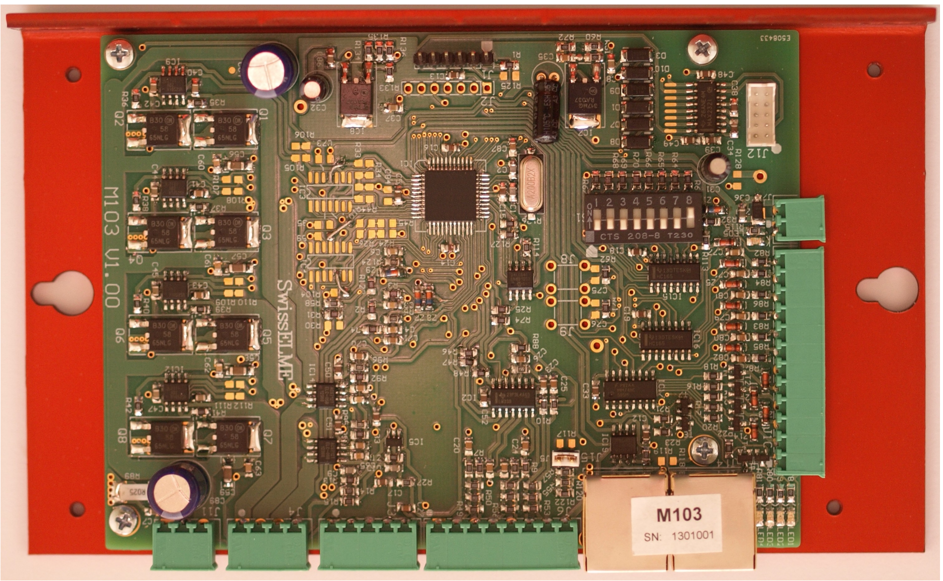
SwissELME

Operational Manual



Stepper motor controller
BLDC motor controller
DC motor controller
Linear motor controller

M101 – M102 – M103

SwissELME
Vic. Motto di Lena 3a
CH-6648 Minusio, Switzerland

Tel. +41 (0)79 221 42 91
Fax +41 (0)91 730 19 81
info@swisselme.com

1 Editorial

This manual gives an explanation of the M101 board which is capable of driving several kinds of motors.

The board was developed in cooperation with the market which was not satisfied by the poor performance or pricing of existing boards.

As the board can be driven by PC, PLC, µ-Processors but also as Stand Alone and can drive Brush-less DC, Stepper, DC and Linear Motors (with or without encoder), do we believe to offer you an interesting product.

The SwissELME-team

PLEASE NOTE:

Although this product is developed with great care, the decision to use this board in whatever application is the responsibility of the user. The producers and developers do not take any responsibility for damages, injuries, etc. created by the use of this product.

2 Index

M101 Manual

3 Specification

3.1 Applications

Suitable for a wide range of stepping motors, for example NEMA size 11, 17, 23 and 34. It can be used in various kinds of machines, such as X-Y tables, engraving machines, labeling machines, laser cutters, pick-place devices, and so on. Particularly adapt to the applications desired with low noise, low heating, high speed and high precision.

3.2 General description

The M101 is designed to drive a lot of different types of motors:

- Linear motor 2 phases (like LinMot Motors)

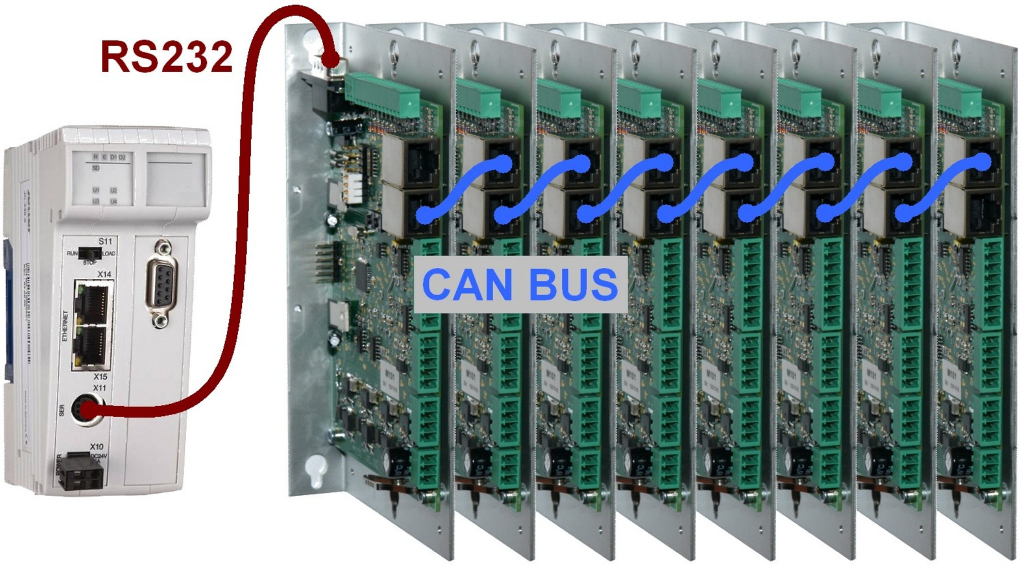
- Linear motor 3 phases (like Faulhaber Linear DC Servomotors)

- Stepper motor 2 phases (Hybrid)

- Stepper motor 3 phases (Hybrid)

- DC motor

- BLDC motor 3 phases (brush-less DC-Motor)



3.3 Linear Motor

Ultra low speed and high positioning resolution (1/3072 pole pitch) are thus possible. A supplementary external high resolution encoder increases the performances.

3.4 Stepper Motor

The M101 is a high performance micro-stepping drive based on pure-sinusoidal current control technology developed by SwissELME. The driven motors can run with smaller noise, lower heating, smoother movement and have better performances at higher speed than most of the drives in the markets. It is suitable for driving 2-phase and 3-phase hybrid stepping motors.

Ultra low speed and very high positioning resolution (pure-sinusoidal current control technology) are thus possible, without creating vibration due to the resonance frequency of the motor. An external encoder allows to monitor the step lost. The optimized movement algorithm reduces a lot of vibration, especially at low speeds, and at the same time is possible to make fast movements (>10000 full-step/s).

3.5 Optimized performances

Setting rump-up and rump-down acceleration value, and maximal speed value, it is possible to modify the movement profile on the fly.

Positioning with high resolution, including limit switches and zero referencing.

3.6 New HW technology

Power MOSFETs with minimal Rdson (on-resistance) allows high current without need of a supplementary heat sink. High performance DSP controller.

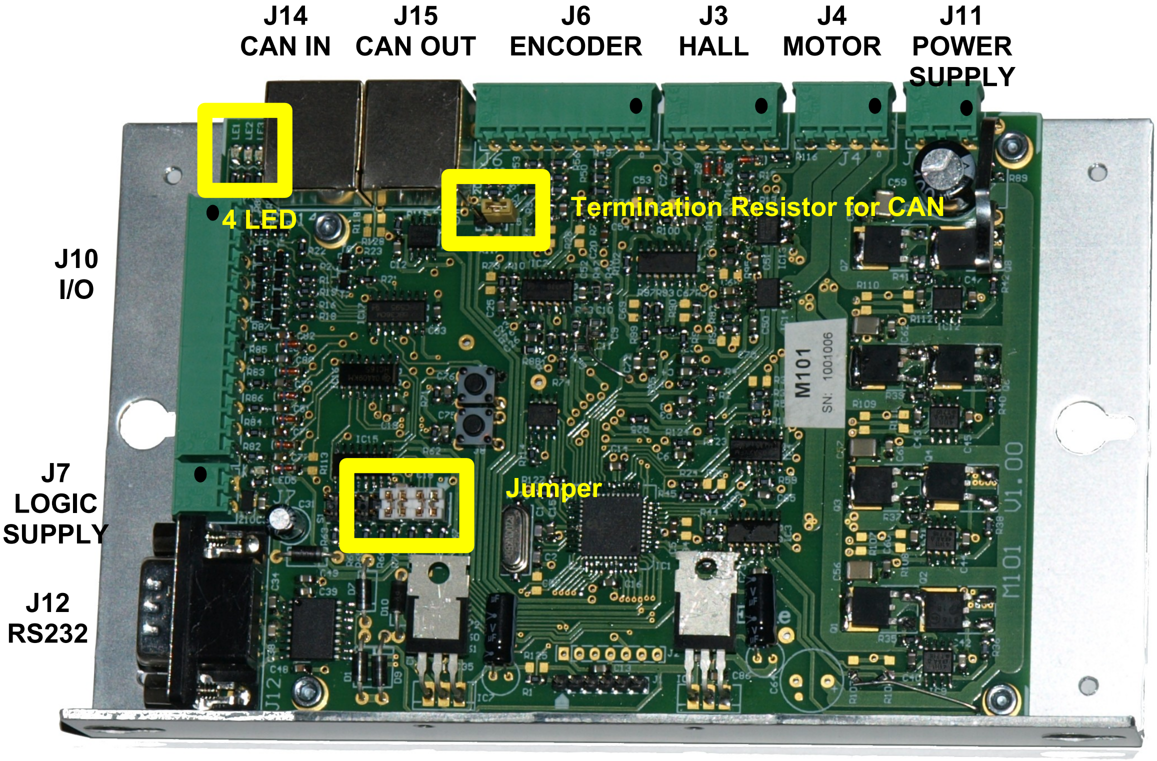
3.7 Features

Power supply for electronic	11.5 .. 12.5	V DC
Power supply for motor	12 .. 48 standard 12 .. 72 high voltage electronic	V DC
Max. continuous output current	7.0 pro Phase	A
Max. peak output current	9.0 (M102), 5.0 (M101)	A
Total standby logic current	190	mA
PWM switching frequency	20 (with 40kHz regulation)	kHz

Digital inputs **Signal level low** **Signal level high** **Current absorption max** **Scanning rate** **Input filter**	6 inputs, programmable, 5..24V 0 .. 2 3.5 .. 24 20 40 (from firmware version 2.32) 16	 V V mA kHz kHz
Digital outputs **Max voltage** **Current max** **Scanning rate**	4 transistor outputs open collector 24 50 5	 V mA kHz
Speed range for stepper motor	0 .. 10000	full-step/s
Speed range for BLDC motor	0 .. 60000	rpm
Speed range for DC motor	0 .. 30000	rpm
Speed range for linear motor	0 .. 5	m/s
Scanning rate	10	kHz
RS232 Interface	max 115.2, default 19.2	kBaud
CAN Interface (between more M101, M102)	Max. 1000, default 125	kbit/s
Operating mode for stepper motor	pure-sinusoidal current control technology for Stepper Motor	
Operating mode	positioning, homing, programmable, automatic idle-current reduction, 1 analog input 0..10V (adjustable)	
Position monitoring	yes, depending on encoder resolution	
Protection circuit	Over-voltage, under-voltage, Short-voltage, over-current, short-circuit	
Current drop	programmable and adjustable in 1% steps	
Temperature range	0 to 40	°C
Weight	210	g
Dimension of print-support	180 x 106 x 24	mm
External digital encoder RS422	max frequency 500	kHz
Encoder resolution with hall sensors for 3-phases Linear motor and BLDC	3072	Inc/Tm
Encoder resolution with hall sensors for 2-phases Linear motor	2047	Inc/Tm
Resolution with external digital encoder	max: 65535	Inc./Tm

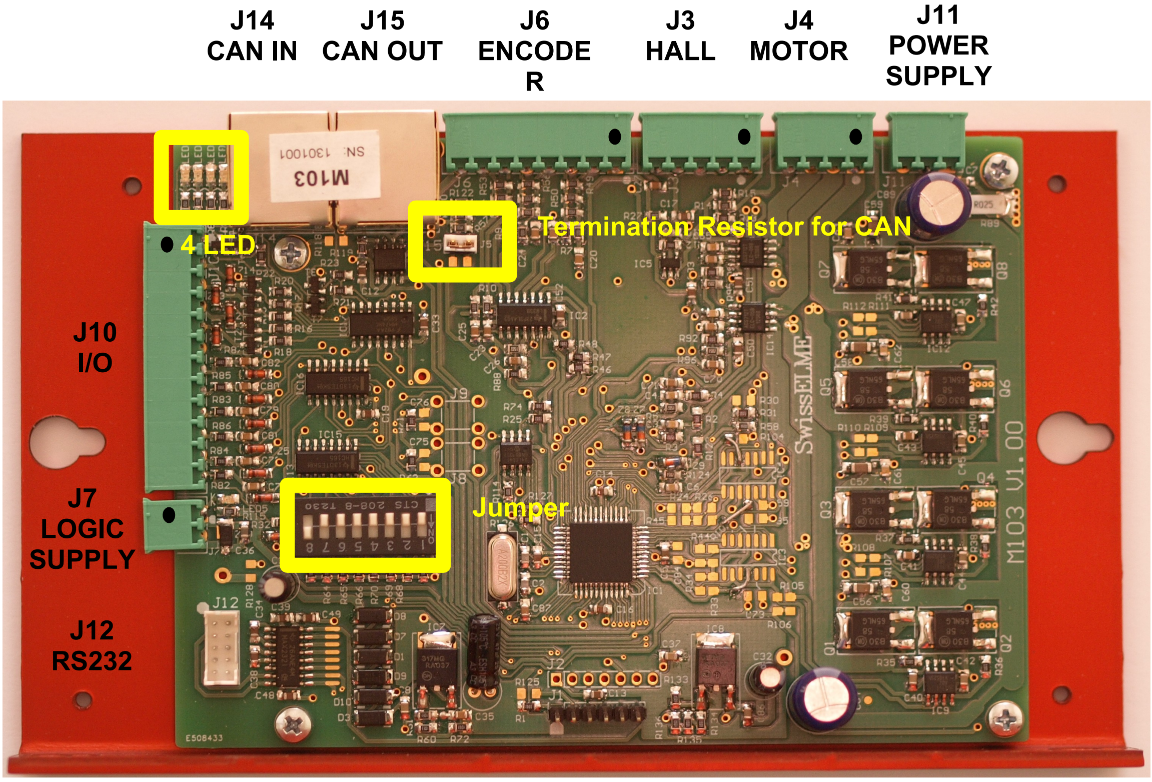
4 Connections and wiring for M101

4.1 Connections and wiring for M101 + M102



● Is the pin number 1 of each connector.

4.2 Connections and wiring for M103



● Is the pin number 1 of each connector.

4.3 J14 - CAN IN

Pin 1	BUS	CAN HIGH
Pin 2	BUS	CAN LOW
Pin 3	OUTPUT	GND
Pin 4		n.c.
Pin 5		n.c.
Pin 6	-	SHIELD
Pin 7	OUTPUT	GND
Pin 8		n.c.

