See page 12 for
quick start

Ethernet Converter Device

FMod-TCP BOX 2

User Manual

Version 2.4

five co
innovative engineering

Version: 2.4
Last revision: March 18, 2019
Printed in Switzerland

Warning
This device is not intended to be used in medical, life-support or space products.

Any failure of this device that may cause serious consequences should be prevented by implementation of backup systems. The user agrees that protection against consequences resulting from device system failure is the user's responsibility. Changes or modifications to this device not explicitly approved by FiveCo will void the user's authority to operate this device.

Support
Web page: http://www.fiveco.ch/product-fmod-tcp-box-2.html
e-mail: support@fiveco.ch

Table of Contents

1 Package and operating conditions 6
Starter Kit contents 6
Electrical characteristics 6

2 Overview 7
Applications 7
Software operating principle 7
Hardware description 8
SOS button 10

3 Quick start 12
Changing IP address 12

4 Controlling the FMod-TCP BOX 2 by TCP or UDP 14
General Information 14
RS232 (TCP # 8000) 14
Device parameters and I/O, A/D and I2C features (TCP # 8010 or UDP # 7010). 15
Easy IP address config (UDP # 7010) 21
Checksum calculation 21

5 Java Applet 24
Overview 24
Main Config 25
Test A/D and I/Os 26
Test RS232 27
Test I2C 28

6 Win32 Application 29
Overview 29
RS232 interface 31
"Load web files" interface 32
Main port interface 33

7 Registers management 35
Memory Organization 35
Full Register Description 36

Revision history

Revision	Date	Author	Note	Firmware version	Applet version	Win32 app version
1.0	05.05.06	AG	- First version	Since 1.0	Since 1.0	Since 3.0
1.1	09.06.06	AG	- Update specifications - Text corrections	Since 1.0	Since 1.0	Since 3.0
1.2	14.08.06	AG	- I2C speed correction. - Warning register bits correction.	Since 1.0	Since 1.0	Since 3.0
1.3	10.01.08	AG	- Text corrections	Since 1.0	Since 1.0	Since 3.0
1.4	02.06.08	AG	- Specifications correction. - Text corrections.	Since 1.0	Since 1.0	Since 3.0
1.5	30.06.08	AG	- Relays pin out correction.	Since 1.0	Since 1.0	Since 3.0
1.6	28.10.08	AG	- AD resolution correction	Since 1.0	Since 1.0	Since 3.0
1.7	15.07.10	AG	- Add I2C error answer message. - Small text corrections - Add comments about EEPROM registers saving.	Since 1.8	Since 1.2	Since 3.6
1.8	27.08.10	AG	- Correction of Inputs and Outputs registers address in their description.	Since 1.8	Since 1.2	Since 3.6
2.0	10.06.11	AG	- Update for Fmod-TCP BOX 2 device specs.	Since 2.0	Since 2.0	Since 3.7
2.1	02.11.11	AG	- Registers addresses correction.	Since 2.0	Since 2.0	Since 3.7
2.2	23.03.17	AG	- Corrections and enhancements in overview chapter.	Since 2.0	Since 2.0	Since 3.7
2.3	01.05.17	AG	- Swap IO1 & IO2 - Add Java Applet deprecation info.	Since 2.0	Since 2.0	Since 3.7
2.4	18.03.19	AG	- DIN con pinout clarification.	Since 2.0	Since 2.0	Since 3.7

1 Package and operating conditions

Starter Kit contents

The FMod-TCP BOX 2 "Starter kit" should contain:

- FMod-TCP BOX 2 device
- RS232 DSub cable
- DIN 41651 40 lines cable
- CD-Rom with dedicated software and Java applet sample
- This manual

Please note that the female connector for the power supply is always provided with the device.

Electrical characteristics

Damage may occur if the device is operated using values beyond those mentioned below.

Parameter	Conditions/Remarks	Min	Typ	Max	Unit
Supply voltage (Vdd)		12	24	33	V
Supply current	Internal			100	mA
	All outputs and supply pins of extension connector altogether.			3.7	A
Inputs voltage		-15		28	V
Outputs 1, 2 current	Relays.			1	A
Outputs 1, 2 voltage	Max voltage to GND.			50	V
Outputs 3-14 current				0.15	A
Outputs 15, 16 current				0.3	A
Outputs 3-16 voltage				Vdd-0.5	V
+5V current	Voltage drops if max is exceeded.			0.3	A
Power output current	Voltage drops if max is exceeded.			1	A
Power outputs voltage				Vdd-0.5	V
Operating temperature		0		70	°C
Storage temperature		-40		120	°C

All external pins are protected against destruction by ESD (2kV).

The power supply input is protected against over voltage by a 33V Zener diode (max 3A for 10ms) and against inverted voltage by a serial Schottky diode.
Damages to the device due to over or inverted voltage are not covered by the warranty.

2 Overview

Applications

The FMod-TCP BOX 2 is a TCP/IP server that allows system integrators to connect different devices such as home appliances, industrial sensors and industrial control systems directly to the Ethernet network, (10Mbits 10BaseT) and to remotely monitor and control those using standard protocols.

It can either be accessed through a TCP socket connection, from a computer, or through a simple Web Page in a standard browser which can be directly loaded to the device (max 108kb). The module is delivered with a default web page including a Java Applet that enables the controlling of the device.

The connection between this device and the user's product can be done through the following interfaces:

Qty	Type	Description	Port
1	RS232	Up to 115200 bps with or without hardware flow control	TCP 8000
2	Relays	Cut up to 50V 1A	TCP 8010 UDP 7010
14	24V Outputs	High side drivers	
16	24V Inputs	Digital 24V inputs or analog -15 V to +28 V on 10 bits A/D	
1	I2C bus	2 wires serial bus	

Note:
Some examples of applications illustrating the use of the FMod-TCP family with serial, I2C, I/O and A/D devices can be found on the FiveCo's website: http://www.fiveco.ch/prod_products.php?getcatid=1

Software operating principle

The operating principle for PC softwares that must exchange data with an FMod-TCP BOX 2 device depends on which interface is used.

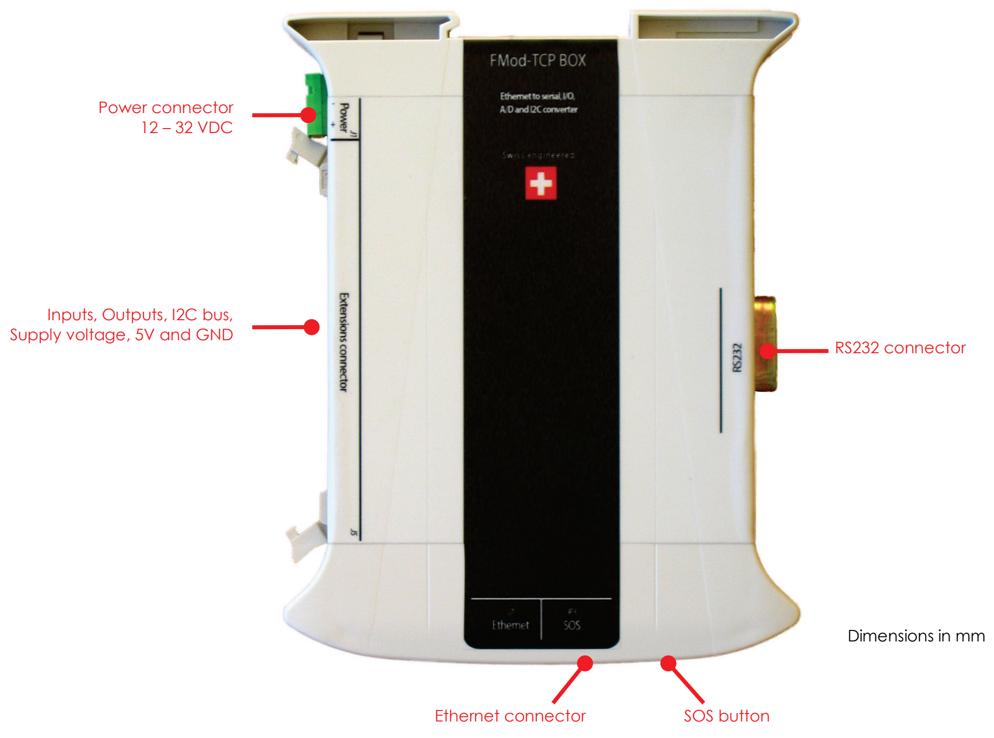
In case of **RS232** use, the operation is really simple. Any byte sent to port TCP #8000 will be redirected to the serial bus and any byte received from the serial bus will be redirected to the TCP connection.

In the other cases (**I/O, A/D and I2C**), the software has to use a dedicated protocol layer on top of the TCP Layer (see chapter 4). This protocol is "Question & Answer" oriented. The PC should send a Question, wait for the Answer and so on.

To configure the device's parameters and to access I/O and A/D features, the protocol uses an Internal Registers Access routine (see chapter 4 and 7).

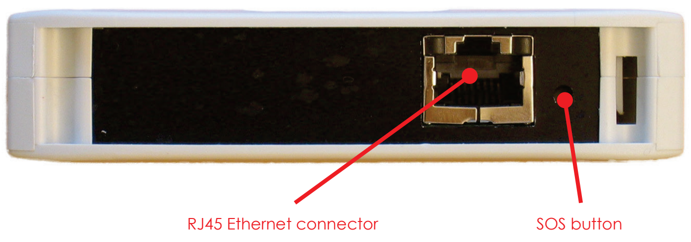
The code samples available on the FiveCo's web site can help programmers get started with development.

Hardware description


FMod-TCP BOX
Ethernet to serial, I/O, A/D and I2C converter
Power connector 12 – 32 VDC
Power
Inputs, Outputs, I2C bus, Supply voltage, 5V and GND
Extensions connector
RS232
RS232 connector
Ethernet
SOS
Dimensions in mm
Ethernet connector
SOS button

Beware:
Before connecting any cable to the device, shut down power supply!

Front view

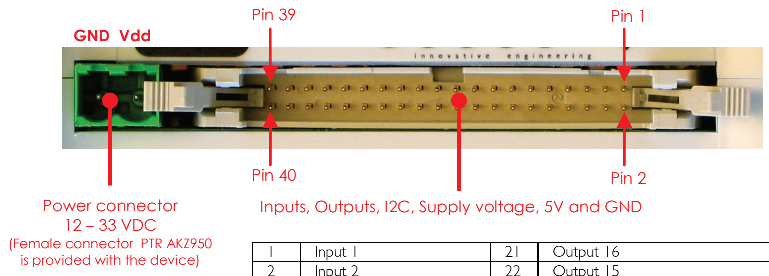


See page 10 to know how the SOS button works.

Two LEDs illuminate the SOS button and the displayed color as the following meanings:

Green	Everything is normal.
Red	There is an error. See Warning register to know the source of the error.
Red-Green blinking	Another device with the same IP address was found on the network. Disable the other device and reboot the FMod-TCP BOX 2.

