FAULHABER

Motion Controller

4-Quadrant PWM for DC-Micromotors and Brushless DC-Servomotors

Series MCDC 5004 | **Series MCBL 5004**

FAULHABER GROUP
We create motion

Operating Instructions

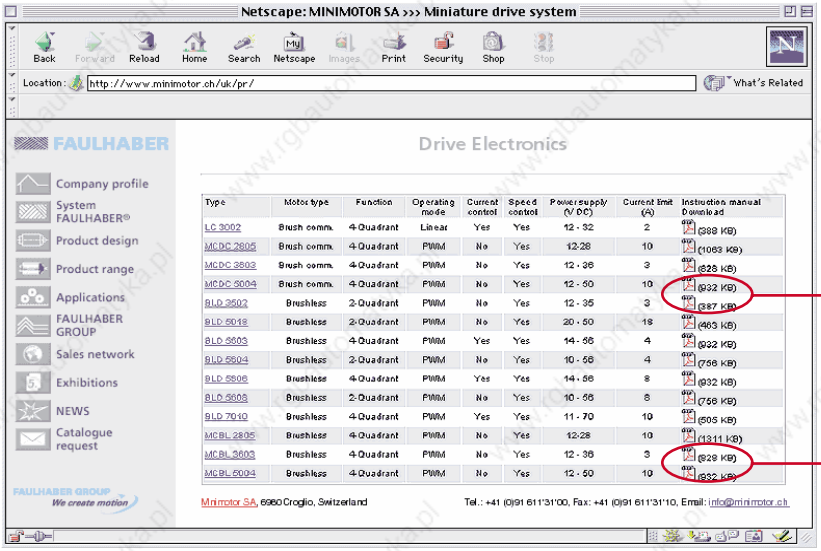
www.faulhaber.com

Miniature Drive Systems

Micro Drives
DC-Micromotors
Precision Gearheads
Servo Components
Drive Electronics

Surf to the following Internet address and you will find the latest edition of the instruction manual on-line:

www.minimotor.ch/uk/pr/



For direct Download:

http://www.minimotor.ch/minicatalog/pdf/DriveCircuits/Manuals/IM_e_MCDC_MCBL_5004.pdf

Index

Description

1. Description

The MCBL 5004 and the MCDC 5004 are very compact motion controllers ideal for our brushless DC-Servomotors and brushed DC-Micromotors.

Each model comprises a PWM servo amplifier.

Technology

Both motion controllers are based on a fast, powerful 16 bit microcomputer system.

This guarantees high dynamics, precise positioning and quiet running, regardless of the motor type used.

The well thought-out design and consistent application of SMD technology ensures a very compact device. The specially developed user software offers high flexibility and simple handling.

Application field

Developed with the use of state-of-the-art technology, the motion controllers are suitable for a wide range of applications: insertion and handling machines, machine tools, robots, X/Y tables, drive and automation systems in medical technology, chemical and food industry, etc.

Programming

One of the most important objectives in the development of these units was to keep its operation as simple as possible. This has been attained with the use of just a few, highly efficient functions.

Manual balancing or potentiometers are no longer required. Menu-guided program and parameter-editing functions are already integrated for operation with an ASCII terminal. In place of internal menu management, the clearly structured command set can be simply integrated into a customer-specific interface, e.g. with Visual Basic, Lab View, Pascal, C++, etc.

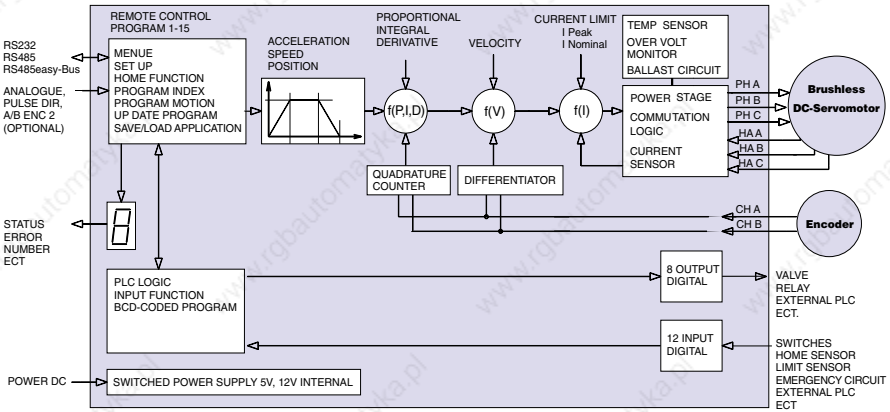
Any PC with Windows operating system can be used as an input terminal. Program up-dates are made directly via the serial interface without changing the hardware.

Communication is made via the serial port RS232 or RS485.

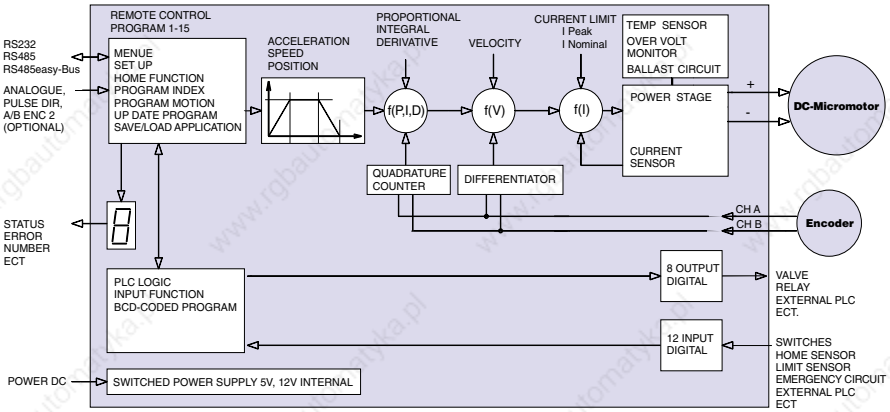
Model overview

2. Model overview

MCBL 5004	Integrated PWM servo amplifier 50V-4A	**Brushless DC-Servomotors with encoder**



MCDC 5004	Integrated PWM servo amplifier 50V-4A	**Brushed DC-Micromotors with encoder**

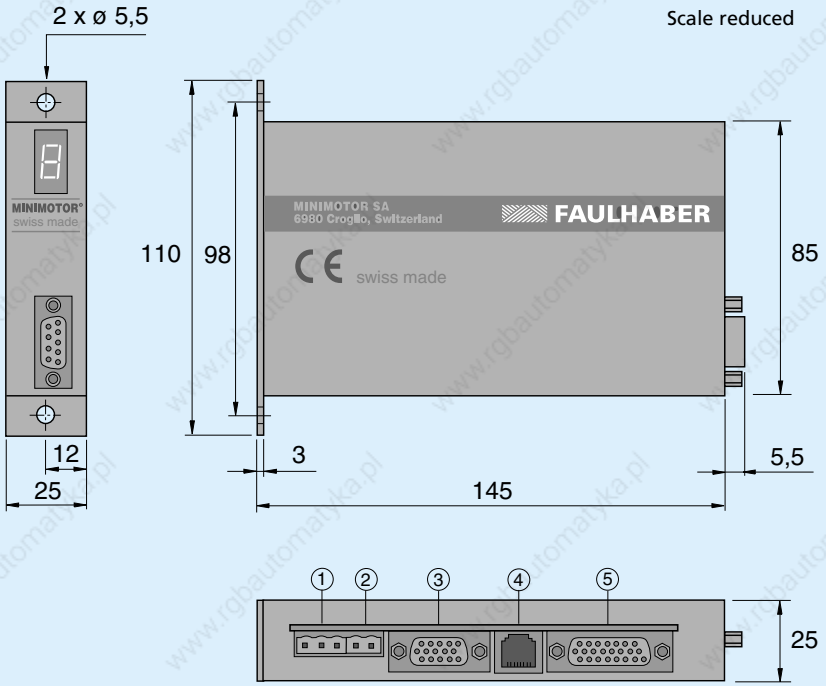


Technical information

3. Technical data	MCBL 5004	MCDC 5004	
Electrical data			
Supply voltage	12 ÷ 50	12 ÷ 50	V DC
PWM switching frequency	20	20	kHz
Max. continuous output current	4	4	A
Max. peak output current	10	10	A
Max. encoder frequency	200	200	kHz
Software data			
Program memory (16 bit access)	256 x 8	256 x 8	kbyte
Sampling period	500	500	µs
Number of programs	15	15	
Lines per program	50	50	
Number of indexes	50	50	
Communication data			
Interface	RS232 / RS485 / RS485easy-Bus		
Status display	7 segment display		
Inputs (5V pull-up standard, optional 24V pull-down)	12		
Outputs (6 x 50V/500mA open collector, 2 x TTL level)	8		
Program and parameter editor	integrated ASCII terminal		
Program up-date	via serial interface		
Application and parameter save / load	via serial interface		
Starting position function	via encoder Z-index / via external sensor		
Temperature rating			
Operating temperature	0 ... + 55		°C
Storage temperature	-20 ... + 80		°C
Weight / Dimensions			
Weight:	370		g
Dimensions: see diagram on page 5			

Dimensions

4. Dimensions for MCBL 5004 and MCDC 5004



Connection for MCBL 5004

① Motor
② Power supply
③ Encoder[1] and Hall sensors
④ Input for special function
⑤ Digital Input / Output

Connection for MCDC 5004

① Motor
② Power supply
③ Encoder[1]
④ Input for special function
⑤ Digital Input / Output

[1] Line driver encoders for noisy environments or long distances can be used.

Start-up procedure

5. Start-up procedure

Here we list a step-by-step start-up procedure for both the electrics and software. Also included are several examples in order to allow the user to test the unit and familiarise himself with programming.

We therefore recommend that this sequence is followed for trouble-free installation:

Start-up Procedure for MCBL 5004

- Connect the motor phases to **MOTOR**
- Connect the encoder and the motor Hall sensor leads to **ENCODER HALL**
- Connect the RS232 (or RS485) to the computer port **COM1**
- Connect the power supply to **PWR**
- Power the motion controller
- Software start-up

Start-up Procedure for MCDC 5004

- Connect the motor terminals to **MOTOR**
- Connect the encoder to **ENCODER**
- Connect the RS232 (or RS485) to the computer port **COM1**
- Connect the power supply to **PWR**
- Power the motion controller
- Software start-up

The computer link is necessary to program the motion controller.
After programming has been completed, the computer link can be disconnected since the programs can be started using the motion controller input functions.