4.4 J15 - CAN OUT

Pin 1	BUS	CAN HIGH
Pin 2	BUS	CAN LOW
Pin 3	OUTPUT	GND
Pin 4		n.c.
Pin 5		n.c.
Pin 6	-	SHIELD
Pin 7	OUTPUT	GND
Pin 8		n.c.

4.5 J6 - Digital encoder connector

Pin 1	OUTPUT	GND
Pin 2	INPUT	A
Pin 3	INPUT	!A
Pin 4	INPUT	B
Pin 5	INPUT	!B
Pin 6	INPUT	Z
Pin 7	INPUT	!Z
Pin 8	OUTPUT	+5V

4.6 J3 - Hall motor sensor connector

Pin 1	OUTPUT	GND
Pin 2	INPUT	Hall A
Pin 3	INPUT	Hall B
Pin 4	INPUT	Hall C
Pin 5	OUTPUT	+5V

4.7 J4 - Motor connector

Pin 1	OUTPUT	Phase A+
Pin 2	OUTPUT	Phase A-
Pin 3	OUTPUT	Phase B+
Pin 4	OUTPUT	Phase B-

4.8 J11 - Motor power supply

Pin 1	INPUT	POWER GND
Pin 2	-	n.c.
Pin 3	INPUT	POWER SUPPLY 18V..48V (absolute max. 55V)

On the board between POWER GND and LOGIC GROUD there is a 10 ohm resistor.

The permissible operating voltage for the M101 lies between +12 and +48 V DC; it must not exceed 50V or fall below 10V.

DANGER of electrical over-voltage

Accidentally swapping the connections will destroy the output stage. Never disconnect motor while operating voltage is applied! Never hot-unplug lines.

4.9 J10 - I/O

Pin 1	OUTPUT	GND
Pin 2	OUTPUT	OUTPUT 1 open collector Pending error (2.5kHz)
Pin 3	OUTPUT	OUTPUT 2 open collector Reserved (2.5kHz)
Pin 4	OUTPUT	OUTPUT 3 open collector IN-Position (2.5kHz)
Pin 5	OUTPUT	OUTPUT 4 open collector Communication OK (2.5kHz)
Pin 6	OUTPUT	+12V
Pin 7	INPUT	INPUT 1 Refresh 2.5kHz
Pin 8	INPUT	INPUT 2 Refresh 2.5kHz
Pin 9	INPUT	INPUT 3 Refresh 10kHz
Pin 10	INPUT	INPUT 4 Refresh 10kHz
Pin 11	INPUT	INPUT 5 Refresh 10kHz
Pin 12	INPUT	INPUT 6 Refresh 10kHz

4.10 J7 - Logic supply

Pin 1	INPUT	LOGIC GND
Pin 2	INPUT	LOGIC SUPPLY 12V +/-10%

4.11 J12 - DSUB9 RS232

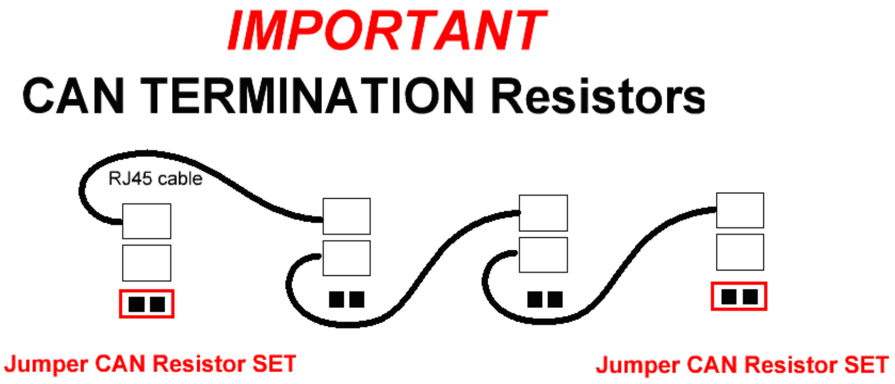
Pin 2	OUTPUT	TX
Pin 3	INPUT	RX
Pin 5	OUTPUT	GND

4.12 Jumper on board

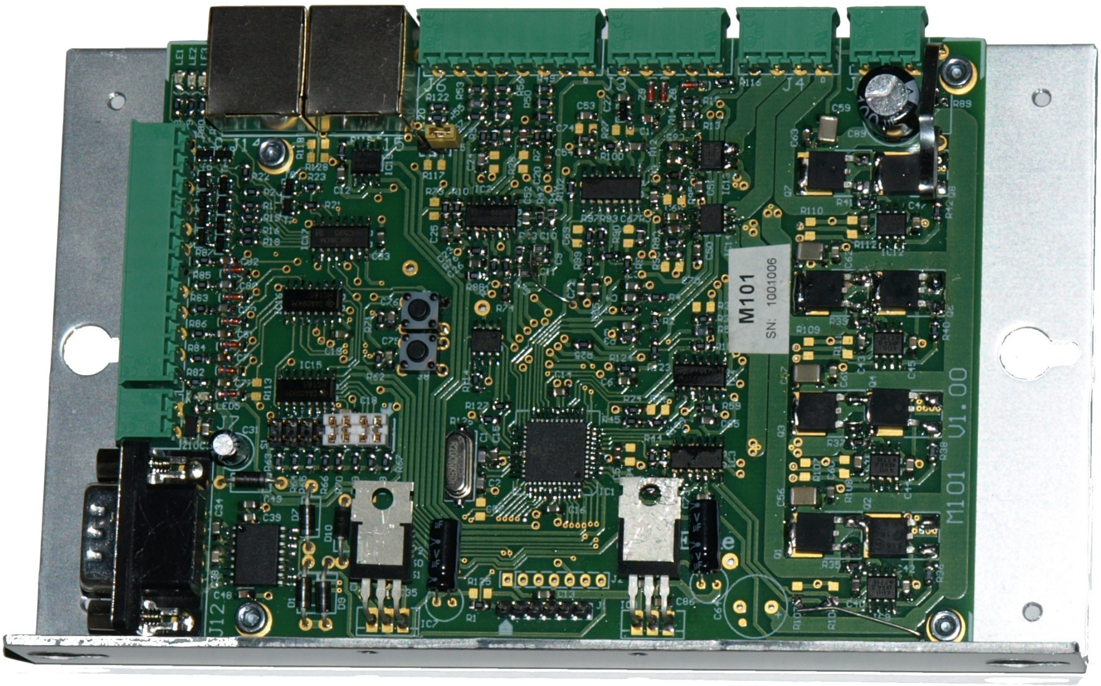
See chapter 8 (Jumpers).

4.13 CAN bus and terminations

When connecting more than one 101 Board, termination resistors are needed to finish the CAN bus electrically correct. Even with only 2 boards this is needed to avoid unwanted behavior of the boards. The boards are connected among them with a standard RJ45 cable.



4.14 Board dimensions



Print: 100mm x 135mm x 18mm

Print with connectors: 105mm x 143mm x 18mm

Board support: 104mm x 172mm x 23mm

5 Modes of operation

5.1 Personal computer

The board can be operated from a PC or Embedded PC through the RS232 port. An USB-RS232 converter can be used as well. Which program is used to make the movements depends on the application. MS Excel...

Underneath are used several ways of operating the M101 or M102. The setup of the programs is roughly described and a simple demo program is written to show a movement of a stepper motor without encoder.

The motor will be set up and made to move 5 rotations to one side and 5 back to the initial position.

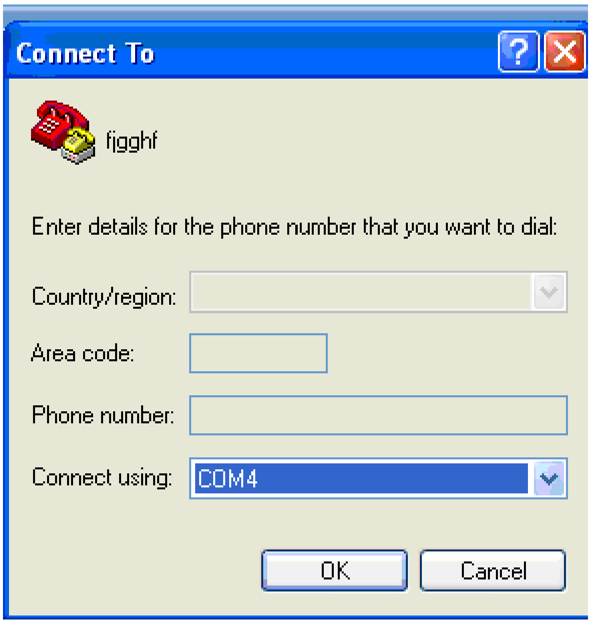
5.1.1 Hyper terminal

This is the most direct way to operate the board.

Connect the M101 or M102 with an Null Modem cable (crossed Transmit/Receive wires), start the Hyper terminal, Do NOT power the 12Vdc yet.

Set up the Hyper terminal:

Set the baud-rate at 19200, 8 Data bits, No Parity, 1 Stop bit, No Flow control.



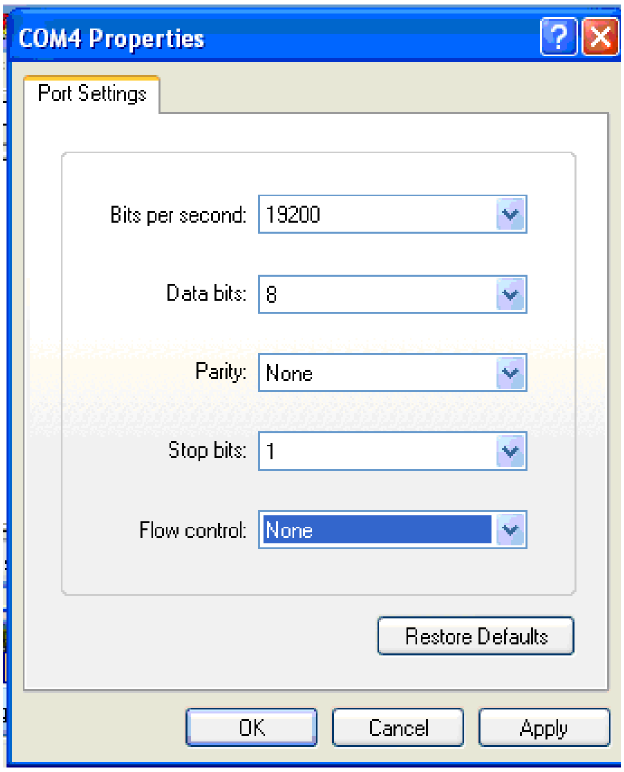
Enter details for the phone number that you want to dial:

Country/region:

Area code:

Phone number:

Connect using: COM4



Bits per second: 19200

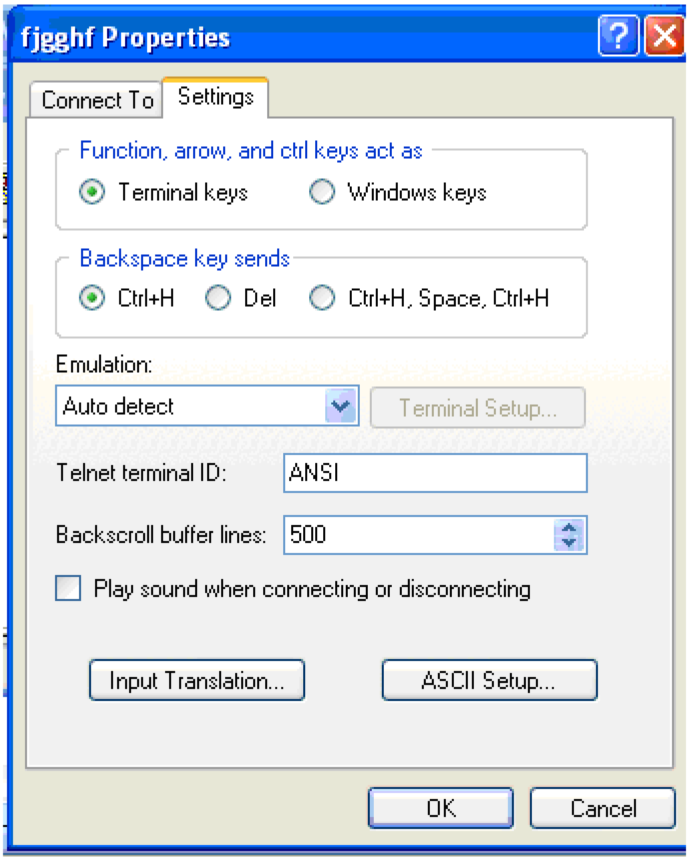
Data bits: 8

Parity: None

Stop bits: 1

Flow control: None

Set the ASCII values as in the figure below.



fjgghf Properties

Connect To | Settings

Function, arrow, and ctrl keys act as

● Terminal keys ○ Windows keys

Backspace key sends

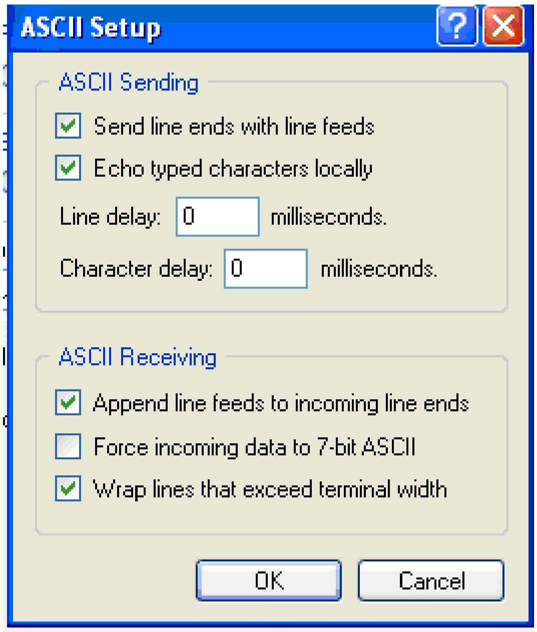
● Ctrl+H ○ Del ○ Ctrl+H, Space, Ctrl+H

Emulation: Auto detect

Telnet terminal ID: ANSI

Backscroll buffer lines: 500

□ Play sound when connecting or disconnecting



ASCII Setup

ASCII Sending

☑ Send line ends with line feeds

☑ Echo typed characters locally

Line delay: 0 milliseconds.

Character delay: 0 milliseconds.

ASCII Receiving

☑ Append line feeds to incoming line ends

□ Force incoming data to 7-bit ASCII

☑ Wrap lines that exceed terminal width

Now switch on the power. Hyper terminal will respond with a message like:

```
M101 version x.xx
```

Now the board is ready to receive instructions.