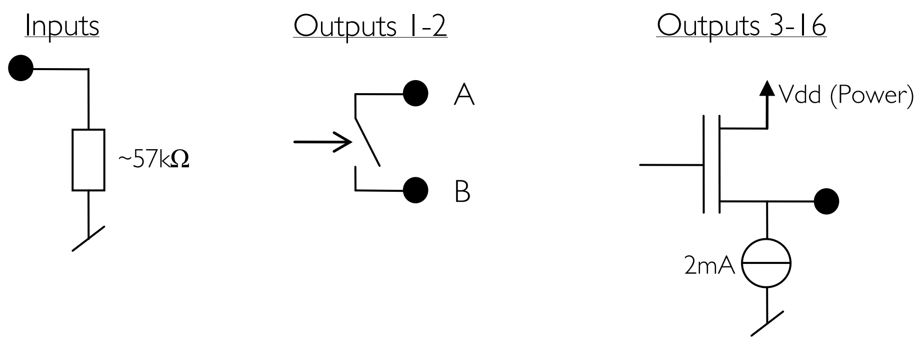
Left side


GND Vdd
Pin 39
Pin 1
Pin 40
Pin 2
Power connector
12 – 33 VDC
(Female connector PTR AKZ950
is provided with the device)
Inputs, Outputs, I2C, Supply voltage, 5V and GND

1	Input 1	21	Output 16
2	Input 2	22	Output 15
3	Input 3	23	Output 14
4	Input 4	24	Output 13
5	Input 5	25	Output 12
6	Input 6	26	Output 11
7	Input 7	27	Output 10
8	Input 8	28	Output 9
9	Input 9	29	Output 8
10	Input 10	30	Output 7
11	Input 11	31	Output 6
12	Input 12	32	Output 5
13	Input 13	33	Output 4
14	Input 14	34	Output 3
15	Input 15	35	Output 2 A (relay 2)
16	Input 16	36	Output 1 A (relay 1)
17	+5V output (0.3A max) [a]	37	Output 2 B (relay 2)
18	GND	38	Output 1 B (relay 1)
19	I2C clock (SCL)	39	Supply GND
20	I2C data (SDA)	40	Supply voltage output (max 1A) [a]

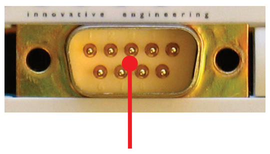
[a]Do not supply the device through these pins. You can use this pin to power your application.

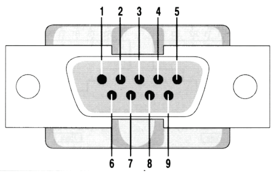
Here are the equivalent electrical specifications of I/O pins (illustrated by black dots):



- Inputs are seen from the outside as 57kΩ loads.
- Relay outputs 1-2 are only switches and are not connected to any internal power lines. You can so cut/connect any line of your application by using both side of the relay (beware: respect the maximum voltage gap between your line and the device GND !).
- Outputs 3-16 either provide Vdd power when they are HIGH or sink a current of 2mA when LOW.
- The I2C pins are provided through a Philips PCA9514AD driver chip. 10kΩ pull-ups are connected to SCL and SDA.
- WARNING: The capacitance of the I2C cable must be lower as possible to avoid errors dues to cross talk. Use also shorter cable as possible.

Right side


Standard RS232 DSub 9 connector
(male like on a computer)



1	NC	6	NC
2	Receive data	7	Request To Send
3	Transmit data	8	Clear To Send
4	NC	9	NC
5	GND		

The RS232 connector is the same as the ones found on any computer.

SOS button

A button is dedicated to restore default IP address or factory settings.

There are tow possible scenarios:

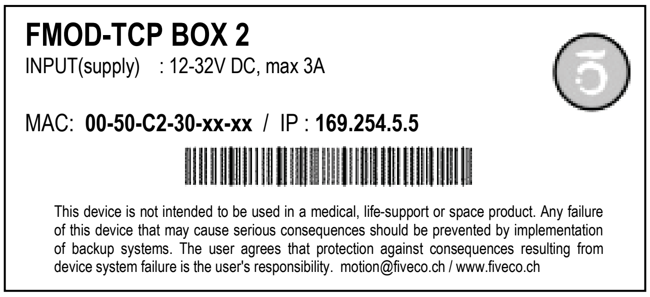
- If you press it **while the device is running**, the IP address will be restored as soon as all TCP connections are closed. *Warning: you have to send the Save Settings command to the device in order to save it into EEPROM.*
- If the **button is pressed during startup**, the default IP address and factory settings of all parameters will be restored AND saved into

EEPROM. In this case, you do not need to send the Save Parameters command.

3 Quick start

This section is intended to help users to quickly plug the module into their system and establish a connection between the computer and the device.

You can find the device's factory communication settings on the following label.



FMOD-TCP BOX 2
INPUT(supply) : 12-32V DC, max 3A

MAC: **00-50-C2-30-xx-xx** / IP : **169.254.5.5**

This device is not intended to be used in a medical, life-support or space product. Any failure of this device that may cause serious consequences should be prevented by implementation of backup systems. The user agrees that protection against consequences resulting from device system failure is the user's responsibility. motion@fiveco.ch / www.fiveco.ch

The Ethernet MAC Address is fixed and cannot be changed. The IP Address can be changed. The complete procedure is described below.

Note: *If the device has already been configured and the IP address has been changed to an unknown value, you can retrieve an SOS IP address (the one on label) by pressing the "SOS button" during the normal operation of the device.* (See page 10).

Changing IP address

To easily change the factory IP address, use the Win32 software provided on the CD-Rom.

1. Plug your new device on your PC network.
2. Start the Win32 application.
3. Click on "File->Easy change IP address".
4. The software will scan the network and display a list of all FiveCo's devices found.
5. Select the MAC address corresponding to your new device.
6. If you have more than one network adapter on your PC, the software will ask you to select the one which is connected to the same network as the FMod-TCP BOX 2.
7. The software will suggest a new IP address with the last byte left open. Choose a new IP (**Not already used on your network!!**) and click the "Change IP address" button.

That's it! The device has a new address and a new subnet mask (the same as your PC). They are automatically saved into EEPROM.

You can now connect the device with the Win32 software or open its web page by typing its new IP address into a web browser.

Notes:

- The IP address won't be changed if a TCP connection exists with the device.
- The protocol used to change the IP address is described later in this manual.

4 Controlling the FMod-TCP BOX 2 by TCP or UDP

General Information

All the device's parameters (configuration registers) and features can be accessed through a TCP or UDP port.
In addition, an HTTP-TCP port is available for web pages downloading and another TCP port for RS232 bus access.

Those ports are:

- TCP Port **#80** for HTTP communication.
- TCP Port **#8000** for RS232 transceiver.
- TCP Port **#8010** to access I/O registers (see chapter 7) and I2C bus.
- UDP Port **#7010** to access I/O registers (see chapter 7) and I2C bus.

With regards to TCP connections, the device allows up to **4 simultaneous connections**.

These ports are described below.

RS232 (TCP # 8000)

The RS232 bus of the microcontroller is accessible through the TCP port number 8000. The module acts simply as a transceiver for this port. Any byte sent from the network (ex: TCP-IP from a PC) to the module will be sent to the other side's RS232 bus, and vice versa. Thus there is no particular protocol dedicated to this feature. See later chapters to know how to change parameters such as baud rate and hardware flow control.
Note: This port supports only one user at a time.

RS232 fixed settings: *No parity / 1 Start Bit / 8 Data Bits / 1 Stop Bit*

Important note about baud rate greater than 9600bds:

Common TCP/IP stacks (on PC, Unix station ...) use a delay of 200ms for the acknowledgement of the TCP received data packets. This is done to reduce traffic on the network because TCP allows the acknowledgement of several packets at one time.

Unfortunately, the FMod-TCP BOX 2 device needs this acknowledgement to remove the data from its internal RS232 receive buffer (if no acknowledge is received from the PC, the module will resend those data after a timeout).

So, with speeds greater than 9600bds, the buffer may be filled faster than data can be sent by TCP and part of those will be lost if no RS232 hardware flow control is used between the FMod-TCP BOX 2 and the RS232 device (CTS and RTS lines).

If you cannot use hardware flow control on RS232 bus and you have to get more than 2048 bytes at one time, the solution is to reduce the TCP acknowledgement delay on your computer.

For Windows^{TM} 2000/XP users, you can add/change the following value in the registry. **BEWARE**: *improperly changes done in the Windows registry can results in a system crash! Such changes are the user's full responsibility!*

Entry:	*HKey_Local_Machine\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\**Interface-name***
Key:	*TcpDelAckTicks*
Value:	*DWORD with value 0x00*

The **interface-name** is the registry name (32 digits number) of your Ethernet card which you use to access the module.
The following web page describes this feature in details:

http://www.microsoft.com/resources/documentation/Windows/2000/server/reskit/en-us/regentry/58801.asp?frame=true

Device parameters and I/O, A/D and I2C features (TCP # 8010 or UDP # 7010)

The main TCP port number 8010 or UDP port number 7010 can be used to change some important parameters of the module:

- **TCP timeout value**
- **IP address**
- **Subnet mask**
- **Module name**
- **Baud rate and flow control**
- **I2C speed**

The user should use the Win32 application enclosed in the package or the default java applet loaded in the module to change those parameters. If the user wants to change the parameters by himself, the protocol is defined below.

This port is also used to **access Inputs and Outputs value, A/D conversion result registers** (see page 35 for a complete description of those registers) **and I2C bus.** The I2C feature is described after "registers access feature".

The last feature accessible through this port is the "Easy IP config" that is used in the "Quick start" chapter of this document.

Registers access feature

TCP/IP works in big endian: most significant byte first, followed by least significant byte.
The access to the data is done through an easy (6 byte header) protocol over TCP.

Structure of each packet:

1. **Function ID (2 bytes),**
2. **Transaction ID (2 bytes)**
3. **Length of the parameters (2 bytes)**
4. **Parameters (X byte)**
5. **Checksum (2 bytes) (described later in this chapter)**

The user (sender) defines the values of the Transaction IDs himself. The module that receives a command sends back an answer (for every command). The answer contains the same Transaction ID as the corresponding command sent. The user is also able to check execution of each command.
Warning: **The maximum length of parameters is 172** on this board! Pay attention to not exceed this value in question AND answer frame. This implies that the maximum registers that can be read at one time are almost 30 (depending on registers size). If the number of registers is too big, the FMod-TCP DB will answer only with the value of some of them. **It is also mandatory that the command has to be transmitted within one TCP packet.** Otherwise, the FMod-TCP DB will ignore it.

Read register(s) command:

Byte#		Number of bits	Example
0x00	Read (0x0021)	16 bits	0x0021
0x02	TransactionID	16 bits	0x1B34
0x04	Number of registers to read (X)	16 bits	0x0001
0x06	X * Registers Addresses	X * 8 bits	0x02
0x06+X	Checksum	16 bits	0x...

