



# **RO-ETH-PROTOCOL**

## **Hardware-Description**

2011 Oktober

# INDEX

<b><u>1. Introduction</u></b>	<b>4</b>
<b><u>2. Accessing registers</u></b>	<b>6</b>
<b><u>2.1. What are registers ?</u></b>	<b>6</b>
<b><u>2.2. Accessing registers with 8, 16, 32 or 64 bit data</u></b>	<b>6</b>
<b><u>2.3. Register assignment</u></b>	<b>7</b>
<b><u>3. The TCP protocol</u></b>	<b>9</b>
<b><u>3.1. IP-adress</u></b>	<b>9</b>
<b><u>3.2. Netmask</u></b>	<b>9</b>
<b><u>3.3. Port</u></b>	<b>9</b>
<b><u>4. TCP request/response string</u></b>	<b>11</b>
<b><u>4.1. Request string structure</u></b>	<b>11</b>
<b><u>4.1.1. JOB-ID</u></b>	<b>12</b>
<b><u>4.1.2. Length of the request string</u></b>	<b>12</b>
<b><u>4.1.3. COMMAND</u></b>	<b>12</b>
<b><u>4.1.4. WIDTH</u></b>	<b>12</b>
<b><u>4.1.5. ADDRESS</u></b>	<b>13</b>
<b><u>4.1.6. DATA</u></b>	<b>13</b>
<b><u>4.2. Response string structure</u></b>	<b>14</b>
<b><u>4.2.1. Error code</u></b>	<b>14</b>
<b><u>4.2.2. JOB-ID</u></b>	<b>14</b>
<b><u>4.2.3. Length of the response string</u></b>	<b>14</b>
<b><u>4.2.4. DATA</u></b>	<b>15</b>
<b><u>5. Appendix</u></b>	<b>17</b>
<b><u>5.1. Revisions</u></b>	<b>17</b>
<b><u>5.2. Copyrights and trademarks</u></b>	<b>18</b>



# Introduction

---



# 1. Introduction

The communication with our modules takes place over the ethernet interface.

The modules reacts on a request (request string), handles the request and replies with a response (response string).

## **The request:**

The request is realized as request string. It is transmitted over the TCP protocol. Using the request, read or write commands are sent.

## **The response:**

After the request has been worked off, the response string is build and then sent to the requester. In doing so, the request may be simply acknowledged, data may be sent back or an error code is returned.

The exact structure of the request and response string will be described in the following chapters.

# Accessing registers

---



## 2. Accessing registers

### 2.1. What are registers ?

Registers are little scratch-pad like storages, buffering writing or reading data. The data stays stored until they are overwritten or until their power supply is cut off. They have an address range in order to address them. With the aid of the address, you can read from the registers or write to them. The registry access is therefore addressed.

A special feature are the registers of the input state change flags. If they are read, their data will be reset at the same time.

### 2.2. Accessing registers with 8, 16, 32 or 64 bit data

Accessing registers can be done in different data widths. The data can be either 8 (byte), 16 (Word), 32 (long) or 64 (eXtralong) bits wide. With the aid of the address, the desired access range is selected. An address refers to 8 bits.

If for example a 32-bit access to address 0004 hex occurs, then 4 x 8 bytes (as a data block) starting with address 0004 hex until address 0007 hex will be read or written.

#### Subdivided table in 8, 16, 32, and 64 bit register accesses

Access width	Address	00	01	02	03	04	05	06	07
8 Bit	0000 hex	X							
16 Bit	0000 hex	X	X						
32 Bit	0000 hex	X	X	X	X				
32 Bit	0004 hex					X	X	X	X
64 Bit	0000 hex	X	X	X	X	X	X	X	X

When writing or reading, the data is written or read byte by byte. You need to pay attention to the byte order. The byte order starts with the low-byte order (byte order: Little Endian).

### Byte order of the data in the register

Access width	Address	Value	00	01	02	03	04	05	06	07
8 Bit	04	1a					1a			
16 Bit	06	1a1b							1b	1a
32 Bit	00	01020304	04	03	02	01				
32 Bit	04	01020304					04	03	02	01
64 Bit	00	0102030405060708	08	07	06	05	04	03	02	01

## 2.3. Register assignment

You will find this in the RO-register assignment document.

# The TCP protocol



## 3. The TCP protocol

Communication takes place using the TCP/IP protocol on the ethernet interface. A network connection between the module and a PC is assumed.

### 3.1. IP-address

The IP-address is modifiable by using the integrated web interface. This is necessary, if the IP-address is already in use.