Every time an instruction is given, the M101 or M102 will repeat the 2 letters of the instruction with the accepted value. If not, the instruction was not correctly given or not understood by the board.

5.2 Programmable controllers PLC's / Micro processors

5.2.1 PLC's with RS232 abilities

The M101 or M102 board uses the RS232 to communicate with the PLC or Microprocessor. The TX and RX signals uses -10V/+10V.

The card always works in polling and doesn't send any spontaneous information. Only exception is when the board is powered on.

The card takes less than 0.1ms response to a serial request. If the request is routed to an other board via CAN, the answer comes after about 1.5ms (CAN 125kBaud). All times are

RS232 communication excluded.

5.2.1.1 ASCII Protocol with 1 M101 or M102 board

ASCII protocol means that the data bytes are in ASCII format. The format of the protocol start with an exclamation mark, then there are 2 characters defining a command, and then there are the data. Not all instruction needs data bytes. Some instructions needs 4 data bytes, other instructions need 8 data bytes. All data are given in HEX format. Here an example:

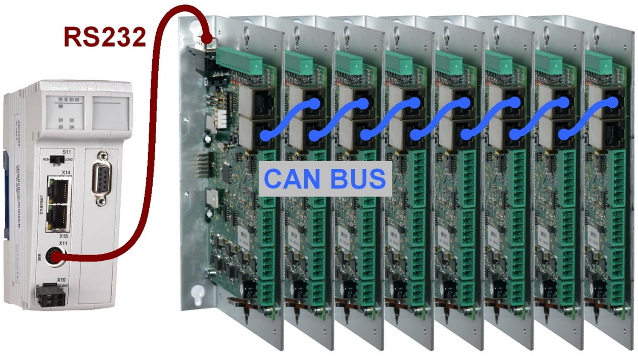
```
! M T 0 0 0 2 [CR]
```

MT is the motor type configuration. 0002 is the stepper motor 2 phases. The carriage return (CR = 0x0D) is needed to complete the instruction. Is is possible to write the instruction as follow too:

```
! M T 2 [CR]
```

If there are communication problems, to empty the internal RS232 buffer, write 10 times the [CR].

5.2.1.2 ASCII Protocol with more M101 or M102 boards connected with CAN bus



Every board have a board-address, defined with the jumpers. The first board can be accessed using the protocol described in the previous chapter. To access the other board is needed an other protocol, in witch is defined the destination board-address. All data are given in HEX format.

```
@ 1 5 M T 0 0 0 2 [CR]
```

The first board, if the address (in this case 0x15 = 21) in the protocol is different as the board-address itself, does send ahead the instruction through the CAN bus.

5.2.1.3 Binary Protocol

The binary protocol is useful to save communication time and it has a check-sum too. All data are given in HEX format.

```
& M T [0x00] [0x02] [5C]
```

Only six bytes are needed. The last byte is the check-sum, so the sum of all bytes must equals zero.

If there are communication problems, to empty the internal RS232 buffer, write 10 times the [CR] or [0x00].

5.2.1.4 Binary Protocol with more M101 or M102 boards connected with CAN bus

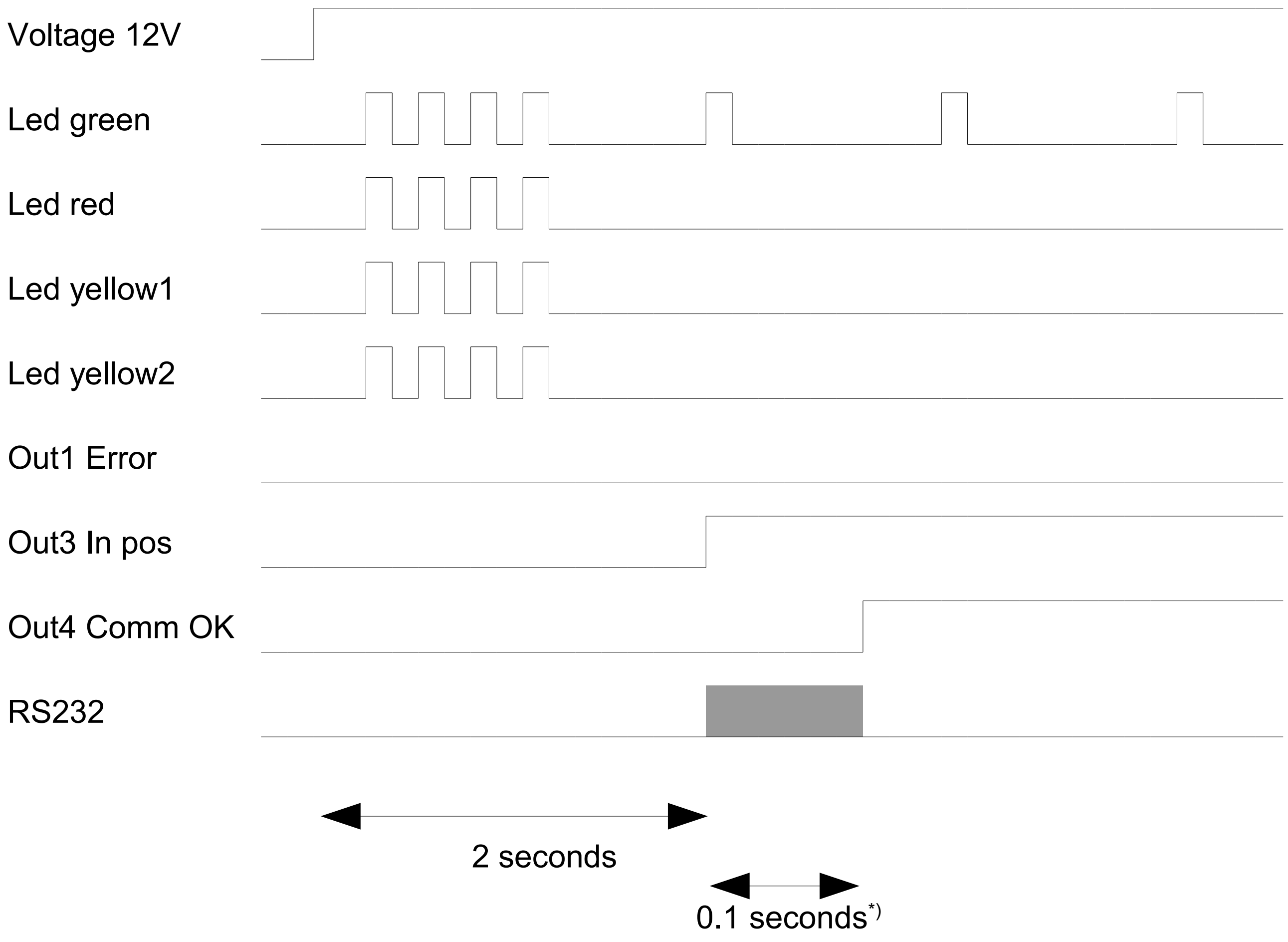
```
# [0x15] M T [0x00] [0x02] [5C]
```

5.2.2 PLC's hardware IO

It is possible to pre program the M101 or M102 with distances, with hardware inputs the movement can be realized. A PLC can give commands by setting inputs and check the result be reading the outputs of the board.

6 Power-up

6.1 Sequence during power up



*) During this time the Board will send a serial message on RS232. The message (send in the configured baud-rate) is like:

```
    M101 version x.xx
```

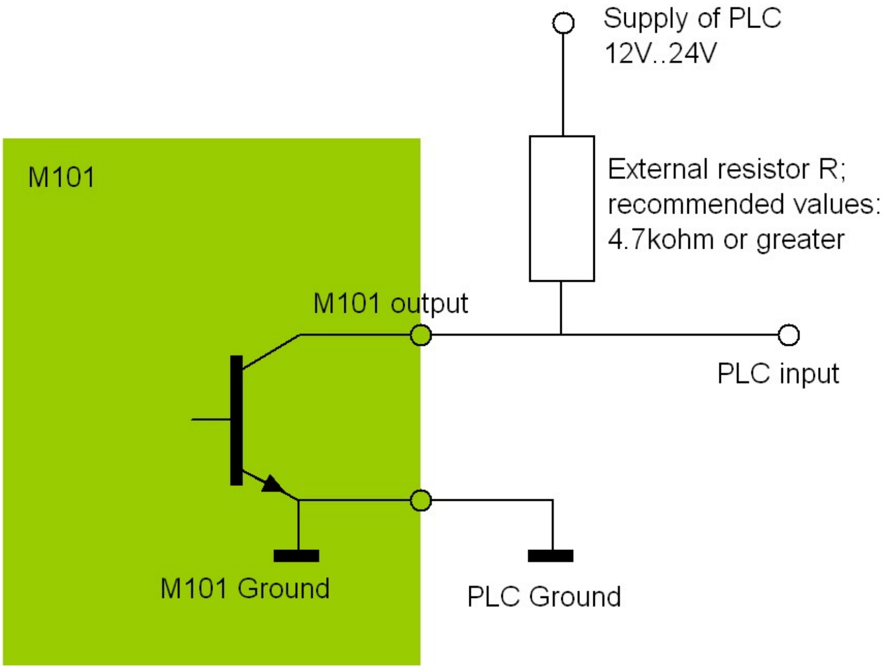
Now the board is ready to receive instructions.

7 Features

7.1 Hardware outputs

Three hardware Open Collector Output are assigned as “hard” outputs to signal the status of the board.

Underneath you will find the electrical circuit which is used to connect the output to the input of a PLC.



7.1.1 Error Output (Pin 2 of connector J10)

When the board faces a situation which is not within the parameters, the board will stop to operate. Depending the level of the problem the Error can be reset by an instruction, QE (quit error) or a Hard Reset will be needed. If the QE is not sufficient to reset the Error, the instruction RS (software reset) might help. After this instruction the board must be initialized from the beginning, the QE will not require the initialization.

If an error occurs, the Pin 11 will go High to signal the problem. What the problem is, cannot be deducted from this pin. An instruction to ask which error occurred is XE, the M101 or M102 will respond with the error code.

7.1.2 In position or motion finished Output (Pin 4 of connector J10)

This output is normally open and pulls the signal to ground, when the command is given to move the motor, this output will go high until the the board finished the operation. Please note that this does not mean that the motor arrived to its position. Without an encoder the movement cannot be checked by the board. Several parameters will be controlled and checked which might lead to an Error message, but if all is within the settings the Error will not be set.

Often it is not needed to have an encoder to check the movements, but it might be needed in certain cases.

7.1.3 Message received, Communication OK (Pin 5 of connector J10)

Pin 5 of connector 12 pin will be cleared, signal low, when a message is incoming, When the message arrived and understood the signal will be set high.

The ASCII communication does not have a check-sum control to verify the message. This is possible with the Binary type of communication.

With the experience we have up till today, it looks that the communication ASCII is stable and the risk of wrong commands or instructions seems to be low. It must be said that this might depend a lot of the ambient where the M101 or M102 is operating. If this is a noisy environment the risk of miss communication might be present.

7.1.4 4 leds status indicators

The M101 and M102 has 4 leds which give a status indication. The 4 leds are: green, red, yellow, 2 Yellow. 1

Underneath you will find the explanation of the codes:

The green led indicates the state of the board:

Green Led	Blinks 1 time	Type of Motor not defined
	Blinks 2 times	Motor defined, not powered
	Blinks 3 times	Motor is powered and controlled

In normal operation the red led is off:

Red Led OFF	Yellow Led 1	Communication OK
	Yellow Led 2	In Position

In error state the red led in on:

Red Led ON	Blinks 1 time	Error class 100
	Blinks 2 times	Error class 200
	Blinks 3 times	Error class 300
	Yellow Led 1	Gives the decimal value of the error
	Yellow Led 2	Gives the unit value of the error

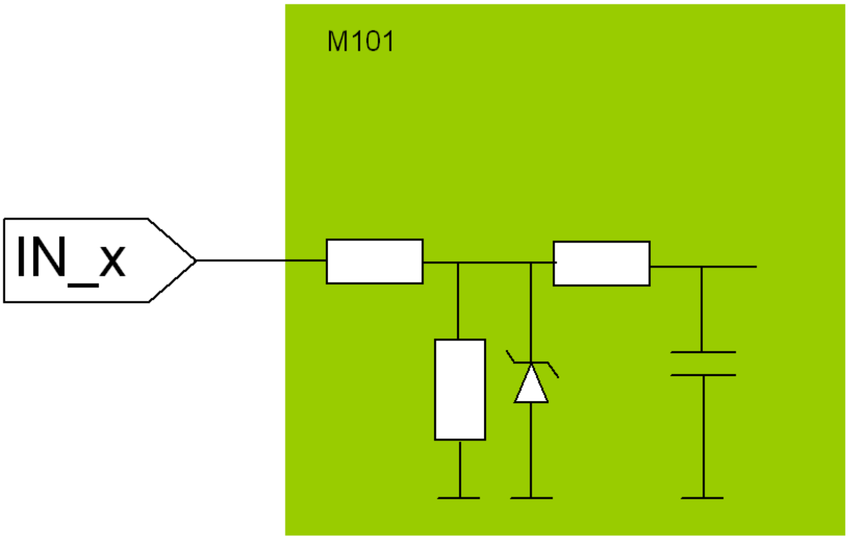
The board is in the boot-loader:

All Led	Continuous blink	board is in the boot-loader

Special error codes are if 4 leds blink together, a software error occurred which blocks the motor and the communication

All Led	1x and 1 second OFF	Trap reset
All Led	2x and 1 second OFF	Watchdog
All Led	3x and 1 second OFF	Brown out (drop in 12Vdc power)
All Led	4x and 1 second OFF	Configuration missing
All Led	5x and 1 second OFF	Illegal Opcode or uninit W

7.2 Hardware inputs



8 Jumpers

Underneath the commands are listed with their meaning. If one M101 or M102 is used, no address will be needed. The commands will be send with preceding the “!” character, like “!MT”.

If more than one M101 or M102 are connected together, the first board will have the RS232 connection, the other boards will be linked/connected with RJ45 cables between them. To communicate with those boards the CAN protocol will be used. The 5 first jumpers will be set to define the CAN addresses.

