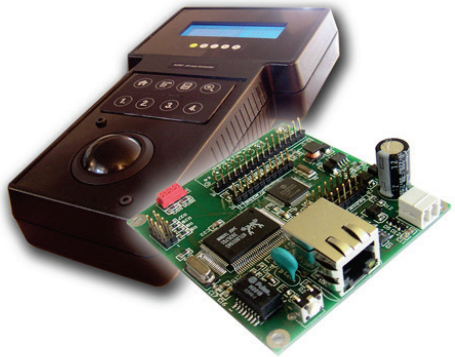
See page 9 for quick start

3 Axes Ethernet Control Board

FMod-IPAXESCTRL

User's Manual

Version 1.5



five co
innovative engineering

Version: **1.5**
Last revision: **October 16, 2019**
Printed in **Switzerland**

Trademarks
Windows® is a registered trademark of Microsoft Corporation.
Ethernet® is a registered trademark of Xerox Corporation.
Java® is a registered trademark of Sun Microsystem.
Philips® is a registered trademark of Koninklijke Philips Electronics N.V.
Borland® is a registered trademark of Borland Software Corporation.

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: https://www.fiveco.ch/product-fmod-ipaxesctrl.html
e-mail: support@fiveco.ch

Revision history

Revision	Date	Author	Note	Firmware version	Applet version	Application version
1.0	16.11.04	XG	- First revision history	Since 1.8	1.3	-
1.1	21.04.05	XG	- Add IP SUBNETMASK register (0x13) - Add TCPCONNECTIONSOPENED register (0x1A) - Easy change IP with broadcast - Add relative coordinate with relative zero - Add AXISxPOSITIONOFFSET registers (relative offset)	Since 1.11	1.6	-
1.2	30.09.05	AG	- Text and layout correction - Add checksum function example	Same	Same	-
1.3	27.01.06	AG	- Correct function ID for easy change IP answer frame.	Same	Same	-
1.4	21.12.07	AG	- New address	Same	Same	-
1.5	16.10.19	AG	- Remove Java Applet and add new application.	Same	Same	1.0

Table of Contents

1. Package and operating conditions

Package contents

- 1 Axes Ethernet Control board : FMod-IPAXESCTRL
- This manual
- (optional) Power Supply :
  Power over Ethernet IEEE802.3af Hub-injector
  (110-240 VAC, 50/60Hz -> +48v, 350 mA)

Operating conditions

- Operating temperature -20 – 70 °C
- Supply voltage Vcc 7-48 VDC
- Input capacity (Power +,-) ~220uF, ~50mOhm ESR

- Power consumption (~0.8W) 65mA when idle @ Vcc 7 V
Without LCD & Trackball 25mA when idle @ Vcc 24 V
15mA when idle @ Vcc 48 V

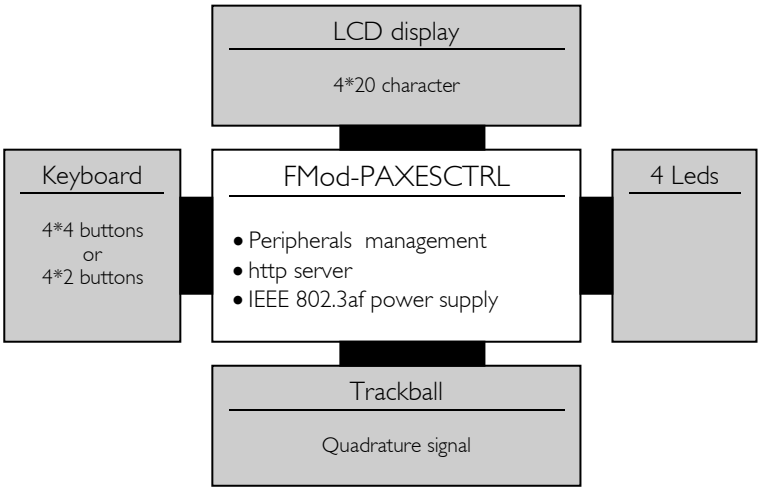
- Power consumption (~1.2W) 150mA when idle @ Vcc 7 V
With LCD & Trackball45mA when idle @ Vcc 24 V
25mA when idle @ Vcc 48 V

2. Overview

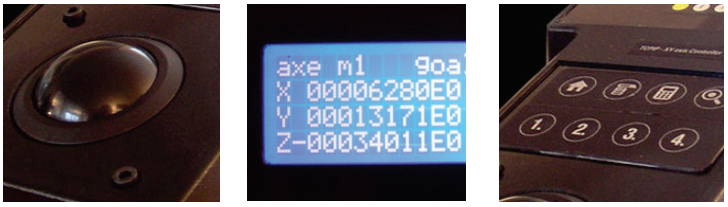
The FMod-AXESCTRL is a board used to control a maximum of 3 DC-motorized axes (card FMod-IPDCMOT or FMod-IPECMOT). The main advantage of this card is its communication protocol (Ethernet: TCP/IP-HTTP) which allows it to be connected to a standard Ethernet network. In addition, the board follows the IEEE 802.3af standard which allows it to be powered through the Ethernet cable (PoE-Powered Device).

The FMod-IPAXESCTRL board works without the use of a computer (PC or Mac). It communicates independently with the selected FMod-IPxxMOT devices (motor control cards) through the Ethernet network. It can also be easily configured with any standard web-browser, using its internal web-pages (http server onboard).

A (4x20char) LCD can be connected in order to locally display information, and a trackball + keyboard are the available inputs to handle the axes.



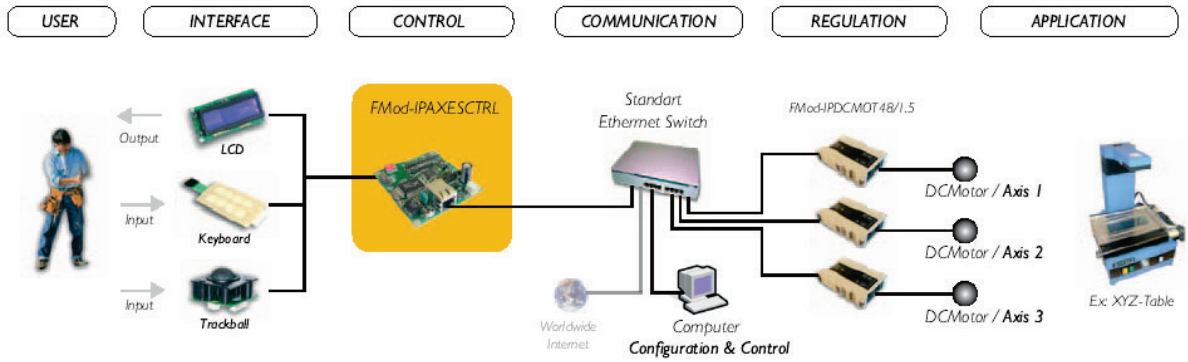
This diagram shows all peripherals that can be connected to the board.



The trackball, the LCD screen and the Keyboard (4x20)

Connection architecture (Ethernet)

The figure below shows the connection diagram of the FMod-IPAXESCTRL board. The user interacts with the MMI (Man Machine Interface), e.g. the LCD, the Keypad and the Trackball, which are connected to the Axes control card. This card is connected together with the Motor control card (FMod-IPDCMOT48/1.5) to a standard/industrial Ethernet Switch (available on the market from 30.- USD) and is used in applications such as XYZ-Tables.



- The axes control board and the motor control boards are **connected together to an "Ethernet 10BaseT compliant Switch"**. The communication between them is done through the UDP-IP protocol.

- An optional computer (PC, Mac, ...) can be connected to the same network (to the Ethernet 10BaseT compliant Switch) in order to **configure the parameters** of the axes control card

- The same Switch used to connect the FMod-IPAXESCTRL and the FMod-IPDCMOT48/1.5 can be connected to the Internet network (under an ADSL modem/router or a professional Router) and therefore make the system **accessible from anywhere (remote)**.

Note : *It is also possible to connect the FMod-IPAXESCTRL directly to an FMod-IPxxMOT motor control card through a simple Ethernet Cross Cable (not using the switch). In this case, the system will only be able to control one (1) motor/axis.*