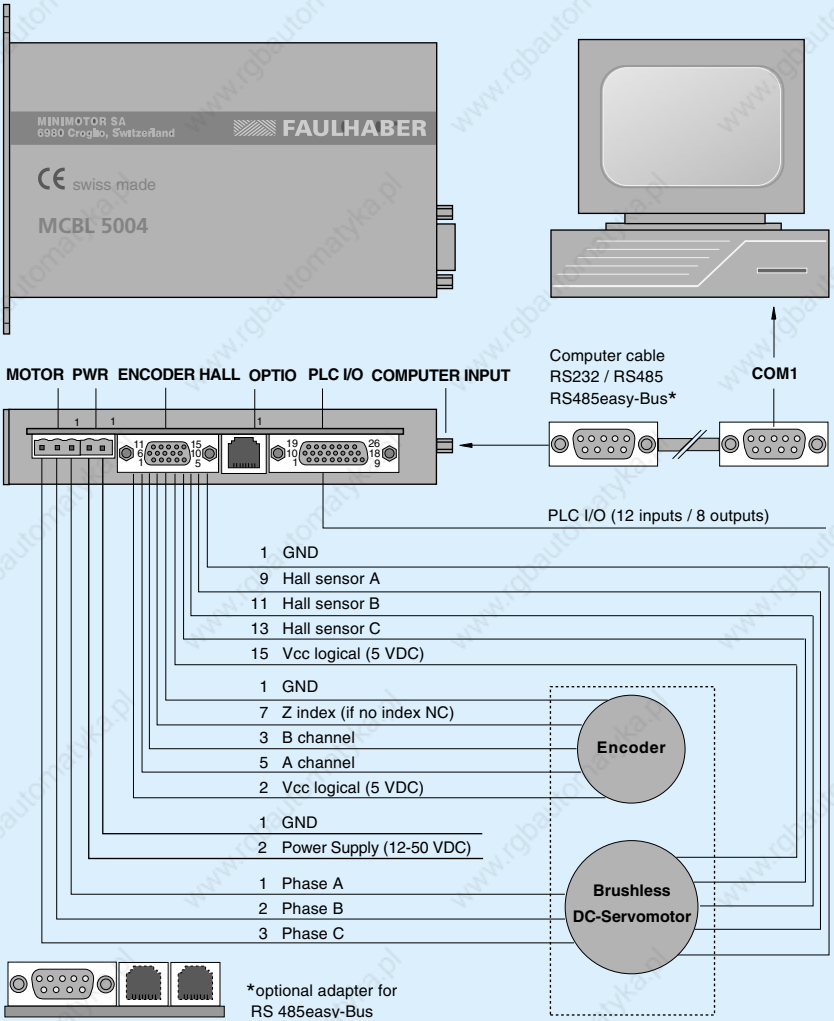
For advanced functions such as:

- Analogue input command
- Stepper motor emulation
- Second encoder input
- RS 485 serial interface
- Multi-axis operation

please refer to the specifique chapters

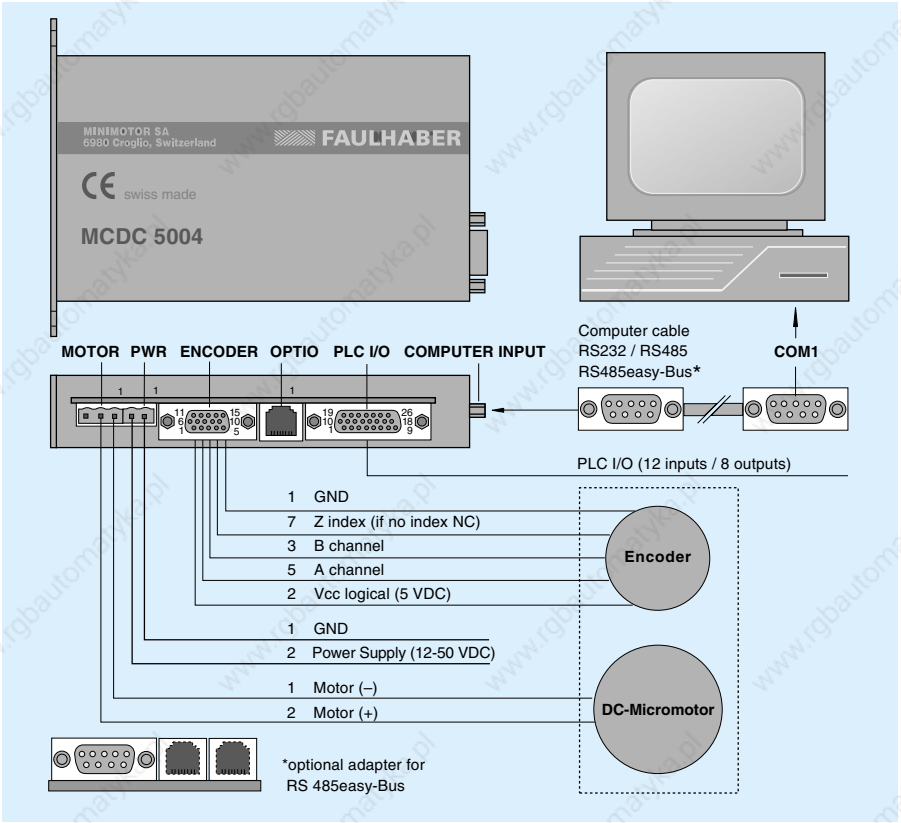
Start-up procedure

6. Connection diagram for MCBL 5004



Start-up procedure

7. Connection diagram for MCDC 5004



PLC I/O description

The PLC I/O port enables direct communication with the position control function without having to use a computer. For example, once a program has been created, it can be executed by simply giving a command to the assigned input. In the initial phases of installation, and to allow the user to better understand the operation of the motion controller, all instructions are given via computer. It is therefore not necessary to connect this port.

RS232 description

This port is the communication link between the motion controller and the external computer via the COM1 connection point. Additional information regarding the connection and set-up is given in chapter 32. The link is make with a standard computer cable which, if necessary, can be optionally supplied by Minimotor.

Start-up procedure

8. Software start-up

The diskette supplied with the motion controller contains:

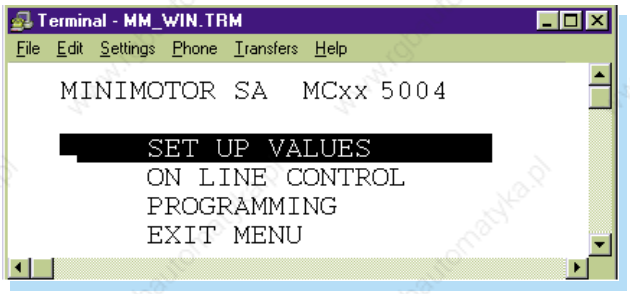
installation program

program MCxx_yyy.S19 (basic motion controller software)

terminal emulator for Windows

program MM_WIN.TRM with the pre-defined terminal emulator parameters

- To copy the software onto your hard disk using Windows, first of all start the installation program from the DOS prompt which copies the contents of the diskette into a directory MIMO_MC which is automatically created. You should now work only on your hard disk.
  The diskette now serves as a back-up for security purposes only.
- Go to the MIMO_MC directory and open the file TERMINAL.EXE.
- Go to FILE menu and open MM_WIN.TRM.
- Press MENU button on the screen. The following menu will appear:



As a check, the motion controller 7-segment display should show 0 (zero).

If this does not happen, you should first check all the connections. If the above menu still does not appear on the screen, you should re-install the software program MCxx_yyy.S19. Instructions for this procedure can be found in chapter 28.

At this point, the start-up procedure is complete. The unit is now ready to be programmed for its specific application.

General software information

Terminal emulator description

The computer is only used as a terminal. The terminal emulator therefore enables communication between the computer and motion controller software. The actual programming is made directly in the motion controller itself.

Motion controller software organisation
The software is constructed on three different levels.

- **Operating system**
  The operating system normally remains invisible to the user and is a background function for:
  – download functions
  – back-up in emergency situations
  For additional information, see chapter 28.
- **Program MCxx_yyy.S19**
  Is the basic working program which realises all described functions and programming possibilities. This program is already installed within the unit and automatically goes into operation once the system is started-up. Actualy version 3.40
- **Application user programs**
  Contains the complete set of customer-defined data and parameters (= “application”).

7-segment display status

Display	Description
0	Device active, Servo amplifier OFF
1	Servo amplifier ON, closed loop system active, ready for motion
F	Operating system active
xx blinking	xx error code, see chapter 27

General programming instructions

General instructions on how to move, insert, delete, etc. within the program:

- Close every entry with the command <ENTER>
- Text can be entered using either small or capital letters
- Use the arrows to move up and/or down the menu lines
- To go back to the previous menu always use <ESC>
- Close erroneous entries with <ENTER> and re-enter data

Delete characters	Back Space
Clear line	<CRTL C>
Insert line	<CRTL I>
Page down	<CRTL D>
Page up	<CRTL U>

Set up values

9. Pre-defined set up values

The SET UP VALUES menu contains all the necessary parameters for the unit to function according to the specific application.

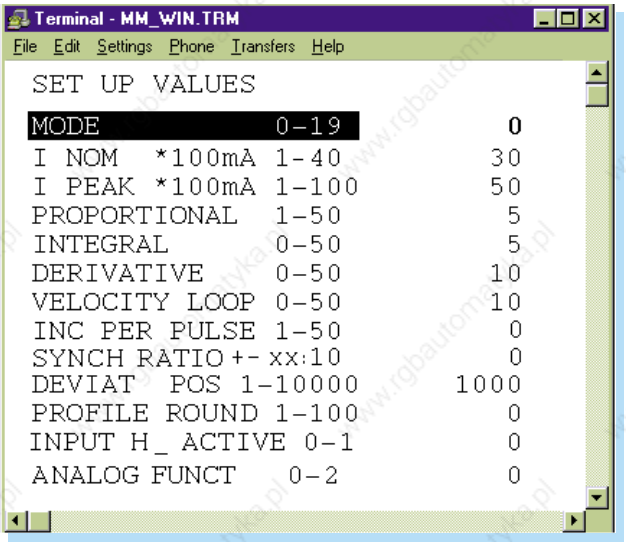
The motion controller has default parameters which allows it to run the majority of motors stably, allowing the user to fine-tune the unit's operation without the risk of causing damage.

For optimum operation it is of course necessary for the individual parameters to be set according to the requirements of the specific application. A description on how to optimise the system is contained in chapter 11.

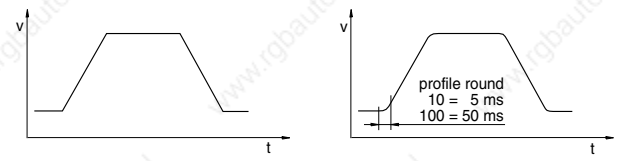
As an initial step the user should familiarise himself with the motion controller instructions which are available via the ON LINE CONTROL menu. As we will see later, it is important to first get the motor running before optimising the parameters.

Attention!

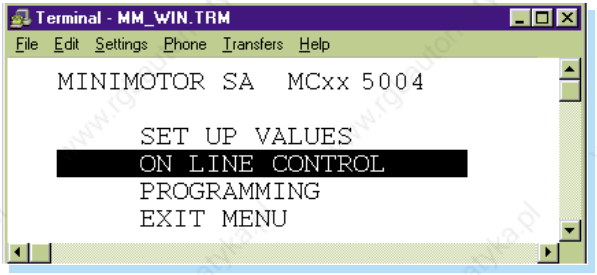
Before giving any instructions to the motor, ensure that it is first running without load since the current limiter is not set to the maximum motor current.



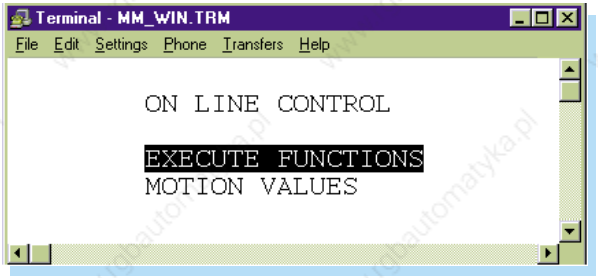
Set up values

Description	
MODE	0 = Programs and commands operated using the standard inputs (see chapter 19) 2 = Programs and commands operated using the standard inputs. PULSE/DIRECTION input signal for stepper control emulation function active (see chapter 23) 3 = Programs and commands operated using the standard inputs. Analogue input signal for digital speed control function active (see chapter 22) 10 = Programs operated using the 4 binary coded inputs and input 8 as starting trigger (see chapter 20) 12 = Programs operated using the 4 binary coded inputs and input 8 as starting trigger. Stepper control emulation function active (see chapter 23) 13 = Programs operated using the 4 binary coded inputs and input 8 as starting trigger. Analogue input signal for digital speed control function active (see chapter 22)
I NOM	Nominal current
I PEAK	Peak current
PROPORTIONAL	Proportional closed loop parameter (stiffness)
INTEGRAL	Integral closed loop parameter (positioning precision)
DERIVATIVE	Differential closed loop parameter (stability, dynamic)
VELOCITY	Velocity closed loop parameter (oscillation prevention)
INC PER PULSE	Increment (lines) per pulse in MODE 2 or 12 for stepper emulation function (see chapter 23)
SYNCH RATIO	Synchronous ratio for operation with optional second encoder (see chapters 24 and 25)
DEVIATE POS	Permissible max. position deviation in lines
PROFILE ROUND	To smooths the speed profile out (see below)  By setting to profile round, the speed profile is smoothed out, this reducing mechanical stress for better live performance.
INPUT H-ACTIVE	0=Input active low, I=Input active
ANALOG FUNCTION	Analog function active with mode 3 or 13 for digital speed control. 0=CW (+), I=CCW (-), 2=Cw and CCW (+/-)

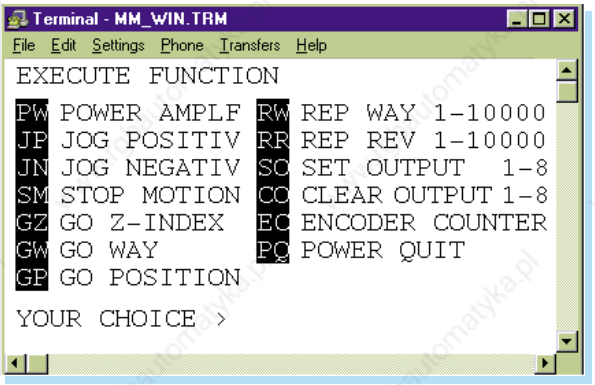
10. On line control



10.1 Execute functions



The EXECUTE FUNCTIONS menu only lists a selection of the most important available instructions. However, other functions can be executed via this menu.