1	2	3	4	5	Address (hex)	Address (dec)
●	-	-	-	-	01	01
-	●	-	-	-	02	02
●	●	-	-	-	03	03
-	-	●	-	-	04	04
●	-	●	-	-	05	05
-	●	●	-	-	06	06
●	●	●	-	-	07	07
-	-	-	●	-	08	08
●	-	-	●	-	09	09
-	●	-	●	-	0A	10
●	●	-	●	-	0B	11
-	-	●	●	-	0C	12
●	-	●	●	-	0D	13
-	●	●	●	-	0E	14
●	●	●	●	-	0F	15
-	-	-	-	●	10	16
●	-	-	-	●	11	17
-	●	-	-	●	12	18
●	●	-	-	●	13	19
-	-	●	-	●	14	20
●	-	●	-	●	15	21
-	●	●	-	●	16	22
●	●	●	-	●	17	23
-	-	-	●	●	18	24
●	-	-	●	●	19	25

-	●	-	●	●	1A	26
●	●	-	●	●	1B	27
-	-	●	●	●	1C	28
●	-	●	●	●	1D	29
-	●	●	●	●	1E	30
●	●	●	●	●	1F	31

Jumper 6 is reserved.

Jumper 7 is for the CAN baud-rate Open (no jumper) = 125kbit
Set (with jumper) = 1Mbit

Jumper 8 is for the RS232 baud-rate Open (no jumper) = 19200
Set (with jumper) = 115200

9 Direction of movement

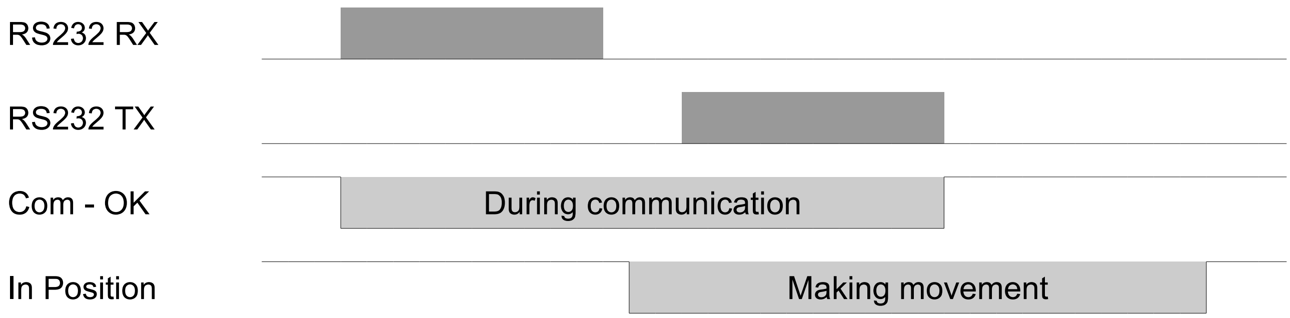
The motor can be moved in 2 directions, positive or negative. The distance or number of rotations is calculated in a Hexadecimal value. To go in the other direction the two's complement value will be used. This sounds more difficult than it is, on the site of SwissELME you will find an Excel tool to help you calculate the distance in both directions.

10 Commands or instructions

10.1 General information about the communication RS232

As example the M101 or M102 will receive a serial instruction on RS232 to make a movement. As the M101 or M102 receive the first serial byte the Com_OK signal will go to low. This signal remain low until the communication is terminated.

As the receive instruction is completely decoded, the movement will start immediately. During the movement the IP_Position signal goes to low.



The M101 or M102 echos normally the instructions to the user. This gives a verification of the received messages and helps to understand the status of the board. What kind of information is given or can be asked is shown underneath.

If there is no answer on the RS232 TX, a problem appears during the receive communication and the COM-OK will remain low. In this case is necessary to write some [cr] characters until the COM_OK signal goes high.

Before send the next instruction wait until the M101 or M102 did answer completely and the COM_IN signal is high. If the function “ANSWER DISALBED” is choose, there is no answer and the COM_OK signal goes high as soon as the receive instruction is understood.

10.1.1 After messages are sent

As mentioned will the M101 or M102 echo the received message or instruction. Instruction will be echoed in the official format. For example: !MT2 (Motor type 2, stepper) will be echoed as “!MT0002”.

10.2 Motor specific commands

IMPORTANT: all data are given in HEX format.

10.2.1 MT - Motor type

Motor type is the first motor specific command you have to define, else the other parameters will not be accepted.

Syntax1	`!MT`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`MT`*`xxxx`*[cr] (*`aa`* is the board address)
Definition	1 = DC motor 2 = Stepper motor 2 phases 3 = Stepper motor 3 phases 4 = Linear motor 2 phases 5 = Linear motor 3 phases 6 = Brush-Less DC motor, voltage regulation mode 7 = Linear motor 3 phases with 2 hall sensors (sin, cos) 8 = Brush-Less DC motor, current regulation mode ENCODER, if a digital encoder is attached, add Hex 0x100, e.g. MT0102 = 2 phase stepper motor with encoder.
Default value	0
Validity	All motor types

10.2.2 ES - Encoder Shift for stepper motor

Syntax1	`!ES`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`ES`*`xxxx`*[cr] (*`aa`* is the board address)
Definition	If an external encoder is used with the stepper motor to prevent lost of steps, this parameter defines the relation between step and number of pulses of the encoder, in 2^n. If the relation is 64, 0x0006 should be written. This command is available only in power off (`!PW0`).
Default value	0
Range	Between 0 and 8
Validity	Stepper motor with encoder

10.2.3 AC – Acceleration

Syntax1	`!AC`*xxxx*[cr]
Syntax2	`@`*aa*`AC`*xxxx*[cr] (*aa* is the board address)
Definition	Set the acceleration value
Default value	10
Range	Between 1 and 255
Validity	All motors

10.2.4 SP - Speed

Syntax1	`!SP`*xxxx*[cr]
Syntax2	`@`*aa*`SP`*xxxx*[cr] (*aa* is the board address)
Definition	Set the maximal speed value
Default value	100
Range	Between 1 and 32767
Validity	All motors

10.2.5 MC - Motor current / Max current

Syntax1	`!MC`*xxxx*[cr]	
Syntax2	`@`*aa*`MC`*xxxx*[cr] (*aa* is the board address)	
Definition stepper motor	The board will take this value as driver current for the stepper motor during a movement	
Definition other motors	The board will cut the current above this level, switch off the board and create an Error 0x212 ERROR_SW_Short_Circuit. Unit is 0.0125A.	
Default value	Stepper DC, BLDC Linear	64 (0.8A) 400 (5A) 80 (1A)
Range	Stepper DC, BLDC Linear	between 10 and 400 (600 for M102) between 10 and 450 (675 for M102) between 10 and 450 (700 for M102)
Validity	All motors	

10.2.6 SC - Standby current for stepper

Syntax1	`!SC`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`SC`*`xxxx`*[cr] (*`aa`* is the board address)
Definition	The standby current is the current which will be used to keep the stepper motor in position when there is no movement. The current should be chosen high enough to keep the motor still, but low enough to keep the motor from heating up.
Default value	32 (0.48A)
Range	Between 10 and 400 (600 for M102)
Validity	Stepper motor

10.2.7 SI - Standby ↔ Motor current increment for stepper

Syntax1	`!SI`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`SI`*`xxxx`*[cr] (*`aa`* is the board address)
Definition	When the movement starts, the motor current will change from the “”standby current” to the “normal motor current”. The change is not instantaneous but is programmable. Example: the default value is 0.0125A/ms, this means that the current amplitude need 240ms to change from 1A to 4A. If the motor need a high acceleration, it is important to increase the current amplitude as fast as possible to have as soon as possible the maximal torque.
Default value	1 (0.0125A/ms = 12.5A/s)
Range	Between 1 and 100
Validity	Stepper motor

10.2.8 CT – Max Current Time

Syntax1	`!CTxxxx`[cr]
Syntax2	`@aaCTxxxx`[cr] (*aa* is the board address)
Definition	The board controls if the current exceed the max values of +/-5A (+/-7.5A for M102). If the current exceed the max current values during a period of “Max Current Time” the the board will generate an error. Sometimes BLDC motor does exceed the max current for a short time, and in this case it is possible to increase this time, so the board will be less susceptible.
Default value	40 (1ms)
Range	Between 10 (0.25ms) and 400 (10ms)
Validity	All motor

10.2.9 PS – Power safe timeout for Stepper

Syntax1	`!PSxxxx`[cr]
Syntax2	`@aaPSxxxx`[cr] (*aa* is the board address)
Definition	Is the time after the motor finish the movement before the motor goes in stand-by current. Unit is 1ms.
Default value	1000 (1s)
Range	Between 10 and 10000 (10ms to 10s)
Validity	Stepper motor

10.2.10 FW Following Error (Stepper motor only with encoder)

Syntax1	`!FWxxxxxxxx`[cr]
Syntax2	`@aaFWxxxxxxxx`[cr] (*aa* is the board address)
Definition	Set the distance which the motor might "compensate" later. If the motor has difficulties to start up or to finish the job it might miss some steps. These misses will be compensated to let the motor arrive at the requested position. If the motor has a problem with the setting, the motor will give a "following Error". Negative numbers are not allowed.
Default value	4000
Range	Between 0 and 16384, 0=function is disabled
Validity	All motors

10.2.11 KD Differential Control for position regulation

Syntax1	`!KD`*xxxx*[cr]
Syntax2	`@`*aa*`KD`*xxxx*[cr] (*aa* is the board address)
Definition	This is the kd parameter for the PID position regulator.
Default value	DC 64 BLDC 24 Linear 10
Range	DC between 0 and 4095 BLDC between 0 and 4095 Linear between 0 and 4095
Validity	DC, BLDC, LIN motors

10.2.12 KP Proportional Control for position regulation

Syntax1	`!KP`*xxxx*[cr]
Syntax2	`@`*aa*`KP`*xxxx*[cr] (*aa* is the board address)
Definition	This is the kp parameter for the PID position regulator.
Default value	DC 64 BLDC 16 Linear 64
Range	DC between 0 and 4095 BLDC between 0 and 4095 Linear between 0 and 4095
Validity	DC, BLDC, LIN motors

10.2.13 KI Integration Control for position regulation

Syntax1	`!KI`*xxxx*[cr]
Syntax2	`@`*aa*`KI`*xxxx*[cr] (*aa* is the board address)
Definition	This is the ki parameter for the PID position regulator.
Default value	DC 0 BLDC 0 Linear 0
Range	DC between 0 and 4095 BLDC between 0 and 4095 Linear between 0 and 4095
Validity	DC, BLDC, LIN motors

10.2.14 FF - feed forward control for friction

Syntax1	`!FF`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`FF`*`xxxx`*[cr] (*aa* is the board address)
Definition	To anticipate the control, a feed forward can be introduced to improve the movement. This parameter is used to compensate the friction, giving a constant voltage regarding of the direction of the movement.
Default value	0
Range	Between 0 and 4095
Validity	DC, BLDC, LIN motors

10.2.15 FS - feed forward control for speed.

Syntax1	`!FS`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`FS`*`xxxx`*[cr] (*aa* is the board address)
Definition	To anticipate the control, a feed forward can be introduced to improve the movement. This parameter is used to compensate the speed constant, giving out a voltage proportional to the speed.
Default value	0
Range	Between 0 and 4095
Validity	DC, BLDC, LIN motors

10.2.16 CP - Proportional Control for current regulation

Syntax1	`!CP`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`CP`*`xxxx`*[cr] (*aa* is the board address)
Definition	This is the kp parameter for the PI current regulator.
Default value	32
Range	Between 0 and 4095
Validity	Stepper motors, LIN motors, BLDC in current regulation mode

10.2.17 CI - Integration Control for current regulation

Syntax1	`!CIxxxx`[cr]
Syntax2	`@aaCIxxxx`[cr] (*aa* is the board address)
Definition	This is the ki parameter for the PI current regulator.
Default value	0
Range	Between 0 and 4095
Validity	Stepper motors, LIN motors, BLDC in current regulation mode

10.2.18 CQ - Proportional Control for current regulation for slow speed

Syntax1	`!CQxxxx`[cr]
Syntax2	`@aaCQxxxx`[cr] (*aa* is the board address)
Definition	This is the kp parameter for the PI current regulator, in case of slow speed. Slow speed is the speed lower than the "Slow speed for proportional Control" (see CL comand). If CQ is zero, the function is disabled.
Default value	0
Range	Between 0 and 4095
Validity	Stepper motors

10.2.19 CL – Slow speed for proportional Control for current regulation

Syntax1	`!CLxxxx`[cr]
Syntax2	`@aaCLxxxx`[cr] (*aa* is the board address)
Definition	This is the "Slow speed for proportional Control for current regulation". This parameter works together with the command CQ. If CQ is zero, the function is disabled.
Default value	1000
Range	Between 1 and 32767
Validity	Stepper motors

10.2.20 FD – Freeze reg delta

Syntax1	`!FDxxxx`[cr]
Syntax2	`@aaFDxxxx`[cr] (*aa* is the board address)
Definition	When the motor is not moving, the position regulator try to stabilize the position. If the position remain in the +/-"*Freeze reg delta*" during the "*Freeze reg time*", then the regulator well be inactive freezing his output.
Default value	DC 2 BLDC 2 Linear 12
Range	Between 1 and 4095
Validity	DC, BLDC, LIN motors

10.2.21 FT – Freeze reg time

Syntax1	`!FTxxxx`[cr]
Syntax2	`@aaFTxxxx`[cr] (*aa* is the board address)
Definition	When the motor is not moving, the position regulator try to stabilize the position. If the position remain in the +/-"*Freeze reg delta*" during the "*Freeze reg time*", then the regulator well be inactive freezing his output.
Default value	1000
Range	Between 0 and 4095, 0=function is disabled
Validity	DC, BLDC, LIN motors

10.2.22 PS – Power safe timeout for DC, LIN, BLDC

Syntax1	`!PSxxxx`[cr]
Syntax2	`@aaPSxxxx`[cr] (*aa* is the board address)
Definition	Is the time after the motor finish the movement before the motor parameter goes in stand-by mode (the derivative regulation the the PID will become more silent). Unit is 1ms.
Default value	1000 (1s)
Range	Between 10 and 10000 (10ms to 10s)
Validity	DC, BLDC, LIN motors

10.2.23 SS - Set speed shift

Syntax1	`!SS`*xxxx*[cr]
Syntax2	`@`*aa*`SS`*xxxx*[cr] (*aa* is the board address)
Definition	Defines the accuracy and the range for position, speed and acceleration for the motor. See next Chapter for detailed information.
Default value	STEPPER 7 DC 8 BLDC 7 Linear 8
Range	Between 2 and 10
Validity	All motors

10.2.24 AS - Set acceleration shift

Syntax1	`!AS`*xxxx*[cr]
Syntax2	`@`*aa*`AS`*xxxx*[cr] (*aa* is the board address)
Definition	Defines the accuracy and the range for position, speed and acceleration for the motor. See next Chapter for detailed information.
Default value	STEPPER 7 DC 2 BLDC 2 Linear 2
Range	Between 2 and 10
Validity	All motors

