

Micro800 Programmable Controllers: Getting Started with Motion Control Using a Simulated Axis

Catalog Numbers Bulletin 2080-LC30, 2080-LC50





Important User Information

Solid-state equipment has operational characteristics differing from those of electromechanical equipment. Safety Guidelines for the Application, Installation and Maintenance of Solid State Controls (publication <u>SGI-1.1</u> available from your local Rockwell Automation sales office or online at <u>http://www.rockwellautomation.com/literature/</u>) describes some important differences between solid-state equipment and hard-wired electromechanical devices. Because of this difference, and also because of the wide variety of uses for solid-state equipment, all persons responsible for applying this equipment must satisfy themselves that each intended application of this equipment is acceptable.

In no event will Rockwell Automation, Inc. be responsible or liable for indirect or consequential damages resulting from the use or application of this equipment.

The examples and diagrams in this manual are included solely for illustrative purposes. Because of the many variables and requirements associated with any particular installation, Rockwell Automation, Inc. cannot assume responsibility or liability for actual use based on the examples and diagrams.

No patent liability is assumed by Rockwell Automation, Inc. with respect to use of information, circuits, equipment, or software described in this manual.

Reproduction of the contents of this manual, in whole or in part, without written permission of Rockwell Automation, Inc., is prohibited.

Throughout this manual, when necessary, we use notes to make you aware of safety considerations.



Allen-Bradley, Micro800, Micro830, Micro850, Connected Components Workbench, PanelView, Rockwell Software, Rockwell Automation, and TechConnect are trademarks of Rockwell Automation, Inc.

Trademarks not belonging to Rockwell Automation are property of their respective companies.

About This Publication

This quick start is designed to provide instructions for implementing a motion control project using Connected Components WorkbenchTM software and a Micro830TM/Micro850TM programmable logic controller (PLC). It makes use of a sample project to illustrate the basic steps that a user needs to perform to use the motion control feature in Micro830 and Micro850 controllers.

To assist in the design and installation of your system, refer to the Micro830 and Micro850 Programmable Controllers User Manual, publication 2080-UM002.

The beginning of each chapter contains the following information. Read these sections carefully before beginning work in each chapter.

- **Before You Begin** This section lists the steps that must be completed and decisions that must be made before starting that chapter. The chapters in this quick start do not have to be completed in the order in which they appear, but this section defines the minimum amount of preparation required before completing the current chapter.
- What You Need This section lists the tools that are required to complete the steps in the current chapter. This includes, but is not limited to, hardware and software.
- Follow These Steps This illustrates the steps in the current chapter and identifies which steps are required to complete the examples using specific networks.

To be able to use the motion control feature effectively, you need to be familiar with programming in function block diagram, structured text, and ladder programming.

This quick start works hand-in-hand with Micro830 and Micro850 Programmable Controllers User Manual, publication <u>2080-UM002</u>.

Required Software

Audience

To complete this quick start, the following software is required:

• Connected Components Workbench revision 2 and later

Connected Components Workbench is the main programming software for Micro800 systems. It provides a choice of IEC 61131-3 programming languages (ladder diagram, function block diagram, structured text) with user defined function block support that optimizes machine control.

You will need the Connected Components Workbench software to configure your axis parameters, write your motion control function block programs, execute your function blocks, and monitor your axis status.

Additional Resources

Resource	Description
Micro830 and Micro850 User Manual, publication 2080-UM002	A detailed description of how to install and use your Micro830 and Micro850 programmable controller and expansion I/O system.
Micro800 Programmable Controller External AC Power Supply Installation Instructions, publication 2080-IN001	Information on wiring and installing the optional AC power supply.
Kinetix 3 Motion Control Indexing Application, publication <u>CC-OS025</u>	Quick start instructions designed to provide instructions for implementing a Kinetix® 3 component-class drive motion control indexing application by using Connected Components Workbench software and a Micro830 programmable logic controller (PLC).
Industrial Automation Wiring and Grounding Guidelines, publication <u>1770-4.1</u>	More information on proper wiring and grounding techniques.
Connected Components Workbench Online Help	Online Help that provides a description of the different elements of the Connected Components Workbench software.

	Important User Information	ii
	Preface	
	About This Publication	iii
	Audience	iii
	Required Software	iii
	Additional Resources	iv
	Where to Start	
	Overview of Sample Project	vii
	Optional: PanelView Component	vii
	Hardware and Software Compatibility Follow These Steps	viii ix
	Chapter 1	
Create a Micro800 Project	Introduction	1
	Before You Begin	1
	What You Need	1
	Create a Micro800 Project in Connected Components Workbenc	:h2
	Chapter 2	
Configure Motion Axis Properties	Introduction	5
	Before You Begin	5
	What You Need	5
	Follow These Steps	6
	Configure General Properties	/
	Configure Limits Properties	10
	Configure Dynamics Properties	10
	Configure Homing Properties	11
	Configure Embedded I/O Properties	13
	Chapter 3	
Write Your Motion Control	Introduction	15
Programs	Before You Begin	15
5	What You Need	15
	Follow These Steps	16
	Create Axis_PowerUp Program	17
	Create Homing Program	21
	Create Program for MC_MoveRelative	
	Create Program for MC_MoveAbsolute Function Block	
	Create Program for MC_Move Velocity Function Block	
	Build and Download Programs	····.43 40

Wire Your Controller for Motion Control

Execute Your Motion Control Function Blocks

Chapter 4

Introduction	
What You Need	
Wire the Controller	50

Chapter 5

Overview of Sample Project

This quick start instruction serves to enable users to the use the motion control feature on Micro830 and Micro850 controllers. It uses a sample simulation program to familiarize the user with motion control instructions and related parameter and wiring configurations. In particular, this project lets the user enable an axis, home an axis, move an axis, and use touch probe to capture current position in a simulated environment.

The project uses the minimum components to use motion control. It does not require a servo drive. The PTO output is wired directly to the high speed counter input. It makes use of high speed counter inputs to count the pulse train output (PTO) for the current position.

The following diagram illustrates how this project simulates motion control. The elements of this project are shown in the following diagram.



TIP HSC wiring for PTO feedback is not shown and is for simulation purposes only.

Download the Simulation Project

You can also download the code for the complete simulation project from the following link:

http://www.rockwellautomation.com/go/scmicro800

The downloadable code includes an optional PanelView Component (PVc) program, which allows the user to easily update and monitor different axis parameters through a PVc screen.

If you opt not to use a PanelView Component, use the Connected Components Workbench software to toggle variable input values and trigger motion instructions. Axis monitoring can also be done through the same software through the Axis Monitor feature.

To get started, check that your Micro800 controller supports motion control.

Hardware and Software Compatibility

The motion control feature on Micro830 and Micro850 controller is implemented through Pulse Train Outputs (PTOs) and motion axes, which are summarized in the following table.

Controller	PTO (built-in)	Number of Axes Supported
10/16 Points 2080-LC30-10QVB 2080-LC30-16QVB	1	1
24 Points 2080-LC30-24QVB 2080-LC30-24QBB 2080-LC50-24QVB 2080-LC50-24QBB	2	2
48 Points 2080-LC30-48QVB 2080-LC30-48QBB 2080-LC50-48QVB 2080-LC50-48QBB	3	3

PTO and Motion Axis Support on Micro830 and Micro850 Controllers

IMP	DRT	ANT
-----	-----	-----

Software and Firmware Requirements

- For programming, motion control is supported on Connected Components Workbench software revision 2 and later.
- Micro830 controllers require firmware revision 2 and later.

ATTENTION: To use the Micro800 motion control feature effectively, users need to have a basic understanding of the following:
 PTO components and parameters
 Sone the Micro830 and Micro850 User Manual publication 2080 UM002, for a general evenue

- See the Micro830 and Micro850 User Manual, publication <u>2080-UM002</u>, for a general overview of Motion components and their relationships.
- Programming and working with elements in the Connected Components Workbench software The user needs to have a working knowledge of ladder diagram, structured text, or function block diagram programming to be able to work with motion function blocks, variables, and axis configuration parameters.



ATTENTION: To learn more about Connected Components Workbench and detailed descriptions of the variables for the Motion Function Blocks, you can refer to Connected Components Workbench Online Help that comes with your Connected Components Workbench installation.

