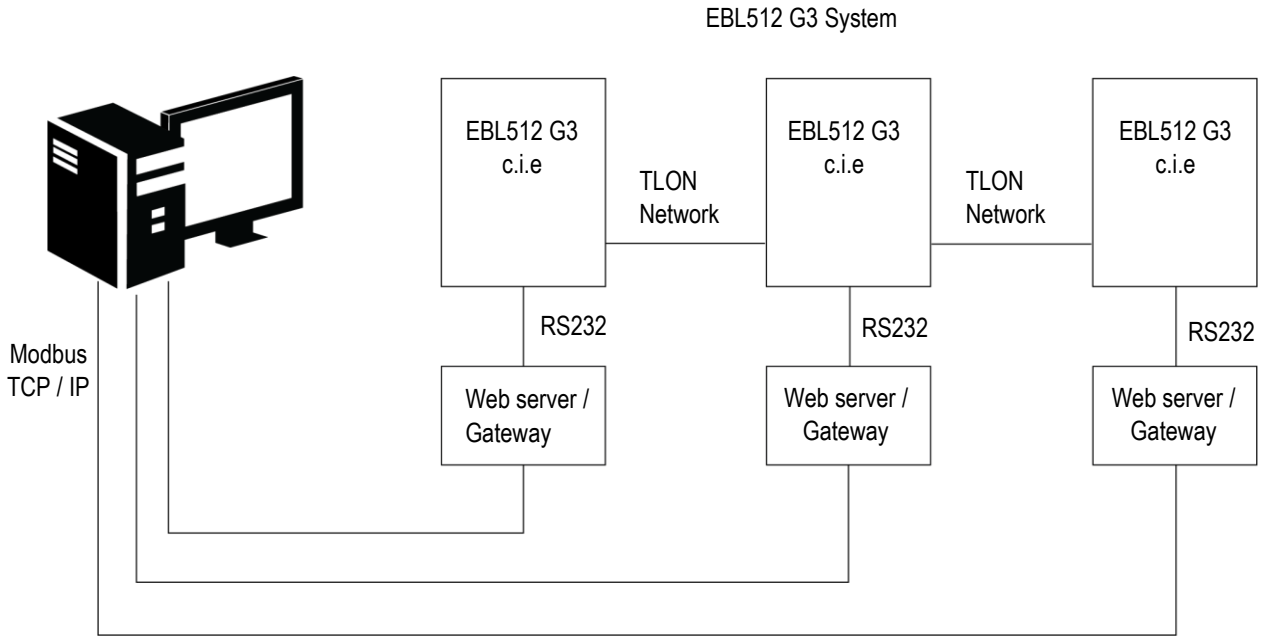
See chapter 5 TECHNICAL DATA.

Modbus Protocol V1 & V2



Modbus Protocol V2

Used when need to monitor more information. Possible to monitor up to 999 zones with 99 addresses each.



4 INTERFACES

4.1 HARDWARE INTERFACE

4.1.1 MODBUS SERIAL

The web-server / gateway is equipped with RS232 serial ports. The application will use the Modem COM port for the Modbus protocol. See [1], section 3.4. MODEM COMMUNICATION (RS232C) for more information. In addition to this, a RS232 to RS485 converter can be supplied. Communication parameters will be configurable:

Baud rate: 2400, 4800, 9600, 19200 or 38400
Data bits: 8
Parity: None, Odd or Even

4.1.2 MODBUS TCP/IP

The web-server / gateway is equipped with a 10Base-T (RJ45) connector for a standard Ethernet cable. The application will use this interface for the Modbus TCP/IP protocol. See [1], section 3.2. ETHERNET COMMUNICATION (10BASE-T) for more information.

4.2 SOFTWARE INTERFACE

The application supports two different versions of Modbus protocol that can be used for either serial or ethernet interfaces.

- Version 1. Is used for serial interface and provides information about fire alarm per zone.
- Version 2. Is used for ethernet interface and provides information of alarm point status, such as fire alarm state, prewarning, disablement, fault, and service.

It will be possible to configure the slave id of the web-server / gateway.

4.2.1 PROTOCOL VERSION 1

The application will implement the Modbus RTU framing mode (see [2], p. 8).

To provide information about fire alarm state for each possible zone, the application will reply to queries from the master for **function code 4, "Read input registers"** (See [2], p. 30). Valid registers are thus 30001 to 30999 and these registers correspond to zones 1 to 999 in the fire alarm system. The least significant bit of the 16 bits ("bit 0") will be set to 1 if there is a fire alarm, or set to 0 if there is no fire alarm. Bits 15 to 1 are unused.