10.2.25 WD - Set “In position” delta position

Syntax1	`!WD`*xxxxxxxx*[cr]
Syntax2	`@`*aa*`WD`*xxxxxxxx*[cr] (*aa* is the board address)
Definition	Defines the delta position for the “In position” output signal. The “In position” output signal will be active, if the destination motor position correspond to the wanted position during at least the “In position time” (see command WT).
Default value	100
Range	Between 0x0001 and 0x0FFF
Validity	All motors

10.2.26 WT - Set “In position” time

Syntax1	`!WT`*xxxx*[cr]
Syntax2	`@`*aa*`WT`*xxxx*[cr] (*aa* is the board address)
Definition	Defines the time for the “In position” output signal. The “In position” output signal will be active, if the destination motor position correspond to the wanted position during at least the “In position time” (see command WD).
Default value	1000 (correspond to 1s)
Range	Between 0x0001 and 0x0FFF
Validity	All motors

10.3 Limit switch positions and sensors

10.3.1 LS - Hardware Limit Switch Mode

Syntax1	`!LS`*xxxx*[cr]
Syntax2	`@`*aa*`LS`*xxxx*[cr] (*aa* is the board address)
Definition	The board can be fitted with 2 hardware limit switches. The sensors can be powered with the on board present 12Vdc, which is enough to power the regular sensors. The M101 or M102 can be operated with hardware limit switches. This can be Micro Switches or Sensors (PNP or NPN). These switches can be used as initial zero search or as hardware security that the movement will not go beyond the mechanical limits. There are 4 modes: • 0 No limit switches present • 1 Negative movement only 1 switch • 2 Positive movement only 1 switch • 3 Both switches present
Default value	0
Range	Between 0 and 3
Validity	All motors

10.3.2 IA – Input number for negative switch

Syntax1	`!IA`*xxxx*[cr]
Syntax2	`@`*aa*`IA`*xxxx*[cr] (*aa* is the board address)
Definition	Defines the input number in witch the sensor is connected. 0 = no sensor for negative switch defined 1..6 = sensor on one of the six inputs **NOTE:** if init mode is set to 1 (`IM` command) and if the negative switch is not defined, then it is not possible to do the initialization (`II` command) in the negative direction (`ID` command). **NOTE:** do not use same inputs on different commands IA, IB, HZ, GA, GB
Default value	0
Range	Between 0 and 6
Validity	All motors

10.3.3 IB – Input number for positive switch

Syntax1	`!IB`*xxxx*[cr]
Syntax2	`@`*aa*`IB`*xxxx*[cr] (*aa* is the board address)
Definition	Defines the input number in witch the sensor is connected. 0 = no sensor for positive switch defined 1..6 = sensor on one of the six inputs **NOTE:** if init mode is set to 1 (`IM` command) and if the positive switch is not defined, then it is not possible to do the initialization (`II` command) in the positive direction (`ID` command). **NOTE:** do not use same inputs on different commands IA, IB, HZ, GA, GB
Default value	0
Range	Between 0 and 6
Validity	All motors

10.3.4 LM - Software Limit Switch Minimum

Syntax1	`!LM`*`xxxxxxxxx`*[cr]
Syntax2	`@`*`aa`*`LM`*`xxxxxxxxx`*[cr] (*aa* is the board address)
Definition	It is possible to define the position of the limit switches in a software way. The limits will be defined in distance, when the movement goes beyond, the board will block the movement and generate an error. This command sets the software limit of the movement in one direction, if the counter of steps exceeds this value the board will block and generate a software limit error. For negative numbers use twos complement. If LM and LN are set to zero this function is disabled.
Default value	0
Range	-2130706432 to 2130706432
Validity	All motors

10.3.5 LN - Software Limit Switch Maximum

Syntax1	`!LN`*`xxxxxxxxx`*[cr]
Syntax2	`@`*`aa`*`LN`*`xxxxxxxxx`*[cr] (*aa* is the board address)
Definition	The same as the minimum but for the opposite side. For negative numbers use twos complement. If LM and LN are set to zero this function is disabled.
Default value	0
Range	Between -2130706432 to 2130706432
Validity	All motors

10.3.6 LP - Sensor polarity for Software Limit Switch Min/Max

Syntax1	`!LP`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`LP`*`xxxx`*[cr] (*aa* is the board address)
Definition	PNP or NPN sensors can be used, depending the parameter setting. Default is the PNP sensor set. 0 = Defines the sensor or switch with active high (24V) 1 = Defines the sensor or switch with active low (0V)
Default value	1
Range	0 or 1
Validity	All motors

10.4 Initialization commands (axis reference) and movement

10.4.1 IM - Init mode

Syntax1	`!IM`*xxxx*[cr]
Syntax2	`@`*aa*`IM`*xxxx*[cr] (*aa* is the board address)
Definition	This command will indicate if switches or the increase in current when the movement blocks on the mechanics method is used. 0001 = sensors or switches are used to make the initialization of the motor 0002 = increase of the following error (FW) is used to make the initialization of the motor 0003 = increase in current when the movement blocks on the mechanics is used to make the initialization of the motor. Use parameter IX to set the level of current 0004 = increase in voltage (PWM regulation) when the movement blocks on the mechanics is used to make the initialization of the motor. Use parameter IX to set the level of voltage
Default value	1
Range	Between 1 and 4
Validity	1 = All motors 2 = Stepper motor with encoder 3 = Linear motor 4 = BLDC motor or DC motor

10.4.2 IS - Init max stroke

Syntax1	`!IS`*xxxxxxxx*[cr]
Syntax2	`@`*aa*`IS`*xxxx*[cr] (*aa* is the board address)
Definition	Define the maximal stroke allowed for the initialization. Writing zero this option is disabled. For negative numbers use twos complement.
Default value	0
Range	Between -2130706432 to 2130706432
Validity	All motors

10.4.3 IW - Go to position after initialization

Syntax1	`!IW`*xxxxxxxx*[cr]
Syntax2	`@`*aa*`IW`*xxxxxxxx*[cr] (*aa* is the board address)
Definition	Defines the position the motor have to move to after a successful initialization. For negative numbers use twos complement.
Default value	0
Range	Between -2130706432 to 2130706432
Validity	All motors

10.4.4 IH - Fast speed during initialization

Syntax1	`!IH`*xxxx*[cr]
Syntax2	`@`*aa*`IH`*xxxx*[cr] (*aa* is the board address)
Definition	Defines the speed for the initialization of the motor
Default value	100
Range	Between 1 and 32767
Validity	All motors

10.4.5 IL - Slow speed during initialization

Syntax1	`!IL`*xxxx*[cr]
Syntax2	`@`*aa*`IL`*xxxx*[cr] (*aa* is the board address)
Definition	Defines the speed for the initialization of the motor by the fine sensor detection.
Default value	10
Range	Between 1 and 32767
Validity	All motors

10.4.6 ID - Init direction

Syntax1	`!ID`*xxxx*[cr]
Syntax2	`@`*aa*`ID`*xxxx*[cr] (*aa* is the board address)
Definition	65535 (0xFFFF = -1) = Defines the initialization direction as negative 1 (+1) = Defines the initialization direction as positive
Default value	65535 (0xFFFF = -1)
Validity	All motors

10.4.7 IX - Init of current or voltage level

Syntax1	`!IX`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`IX`*`xxxx`*[cr] (*aa* is the board address)
Definition	This parameter is used only in *"Init mode"* 2, 3 or 4. For every motor this parameter have a specific significance: **LIN**: defines the max current level (1=0.0125A, 80=1A). **BLDC**, **DC**: defined the max output regulation value (1=0.2%, 250=50%).
Default value	10
Range	Between 1 and 256
Validity	LIN, BLDC, DC motor, only in *"Init mode"* 2, 3 or 4

10.4.8 II - Execute initialization

Syntax1	`!II`[cr]
Syntax2	`@`*`aa`*`II`[cr] (*aa* is the board address)
Definition	Execute the initialization
Validity	All motors

10.5 Movement command

10.5.1 SD - Set DELTA position

Syntax1	`!SD`*`xxxxxxxx`*[cr]
Syntax2	`@`*`aa`*`SD`*`xxxxxxxx`*[cr] (*aa* is the board address)
Definition	The *Delta positioning* is useful when a distance must be repeated. One time set, with each Go Delta instruction, this distance will be covered. The answer contains the absolute destination position.
Default value	0
Range	Between -2130706432 to 2130706432
Validity	All motors

10.5.2 SA - Set ABSOLUTE position

Syntax1	`!SD`*xxxxxxxx*[cr]
Syntax2	`@`*aa*`SD`*xxxxxxxx*[cr] (*aa* is the board address)
Definition	With the Absolute positioning, a position will be set which the controller will remember, when the Go Absolute instruction is given the motor will go to that position. The default value is 0, but its is better to define it, if used. If the GA (Go Absolute) instruction is given in sequence, the motor will not move after the first command as the motor is in position. The Go Delta instruction, as mentioned above, will move with every instruction the defined distance.
Default value	0
Range	Between -2130706432 to 2130706432
Validity	All motors

10.5.3 SH - Set Home Position

Syntax1	`!SH`*xxxxxxxx*[cr]
Syntax2	`@`*aa*`SH`*xxxxxxxx*[cr] (*aa* is the board address)
Definition	The home position gives the virtual zero of the system. It avoids also calculation errors. If a motor has to run a high number of rotations, the system can not handle the number of steps. The home position resets the counter. If a motor rotates in cycles, the steps will be accumulated. Resetting the counter before each movement will avoid the counter overflow
Range	Between -2130706432 to 2130706432
Validity	All motors

10.5.4 GD - Go Delta

Syntax1	`!GD`[cr]
Syntax2	`@`*aa*`GD`[cr] (*aa* is the board address)
Answer	The answer has a 8 byte data length containing the absolute destination position
Definition	Instruction to execute the previous set Delta distance
Validity	All motors

10.5.5 GA - Go Absolute

Syntax1	`!GA`[cr]
Syntax2	`@`*aa*`GA`[cr] (*aa* is the board address)
Answer	The answer has a 8 byte data length containing the absolute destination position
Definition	Command to move to the absolute position
Validity	All motors

10.5.6 GH - Go Home

Syntax1	`!GH`[cr]
Syntax2	`@`*aa*`GH`[cr] (*aa* is the board address)
Answer	The answer has a 8 byte data length containing the absolute destination position
Definition	Command to move to the home position (0)
Validity	All motors

10.5.7 GP - Go to Position

Syntax1	`!GP`*xxxxxxxx*[cr]
Syntax2	`@`*aa*`GP`*xxxxxxxx*[cr] (*aa* is the board address)
Definition	This is an interesting command, it will define the position where to go to and will immediately execute the command. This saves communication time.
Range	Between -2130706432 to 2130706432
Validity	All motors

10.5.8 ST - Stop

Syntax1	`!ST`[cr]
Syntax2	`@`*aa*`ST`[cr] (*aa* is the board address)
Definition	The board has the feature of doing a zero search with one of the limit switches. If the user wants to have a routine which does it in a different way, it is possible. Starting the movement and stop it at a certain point. The Stop command will hold the movement. It can also be used in other situations
Validity	All motors

10.6 Analog positioning mode

10.6.1 AZ – Analog positioning mode (M102 and above)

Syntax1	`!AZxxxx`[cr]
Syntax2	`@aaAZxxxx`[cr] (*aa* is the board address)
Answer	The answer has a 4 byte data length
Definition	Setting 1 the position of the motor is driven by the external analog input signal. Analog signal has range from 0V to 10V. Setting 2 the maximal speed of the motor is set by the external analog input signal. Analog signal has range from 0V to 10V.
Range	0, 1, 2
Default value	0
Validity	All motors

10.6.2 AM – Analog positioning at 0V (M102 and above)

Syntax1	`!AMxxxxxxxx`[cr]
Syntax2	`@aaAMxxxxxxxx`[cr] (*aa* is the board address)
Answer	The answer has a 8 byte data length
Definition	This command defines the position corresponding to the analog input when it is 0V.
Range	Between -2130706432 to 2130706432
Default value	0
Validity	All motors

10.6.3 AN – Analog positioning at 10V (M102 and above)

Syntax1	`!ANxxxxxxxx`[cr]
Syntax2	`@aaANxxxxxxxx`[cr] (*aa* is the board address)
Answer	The answer has a 8 byte data length
Definition	This command defines the position corresponding to the analog input when it is 10V.
Range	Between -2130706432 to 2130706432
Default value	10000
Validity	All motors

10.6.4 AF – Analog filter (M102 and above)

Syntax1	`!AFxxxx`[cr]
Syntax2	`@aaAFxxxx`[cr] (*aa* is the board address)
Answer	The answer has a 2 byte data length
Definition	This in aa hysteresis filter of the analog input signal. The analog input is converted as a 10 bit value, the resolution of the analog signal is 10mV. If you set !AF4, the analog input will be filtered with an hysteresis of +/- 40mV. This filter allow you to reduce the noise.
Range	Between 0 and 32
Default value	0
Validity	All motors

10.7 Status command

10.7.1 XW - Get W Position

Syntax1	`!XP`[cr]
Syntax2	`@aaXP`[cr] (*aa* is the board address)
Answer	The answer has a 8 byte data length
Definition	This instruction will ask the board what it has as wanted position.
Validity	All motors

10.7.2 XA - Get A Position

Syntax1	`!XA`[cr]
Syntax2	`@aaXA`[cr] (*aa* is the board address)
Answer	The answer has a 8 byte data length
Definition	This instruction will ask for the actual real position measured if available with an external encoder. If external encoder is not available, then is used the hall sensor position.
Validity	All motors

10.7.3 XE - Get Error messages

Syntax1	`!XE`[cr]
Syntax2	`@aaXE`[cr] (*aa* is the board address)
Answer	The answer has a 4 byte data length
Definition	If the board is blocked due to whatever reason, with this command the board can be asked what caused the problem.
Validity	All motors

10.7.4 XS - Get supply voltage

Syntax1	`!XS`[cr]
Syntax2	`@aaXS`[cr] (*aa* is the board address)
Answer	The answer has a 4 byte data length
Definition	With this command is possible to get the actual voltage power supply
Validity	All motors

10.7.5 XF - Get finish movement

Syntax1	`!XF`[cr]
Definition	The motor is doing a movement and you want to know asap when the movement is finished. Send the command !XF during the movement and when you get the answer the movement is finished.
Validity	All motors, only true RS232

