LabVIEW® -VI MCC

phytron

Virtual Instruments
for MCC Control Units

Manual 1253-A001 GB

customized solutions
in motion

LabVIEW® Virtual Instruments

for MCC Control Units

TRANSLATION OF THE GERMAN ORIGINAL MANUAL

Phytron GmbH

Industriestraße 12

82194 Gröbenzell, Germany

Tel.: +49 8142/503-0

Fax: +49 8142/503-190

Every possible care has been taken to ensure the accuracy of this technical manual. All information contained in this manual is correct to the best of our knowledge and belief but cannot be guaranteed. Furthermore we reserve the right to make improvements and enhancements to the manual and / or the devices described herein without prior notification. You find the updated version of this manual on the website of www.phytron.de.

We appreciate suggestions and criticisms for further improvement. Please send your comments to the following email address: doku@phytron.de

Contents

1 What is LabVIEW® ?

This manual describes the LabVIEW graphical programming language from National Instruments, which uses icons to create the application.
LabVIEW stands for „Laboratory Virtual Instrument Engineering Workbench“.
LabVIEW programs are called Virtual Instruments or VIs. These VIs content the front panel, the user interface and the block diagram. The block diagram is a graphical program code, which is compiled like other high level programming languages.
This manual describes the LabVIEW use for the Phytron controllers MCC-1, MCC-2 and MCC-2 LIN.

LabVIEW is a Trademark of National Instruments Corporation. See chap. 7.

1.1 Requirements

A well-trained LabVIEW user is expected for using the Phytron MCC controller VIs. He knows the programming environment. Basic knowledge in programming like data types, loops etc. are required.

The MCC VIs are built for LabVIEW 8.0 or higher.

1.2 Extent of Supply

The package contents two LabVIEW libraries:

• MCC.llb	Encloses the VIs to be applied for the MCC controllers
• Demo-MCC.llb	Demo-applications which make the VIs clear

2 To Consider Before Installation

Read all manuals very carefully before installing and operating.
Observe the safety instructions in the following chapter!

2.1 Qualified Personnel

Design, installation and operation of systems may only be performed by qualified and trained personnel.

These persons should be able to recognize and handle risks emerging from electrical, mechanical or electronic system parts.

The qualified personnel must know the content of this manual and be able to understand all documents belonging to the product. Safety instructions are to be planned.

The trained personnel must know all valid standards, regulations and rules for the accident prevention, which are necessary for working with the product.

WARNING

Without proper training and qualifications damages to devices and injury might result!

2.2 Warning Regarding Use of Software Products

1. In any application the reliability of operation of the software products can be impaired by adverse factors, e. g. differences in electrical power supply or, computer hardware malfunctions.
   To avoid damage by system failures the user must take appropriate safety measures, including back-up or shut down mechanism.

2. Malfunctions are possible while programming the instruction codes – e. g. sudden run of a connected motor, braking etc.
   Please test the program flow step by step!

3. Each end user system is customized and differs from the testing platform. Therefore the user or application designer is responsible for verifying and validating the suitability of the application.

3 General VI Description

LabVIEW programs are called virtual instruments or VIs.
Every VI can be used as a stand-alone program or as a subroutine called sub VI.

The user defines the data flow by connecting the VIs with connection lines (wires).
The VI executes when all input data are available.
If the complete VI is finished, the results will be on the outputs. The sequence of the execution is defined by the dependency of the data. There is no predefined sequence (e. g. from right to left).

4 Description of the MCC.llb

4.1 General

Inputs and outputs which are the same for all VIs in the library:

Name	I/O	Meaning
VISA resource name in	Input	Transfer of the interface parameters
Error in	Input	Input of the error clusters
VISA resource name out	Output	Display of the interface parameters
Error out	Output	Output of the error cluster

Inputs and outputs have the same function. They are only described once.

A cluster is the bundling of different data types in LabVIEW. It can be used as an input or output.

4.2 AD-MCC.VI

Read the A/D value of the MCC.
This VI provides the current A/D converter value as a 16-bit unit to the output while executing.

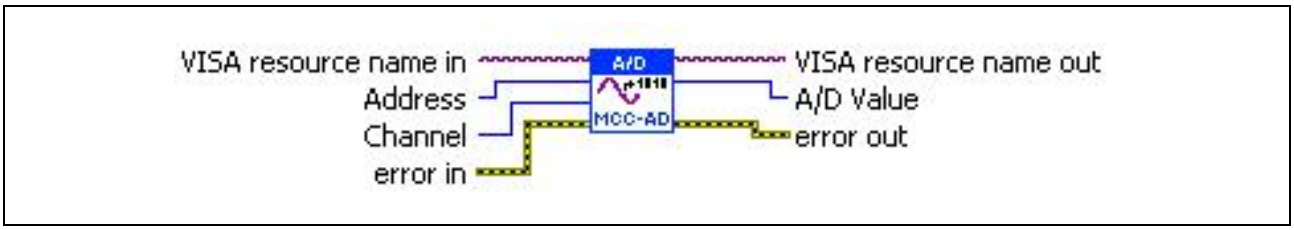


Fig. 1: AD-MCC.vi

Name	I/O	Meaning
Address	Input	Adjusted address at the controller (0 -15 8-bit unit)
Channel	Input	Channel of the A/D converter, which should be read (1 or 2, 8-bit unit)
A/D Value	Output	A/D value in increments (0 – 1023, 10-bit)

4.3 COMM-MCC.vi

This VI is internal used by other VIs. It should not be applied for programming user specific applications.

4.4 Counter-MCC.vi

This VI reads the selected axis counter.
It displays the counter value of the selected axis by reading the parameter 20 (P20).
You´ll find the parameter description in chapter 6.