Read register(s) answer:
The device answers with the following sequence:

Byte#		Number of bits	Example
0x00	Read Answer (0x0023)	16 bits	0x0023
0x02	TransactionID (same as demand)	16 bits	0x1B34
0x04	Number of bytes in answer	16 bits	0x0019
0x06	Register address	8 bits	0x02
…	Register value	8–128 bits (16B)	0x12345
The two previous entries are replicated for every register that has been asked for reading			
…	Checksum	16 bits	0x…

Write register(s) command:

Byte#		Number of bits	Example
0x00	Write (0x0022)	16 bits	0x0022
0x02	TransactionID	16 bits	0x1B34
0x04	Number of bytes in command	16 bits	0x0003
0x06	Register Addresses	8 bits	0x02
0x07	Register value	8 – 64 bits	0x1234
The two previous entries are replicated for every register that has been asked for reading			
…	Checksum	16 bits	0x…

Write register(s) answer:
The device answers with the following sequence:

Byte#		Number of bits	Example
0x00	Write Answer (0x0024)	16 bits	0x0024
0x02	TransactionID (same as demand)	16 bits	0x1B34
0x04	0x0000	16 bits	0x0000
0x06	Checksum	16 bits	0x…

I2C feature

The Inter-IC bus, commonly known as the I^2C bus, is a control bus that provides the communications link between integrated circuits in a system. Developed by Philips in the early 1980's, this simple two-wire bus has become the de facto worldwide standard for system control, finding its way into everything from temperature sensors and voltage level translators to EEPROMs, general-purpose I/O, A/D and D/A converters, CODECs, and microprocessors of all kinds.
You can find the I2C's specifications on the Philips web site at the following link: http://www.semiconductors.philips.com/

The I2C protocol can access a device by three different manners:

- **Write** (Start, AddW, Byte1, Byte2, ..., Stop)

St	AddW	A	Bytes to W	A	Sp

- **Read** (Start, AddR, Byte1, Byte2, ..., Stop)

St	AddR	A	Bytes to R	nA	Sp

- **Read After Write** (Start, AddW, ByteW1, ByteW2, ..., ReStart, AddR, ByteR1, ByteR2, ...Stop)

St	AddW	A	Bytes to W	R	A	AddR	A	Bytes to R	nA	Sp

I2C Read/Write with ack command:
To be able to do all of these 3 sequences, use this command:

Byte#		Number of bits	Example
0x00	I2CRWwithAck (0x0007)	16 bits	0x0007
0x02	TransactionID	16 bits	0x1B34
0x04	LengthOfParameters (X + 3)	16 bits	0x0005
0x06	7 bits Address (bit 7 = 0)	8 bits	0x28
0x07	X (number of bytes to write)	8 bits	0x02
0x08	xBytes	X bytes	0xAF1D
....	Y (number of bytes to read)	8 bits	0x05
The four previous entries can be replicated to access the same or other I2C slaves within this command sequence.			
	Checksum	16 bits	0x...

The FMod-TCP DB translates automatically those commands to I2C sequences:

- If X = 0, the Read method is used.
- If Y = 0, the Write method is used.
- If X & Y ≠ 0, the Read after Write method is used.

Warning: Please check that $2X + 3Y \leq 240$. In other case, the command will be ignored by the board.

Note: the max length of parameters is 172 bytes on this board. **Pay close attention to building sequences that do not exceed this and not to ask too much byte in answer! It is also mandatory that the command has to be transmitted within one TCP packet.** Otherwise, the FMod-TCP DB will ignore it.

I2C Read/Write with ack answer:
The answer sequence is the following one:

Byte#		Number of bits	Example
0x00	I2CReadAnswer (0x0008)	16 bits	0x0008
0x02	TransactionID (same as demand)	16 bits	0x1254
0x04	Number of bytes in answer	16 bits	0x0005
0x06	Answer bytes	Y bytes	0x1A25…
…	Ack state of the I2C com.	1 byte	0x87
If the same or other I2C slave have been accessed in the command, the answer bytes and ack state is added here.			
…	Checksum	16 bits	0x…

The "Ack state" byte is composed of the following bits:

0	Address ack in write sequence	0 = No answer to this address 1 = ack received
1	Bytes written ack (each byte was acked)	0 = Bytes not acknowledged 1 = ack received
2	Address ack in read sequence	0 = No answer to this address 1 = ack received
3-6	Reserved	-
7	Must be always 1	1

The user can use these bits to check for the presence of his I2C devices and monitor hardware issues.

I2C Read/Write with ack error answer:
If an error occurred, the answer sequence is the following one:

Byte#		Number of bits	Example
0x00	I2CRWwithAckError (0x0009)	16 bits	0x0009
0x02	TransactionID (same as demand)	16 bits	0x1254
0x04	Number of bytes in answer	16 bits	0x0001
0x06	Error code	1 byte	0x01
0x07	Checksum	16 bits	0x…

The error code can be:

0x01	The I2C bus is not ready.
0x02	A timeout occurred when waiting answer from the I2C slave.
0x03	2X + 3Y > 240 in I2CRWwithAck received.

I2C Bus scanning

The following command allows user to ask an I2C bus scanning to list which addresses answer with an acknowledge.

I2C scan command:

Byte#		Number of bits	Example
0x00	I2CScan (0x0005)	16 bits	0x0005
0x02	TransactionID	16 bits	0x2001
0x04	Number of addresses to scan	16 bits	0x0001
0x08	X Addresses	X bytes	0x1A
	Checksum	16 bits	0x...

Note: If there is no address in the I2C Scan command, the FMod-TCP BOX 2 will scan all addresses between 1 and 127!

I2C scan answer:
The FMod-TCP BOX 2 answers with the following sequence:

Byte#		Number of bits	Example
0x00	I2CScanAnswer (0x0006)	16 bits	0x0006
0x02	TransactionID (same as demand)	16 bits	0x2001
0x04	Number of valid addresses	16 bits	0x0001
0x06	Valid addresses list	n bytes	0x1A
	Checksum	16 bits	0x...

I2C speed change (advanced features)

The I2C bus speed can be changed at any time by changing the content of the I2CSPD parameter.

Beware:

1. Do not change I2C speed if it is not mandatory. Speed greater than 100kHz are not supported by all I2C slaves.
2. Do not change I2C speed during I2C communication.
3. The I2C controller does not match all I2C specifications at speed higher than 100kHz. This feature will therefore not work with all I2C slaves.
4. The FMod-TCP BOX 2 does not support speed greater than 400kHz.

The value of the I2CSPD parameter must be computed with the following formula:

$$I2CSPD = \frac{10^7}{I2CSpeed_{wanted}} - 1$$

The I2C speed cannot be saved in EEPROM.

Easy IP address config (UDP # 7010)

A really useful feature of the UDP port #7010 is the "Easy IP config" one.

The user who wants to design his own software can use this feature to do a "quick start/install" method. Indeed, since this protocol uses a broadcast UDP packet, even if the device is not in the same subnet, it should receive its new IP address and subnet mask.

Procedure:
Send a UDP broadcast message (using a local or direct broadcast IP address) to your network (inside which the FMod-TCP BOX 2 is connected) with the following command:

Byte#		Number of bits	Example
0x00	Change IP fct (0x002A)	16 bits	0x002A
0x02	TransactionID	16 bits	0x0000
0x04	Length of params (0x000E)	16 bits	0x000E
0x06	Device Mac Address	6 bytes	0x0050C2308101
0x0C	Device new IP Address	4 bytes	0xC0A81064
0x10	Device new SubnetMask	4 bytes	0xFFFF0000
0x14	Checksum	16 bits	0x...

If the FMod-TCP BOX 2 recognizes its MAC address, it will answer this command with a simple acknowledges and change its IP address and subnet mask IF NO TCP CONNECTION IS ESTABLISHED WITH THE DEVICE.

Byte#		Number of bits	Example
0x00	Change IP fct ack (0x002B)	16 bits	0x002B
0x02	TransactionID	16 bits	0x0000
0x04	Length of params (0x0000)	16 bits	0x0000
0x14	Checksum	16 bits	0x...

Checksum calculation

This checksum is the same as the IP checksum.

Definition: sum of 1's complement of all 16 bits words of whole message (FiveCo packet) except checksum bytes.

Note: all values are unsigned!

Sequence:

1. Clear accumulator

Loop

x. Only if last word is not made of two bytes, the data byte is the upper byte (big endian)
2. Compute 1's complement of each 16bits word, result is 16bits
3. Convert last result from 16 bits to 32 bits, result is 32bits: 0x0000+last result
4. Add last result to the 32 bits accumulator

Try the Loop

5. Convert accumulator in two 16bits words
6. Add those two 16bits words, result is 16bits word.
7. If an overflow occurs with the last addition (Carry), add 1 to the last result.
8. Last result is the final result

Example (in hexadecimal):

```
 !0x0021  (0XFFDE) → 0x0000FFDE (Read)
+!0x1234  (0xEDCB) → 0x0001EDA9 (TransID)
+!0x0003  (0xFFFC) → 0x0002EDA5 (3 reg to read)
+!0x0A10  (0XF5EF) → 0x0003E394 (reg 0A,10,02)
+!0x02(00)(0XFDFF) → 0x0004E193
```

Note that in this case a last 00 is implicitly used. (02 → 02 00).

```
0x0004 + 0xE193 = 0xE197, (carry=0)
0xE197 + carry  = 0xE197
```

Checksum = 0xE197

Here is an example of a checksum calculation function in C:

```
int RetCheckSum(Byte* ByteTab, int Size)
{
    // This function returns the calculated checksum
    unsigned int     Sum=0;
    bool             AddHighByte=true;
    unsigned int     ChecksumCalculated;

    for(int i=0;i<Size;i++)
    {
        if(AddHighByte)
        {
            Sum+=((ByteTab[i])<<8)^0xFF00;
            AddHighByte=false;
        }
        else
        {
            Sum+=(ByteTab[i])^0x00FF;
            AddHighByte=true;
        }
    }
    if (AddHighByte==false)
    Sum+= 0xFF;

    ChecksumCalculated = ((Sum>>16)&0xFFFF)+(Sum&0xFFFF);

    ChecksumCalculated = ((ChecksumCalculated>>16)&0xFFFF)
                   +(ChecksumCalculated&0xFFFF);

    return ChecksumCalculated;
}
```

This function needs a Byte array (ByteTab) containing the command sequence and this array's length (Size) as input, it returns the checksum as an int.

5 Java Applet

A specific Java Applet is provided with the module to control any of its ports without having to write any specific code.

Beware:
Since year 2015, internet browsers providers are removing Java Applet support from their software. Nowadays (Q2 2017), Only MS Internet Explorer and Apple Safari still support java applets!
Use of Java applet in your application is so no more recommended.

Overview

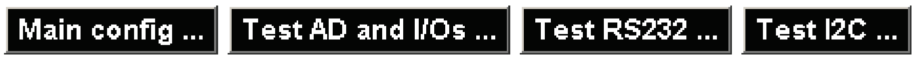
To connect to the http server on the device, simply open your web browser and type the IP address of the module. Example with default address:

"http://169.254.5.5"

The applet is downloaded from the device to your computer and runs as a local process (on your computer). You need to use an internet web browser that is compatible with Java 1.1. Most of browsers support Java 1.1 without plugin, but if yours does not display the applet, please download the Sun Java plugin at the following address: http://www.java.com/

Please note that on an MSWindows™ based computer, a few seconds delay can occur when you download the applet due to an MSWindows™ NetBios issue.