Follow These Steps

The major subsections for this quick start project are outlined in the following flowchart. Follow the steps under each subsection to become familiar with the required procedure to configure your controller and set up a simulation project for motion control.



Notes:

Create a Micro800 Project

Introduction

In this chapter, you will create a sample Micro830/Micro850 controller project through the Connected Components Workbench.

Before You Begin

Ensure that you have Connected Components Workbench revision 2 properly installed.

What You Need

- Connected Components Workbench revision 2 or later
- Firmware revision 2 and later for Micro830 controllers

Create a Micro800 Project in Connected Components Workbench

Launch the Connected Components Workbench software.

1. On the Device Toolbox, expand the list of Controllers by clicking the + sign.

Device Toolbox	- 4	×
E Discover		
🗆 Catalog		
2080-LC50	-24QBB	^
2080-LC50	-24QVB	
2080-LC50	-24QWB	
2080-LC50	-48AWB	
2080-LC50	-48QBB	
2080-LC50	-48QVB	
2080-LC50	-48QWB	
Expansion Module:	5 8	
Drives		Y

TIP	If your controller is online, use the Discover feature to automatically discover your controller.
IMPORTANT	Make sure that your controller is one of the following compatible controllers:
	• 2080-LC30-24QBB
	• 2080-LC30-24QVB
	• 2080-LC30-48QBB
	• 2080-LC30-48QVB
	• 2080-LC50-24QBB
	• 2080-LC50-24QVB
	• 2080-LC50-48QBB
	• 2080-LC50-48QVB
IMPORTANT	Motion control on Micro830 controllers require firmware revision 2 or later.

2. Select major revision 2 when manually adding a Micro830 controller.

001000100101110	add to project		
2080-1	C30-24QBB		
Maior	Revision: 2	•	
Chook this h	ox to always u	se latest firmw	are revision.
Crieck this L			
Crieck this t			

3. Drag your controller onto the Project Organizer pane.

Name: Project40*	
Micro850 Programs Global Variables	
DataTypes	

- Go to File → Save Project As. Then, provide a project name for your project.
- 5. Click OK.

ł	Co	nnect	ed Co	npone	nts
ļ	File	Edit	View	Build	De
		New			
		Open			
ĺ		Close			
l		Recer	nt Projec	ts	F
l		Save			
l		Save	Project	As	
		Exit			

Name:	Simulator	
Location:	C:\Documents and Settings\My Documents\CCW\Simulator	Browse

TIP Optional PanelView Component Program

This simulation project can also include a PanelView Component program to allow for easier monitoring and toggling of axis parameter values through a PVc screen. The code for this optional program is downloadable from the following link, along with the complete code for this project:

http://www.rockwellautomation.com/go/scmicro800

Notes:

Configure Motion Axis Properties

Introduction

In this chapter, you will configure the axis parameters through Connected Components Workbench.

Торіс	Page
Configure General Properties	7
Configure Motor and Load Properties	9
Configure Limits Properties	10
Configure Dynamics Properties	11
Configure Homing Properties	12

Before You Begin

Acquire a basic understanding of the different motion axis parameters by referring to the Micro830 and Micro850 Programmable Controllers User Manual, publication <u>2080-UM002</u> and/or the Connected Components Workbench Online Help.

What You Need

- Connected Components Workbench revision 2 or later
- Firmware revision 2 and later for Micro830 controllers

Follow These Steps

To configure your axis, follow these steps.



Configure General Properties

Launch the Connected Components Workbench software.

- 1. Open the project you have created in the previous chapter.
- 2. On the Project Organizer pane, double-click the controller name to bring up the Device Properties pane.



- 3. Under the Controller properties tree, go to Motion. Right-click <New Axis> and choose Create.
- **4.** Provide the name "Simulator" for the axis. Alternatively, you can press F2 to name the axis.



5. Click General to bring up the General properties tab.

Simulator - General					
Axis Name:	Simulator				
PTO Channel:	EM_0	×			
Pulse Output:	IO_EM_DO_00				
Direction Output:	IO_EM_DO_03				
✓ Drive Enable Output			In-Position Input		
Output:	IO_EM_DO_06	~	Input:	IO_EM_DI_10	V
Active Level	High	~	Active Level:	High	~
Drive Ready Input			Touch Probe Input		
Input:	IO_EM_DI_09	\sim	Input:	IO_EM_DI_03	
Active Level:	High	~	Active Level:	High	~

6. Configure the general properties as shown in the table.

General Properties Parameters

Parameter	Value
Axis Name	Simulator
PTO Channel	EM_0
Enable Drive Enable Output	Tick option box to enable
Drive Enable Output	IO_EM_DO_06
Drive Enable Output Active Level	High
Enable Touch Probe Input	Tick option box to enable
Touch Probe Input	IO_EM_DI_03
Touch Probe Input Active Level	High

Configure Motor and Load Properties

 On the Controller Configuration tree, under Motion, click Motor and Load to bring up the Motor and Load tab.

mm	v
sec	
olution parameters may ca	use Axis runaway.
10000	2.0
10000	
10	0.0 mm
Inverted	▼
Bi-Directional	~
10 🔵 ms	
	mm sec olution parameters may ca 10000 11 Inverted Bi-Directional 10 ms

2. Configure Motor and Load parameters as follows.

Motor and Load Properties

Parameter	Value
Position	mm
Steps per revolution	10000
Travel per revolution	10 mm
Polarity	Inverted
Mode	Bi-directional
Change delay time	10 ms

Configure Limits Properties

 On the Controller Configuration tree, under Motion, click Limits to bring up the Limits properties tab.

- Simulator - Limits							
Hard Limits							
When hard limit is reach	hed, apply:	Emergency Stop Profile					
Lower Hard Limit Active Level:	High	Upper Hard Limit Active Level:	High				
Switch Input.	10_11_01_00	Switch Input.	10_01_01_01				
Soft Limits							
(i) When soft limit is	reached, Emergency	Stop Profile will be applied.					
Lower Soft Limit:	0.0	mm Upper Soft Limit:	0.0 mm				

2. Configure Limits parameters as shown in the table.

Limits Properties

Parameter	Value		
When hard limits is reached, apply	Emergency Stop Profile		
Lower Hard Limit	Tick option box to enable		
Lower Hard Limit Active Level	High		
Upper Hard Limit	Tick option box to enable		
Upper Hard Limit Active Level	High		

Configure Dynamics Properties

 Under Motion, click Dynamics to bring up the Dynamics properties tab.



2. Configure Dynamics parameters as shown in the table.

Dynamics Properties

Parameter	Value
Start/Stop Velocity	20.0 mm/sec 120.0 rpm
Max Velocity	100.0 mm/sec 600.0 rpm
Max Acceleration	1000 mm/sec ²
Max Deceleration	1000 mm/sec ²
Max Jerk	10000 mm/sec ³

Configure Homing Properties

 Under Motion, click Homing to bring up the Homing properties tab.



2. Configure homing parameters as shown in the table.

Parameter	Value
Homing Direction	Negative
Homing Velocity	20 mm/sec
Homing Acceleration	100 mm/sec ²
Homing Deceleration	100 mm/sec ²
Homing Jerk	400 mm/sec ³
Creep Velocity	1.0 mm/sec
Home Offset	0.0 mm
Home Switch Input	Tick option box to enable
Home Switch Input Active Level	High

Configure Embedded I/O Properties

Controller - Embedded I/O General . Memory Serial Port Input Filters USB Port . Ethernet Inputs Input Filter Internet Protocol Default ÷ • Ξ - Port Settings DC 5 µs ÷ 2 - 3 Port Diagnostics 4 - 5 DC 5 µs Ŧ Date and Time Ξ DC 5 µs ÷ Interrupts 6 - 7 Startup/Faults Default ÷ 8 - 9 Modbus Mapping 10 - 11 Default ÷ Embedded I/O Default 12 - 13 • - Motion

Go to Controller Properties \rightarrow Embedded I/O, and update the input filter values as shown below.

Input filters are configured so that high speed pulse from the PTO is properly captured. When High Speed Counters and Touch Probe are used, input filters need to be configured to match the high speed input.