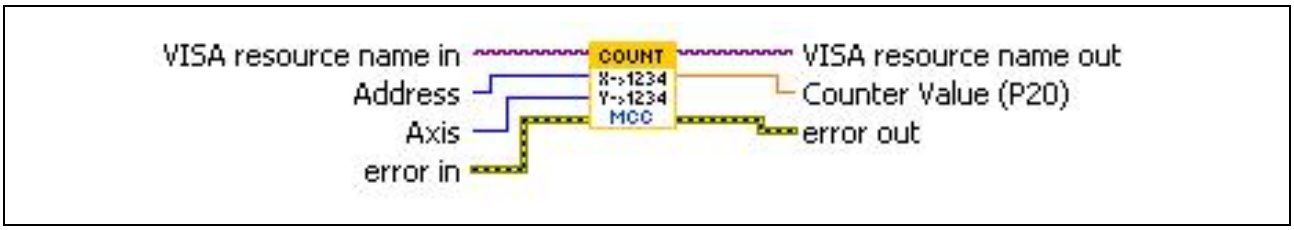


Fig. 2: Counter-MCC.vi

Name	I/O	Meaning
Address	Input	Adjusted address at the controller (0 -15 8-bit unit)
Axis	Input	The counter of the axis should be read (1 or 2, 8-bit unit)
Counter Value	Output	Counter value of the axis (Double)

4.5 Directmode-MCC.vi

An instruction is transmitted to the controller.
The VI transmits the string at the input Send String to the controller and picks up the answer from the controller.
For detailed description of the controller commands see chap.6.

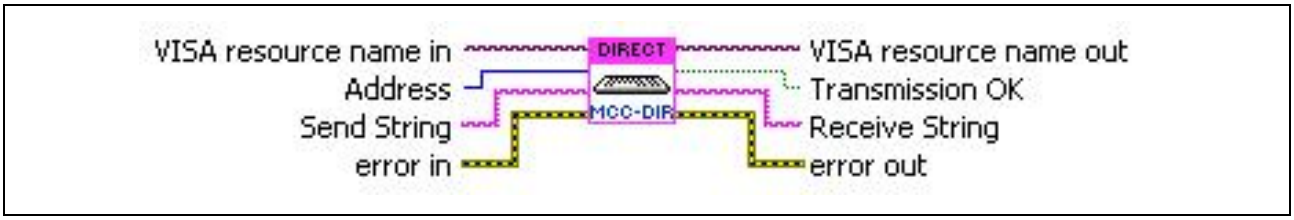


Fig. 3: Directmode-MCC.vi

Name	I/O	Meaning
Address	Input	Adjusted address at the controller (0 -15 8-bit unit)
Send String	Input	Command, which is transmitted to the controller (e. g. X+1000 corresponds to drive 1000 steps)
Transmission OK	Output	True, if the controller acknowledged the command (ACK) False, if the command was invalid (NAK)
Receive String	Output	Answer string of the controller (without control character and ACK) It´s empty when the commands have no response.

4.6 Drive-MCC.vi

This VI sends drive instructions to the MCC.
This functional module reads the values adjusted in the cluster and generates a drive instruction for the MCC.

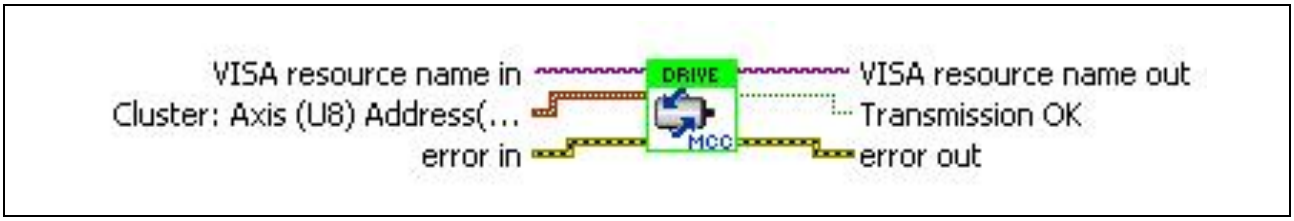


Fig. 4: Drive-MCC.vi

Name	I/O	Meaning
Cluster: Axis Address Position Mode Distance	Input	It has the following data types: • Axis (8-bit unit): axis, where the drive instruction is displayed (1 or 2) • Address (8-bit unit): Controller address (0-15) • Position Mode (ENUM): the following adjustments are available: ○ Relative, generates and transmits a relative drive instruction ○ Absolute, generates and transmits an absolute drive instruction ○ Initialization Plus, generates and transmits an initialization toward the positive direction ○ Initialization Minus, generates and transmits an initialization toward the negative direction ○ Free Run Plus, starts a free run toward the positive direction ○ Free Run Minus, starts a free run toward the negative direction • Distance (DLB): this input is used as a distance for relative and absolute drive instructions
Transmission OK	Output	True: the controller accepts the instruction False: invalid command

4.7 Encoder-MCC.vi

The encoder counter reads the selected axis.
Parameter 22 (P22) is read out for the corresponding axis.
You´ll find the description of the parameters in chapter 6 parameters.