Board Layout

JP3 SOS button

J1 IN Power +
IN Power -

J7 Ethernet RJ45

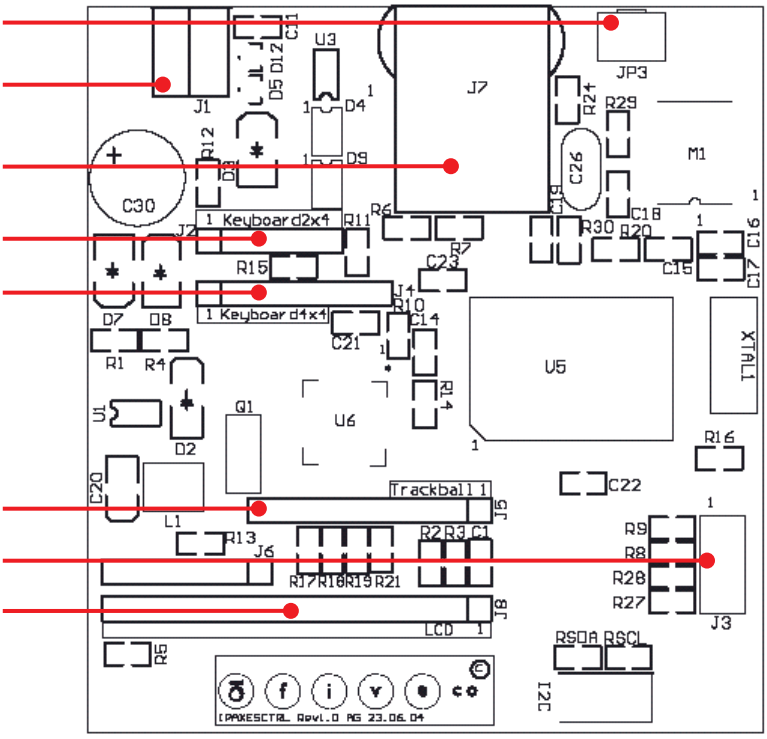
J4 Keyboard 2x4

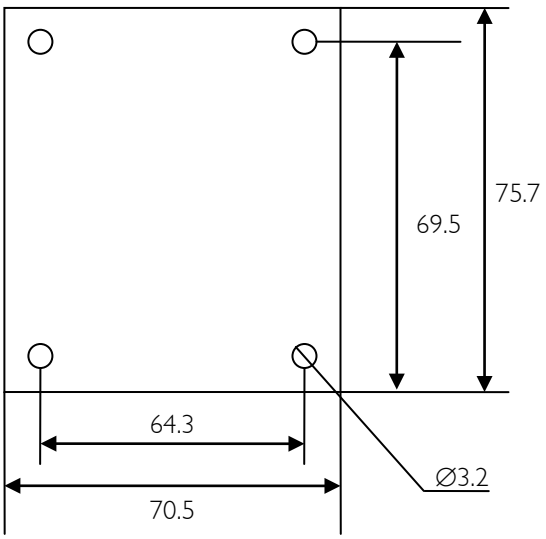
Keyboard 4x4

J5 Trackball

J3 LEDs indicator

J8 LCD display



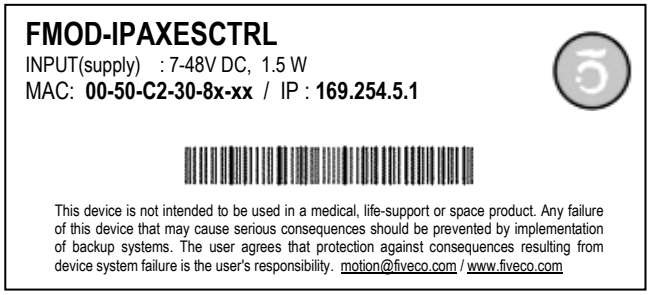


Max height: 20mm

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 used for the initial configuration and the card. Detailed information about hardware and software can be found further in this document.

You can find the board factory communication settings on the label that is found on the board and on the box.

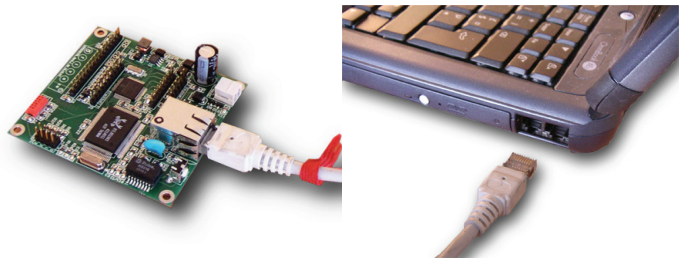


The MAC Address is the 48bits unique identifier on Ethernet networks. The IP Address can be modified. The complete procedure is described further in this manual.

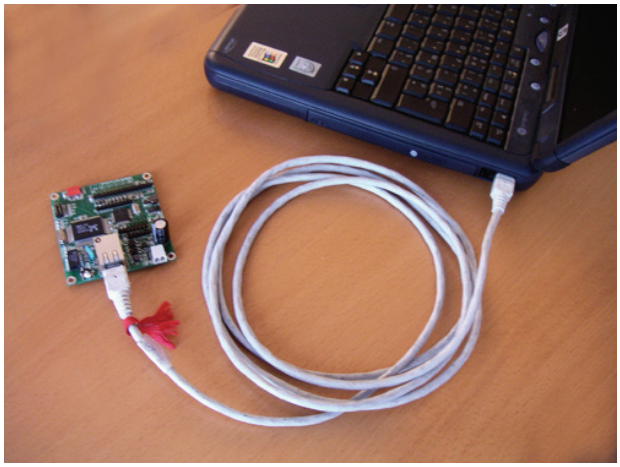
Note: *If the module has already been configured and the IP address has been changed to an unknown value, you can retrieve an SOS IP address (on label) by pressing the "SOS button" when the card is operating normally.*

Power and network

1. Connect the card to a computer using a RJ45 **cross wired** cable (direct-link), or with a straight cable to an Ethernet-switch.

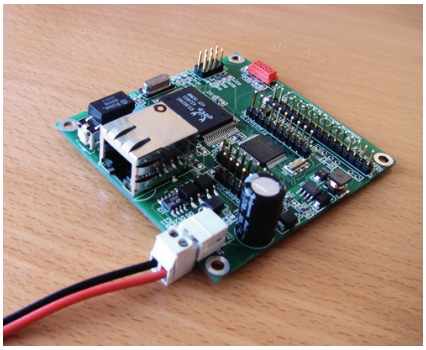


Connect the cable to the card and to the PC



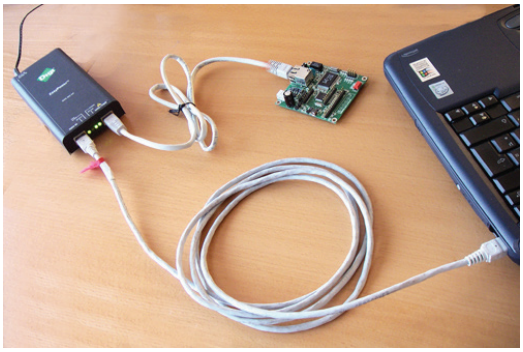
Connection with an Ethernet crosscable

2. Connect a power supply (7-48V) or a POE injector to the module.



Discrete power supply

OR



PoE with a Power injector

Changing IP address

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

1. Plug your new card 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 card.
6. If you have more than one network adapter on your PC, the software will ask you to select the one that is connected to the same network as the FMod-IPAXESCTRL.
7. The software will suggest a new IP address without the last byte. Choose a new IP (**that is not already used on your network!!**) and click the "Change IP address" button.

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

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

Remark:
The IP address will not be changed if a TCP connection exists with the card.

4. Hardware

There are different configurations, depending on the needs of your application:

- FMod-IPAXESCTRL + Keyboard
- FMod-IPAXESCTRL + Keyboard + trackball
- FMod-IPAXESCTRL + Keyboard + LCD
- FMod-IPAXESCTRL + Keyboard + LCD + trackball

An independent connector for each peripheral is present on the card. No particular configuration is needed to enable/disable each of them. You simply have to connect a peripheral before powering-up the card.

Note: It is better NOT to unplug the peripherals while the system is powered.

Power supply

The card can be powered in two modes:

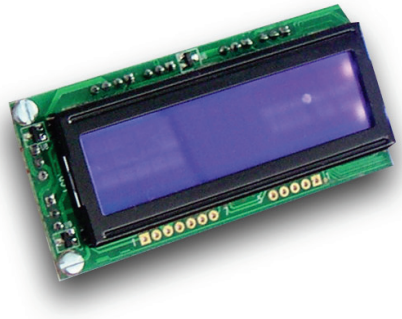
WARNING: do NOT use both modes at the same time!

1. **POE** (Power Over Ethernet) 48V only, 350 mA on connector J7 (RJ45), pins 4&5(+), 7&8(-). Swapped polarity works too. Compatible with IEEE 802.3af mode B & A.

2. Discrete 7-48 V DC on connector J1. Negative pole is near the J7(RJ45) connector.

LCD Display

The FMod-IPAXESCTRL board drives an LCD display with 20 columns and 4 lines. LED backlight is also powered from the FMod-IPAXESCTRL, and is always ON.



Different LCD colors are available, but FiveCo recommends Crystalfontz' LCDs (www.crystalfontz.com)

The LCD used by FiveCo is Crystalfontz' model *CFAH2004A-TMI-JP (with* LSI HD44780 driver inside), blue-backlit with white characters (pictured on the left).

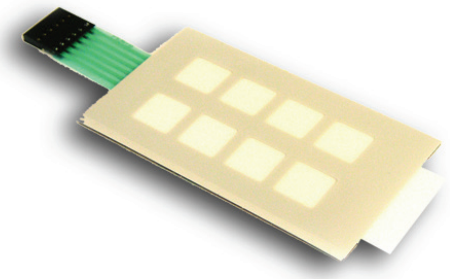
The connector on the FMod-IPAXESCTRL board has the same pin configuration as the one of the LCD (allowing an easy flat-cable connection), but the LCD has to be used in 4bits data mode only.

Board **J8** connector pinout:

1	GND
2	5V
3	VO (Contrast)
4	RS
5	R/W
6	E
7	- (nc)
8	- (nc)
9	- (nc)
10	- (nc)
11	DB4
12	DB5
13	DB6
14	DB7
15	Backlight 5V with a 33Ohm resistor already in series.
16	Backlight GND

Keyboard

The board offers the possibility to connect two kind of MATRIX KEYBOARDS: 2x4 or 4x4.