Notes:

Write Your Motion Control Programs

Introduction

In this chapter, you will write movement function block programs that will allow you to control the movement profile of your axis.

Торіс	Page
Create Axis_PowerUp Program	17
Create Homing Program	21
Create Program for MC_MoveRelative	28
Create Program for MC_MoveAbsolute Function Block	31
Create Program for MC_MoveVelocity Function Block	36
Create Program for MC_TouchProbe Function Block	43

Before You Begin

Learn about motion control function blocks by referring to the Micro830 and Micro850 Programmable Controllers User Manual, publication <u>2080-UM002</u>, and the Connected Components Workbench Online Help.

What You Need

- Connected Components Workbench revision 2 or later
- Firmware revision 2 and later for Micro830 controllers

Follow These Steps

To write your motion control function blocks, follow these steps.





Create Axis_PowerUp Program

- 1. Open the project for your controller in Connected Components Workbench.
- 2. On the Project Organizer pane, right-click Programs, select Add New LD: Ladder Diagram. Press F2 to rename the program to Axis_PowerUp. Press Enter.
- 3. Right-click Axis_PowerUp program, choose Open.





4. From the Toolbox, double-click Block to add it to the rung. Alternatively, you can drag and drop Block onto the rung. Your ladder rung should appear as shown.



5. On the Instruction Block Selector window that appears, type MC_Power on the default entry field to filter the MC_Power function block. Choose MC_Power. Click OK.

The Instruction Block Selector; MC_Power							
Controller : 2	080LC5024QBE	В					
	Name	2	Catego	огу	1 Туре		
MC.	_Power			•	• dt* ▼ dt	1	
MC_Po	wer	Mot	ion		C.	Contri	ols the power st
<							>
- Parameters -							
	Name		Data	Туре	Directio	n 🗼	Dimensic 📥
		- a	A*	* A*		* A*	- (=
En	able		BOOL	•	VarInput	•	
Sta	atus		BUUL	•	VarUutput	•	
Err	oriD		UINI	-	VarUutput	-	
1	III		RIUU		O SRI II IPSI IP		•
nstance:	MC_Power_1		*	. 🗹 🤅	Show Paramet	ers	
nputs:	\$ 4			V	EN ZENO		
Scope:	UntitledLD1						
					OK		Cancel

Your ladder rung should appear as follows.



6. From the Toolbox, double-click Block to add it to the rung. Alternatively, you can drag and drop Block onto the rung.

-®- Pulse Falling Edge Coil	
⊣⊢ Direct Contact	
-//⊢ Reverse Contact	
⊣P⊢ Pulse Rising Edge Contact	
⊣¤⊢ Pulse Falling Edge Contact	
Elock	
🖃 General	
There are no usable controls in t	nis group. Drag an item onto this

text to add it to the toolbox.

7. On the Instruction Block Selector window that appears, type MC_Reset on the default entry field. Choose MC_Reset and click OK.

_	ridille	<u> </u>	Catego	ry	1 Тур	e		Com
h	MC_Reset	- de			01 + 0	e*		
MC_	Reset	Motion			Ē.	Rese	ts all internal a	kis-related errors
1								
ramete	ers Name	e	Data	[vpe	Directi	on 1	Dimension	Alia
		- A*		- A⁺		× #*	- A*	
_	ErrorID		UINT		VarOutput			ErID
	LINGING			_	1 Care			
	ExecEdge		BOOL	× .	vai			
	ExecEdge Execute		BOOL	÷	Val VarInput	*		Exec
	ExecEdge Execute Error		BOOL BOOL BOOL	*	VarInput VarOutput	*		Exec Err
	ExecEdge Execute Error Done		BOOL BOOL BOOL BOOL	*	Val Varlnput VarOutput VarOutput	*		Exec Err Done
	ExecEdge Execute Error Done	1	BOOL BOOL BOOL BOOL	* * *	Vailnput VarDutput VarDutput	*		Exec Err Done
(ExecEdge Execute Error Done	1	BOOL BOOL BOOL BOOL	· · ·	Vailnput Vai0utput Vai0utput	* * *		Exec Err Done

Your ladder rung should appear as follows.



Create and Assign Variables and Values for the Axis_PowerUp Program

 On the Project Organizer pane, double-click Global Variables to bring up the Variables window. Add the following variables with these corresponding data types and initial values.

Variables and Initial Values for Axis_PowerUp Program

Variable Name	Data Type	Initial Value
Simulator	AXIS_REF	
PowerUpAxis	BOOL	TRUE
PowerDone	BOOL	
SWReset	BOOL	False

2. Assign the variables to the function block elements as shown.



1 MC_Han MC_Ha ENO Simulator Axis SWhome Execute Don HomeSetPos Position Bus 0 Homin BFMD BufferMode Erre ErrorID 2 SW/home SWrstEncode 4 F Simulator. AxisHomed - H SWhome Homing 3 Homing - P 4 Simulator.AxisHomed • -IPH TON_1 TON R TRIG 1 R_TRIG CLK Q IN t#1ms ET PT

Create Homing Program

 Click Programs, select Add → New LD: Ladder Diagram. Press F2 and rename the program to Action_Homing.



2. From the Toolbox, double-click Block to add it to the rung. Alternatively, you can drag and drop Block onto the rung.

-®- Pulse Falling Edge Coil	
+⊢ Direct Contact	
- ∕ - Reverse Contact	
-IP는 Pulse Rising Edge Contact	
네바는 Pulse Falling Edge Contact	
Ellock	
🗆 General	

3. On the Instruction Block Selector window that appears, type MC_Home on the default entry field to filter out MC_Home.



4. Choose MC_Home. Click OK. Your ladder rung should appear as follows.



5. Create a second rung. From the Toolbox, select Rung and drag it onto the space just below the first rung.

Toolbox	×		
🖃 LD			
R Pointer			
🗢 Return			
→>> Jump			

- 6. Select Branch from the Toolbox and drag it onto the second rung.
- Toolbox
 Image: Constraint of the second second

-®- Reset Coil

-®- Pulse Rising Edge Coil

- 7. Select Direct Contact from the Toolbox and drag it onto the second rung as shown.
- 8. Select Direct Contact from the Toolbox and drag it onto the branch on the second rung as shown.
- 9. Create a third rung. From the Toolbox, select rung and drag it onto the space just below the second rung.
- **10.** Select Pulse Rising Edge Direct Contact from the Toolbox and drag it onto the third rung.
- Select Set Coil from the Toolbox and drag it onto the third rung. Your third rung should appear as follows.



Toolbox	×
🗆 LD	
Revinter	
Hod Rung	
🗢 Return	
-≫ Jump	
🗂 Branch	

0 50.00
-®- Reset Coil
-@- Pulse Rising Edge Coil
-®- Pulse Falling Edge Coil
⊣ ⊢ Direct Contact
+/- Reverse Contact
-IP⊢ Pulse Rising Edge Contact
⊣N⊢ Pulse Falling Edge Contact
Eleck



- **12.** Create a fourth rung. From the Toolbox, select rung and drag it onto the space just below the third rung.
- **13.** Select Branch from the Toolbox and drag it onto the fourth rung.