The navigation through the four panels of the applet is done through the menu bar:

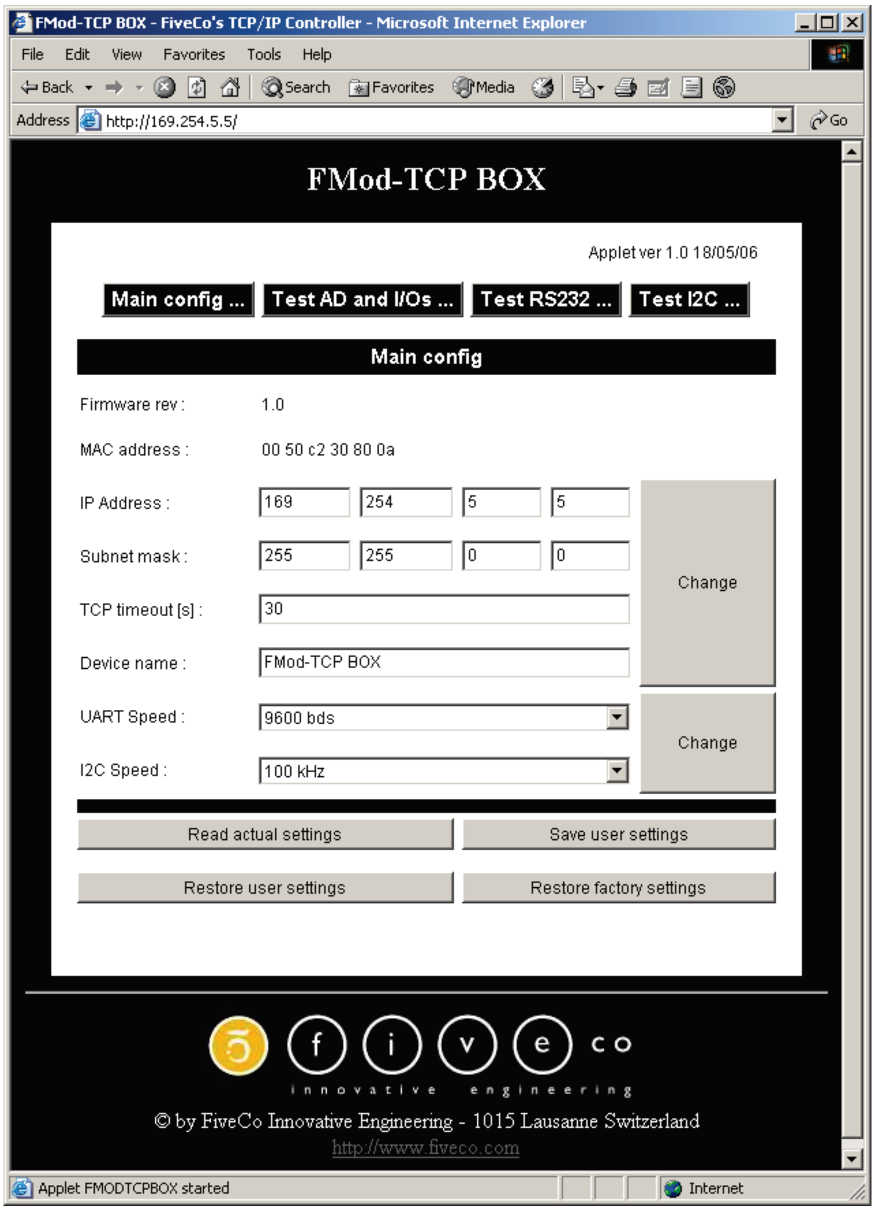


Main Config

The Main panel shows the general information related to the device.

• The first part allows the user to change the main settings of the device. Don't forget to use the "Save user parameters" button to make changes permanent!

• The second part allows the user to save/restore user and factory parameters and to read the actual configuration of the device if it has been modified by another application.


FMod-TCP BOX - FiveCo's TCP/IP Controller - Microsoft Internet Explorer

Address http://169.254.5.5/

FMod-TCP BOX

Applet ver 1.0 18/05/06

Main config ... | Test AD and I/Os ... | Test RS232 ... | Test I2C ...

Main config

Firmware rev :	1.0
MAC address :	00 50 c2 30 80 0a
IP Address :	169 254 5 5
Subnet mask :	255 255 0 0
TCP timeout [s] :	30
Device name :	FMod-TCP BOX
UART Speed :	9600 bds
I2C Speed :	100 kHz

Change

Change

Read actual settings | Save user settings

Restore user settings | Restore factory settings

fiveco innovative engineering

© by FiveCo Innovative Engineering - 1015 Lausanne Switzerland

http://www.fiveco.com

Applet FMODTCPBOX started

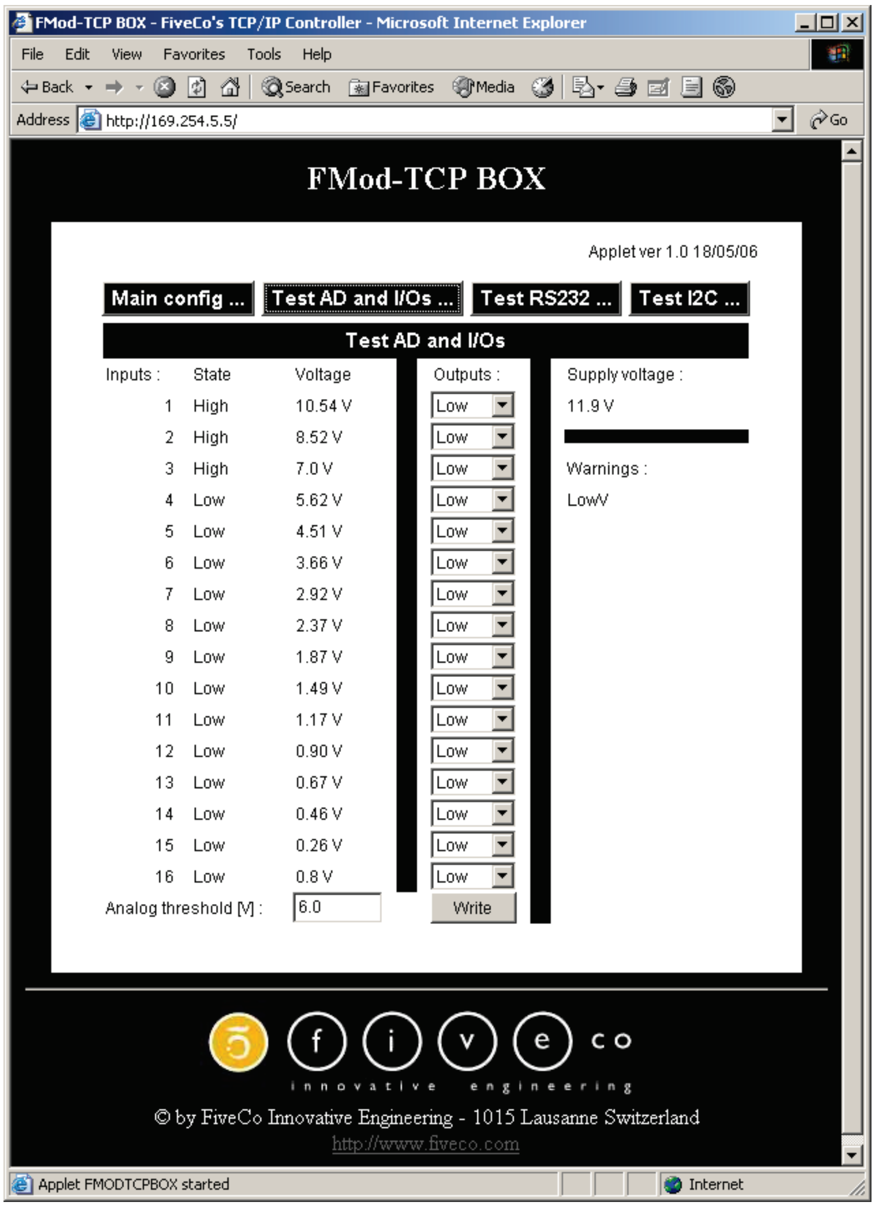
Internet

Test A/D and I/Os

This page can be used to access Inputs state and voltage and change Outputs state.

The Inputs state and voltage are regularly updated automatically (20 kHz). You can change the low to high threshold of the inputs between 0 and 24V.

This page displays also the device supply voltage and warnings (Too low/high supply voltage, Outputs driver over current or temperature).


FMod-TCP BOX - FiveCo's TCP/IP Controller - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://169.254.5.5/
Go

FMod-TCP BOX

Applet ver 1.0 18/05/06

Main config ...
Test AD and I/Os ...
Test RS232 ...
Test I2C ...

Test AD and I/Os

Inputs :	State	Voltage	Outputs :
1	High	10.54 V	Low
2	High	8.52 V	Low
3	High	7.0 V	Low
4	Low	5.62 V	Low
5	Low	4.51 V	Low
6	Low	3.66 V	Low
7	Low	2.92 V	Low
8	Low	2.37 V	Low
9	Low	1.87 V	Low
10	Low	1.49 V	Low
11	Low	1.17 V	Low
12	Low	0.90 V	Low
13	Low	0.67 V	Low
14	Low	0.46 V	Low
15	Low	0.26 V	Low
16	Low	0.8 V	Low