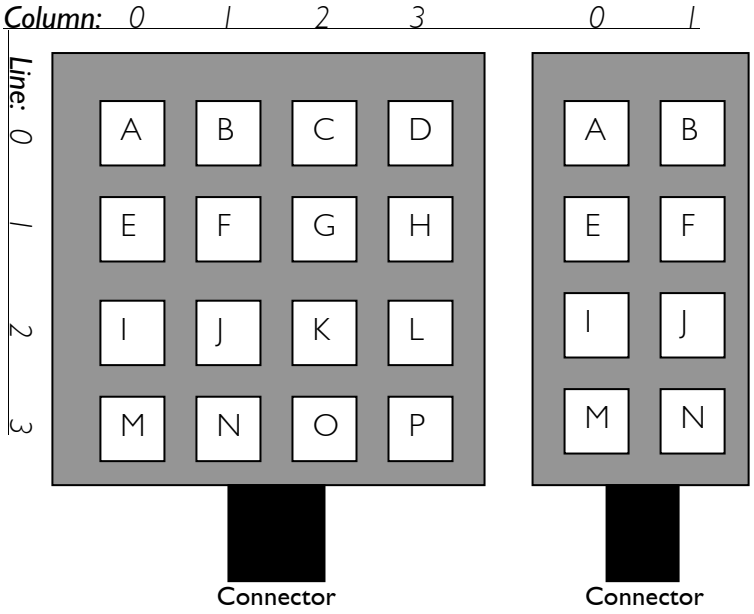
Board **J4(up)** connector pinout **(2x4 version)**:

1	Column 0
2	Column 1
3	Line 3
4	Line 2
5	Line 1
6	Line 0

Board **J4(down)** connector pinout **(4x4 version)**:

1	Column 0
2	Column 1
3	Column 2
4	Column 3
5	Line **2**
6	Line **3**
7	Line **1**
8	Line **0**

The following diagram shows the position of each key on the keyboard (useful for the configuration).



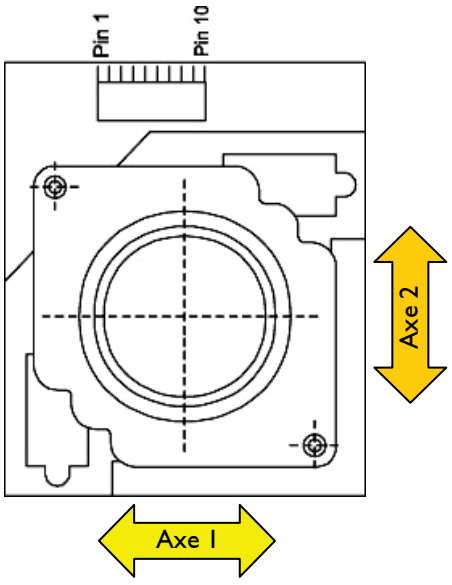
Trackball

A two axes trackball enables the user to manage axis 1 and axis 2 (axis 3 is manageable only from keyboard). Each channel must have quadrature signal (A, B).

PEWATRON model LP150-5FV00 is the reference for this board. It is also possible to use other kinds of trackballs with different resolutions or even gravity independency (ex: Megatron model 816TC).

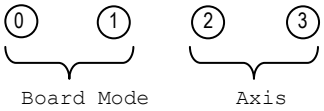
Board **J5** connector pinout and trackball connector pinout and axes:

1	- (not connected)
2	- (nc)
3	- (nc)
4	5V
5	Xa
6	Yb
7	Ya
8	Xb
9	- (nc)
10	GND



LEDs

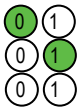
The FMod-IPAXESCTRL is provided with a connector for 4 LED **(J3).** They give information about the mode (1, 2, 3) of the board and the state (activity or not) of the axes. Here is the LED's layout:



Board Mode

The different modes define the way the input (trackball and Keyboard) and output devices (LCD display) interact with the physical system (commands sent to the motor). See "Axes parameters Panel" in "Java applet" chapter for more detailed information.

Mode 1 = LED0 on, LED1 off
Mode 2 = LED0 off, LED1 on
Mode 3 = LED0 off, LED1 off

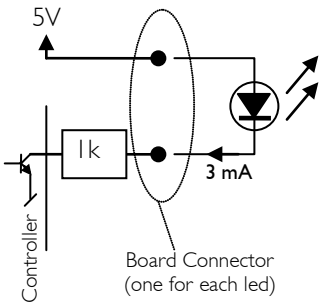


Axis State

The LEDs are especially useful in order to determine the mode on how the axes are controlled by the trackball (axes 1 and 2). When an axis is found and is ready to receive data, then its corresponding LED is on; if an axis is "suspended", its LED is off. It is possible to drive an axis (for ex: axis1) without modifying the other one (ex: axis2) which has been put in a "suspended" state with the press of a keyboard button.

Notes:

- No LED exists for axis 3 since it cannot be driven by the trackball.
- The LEDs display the information in any case, even if the LCD is not connected.

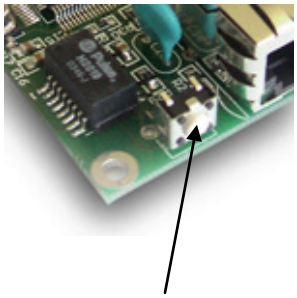


The board drives the lower pin (Cathode /-) of the LED while the upper one (Anode /+) is connected to 5V. An onboard 1kOhm resistor is connected in series to limit the current to ~3mA.

Board **J3** Connector:

1	Led 0	resolution mode 1
2	5V	
3	Led 1	resolution mode 2
4	5V	
5	Led 2	axis 1 ON/OFF
6	5V	
7	Led 3	axis 2 ON/OFF
8	5V	

SOS button


SOS button

This button has 2 different purposes, depending on the moment when it is pressed: at system power-up or when the system is already running.

POWER UP ACTION (the SOS Button is pressed during power up)

This action is useful when the "out-of-factory" state of the card needs to be exactly restored.
The *RestoreFactoryParamters* (0x05) function is automatically called, and the actual IP address is overwritten with the SOS IP address written on the sticker of the card (169.254.5.1). After that, all these parameters are automatically saved in the User's parameters memory with the *SaveUserParameters* (0x03) function.

RUNNING ACTION (the SosButton is pressed while running)

This action is useful when the user has lost the actual IP address of the card, and temporarily needs to connect to it without changing all of the card's parameters.
Only the IP address is overwritten, using the SOS IP address that appears on the card's sticker (169.254.5.1). This new IP will be valid only after all TCP/IP connections (web pages included) have been closed.
In this case the SOS IP address is not automatically saved, so if you need it, you will have to manually call the *SaveUserParamters* (0x03) function. Otherwise, the previous IP will be reloaded upon the next power-up.

Warning:
Do not use the "*SOS IP address*" as the normal IP for your module (during normal board operation). If you plug another module on the same network, both will have the same IP and will therefore be impossible to both configure.

5. TCP/UDP Server

General

The board provides an Ethernet port (RJ45 – connector J7) which allows access to all parameters (registers) of the module through a TCP or UDP connection.

Here you will find a small comparison table between these two protocols (non exhaustive):

Features	UDP	TCP
CheckSum (Data integrity)	YES	YES
Multiport (data multiplexing)	YES	YES
Flow control	NO	YES
Acknowledge Data	NO	YES
Multipacket order reconstruction	NO	YES

Two ports are available when using the **TCP protocol**:

- Port #80 for HTTP communication.
- Port #8010 Access to the control port

Only one port is accessible through the **UDP protocol**:

- Port #7010 Access to the control port

You will find a detailed description of the different ports in the next pages.

Note: *The board allows up to 4 simultaneous TCP connections. That means, for example, that 4 users can connect to the #80 port to see the web page, or 4 users can be connected to the #8010 port and control the I/Os, or even that two see the page and two control the I/Os, etc. In UDP protocol, there is no limitation on the number of users connected.*

HTTP port (TCP # 80)

This port is used to access the web page stored on the module.

The user can simply access that port and ask for a particular page, through the use of a standard Web browser, and typing the IP address of the card:

Ex: type http://169.254.5.1/index.htm in the address bar of the browser and the "index.htm" page will be loaded.

Control port (TCP # 8010 or UDP # 7010)

This port is used to access the registers described in the chapter "Registers management" of this manual.

Note: *If you plan to configure all registers only from the onboard Webpage, you can skip this section and go to the Java-Applet section. (The Java-Applet also uses this port to read and write the different settings!)*

TCP/IP works in big endian: most significant byte first, followed by least significant byte(s). The access is done by sending a packet that follows a simple (6 byte header) protocol.

Structure of each packet:

1) Function ID (2 bytes)	*Code of the function that has to be executed.*
2) Transaction ID (2 bytes)	*Number that defines this packet*
3) Length of the parameters (2 bytes)	*Number of the parameters + data bytes*
4) Parameters (X Byte)	*Parameters + Data*
5) Checksum (2 bytes)	*Control Sum of packet's bytes*

Function ID

The specific code for each function can be found on the next page of this manual.

Transaction ID

The user defines himself the values of the *Transaction ID*. Normally, each packet/transaction (communication request) should have a different ID (even though this is not mandatory). When the FMod-IPAXESCTRL receives a command/packet, it sends back an answer (at each request). This answer contains the same *Transaction ID* than the corresponding command previously sent. In that way, the user is able to check the execution of each command.

Length of the parameters

This 2byte value corresponds to the length (in bytes) of the next section of the packet (parameters only).

Parameters

This part of the packet contains all the parameters (mainly the data that are sent).

Checksum

This 2 bytes value is the Check Sum of all the bytes of the packet (more information on next pages).

READ register(s) value 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...

The maximum number of registers that can be read at one time is almost 30. The answer sequence should not be greater than 180 bytes. If the number of registers is too high, the FMod-IPAXESCTRL will answer only with the value of some of them.