Toolbox	X
🖃 LD	
Review Pointer	
├	
🗢 Return	
-» Jump	
🗂 Branch	
-O- Direct Coil	
-Ø- Reverse Coil	

14. Select Pulse Rising Edge Direct Contact from the Toolbox and drag it onto the fourth rung as shown.



15. Select Block from the Toolbox and drag it onto the branch on fourth rung as shown.

-//− Reverse Contact
-IP- Pulse Rising Edge Contact
⊣ฟ⊢ Pulse Falling Edge Contact
E Block
🖃 General
There are no usable controls in this aroun. Durg an item enter this

There are no usable controls in this group. Drag an item onto this

16. On the Instruction Block Selector window that appears, type TON to filter out the TON function block. Choose TON. Click OK



- 17. Select Block from the Toolbox and drag it onto the branch on the fourth rung as shown.
- 18. On the Instruction Block Selector window that appears, type R_TRIG to filter out the R_TRIG function block. Choose R_TRIG. Click OK

+ - Direct Contact
-1/1- Reverse Contact
-IPI- Pulse Rising Edge Contact
⊣N⊢ Pulse Falling Edge Contact
Ellock
🗆 General

🔤 Instruction Block Selector: R_TRIG							
Controller : 2080LC5024QBBB							
	Name	2	Catego	ry	1 Туре		
	R_TRIG 🗸 🗸	4 °		•	A* ▼A	1	
▶ <u>R_</u> 1	IRIG	Boolea	in operati	ons	SB	Rising	g edge detection
2							>
Paramete	ers						
	Name		Data	Гуре	Directio	n 1	Dimensic 🔶
		- A*		• A*		- A*	
	CLK		BOOL	•	VarInput	*	=
	Redge 0		BOOL	* *	Var VarDutout	* *	
*			DOOL	Ŧ	Varbacpat	*	×
Instance:	R_TRIG_2		*	?	Show Paramet	ers	
Inputs:	⇒ 1				EN / ENO		
Scope:	UntitledLD						
					01	_	Consel
							Lancer

19. Select Reset Coil from the Toolbox and drag it onto the fourth rung as shown.



Create and Assign Variables and Values for the MC_Home Function Block

 Create the following Global Variables with the data types and initial values (if any) as shown in the table.

Variable Name	Data Type	Initial Value
SWHome	BOOL	
HomeSetPos	Real	110
BFMD	SInt	
t#1ms	time	
Simulator.AxisHomed	BOOL	
SWrstEncoder	BOOL	

2. On the first rung, assign the variables for the MC_Home function block elements as shown.


3. On the second rung, assign the variables as shown below.



4. On the third rung, assign the variables as shown below.



5. On the fourth rung, assign the variables as shown below.

Simulator.AxisHome	d		Homing
	TON 1	R TRIG 1	U U
	TON	R_TRIG	
		(CLK Q)	
t#1ms			
	OT ET		

1					
	Simulator Axis Homed				
					Return
2		MC_MoveR MC_Movel	elative_1 Relative ENO		
	Simulator				
		- Axisln	Axis —		
	SWrelative				
		Execute	Done -		
	Rel_Distance_Input				
		Distance	Busy -		
	Rel_Velocity				
	L	Velocity	Active -		
	Rel_Acc				
		Acceleration (Commanda		
	Rel_Decc				
		Deceleration	Error		
	Rel_Jerk				
		Jerk	ErrorID		
	BFMD				
		BufferMode			

Create Program for MC_MoveRelative

- Click Programs, select Add → New LD: Ladder Diagram. Press F2 and rename the program to Action_MoveRelative.
- 2. From the Toolbox, select Reverse Contact and drag and drop it onto the rung.



-®- Reset Coil
-@- Pulse Rising Edge Coil
-®- Pulse Falling Edge Coil
H Direct Contact
-I∕I- Reverse Contact
- Pի Pulse Rising Edge Contact
HN- Pulse Falling Edge Contact

 From the Toolbox, select Return and drag and drop it onto the rung. Your first ladder rung should appear as shown.



- 4. Create a second rung. From the Toolbox, select Rung and drag and drop it just below the first rung.
- 5. From the Toolbox, select Block and drag and drop it onto the second rung.

OOIDOX	
🖃 LD	
Revinter	
Hod Rung	
- Return	
-≫ Jump	
🗂 Branch	

-I∕I− Reverse Contact
-IP는 Pulse Rising Edge Contact
⊣¤⊢ Pulse Falling Edge Contact
E Block
🖃 General
There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.

6. On the Instruction Block Selector window that appears, type MC_MoveRelative to filter out the MC_MoveRelative function block. Choose MC_MoveRelative. Click OK. Your second rung should appear as shown.



Create and Assign Variables and Values to the MC_MoveRelative Function Block

1. Create the following Global Variables with these data types and initial values.

Name 🔺		Data	Туре	Initial Value	
	 	REAL		* A*	
Rel_Acc	R	EAL	+	1000.0	
Rel_Decc	R	EAL	•	1000.0	
Rel_Distance_Input	R	EAL	*		
Rel_Jerk	R	EAL	*	10000.0	
Rel_Velocity	R	EAL	+	100.0	
			_		
Name		Data	Туре	Initial Value	
2000	- A	1	- A*	- A	
 SWrelative		BOOL	-		

2. On the first rung, assign the variables as shown.



3. On the second rung, assign the variables as shown in the picture.





Create Program for MC_MoveAbsolute Function Block

 Click Programs, select Add → New LD: Ladder Diagram. Press F2 and rename the program to Action_MoveAbsolute.



2. From the Toolbox, select Reverse Contact and drag and drop it onto the rung.

⊣⊢ Direct Contact	
-∦- Reverse Contact	
⊣P⊢ Pulse Rising Edge Contact	
⊣∾⊢ Pulse Falling Edge Contact	
E Block	

3. From the Toolbox, select Return and drag and drop it onto the rung. Your first ladder rung should appear as shown.



4. Create a second rung. From the Toolbox, select Rung and drag and drop it just below the first rung.

Toolbox	
🖃 LD	
Revinter	
- Return	
-≫ Jump	

5. From the Toolbox, select Block and drag and drop it onto the second rung.

⊣⊢ Direct Contact	
- ∕ - Reverse Contact	
-IP는 Pulse Rising Edge Contact	
⊣№⊢ Pulse Falling Edge Contact	
Ellock	
🗆 General	

 On the Instruction Block Selector window that appears, type MC_MoveAbsolute to filter out the MC_MoveAbsolute function block. Choose MC_MoveAbsolute. Click OK. Your second rung should appear as shown in the picture.



- 7. Create a third rung. From the Toolbox, select Rung and drag and drop it just below the second rung.
- 8. From the Toolbox, select Block and drag and drop it onto the third rung.

Toolbox	X
🖃 LD	
Revinter	
- Return	
-≫ Jump	

⊢ Direct Contact	
- ∕ - Reverse Contact	
⊣P⊢ Pulse Rising Edge Contact	
- 에는 Pulse Falling Edge Contact	
🔁 Block	
😑 General	

9. On the Instruction Block Selector window that appears, type MC_MoveAbsolute to filter out the MC_MoveAbsolute function block. Choose MC_MoveAbsolute. Click OK. Your third rung should appear as shown in the picture.



Create and Assign Variables and Values to the MC_MoveAbsolute Function Block

1. Create the following Global Variables with these data types and initial values.

Name	Data	Туре	Initial Value
+ A		- A*	* A
SWabs2	BOOL	-	
AB51_Velocity	REAL		100.0
ABS1_Acc	REAL	*	1000.0
AB51_Decc	REAL	+	1000.0
ABS1_Jerk	REAL		10000.0
ABS2_Velocity	REAL	*	100.0
AB52_Acc	REAL	÷	1000.0
AB52_Decc	REAL		1000.0
ABS2_Jerk	REAL	*	10000.0
Name	Data	Туре	Initial Value
- A	1	· A	- A
SWabsolute	BOOL	+	
Abs Distance Input	REAL	+	

2. On the first rung, assign the variables as shown.

Simulator.AxisHomed		
		Return

3. On the second rung, assign the variables as shown.



4. On the third rung, assign the variables as shown.





Create Program for MC_MoveVelocity Function Block

 Click Programs, select Add → New LD: Ladder Diagram. Press F2 and rename the program to Action_MoveVelocity.

Micro850		2	Home_1
	Add 🔸	2	New ST : Structured Text
₽ <mark>FJ</mark> C	A Paste	#O	New LD : Ladder Diagram
	Properties		New FBD : Function Block Diagram

2. From the Toolbox, select Reverse Contact and drag and drop it onto the rung.

-®- Pulse Falling Edge Coil
⊣ ⊢ Direct Contact
+∕⊢ Reverse Contact
내라 Pulse Rising Edge Contact
⊣∾⊢ Pulse Falling Edge Contact
Block

3. From the Toolbox, select Return and drag and drop it onto the rung. The first rung should appear as shown.



- 4. Create a second rung. From the Toolbox, select Rung and drag and drop it onto the space just below the first rung.
- **5.** From the Toolbox, select Block and drag and drop it onto the second rung.