Analog threshold [V] : 6.0

Supply voltage : 11.9 V

Warnings : LowV

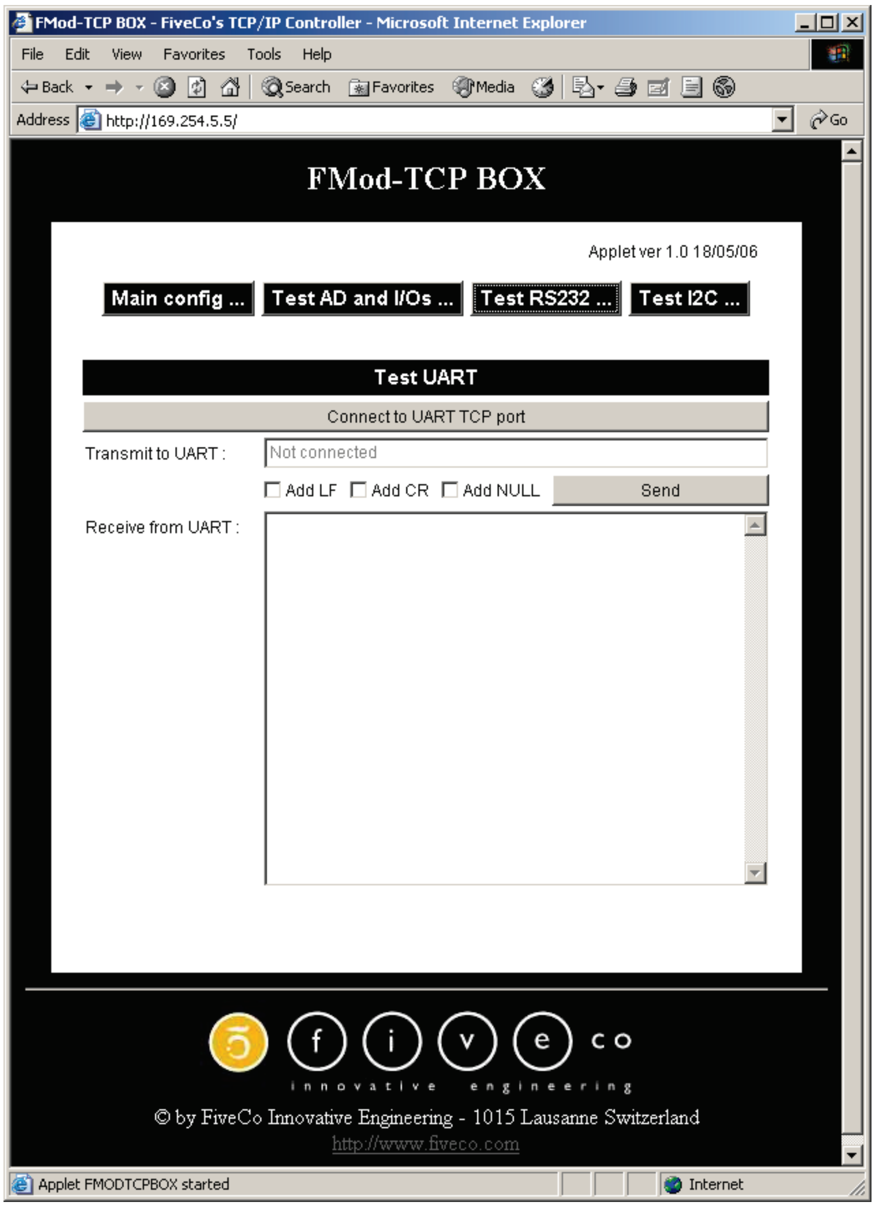
five co
innovative engineering

© by FiveCo Innovative Engineering - 1015 Lausanne Switzerland
http://www.fiveco.com

Test RS232

This page can be used to access RS232 bus.

If you click on the "Connect to RS232 TCP Port" you can send ASCII data to RS232 bus and receive data from it (only ASCII is visible in this application).

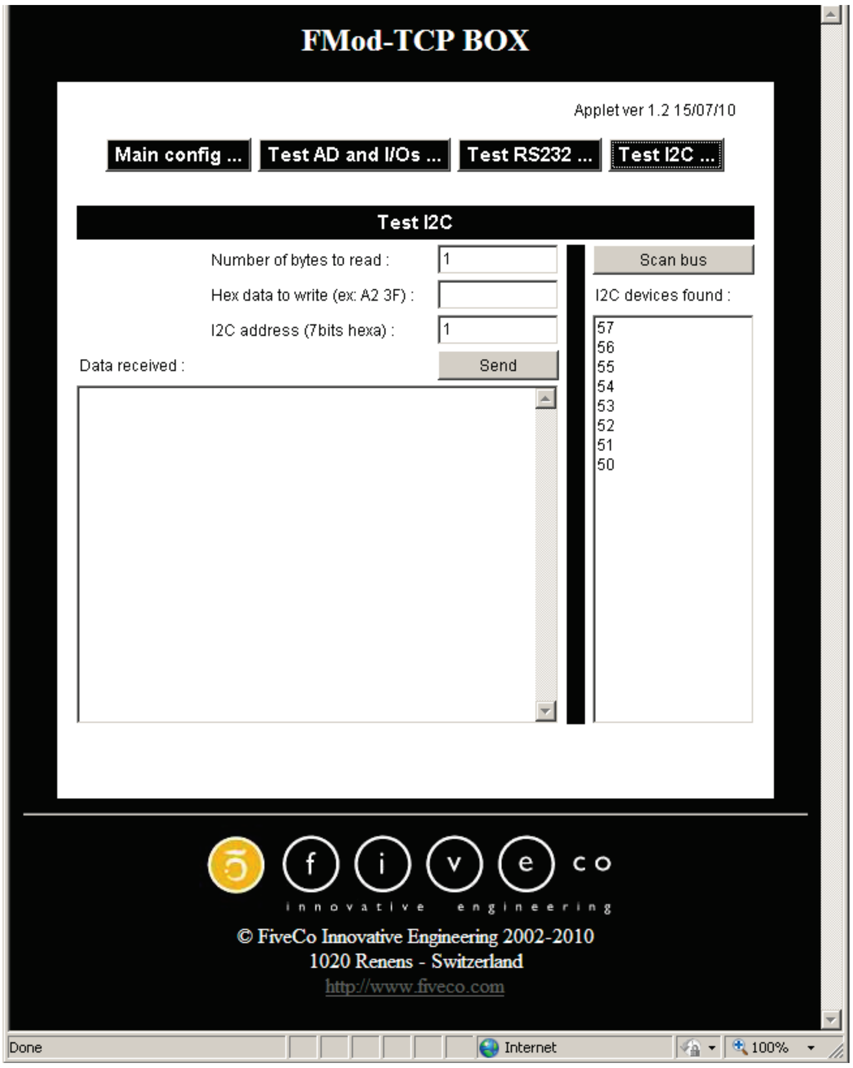

FMod-TCP BOX - FiveCo's TCP/IP Controller - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://169.254.5.5/
Go
FMod-TCP BOX
Applet ver 1.0 18/05/06
Main config ...
Test AD and I/Os ...
Test RS232 ...
Test I2C ...
Test UART
Connect to UART TCP port
Transmit to UART :
Not connected
Add LF
Add CR
Add NULL
Send
Receive from UART :
© by FiveCo Innovative Engineering - 1015 Lausanne Switzerland
http://www.fiveco.com
Applet FMODTCPBOX started
Internet

You can check "Add LF", "Add CR" and/or "Add Null" boxes to add a Line Feed, a Carriage Return and/or a Null Byte at the end of the ASCII chain sent to the device.

Test I2C

This page can be used to access I2C bus. It is divided in two parts:

- The part on the right allows scanning the I2C bus to find available slave devices.
- The part on the left allows sending a read or a write command to a device on the I2C bus.



FMod-TCP BOX

Applet ver 1.2 15/07/10

Main config ... | Test AD and I/Os ... | Test RS232 ... | Test I2C ...

Test I2C

Number of bytes to read : 1

Hex data to write (ex: A2 3F) :

I2C address (7bits hexa) : 1

Data received :

Send

Scan bus

I2C devices found :

```
57
56
55
54
53
52
51
50
```

© FiveCo Innovative Engineering 2002-2010
1020 Renens - Switzerland
http://www.fiveco.com

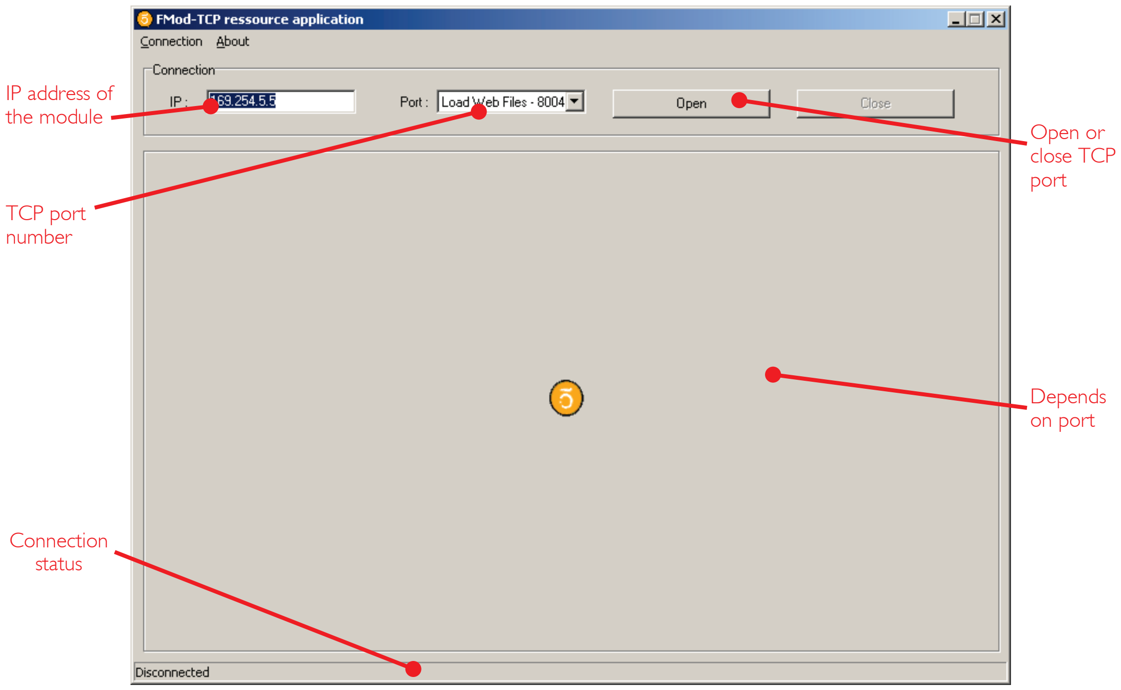
Done | Internet | 100%

You should first check if you want a read or a write. Then you have to write hex data to write to the device in the "Hex data to write" field (if you checked the "Read" box, the FMod-TCP BOX 2 will use the read after write I2C feature).
After writing the I2C address (7bits hexa) and the number of bytes to read (if applicable), click on the "Send" button. The answer is displayed in hex in the "Data received" field.

6 Win32 Application

A specific application is provided with the module to control any of its ports without having to write a specific code.

Overview

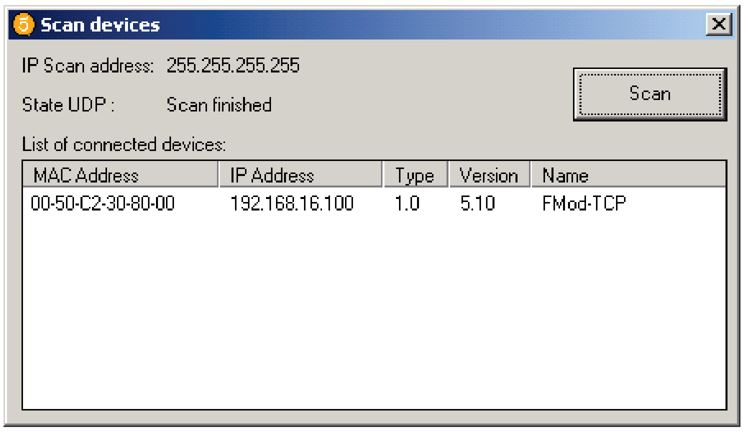


To open a TCP port on the module, the user has to set the correct IP address of the module, to choose the correct TCP port in the list and to click on the "Open" button. To close the port, simply click on the "Close" button.