On line control

Command	Parameter	Description
AC	1 000 - 4 000 000 lines/s^2	Acceleration
AIX	10 - 50 000 (x 1 000) lines/s^2	Overwrite acceleration index at preloaded NIX number by remote control
ANF	0 - 2	Analog function mode 3/13 0 = CW (+) 1 = CCW (-) 2 = CW and CCW (+/-)
CI	0-100	Card identifier for RS485easy-Bus
CO	1 - 8	Clear output
CLO	0 - 1	Clear outputs after HOME function 0 = no, 1 = yes
DIX	± 2 000 000 000 lines	Overwrite distance index at preloaded NIX number by remote control
DRH	1 - 2	Direction of motor rotation, for seeking coarse sensor 1 = CW, 2 = CCW
DRZ	1 - 2	Direction of motor rotation, for seeking Z mark sensor 1 = CW, 2 = CCW
DP	lines	Permitted position deviation in lines
DV	0 - 50	Differential closed loop parameter
EC		Encoder counter on-line diagnosis
ED	1 000 - 5 000 000 lines/s^2	Emergency deceleration with Exit function EE and Limit-switch function LL and LR
GP		Go to position (absolute)
GW		Go way (relative)
GZ		Go to Z-index (encoder)
HO		HOME function according to program
HOF	0 - 100 000 increment	Offset after edge coarse sensor no stop same direction, if HOF is not 0 this value is indicated on HOME menu
ICP	1 - 50	Inc. per pulse, mode 2/12, pulse / direction control
IHA	0 - 1	0 = input low active, 1 = input high active
IN	1 - xx	Nominal current
INH	1 - 8	HOME sensor input number
IP	1 - xx	Peak current
IT	0 - 50	Integral closed loop parameter
IX	1 - 50	Run index # according to program
JP		Jog (run) positive, constant speed
JN		Jog (run) negative, constant speed
JNZ[letter]	1 - 50	Indicate loop reference letter (from A to E). Decrements the loop repeats, whereby if not zero, jump to line xx
NIX	1 - 50	Number index pre-load for changing index parameters by remote control
PG	1 - 15	Run program #
PO	± 2 000 000 000 lines	Position (absolute)
PP	1 - 50	Proportional closed loop parameter
PRF	1 - 100	Rounding of speed profile (should be value), smooth start and smooth stop
PQ		Servo amplifier power OFF
PW		Power ON, reset position counter
PWC		Power ON continue, keep position counter
RI	0 - 100	Required identifier for RS485easy-Bus
*RI	1 - 99	Get back identifier, position, and status complete
RR	1 - 10 000	Repeat way CW/CCW
RW	1 - 10 000	Repeat way (same direction)
SET[letter]	1 - 10 000	Set loop reference letter (five possibilies, from A to E) and number of repeats xxxx

On line control

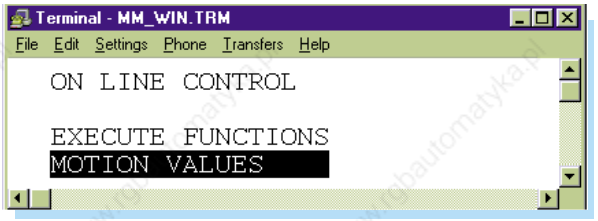
Command	Parameter	Description
SIX	25 - 1 000 000 lines/s	Overwrite speed index at preloaded NIX number by remote control
SM		Stop motion
SO	1 - 8	Set output
SP	25 - 1 000 000 lines/s	Speed
SR	± 1 - 100 :10	Synchronisation ratio with optional second encoder
TE		Tell error codes 01-99
TGD	± 2 000 000 000	Trigger downward count, absolute, at output x (5 ms) defined in output function
TGU	± 2 000 000 000	Trigger upward count, absolute, at output x (5 ms) defined in output function
TI	? or 1 - 12	Tell status input, 0=Low 1=High
TP		Tell actual position ± 2*10E9
TS		Tell status: 0 = power OFF 1 = power ON 2 = moving 3 = program active 9 = error
VL	1 - 50	Velocity closed loop parameter
WA	± 2 000 000 000 lines	Way (relative)
WT	x 10 ms	Waiting time

Here are some examples of how the system's performance can be controlled:

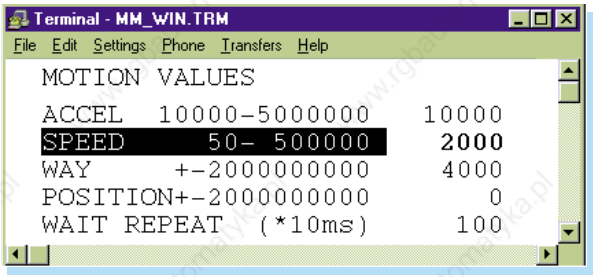
> **PW <ENTER>**	The 7-segment display indicates "1", the motor is stationery and maintains its position.
> **JP <ENTER>**	The motor runs at a constant CW speed.
> **SP10000 <ENTER>**	The motor runs at 2 500 lines/s.
> **EC <ENTER>**	The encoder counter value appears on the screen. To return to the previous menu, press ESC.
> **SM <ENTER>**	The motor stops and maintains its position.
> **PQ <ENTER>**	The power stage is switched off. The 7-segment display indicates "0".

10.2 Motion values

As well as describing the function of the MOTION VALUES menu, there is also a graphical description of how a speed profile can be programmed.

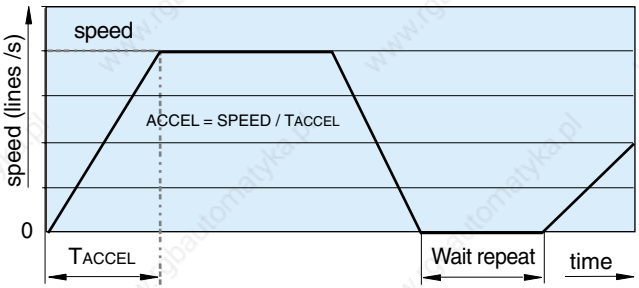


On line control



ACCEL	Acceleration in lines per second^2
SPEED	Speed in lines per second
WAY	Way in lines (relative) for RR and RW instructions
POSITION	Position in lines (absolute)
WAIT REPEAT	Wait time for RR and RW instructions

Parameters graphic explanation (example with RW)



Encoder resolution

Using the two encoder channels, the motion controller internal electronics multiply the encoder resolutions by 4. For example, when using a 500 line incremental encoder, one revolution of the motor shaft corresponds to 2 000 lines.
Therefore WAY 2 000 corresponds to one motor shaft revolution.

Acceleration and deceleration

When setting the acceleration, the system automatically sets the deceleration to the same value.

Set-up values

11. System parameter set-up

Current limiter set up

The current limits I NOM and I PEAK must be set according the motor used.
The value of I NOM should not exceed the motor's recommended current for continuous operation.
I NOM limit is only active during constant speed operation.
I PEAK limit is only active during acceleration and deceleration.

There is a continuous monitoring of incremental feedback. If the motor is blocked more than 0,5 seconds then the current will be automatic reduced.

Optimising the closed loop parameters

The closed loop system can be optimised by running the motor (including assembled mechanical parts) directly on line and by adjusting the following parameters via the SET UP VALUES menu:

PROPORTIONAL	(1 - 50)
INTEGRAL	(0 - 50)
DERIVATIVE	(1 - 50)
VELOCITY	(1 - 50)

This optimisation is best carried out by running the motor with the RW and/or RR instructions.

When executing these instructions, all parameters (even set-up) can be changed on line, thus enabling the user to see the reaction of the system whilst making changes. One helpful function is the EC (encoder counter) which gives information on the actual motor shaft position.

Improved dynamics

If your application requires more dynamics, this can be obtained by increasing the PROPORTIONAL, DERIVATIVE and VELOCITY LOOP values (e.g. to 10, 20 and 20).
If the motor is noisy or vibrates (indicating system instability), these parameters should be reduced.

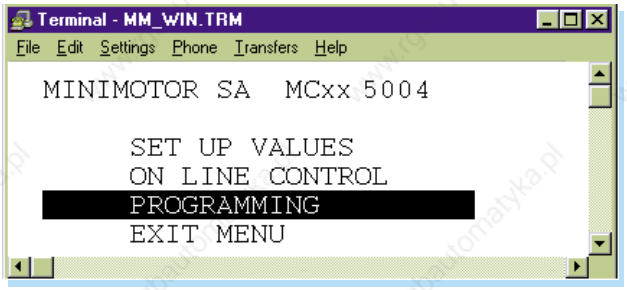
Precise positioning

If you need to improve the motor's position holding, an INTEGRAL value should be given (e.g. 5). The INTEGRAL value is only activated when the motor reaches the requested position. In this way the system's dynamic is not influenced by this value.

To control the exact position of the motor, the EC (encoder counter) command is used via the ON LINE CONTROL menu.

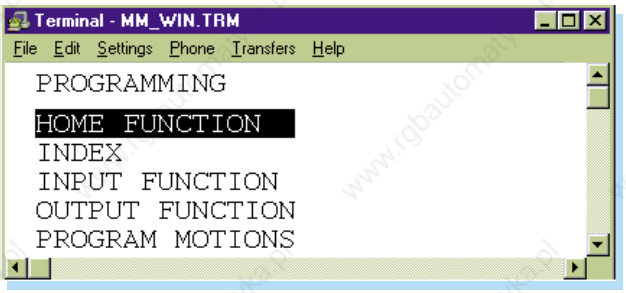
12. Programming

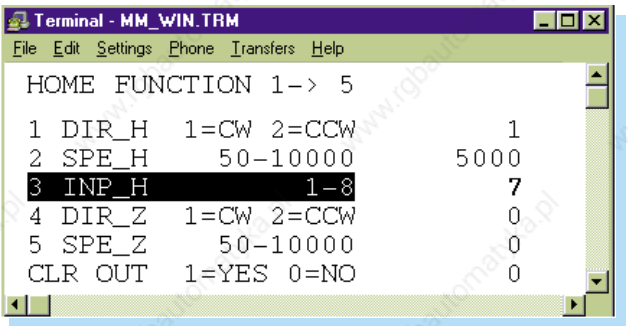
Here we look into the individual PROGRAMMING menus of the system.
Practical examples of how the motion can be controlled are to be found at the end of this section.



12.1 Home function

This is the fixed reference starting (home) position. The home command can be given whenever the system should return to the home position.





Programming

DIR_H	Direction of rotation to go to external sensor
SPE_H	Speed for finding external sensor
INP_H	Assign external sensor input
DIR_Z	Direction of rotation to go to encoder Z index
SPE_Z	Speed for finding encoder Z index
CLR OUT	Clear all outputs after HOME

The home function runs from line 1 through line 5 and can be executed using either an external sensor or the encoder Z index. Therefore, for example:

- If the home function is executed using an external sensor, lines 4 and 5 are set as follows:

  DIR_Z **0**

  SPE_Z **0**

- If the home function is executed using the encoder Z index, the first 3 lines are set as follows:

  DIR_H **0**

  SPE_H **0**

  INP_H **0**

- For an extremely precise home position use first the external position sensor and then the encoder Z-index. In this case all 5 lines in the home function menu must be set.