Toolbox	X
🖃 LD	
Pointer	
-🔿 Return	
-≫ Jump	

-19- Pulse Falling Edge Coll	
⊣ ⊢ Direct Contact	
+//- Reverse Contact	
-IP는 Pulse Rising Edge Contact	
⊣¤⊢ Pulse Falling Edge Contact	
Elock	
-	

6. On the Instruction Block Selector window that appears, type ANY_TO_SINT to filter out the function block. Choose ANY_TO_SINT. Click OK.

Instruction Block Selector: ANY_TO_SINT	
Controller : 2080LC5024QBBB	
Name 💁 Category 🗼 Type	
ANT_IU_SINT Data conversion 300 Conversion or an	<u>y v</u> a
	>
Parameters	
raiaileteis	
Name Data Type Direction 1 Dimension	on
$\tau \sigma t^* = \sigma t^* + \sigma t^* + \sigma t^*$	A*
o1 SINT VarOutput V	
*	
	_
Instance: Show Parameters	
Inputs:	
Scope: Untitled D	
OK Cance	<u>ا او</u>

Your second rung should appear as follows:



7. Create a third rung. From the Toolbox, select Rung and drag and drop it onto the space just below the second rung.

Toolbox	×
🖃 LD	
Revinter	
🗢 Return	
-≫ Jump	

8. On the Instruction Block Selector window that appears, type MC_MoveVelocity to filter out the MC_MoveVelocity function block. Choose MC_MoveVelocity. Click OK.

MoveVelocity									
Con	troller : 2080LC5024QBBB								
	Name 💡		Category		1 T	уре			^
	▼ A				A* -	A.			
	MC_Halt	Motion			1	ē.	Comn	nands a cor	
	MC_Home	Motion			1	5	Comn	nands the a	
	MC_MoveAbsolute	Motion			1	5	Comn	nands a cor	
	MC_MoveRelative	Motion			1	ēt.	Comn	hands a cor	
Para	neters Name		Data Tj	pe	Dire	ction	1	Dimensio	
			*	Æ			A*		
	CommandAborted		BOOL	•	VarOutpu	ut 👘	•		
	InVelocity		BOOL	•	VarOutpu	ut 👘	•		
	ErrorID		UINT		VarOutpu	ut	-		.
•	Active		RUUI	*	VarDutor	ił.	*	►	
Instanc	e: MC_MoveVelocity_1		~	•	Show Para	ameter	rs		
Inputs:	3			V	EN ZENO				
Scope:	UntitledLD								
						OK		Cance	

Your third rung should appear as follows:



9. Create a fourth rung. From the Toolbox, select Rung and drag and drop it onto the space just below the second rung.

Toolbox	X
🖃 LD	
R Pointer	
- Return	
-≫ Jump	

- **10.** From the Toolbox, select Block and drag and drop it onto the fourth rung.
- +Ulse Falling Edge Coll

 +I F Direct Contact

 +/- Reverse Contact

 -IP Fulse Rising Edge Contact

 -IM Fulse Falling Edge Contact

 -IM Fulse Falling Edge Contact
- On the Instruction Block Selector window that appears, type MC_Halt to filter out the MC_Halt function block. Choose MC_Halt. Click OK.

Name	0	Category	1 Tune		
MC Halt	▼ d* 2	Guiugoij	▼ dt [*] ▼ dt [*]		
MC_Halt	Motio	เ า	5	Comm	iands a contro
<					>
Parameters					
Name		Data Type	Direction	1	Dimensic
	- A	- A	•	- A*	
CommandAborted		BOOL 🔹	VarOutput	-	1
ErrorID		UINT 🚽 👻	VarOutput	-	
Active		BOOL 🔹 🔻	VarOutput	•	
ExecEdge		BOOL 🔹 🔻	Var	•	
		AXIS_REF 🔻	Var	•	
Error		BOOL 🔹 🔻	VarOutput	•	
		2001	0.1		
tance: MC_Halt_1		✓	Show Paramete	ers	
uts:			EN / ENO		
ope: UntitledLD					

Your fourth rung should appear as shown:



Create and Assign Variables and Values to the MC_MoveVelocity Function Block

1. Create the following Global variables with these data types and initial values.

Name	Data	Туре	Initial Value
- A*		- A*	- A*
SWvelocity	BOOL	*	
Vel_Input	REAL		50.0
Vel_Direct_Input	SINT	Ψ.	1
Vel_Acc_Input	REAL	-	1000.0
Vel_Dir_INT	INT		
Vel_Decc_Input	REAL	*	1000.0
Vel_Jerk_Input	REAL	*	10000.0
Halt_Dece_Input	REAL	*	1000.0
Halt_Jerk_Input	REAL	Ŧ	10000.0

2. On the first rung, assign the variables as shown in the following picture.

	5	imulator AxisHomed		
-			Return	
4	-		Hotan	

3. On the second rung, assign the variables as shown.

			ANY_TO_SINT	
≻	Vel_Dir_INT	• EN	ENO -	Vel_Direct_Input
		ui1	01	-

4. On the third rung, assign the variables as shown.



5. On the fourth rung, assign the variables as shown.



1					
	Simulator Avis Homed				
					Return
2	1/1				
2604					
	MC_TouchProbe_1.Done				TP_EXE
	1/1				_0
3					
	MC TouchProbe	1			
	MC_TouchProbe	ENO			
	Simulator				
	AxisIn	Axis -			
	NotUsingInThisRelease				
	TP EXE	rinput			
	Execute	Done -			
	TP_windowonly				
	WindowOnly	Busy			
	TP_FirstPos				
	TP_LastPos	andA			
	- LastPosition	Error -			
	Er				
	Becord				
4					
57.					
	MC_TouchProbe_1.Done	R TRIG 1 R_TRIG	ſ	1 gain	
		CLK Q	TP POSNOW	N ENÔ	TP recordedPos
				01	

Create Program for MC_TouchProbe Function Block

 Click Programs, select Add → New LD: Ladder Diagram. Press F2 and rename the program to Action_TouchProbe.



2. From the Toolbox, select Reverse Contact and drag and drop it onto the rung.

TOT Puise Failing Euge Coll
⊣⊢ Direct Contact
-//⊢ Reverse Contact
⊣P⊢ Pulse Rising Edge Contact
⊣∾⊢ Pulse Falling Edge Contact
Ellock

 From the Toolbox, select Return and drag and drop it onto the rung. Your first ladder rung should appear as shown.



- 4. Create a second rung. From the Toolbox, select Rung and drag and drop it below the first rung.
- 5. From the Toolbox, select Reverse Contact and drag and drop it onto the second rung.

O box com
-®- Reset Coil
-@- Pulse Rising Edge Coil
-®- Pulse Falling Edge Coil
+⊢ Direct Contact
-I/- Reverse Contact
-IP- Pulse Rising Edge Contact
-IN- Pulse Falling Edge Contact
Deck

6. From the Toolbox, select Direct Coil and drag and drop it onto the second rung.



7. Create a third rung. From the Toolbox, select Rung and drag and drop it below the second rung.

Toolbox	
🖃 LD	
R Pointer	
-🔿 Return	
-≫ Jump	
🗂 Branch	
-O- Direct Coil	
0 parrow C-1	

8. From the Toolbox, select Block and drag and drop it onto the third rung.

 On the Instruction Block Selector window that appears, type MC_TouchProbe to filter out the MC_TouchProbe function block. Choose MC_TouchProbe. Click OK.

🔤 Instruction Block Selector: M	C_TouchPr	obe			
Controller : 2080LC5024QBBB					
Name 🔒	Catego	ry.	1 Туре		
MC_touchprobe 💌 🗨		-	A* 🔻 A*		
MC_TouchProbe	lotion		ō	Reco	rds an axis posi
)		2
Parameters					
Name	Data	уре	Direction	1	Dimensic
-	A*	• A *		• A *	~ .
CommandAborted	BOOL	× 1	√arOutput	•	
ErrorID	UINT	- 1	√arOutput	•	
WindowOnly	BOOL	×)	√arInput	•	
	RUUI	~ 1	Jar	*	
NC TauakDaka 1					
Instance: MC_TouchHobe_T	~	V 51	now Paramete	ers.	
Inputs:		EI 💟	N / ENO		
Scope: UntitledLD					
			ОК		Cancel

Your third rung should appear as shown.