### 3.2. Netmask

If the IP-address is outside of the IP-address range and/or the IP-address range is subdivided differently, then the netmask must be modified using the integrated webserver.

### 3.3. Port

The TCP-transfer to send and receive is realized using the "**default Port 9912**". Should this be a problem (e.g. due to a firewall), than the port may be modified (→ **manual RO-Series**).

# TCP request/response string

---



IV

## 4. TCP request/response string

At the beginning, the request string will be explained in detail with a request string example. Afterwards, the request string will be explained with a response to the request example.

### 4.1. Request string structure

Data is transmitted in hexadecimal.

Example for a write command:

The following example writes the data 0f hex to the address 0012 hex (the request string is build up of 11 characters). The JOB-ID is 01.

Character no.	Description	Value	Hexadecimal
1	Packet_ID_0		63
2	Packet_ID_1		9a
3	TCP_CMD_RO_1		01
4	JOB_ID		01
5	Length of request string (bit 8-15)	11 dec. (amount of characters)	00
6	Length of request string (bit 0-7)		0b
7	COMMAND	ASCII "W"	57
8	WIDTH (data width)	ASCII "B"	42
9	ADDRESS bit 8-15	0012 hex	00
10	ADDRESS bit 0-7		12
11	DATA (= 0f hex)	0f hex	0f

### 4.1.1. JOB-ID

The JOB-ID denotes the actual request string. Two adjacent request strings may not have the same JOB-ID.

HINT:

After a successful transmission, the request routine should increase the JOB-ID by "1". That way, the numbers from "0" to "255" and then "0" again are transferred successively.

### 4.1.2. Length of the request string

The length of the request string is build up of 2 bytes and indicates how much bytes the request string is build of. The most significant bits are transmitted first. In the example, the length is 11 bytes.

### 4.1.3. COMMAND

With COMMAND, the character "W" is sent to write to registers while "R" is used to read from registers.

### 4.1.4. WIDTH

In dependency of the used "COMMAND" and the data "WIDTH", the sent data is different in width.

COMMAND	WIDTH	Width of data (bits)	Amount bytes
W	B	8	1
W	W	16	2
W	L	32	4
W	X	64	8
R		0	0

The COMMAND = "R" has no influence to the WIDTH-character of the request string. It controls the data length of the response string.

#### **4.1.5. ADDRESS**

The registers are 16 bit large and are represented by 2 hexadecimal characters. This is realized using 2 bytes. Durring the transmission, the most significant bits are sent first. The transmission begins with bits 8-15, then bits 0-7 follows.

#### **4.1.6. DATA**

The data field only exists durring a write command. It is left out for a read command.

## 4.2. Response string structure

The response string is in hexadecimal and begins with 2 bytes. They are labeled as PACKET\_ID\_0 (hex 63) and PACKET\_ID\_1 (hex 9a) and followed by TCP\_ANSWER\_RO\_1 (hex 81).

Now comes an error code and successive hexadecimal characters. In case the request string is a read command, read-out data will be appended at the end of the string.

OK-response to the example's request string from chapter 4

Character no.	Description	Value	Hexadecimal
1	DEDITEC_PACKET_ID_0		63
2	DEDITEC_PACKET_ID_1		9a
3	DEDITEC_TCP_ANSWER_RO_1		81
4	OK (OK = 0x00)		00
5	JOB-ID		61
6	Length of response string (bit 8-15)	7	00
7	Length of response string (bit 0-7)		37

### 4.2.1. Error code

The error code indicates with 0 (hex) = ok, that the requester's data were successfully received. Any other error code indicates an error.

### 4.2.2. JOB-ID

The received JOB-ID from the requester is sent back. Two consecutive request strings may not have the same JOB-ID.

### 4.2.3. Length of the response string

The length of the response string consists of 2 bytes and indicates the byte-length of the response string. The most significant bits are sent first. In the example, the length is 7 bytes.

#### 4.2.4. DATA

The data field only exists in case of a reply to a read command. A reply to a write command has no data field.

# Appendix



# 5. Appendix

## 5.1. Revisions

Rev 1.00	First issue
Rev 2.00	Design change

## **5.2. Copyrights and trademarks**

Linux is registered trade-mark of Linus Torvalds.

Windows CE is registered trade-mark of Microsoft Corporation.

USB is registered trade-mark of USB Implementers Forum Inc.

LabVIEW is registered trade-mark of National Instruments.

Intel is registered trade-mark of Intel Corporation

AMD is registered trade-mark of Advanced Micro Devices, Inc.