The module **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 each register that has been asked for reading			
...	Checksum	16 bits	0x...

WRITE register(s) value 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 each register that has been asked for reading			
...	Checksum	16 bits	0x...

The maximum length of this sequence is 180 bytes.

The module **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...

Easy IP address config (UDP # 7010)

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

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

Procedure:
Send a UDP broadcast message (using a local or direct broadcast IP address) to your network (to which the FMod- IPAXESCTRL 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- IPAXESCTRL recognizes its MAC address, it will answer this command with a simple acknowledges. It will then change its IP address and subnet mask IF NO TCP CONNECTION IS MADE TO THE BOARD.

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;
}
```

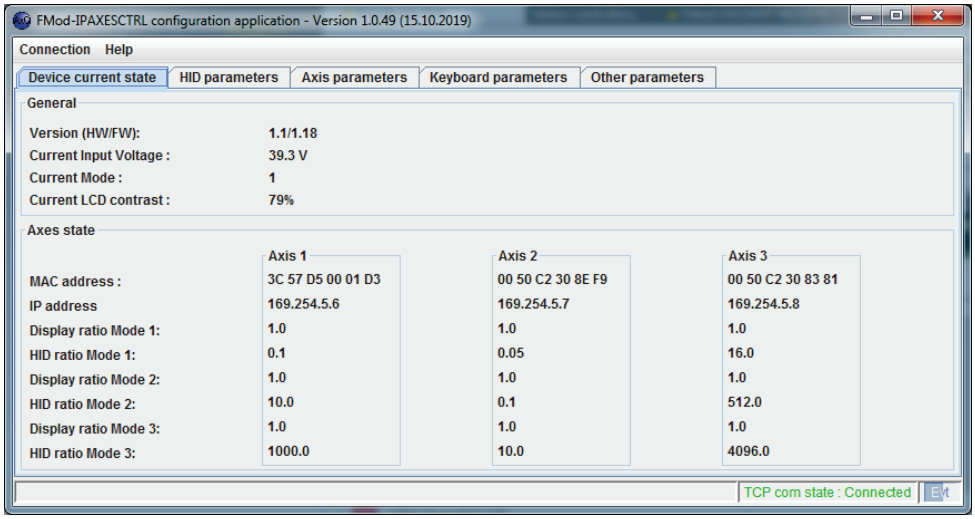
This function needs a Byte array (ByteTab) containing the command sequence and the array's length (Size) as input. It returns the checksum as an int.

6. Configuration Application

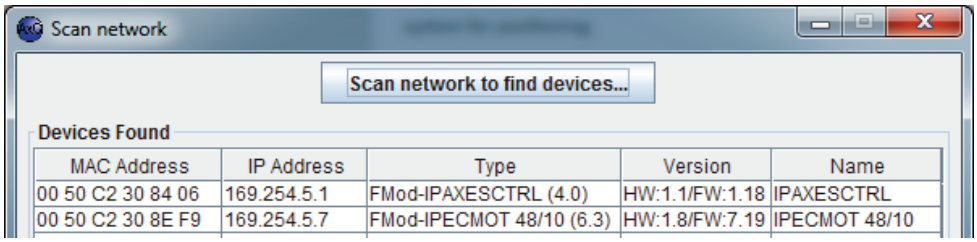
A specific Java application is provided with the module to control all of its parameters without having to write any specific software.
A Java Runtime Environment (1.8+) needs to be present on the computer to be able to run the application. If the correct JRE is not found, a web page will be opened on the application launch to allow user to download it.

Overview

Run the FMod-IPAXESCTRLApp.exe provided on our web server at the following address : https://www.fiveco.ch/product-fmod-ipaxesctrl.html.



Use the "Connection->Scan network" menu to scan the network to find the FMod-IPAXESCTRL you want to configure. If you have to change the IP address of the device, you can use the easy tool under "Connection->Change IP Adress easily...".

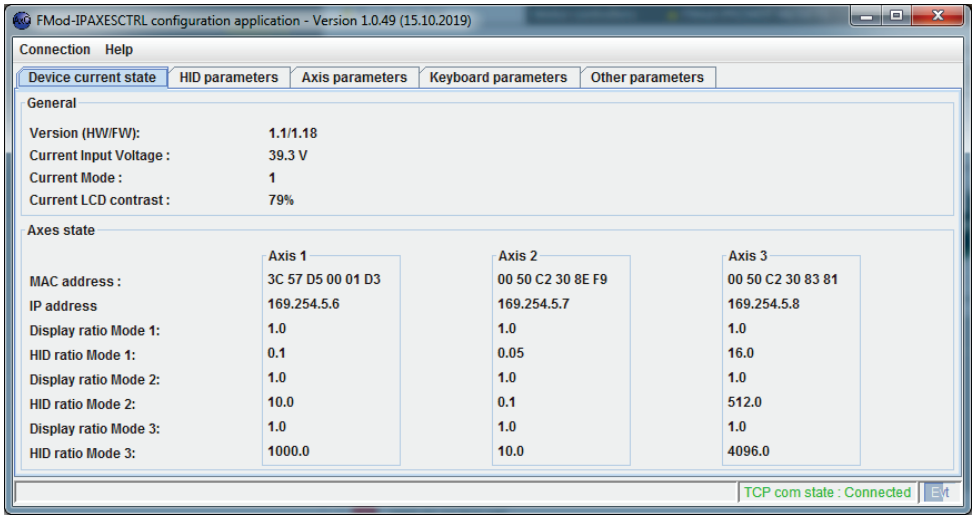


If several device are found by the scan device feature, you can check the MAC address of the one you want to configure to select the correct one in the table.
A click on one device will connect the application with it. Please note that the device type will be checked and you will not be able to an incorrect device type.

The following paragraphs describe this application.

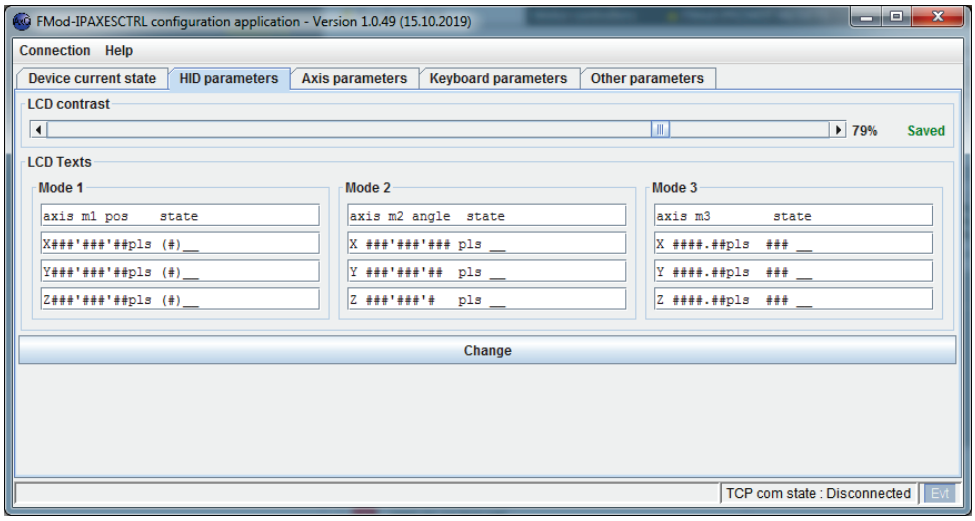
Device Current State Panel

The Device Current State panel shows the information specifically related to the module. The values of the parameters are displayed.



The others panels allow to configure the device. Please note that you have to click on the "Change" button of the panel in which you made modifications to send them to the device (automatically saved in user EEPROM).

HID Parameters Panel



The LCD contains 4 lines of 20 characters each, and the user can also choose the contrast of the screen. For each board's mode, (1,2,3) the LCD display can be different.

In the "LCD texts - Mode 1/2/3", the user can define the text and values displayed on the 4 lines separately. Here is an example:

Configuration String	Output on display
"axe m1 pos state "	axe m1 pos state
"X ###'### pls ### --"	X 123'456 pls 789 Ok

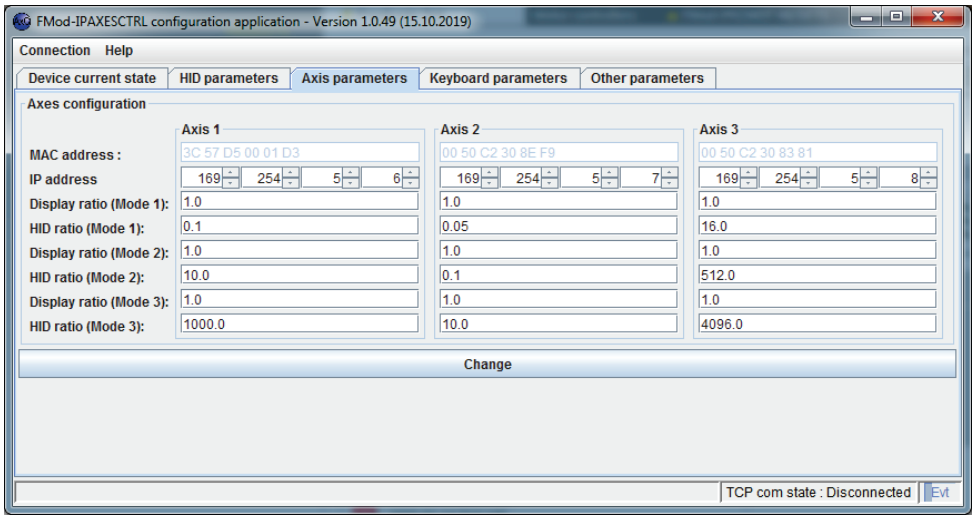
See description of "LCD MODE(i) TEXT" register (p.56) for more details

The measured value contains 8 digits (+1 more for the sign). On the display, the character '#' represents one digit of that value. If less than 9 '#' are present on a line, the least significant digits should be skipped. Other (ASCII) characters can be used with the # values.

Note: *The first line is reserved for text (title), the second one shows the values of axis1, the third one the values of axis2 and the last one shows the values of axis 3.*

Axes Parameters Panel

This panel allows management of the parameters of the 3 motor control boards [FMod-IPxxMOT] (IP address of each Axis) that can be controlled through the FMod-IPAXESCTRL card. For each axis, the user can define 3 DIFFERENT MODES that can be configured with different settings (HIDRatio/DiplayRatio).

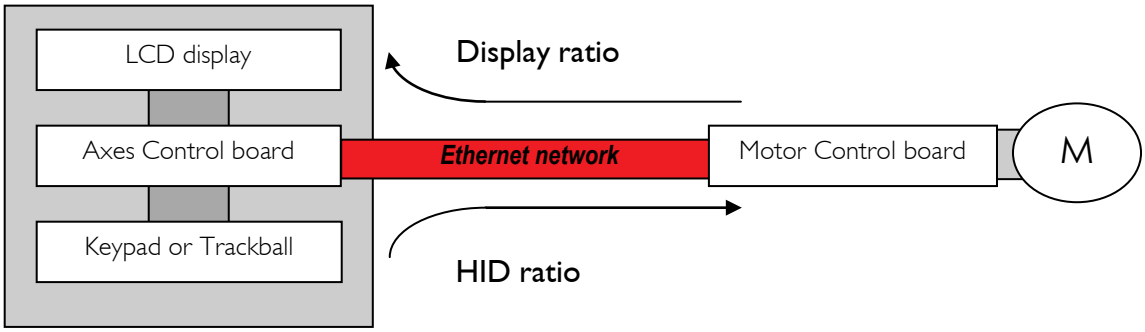


- The ratio between the motor pulses and the value displayed with the ***Display ratio.***

Example: value of 10 means: 1 motor pulse = 10 increment on LCD

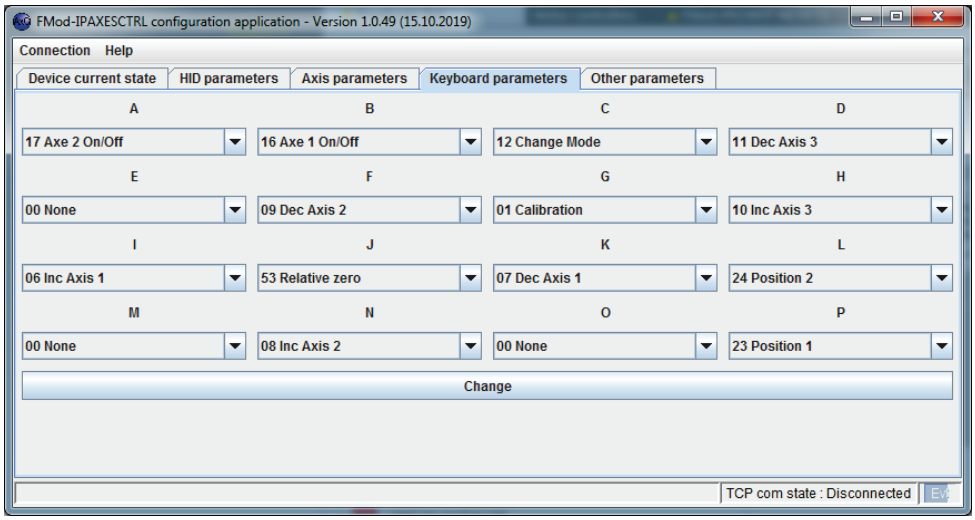
- The ratio between human interface devices (keyboard or trackball) and motor pulses with ***HID ratio.***

Example: value of 10 means: 1 trackball pulses = 10 increment on motor



Keyboard Parameters panel

This panel shows the keyboard's parameters.



Each key of the keypad is represented by a character: from A (line0, column0) to P (line3, column3) and the corresponding list of available functions. Just press the corresponding scroll menu and choose the function you wish to program for this specific button.

N°	Function
0	None
1	Set all axes to Position mode/Homing
2	Stop all axes (break mode)
3	Free all axes
4	Free selected axes
5	Set all axes to Position mode
6	Increment axis 1
7	Decrement axis 1
8	Increment axis 2
9	Decrement axis 2
10	Increment axis 3

N°	Function
27	Go / Save position 5
28	Go / Save position 6
29	Go / Save position 7
30	Go / Save position 8
31	Go / Save position 9
32	Go / Save position 10
33	Increment axis 1 in Mode 1
34	Decrement axis 1 in Mode 1
35	Increment axis 2 in Mode 1
36	Decrement axis 2 in Mode 1
37	Increment axis 3 in Mode 1

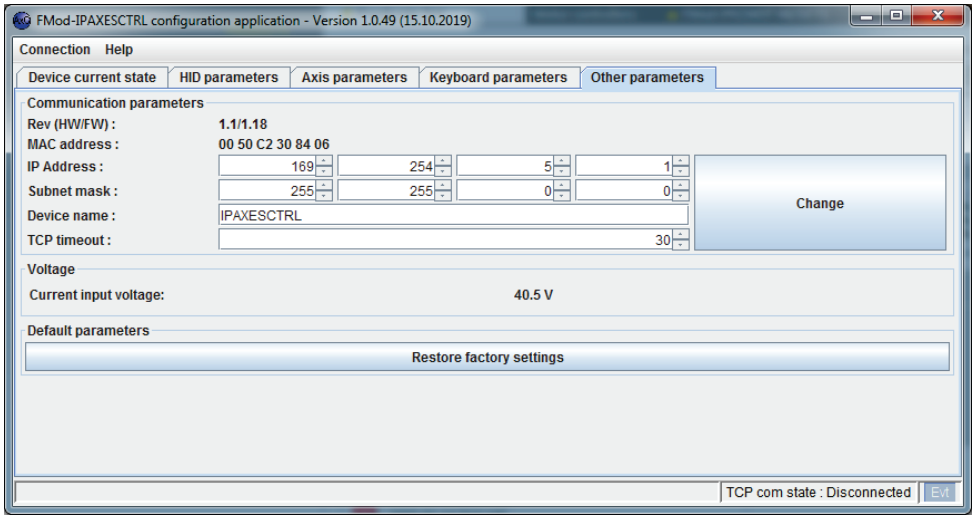
11	Decrement axis 3
12	Change mode (1→2→3→1...)
13	Mode 1
14	Mode 2
15	Mode 3
16	Axis 1 ON-OFF
17	Axis 2 ON-OFF
18	Axis 3 ON-OFF
19	Axis 1 suspended while pressing
20	Axis 2 suspended while pressing
21	Axis 3 suspended while pressing
22	Go / Set position zero absolute
23	Go / Save position 1
24	Go / Save position 2
25	Go / Save position 3
26	Go / Save position 4

38	Decrement axis 3 in Mode 1
39	Increment axis 1 in Mode 2
40	Decrement axis 1 in Mode 2
41	Increment axis 2 in Mode 2
42	Decrement axis 2 in Mode 2