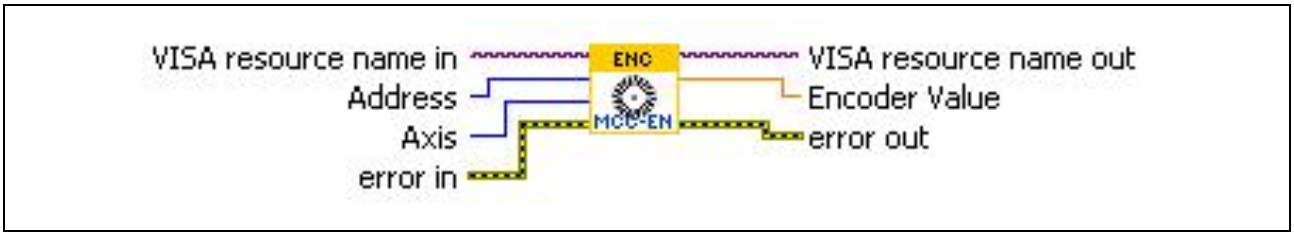


Fig. 5: Encoder-MCC.vi

Name	I/O	Meaning
Address	Input	Adjusted address at the controller (0 -15 8-bit unit)
Axis	Input	The count of the axis is to be read (1 or 2, 8-bit unit)
Encoder Value	Output	Read Encoder value of the axis (Double)

4.8 Init-MCC.vi

This VI displays the initiator status.
The MCC initiator status is imported and displayed as Boolean Cluster.

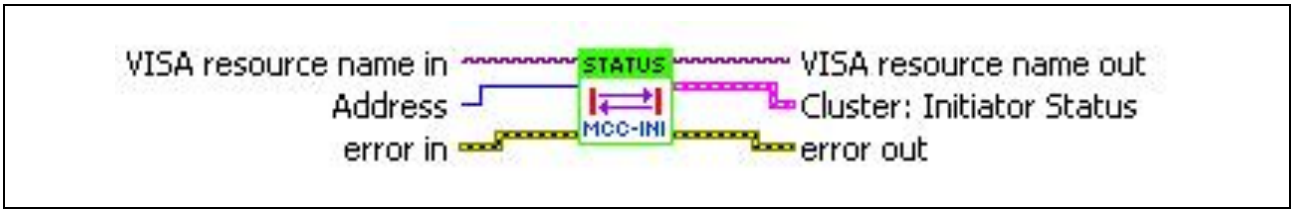


Fig. 6: Init-MCC.vi

Name	I/O	Meaning
Address	Input	Adjusted address at the controller (0 -15 8-bit unit)
Cluster: Initiator Status	Output	The Initiator Status consists of four elements (BOOL) • Axis 1+, activated = TRUE, free = FALSE • Axis 1-, activated = TRUE, free = FALSE • Axis 2+, activated = TRUE, free = FALSE • Axis 2 -, activated = TRUE, free = FALSE

4.9 Input-MCC.vi

Reads the MCC input status.
The status of the MCC is displayed as a Boolean Cluster.

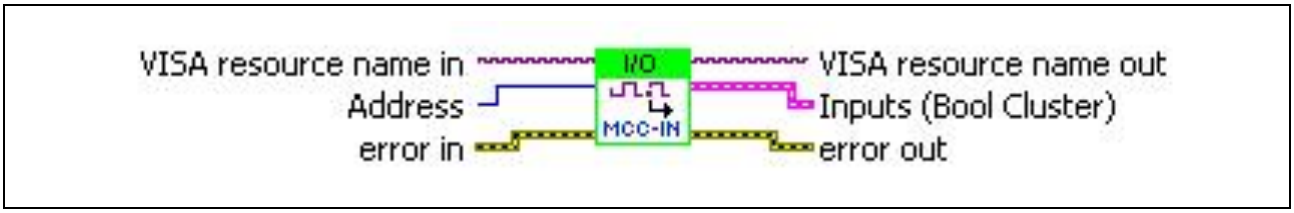


Fig. 7: Input-MCC.vi

Name	I/O	Meaning
Address	Input	Adjusted address at the controller (0 -15 8-bit unit)
Cluster: Inputs	Output	Inputs, consist of eight elements (Boolean) TRUE = Input High FALSE = Input Low

4.10 Output-MCC.vi

This VI sets the outputs at the MCC.
The status at the input is set as output status during execution.

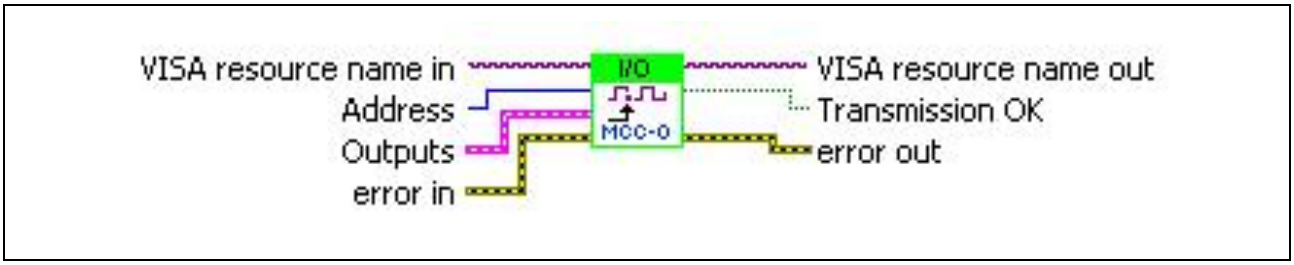


Fig. 8: Output-MCC.vi

Name	I/O	Meaning
Address	Input	Adjusted address at the controller (0 -15 8-bit unit)
Cluster: Outputs	Input	Outputs, consist of eight elements (Boolean) TRUE = Output High FALSE = Output Low
Transmission OK	Output	True: the controller accepts the instruction False: invalid command

4.11 Parameter-MCC.vi

This VI reads or sets the MCC parameters.
The MCC reads or transmits the parameter which is adjusted in the Parameter Number.