For example, to find out if there is a fire alarm in zone 123, the master should request to read register 30123 from the web-server / gateway:

TABLE 1. QUERY EXAMPLE

Field Name	(Hex)
Slave Address	[Configurable]
Function	04
Starting Address Hi	00
Starting Address Lo	7A
No. of Points Hi	00
No. of Points Lo	01
Error Check (LRC or CRC)	—

The application will then reply as follows:

TABLE 2. RESPONSE EXAMPLE

Field Name	**(Hex)**
Slave Address	[Configurable]
Function	04
Byte Count	02
Data Hi (Register 30123)	00
Data Lo (Register 30123)	00 = no fire alarm 01 = fire alarm
Error Check (LRC or CRC)	—

In addition to this, we will present communication status between web-server / gateway and control panel in register 31000. The least significant bit of the 16 bits (“bit 0”) will be set to 0 if there is a communication failure, or set to 1 if communication is normal. Bits 15 to 1 are unused.

TABLE 3. QUERY OF COMMUNICATION STATUS

Field Name	**(Hex)**
Slave Address	[Configurable]
Function	04
Starting Address Hi	03
Starting Address Lo	E7
No. of Points Hi	00
No. of Points Lo	01
Error Check (LRC or CRC)	—

TABLE 4. RESPONSE WITH COMMUNICATION STATUS

Field Name	**(Hex)**
Slave Address	[Configurable]
Function	04
Byte Count	02
Data Hi (Register 31000)	00
Data Lo (Register 31000)	00 = communication down 01 = communication ok
Error Check (LRC or CRC)	—

4.2.2 PROTOCOL VERSION 2

The application will implement the Modbus TCP/IP with two protocols:

1. **Modbus RTU over TCP**. This protocol frame have basically the same ADU as in RTU framing mode (see [2], p. 8) that is used over Modbus Serial, but is wrapped with an ethernet frame without any byte changes of the ADU.
2. **Modbus TCP**. This protocol frame uses the Modbus PDU (Function code and Data) together with a dedicated header used over TCP/IP to identify Modbus ADU. The header is called MBAP, Modbus Application Protocol Header (see [3], section 3.1.2).

The application will determine which of these protocols to be used by examining the master's request frame.

To provide status information about alarm-points, the application will reply to queries from the master for **function code 4, "Read input registers"** (See [2], p. 30). Valid registers are thus 30001 to 39999 and these registers correspond to alarm-point **x**01-01 to **x**99-99 in the fire alarm system. A web-server / gateway Id is used to identify the digit **x** in an alarm-point.
To provide status information for alarm-point with zone 301, we need to use a web-server / gateway with Id = 3 (configurable in EBLWin, via web-server / gateway technical address).
The 16 bit input register will hold for different status data of an alarm-point. Currently, only the low byte ("bit 0-7") is used, while the high byte ("bit 8-15") is saved for future usage.
The bit of interest will be set to 1 if the state is active, or set to 0 if the state is inactive.

The communication status between web-server / gateway and control panel will be provided in register 31000. The least significant bit of the 16 bits ("bit 0") will be set to 0 if there is a communication failure, or set to 1 if communication is normal. Bits 15 to 1 are unused.

TABLE 1. Status data bits (Low byte)

Bit no.	Field Name
0	Fire alarm state
1	Prewarning state
2	Heavy/Much fire state
3	Fault
4	Disabled
5	Disabled by Zone
6	Service signal
7	Not used

For example:
If there is a fire on Zone : 124 Address : 45
The system asks for register point "32445" on web-server 1
And got a reply as : 0000000**1**

If there is a fault on Zone : 425 Address : 86
The system asks for register point "32586" on web-server 4
And got a reply as : 0000**1**000

4.2.2.1 Modbus RTU over TCP

An example with Modbus RTU over TCP to find out the alarm point status of point 012-34. The master should request to read register 31234 from the web-server / gateway:

TABLE 2. QUERY EXAMPLE (MODBUS RTU OVER TCP)

Field Name	**(Hex)**
Slave Address	[Configurable]
Function	04
Starting Address Hi	04
Starting Address Lo	D1
No. of Points Hi	00
No. of Points Lo	01
Error Check (LRC or CRC)	—

TABLE 3. RESPONSE EXAMPLE (MODBUS RTU OVER TCP)