10.7.6 TH - Get actual time (hours)

Syntax1	`!TH`[cr]
Syntax2	`@aaTHxxxxxxxx`[cr] (*aa* is the board address)
Definition	This command return the total time in hours since the board was powered-on or resetted. The unit is 1 hour.
Validity	All motors

10.7.7 TL - Get actual time (minutes, seconds, milliseconds)

Syntax1	`!TL`[cr]
Syntax2	`@`*aa*`TLxxxxxxxx`[cr] (*aa* is the board address)
Definition	This command return the total time since the board was powered-on or resetted. The unit is 1ms. 3599999 correspond to 59 minutes, 59 seconds and 999 milliseconds.
Validity	All motors

10.8 Information commands

10.8.1 XV - Get Firmware Version

Syntax1	`!XV`[cr]
Syntax2	`@`*aa*`XV`[cr] (*aa* is the board address)
Answer	The answer has a 4 byte data length
Definition	To get the current firmware version, the XV command will ask the board. When the Board is powered up (12Vdc) the board will echo the version also.
Validity	All motors

10.8.2 XN - Get Board Serial Number

Syntax1	`!XN`[cr]
Syntax2	`@`*aa*`XN`[cr] (*aa* is the board address)
Answer	The answer has a 8 byte data length
Definition	To get the current firmware version, the XV command will ask the board. When the Board is powered up (12Vdc) the board will echo the version also.
Validity	All motors

10.8.3 BT – Get Board Type

Syntax1	`!BT`[cr]
Syntax2	`@`*aa*`BT`[cr] (*aa* is the board address)
Answer	The answer has a 4 byte data length
Definition	Respond with the board type (value is to convert in decimal)
Validity	All motors

10.8.4 PT - Get Processor Type

Syntax1	`!PT`[cr]
Syntax2	`@aaPT`[cr] (`aa` is the board address)
Answer	The answer has a 4 byte data length
Definition	Respond with the processor type (value is to convert in decimal)
Validity	All motors

10.8.5 PR - Get Processor Revision

Syntax1	`!PR`[cr]
Syntax2	`@aaPR`[cr] (`aa` is the board address)
Answer	The answer has a 4 byte data length
Definition	Respond with the processor revision (value is to convert in decimal)
Validity	All motors

10.8.6 XC - Get Board Address

Syntax1	`!XN`[cr]
Answer	The answer has a 4 byte data length
Definition	Get the board address
Validity	All motors

10.9 General commands

10.9.1 AW – answer Mode

Syntax1	`!AW`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`AW`*`xxxx`*[cr] (*aa* is the board address)
Definition	To speed up the communication, it can be decided to suppress the answers of the M101 or M102. The status of the board can be checked by outputs and leds. It is obvious that the available information will be reduced. As default the M101 or M102 will reply to all commands with an Echo. This can be disabled with the AW command. This command will only be needed for the master board (the board which has the RS232 connection).
Default value	1
Range	0 (disable) or 1 (enable)
Validity	All motors

10.9.2 VM – set voltage min

Syntax1	`!VM`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`VM`*`xxxx`*[cr] (*aa* is the board address)
Definition	Defines the min supply voltage when the motor is enabled. If the the supply voltage is lower than this min voltage an error will be generated. Unit is 0.1V.
Default value	120 (12V)
Range	Between 100 (10V) and 450 (45V)
Validity	All motors

10.9.3 VN – set voltage max

Syntax1	`!VN`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`VN`*`xxxx`*[cr] (*aa* is the board address)
Definition	Defines the max supply voltage when the motor is enabled. If the the supply voltage is higher than this max voltage an error will be generated. Unit is 0.1V.
Default value	550 (55V) or 780 (78V) for high voltage board version
Range	Between 300 (30V) and 550 (55V) or 780 (78V) for high voltage board version
Validity	All motors

10.9.4 SR – soft reset

Syntax1	`!SR`[cr]
Syntax2	`@`*`aa`*`SR`[cr] (*aa* is the board address)
Answer	NO ANSWER. Attention: you will get the RS232 start-up message like "`M102 SW V2.41`"
Definition	With this command the board will be resetted. After this is necessary to initialize all parameters again.
Validity	All motors

10.9.5 QE – Quit error

Syntax1	`!QE`[cr]
Syntax2	`@`*`aa`*`QE`[cr] (*aa* is the board address)
Definition	If an error is active on the board, it is necessary to quit it.
Validity	All motors

10.9.6 PW - Power Enable

Syntax1	`!PW`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`PWxxxx`[cr] (*aa* is the board address)
Definition	Enables the motor and activate the current control. User this instruction after all parameters and motor type were downloaded.
Default value	0
Range	0 or 1
Validity	All motors

10.9.7 EM – Enable protocol error detection on serial communication

Syntax1	`!EM`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`EMxxxx`[cr] (*aa* is the board address)
Definition	Enables protocol error detection on serial communication. Normally if there is a protocol error during the serial communication the board will ignore the frame without generate an error. If this mode is enabled (EM0001), on any kind of detected error during the serial communication, the board ignore the frame and generate an error.
Default value	0
Range	0 or 1
Validity	All motors

10.10 Digital Input and Output configuration

10.10.1 IZ – Input number for init motor

Syntax1	`!IZ`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`IZ`*`xxxx`*[cr] (*aa* is the board address)
Definition	Define the input number to enable the initialization of the motor (correspond to the !II command).
Default value	0
Range	0=(disable), 1..6 (input number)
Validity	All motors

10.10.2 IY – Input number for power-on motor

Syntax1	`!IY`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`IY`*`xxxx`*[cr] (*aa* is the board address)
Definition	Define the input number to enable the power-on of the motor (correspond to the !PW command).
Default value	0
Range	0=(disable), 1..6 (input number)
Validity	All motors

10.10.3 IU – Input number for move in positive direction with constant speed

Syntax1	`!IU`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`IU`*`xxxx`*[cr] (*aa* is the board address)
Definition	Define the input number to move in positive direction with constant speed
.Default value	0
Range	0=(disable), 1..6 (input number)
Validity	All motors

10.10.4 IV – Input number for move in negative direction with constant speed

Syntax1	`!IU`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`IU`*`xxxx`*[cr] (*aa* is the board address)
Definition	Define the input number to move in negative direction with constant speed
.Default value	0
Range	0=(disable), 1..6 (input number)
Validity	All motors

10.10.5 TP – Output period (on output 2)

Syntax1	`!TP`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`TP`*`xxxxxxxx`*[cr] (*aa* is the board address)
Definition	It is possible to use this electronic as master and use the output2 to drive a slave. The output2 will generate a signal following the motor position, with the period defined with TP. The unit is the same as the position.
Default value	0
Range	0 or 0x00000010 .. 0x00400000
Validity	All motors

10.11 Multi trigger

10.11.1 HZ Multi trigger mode

Syntax1	`!HZ`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`HZ`*`xxxx`*[cr] (*aa* is the board address)
Definition	**Mode 2 (!HZ0002) with 1 Trigger + 1 Input** • Input 1 is used as trigger • Input 2 is the first digital input The board is programmed that if input 2 is high the board will go to a certain position. If input 2 is low, the motor will go to another position. This means that there are 2 combinations possible and thus 2 positions programmable (*position0* to *position1* corresponding to the combinations 0-1; see instruction *H{A..P}* and *J{A..P}*) **Mode 3 (!HZ0003) with 1 Trigger + 2 Inputs** • Input 1 is used as trigger or enable • Input 2 is the first digital input

- Input 3 is the second digital input
  This means that there are 4 combinations possible and thus 4 positions programmable (*position0* to *position3* corresponding to the combinations 00-01-10-11; see instruction *H{A..P}* and *J{A..P}*)

Mode 4 (!HZ0004) with 1 Trigger + 3 Input

- Input 1 is used as trigger or enable
- Input 2 is the first digital input
- Input 3 is the second digital input
- input 4 is the third digital input
  This means that there are 8 combinations possible and thus 8 positions programmable (*position0* to *position7* corresponding to the combinations 000-001-010-011-100-101-110-111; see instruction *H{A..P}* and *J{A..P}*).

NOTE IMPORTANT for mode 2, 3 and 4: If the trigger is used, the Up-going flank of the trigger input will start the movement. If this movement is detected more times during the movement and the movement is set as a DELTA positioning, the distance will be accumulated! This might lead to unexpected behavior.

Mode 12 (!HZ0012)

- Input 1 is the first digital input
- Input 2 is the second digital input
  This means that there are 4 combinations possible and thus 3 positions programmable (*position1* to *position3* corresponding to the combinations 01-10-11; see instruction *H{A..P}* and *J{A..P}*).

Mode 13 (!HZ0013)

- Input 1 is the first digital input
- Input 2 is the second input
- Input 3 is the third input
  This means that there are 8 combinations possible and thus 7 positions programmable (*position1* to *position7* corresponding to the combinations 001-010-011-100-101-110-111; see instruction *H{A..P}* and *J{A..P}*).

Mode 14 (!HZ0014)

- Input 1 is the first digital input
- Input 2 is the second digital input
- Input 3 is the third digital input
- input 4 is the fourth digital input
  This means that there are 16 combinations possible and thus 15 positions programmable (*position1* to *position15* corresponding to the

	combinations 0001-0010-0011-0100-0101-0110-0111-1000-1001-1010-1011-1100-1101-1110-1111; see instruction *H{A..P}* and *J{A..P}*). ***NOTE IMPORTANT for mode 12, 13 and 14:*** In this case the M101 or M102 will decide (with the JZ parameter) when the input status is stable and will execute the positioning. The *Position0* is disabled: when all inputs are low then the state is in idle. In mode 2, 3 and 4 is the trigger input which will give the OK to execute the movement. **NOTE:** do not use same inputs o different commands IA, IB, HZ, GA, GB
Default value	0
Range	0, 2, 3, 4, 12, 13 or 14
Validity	All motors

10.11.2 JZ Multi trigger stability

Syntax1	`!JZ`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`JZ`*`xxxx`*[cr] (*`aa`* is the board address)
Definition	The inputs are scanned with a frequency of 10kHz. To be sure no spikes or disturbances trigger the positioning, with the JZ parameter the number of scans can be set that the inputs must be stable. If the inputs are different in that period, the count will restart. This is for the Trigger mode as well as the multi input without trigger valid. Unit correspond to 25µs.
Default value	1 (25µs)
Range	Between 0 (0ms) and 10000 (250ms)
Validity	All motors

10.11.3 H{A..P} Multi trigger values (position, speed, etc.)

Syntax1	`!H{A..P}xxxxxxxx`[cr]
Syntax2	`@aaH{A..P}xxxxxxxx`[cr] (*aa* is the board address)
Definition	!H?xxxxxxxx[cr] ? = A to P **With Multitrigger Mode 2, 3 and 4** !HA00000000 --> position0 (0000) !HB00001000 --> position1 (0001) !HC00005000 --> position2 (0010) ….... !HP0000A000 --> position15 (1111) **With Multitrigger Mode 12, 13 and 14** !HA00000000 --> position0 (0001) !HB00001000 --> position1 (0010) !HC00005000 --> position2 (0011) ….... !HO0000A000 --> position14 (1111)
Default value	0
Range	Between -2130706432 to 2130706432
Validity	All motors

10.11.4 J{A..P} Multi trigger movement types

Syntax1	`!J{A..P}`*xxxxxxxx*[cr]
Syntax2	`@`*aa*`J{A..P}`*xxxxxxxx*[cr] (*aa* is the board address)
Definition	!J?xxxxxxxx[cr] ? = A to P **With Multitrigger Mode 2, 3 and 4** !JA00000000--> position0 (0000) !JB00000001--> position1 (0001) !JC00000001--> position2 (0010) ….... !JP00000001--> position15 (1111) **With Multitrigger Mode 12, 13 and 14** !JA00000000--> position0 (0001) !JB00000001--> position1 (0010) !JC00000001--> position2 (0011) ….... !JO00000001--> position15 (1111) 0: absolute movement: If the position is set as Absolute movement, the position will not change if the input is set more times. So if the Trigger or enable is set several times, the position of the motor will not change. 1: Delta movement. If the position is set as a Delta or incremental movement, the trigger can be set several times and the motor will move with every up-going flank the set distance. If the distance is long enough and the up going flank is detected several times within the movement, the movement will be repeated the same number of times!!! 2: Set maximal speed. It is possible, with Multi-trigger functionality, to modify the maximal speed.
Default value	0
Range	0, 1 or 2
Validity	All motors

10.11.5 IG Input triggers Go Absolute command (!GA)

Syntax1	`!IGxxxx`[cr]
Syntax2	`@aaIGxxxx`[cr] (*aa* is the board address)
Definition	Defines the input number is used to generate the GO Absolute instruction (see SA instruction): 0 = no input defined for this function 1..6 = input defined on one of the six inputs for this function **NOTE:** do not use same inputs on different commands IA, IB, HZ, GA, GB
Default value	0
Range	Between 0 and 6
Validity	All motors

10.11.6 IF Input triggers Go Delta command (!GD)

Syntax1	`!IFxxxx`[cr]
Syntax2	`@aaIFxxxx`[cr] (*aa* is the board address)
Definition	Defines the input number is used to generate the GO Delta instruction (see SD instruction): 0 = no input defined for this function 1..6 = input defined on one of the six inputs for this function **NOTE:** do not use same inputs on different commands IA, IB, HZ, GA, GB
Default value	0
Range	Between 0 and 6
Validity	All motors

10.12 Hall sensors adjustment

10.12.1 NA, NB, NC – Gain adjustment Hall_A, Hall_B and Hall_C

Syntax1	`!NA`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`NA`*`xxxx`*[cr] (*aa* is the board address)
Definition	Amplify the Hall sensor signal.
Default value	0x1000 (correspond to 100%)
Range	0x0800 .. 0x2000 (50% .. 200%)
Validity	LIN+BLDC

10.12.2 ND, NE, NF – Offset adjustment Hall_A, Hall_B and Hall_C

Syntax1	`!ND`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`ND`*`xxxx`*[cr] (*aa* is the board address)
Definition	Add an offset to the Hall sensor signal. 1 correspond to 5mV of the sensor signal.
Default value	0
Range	-0x0100 .. +0x0100 (0xFF00 .. 0x0100) ()
Validity	LIN+BLDC

10.12.3 NG – Angle offset for commutation table

Syntax1	`!NG`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`NG`*`xxxx`*[cr] (*aa* is the board address)
Definition	Do not modify this parameter, only for advanced users
Default value	Depend on motor type
Range	-0x1000 .. 0x1000
Validity	LIN+BLDC

10.12.4 NH – Speed-advance for commutation table

Syntax1	`!NH`*`xxxx`*[cr]
Syntax2	`@`*`aa`*`NH`*`xxxx`*[cr] (*aa* is the board address)
Definition	Do not modify this parameter, only for advanced users
Default value	Depend on motor type
Range	0x0000 .. 0x4000
Validity	LIN+BLDC