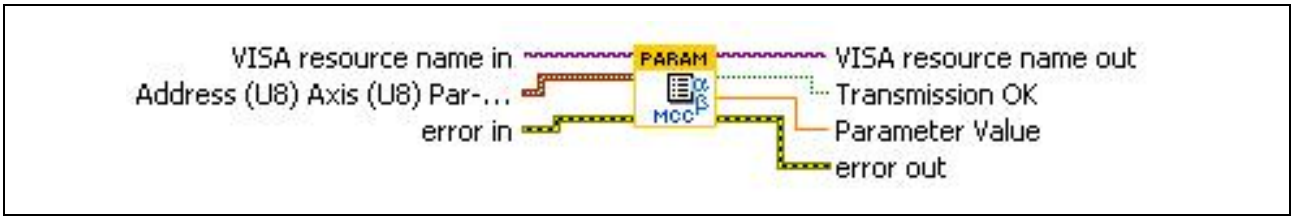


Fig. 9: Parameter-MCC.vi

Name	I/O	Meaning
Cluster: Address Axis Parameter Number Parameter Value Read / Write	Input	It has the following data types: • Address (8-bit unit): controller address (0-15) • Axis (8-bit unit): Axis, of which the parameter is read/written (1 or 2) • Parameter Number (8-bit unit): Parameter number to be read or written • Parameter Value (Double): Parameter value to be written. Only with the choice ‚write'! • Read / Write (Enum): Contains the items Read and Write. Read: Parameter is read and is displayed at the Parameter Value output. Write: Parameter is written with the value from the Parameter Value input.
Transmission OK	Output	True: the controller accepts the instruction False: invalid command(Boolean)
Parameter Value	Output	The Read function displays the parameter value of the selected parameter (Double).

4.12 Register-MCC.vi

Reads or sets the MCC register.
The register set in Register Number is read or transmitted to the MCC.

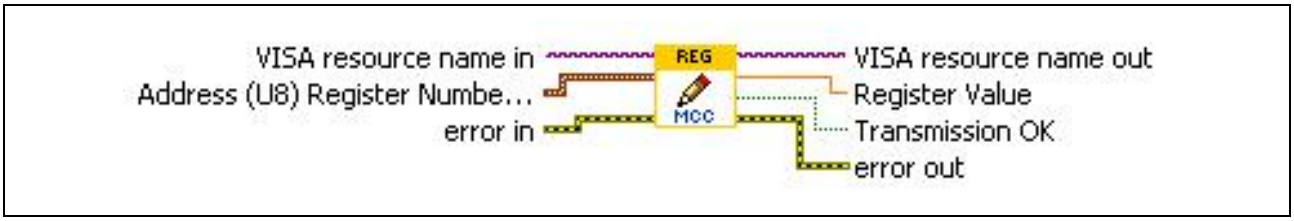


Fig. 10: Register-MCC.vi

Name	I/O	Meaning
Cluster: Address Register Number Register Value Read / Write	Input	It has the following data types: • Address (8-bit unit): controller address (0-15) • Register Number (16-bit unit): Register number, to be read or written. • Register Value (Double): Register value to be written. Only with the choice ‚write'! • Read / Write (Enum): Contains the items Read and Write. Read: Parameter is read and is displayed at the Register Value output. Write: Parameter is written with the value from the Register Value input.
Transmission OK	Output	True: the controller accepts the instruction False: invalid command (Boolean)
Register Value	Output	The Read function displays the register value of the selected parameter (Double).

4.13 Status-MCC.vi

Reads the MCC status and displays the result as Boolean Cluster.

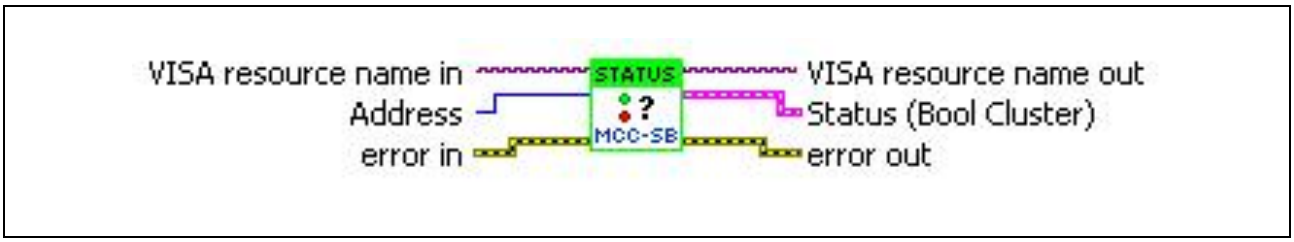


Fig. 11: Status-MCC.vi

Name	I/O	Meaning
Address	Input	Adjusted address at the controller (0 -15 8-bit unit)
Cluster: Status	Output	Displays the MCC status as Boolean cluster. The status is read binary. You´ll find further information in the MINILOG Programming Manual for MCC available under the SB instruction.

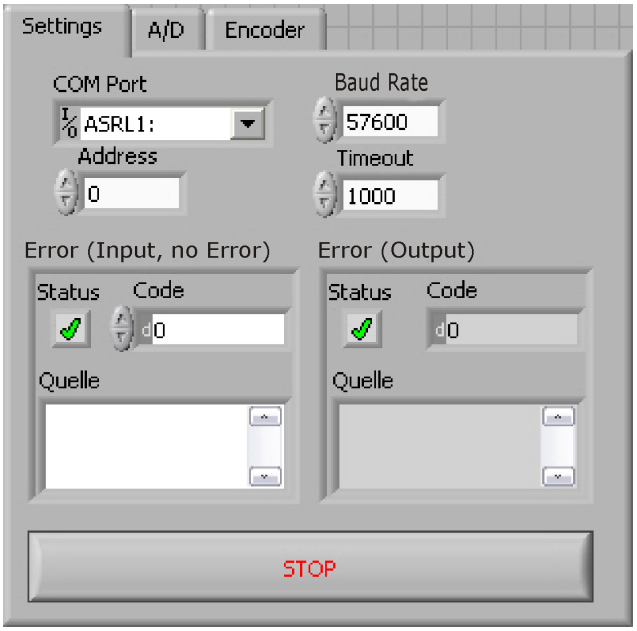
5 Demo-MCC.vi

5.1 General Description

The VI demos use the VIs from the MCC.llb. These are demos of register cards with different tabs.
The first register card (Settings) is equal in all demos and only described once.