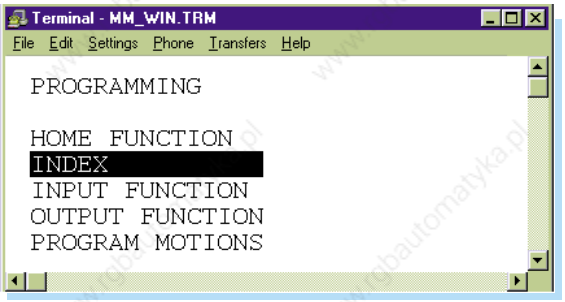
- There is also a possibility to set the home function without using either an external sensor or the encoder. Only in this case all parameters are set to zero and the system's position at the time of executing home is considered the home position.

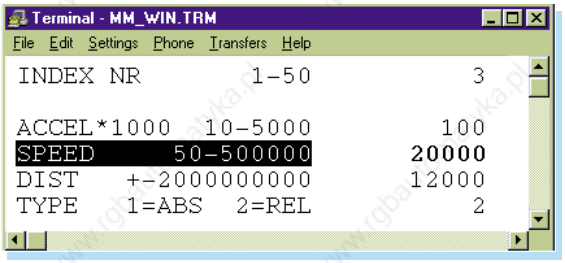
To complete the home function you can also define if the digital outputs should be cleared or not (CLEAR OUTPUT).

If the external sensor is already activated (INP_H activated) when this function is executed, the system double-checks this position by moving backwards and forwards to always return to the home position from the same direction.

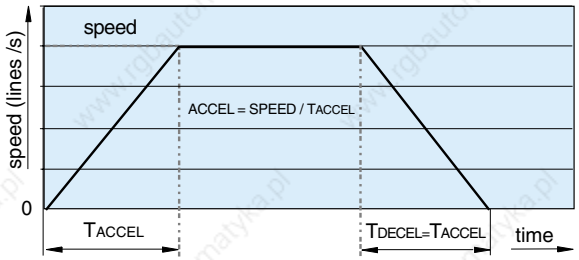
12.2 Index

The INDEX menu allows the definition of 50 different movements. To define each movement, the acceleration, maximum speed and the distance to be travelled must be programmed.





ACCEL	Acceleration k lines per second²
SPEED	Speed in lines per second
DIST	Distance in lines
TYPE	Absolute to the home position, or relative to the actual position (see chapter 13)



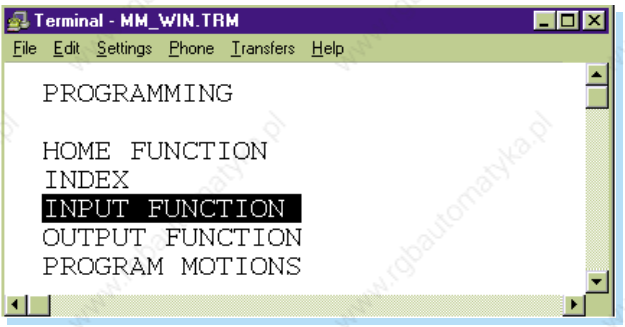
For example, to execute index 3, in the ON LINE CONTROL menu you must enter

> **IX3 <ENTER>**

Programming

12.3 Input function

The INPUT FUNCTION menu allows us to assign a specific function or instruction to a digital input on the PLC I/O port.

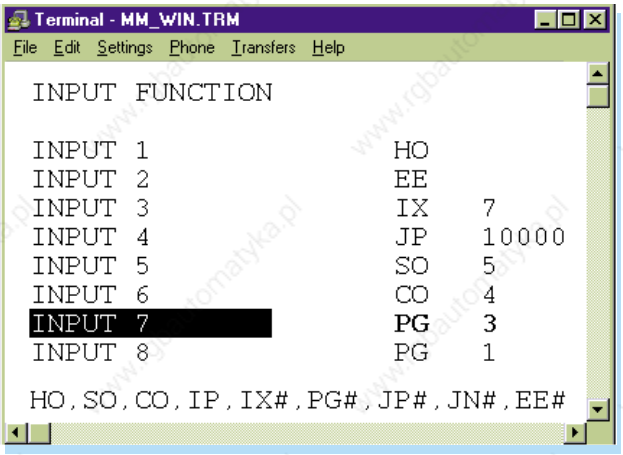


The following is a list of the instructions that can be executed directly via the INPUT FUNCTION menu:

EE		Emergency exit with ramp and power quit *
EE	**1**	EE with ramp, stays in position, power on *
IP		Interrupt program, as long as input is active
LL		Limit-switch left *
LR		Limit-switch right *
HO		Execute HOME according to HOME FUNCTION
IX	**xx**	Execute index number xx
IX	**xx = yy**	Index number xx distance is yy (IX 1=1 000)
IX	**xx – yy**	Index number xx distance is –yy (IX 3 – 5)
IX	**xx + yy**	Index number xx distance is +yy (IX 4 + 5)
JN	**xxxxxx**	Jog negative (constant speed xxxxxx lines/sec); CCW as long as input active
JP	**xxxxxx**	Jog positive (consistent speed xxxxxx lines/sec); CW as long as input active
SI		Stop impuls edge activation *
SO	**x**	Set output x
CO	**x**	Clear output x
PG	**xx**	Execute program xx

* Stop with a predefined deceleration using ED command (Emergency Deceleration) (see page 14)

Programming



Note

For a fast deceleration in emergencyes (LL, LR, EE, EE1) a predefined deceleration can be set using the ED command, refer to the command set page 14.

The emergency exit function has high priority and is always executed immediately.

As long as EE is active no other function can be executed.

In a LL (Limit switch Left) or LR (Limit switch Right) emergency situation the corresponding opposite direction is only possible.

For all other functions, an already active function is always completed before the next one is executed.

If several functions are active at the same time, the one with the lowest input number is executed first (from top to bottom).

To run a program endlessly (activated via a INPUT, see example in INPUT 7 above) simply keep the selected input active.

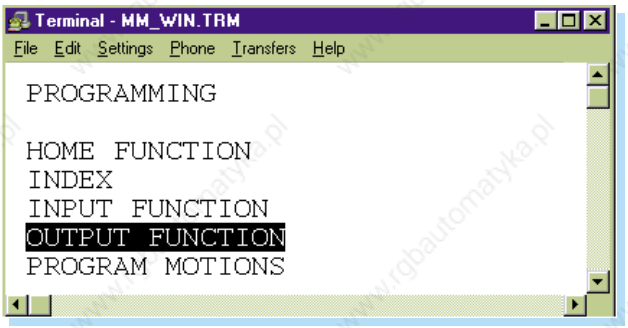
The current program can be interrupted by the interrupt instruction IP.

By disactivating IP the interrupted program immediately continues.

Programming

12.4 Output function

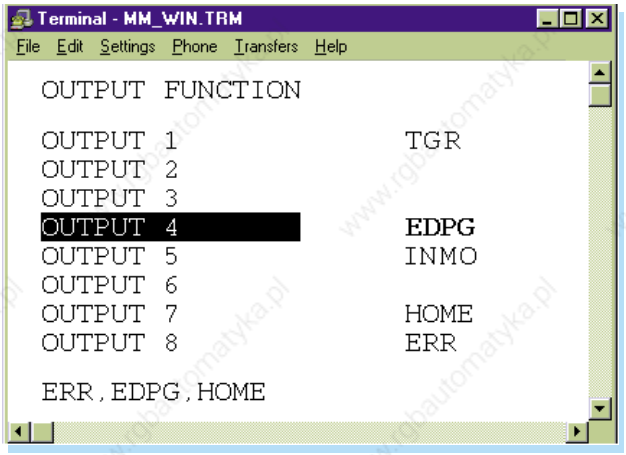
We have seen that functions or instructions can be assigned to inputs. The outputs can also be similarly programmed, but with different possibilities.



The following is a list of the instructions that can be executed directly via the OUTPUT FUNCTION menu:

EDPG	End of program
ERR	Error
HOME	Home has been executed
INMO	In motion, motor running
TGR	Trigger (5 ms) defined with TGU or TGD

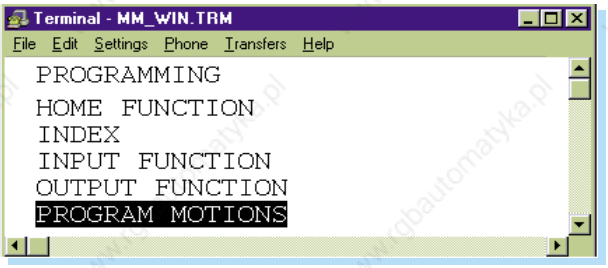
Output are set when status is occurring (open collector to logic 0)



Programming

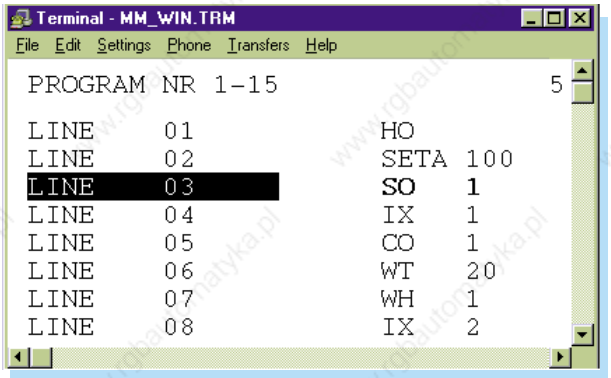
12.5 Program motion

This menu allows us to program up to 15 separate programs with a maximum of 50 lines each. The system selects and executes the instructions line by line. The system considers the program as finished when it comes to an empty line (refere to page 10 for general programming instructions).



The following functions and instructions can be used in the creation of a program:

HO		Execute home function
IX	xx	Execute index number xx
SO	x	Set output number xx
CO	x	Clear output number xx
WT	xxx	Wait time xxx (x 10 ms)
WH	x	Wait for high input number xx
WL	x	Wait for low input number xx
SET[letter]	xxxx	Set loop reference letter (five possibilies, from A to E) and number of repeats xxxx (1-10'000)
JNZ[letter]	xx	Indicate loop reference letter (from A to E). Decrements the loop repeats, whereby if not zero, jump to line xx



Program example

13. Typical program example

Description of the application

For this example, using the MCBL 5004 we want to control the rotation of the Brushless DC-Servomotor type 3564 K 024 B. The speed and position are measured by the HEDS 5540 (2 channel, 500 lines with Z index) encoder.
The motor shaft should make 5 CW rotations (a total of 1'800°) in 5 seconds at a maximum speed of 61,5 rpm. Once the 5 rotations have been completed, the motor should stop for 2 seconds and then make 3 CCW rotations in 0.5 seconds at a maximum speed of 450 rpm.
The second displacement is performed using both relative and absolute positioning.
The home (starting) position is defined using the encoder Z index.

Input sequence

This particular sequence can be subdivided into four distinct phases:

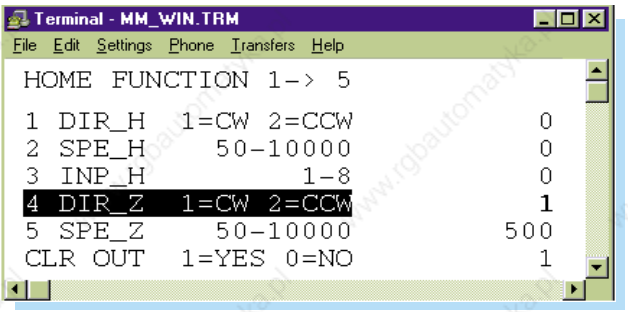
- Starting position (home)
- First displacement (index 1)
- Waiting time WT
- Second displacement (index 2)

How to enter the program data

It is important to remember that the motion controller works using the encoder line information. In other words, the displacement/movement, speed and acceleration parameters are entered in lines, lines/second and lines/second^2 respectively.
As already indicated, the motion controller's internal electronics multiply the encoder resolutions by 4.
The first step is to define the parameters of the various separate phases.