The status bar displays the status of the connection:

- ***Connected***
- ***Disconnected***
- ***Error of connection*** (if connection was not established within 30s)

Note: *If you forgot the IP address of the module, you can use the "Scan network" feature of the "Connection" menu.*



IP Scan address: 255.255.255.255

State UDP : Scan finished

List of connected devices:

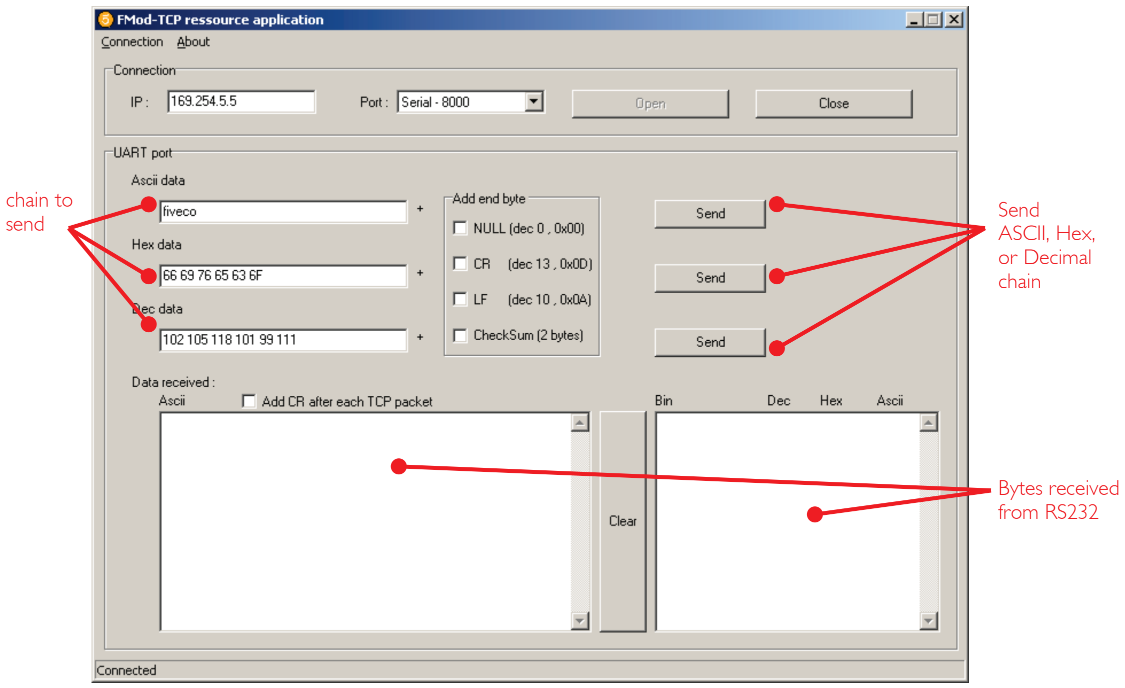
MAC Address	IP Address	Type	Version	Name
00-50-C2-30-80-00	192.168.16.100	1.0	5.10	FMod-TCP

This application uses the IP address of your computer and its subnet mask to find the scan broadcast address. When you click on scan, a broadcast message will be sent to all devices in your subnet and answers will be listed.

WARNING: it only works with devices in your subnet!

<u>How does it work?</u>
When you click "Scan", the software simply sends a "Read registers" command to a broadcast address on UDP port number 7010 (see chapter about main port at page 15).

The broadcast address depends on the network subnet mask present on your PC.



FMod-TCP ressource application

Connection About

Connection

IP : 169.254.5.5

Port : Serial - 8000

Open Close

UART port

Ascii data: fiveco

Hex data: 66 69 76 65 63 6F

Dec data: 102 105 118 101 99 111

Add end byte

☐ NULL (dec 0 , 0x00)

☐ CR (dec 13 , 0x0D)

☐ LF (dec 10 , 0x0A)

☐ CheckSum (2 bytes)

Send Send Send

Data received :

Ascii ☐ Add CR after each TCP packet

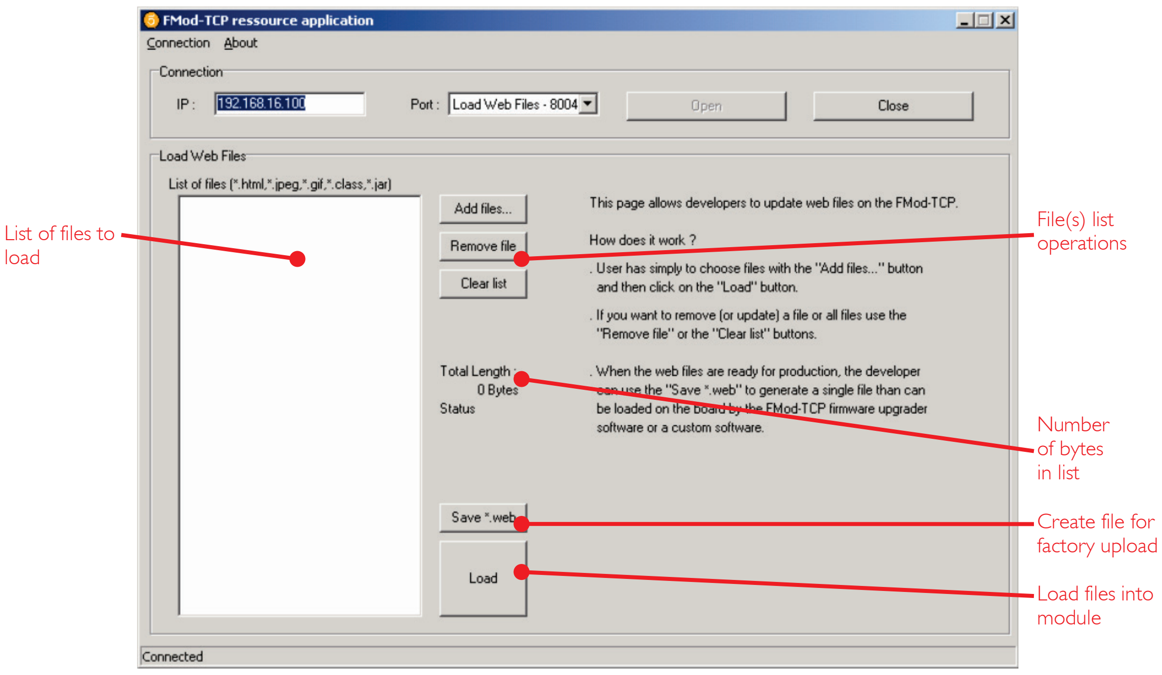
Clear

Bin Dec Hex Ascii

Connected

The RS232 interface is easy to use. Just write the ASCII, HEX or decimal string in the dedicated text box and click corresponding "Send" button. The received bytes are displayed into the "Data received" boxes (same data but different representation).

“Load web files” interface


FMod-TCP ressource application

Connection About

Connection

IP : 192.168.16.100 Port : Load Web Files - 8004 Open Close

Load Web Files

List of files (*.html,*.jpeg,*.gif,*.class,*.jar)

Add files... Remove file Clear list

This page allows developers to update web files on the FMod-TCP.

How does it work ?

. User has simply to choose files with the "Add files..." button and then click on the "Load" button.

. If you want to remove (or update) a file or all files use the "Remove file" or the "Clear list" buttons.

. When the web files are ready for production, the developer can use the "Save *.web" to generate a single file than can be loaded on the board by the FMod-TCP firmware upgrader software or a custom software.

Total Length : 0 Bytes

Status

Save *.web

Load

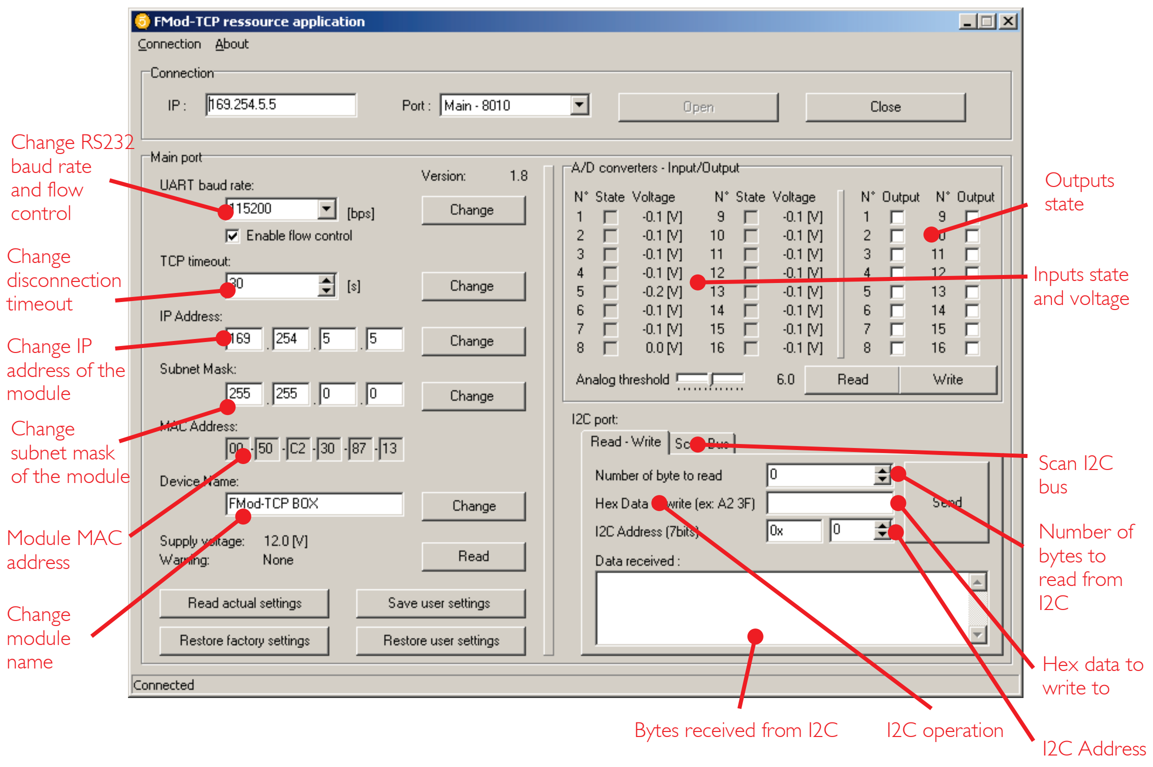
Connected

This option allows users to load their personal web files (html, java, jpeg, gif, txt ...) into the flash memory. A maximum of **44kB** is available for that application.

An example of pages and Default Java Applet code is available on the Starter Kit CDRom or on the FiveCo's web page.

For factory web upload, the user can save a single file yourname.web. See web page upload manual on the FiveCo web site.

Main port interface



Connection

IP : 169.254.5.5 Port : Main - 8010

Main port

UART baud rate: 115200 [bps]

☑ Enable flow control

Version: 1.8

TCP timeout: 30 [s]

IP Address: 169 . 254 . 5 . 5

Subnet Mask: 255 . 255 . 0 . 0

MAC Address: 00 - 50 - C2 - 30 - 87 - 13

Device Name: FMod-TCP BOX

Supply voltage: 12.0 [V]

Warning: None

A/D converters - Input/Output

N°	State	Voltage	N°	State	Voltage
1	☐	-0.1 [V]	9	☐	-0.1 [V]
2	☐	-0.1 [V]	10	☐	-0.1 [V]
3	☐	-0.1 [V]	11	☐	-0.1 [V]
4	☐	-0.1 [V]	12	☐	-0.1 [V]
5	☐	-0.2 [V]	13	☐	-0.1 [V]
6	☐	-0.1 [V]	14	☐	-0.1 [V]
7	☐	-0.1 [V]	15	☐	-0.1 [V]
8	☐	0.0 [V]	16	☐	-0.1 [V]

N°	Output	N°	Output
1	☐	9	☐
2	☐	10	☐
3	☐	11	☐
4	☐	12	☐
5	☐	13	☐
6	☐	14	☐
7	☐	15	☐
8	☐	16	☐

Analog threshold 6.0

I2C port:

Number of byte to read 0

Hex Data to write (ex: A2 3F)

I2C Address (7bits) 0x 0

Data received :

Connected

The configuration interface allows the user to change the device's settings.