43	Increment axis 3 in Mode 2
44	Decrement axis 3 in Mode 2
45	Increment axis 1 in Mode 3
46	Decrement axis 1 in Mode 3
47	Increment axis 2 in Mode 3
48	Decrement axis 2 in Mode 3
49	Increment axis 3 in Mode 3
50	Decrement axis 3 in Mode 3
51	Mode 2 → Mode 1 (released)
52	Mode 3 → Mode 1 (released)
53	Go / Set position zero relative

Warning: Some aforementioned function numbers have two behaviors (separated with a /) depending on the moment when the button is pressed. The first function will take place when the button is pressed for less than 3 seconds, and the second function takes place in the other case.

Example:
Function #23: If the press is short, "Go" is launched. If the button is pressed for more than 3 seconds, position 1 is saved.

Other Parameters Panel



This panel can be used to change the FMod-IPAXESCTRL board's communication parameters (user should better use "Connection->Change IP Adress easily..." menu.):

- **IP address:** Press the "Change" button. The new IP address will be valid only once all current connected users have closed their connection.
- **Subnet mask:** useful only for special UDP directed broadcasts.

- **Module name** of your choice.
- **TCP Timeout**: Maximum time before automatic disconnection when the client does not send any packets.

This panel allows also seeing the current input voltage of the power supply.

Finally, if you want to reset all devices parameters to the factory state, click on the corresponding buttons.

7. Reference zero absolute/relative

FMod-IPAXESCTRL can work with the exact position coming from each FMod-IPxxMOT, in which case we speak about an absolute zero, because the "0" displayed on the FMod-IPAXESCTRL means that the corresponding axis *POSITION* register is really at position 0.

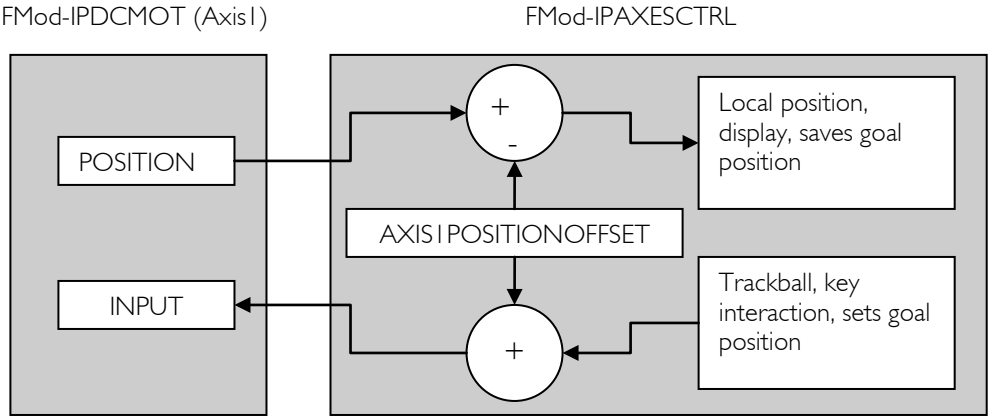
It is sometimes useful to set a new reference "0" that is not the "0" set during the calibration (homing) of the FMod-IPxxMOT.

There are two cases in which to perform that action:

1. A new "0" is sent to the corresponding FMod-IPxxMOT *POSITION* register. Absolute zero is changed, and the software minimum and maximum limitations (*INPUTMIN* & *INPUTMAX* registers) could make no sense because their reference "0" is changed, especially if a home position (homing) is done at power-up.
2. FMod-IPxxMOT *POSITION* register is kept unchanged but an offset value will be added locally in FMod-IPAXESCTRL.

In both cases, the value displayed and all the interactions are the same. However, with the "0 relative reference", the inside FMod-IPxxMOT reference will not change. That is very important when a range of suitable positions (e.g. hardware limitation) is active.

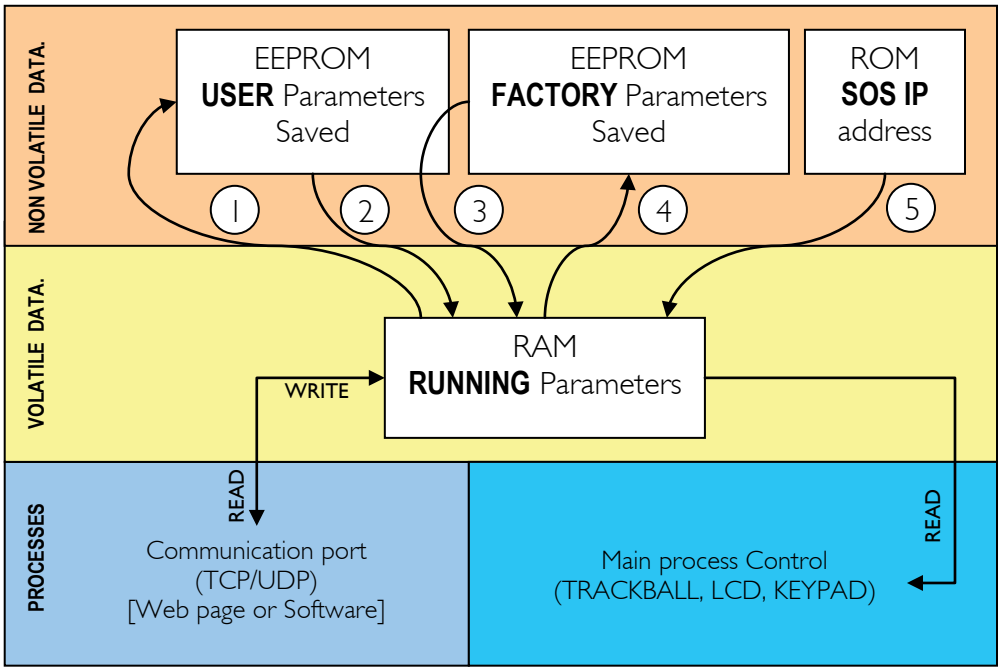
What happens inside (when key function is pressed):
"Set position zero absolute" :
AXIS1POSITIONOFFSET = 0
"Set position zero relative" :
AXIS1POSITIONOFFSET += *POSITION*



8. 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 at power-down. It is required to save them to "User Parameters" or "Factory Parameters", using the corresponding functions.



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 pressing "SOS Button" after power-up
3 + 5 + 1	By pressing "SOS Button" during power-up

Full Register Description

List of registers

Address	Bytes	Name
General Information		
0x00	4	*TYPE*
0x01	4	*VERSION*
0x02	0 (fct)	*RESETCPU*
0x03	0 (fct)	*SAVEUSERPARAMETERS*
0x04	0 (fct)	*RESTOREUSERPARAMETERS*
0x05	0 (fct)	*RESTOREFACTORYPARAMETERS*
0x06	0 (fct)	*SAVEFACTORYPARAMETERS*
0x07	4	*VOLTAGE*
0x08	4	*WARNING*
Communication		
0x10 (16)	4	*COMMUNICATIONOPTIONS*
0x11 (17)	6	*ETHERNETMAC*
0x12 (18)	4	*IPADDRESS*
0x13 (19)	4	*SUBNETMASK*
0x14 (20)	1	*TCPTIMEOUT*
0x15 (21)	16	*MODULENAME*
0x1A (26)	1	*TCPCONNECTIONSOPENED*
General configuration		
0x20 (32)	4	*DISPLAYOPTIONS*
0x21 (33)	4	*TRACKBALLOPTIONS*
0x22 (34)	4	*KEYBOARDOPTIONS*
0x23 (35)	16	*KEYBOARDFUNCTIONS*
0x24 (36)	1	*MODE*
0x28 (40)	1	*LCDCONTRAST*
0x29 (41)	80	*LCDMODE1TEXT*
0x2A (42)	80	*LCDMODE2TEXT*
0x2B (43)	80	*LCDMODE3TEXT*
Axis1		
0x30 (48)	4	*AXIS1POSITIONOFFSET*
0x31 (49)	6	*AXIS1MACADDRESS*
0x32 (50)	4	*AXIS1IPADDRESS*
0x33 (51)	4	*AXIS1MODE1DISPLAYRATIO*
0x34 (52)	4	*AXIS1MODE1HIDRATIO*
0x35 (53)	4	*AXIS1MODE2DISPLAYRATIO*
0x36 (54)	4	*AXIS1MODE2HIDRATIO*
0x37 (55)	4	*AXIS1MODE3DISPLAYRATIO*
0x38 (56)	4	*AXIS1MODE3HIDRATIO*

Lis of registers (continued):

Address	Bytes	Name
Axis2		
0x40 (64)	4	*AXIS2POSITIONOFFSET*
0x41 (65)	6	*AXIS2MACADDRESS*
0x42 (66)	4	*AXIS2IPADDRESS*
0x43 (67)	4	*AXIS2MODE1DISPLAYRATIO*
0x44 (68)	4	*AXIS2MODE1HIDRATIO*
0x45 (69)	4	*AXIS2MODE2DISPLAYRATIO*
0x46 (70)	4	*AXIS2MODE2HIDRATIO*
0x47 (71)	4	*AXIS2MODE3DISPLAYRATIO*
0x48 (72)	4	*AXIS2MODE3HIDRATIO*
Axis3		
0x50 (80)	4	*AXIS3POSITIONOFFSET*
0x51 (81)	6	*AXIS3MACADDRESS*
0x52 (82)	4	*AXIS3IPADDRESS*
0x53 (83)	4	*AXIS3MODE1DISPLAYRATIO*
0x54 (84)	4	*AXIS3MODE1HIDRATIO*
0x55 (85)	4	*AXIS3MODE2DISPLAYRATIO*
0x56 (86)	4	*AXIS3MODE2HIDRATIO*
0x57 (87)	4	*AXIS3MODE3DISPLAYRATIO*
0x58 (88)	4	*AXIS3MODE3HIDRATIO*

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 of a *Type* and *Model* number.
It defines which kind of module it is.
Normally, different *TYPE* modules are not software compatible.

Example:
Module with *TYPE* = 0x00040000 means *Type*=4 (4= IP axes controller), *Model* = 0.

Limits:
None

Active:
Each time the processor is running.

VERSION

Register Address	Register Name	Function	Read/Write Control
0x01	*VERSION*	Software ID	Read only

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

Description:
Firmware identifier composed of a Version and Revision number.
Normally, same Version is backward compatible with different Revision.

Example:
Firmware 0x00010007 = Version 1, Revision 7 is compatible with all earlier revisions of the same version (ver 1.0 to 1.6) but might have new functionalities (which will be deactivated by default) or code optimizations.

Limits:
None

Active:
Each time the processor is running.

RESET CPU

Function Address	Function Name	Function	Read/Write Control