Definition of the home (starting) position

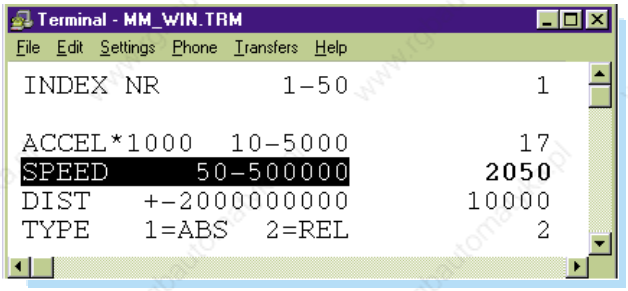
This instruction is given via the HOME FUNCTION menu



If, as in this case, the home function is executed using the encoder Z index, the first 3 lines are set at zero as above. In this example, the Z index is found by rotating the motor shaft CW at a speed of 125 lines/second. Once this position has been reached, all outputs are cleared.

Program example

Entering the first displacement parameters
Go to the INDEX menu



Total distance to be travelled

Equation: $DIST\ [lines] = REV_{shaft}\ [–] \cdot 4 \cdot CPR_{encoder}\ [lines]$

Therefore: $DIST\ [lines] = 5 \cdot 4 \cdot 500 = 10000$ lines

The distance can also be measured using the encoder counter command, EC, in the execute function menu when the servo amplifier is unpowered (7-segment display at zero).

Speed

Equation: $SPEED\ [lines/s] = \frac{SPEED\ [rpm]}{60} \cdot 4 \cdot CPR_{encoder}\ [lines]$

Therefore: $SPEED\ [lines/s] = \frac{61.5\ rpm}{60} \cdot 4 \cdot 500\ lines = 2050\ lines/s$

Acceleration (= deceleration)

Equation: $ACCEL[lines/s^2] = \frac{SPEED\ [lines/s]}{T_{ACCEL}\ [s]}$

Therefore: $ACCEL[lines/s^2] = \frac{2050\ lines/s}{0{,}122\ s} = 16\,803\ lines/s^2 \approx 17\,000\ lines/s^2$

Program example

The acceleration (deceleration) time is calculated as follows:

Equation: $T_{ACCEL}\,[s] = T_{TOT}[s] - \frac{\text{DIST [lines]}}{\text{SPEED [lines/s]}}$ $T_{ACCEL} > 0$

Therefore: $T_{ACCEL}\,[s] = 5\,s - \frac{10000\ \text{lines}}{2050\ \text{lines/s}} = 0{,}122\,s$

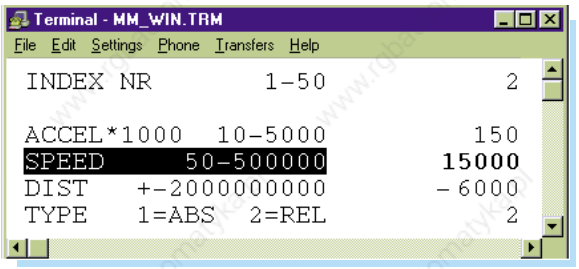
Note: The minimum acceleration accepted by the system is 10000 lines/s^2.

Entering the second displacement parameters

This is also programmed via the INDEX menu in the same way as the first displacement. From the last item on the INDEX menu we have the possibility of defining the two possible types of displacement (relative or absolute). The method for doing this is as described below:

Relative displacement

By relative we mean that the displacement commences from the position of the motor shaft at the time that the instruction is executed.

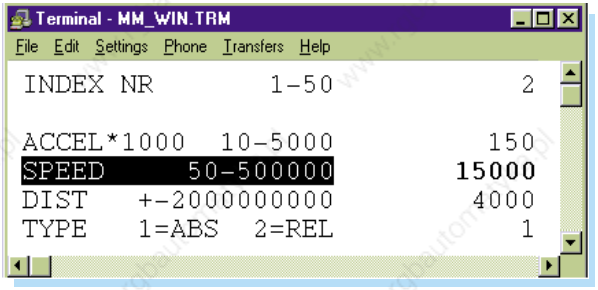


Therefore to make three complete CCW rotations, the motor shaft is instructed to go back 6000 lines (negative) from its actual position.

Program example

Absolute displacement

Absolute displacement takes its reference point from the program's home position.



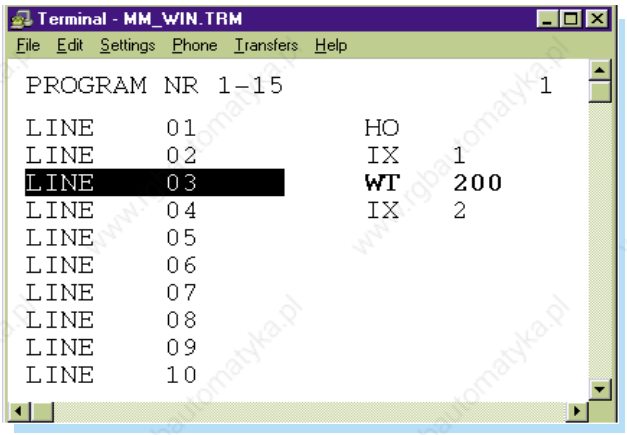
In this case to make three CCW rotations the instruction is given to go to the 4000 line position. In other words, to "return" to the 4 000 line position, the motor has to go back 6 000 lines from its actual position at 10 000 lines.

Waiting time

The 2 second waiting time is simply defined using the WAIT TIME instruction.

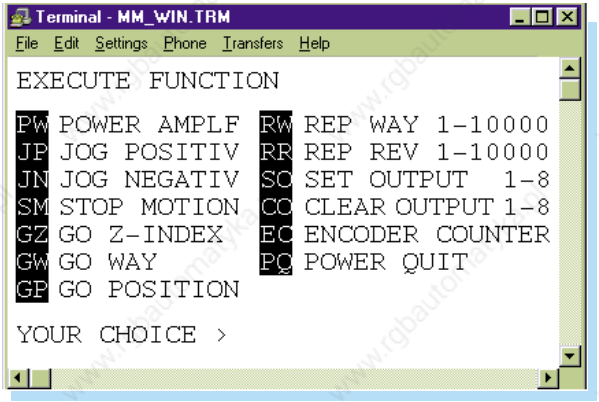
Writing the program

Now that all the phases have been defined, the final instructions are entered in the PROGRAM menu in the correct sequence. (See "input sequence" above).



Program example

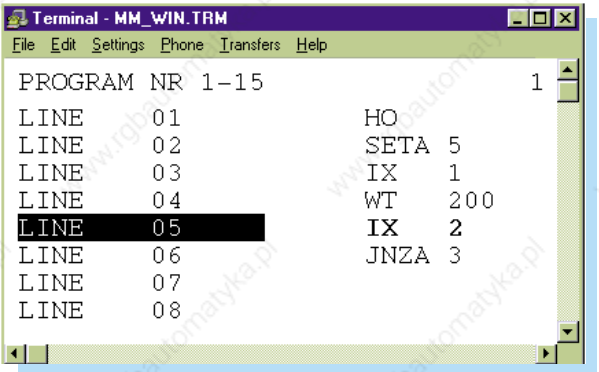
Test



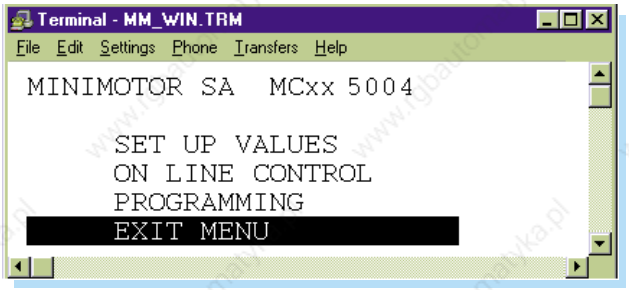
By going to the EXECUTE FUNCTION menu, the whole program (PG1) and the single lines can be tested in practice. For the above example, the lines are tested by entering HO to go to home, IX1 to execute the first displacement, IX2 for the second displacement. Entering PG1 executes the complete program.

Loop

There are two loop instructions, SET[letter] and JNZ[letter], which permit a particular program part to be repeated for a pre-defined number of times. Within any one program a maximum of five loops can be defined (from A to E).
In the following we instruct part of program PG1 (IX1, WT and IX2) to be repeated five times



14. Program loading, saving, verification and installation



These functions are accessed by first exiting the main menu. Press <CR> until the prompt sign appears on a blank screen. At this point, the following procedures can be implemented:

15. Loading an application program from the computer to the motion controller

```
> LAPP <ENTER>
```

Important: **The application already loaded in the motion controller is deleted when executing the LAPP command !! (If the application should be saved, first use the SAPP command prior to using LAPP command, see chapter 16)**

The following message appears and 7-segment display goes to "F":

```
Ready for receiving application
PLEASE TRANSMIT <filename.xxx>
User program terminated ......
```

```
[G] >
```

Select the following functions from the Terminal-MM_WIN.TRM screen:

```
Transfers
Send Text file ...
```

Choose the name of the application to be loaded

```
<filename.xxx>
```

and press [OK]

```
[G] > IL
Loading done
>
```

When ">" reappears, start the motion controller by writing G "space" R followed by two line returns. Alternatively turn the system off and on. The new application is now loaded.

16. Saving the motion controller application on disk

> SAPP <ENTER>

The following message appears:

```
Save application
PLEASE PREPARE RECEIVING - THEN GO WITH <CTRL G>
```

Select the following functions from the Terminal-MM_WIN.TRM screen:

```
Transfers
Receive Text File ...
```

Name the application to be saved

```
<filename.txt>
```

and press [OK]

Then press CTRL G followed by ENTER

```
IL
S2........
S2........
..........
S8........ (Note: S8 ... is the last line)
```

Select STOP. The application is now saved on hard disk/diskette under given file name. Press <CR> to return to the motion controller program.

>

17. Verification of installed version

> VER <ENTER>

The following text will should appear on the screen:

```
FAULHABER GROUP MCxx
Program Version yyy
© Copyright MINIMOTOR SA 2000
```

Loading, saving, verification and installation

18. Installation of new (up-date) basic program

```
> LPROG <ENTER>
```

Important: **With LPROG the existing program MCxx_yyy.S19 will be deleted !!**
The user's application, parameters and data will remain unchanged.