RS232 baud rate:	The user can change the RS232 baud rate and enable or disable the hardware flow control (CTS/RTS).
TCP timeout:	The user can change the number of seconds allowed before TCP port is disconnected. This feature avoids the problems due to the crash of a TCP client (PC).
IP address:	The user can change the IP address of the module.
Subnet mask:	The user can change the subnet mask of the module.
MAC address:	Mac address of the module (cannot be changed).
Name:	Module name.

The Inputs state and voltage are simply obtained with a click on the corresponding "Read" button.
The Outputs state and the analog threshold are send to the device with a click on the corresponding "Write" button.

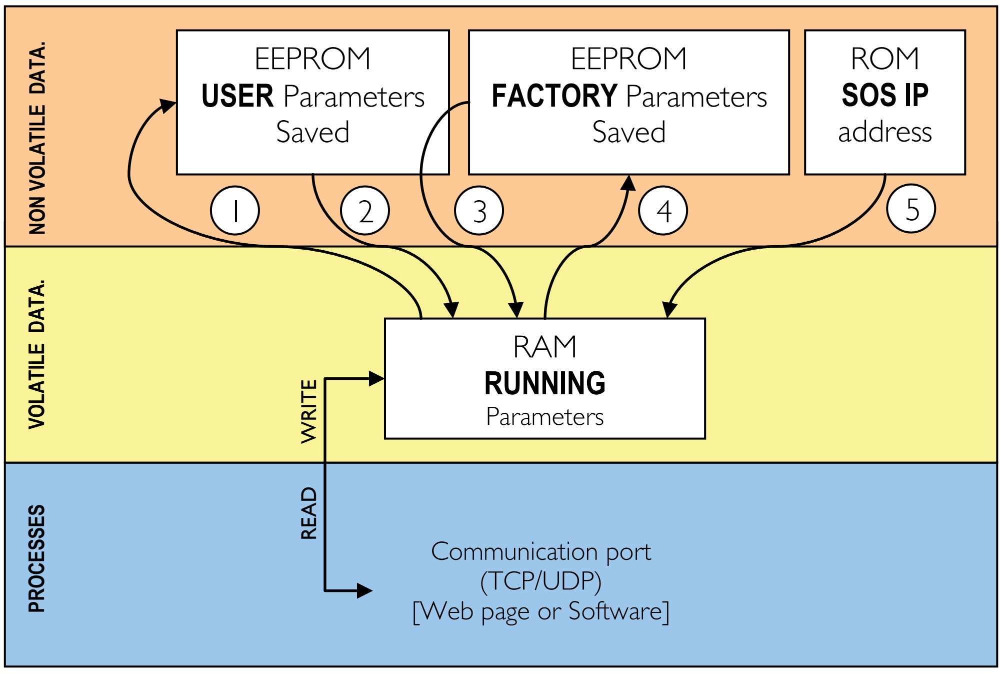
The I2C interface is an easy way to test the I2C connection between the module and the user's electronic.

The bytes to write (in hexadecimal like 41 42 45 separated by spaces!) must be written in the dedicated line and the I2C address in the dedicated Address space (in hexadecimal or in decimal). Specify how many bytes have to be read and then click "Send".

7 Registers management

Memory Organization

The user must know that a new register value sent through the communication port is loaded to the running parameters in RAM and used for the current process. All these parameters are lost upon power-down. It is required to save them to "User Parameters" or "Factory Parameters" with the corresponding function.



Action Number and description:

①	***SaveUserParameters*** (0x03) function
②	During standard power-up or calling ***RestoreUserParameters*** (0x04) function
③	***RestoreFactoryParameters*** (0x05) function
④ + ①	***SaveFactoryParameters*** (0x06) function [For integrators engineers only]
⑤	By setting "SOS Jumper" after power-up
③ + ⑤ + ①	By setting "SOS Jumper" during power-up

List of registers

Address	Bytes	Access type	Name
General Information			
0x00 (00)	4	R	*TYPE*
0x01 (01)	4	R	*VERSION*
0x02 (02)	0 (fct)	Fct	*RESET*
0x03 (03)	0 (fct)	Fct	*SAVE USER PARAMETERS*
0x04 (04)	0 (fct)	Fct	*RESTORE USER PARAMETERS*
0x05 (05)	0 (fct)	Fct	*RESTORE FACTORY PARAMETERS*
0x06 (06)	0 (fct)	Fct	*SAVE FACTORY PARAMETERS*
0x07 (07)	4	R	*VOLTAGE*
0x08 (08)	4	RW	*WARNINGS*
0x0B (11)	4	R	*NB POWER UP*
0x0C (12)	4	R	*TIME IN SERVICE*
Communication			
0x10 (16)	4	RW	*COM OPTIONS*
0x11 (17)	6	R	*ETHERNET MAC*
0x12 (18)	4	RW	*IP ADDRESS*
0x13 (19)	4	RW	*SUBNET MASK*
0x14 (20)	1	RW	*TCP TIMEOUT*
0x15 (21)	16	RW	*MODULE NAME*
0x16 (22)	1	RW	*RS232 CONFIG*
0x18 (24)	1	RW	*I2C SPD**
0x1A (26)	1	R	*TCP CONNECTIONS OPENED*
I/Os and AD			
0x20 (32)	4	RW	*ANALOG INPUTS THRESHOLD*
0x21 (33)	2	RW	*OUTPUTS**
0x23 (35)	2	R	*INPUTS*
Analog voltage at inputs			
0x30 (48)	4	R	*INPUT VOLTAGE 0 VALUE*
0x31 (49)	4	R	*INPUT VOLTAGE 1 VALUE*
0x32 (50)	4	R	*INPUT VOLTAGE 2 VALUE*
0x33 (51)	4	R	*INPUT VOLTAGE 3 VALUE*
0x34 (52)	4	R	*INPUT VOLTAGE 4 VALUE*
0x35 (53)	4	R	*INPUT VOLTAGE 5 VALUE*
0x36 (54)	4	R	*INPUT VOLTAGE 6 VALUE*
0x37 (55)	4	R	*INPUT VOLTAGE 7 VALUE*
0x38 (56)	4	R	*INPUT VOLTAGE 8 VALUE*
0x39 (57)	4	R	*INPUT VOLTAGE 9 VALUE*
0x3A (58)	4	R	*INPUT VOLTAGE 10 VALUE*
0x3B (59)	4	R	*INPUT VOLTAGE 11 VALUE*
0x3C (60)	4	R	*INPUT VOLTAGE 12 VALUE*
0x3D (61)	4	R	*INPUT VOLTAGE 13 VALUE*
0x3E (62)	4	R	*INPUT VOLTAGE 14 VALUE*
0x3F (63)	4	R	*INPUT VOLTAGE 15 VALUE*

* Those registers are not saved to EEPROM and must be configured after each power up.

TYPE

Register Address	Register Name	Function	Read/Write Control
0x00	*TYPE*	Product ID	Read only

Register Size	Register structure	
4 Bytes	Unsigned Int 16bits (HH-HL) *TYPE*	Unsigned Int 16bits (LH-LL) *MODEL*

Description:
Product identifier composed with a *Type* and *Model* number.
It defines which kind of peripheral it is.
Normally different modules *TYPE* are not software compatible.

Example:
TYPE = 0x00080000 means *Type*=8, *Model* = 0 (FMod-TCP BOX)
TYPE = 0x00080000 means Type=8, Model = 2 (FMod-TCP BOX 2)

VERSION

Register Address	Register Name	Function	Read/Write Control
0x01	*VERSION*	Hardware & Firmware rev	Read only

Register Size	Register structure	
4 Bytes	*Hardware Version (8bits.8bits)*	Firmware *Version (8bits.8bits)*

Description:
Hardware and firmware version are composed with a *Version* and *Revision* number.
Same *Version* with different *Revision* is backward compatible.
MSB contains version number and LSB contains revision number.

Example:
0x02000201 = Hardware 2.0 / Firmware 2.1
Hardware Version 2, *Revision* 0
Firmware Version 2, *Revision* 1 is compatible with all earlier revisions of the same version (ver 2.0 in this case). However, it has new functionalities (which are deactivated by default) or code optimizations.

RESET

Function Address	Function Name	Function	Read/Write Control
0x02	*RESET*	Restart device	Write only

Register Size	Register structure	Unit
0 Byte	none	none

Description:
Reboot the device. The communication will be lost.

SAVE USER PARAMETERS

Function Address	Function Name	Function	Read/Write Control
0x03	*SAVEUSERPARAMETERS*	Saves all in EEPROM	Write only

Register Size	Register structure	Unit
0 Byte	none	none

Description:
Saves the following parameters to user EEPROM space:

0x12	*IPADDRESS*
0x13	*SUBNETMASK*
0x14	*TCPTIMEOUT*
0x15	*MODULENAME*
0x16	*RS232CONFIG*

RESTORE USER PARAMETERS

Function Address	Function Name	Function	Read/Write Control
0x04	*RESTOREUSERPARAMETERS*	Restores saved values	Write only

Register Size	Register structure	Unit
0 Byte	none	none

Description:
Restores the following parameters from user EEPROM space:

0x12	*IPADDRESS*
0x13	*SUBNETMASK*
0x14	*TCPTIMEOUT*
0x15	*MODULENAME*
0x16	*RS232CONFIG*

RESTORE FACTORY PARAMETERS

Function Address	Function Name	Function	Read/Write Control
0x05	*RESTOREFACTORYPARAMETERS*	Factory default	Write only

Register Size	Register structure	Unit
0 Byte	none	none

Description:
Restores the following parameters from factory EEPROM space:

0x12	*IPADDRESS*
0x13	*SUBNETMASK*
0x14	*TCPTIMEOUT*
0x15	*MODULENAME*
0x16	*RS232CONFIG*

Note:
SAVEUSERPARAMETERS should be performed after this function in order to save restored parameters as user parameters.

SAVE FACTORY PARAMETERS

Function Address	Function Name	Function	Read/Write Control
0x06	*SAVEFACTORYPARAMETERS*	Save factory default	Write only

Register Size	Register structure	Unit
0 Byte	none	none

Description:
Saves the following parameters to factory EEPROM space:

- 0x12 *IPADDRESS*
- 0x13 *SUBNETMASK*
- 0x14 *TCPTIMEOUT*
- 0x15 *MODULENAME*
- 0x16 *RS232CONFIG*

Note:
This feature should only be used by a system integrator that would change the initial factory default settings.