0x02	*RESETCPU*	Restart processor	Write only

Register Size	Register structure	Unit
0 Bytes	none	none

Description:
Restarts the CPU of the board. The communication will be lost.

Active:
Each time the processor is running.

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 Bytes	none	none

Description:
Saves the following parameters in EEPROM:

0x10 *COMMUNICATIONOPTIONS*
0x12 *IPADDRESS*
0x13 SUBNETMASK
0x14 *TCPWATCHDOG*
0x15 *MODULENAME*

0x20 *DISPLAYOPTIONS*
0x21 *TRACKBALLOPTIONS*
0x22 *KEYBOARDOPTIONS*
0x23 *KEYBOARDFUNCTIONS*
0x28 *LCDCONTRAST*
0x29 *LCDMODE1TEXT*
0x2A *LCDMODE2TEXT*
0x2B *LCDMODE3TEXT*

0x30 *AXIS1POSITIONOFFSET*
0x32 *AXIS1IPADDRESS*
0x33 *AXIS1MODE1DISPLAYRATIO*
0x34 *AXIS1MODE1HIDRATIO*
0x35 *AXIS1MODE2DISPLAYRATIO*
0x36 *AXIS1MODE2HIDRATIO*
0x37 *AXIS1MODE3DISPLAYRATIO*
0x38 *AXIS1MODE3HIDRATIO*

0x40 *AXIS2POSITIONOFFSET*
0x42 *AXIS2IPADDRESS*
0x43 *AXIS2MODE1DISPLAYRATIO*
0x44 *AXIS2MODE1HIDRATIO*
0x45 *AXIS2MODE2DISPLAYRATIO*
0x46 *AXIS2MODE2HIDRATIO*
0x47 *AXIS2MODE3DISPLAYRATIO*
0x48 *AXIS2MODE3HIDRATIO*

0x50 *AXIS3POSITIONOFFSET*
0x52 *AXIS3IPADDRESS*
0x53 *AXIS3MODE1DISPLAYRATIO*
0x54 *AXIS3MODE1HIDRATIO*
0x55 *AXIS3MODE2DISPLAYRATIO*
0x56 *AXIS3MODE2HIDRATIO*
0x57 *AXIS3MODE3DISPLAYRATIO*
0x58 *AXIS3MODE3HIDRATIO*

Limits:
Do not change any of these parameters during 1 sec while saving!

Active:
Each time the processor is running.

RESTORE USER PARAMETERS

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

Register Size	Register structure	Unit
0 Bytes	none	none

Description:
Restores the user parameters from EEPROM.

See *SAVEUSERPARAMETERS* (0x03) register list.

Active:
Each time the processor is running.

RESTORE FACTORY PARAMETERS

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

Register Size	Register structure	Unit
0 Bytes	none	none

Description:
Restores factory parameters.

See *SAVEUSERPARAMETERS* (0x03) register list.

Active:
Each time the processor is running
SAVEUSERPARAMETERS must be performed after this function for next reboot to keep these parameters.

SAVE FACTORY PARAMETERS

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

Register Size	Register structure	Unit
0 Bytes	none	none

Description:
This function is reserved for integrator engineers, and not for end-user.
Use it when all parameters have been approved for an application. It saves in EEPROM all configurable registers for both factory parameters and user parameters.

Warning: This function also executes the *SAVEUSERPARAMETERS* function.

See *SAVEUSERPARAMETERS* (0x03) saved register list.

Limits:
Do not change any of these parameters during 1 sec while saving!

Active:
Each time the processor is running.

VOLTAGE

Register Address	Register Name	Function	Read/Write Control
0x07	*VOLTAGE*	Power module 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 (2311527/65536)

Information:
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
0x07	*WARNING*	Bit to bit state	R/W

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

Description:
This register is reserved for future use.

Active:
Each time the processor is running,

COMMUNICATION OPTIONS

Register Address	Register Name	Function	Read/Write Control
0x10 (16)	*COMUNICATIONOPTIONS*	Bit to bit settings	Write (Read)

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

Description:
This register is reserved for future use.

Active:
Each time the processor is running

ETHERNET MAC

Register Address	Register Name	Function	Read/Write Control
0x11 (17)	*ETHERNETMAC*	Hardware network ID	Read only

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

Description:
Standard hardware unique identifier (world) for each device on an ethernet network.

Information:
If user writes into this register, the address will not be modified. This register is available only for information 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 communication ports are closed.
Do not forget to run 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.

Active:
Each time the processor is running.

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 communication ports are closed.
Do not forget to run a *SAVEUSERPARAMETERS* command.

If you do not want to use subnets, use one of the following subnet masks 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

Active:
Each time the processor is running.

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 board 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

Active:
Each time the processor is running.

MODULE NAME

Register Address	Register Name	Function	Read/Write Control
0x15 (21)	*MODULENAME*	Module's ASCII name	R/W

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

Description:
Name 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.

Active:
Each time the processor is running.

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 card using TCP.
Value can be 0 to 4.

Active:
Each time the processor is running.

DISPLAY OPTIONS

Register Address	Register Name	Function	Read/Write Control
0x20 (32)	*DISPLAYOPTIONS*	Bit to bit settings	Write (Read)

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

Description:
This register is reserved for future use.

Active:
Each time the processor is running.

TRACKBALL OPTIONS

Register Address	Register Name	Function	Read/Write Control
0x21 (33)	*TRACKBALLOPTIONS*	Bit to bit settings	Write (Read)

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

Description:
This register is reserved for future use.

Active:
Each time the processor is running.

KEYBOARD OPTIONS

Register Address	Register Name	Function	Read/Write Control
0x22 (34)	*KEYBOARDOPTIONS*	Bit to bit settings	Write (Read)

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

Description:
This register is reserved for future use.

Active:
Each time the processor is running.

KEYBOARD FUNCTIONS

Register Address	Register Name	Function	Read/Write Control
0x23 (35)	*KEYBOARDFUNCTIONS*	A function for each key	Write (Read)

Register Size	Register structure	Unit
16 Byte	16 (only) x Unsigned Byte	none

Description:
On the keyboard, there are a number of keys but each function is not already defined to a dedicated key.
KEYBOARDFUNCTIONS register is able to link a programmed function with a key.
Each Byte represents a key (0-15). Line L0-column C0 = Byte 0, L0-C1= Byte 1, L3-C3= Byte 15.
The value of that Byte represents a function (0x00-0xFF). Not all functions already exist, see below.

The functionality of each button can be chosen between this list of values:

0x00	None
0x01	Set all axes to Position control mode/ Homing
0x02	Stop all axes (break mode)
0x03	Free all axes
0x04	Free selected axes
0x05	Set all axes to Position mode
0x06	Increment axis 1 / Repeat
0x07	Decrement axis 1 / Repeat
0x08	Increment axis 2 / Repeat
0x09	Decrement axis 2 / Repeat
0x0A	Increment axis 3 / Repeat
0x0B	Decrement axis 3 / Repeat
0x0C	Change mode (1,2,3,1...)
0x0D	Mode 1
0x0E	Mode 2
0x0F	Mode 3
0x10	Axis 1 ON-OFF
0x11	Axis 2 ON-OFF
0x12	Axis 3 ON-OFF
0x13	Axis 1 suspended while pressing
0x14	Axis 2 suspended while pressing
0x15	Axis 3 suspended while pressing
0x16	Go / Set position zero absolute
0x17	Go / Save position 1
0x18	Go / Save position 2
0x19	Go / Save position 3
0x1A	Go / Save position 4

0x1A	Go / Save position 5
0x1C	Go / Save position 6
0x1D	Go / Save position 7
0x1E	Go / Save position 8
0x1F	Go / Save position 9
0x20	Go / Save position 10
0x21	Inc. axis 1 in Mode 1 / Repeat
0x22	Dec. axis 1 in Mode 1 / Repeat
0x23	Inc. axis 2 in Mode 1 / Repeat
0x24	Dec. axis 2 in Mode 1 / Repeat
0x25	Inc. axis 3 in Mode 1 / Repeat
0x26	Dec. axis 3 in Mode 1 / Repeat
0x27	Inc. axis 1 in Mode 2 / Repeat
0x28	Dec. axis 1 in Mode 2 / Repeat
0x29	Inc. axis 2 in Mode 2 / Repeat
0x2A	Dec. axis 2 in Mode 2 / Repeat
0x2B	Inc. axis 3 in Mode 2 / Repeat
0x2C	Dec. axis 3 in Mode 2 / Repeat
0x2D	Inc. axis 1 in Mode 3 / Repeat
0x2E	Dec. axis 1 in Mode 3 / Repeat
0x2F	Inc. axis 2 in Mode 3 / Repeat
0x30	Dec. axis 2 in Mode 3 / Repeat
0x31	Inc. axis 3 in Mode 3 / Repeat
0x32	Dec. axis 3 in Mode 3 / Repeat
0x33	Mode 3 → Mode 1 (when released)
0x34	Mode 2 → Mode 1 (when released)
0x35	Go / Set position zero relative

Note: All description with "/" have a different functionality if the key is pressed shortly (<3 seconds) or pressed for more than 3 seconds.

Limits:
For each Byte:
Min: 0x00 (0)
Max: 0x35 (53)
Default: 0x00

Example:
With a 1 line and 4 columns Keyboard, with "Set all axes to Position mode" (0x01), increment (0x06), decrement Axis 1 (0x07) and "Go/Save position 1" (0x17) , write:
0x00000000000000000000000017070601

Active:
Each time the processor is running.

MODE

Register Address	Register Name	Function	Read/Write Control
0x24 (36)	*MODE*	Identifier of interface mode	Read (Write)

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

Description:
There are 3 independent modes. The value of this register is very important, since it decides which set of other registers are in use.

It can be changed with the communication protocol or with a keyboard functions.

Each mode has a specific LCD display and a specific input/output ratio (input=HID/output=Display).

Here is the list of registers selected with *MODE*=1:

0x29 *LCDMODE1TEXT*

0x33 *AXIS1MODE1DISPLAYRATIO*
0x34 *AXIS1MODE1HIDRATIO*

0x43 *AXIS2MODE1DISPLAYRATIO*
0x44 *AXIS2MODE1HIDRATIO*

0x43 *AXIS3MODE1DISPLAYRATIO*
0x44 *AXIS3MODE1HIDRATIO*

Limits:
Min: 1
Max: 3

Default:
1 (at power up)

LCD CONTRAST

Register Address	Register Name	Function	Read/Write Control
0x28 (40)	*LCDCONTRAST*	Define LCD contrast	Write (Read)

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

Description:
This value is converted to a voltage which is needed by the LCD for its contrast.

WARNING: HIGH CONTRAST CAN REDUCE THE LIFE OF THE LCD.

To select the contrast of the LCD, begin at 0 and increase it until the visibility is good. If you keep increasing it, you will probably not see a real difference, but the LCD will not appreciate it on the long term. Too high levels can be detected after a long use (>1 day) when older characters (image retention) still appear on the LCD positions that should be filled with blank (space) characters.

Each LCD has an operating voltage pin (Vo) depending of LCD type and temperature.
Typically at 25°C Vo is ~0.5V (Vdd-Vo=4.5v).

Examples:

0-127	T°>25°C
128-255	T<25°C

Limits:

Min:	0	~2V (for 50°... 70°)
Max:	255	~0V (for 0 ... -20°C)

Default:
127 = ~0.5V for 25°

Active:
Each time the processor is running.

LCD MODE1 TEXT

Register Address	Register Name	Function	Read/Write Control
0x29 (41)	*LCDMODE1TEXT*	Mode's 1 display text	Write (Read)

Register Size	Register structure	Unit
80 Byte (only)	80 x Char (Unsigned Byte)	none

Description:
When register *MODE*=1, the text defined with *LCDMODE1TEXT* is displayed on the LCD.

The first line is static (text), while the other 3 lines are dedicated to one axis each (line2=axis1, line3=axis2, line4=axis3). A line ALWAYS has 20 characters.

Each axis position is multiplied with *AXIS(i)MODE(j)DISPLAYRATIO* , (i,j=1,2,3) depending on the mode used. The integer result can be displayed on the corresponding line on the LCD.
8 digits and 1 more character for the sign are displayable. Values range from -16'777'216 to16'777'215. With bigger values, "MAX" is displayed, and with lower values "MIN" is displayed. To define the place of each character, put "#" in the text. If all 9 "#" are not defined, the least significant characters are skipped.
For example: if -07'654'321 is the result of a multiplication. The 2 LSB are useless, only 7 # will shows the good result: -076543.

The 2 last characters of the line are always overwritten with the status of the axis, so do not use them.

For ASCII table compatibility, see the LCD datasheet pattern generator to know which characters are displayable.

Example:

	Byte Id	01234567890123456789
Byte range		
0 ... 19		"Module Mode 1 "
20 ... 39		"Axis1 ####.### um __"
40 ... 59		"2 ###'###'### pls __"
60 ... 79		"3####mm ###um (#) __"

WARNING: Be sure to always send 20 characters for each line (4x), for a total of 80 characters (bytes) to this register.