10. Create a fourth rung. From the Toolbox, select Rung and drag and drop it below the third rung.

Toolbox	×
🖃 LD	
Revinter	
-O Return	1
->> Jump	

- From the Toolbox, select Block and drag and drop it onto the fourth rung.
- Image: Point of the second second
- 12. On the Instruction Block Selector window that appears, type R_TRIG to filter out the R_TRIG function block. Choose R_TRIG. Click OK.

	0	Calassa	1 7		
Nam		Lategory	lype		
	Boole	an operations		Bising	edae detectio
			75257	g	
					>
Deservations					
- Parameters					
	Name	Data Type	Direction	1	Dimensic 🔶
	- A	- A		A*	- 6
CLK		BOOL 🔹 🔻	VarInput	•	
Redge		BOOL 🔹	Var	•	
Q		BOOL 🔹	VarOutput	•	
		*		*	•
	1		Show Paramete	rs	
nstance: R_TRIG_					
nstance: R_TRIG_			EN / ENO		
nstance: R_TRIG_			EN / ENO		

 From the Toolbox, select Block and drag and drop it onto the fourth rung.

-1/1- Reverse Contact	
-IP는 Pulse Rising Edge Contact	
니아는 Pulse Falling Edge Contact	
🔁 Block	
🖃 General	

14. On the Instruction Block Selector window that appears, type 1 gain to filter out 1 gain. Choose 1 gain. Click OK.

Instruction Block Selector:	1 gai	n			
Controller : 2080LC5024QBBB					
Name 🔒		Category	1 Туре		
1 • #			A* ▼ A*		
▶ 1 gain	Arithme	etic	0P	Assign	nment of one va
Parameters					
Name		Data Type	Direction		Dimension
a	* A*	* A**	Varlinout	• <i>A</i> *	- A*
01			VarDutput	-	
*		*		-	
					
Instance:		Y ≤ 1	Show Paramete	rs	
Inputs:		V	EN ZENO		
Scope: UntitledLD					
			OK		Canaal
					Laricel

Your fourth rung should appear as follows.

R TRIG 1 R_TRIG	EN EN	
		1

Create and Assign Variables and Values to the Action_TouchProbe program

1. Create the following Global Variables with these data types and initial values.

Name		Data Type		Initial Value	
	- A*		- A*		 <i>A</i>⁺
TP_windowonly		BOOL	*	FALSE	
TP_FirstPos		REAL		110.0	
TP_LastPos		REAL		900.0	
TP_recordedPos		REAL	*		

2. Assign variables to the elements on the first rung as shown.

1			
Simulator.AxisHomed			
1/1			Deturn
1/1			Return

3. Assign variables to the elements on the second rung as shown.



4. Assign variables to the elements on the third rung as shown.



5. Assign variables to the elements on the fourth rung as shown.

MC_TouchProbe_1.Done		<i>R TRIG 1</i> R_TRIG			1 gain	
	(CLK	<u> </u>	TP_POSNOW	• EN	ENO -	TP_recordedPos
			-	ei1	01	-

6. Save the program.

Build and Download Programs

After writing the programs, build and download the user program into the controller.

Wire Your Controller for Motion Control

Introduction

In this chapter, you will wire your controller based on the sample project wiring configuration.

IMPORTANT The wiring diagram presented in this chapter serves the purpose of a simulation project only. To help you wire your controller to an actual servo drive for motion control, refer to the chapter, "Positioning with Embedded Pulse Train Outputs" in the Micro830 and Micro850 Programmable Controllers User Manual, <u>2080-UM002</u>.

What You Need

- 2080-LC30-xxQVB, 2080-LC30-xxQBB, 2080-LC50-xxQBB, 2080-LC50-xxQVB controller
- PanelView Component (optional)

Wire the Controller

Wire the controller based on the wiring configuration indicated in the following drawing and table.



Input Terminal Block

Output Terminal Block

Input Channel	Logical Variable Name in Software	Description
2	_I0_EM_DI_00	Limit_L
3	_I0_EM_DI_01	Limit_R
4	_I0_EM_DI_02	Home
5	_I0_EM_DI_03	Probe
6	_I0_EM_DI_04	PTO0-IN
7	_I0_EM_DI_05	DIRO-IN
8	_I0_EM_DI_06	
9	_I0_EM_DI_07	
11	_I0_EM_DI_08	
12	_I0_EM_DI_09	DriverReady
13	_I0_EM_DI_10	
14	_I0_EM_DI_11	
15	_I0_EM_DI_12	
16	_I0_EM_DI_13	

Output Channel	Logical Variable Name in Software	Description
4	_I0_EM_D0_00	PTOO
5	_I0_EM_D0_01	
8	_I0_EM_D0_02	
9	_I0_EM_D0_03	DIRO
10	_I0_EM_D0_04	Driver Ready Sensor
11	_I0_EM_D0_05	Touch Probe Sensor
12	_I0_EM_D0_06	Drive Enable
13	_I0_EM_D0_07	Home Sensor
14	_I0_EM_D0_08	L Limit Sensor
15	_I0_EM_D0_09	R Limit Sensor

Execute Your Motion Control Function Blocks

Introduction

In this chapter, you will use the different motion function blocks to control the movement and direction of the axis. At the end of the chapter, you should be more familiar with the movement function blocks and their operation.

Торіс	Page
Power Up the Motion Axis	54
Execute MC_Home	55
Execute MC_MoveRelative	58
Execute MC_MoveAbsolute	61
Execute MC_MoveVelocity and MC_Halt	66
Execute MC_TouchProbe	69

Before You Begin

Familiarize yourself with the motion control function blocks by referring to the Micro830 and Micro850 Programmable Controllers User Manual, publication <u>2080-UM002</u> and/or the Connected Components Workbench Online Help.

What You Need

- Connected Components Workbench revision 2 or later
- Firmware revision 2 and later for Micro830 controllers

Follow These Steps



To execute the motion function blocks, perform the following steps.

Go to Remote RUN Mode

Before you execute your motion control function blocks, go to Remote RUN mode, and enable Debug mode through the Connected Components Workbench software.

Axis Monitoring

Note that while in DEBUG mode, you can always check the status of your axis and access diagnostic information through the Axis Monitor feature in Connected Components Workbench.

💐 (Running) - Connected Componen	ts Workbench	
File Edit View Build Debug Tools	Communications Window Help	
1 🛅 🗁 🔠 - 🔙 🕹 🖻 🛍 🔊 - 1	🔍 - 🚇 - 🖳 🔒 inExeStop1 🔹 - 🚰 🎌 🖕 🔛 🕨 🖕 🖓 🖓	·)
· · · B I	<u>u</u> ∗ A ≡ ≡ ≡ Ξ ,	
Project Organizer 🛛 🗸 🕂 🗙	Micro850 - Axis Monitor Micro850 Motion-POU	
Name: PTO-pulse	Axis Name: Axis0 Axis State: Discrete Motion Axis Homed: No Movement: Constant Velocity Error Description:	_
DataTypes	Command Position: 946.363 mm Command Velocity: 80.0 mm/se	c
User-Defined Function Blocks	Target Position: 2345.678 mm Target Velocity: 80.0 mm/se	c

Power Up the Motion Axis

The MC_Power function block controls the power stage of the axis, whether ON or OFF. The MC_Reset function block transitions the axis state from ErrorStop to StandStill by resetting all internal axis-related errors. The outputs of the function block instances are not changed.

The Axis_PowerUp variable has an initial value of TRUE so the axis powers up as soon as the controller enters RUN mode. When the axis is powered up, the DriveEnable output is TRUE.



The SWReset variable is set to initial value of False so that the MC_Reset function block does not execute until thePanelView Component sets it to True (or SWReset is set to True in Debug mode in Connected Components Workbench using the Variable Monitor). Setting the SWReset variable to True clears any errors.

To use this sample project with the PanelView Component, you can download the PanelView Component program code for the project from the following link:

http://www.rockwellautomation.com/go/scmicro800

Execute MC_Home

The MC_Home function block commands the axis to perform the "search home" sequence. The Home Sensor is connected to the configured Home Switch input (see <u>Configure Homing Properties on page 12</u> for the configuration).