The following message appears and the 7-segment display goes to "F":

```
Ready for receiving program
PLEASE TRANSMIT MCxx_yyy.S19
User program terminated ! ...

[G] >
```

Select the following functions from the Terminal-MM_WIN.TRM screen:

```
Transfers
Send Text File ...
```

Select the file type option to "all data (*.*)" and then choose the file MCxx_yyy.S19 and press OK

```
[G] > IL
File MCxx_yyy
Loading done
>
```

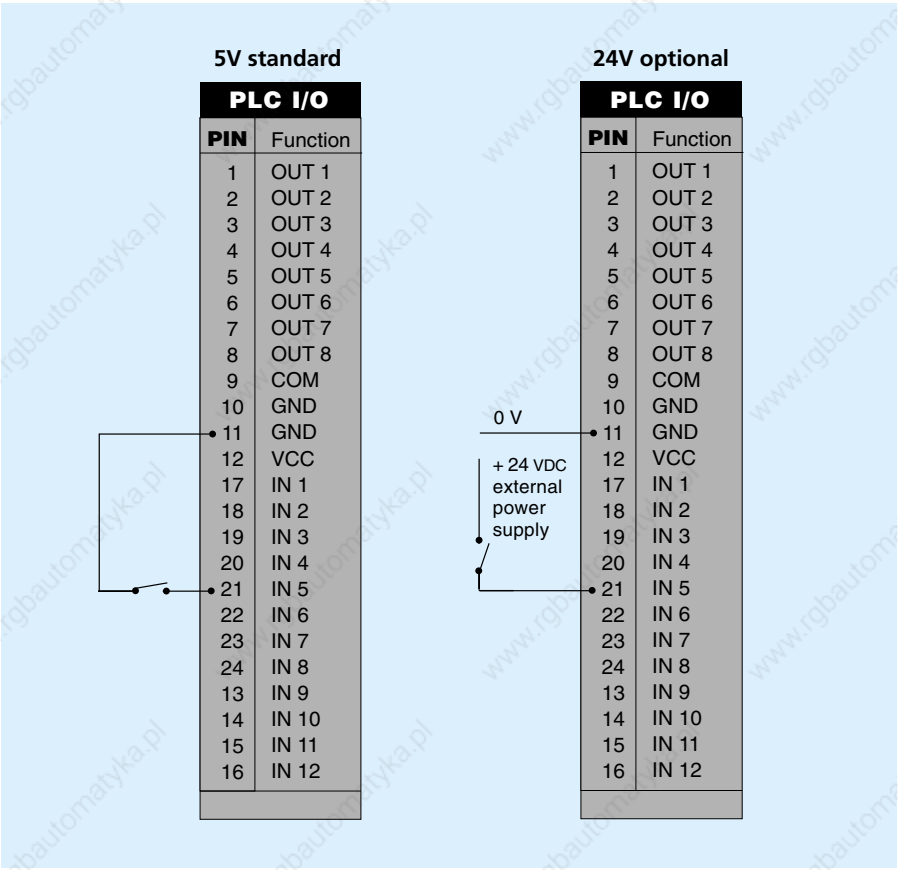
When ">" reappears, start the motion controller by writing G "space" R followed by two line returns. Alternatively turn the system off and on. The new application is now loaded.

Call up program

19. Call up program from normal inputs

The procedure to execute a program or another instruction via the 8 normal inputs is as follows.

- assign the instruction to the desired input via the INPUT FUNCTION menu
- activate the input via an external circuit (see example below)



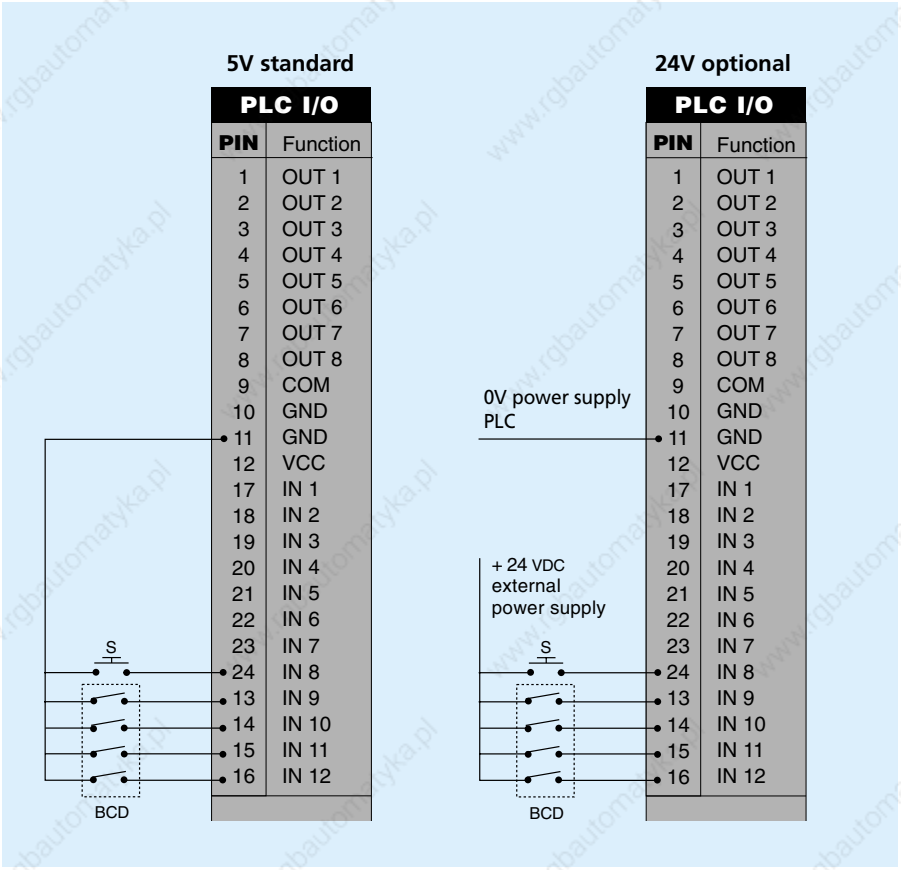
Call up program

20. Call up program from binary coded digital inputs

When the application uses more than 8 digital inputs, the user should call them up via the binary coded digital inputs. In this case, the MODE in SET-UP VALUES menu should be set to 10 (or 12 or 13).

The input lines 9 - 12 are used as binary coded program numbers. The trigger to start the pre-selected program is input line 8. Program number 0 is not used. Therefore:

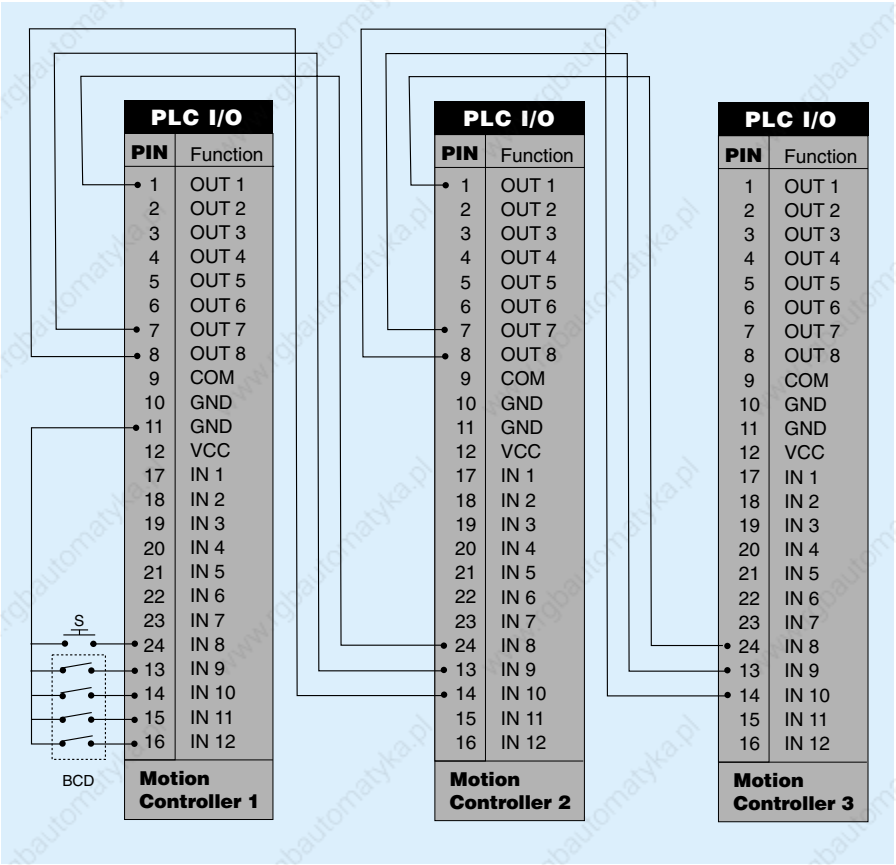
- pre-select program number with binary-switch (numbers 1-15)
- start program with start button S



Call up program

21. Example of sequential multi-axis application

The program number and program start of the sequential follow-on motion controllers are specified through the output of the lined up motion controller.



Advanced functions

22. Analogue input command

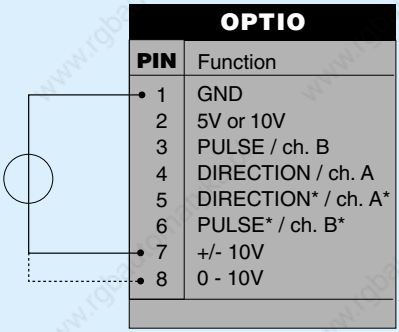
To activate the analogue input command function the MODE parameter in the SET UP VALUES menu must be set to 3 or 13 and the ANALOG FUNCT to:

0 for CW operation
1 for CCW operation
2 for CW and CCW operation

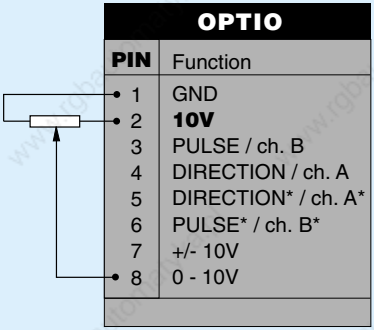
Attention: to start the system the home function (with all the parameters set to 0) must be execute first.

The maximum speed is defined with the **SP** command.
For high dynamics we recommend increasing the **AC** value (**AC** > 2 000 000).

External voltage



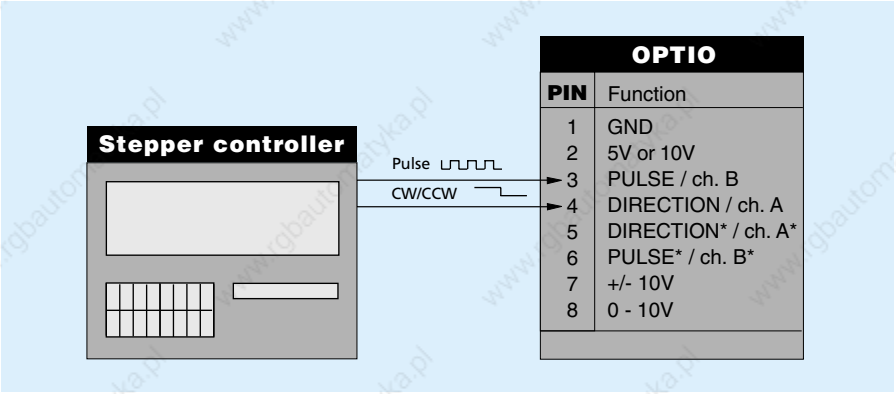
Potentiometer



Advanced functions

23. Stepper motor emulation

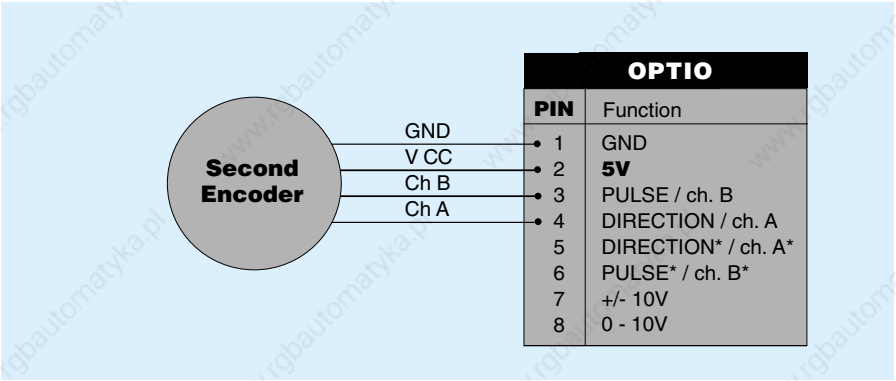
To activate the stepper motor emulation function the MODE parameter in the SET UP VALUES menu must be set to 2 or 12 and the INC PER PULSE according to the application requirements.



24. Second encoder - handwheel

Attention: function only available with optional hardware configuration.