Settings of the register card:

Here general settings for the interface are made:

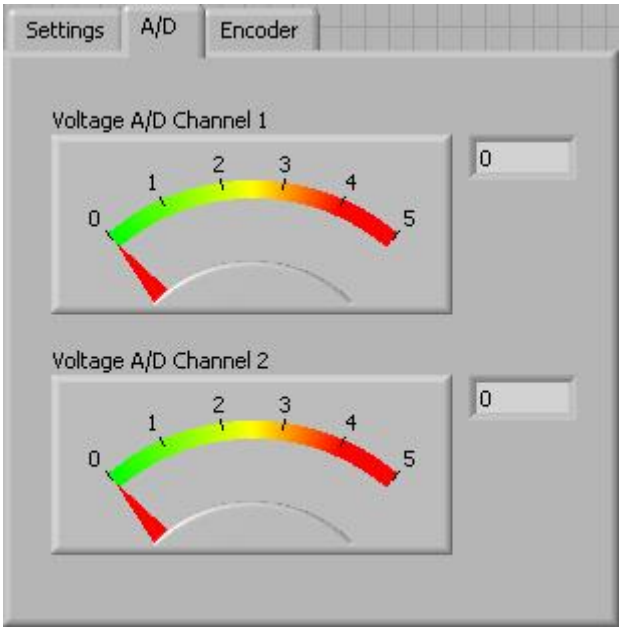


COM Port	Configures the used interface. In this case the serial interface COM4.
Address	The Address adjusted at the controller
Baud Rate	Configuration of the controller´s baud rate, e. g. 57 600.
Timeout	The time waiting for an answer. If there is no answer in the specified time, VISA-VI outputs an error.
Error	This output displays an error message, which occurs during communication.
Stop	Stops the program.

5.2 DEMO A/D-ENC MCC A/D Inputs and Encoder Timer Reading

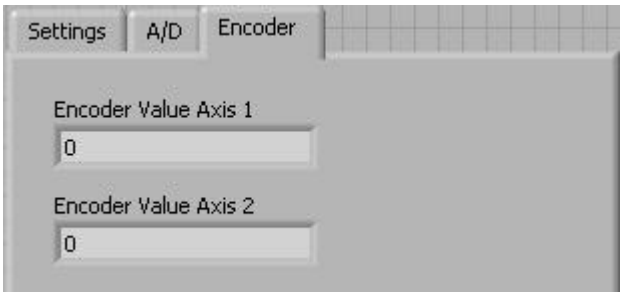
This demo has one register card with three tabs: Settings, A/D and Encoder.
(For the description of Settings register card please see above.)

A/D register card:



This register card shows the present voltage at channel 1and 2 of the MCC A/D inputs. These are both graphically displayed as well as a number in the text box.

Encoder register card:

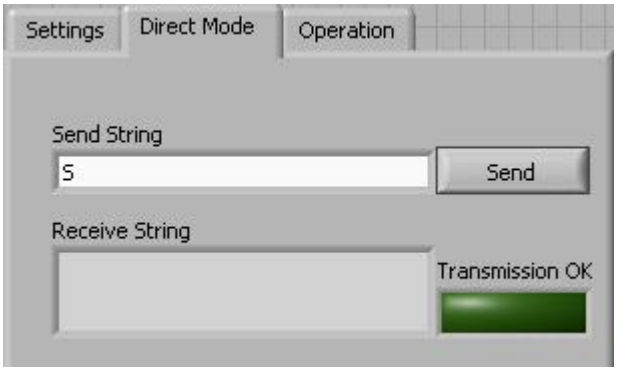


The MCC encoder counters and the counter value are imported and displayed in the text boxes.

5.3 Application for Direct Mode and Motor Driving

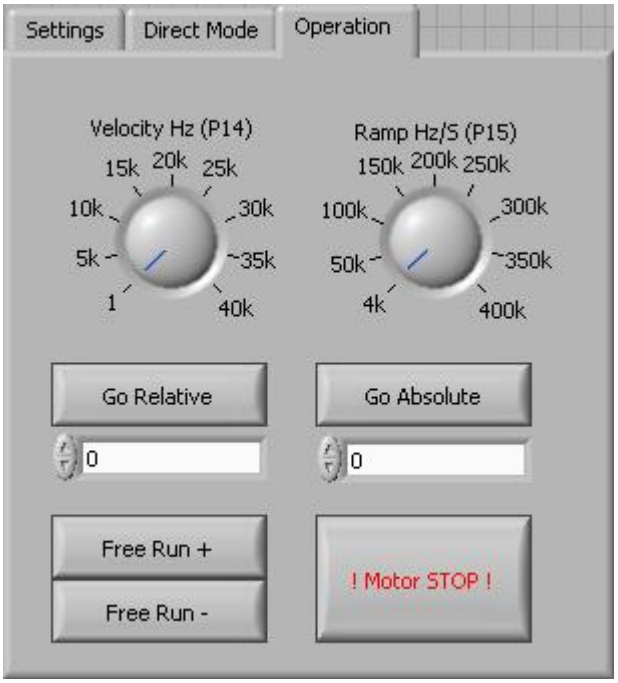
This demo is a small application to demonstrate
- **Directmode-MCC.vi**
- **Parameter-MCC.vi** and
- **Drive-MCC.vi** files.