For this simulation, the initial position is initialized to 300 mm so that when the homing sequence starts in the negative direction, the axis moves to the left until the Home Sensor is reached. The axis then reverses direction and creeps back until the Home Sensor is encountered again and the Home Sensor input transitions from True to False. The negative edge is used to mark the Home position.



The following sequence describes the homing method used in this simulation project:

- 1. Moving part moves to its left side (in negative direction);
- 2. When home switch is detected, the moving part decelerates to a stop;
- **3.** Moving part moves back (in positive direction) in creep velocity to detect Home Switch $On \rightarrow Off$ edge.
- **4.** Once Home Switch On \rightarrow Off is detected, record the position as mechanical home position, and decelerate to stop;
- **5.** Move to the configured home position (the mechanical home position recorded during moving back sequence, plus the home offset configured in axis configuration in Connected Components Workbench.

The function block completes at "StandStill" if the homing sequence is successful.

MC_Home can only be aborted by the function blocks MC_Stop or MC_Power. Any abort attempt from other moving function blocks will result in function block failure with Error ID = MC_FB_ERR_STATE. However, homing operation is not interrupted, and can be executed as usual.



The MC_Home function block should initially appear as follows.

 Set the SWHome variable input to True to trigger the MC_Home function block.

When the Homing sequence is complete with no error, the Done output will be True and the Home position will be set to 110 mm.

(The Home Sensor is at position 100 mm and the width of the pulse is 10 mm.)



		MC Home 1 MC Home			
	Simulator				
	0	-AxisIn Ax	cis — 0		
	SWhome				
	True	Execute Dor	True		
	HomeSetPos		Entre		
	110.0	Position Bus	sy False		
	0		False		
	BEMD	Homina Mo Activ			
	0	-BufferMode CommandA	False		
		Em	or False		
		Error	0		
		LIOI			
2	Trigger for Encoder to initia	lise			
2	Trigger for Encoder to initia	lise			
2	Trigger for Encoder to initia	SWhome			
2	Trigger for Encoder to initia	SWhome			
2	Trigger for Encoder to initia	SWhome Simulator,AxisHomed			
2	Trigger for Encoder to initia	SWhome Simulator.AxisHomed			
2	Trigger for Encoder to initia SWhome	SWhome Simulator.AxisHomed			
2	Trigger for Encoder to initia SWhome	SWhome Simulator.AxisHomed Simulator.AxisHomed			
2	Trigger for Encoder to initia SWhome	SWhome Simulator.AxisHomed Simulator.AxisHomed			
2 3 4	Trigger for Encoder to initia SWhome	SWhome Simulator.AxisHomed Simulator.AxisHomed	70W 1 TON		R TRIG 1 R TRIG
2 3 4	Trigger for Encoder to initia SWhome	SWhome Simulator.AxisHomed Simulator.AxisHomed	<i>TDN 1</i> TON 1 TON		R TRIG 1 R TRIG CLK 0
2 3 4	Trigger for Encoder to initia SWhome P	SWhome Simulator.AxisHomed Simulator.AxisHomed	7 <i>DN</i> 1 TON IN		R TRIG 1 R TRIG CLK 0

After execution, the MC_Home function block should appear as shown. When SWHome becomes False, the Done and Error bits gets cleared.

Execute MC_MoveRelative

The MC_MoveRelative function block commands an axis of a specified distance relative to the current position at the time of the execution.



In this section, you will use the MC_MoveRelative function block to move your axis towards a positive or negative direction. The initial state of your MC_MoveRelative program (that is, before execution) is shown below.

Action_MoveRelative-Pl	DU Action_MoveVelocity-VAR	R Micro850	Action_MoveVeloc	ity-POU		
1						
	Simulator.AxisHomed					
	<u></u> //					Return
2		MC_MoveF	Relative_1 Relative			
		EN EN	ENO			
	Simulator			_		
	0	AxisIn	Axis 🗕	0		
	SWrelative			_		
	False	Execute	Done -	False		
	Rel_Distance_Input			_		
	0.0	Distance	Busy -	False		
	Rel_Velocity			_		
	0.0	Velocity	Active	False		
	Rel_Acc			_		
	0.0	Acceleration (Commanda 🗕	False		
	Rel_Decc			_		
	0.0	Deceleration	Error	False		
	Rel_Jerk			_		
	0.0	Jerk	ErrorID	0		
	BFMD					
	0	BufferMode				

Before execution, the value of inputs such as distance, velocity, acceleration, jerk, deceleration, and buffer mode is 0. Execute is False as the application has not been run yet. Outputs such as Done, Busy, Active, CommandAborted, Error, are False.

1. Update the values of the input *MC_MoveRelative_1* MC_MoveRelative variables as shown and set 2 SWRelative to True to trigger EN ENO the MC_MoveRelative Simulator 0 function block. AxisIn Axis SWrelative True Execute Done Rel_Distance_Input 200.0 False Distance Busy Rel_Velocity 100.0 Velocity Active Rel_Acc 1000.0 False Acceleration Commanda. Rel_Decc 1000.0 False Error Deceleration Rel_Jerk 10000.0 ErrorID - Jerk BFMD 0 BufferMode

These values move your axis to a positive (right) direction by a distance of 200 mm, relative to the initial distance input (which is 0). The axis moves at a velocity of 100 mm/sec², an acceleration of 1000 mm/sec³, and deceleration of 10000 mm/sec^3 .

IMPORTANT	Deceleration or acceleration inputs should have a positive value. If deceleration or acceleration is set to be a non-positive value, an error is reported.
IMPORTANT	The Jerk input should have a non-negative value. If Jerk is set to be a negative value, error will be reported.
	If maximum jerk is configured as zero in Connected Components Workbench motion configuration, all jerk parameters for the motion function block has to be configured as zero. Otherwise, the function block reports an error.

Action_MoveRelative-VAR Action_MoveRelative-POU Micro850 1 ator AxisHo 1/1 Return 2 *MC_MoveRelative_1* MC_MoveRelative ENO EN Simulator 0 AxisIn Axis SWrelative True Execute Rel_Distance_Input 200.0 Busy - Distance Rel_Velocity 100.0 - Velocity Active Rel_Acc 1000.0 - Acceleration Comm nanda. Rel_Decc 1000.0 - Deceleration Error Rel_Jerk 10000.0 ErrorID 🗕 Jerk BFMD 0 BufferMode

While the axis is moving, note the Busy output is True.

Execute MC_MoveAbsolute

The MC_MoveAbsolute function block allows you to command an axis to a specified absolute position at the rate of Velocity, Acceleration, Deceleration, and Jerk inputs specified.



2. Specify a distance value of $200 \text{ mm.}^{(1)}$ Then, set the Simulator.AxisHomed SWAbsolute variable to True to ╢ 2 trigger the function block. ENO Simulator 0 SWabsolute False Execute Don Abs Distance Input 200.0 Position Busy ABS1_Velocity 100.0 Active Velocity ABS1 Acc 1000.0 Acceleratio andA ABS1_Decc 1000.0 Error Deceleration ABS1_Jerk 10000.0 Jerk ErrorID ABSMoveDirection 0 Dire BFMD 0 uffermo 3

The objective is to move the axis to an absolute position of 200 mm at the end of execution, regardless of the initial value of the position input. The move will be at the velocity rate of 100 mm/sec², acceleration rate of 1000 mm/sec³, deceleration rate of 1000 mm/sec³, and jerk rate of 10000 mm/sec³.

TIP	For MC_MoveAbsolute, direction input is ignored.
IMPORTANT	Deceleration or acceleration inputs should have a positive value. If deceleration or acceleration is set to be a non-positive value, an error will be reported.
IMPORTANT	The Jerk input should have a non-negative value. If Jerk is set to be a negative value, error will be reported.
	If maximum jerk is configured as zero in Connected Components Workbench motion configuration, all jerk parameters for the motion function block has to be configured as zero. Otherwise, the function block reports an error.