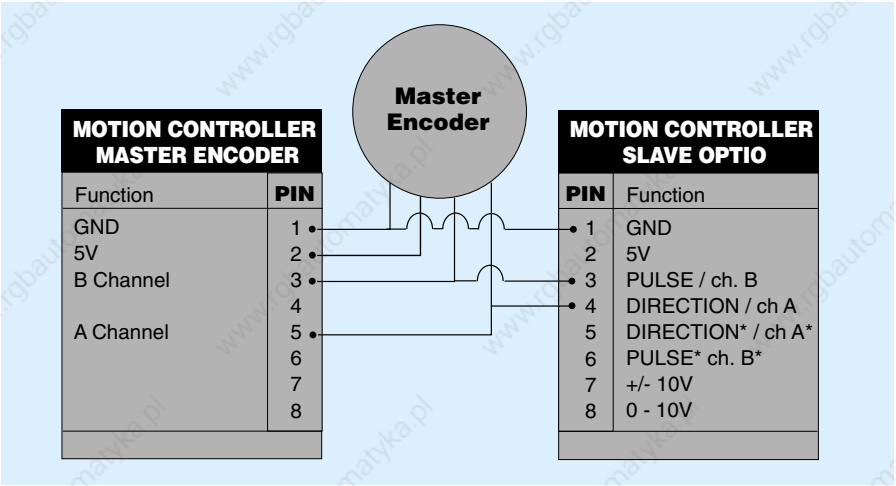
To activate this function the SYNCH RATIO parameter in the SETUP VALUES must be set according to the application requirements. This parameter fixes the speed ratio between the second encoder and the first encoder mounted on the motor shaft.



Advanced functions

25. Second encoder - gearing and master - slave

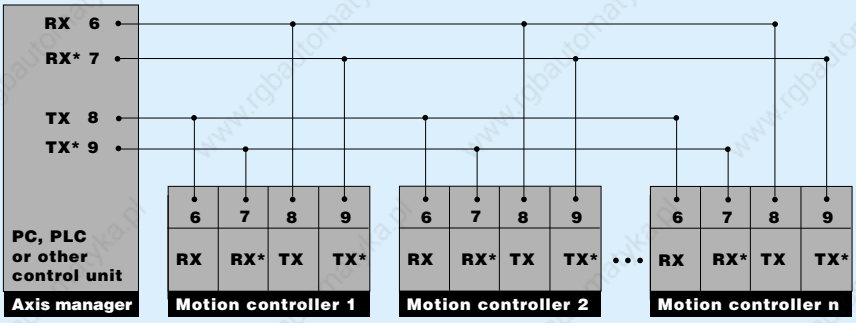
The function is activated as for handwheel operation. In this case the second (master) encoder does not require a DC voltage supply



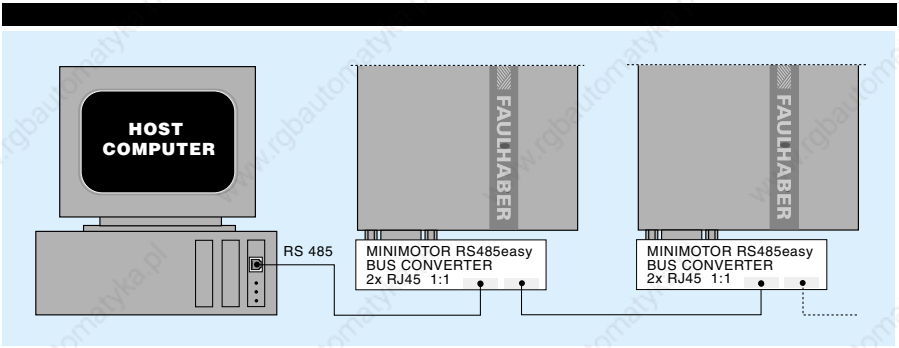
26. RS485easy-Bus

With this feature up to 32 motion controllers can be adressed and controlled by one host computer using a simple RS485 interface. The connection principle is show below.

RS485easy-Bus full duplex



Advanced functions



To simplify the connection Minimotor offers a special RS485 easy-Bus adapter and RJ45 cables.

Cable specification:
Modular RJ45 round shielded cable configuration 1:1. Twisted pairs 1&2, 3&6, 4&5, 7&8

Start-up Procedure
Attention: function only available with software version ≥ 3.00

1) **Assign an adress** (number) to each motion controller (axis).
This operation is made connecting the single motion controller to the RS232 interface and using the CI (card identifier) command.

Example: **CI 5 <CR>** to assign the number **5** to a motion controller.
The number can be checked using the command **CI ? <CR>**

Attention: Each motion controller in the system must have a different number.
Once the number is assigned it is memorized even if the power supply is switched off.
The **CI** value goes from **0-99**. The number **0** is used as a default value for single axis application. For multi-axis operation a number from **1-99** should be used, Number **1** must always (see below).

2) **Realise an RS485 connections** and set the baud rate in the terminal emulator software to **19 200**. To use the RS485 it is necessary to have a RS232/RS485 converter since PCs usually only offer a RS232 interface.

3) **To operate a motion controller** it is first necessary to adress it using the **RI** (request identifier) command.

Example: **RI 5 <CR>**. To make the prompt appear.
If it does not appear check the RS485 connection and baud rate.

Using the **RI 0** command the host computer can control all the axis at the same time.
In this case the echo from Motion Controller number **1** (in multi-axis the number **1** must always be used) will appear on the computer screen.

Trouble-shooting

27. Trouble-shooting

Error messages are shown on the 7-segment display as 2-digit alternating blinking numbers. There are two types of error code: one for input errors (WH wait high or WL wait low) and the other for controller errors (DP deviation position or over-heating).

Error code	Description	Remarks
01 to 12	Waiting for input (low or high)	- Continues if status has been reached or restart with HO, SM or PQ, PW.
50	Deviation position too great	- Difference between the internal calculated position and actual motor position greater than the number of increments defined in DP (deviation position).
60	Power stage over-heating	- > 80° C detected by the temperature sensor.
61	Power supply over voltage	- Power supply voltage or retarding energy on ballast circuit to high.
62	Ballast circuit active too long	- If the ballast circuit is active for more than 5 seconds the power stage is switched off.

28. Operating system error

If the message “user program corrupted” appears on the screen, the operating system has to be turned off and re-started manually.
This is done as follows:

- **remove the motion controller cover and set the S1 switch from OFF to ON**
- **re-load the MCxx_yyy.S19 program**
- **turn the S1 switch from ON to the OFF position and replace the cover**

The system can now be switched on normally.

Notice of use

29. General usage instruction

Power supply and fuse

Any unstabilised DC power supply voltage within the motion controllers range:

- **MCBL 5004** 12 V ≤ Vm ≤ 50V
- **MCDC 5004** 12 V ≤ Vm ≤ 50V

may be used, although it is advisable to keep this voltage as low as possible in order to minimize the EMI noise. Thus the optimum power supply voltage is given by the following equation:

$$Vm\ [V] = 5V + R\ [\Omega] \cdot I_{max}\ [A] + k_E\ [V/rpm] \cdot n_{max}\ [rpm]$$

Where:
- R = motor terminal resistance
- k_E = motor back-EMF constant
- I_{max} = max. requested motor current for acceleration (= I_{PEAK})
- n_{max} = max. motor speed reach in the application Both motion controllers are provided with an internal fuse.

Braking energy
When decelerating the motor, brake energy is developed. This energy increases the motion controller voltage supply. Therefore the motion controllers supplied with an internal ballast circuit which converts this energy into heat.

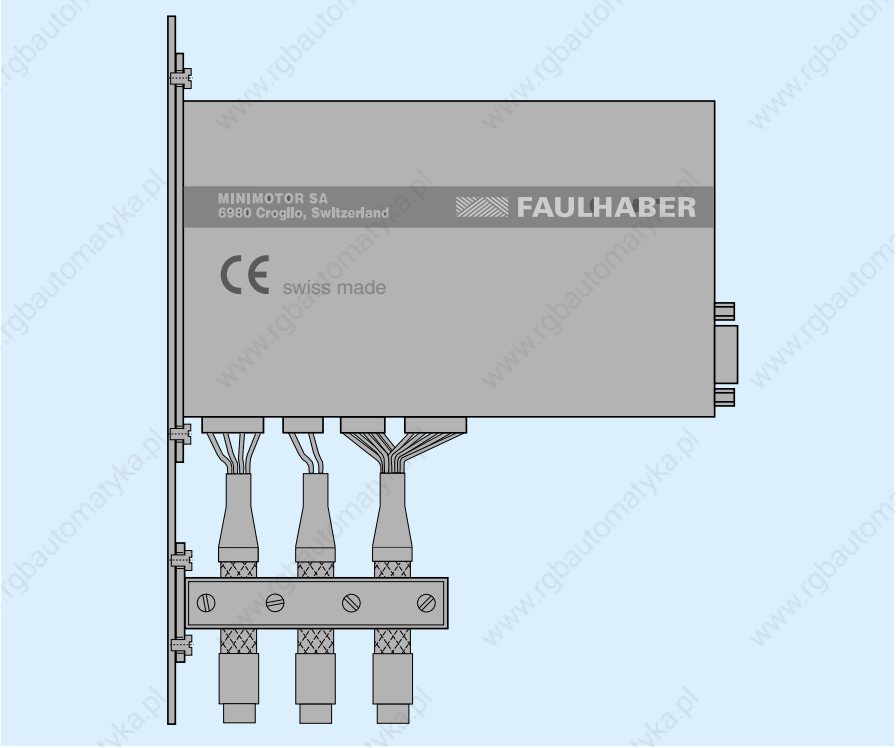
Wiring
A well known disadvantage of PWM (pulse width modulation), is that it generates a lot of interference. In order to reduce the effect of the interference there are some basic rules to follow:

- **Use wires as short as possible**
- **Avoid running signal wires (logical and analog signal) in close proximity to power lead wires (power supply and motor power leads)**
- **Use shielded wires**

Electro-magnetic compatibility

30. Electro-magnetic compatibility

EMC-approved shielding should be made according to the diagram below:



The motion controllers conform to the 89/336/CEE directive on electro-magnetic compatibility. For their evaluation, the following norms have been considered as far as applicable:

EN 50081-2 for emissions and EN 50082-2 for immunity

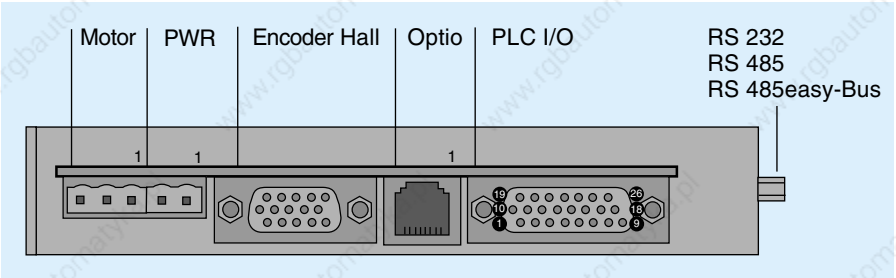
In order to respect these norms after installation in the application, the following must be observed:

- The installation instructions in the user's manuals must be respected.
- Connections must be made using shielded cables.
- All the components of the system must conform to the cited norms.
- Metal parts and shielding must be connected to common earth.
- The system must be supplied with a voltage corresponding to the specifications.