LCD MODE2 TEXT

Register Address	Register Name	Function	Read/Write Control
0x2A (42)	*LCDMODE2TEXT*	Mode's 2 display text	Write (Read)

Register Size	Register structure	Unit
80 Byte	80 x Char (Unsigned Byte)	none

Description:
When register *MODE*=2, the text defined with *LCDMODE2TEXT* is displayed on the LCD.
See *LCDMODE1TEXT* (0x29) for more details.

LCD MODE3 TEXT

Register Address	Register Name	Function	Read/Write Control
0x2B (43)	*LCDMODE3TEXT*	Mode's 3 display text	Write (Read)

Register Size	Register structure	Unit
80 Byte	80 x Char (Unsigned Byte)	none

Description:
When register *MODE*=3, the text defined with *LCDMODE3TEXT* is displayed on the LCD.
See *LCDMODE1TEXT* (0x29) for more details.

AXIS 1 – POSITION OFFSET

Register Address	Register Name	Function	Read/Write Control
0x30 (48)	*AXIS1POSITIONOFFSET*	define relative reference	R/W

Register Size	Register structure	Unit
4 Byte	Signed (2's complement) Int 32	Pulse

Description:
Creates an offset with corresponding *POSITION* (and *INPUT)* register of axis 1.
When this register is updated with Key function "Set zero absolute" or "Set zero relative", its value is automatically saved in User-EEPROM.

Limits:
Max 0x7FFFFFFF = 2'147'483'647
Min 0x80000000 = -2'147'483'648

Default:
0

Active:
Each time the processor is running. To disable relative reference, write 0 or press Key with function "Set zero absolute".

AXIS 1 - MAC ADDRESS

Register Address	Register Name	Function	Read/Write Control
0x31 (49)	*AXIS1MACADDRESS*	Hardware network ID	Read only

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

Description:
Standard ethernet network's hardware unique identifier for axis number 1.

Information:
This register is available only for information purposes.

Default:
0xFFFFFFFFFFFF while a connection has not already been made.

AXIS 1 - IP ADDRESS

Register Address	Register Name	Function	Read/Write Control
0x32 (50)	*AXIS1IPADDRESS*	IP network ID	Write (Read)

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

Description:
Network identifier for axis number 1 used for UDP/IP.
To disable the use of this axis, write 0x00000000 to this register.
The values 255 (0xFF) are reserved for broadcast and network address and should not be used in this register.

Default:
169.254.5.5 (Axis 1) = default IP address of module FMod-IPDCMOT or FMod-IPECMOT.
169.254.5.6 (Axis 2)
169.254.5.7 (Axis 3)

Example:
For the IP=192.168.16.14 (0xC0, 0xA8, 0x10, 0x0E) , write 0xC0A8100E to *AXIS1IPADDRESS.*

AXIS 1 - MODE 1 - DISPLAY RATIO

Register Address	Register Name	Function	Read/Write Control
0x33 (51)	*AXIS1MODE1DISPLAYRATIO*	Convert the display value	Write (Read)

Register Size	Register structure	Unit
4 Byte	Float 32 bits , (IEEE 754-1985)	none

Description:
While *MODE*=1 only. For axis 1 only.
The incoming position (integer value) of the axis is multiplied with this value and the result (integer value) is displayed on the corresponding axis line on the LCD.

Default:
1.0

Example:
If you want to display micrometers (while *MODE*=1), and 1000 increments of axis 1 correspond to 1um, set *AXIS1MODE1DISPLAYRATIO* to 0.001 (1/1000).

Active:
Each time the processor is running,

AXIS 1 - MODE 1 - HID RATIO

Register Address	Register Name	Function	Read/Write Control
0x34 (52)	*AXIS1MODE1HIDRATIO*	Convert human input to motor goal	Write (Read)

Register Size	Register structure	Unit
4 Byte	Float 32 bits , (IEEE 754-1985)	none

Description:
While *MODE*=1 only. For axis 1 only.
When the end-user interacts with the keyboard or with the trackball to move an axis, the human interface device (HID) output is multiplied with *AXIS1MODE1HIDRATIO* and the result is sent to the motor's new goal.

Default:
1.0

Example:
For big motor movement, an HID increment (a key pressed once, or a trackball pulse) can result in multiple position increments on the motor's side. *AXIS1MODE1HIDRATIO* > 1.

For small motor movement, multiple HID increments (a key pressed repetitively or several trackball pulses) can result in 1 position increments on the motor's side. *AXIS1MODE1HIDRATIO* < 1.

Active:
Each time the processor is running,

AXIS 1 - MODE 2 - DISPLAY RATIO

Register Address	Register Name	Function	Read/Write Control
0x35 (53)	*AXIS1MODE2DISPLAYRATIO*	Convert the display value	Write (Read)

Register Size	Register structure	Unit
4 Byte	Float 32 bits	none

Description:
For *MODE*=2 and axis 1 only.
See *AXIS1MODE1DISPLAYRATIO* (0x33) for details.

AXIS 1 - MODE 2 - HID RATIO

Register Address	Register Name	Function	Read/Write Control
0x36 (54)	*AXIS1MODE2HIDRATIO*	Convert human input to motor goal	Write (Read)

Register Size	Register structure	Unit
4 Byte	Float 32 bits	none

Description:
While *MODE*=2 only. For axis 1 only.
See *AXIS1MODE1HIDRATIO* (0x34) for details.

AXIS 1 - MODE 3 - DISPLAY RATIO

Register Address	Register Name	Function	Read/Write Control
0x37 (55)	*AXIS1MODE3DISPLAYRATIO*	Convert the display value	Write (Read)

Register Size	Register structure	Unit
4 Byte	Float 32 bits	none

Description:
While *MODE*=3 only. For axis 1 only.
See *AXIS1MODE1DISPLAYRATIO* (0x33) for details.

AXIS 1 - MODE 3 - HID RATIO

Register Address	Register Name	Function	Read/Write Control
0x38 (56)	*AXIS1MODE3HIDRATIO*	Convert human input to motor goal	Write (Read)

Register Size	Register structure	Unit
4 Byte	Float 32 bits	none

Description:
While *MODE*=3 only. For axis 1only.
See *AXIS1MODE1HIDRATIO* (0x34) for details.

Other AXIS 2 registers

Register Addresses	Register concerned	See Axis 1
0x40 – 0x48	All Axis 2 registers	0x30 – 0x38

The use of these registers (0x40-0x48) is the same as described for AXIS 1, but applied to AXIS 2. Please refer to the corresponding pages in this manual.

Other AXIS 3 registers

Register Addresses	Register concerned	See Axis 1
0x50 – 0x58	All Axis 3 registers	0x30 – 0x38

The use of these registers (0x50-0x58) is the same as described for AXIS 1, but applied to AXIS 3. Please refer to the corresponding pages in this manual.

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