VOLTAGE

Register Address	Register Name	Function	Read/Write Control
0x07	*VOLTAGE*	Power input voltage	Read only

Register Size	Register structure	Unit
4 Bytes	Signed (2's cplt) Int 16 (HH-HL) +16 bits fixed point (LH-LL)	Volt

Description:
Input Voltage

Limits:

Max	0x7FFFFFFxx = 32'767.996
Min	0x000000xx = 0.0
Step	0x000001xx = 0.004

Example:
When read 0x00234567 = 2311527 , Voltage = 35.27 (2311527/655636)

Information:
Over 32 V (0x00200000) the overvoltage warning bit is set and the overvoltage protection diode should increase power current.

Below 12 V (0x000C0000) the undervolatge warning bit is set.

Below effective 6.5 V (0x00068000), this value has no meaning.

Active:
Each time the processor is running.

WARNING

Register Address	Register Name	Function	Read/Write Control
0x08	*WARNING*	Bit to bit state	R/W

Register Size	Register structure	Unit
4 Byte	Unsigned Int 32 bits, each bit independent	none

Description:
Each information/warning/error is contained in 2 bits: the first one (from LSB to MSB) shows the actual state, the next one shows if this state appeared previously.
Only the bits that show the past states can be cleared by writing 0x00000000 to *WARNING* register.

Bits	when set
0-1	Over-temperature occurred on outputs stage.
2-3	Over-current occurred on outputs stage.
4-5	Under-voltage of the power supply.
6-7	Over-voltage of the power supply.

Other bits are reserved.

If any warning bit is set, the LED on the SOS button will become red.

Default value: bits 31 -> 0
0x00000000

Active:
Each time the processor is running.

NB POWER UP

Register Address	Register Name	Function	Read/Write Control
0x0B	*NBPOWERUP*	Number of boot	R

Register Size	Register structure	Unit
4 Byte	Unsigned Int 32 bits	none

Description:
This register counts device power ups. Each time that device initialization is done, this register is incremented by one.

Initial value:
0

Active:
Each time the processor is running.

TIME IN SERVICE

Register Address	Register Name	Function	Read/Write Control
0x0C	*TIMEINSERVICE*	Usage time	R

Register Size	Register structure	Unit
4 Byte	Unsigned Int 32 bits	none

Description:
This register counts the service time of the device in seconds.
Since it is saved every 2 hours to EEPROM, a bias compensation of 1 hour is added at each power-up.

Initial value:
0

Active:
Each time the processor is running.

COM OPTIONS

Register Address	Register Name	Function	Read/Write Control
0x10	*COMOPTIONS*	Communication options	Read/Write

Register Size	Register structure	Unit
4 Bytes	32 individual bits	none

Description:
This register is reserved for future use.

ETHERNET MAC

Register Address	Register Name	Function	Read/Write Control
0x11	*ETHERNETMAC*	Hardware network ID	Read only

Register Size	Register structure	Unit
6 Bytes	6 x Unsigned Bytes	none

Description:
A standard hardware unique identifier (worldwide) for each device on an Ethernet network.

Note:
If the user writes into this register, the MAC address will not be modified. This register is available only for informational purposes.

IP ADDRESS

Register Address	Register Name	Function	Read/Write Control
0x12	*IPADDRESS*	IP network ID	Read/Write

Register Size	Register structure	Unit
4 Bytes	4 x Unsigned Bytes	none

Description:
Network identifier used for TCP/IP and UDP/IP.
The values 255 (0xFF) and 0 (0x00) are reserved for broadcast and network addresses and should not be used in this register.

Notes:
The module will change for a new IP address only when all of its communications ports are closed.
Do not forget to use a *SAVEUSERPARAMETERS* command.

Default value:
169.254.5.5

Example:
For the IP=192.168.16.14 (0xC0, 0xA8, 0x10, 0x0E), write 0xC0A8100E to IPADDRESS.

SUBNET MASK

Register Address	Register Name	Function	Read/Write Control
0x13	*SUBNETMASK*	IP subnet mask	Read/Write

Register Size	Register structure	Unit
4 Bytes	4 x Unsigned Bytes	none

Description:
Network IP subnet mask used for TCP/IP and UDP/IP.

Notes:
The module will change for a new subnet mask only when all of its communications ports are closed.
Do not forget to use a *SAVEUSERPARAMETERS* command.

If you do not want to use subnets, use the following subnet mask when IP address byte 0 is:

>0 and <=127	:	255.0.0.0	(Class A addresses)
>127 and <=191	:	255.255.0.0	(Class B addresses)
>191 and <=223	:	255.255.255.0	(Class C addresses)

Default value:
255.255.0.0

Example:
For the IP=10.2.6.45 and subnet mask = 255.255.0.0:
IP address class = A → netID = 10, subNetID = 2 and hostID = 6.45

TCP TIMEOUT

Register Address	Register Name	Function	Read/Write Control
0x14	*TCPTIMEOUT*	Timeout for TCP connection	Read/Write

Register Size	Register structure	Unit
1 Byte	Unsigned Int 8 bits	sec

Description:
The TCP timeout is a value (in seconds) after which the user will be disconnected if the device has not been accessed in the meantime.

If the value is 0, the TCP timeout is deactivated. In this case however, if the client crashes during connection, the communication will never be closed on the module's side! Because a maximum of 4 communications are allowed at the same time on the module, one of them will be blocked. If the client crashes four times, all of the 4 communications will be blocked and the module will have to be reset!

The timeout for each TCP/IP connection is reloaded when there is traffic through the port.

Default value:
30

Limitations:
Max value: 255

MODULE NAME

Register Address	Register Name	Function	Read/Write Control
0x15	*MODULENAME*	Module's ASCII name	Read/Write

Register Size	Register structure	Unit
16 Bytes	16 (only) x Unsigned Bytes (CHAR)	none

Description:
Name and/or description of the module.

Example:
For the name "Hello Module"; extend to 16 byte the name: "Hello Module"+5x space=16 Byte.
So write 0x48656C6C 6F204D6F 64756C65 20202020.

RS232 CONFIG

Register Address	Register Name	Function	Read/Write Control
0x16	*RS232CONFIG*	RS232 baud rate and flow control configuration	Read/Write

Register Size	Register structure	Unit
1 Byte	Unsigned Int 8 bits	none

Description:
RS232 baud rate and flow control configuration.

Bits 0-2	Baud rate configuration : 0 : 4800 bds 1 : 9600 bps (default) 2 : 19200 bps 3 : 38400 bps 4 : 57600 bps 5 : 115200 bps 6 : 115200 bps 7 : 115200 bps
Bits 3-6	Reserved
Bit 7	Hardware flow control bit (0 = disabled, 1 = enabled)

Default value:
1 (9600 bps without flow control)

I2C SPeeD

Register Address	Register Name	Function	Read/Write Control
0x18	*I2CSPD*	I2C speed setting	Read/Write

Register Size	Register structure	Unit
1 Byte	Unsigned Int 8 bits	none

Description:
I2C speed setting between ~39kHz and 400kHz.

The value of this parameter must be computed with the following formula based on the wanted speed:

$$I2CSPD = \frac{10^7}{I2CSpeed_{wanted}} - 1$$

Note:
Speeds greater than 100kHz have some limitations (see page 20).
Do not use a speed greater than 400kHz (<24). Greater speed will not work.
This register cannot be saved to EEPROM and must be configured at each power-up.

Default value:
99 (100kHz)

Examples:
Most common speeds:
100kHz : I2CSPD = 99
400kHz : I2CSPD = 24

TCP CONNECTIONS OPENED

Register Address	Register Name	Function	Read/Write Control
0x1A	*TCPCONNECTIONSOPENED*	Number of TCP connections that are opened	Read only

Register Size	Register structure	Unit
1 Byte	Unsigned Int 8 bits	none

Description:
Number of users connected to the device using TCP.
Value can be 0 to 4.

ANALOG INPUTS THRESHOLD

Register Address	Register Name	Function	Read/Write Control
0x20	*ANALOGINPUTSTHRESHOLD*	AD converter threshold on inputs	Read/Write

Register Size	Register structure	Unit
4 Bytes	Signed (2's cplt) Int 16 (HH-HL) +16 bits fixed point (LH-LL)	Volt

Description:
Defines the threshold used by the AD converter on inputs pins to define input state as low or high (0 or 1).

Default:
6.0 V

Example:
If your inputs use 5V TTL signals, set this value between 1 and 4 V.

OUTPUTS

Register Address	Register Name	Function	Read/Write Control
0x21	*OUTPUTS*	Sets outputs states	Read/Write

Register Size	Register structure	Unit
2 Bytes	Unsigned Int 16 bits, each bit independent	none

Description:
Controls the state of each of the output pins.

Bit 0	OUTPUT#1 (Relay A)
Bit 1	OUTPUT#2 (Relay B)
Bit 2	OUTPUT#3
Bit 3	OUTPUT#4
Bit 4	OUTPUT#5
Bit 5	OUTPUT#6
Bit 6	OUTPUT#7
Bit 7	OUTPUT#8
Bit 8	OUTPUT#9
Bit 9	OUTPUT#10
Bit 10	OUTPUT#11
Bit 11	OUTPUT#12
Bit 12	OUTPUT#13
Bit 13	OUTPUT#14
Bit 14	OUTPUT#15
Bit 15	OUTPUT#16

Note:
This register cannot be saved to EEPROM and must be configured at each power-up.

Example:
b'0000 0000 0000 0111' → Outputs number 1, 2 and 3 are high and others are low.

INPUTS

Register Address	Register Name	Function	Read/Write Control
0x23	*INPUTS*	16 inputs states	Read only

Register Size	Register structure	Unit
2 Bytes	Unsigned Int 16 bits, each bit independent	none

Description:
Show digital state of each inputs pin. The state is defined by comparing actual input voltage with the ANALOG INPUTS THRESHOLD.

Bit 0	INPUT #1
Bit 1	INPUT #2
Bit 2	INPUT #3
Bit 3	INPUT #4
Bit 4	INPUT #5
Bit 5	INPUT #6
Bit 6	INPUT #7
Bit 7	INPUT #8
Bit 8	INPUT #9
Bit 9	INPUT #10
Bit 10	INPUT #11
Bit 11	INPUT #12
Bit 12	INPUT #13
Bit 13	INPUT #14
Bit 14	INPUT #15
Bit 15	INPUT #16

Example:
b'0000 0000 0000 0111' → Inputs number 1, 2 and 3 are high and others are low.

INPUT VOLTAGE x VALUE

Registers Addresses	Register Name	Function	Read/Write Control
0x30 to 0x3F	*INPUTVOLTAGExVALUE*	Last INPUTVx A/D conversion result	Read only

Register Size	Register structure	Unit
4 Bytes	Signed (2's cplt) Int 16 (HH-HL) +16 bits fixed point (LH-LL)	Volt

Description:
Voltage value of the 16 inputs pins between -12V and 12V. This value saturates at ~ -15.5V and 24V.

Notes:
The A/D converter has a 10 bits resolution (~0.043V).

The acquisition is done every 850µs and is asynchronous with the read access of the A/D VALUE registers. When you access this register, you get the last A/D conversion result, which can be up to 850µs old.

Contact address:

FiveCo - Innovative Engineering
En Budron H11
CH-1052 Le Mont-sur-Lausanne
Switzerland
Tel: +41 21 632 60 10
Fax: +41 21 632 60 11

www.fiveco.ch
info@fiveco.ch

f i v e co
innovative engineering