⁽¹⁾ The PanelView Component can be used to set the inputs and monitor the outputs. For non-PVc users, you can refer to the Connected Components Workbench to monitor the state of your outputs.
Abort a Movement Function Block

1. Set the MC_MoveAbsolute1 distance input to 600.



2. To perform an aborted move, set the MC_MoveAbsolute2 function block distance input to 700.



3. Trigger MC_MoveAbsolute1. Before the function block finishes executing, trigger MC_MoveAbsolute2. In this case, MC_MoveAbsolute2 will abort MC_MoveAbsolute1.

Note that all MC_MoveAbsolute1 output bits will be set to False, while CommandAborted goes True.

All output bits for the aborting function block MC_MoveAbsolute2 will be set to False, while Done output is True.



For the aborted function block MC_MoveAbsolute1, all output bits will be set to False, and CommandAborted is set to True.

MC_MoveAbsolute_1 MC_MoveAbsolute			
	EN	ENO	
Simulator			
0	AxisIn	Axis	0
SWabsolute			
False	Execute	Done	True
Abs_Distance_Input			
600	Position	Busy	False
ABS1_Velocity			
100.0	Velocity	Active	False
ABS1_Acc			
1000.0	Acceleration Co	ommandA•	False
ABS1_Decc			
1000.0	Deceleration	Error	False
ABS1_Jerk			
10000.0	Jerk	ErrorID	0
ABSMoveDirection			
0	Direction		
BFMD			
0	Buffermode		

For the aborting function block MC_MoveAbsolute1, all output bits will be set to False, and CommandAborted is set to True. While executing, notice that outputs Busy and Active are True. After successful execution, Done should be True. If any error occurs during execution, the Error output will be True. You can check the status of your axis through the Axis Monitoring page of the Connected Components Workbench software.



Execute MC_MoveVelocity and MC_Halt

The MC_MoveVelocity function block commands a never ending axis to move at a specified velocity until the hard or soft limit is reached.

TIP Velocity can be a signed value. Users are advised to use positive velocity.

Direction input for the MC_MoveVelocity function block can be used to define the direction of the move (that is, negative velocity x negative direction = positive velocity).

Before execution, the MC_MoveVelocity program should appear as shown.



For MC_MoveVelocity, direction input value can be 1 (positive direction), 0 (current direction) or -1 (negative direction).

For any other value, only the sign is taken into consideration. For example, -3 denotes negative direction, +2 denotes positive direction, and so on.

For MC_MoveVelocity, the resulting sign of the product value derived from velocity x direction decides the motion direction, if the value is not 0. For example, if velocity x direction = +300, then direction is positive.

TIP

- **1.** Update the MC_MoveVelocity 2 input variables as shown. ANY_TO_SINT EN ENO Vel_Dir_INT Vel_Direct_Input 0 0 ₩i1 01 3 MC N ENO ΕN Simulator 0 Axis SWvelocity False InVeloci Vel_Input 50.0 Busy Vel_Acc_Input 1000.0 Ac Vel_Decc_Input 1000.0 Directio Vel_Jerk_Input 10000.0 CommandA. Vel_Direct_Input 0 Erro BFMD 0 ErrorID
- 2. To move the axis to the right, set the direction input value to 1.

3			
		MC_MoveVeloaty_1 MC_MoveVeloaty	
	Simulator 0	Avirla Avir	0
	SWvelocity False	- Execute InVelocity -	False
	Vel_Input 50.0	Velocity Busy	False
	Vel_Acc_Input 1000.0	- Acceleration Active -	False
	Vel_Decc_Input 1000.0	- Deceleration Direction -	1
	Vel_Jerk_Input 10000.0	-Jerk CommandA	False
	Vel_Direct_Input 1	Direction In Error	False
	BFMD 0	BufferMode ErrorID	0

IMPORTANT

Once an axis is flagged with error, and the error ID is not zero, the user needs to reset the axis (using MC_Reset) before issuing any other movement function block.

IMPORTANT

The update for axis status is performed at the end of one program scan cycle, and the update is aligned with the update of Motion Axis status.

3. To move the axis to the left, set the direction input to -1.

3			
		MC_MoveVeloaiy_1 MC_MoveVeloaiy EN ENO	
	Simulator 0	AxisIn Axis	0
	SWvelocity False	Execute InVelocity	False
	Vel_Input 50.0	-Velocity Busy-	False
	Vel_Acc_Input 1000.0	Acceleration Active	False
	Vel_Decc_Input 1000.0	Deceleration Direction	-1
	Vel_Jerk_Input 10000.0	Jerk CommandA.	False
	Vel_Direct_Input -1	DirectionIn Error	False
	BFMD 0	BufferMode ErrorID	0

Execute MC_TouchProbe

The MC_TouchProbe function block records the current position when the touchprobe input becomes True. This function block is useful for registering the absolute position of the asynchronous object, for example, the registration mark of a film for a vertical form fill seal machine.

To record the current position while an axis is moving, set TP_EXE variable to True to trigger the function block. Then, you have to manually trigger the touchprobe input from the hardware.

To do so, refer to <u>Wire the Controller on page 50</u>. The output _IO_EM_DO_05 is connected to the touchprobe input. To trigger it, you can use Connected Components Workbench to force turn on _IO_EM_DO_05.



Implementing Motion Control in an Actual Environment

While this quickstart allowed you to become more familiar with the different motion control function blocks and enabled you to power up, home, and move a simulated axis, you will need to implement a different wiring configuration for an actual environment (that is, with a Servo drive, motor).

To do this, you need to refer to the following publications for wiring and supporting configuration information:

- Micro830 and Micro850 Programmable Controllers User Manual, <u>2080-UM002</u> (see the chapter, "Positioning with Embedded Pulse Train Outputs")
- Kinetix 3 Motion Control Indexing Application, <u>CC-QS025</u>

Rockwell Automation Support

Rockwell Automation provides technical information on the Web to assist you in using its products. At <u>http://www.rockwellautomation.com/support/</u>, you can find technical manuals, a knowledge base of FAQs, technical and application notes, sample code and links to software service packs, and a MySupport feature that you can customize to make the best use of these tools.

For an additional level of technical phone support for installation, configuration, and troubleshooting, we offer TechConnect support programs. For more information, contact your local distributor or Rockwell Automation representative, or visit <u>http://www.rockwellautomation.com/support/</u>.

Installation Assistance

If you experience a problem within the first 24 hours of installation, review the information that is contained in this manual. You can contact Customer Support for initial help in getting your product up and running.

United States or Canada	1.440.646.3434
Outside United States or Canada	Use the <u>Worldwide Locator</u> at <u>http://www.rockwellautomation.com/support/americas/phone_en.html</u> , or contact your local Rockwell Automation representative.

New Product Satisfaction Return

Rockwell Automation tests all of its products to ensure that they are fully operational when shipped from the manufacturing facility. However, if your product is not functioning and needs to be returned, follow these procedures.

United States	Contact your distributor. You must provide a Customer Support case number (call the phone number above to obtain one) to your distributor to complete the return process.
Outside United States	Please contact your local Rockwell Automation representative for the return procedure.

Documentation Feedback

Your comments will help us serve your documentation needs better. If you have any suggestions on how to improve this document, complete this form, publication <u>RA-DU002</u>, available at <u>http://www.rockwellautomation.com/literature/</u>.

Rockwell Otomasyon Ticaret A.Ş., Kar Plaza İş Merkezi E Blok Kat:6 34752 İçerenköy, İstanbul, Tel: +90 (216) 5698400

www.rockwellautomation.com

Power, Control and Information Solutions Headquarters

Americas: Rockwell Automation, 1201 South Second Street, Milwaukee, WI 53204-2496 USA, Tel: (1) 414.382.2000, Fax: (1) 414.382.4444 Europe/Middle East/Africa: Rockwell Automation NV, Pegasus Park, De Kleetlaan 12a, 1831 Diegem, Belgium, Tel: (32) 2 663 0600, Fax: (32) 2 663 0640 Asia Pacific: Rockwell Automation, Level 14, Core F, Cyberport 3, 100 Cyberport Road, Hong Kong, Tel: (852) 2887 4788, Fax: (852) 2508 1846

Rockwell Automation Publication 2080-QS001A-EN-E - January 2013