10.12.5 NZ – Start hall self-calibration

Syntax1	`!NZ`[cr]
Syntax2	`@`*`aa`*`NZ`[cr] (*`aa`* is the board address)
Definition	Start the self calibration of all “gain” and “offset” adjustment for the Hall sensors. After this command, during 5 seconds, move slowly the motor forward and backward. The processor will sample the Hall sensor values and will automatically calculate the “gain” and “offset” adjustment for the Hall sensors.
Validity	LIN+BLDC

10.13 Save parameters in EEPROM commands

10.13.1 EW – Writes all parameters in EEPROM

Syntax1	`!EW`[cr]
Syntax2	`@`*aa*`EW`[cr] (*aa* is the board address)
Definition	It is possible to save all parameters to the EEPROM, so in the next start it is no more needed to set all parameters. The stored parameters are: Motor_Type, Encoder_type Init_mode, Init_X_level Init_high_speed, Init_low_speed Init_direction, Init_sensor_polarity Init_max_stroke, Init_position w_acc_max, w_speed_max kd, kp, ki, curr_kp, curr_ki FF_friction, FF_speed Freeze_reg_time, Freeze_reg_delta following_pos_max Soft_regulation_time current_max, PP_current_stby HW_limit_switch_mode soft_limit_switch_min, soft_limit_switch_max Input_nr_limit_switch_pos, Input_nr_limit_switch_neg Voltage_min, Voltage_max Input_nr_for_go_abspos, Input_nr_for_go_deltapos pos_shift, enc_shift Output_period, PP_current_incr curr_kp_low_speed, PP_low_speed_level High_speed_Inputs Input_nr_for_power_enable, Input_nr_for_init_motor IN_position_delta_time, IN_position_delta_pos overcurrent_time **Only in M102 board and above:** MT_mode, MT_stability, MT_database analog_input_mode_enabled, analog_in_filter analog_in_min_pos, analog_in_max_pos Oscilloscope parameters: are not described in this manual.
Validity	All motors

10.13.2 ER – Reset parameters in EEPROM

Syntax1	`!ER`[cr]
Syntax2	`@`*`aa`*`ER`[cr] (*`aa`* is the board address)
Definition	It is possible to reset the parameters stored in the EEPROM.
Validity	All motors

11 The position-, speed- and acceleration range

11.1 Introduction

Each motor has a specific range for the position, speed and acceleration. In the following excel sheets are available a lot of information. Generic information as regulation frequency, PWM frequency and other are available for all motors. Only the cells marked with this color are modifiable 🟧.

11.2 Stepper

First is to define the steps for a revolution of the stepper motor.

11.3 Linear Motor

First is to define the magnetic pitch and the number of phases of the linear motor. Magnetic pitch correspond to a full magnetic period of the magnets. Linear motor can be defined as 2 or 3 phases.

11.4 DC Motor

First is to define the steps for a revolution the the encoder.

11.5 BLDC Motor

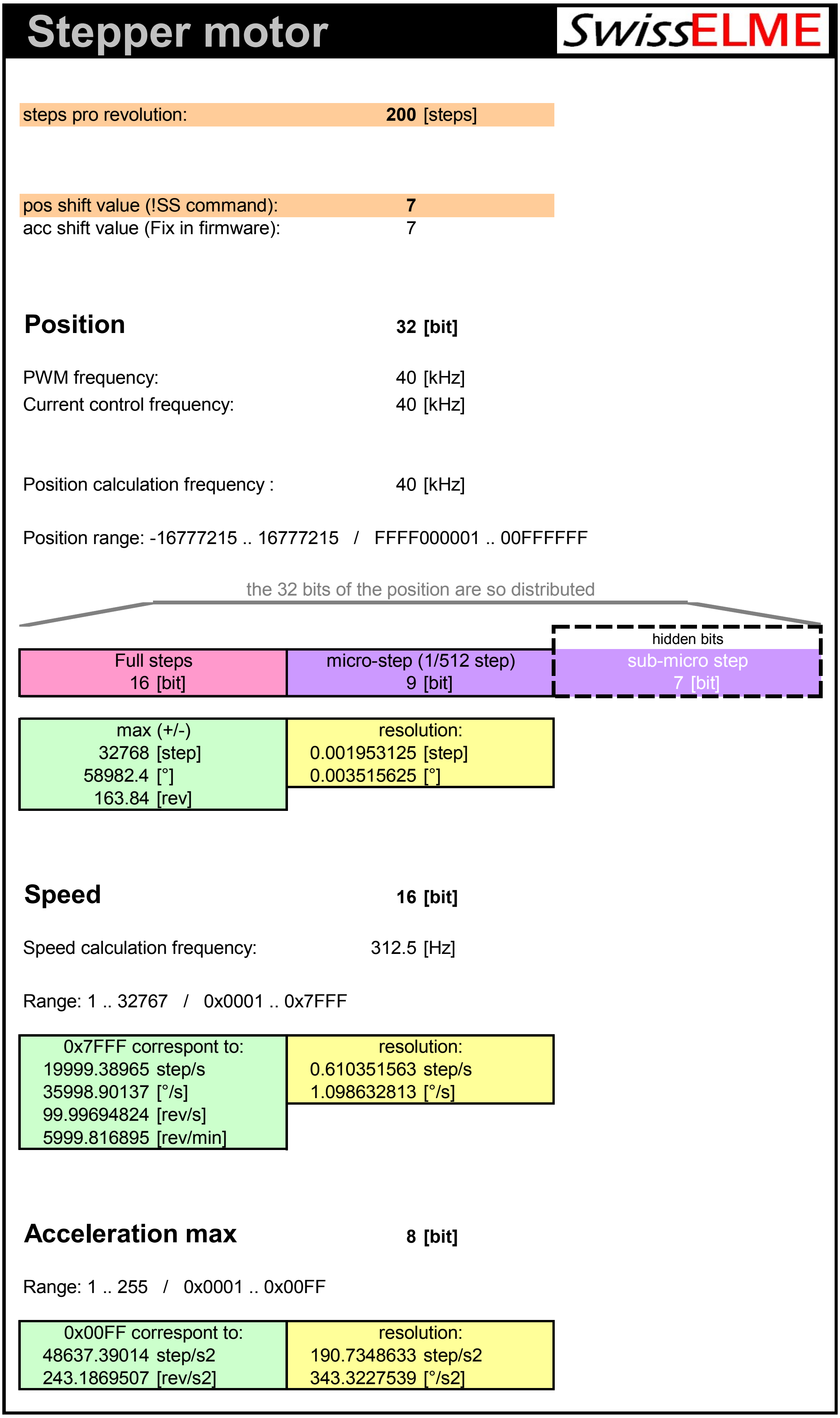
No parameters are requested. BLDC motor is a 3 phases.

11.6 Position shift value

If is necessary to increase the position range, position shift value is to be decreased. The speed range and the acceleration range are increased too. Position, speed and acceleration resolution are worse.

Increase the position shift value more as the default value is not recommended.

11.7 Excel sheets (default values)



Stepper motor

SwissELME

steps pro revolution:	**200** [steps]
pos shift value (!SS command):	**7**
acc shift value (Fix in firmware):	7

Position

32 [bit]

PWM frequency:	40 [kHz]
Current control frequency:	40 [kHz]
Position calculation frequency :	40 [kHz]

Position range: -16777215 .. 16777215 / FFFF000001 .. 00FFFFFF

the 32 bits of the position are so distributed

		hidden bits
Full steps 16 [bit]	micro-step (1/512 step) 9 [bit]	sub-micro step 7 [bit]

max (+/-)	resolution:
32768 [step] 58982.4 [°] 163.84 [rev]	0.001953125 [step] 0.003515625 [°]

Speed

16 [bit]

Speed calculation frequency: 312.5 [Hz]

Range: 1 .. 32767 / 0x0001 .. 0x7FFF

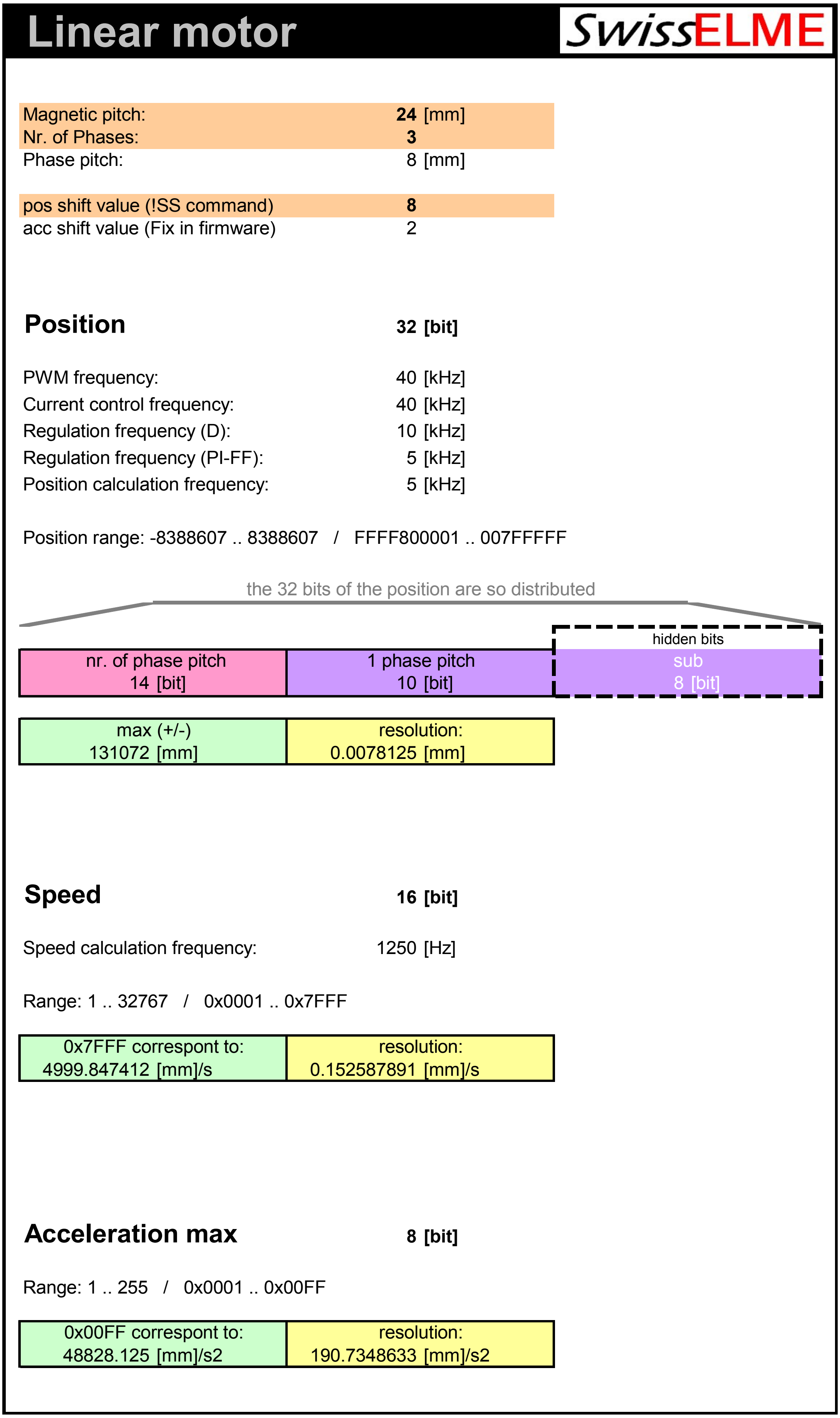
0x7FFF correspont to:	resolution:
19999.38965 step/s 35998.90137 [°/s] 99.99694824 [rev/s] 5999.816895 [rev/min]	0.610351563 step/s 1.098632813 [°/s]

Acceleration max

8 [bit]

Range: 1 .. 255 / 0x0001 .. 0x00FF

0x00FF correspont to:	resolution:
48637.39014 step/s2 243.1869507 [rev/s2]	190.7348633 step/s2 343.3227539 [°/s2]



Linear motor

SwissELME

Magnetic pitch:	**24**	[mm]
Nr. of Phases:	**3**	
Phase pitch:	8	[mm]

pos shift value (!SS command)	**8**
acc shift value (Fix in firmware)	2

Position **32 [bit]**

PWM frequency:	40	[kHz]
Current control frequency:	40	[kHz]
Regulation frequency (D):	10	[kHz]
Regulation frequency (PI-FF):	5	[kHz]
Position calculation frequency:	5	[kHz]

Position range: -8388607 .. 8388607 / FFFF800001 .. 007FFFFF

the 32 bits of the position are so distributed

nr. of phase pitch 14 [bit]	1 phase pitch 10 [bit]	hidden bits sub 8 [bit]
max (+/-) 131072 [mm]	resolution: 0.0078125 [mm]	

Speed **16 [bit]**

Speed calculation frequency: 1250 [Hz]

Range: 1 .. 32767 / 0x0001 .. 0x7FFF

0x7FFF correspont to: 4999.847412 [mm]/s	resolution: 0.152587891 [mm]/s

Acceleration max **8 [bit]**

Range: 1 .. 255 / 0x0001 .. 0x00FF

0x00FF correspont to: 48828.125 [mm]/s2	resolution: 190.7348633 [mm]/s2

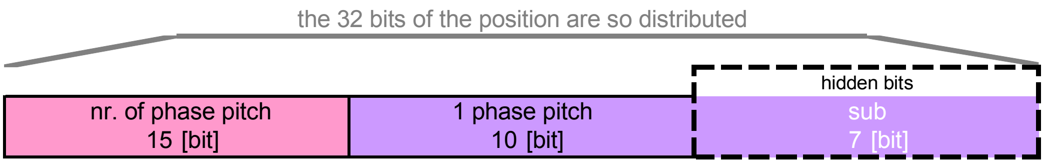
BLDC Motor 3 Phases

pos shift value (!SS command)	**7**
acc shift value (Fix in firmware)	2

Position **32 [bit]**

PWM frequency:	40 [kHz]
Regulation frequency (D):	10 [kHz]
Regulation frequency (PI-FF):	5 [kHz]
Position calculation frequency:	5 [kHz]

Position range: -16777215 .. 16777215 / FFFF000001 .. 00FFFFFF



max (+/-)	resolution:
3932160 [°] 10922.66667 [rev]	0.1171875 [°]

Speed **16 [bit]**

Speed calculation frequency: 1250 [Hz]