Direct Mode register card:



Send String	Input of the transmitting command
Send	Transfer of the command
Receive String	Display of the MCC answer
Transmission OK	Command recognized: LED on (ACK) command not recognized: LED off (NAK)

Operation register card:

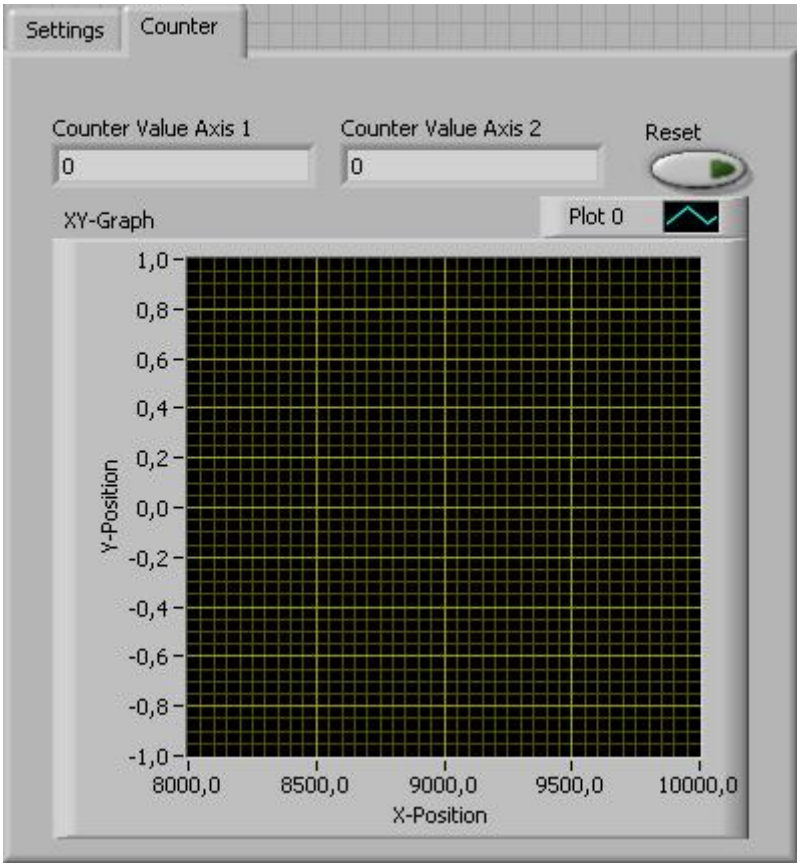


Velocity Hz (P14)	Sets the driving speed of the MCC. Free Run + or – enables to change the velocity also during the run.
Ramp Hz/S (P15)	Sets the acceleration- and axes ramp. The value can only be transferred when the motor is at a standstill.
Go Relative	Moves from the actual position by entered value (textbox).
Go Absolute	Moves the entered value (textbox) referred to the zero point..
Free Run +	Starts a free run toward positive direction
Free Run -	Starts a free run toward negative direction
! Motor STOP !	Cancels each running positioning and stops the motor.

5.4 Internal Distance Counter Reading and Displaying

This demo reads the internal MCC distance counter (P20) and displays it as a counter value as well as a diagram.

Counter register card:

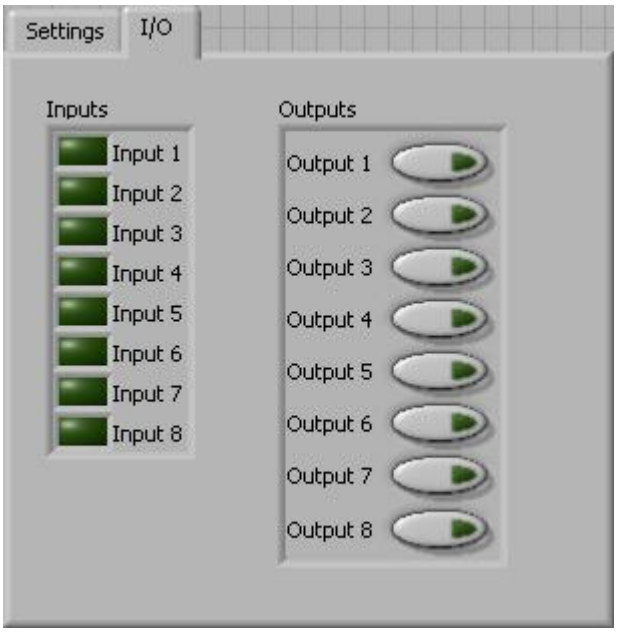


Counter Value Axis 1	Counter value display of axis 1 (X)
Counter Value Axis 2	Counter value display of axis 2 (Y)
Reset	Deletes the graph
XY Graph	Diagram of the varying counter value at axis 1 and in the system of coordinates.

5.5 The MCC Inputs / Outputs Reading / Setting

This demo reads and displays the MCC inputs and activates the outputs.

I/O register card:

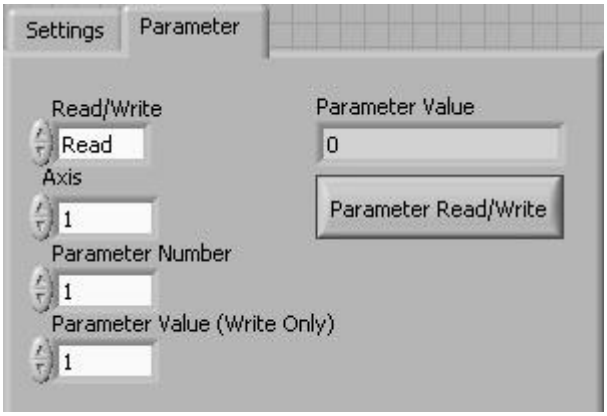


Inputs	Display of the MCC input status: LED on = Status High LED off = Status Low
Outputs	The respective MCC outputs can be switched.