Field Name	**(Hex)**
Slave Address	[Configurable]
Function	04
Byte Count	02
Data Hi (Register 31234)	00
Data Lo (Register 31234)	Status data bits (see table 1.)
Error Check (LRC or CRC)	—

TABLE 4. QUERY OF COMMUNICATION STATUS (MODBUS RTU OVER TCP)

Field Name	**(Hex)**
Slave Address	[Configurable]
Function	04
Starting Address Hi	03
Starting Address Lo	E7
No. of Points Hi	00
No. of Points Lo	01
Error Check (LRC or CRC)	—

TABLE 5. RESPONSE WITH COMMUNICATION STATUS (MODBUS RTU OVER TCP)

Field Name	**(Hex)**
Slave Address	[Configurable]
Function	04
Byte Count	02
Data Hi (Register 31000)	00
Data Lo (Register 31000)	00 = communication down 01 = communication ok
Error Check (LRC or CRC)	—

4.2.2.2 Modbus TCP

An example with Modbus TCP to find out the alarm point status of point 012-34. The master should request to read register 31234 from the web-server / gateway:

TABLE 6. QUERY EXAMPLE (MODBUS TCP)

Field Name	**(Hex)**
Transaction Id Hi	00
Transaction Id Lo	01 (Increments for each message)
Protocol Id Hi	00
Protocol Id Lo	00 = Modbus protocol
Length Hi	00
Length Lo	06 (Number of following bytes)
Unit Id / Slave Address	[Configurable]
Function	04
Starting Address Hi	04
Starting Address Lo	D1
No. of Points Hi	00
No. of Points Lo	01

TABLE 7. RESPONSE EXAMPLE (MODBUS TCP)

Field Name	**(Hex)**
Transaction Id Hi	00
Transaction Id Lo	01 (Copy from query)
Protocol Id Hi	00
Protocol Id Lo	00 = Modbus protocol
Length Hi	00
Length Lo	06 (Number of following bytes)
Unit Id / Slave Address	[Configurable]
Function	04
Byte Count	02
Data Hi (Register 31234)	00
Data Lo (Register 31234)	Status data bits (see table 1.)

TABLE 8. QUERY OF COMMUNICATION STATUS (MODBUS TCP)

Field Name	**(Hex)**
Transaction Id Hi	00
Transaction Id Lo	01 (Increments for each message)
Protocol Id Hi	00
Protocol Id Lo	00 = Modbus protocol
Length Hi	00
Length Lo	06 (Number of following bytes)
Unit Id / Slave Address	[Configurable]
Function	04
Starting Address Hi	03
Starting Address Lo	E7
No. of Points Hi	00
No. of Points Lo	01

TABLE 9. RESPONSE WITH COMMUNICATION STATUS (MODBUS TCP)

Field Name	**(Hex)**
Transaction Id Hi	00
Transaction Id Lo	01 (Copy from query)
Protocol Id Hi	00
Protocol Id Lo	00 = Modbus protocol
Length Hi	00
Length Lo	06 (Number of following bytes)
Unit Id / Slave Address	[Configurable]
Function	04
Byte Count	02
Data Hi (Register 31000)	00
Data Lo (Register 31000)	00 = communication down 01 = communication ok
Error Check (LRC or CRC)	—

4.2.3 LIMITATIONS

4.2.3.1 PROTOCOL VERSION 1

Modbus protocol version 1 is only supported for serial communication.

4.2.3.2 PROTOCOL VERSION 2

Modbus protocol version 2 is only supported for TCP/IP communication and
the application work as a single end-to-end communication and only supports establishment of one connection at the time. When the application is already busy with a connection, a second connection has to wait until the first connection is closed.

When the application does not receive any Modbus query within 30 seconds, it will conclude that the connection is down and closes the connection.

5 TECHNICAL DATA

Hardware	1598 - Web-server II 5088 - Gateway

EBL-system	**EBL512 V. 2.5.X / United V.2.7.X**	**EBL128 / EBL512 G3 ≥2.1.X**	**EBL512 G3 ≥2.6.X**	**EBL512 G3 ≥2.9.X**
Web-server software:	Modbus 512	EBLWeb	EBLWeb	
Gateway Software:				EBLWeb
Configuration	Modbus II Config Tool	EBLWin	EBLWin	EBLWin
Protocol version	V.1	V.1	V.1 and V.2	V.1 and V.2

NOTE! Modbus cannot be configured for other EBL versions than above.