Hardware

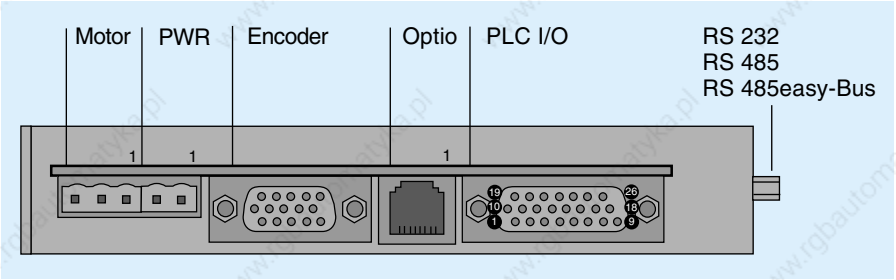
31. Hardware

Connector layout for MCBL 5004



Connect	Pin	Function	Type
Motor	3	Motor phases	WAGO Multiconnector 5,0mm
PWR	2	Power supply	WAGO Multiconnector 5,0mm
Encoder Hall	15	Encoder input, encoder 1phases, Hall effect sensors	D-SUB High-Density
Optio	8	Pulse/dir, analogue, encoder 2	Modular RJ45
PLC I/O	26	12 inputs / 8 outputs	D-SUB High-Density
RS232/ RS485	9	Serial interface RS232/RS485/ RS485easy-Bus	D-SUB normal

Connector layout for MCDC 5004



Connect	Pin	Function	Type
Motor	3	Motor terminals Power supply	WAGO Multiconnector 5,0mm
PWR	2	Encoder input, encoder 1	WAGO Multiconnector 5,0mm
Encoder	15	Pulse/dir, analogue, encoder 2	D-SUB High-Density
Optio	8	12 inputs / 8 outputs	Modular RJ45
PLC I/O	26	Serial interface RS232/RS485/	D-SUB High-Density
RS232/ RS485	9	RS485easy-Bus	D-SUB normal

PIN configuration

32. PIN configuration

Serial interface RS 232 or RS 485, 9 POLE D-SUB

Pin 1	NC	Not connected
Pin 2	RS232	Receiver Rx
Pin 3	RS232	Transmitter Tx
Pin 4	NC	Not connected
Pin 5	RS232	GND
Pin 6	RS485	Receiver $\overline{R}$
Pin 7	RS485	Receiver R
Pin 8	RS485	Transmitter $\overline{T}$
Pin 9	RS485	Transmitter T

RS232 set up

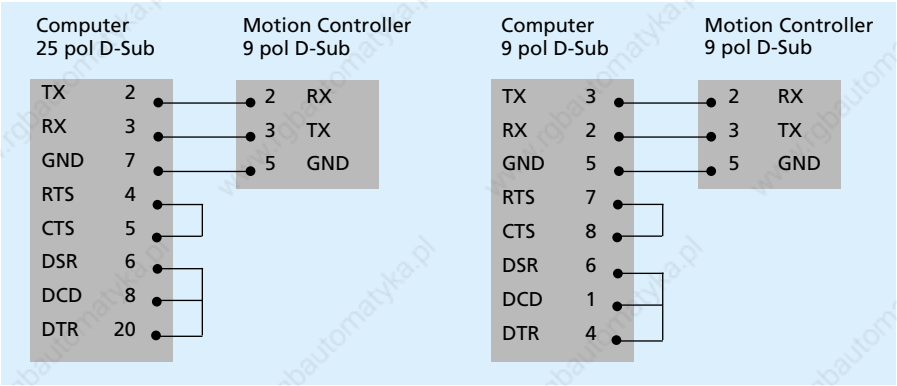
Set the baud rate RS232 via 6-bit CONFIG switch S1 (remove the cover and you find the small 6 bit SMD multiswitch)

data	8bit
stop bit	1
parity	no

bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	Function
x	x	x	OFF	OFF	x	RS232 9 600 baud (default)
x	x	x	OFF	ON	x	RS232 2 400 baud
x	x	x	ON	OFF	x	RS232 4 800 baud
x	x	x	ON	ON	x	RS232 19 200 baud

By turning system off and back on the new baud rate will be activated.

RS232 electrical connection



PIN configuration

RS485 set up
Setting of the baud rate RS485 over 6-bit CONFIG switch S1: data 8bit, stop bit 1, parity no

bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	Function
x	OFF	OFF	x	x	x	RS485 19 200 baud (default)
x	OFF	ON	x	x	x	RS485 9 600 baud
x	ON	OFF	x	x	x	RS485 38 400 baud
x	ON	ON	x	x	x	RS485 free

By turning system off and back on the new baud rate will be activated.

Bus RS485easy, MODULAR RJ45

Pin 1	NC	
Pin 2	NC	
Pin 3	NC	
Pin 4	RS485	Receiver $\overline{R}$
Pin 5	RS485	Receiver R
Pin 6	NC	
Pin 7	RS485	Transmitter $\overline{T}$
Pin 8	RS485	Transmitter T

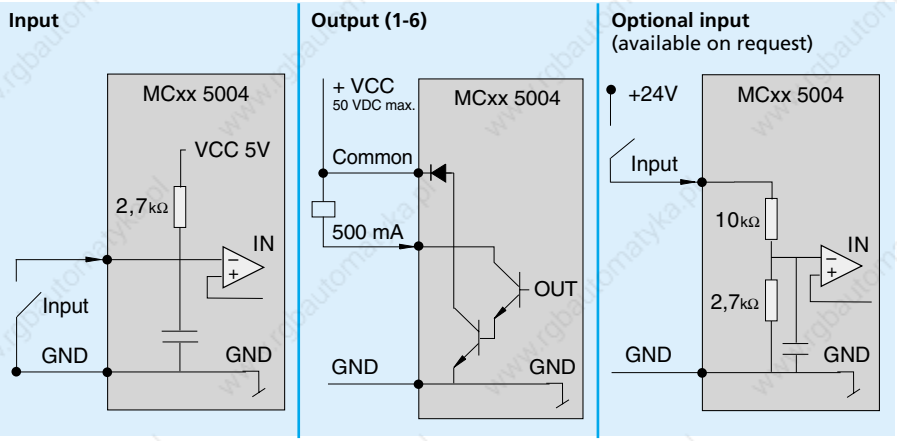
PLC 12 Input / 8 Output available to the user

Pin 1	Output 1	Active low, open collector 50 V / 500 mA on GND, free-wheeling diode
Pin 2	Output 2	
Pin 3	Output 3	
Pin 4	Output 4	
Pin 5	Output 5	
Pin 6	Output 6	
Pin 7	Output 7	Active low 0 V / 50 mA, high 5 V / 50 mA
Pin 8	Output 8	Active low 0 V / 50 mA, high 5 V / 50 mA
Pin 9	COMMON	Joint cathodes of output free wheeling diodes 1 - 6
Pin 10	GND	2A
Pin 11	GND	2A
Pin 12	VCC	5 V /250 mA
Pin 17	Input 1	Pull up 2,7 k on VCC 5 V standard or optional 24 V (for PNP sensors)
Pin 18	Input 2	
Pin 19	Input 3	
Pin 20	Input 4	
Pin 21	Input 5	
Pin 22	Input 6	
Pin 23	Input 7	
Pin 24	Input 8 [1]	
Pin 25	GND	2A
Pin 26	5V	250 mA
Pin 13	Input 9 [2]	Pull up 2,7 k on VCC 5 V standard or optional 24 V *B*it for BCD program decoder.
Pin 14	Input 10 [2]	
Pin 15	Input 11 [2]	
Pin 16	Input 12 [2]	

[1] Program start trigger with BCD coded input (MODE = 10)
[2] BCD coded input for program,1-15, selection (MODE = 10)

PIN configuration

Input and output internal electrical circuit



Encoder Hall, 15 Pole HD DSUB

Pin 1	GND	GND for both, encoder and Hall
Pin 2	5V Encoder	150 mA
Pin 3	Encoder B	Pull up 2,4k to 5V, differential input 26LS32
Pin 4	Encoder $\overline{B}$	middle level:pull up 2,4k to 5V, pull down 2k, differential input 26LS32
Pin 5	Encoder A	Pull up 2,4k to 5V, differential input 26LS32
Pin 6	Encoder $\overline{A}$	middle level:pull up 2,4k to 5V, pull down 2k, differential input 26LS32
Pin 7	Encoder Z	Pull up 2,4k to 5V, differential input 26LS32
Pin 8	Encoder $\overline{Z}$	middle level:pull up 2,4k to 5V, pull down 2k, differential input 26LS32
Pin 9	Hall A	Pull up 2,4k to 5V, differential input 26LS32
Pin 10	Hall $\overline{A}$	middle level:pull up 2,4k to 5V, pull down 2k, differential input 26LS32
Pin 11	Hall B	Pull up 2,4k to 5V, differential input 26LS32
Pin 12	Hall $\overline{B}$	middle level:pull up 2,4k to 5V, pull down 2k, differential input 26LS32
Pin 13	Hall C	Pull up 2,4k to 5V, differential input 26LS32
Pin 14	Hall $\overline{C}$	middle level:pull up 2,4k to 5V, pull down 2k, differential input 26LS32
Pin 15	5V Hall	150 mA

Power supply

Pin 1	GND	
Pin 2	POWER	MCDC 12 - 50V (over voltage limited with protection diode)
		MCBL 12 - 50V (over voltage limited with protection diode)

Motor connector

		DC motor	**BL motor**
Pin 1		Motor (-)	Phase A
Pin 2		Motor (+)	Phase B
Pin 3		NC	Phase C

PIN configuration

Optional function, 8 pole modular RJ45

Pin 1	GND	GND internal
Pin 2	10V (5V)	10V default voltage (5V with option second Encoder on request)
Pin 3	Pulse / ch. B	pull up 2,4k to 5V, differential input 26LS32
Pin 4	Direction / ch. A	pull up 2,4k to 5V, differential input 26LS32
Pin 5	Direction / ch. Ā	middle level: Pull up 2,4k to 5V, pull down 2k, differential input 26LS32
Pin 6	Pulse / ch. B̄	middle level: Pull up 2,4k to 5V, pull down 2k, differential input 26LS32
Pin 7	+/- 10V	analogue input reference, range +/- 10V
Pin 8	0- 10V	analogue input reference, range 0-10V

Note

Starter Kit's

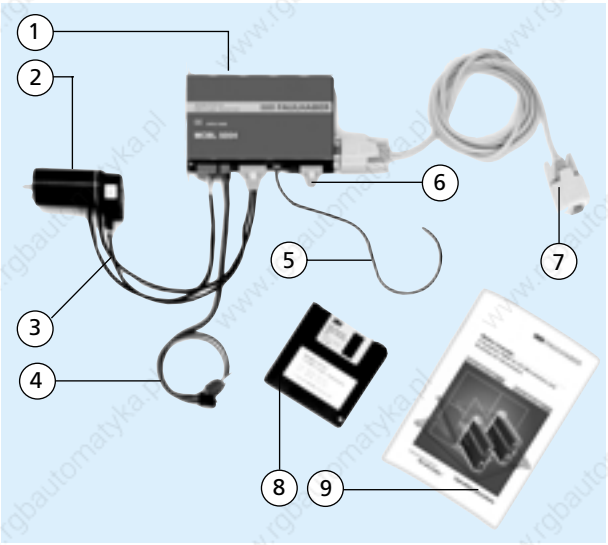
4 attractive starter kits are offered. These starters kits are complete with cables and connectors pre-configurated, ready to operate.

Starter Kit #1
Motion Controller MCBL 5004 with Brushless DC-Servomotor 2444 S 024 B with Encoder HEDS 5540A-500L/3CH

Starter Kit #2
Motion Controller MCBL 5004 with Brushless DC-Servomotor 3564 K 024 B with Encoder HEDS 5540A-500L/3CH

Starter Kit #3
Motion Controller MCDC 5004 with DC-Micromotor Type 2342 S 024 CR with Encoder HEDS 5540A-500L/3CH

Starter Kit #4
Motion Controller MCDC 5004 with DC-Micromotor Type 3557 K 024 C with Encoder HEDS 5540A-500L/3CH



1. Motion Controller MCBL 5004
2. DC-Motor
3. HEDS Encoder cable
4. Power cable
5. Connector with cable for advanced function
6. Connector for digital I/O
7. RS232 computer cable (DB9-DB9)
8. Software
9. Instructions Manual

FAULHABER

The FAULHABER Group:

DR. FRITZ FAULHABER GMBH & CO. KG
Daimlerstraße 23
71101 Schönaich · Germany
Tel.: +49 (0)7031/638-0
Fax: +49 (0)7031/638-100
Email: info@faulhaber.de
www.faulhaber.de

MINIMOTOR SA
6980 Croglio · Switzerland
Tel.: +41 (0)91 611 31 00
Fax: +41 (0)91 611 31 10
Email: info@minimotor.ch
www.minimotor.ch

MicroMo Electronics, Inc.
14881 Evergreen Avenue
Clearwater · FL 33762-3008 · USA
Phone: +1 (727) 572-0131
Fax: +1 (727) 573-5918
Toll-Free: (800) 807-9166
Email: info@micromo.com
www.micromo.com

MA15006, english, 2. issue, 04.04.2002
Version 3.40