5.6 Parameters Reading / Writing

This demo reads and writes the MCC parameters.

Parameter register card:

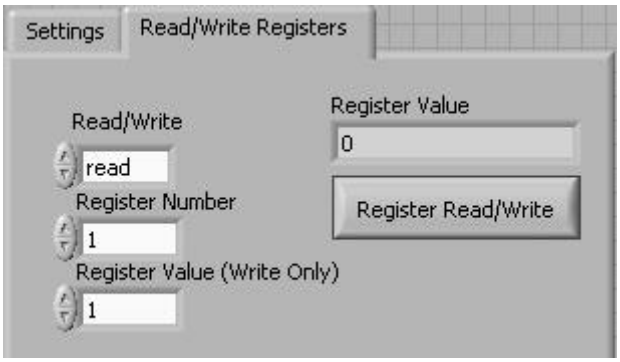


Read/Write	Reading or writing the parameters
Axis	Axis, whose parameters are changed
Parameter Number	Parameter number, which is modified
Parameter Value (Write Only)	Value to which the parameter is changed (write only)
Parameter Value	Read parameter of the controller (read only)

5.7 Register Reading and Writing

This demo reads and writes the MCC Register.

Read/Write Registers register card:

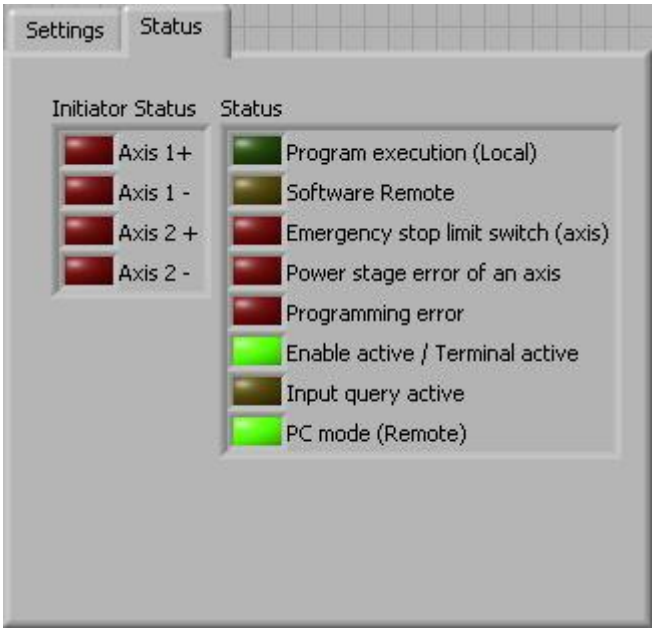


Read/Write	Reads or writes the registers
Register Number	Register number which is read or written
Register Value (Write Only)	Value that is written in the register. (write only)
Register Value	Value that is written out of the register. (read only)

5.8 Initiator and Controller Status Reading

This demo reads and displays the initiator status and the general MCC status.

Status register card:



Initiator Status	Initiator status of the controller. The LED is on, when the initiator is activated. The SUI command is used.
Status	General status of the controller. The LEDs display the status. The SB command is used.

The LED colors are described in the MiniLog Programming Manual.

6 Parameters

For operating a stepper motor controller different presettings as speed, acceleration ramps or waiting time are required which are called **Parameters**.

Default parameters are stored at delivery which can be used in several applications. You can read and edit these parameters with LabVIEW Parameter-VI or MiniLog-Comm.

Counters also belong to the list of parameters, which will be continuously actualized by the program. The counters can be read and some of them can be edited, too.

- For each axis separate parameters have to be set. Insert an X or Y to mark the axis in front of the parameter number (also valid: 1 or 2).

  Example: XP15 is the acceleration ramp value for axis X.
- Parameters (e.g. speeds) may be modified several times within a program, too.
- Parameter values can be entered or read.
- P48 and P49 can only be read.
- P19 to P22 are counters. They will be actualized by the program during axis movement.
- P27 to P49 are special parameters for MCC-2.

List of Parameters

No.	Meaning	Default
P01	Type of movement 0 = rotational Rotating table, 1 limit switch for mechanical zero (referencing) 1 = linear for XY tables or other linear systems, 2 limit switches: Mechanical zero and limit direction – Limit direction +	0
P02	Measuring units of movement 1 = step 2 = mm 3 = inch 4 = degree	1
P03	Conversion factor for the thread 1 step corresponds to ... If P03 = 1 (steps) the conversion factor is 1. Computing the conversion factor: $Conversion\,factor = \frac{Thread}{Number\,of\,steps\,per\,revolution}$ Example: 4 mm thread pitch 200-step motor = 400 steps/rev. in the half step mode $Conversion\,factor = \frac{4}{400} = 0.01$	1
P04	Start/stop frequency The start/stop frequency is the maximum frequency to start or stop the motor without ramp. At higher frequencies, step losses or motor stop would be the result of a start or stop without ramp. The start/stop frequency depends on various factors: type of motor, load, mechanical system, power stage. The frequency is programmed in Hz.	400
P05 **P06**	not used	
P07	Emergency stop ramp The frequency is programmed in 4000-Hz/sec-steps.	100 000