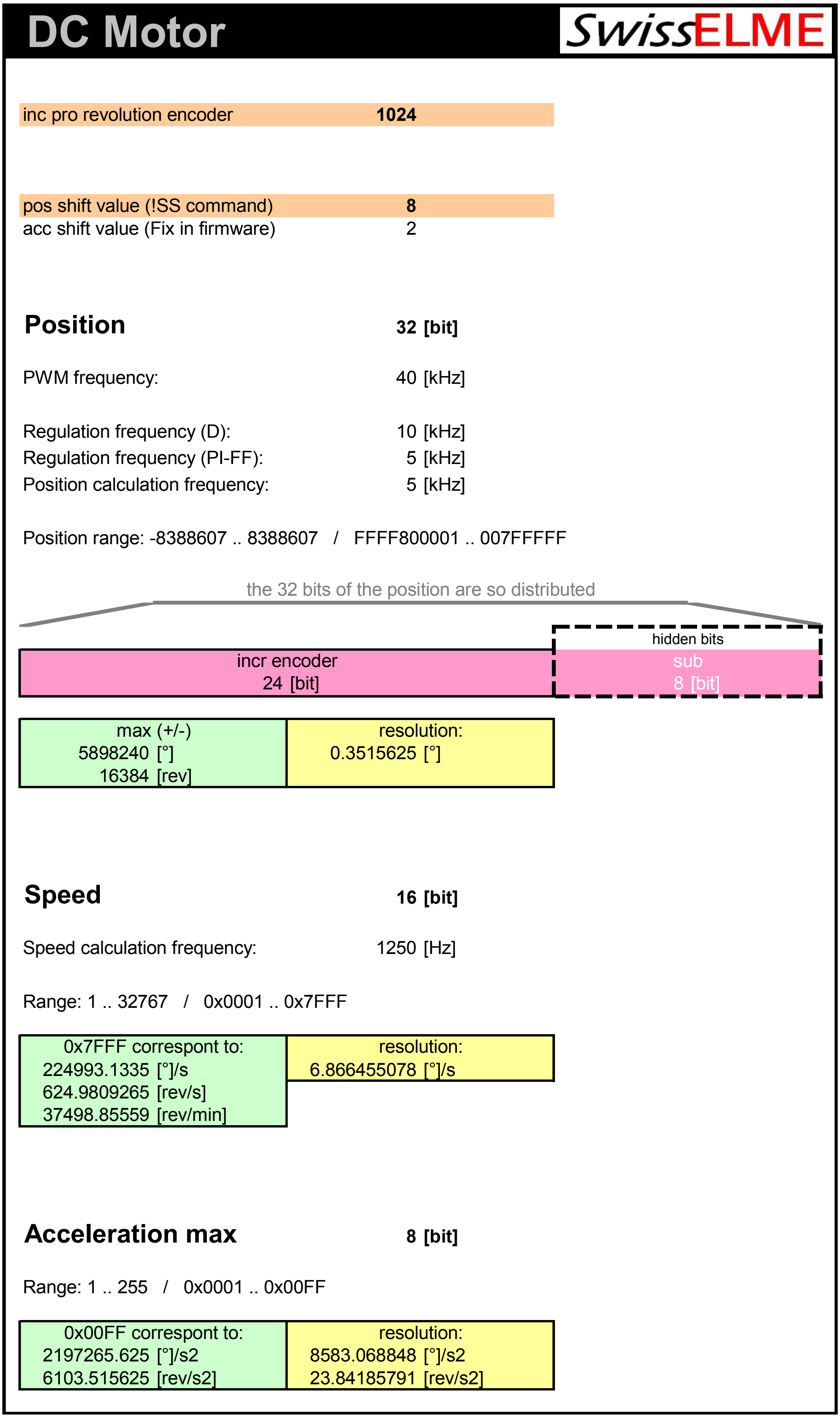
Range: 1 .. 32767 / 0x0001 .. 0x7FFF

0x7FFF correspont to:	resolution:
149995.4224 [°]/s 416.653951 [rev/s] 24999.23706 [rev/min]	4.577636719 [°]/s

Acceleration max **8 [bit]**

Range: 1 .. 255 / 0x0001 .. 0x00FF

0x00FF correspont to:	resolution:
1464843.75 [°]/s2 4069.010417 [rev/s2]	5722.045898 [°]/s2 15.89457194 [rev/s2]



DC Motor

SwissELME

inc pro revolution encoder	**1024**
pos shift value (!SS command)	**8**
acc shift value (Fix in firmware)	2

Position

32 [bit]

PWM frequency:	40 [kHz]
Regulation frequency (D):	10 [kHz]
Regulation frequency (PI-FF):	5 [kHz]
Position calculation frequency:	5 [kHz]

Position range: -8388607 .. 8388607 / FFFF800001 .. 007FFFFF

the 32 bits of the position are so distributed

incr encoder 24 [bit]	hidden bits sub 8 [bit]

max (+/-) 5898240 [°] 16384 [rev]	resolution: 0.3515625 [°]

Speed

16 [bit]

Speed calculation frequency: 1250 [Hz]

Range: 1 .. 32767 / 0x0001 .. 0x7FFF

0x7FFF correspont to: 224993.1335 [°]/s 624.9809265 [rev/s] 37498.85559 [rev/min]	resolution: 6.866455078 [°]/s

Acceleration max

8 [bit]

Range: 1 .. 255 / 0x0001 .. 0x00FF

0x00FF correspont to: 2197265.625 [°]/s2 6103.515625 [rev/s2]	resolution: 8583.068848 [°]/s2 23.84185791 [rev/s2]

12 Various

12.1 Stand alone

The Board can also be used as a stand alone application. The board will be one time programmed through the RS232 and the instructions will be memorized in the EEPROM memory of the M101 or M102. After the board is detached from the Computer, the board will run its own program. When the board is switched off and switched back on again, the program will restart the instruction sequence given.

13 Error messages

13.1 0x100 family of irreversible errors, power off

For these types of errors you must power off and power on the board, or perform a soft reset.

0x111	AltDMACError
0x112	AltMathError
0x113	AltStackError
0x114	AltAddressError
0x115	AltOscillatorFail
0x116	DMACError
0x117	MathError
0x118	StackError
0x119	AddressError
0x121	OscillatorFail
0x122	T1Interrupt
0x123	MPWM1Interrupt
0x124	ADC1Interrupt
0x125	SPI1ErrInterrupt
Cause	This is a special error. If this happens please contact SwissELME

0x126	EEPROM ACK
0x127	EEPROM STOP
Cause	This is a special error. If this happens please contact SwissELME

0x128	Hall Current wrong
Cause	This is a special error. If this happens please contact SwissELME

0x129	Motor Type not defined
Cause	Instruction could not execute because no motor type is defined. Do instruction MT to define motor type

0x131 0x132 0x133 0x134 0x135 0x136	ERROR CAN TXBO ERROR CAN TXBP ERROR CAN RXBP ERROR CAN TXWAR ERROR CAN RXWAR ERROR CAN RBOVIF
Cause	This error appears if there are some problems on the CAN BUS. Refer to chapter 4.13 CAN bus and terminations.

13.2 0x200 family reversible errors, power off

These errors take away the power of the motor for safety. You will need quit then the error (QE) and restore power.

0x211	Power is disabled
Cause	This error appears if user try to execute a movement but power is off (PW) or is pending error (XE)
Solution	Quit error or power on the motor

0x212	Software short circuit detected
Cause	This error appears if board detected a software overcurrent.
Solution	Check acceleration, speed, cables, regulation parameters

0x213	Hardware short circuit detected
Cause	This error appears if board detected a short circuit on motor.
Solution	Check acceleration, speed, cables, regulation parameters

0x214	Hardware negative limit switch
Cause	This error appears if the motor reached the min position defined by instruction LM.
Solution	Check the movement and the motor position

0x215	Hardware positive limit switch
Cause	This error appears if the motor reached the max position defined by instruction LN.
Solution	Check the movement and the motor position

0x216	Software max current detected for DC motor
Cause	This error appears if DC motor or BLDC motor reached the max defined allowed current by instruction MC.
Solution	Check acceleration, speed, cables, regulation parameters

0x217	Following error
Cause	This error appears if the motor had a problem to follow the wanted position.
Solution	Check acceleration, speed, cables, regulation parameters

0x218	Voltage low
Cause	This error appears if the power supply reached the min voltage value defined by SV
Solution	Check the power supply, and check if the power supply have enough watt.

0x219	Motor out of sensor (BLDC, LIN)
Cause	This error appears if there ware a problem reading the hall sensors of BLDC or linear motor
Solution	Check the cables and check if the slide of the linear motor goes out of the correct position range

0x221	Soft limit switch
Cause	This error appears if the motor reached the programmed limit switch
Solution	Check the movement

0x222	Math limit switch
Cause	This error appears if the desired destination position is out of range
Solution	Check the movement

0x223	Angle delta error
Cause	This is a special error. If this happens please contact SwissELME

0x224	Voltage is off
Cause	This error appears if the power supply is not present.
Solution	Check power supply before PW0001

0x225	Voltage high
Cause	This error appears if the power supply is too high
Solution	Check power supply before PW0001

0x231 0x232 0x233 0x234 0x235 0x236 0x237 0x238	ERROR2 MOS error Q7 4H and Q8 4L ERROR2 MOS error Q7 4H or Q8 4L ERROR2 MOS error Q3 2H and Q4 2L ERROR2 MOS error Q3 2H or Q4 2L ERROR2 MOS error Q1 1H and Q2 1L ERROR2 MOS error Q1 1H or Q2 1L ERROR2 MOS error Q5 3H and Q6 3L ERROR2 MOS error Q5 3H or Q6 3L
Cause	This error appears if there is a hardware problem on one of the four power bridges.
Solution	Repair the board

0x226	Value out of range
Cause	This error appears if the programmed value is outside the valid range
Solution	Check command

13.3 0x300 family reversible errors, power on

These errors do not take away the power of the motor. You will need quit then the error.

0x311	Initialize max stroke error
Cause	This error appears if during initialization process the motor moves too much, reaching the max defined acceptable initialization stroke defined by IS
Solution	Check if there is a problem during initialization

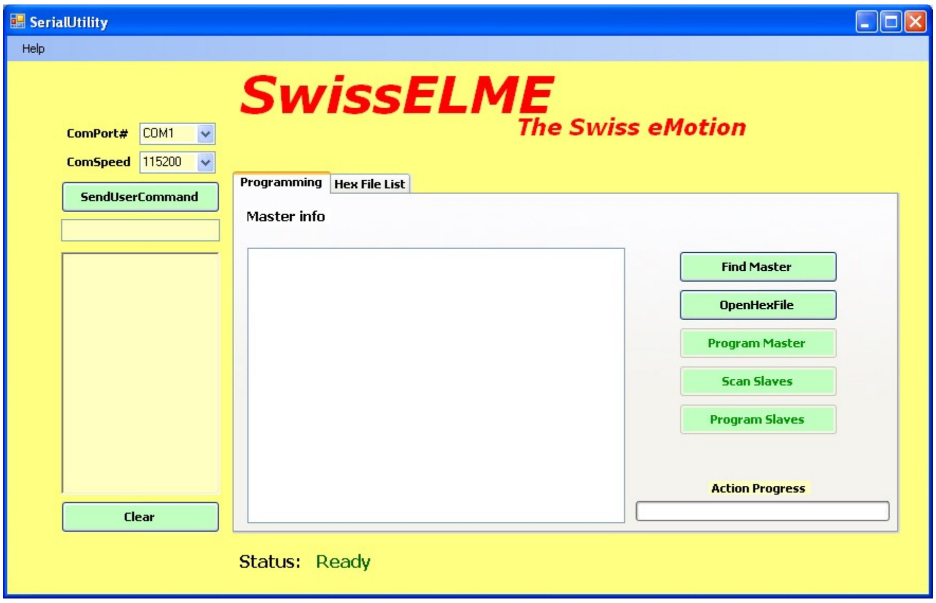
0x312	Initialize not possible
Cause	This error appears if user try to execute initialization and no limit switch is defined (IA or IB).
Solution	Correct the parameter

0x313	Power is enabled
Cause	This error appears if user try to execute a command that is not available when the board in in power on state (`!PW1`).
Solution	Execute command in power off state (`!PW0`)

14 Firmware upgrade

14.1 Serial utility

The serial software utility will upgrade the board firmware via RS232. Software is compatible with MS-Windows 2000/XP/Vista/7.

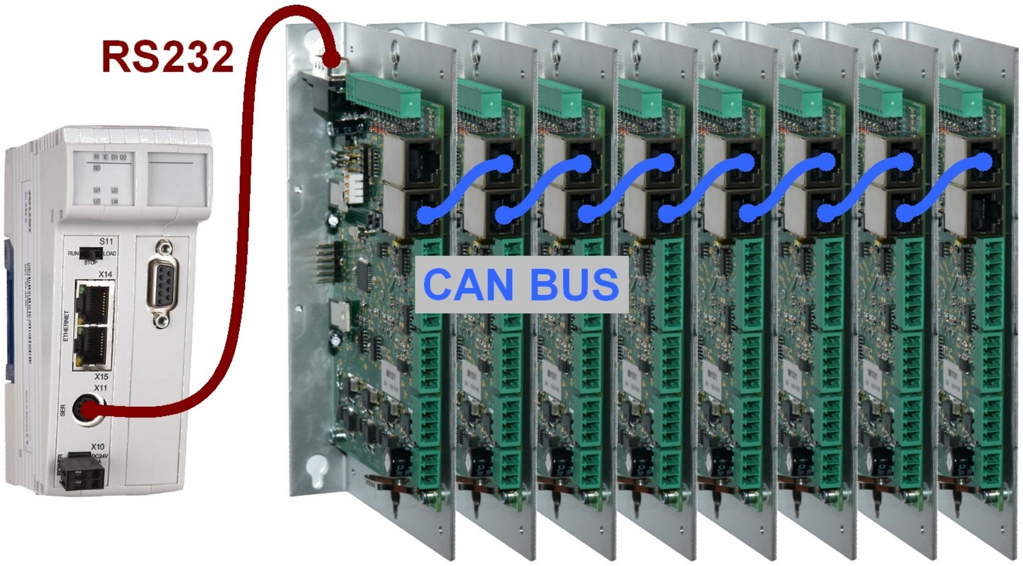


ComPort# COM1

ComSpeed 115200

Status: Ready

This software will also run the upgrade on the cards slaves connected via CAN bus.



SwissELME developed the Serial Utility to allow the firmware upgrade by the customer itself with standard RS232. This tool allow to upgrade all the slaves connected to the master too.

15 HW / Firmware compatibility

M101 does support all firmware
M102 does support firmware from 2.35
M103 does support firmware from 2.42