No.	Meaning	Default
P08	f_{max} MØP (mechanical zero point) Run frequency during initializing (referencing) Enter in Hz (integer value)	4000
P09	Ramp MØP Ramp during initializing, associated to parameter P08 Enter in 4000-Hz/sec-steps	4000
P10	f_{min} MØP Run frequency for leaving the limit switch range Enter in Hz	400
P11	MØP offset for limit switch direction + Distance between reference point MØP and limit switch activation Unit: is defined in parameter P02	0
P12	MØP offset for limit switch direction – Distance between reference point MØP and limit switch activation Unit: is defined in parameter P02	0
P13	Recovery time MØP Time lapse during initialization Enter in msec	20
P14	f_{max} Run frequency during program operation Enter in Hz (integer value) (40 000 maximum)	4000
P15	Ramp for run frequency (P14) Input in 4000-Hz/sec-steps (4000 to 500 000 Hz/sec)	4000
P16	Recovery time position Time lapse after positioning Input in msec	20

No.	Meaning	Default
P17	Boost (defined in P42) 0 = off 1 = on during motor run 2 = on during acceleration and deceleration ramp Remarks: The boost current can be set in parameter P42. You can select with parameter P17 in which situation the controller switches to boost current. P17 = 1 means, the boost current always is switched on during motor run. During motor standstill the controller switches to stop current.	0
P18	not used	
P19	Electronically zero counter Used for setting operating points. At standstill of the axis, P19 can be read or programmed during program execution.	0
P20	Mechanical zero counter This counter contains the number of steps referred to the mechanical zero (MØP). Can be read at axis standstill. If the axis reaches the MØP, P20 will be set to zero.	0
P21	Absolute counter Indicates the true position. At any time P21 can be asked, programmed or modified. P21 is not automatically set to zero when the MØP is reached.	0
P22	Encoder counter Indicates the true encoder position.	0
P23	Axial limitation pos. direction + If the number of steps is reached, the run in + direction is aborted. 0 = no limitation	0

No.	Meaning	Default
P24	Axial limitation neg. direction – If the number of steps is reached, the run in – direction is aborted. 0 = no limitation	0
P25	Compensation for play Indicates the step number, the target position in the selected direction is passed over and afterwards is started in reverse direction. 0 = no compensation for play	0
P26	not used	
P27	Initiator type 0 = PNP normally closed contact (NCC) 1 = PNP normally open contact (NOC)	0
P 28 to **P33** not used		
P34	Encoder type 0 = no 1 = incremental 2 = serial interface SSI binary Code 3 = serial interface SSI Gray Code  Connect the **correct** encoder type! Do not parameterize an incremental encoder as SSI. Danger of damage!	0
P35	Encoder resolution for SSI encoder Enter max. encoder resolution in bit (max. 31 Bit)	10
P36	Encoder function 0 = counter	0
P37	not used	
P38	Encoder preferential direction of rotation 0 = + (positive) 1 = – (negative)	0
P39	Encoder conversion factor 1 increment corresponds to ...	1

No.	Meaning	Default
P40	Stop current in 0.1 A steps Values: 0 to 2.5 A Input: 0 to 25	2
P41	Run current in 0.1 A steps Values: 0 to 2.5 A Input: 0 to 25	6
P42	Boost current in 0.1 A steps Values: 0 to 2.5 A Input: 0 to 25	10
P43	Current delay time in msec	20
P44	not used	
P45	Step resolution 1 to 256 1 = Full step 10 = 1/10 step 2 = Half step 16 = 1/16 step 4 = 1/4 step 128 = 1/128 step 8 = 1/8 step 256 = 1/256 step	4
P46	Current Shaping (CS), also see appendix A 0 = Off 1 = On Recommended setting: P46 = 1	1
P47	Chopper frequency 0 = low 1 = high The chopper frequency value depends on P46: If P46 = 0, then is applied: P47 = 0: 16 kHz P47 = 1: 22.5 kHz If P46 = 1, then is applied: P47 = 0: 50 kHz P47 = 1: 75 kHz Recommended setting: P47 = 1	1
P48	Power stage type (read only) 0 = linear 1 = chopper	(read only)
P49	Power stage temperature in °C (read only) (only for linear power stage type)	(read only)

7 Copyright and Disclaimer of Warranty

The LabVIEW communication software and its documentation are protected by copyright law. The manual must not be copied, reproduced, put into machine readable form, neither complete nor in parts, without the prior written permission of National Instruments Corporation. It is permitted to create backups of the Phytron Freeware VIs for personal use. However, the program must not be modified or sold.

Disclaimer of warranty

The Phytron specific VIs and the associated manual were made with great care and reproduced with the involvement of effective control measures. Each CD-ROM is checked before delivery with a well-known scanner program for viruses of all kinds. Nevertheless errors can exist.

Phytron does not warrant that the software is free of errors.
Phytron is not liable for any damages arising by using this software. The user or application designer is ultimately responsible for the suitability of the software.

We refer to our terms of delivery and payment, in particular to item VII liability and item IX software use.

You'll agree to our delivery and payment conditions by installing, copying or using the software otherwise. If you disagree to these conditions you aren't authorized to install or use the software.

Trademarks

In this manual several trademarks are used which are no longer explicitly marked as trademarks within the text. The lack of this signs may not be used to draw the conclusion that these products are free of rights of third parties.
Some product names used herein are for instance:

- Microsoft is a registered trademark and WINDOWS is a trademark of the Microsoft Corporation in the USA and other countries.
- LabVIEW is a registered trademark of National Instruments Corporation.

8 Index

Phytron GmbH • Industriestraße 12 • 82194 Gröbenzell, Germany
Tel. +49(0)8142/503-0 • Fax +49(0)8142/503-190 • E-Mail info@phytron.de • www.phytron.de

Phytron, Inc. • 600 Blair Park Road Suite 220 • Williston, VT 05495 USA
Tel. +1-802-872-1600 • Fax +1-802-872-0311 • Email info@phytron.com • www.phytron.com