Technical Paper
PLCopen Technical Committee 2

Function Blocks for Motion Control
Part 6 – Fluid Power Extensions

Version 2.0, Published

Date: November 18, 2011

Function blocks for Fluid Power Motion Control

This document is a specification as developed by the PLCopen Fuid Power Motion Control Task Force. As such it is an addition to the PLCopen Technical Committee 2 Task Force Function Blocks for Motion Control, Version 2.0.

The present specification was written thanks to the following members of the PLCopen Task Force Motion Control:

Kenneth Ryan	Alexandria Technical College
Edwin Schwellinger	3S, Smart Software Solutions
Joachim Unfried	B & R Automation
Kurt Zehetleitner	B & R Automation
Ralf Bonefeld	Bosch Rexroth
Edgar Holembowski	Bosch Rexroth
Andrew DeRung	Eaton
Tiberiu Racsa	Eaton
Steve Zumbusch	Eaton
John Blomster	Eaton
Igor Oskroban	Eckelmann
Juergen Hipp	ISG
Roland Bublitz	Parker Hannifin
Willi Gagsteiger	Siemens
Eelco van der Wal	PLCopen

Change Status List:

Version number	Date	Change comment
V 0.1	June 24, 2009	Preliminary version – result of the decision from the June 23-24, 2009 Kick-off meeting to extend the current PLCopen Motion Control Function Blocks to facilitate MCFB utilization in control schemes involving fluid power.
V 0.2	January 1, 2010	Documenting the result of the decision finalized during 6 online meetings from June 24, 2009 to December 31, 2009.
V 0.3		Documenting the result of the decision finalized during 8 online meetings from December 31, 2009 to June 30, 2010.
V 0.4	September 18, 2010	Last release for comments within working group
V 0.9	December 20, 2010	Final document before Release for Comments
V 0.91	January 19, 2011	Result of feedback and web meeting Jan 14, 2011
V 091A	June 23, 2011	Due to inconsistencies in documents vs. decisions
V 092	October 24,2011	Contains feedback items on open issues
V 2.0	November 18, 2011	Released version

Table of Contents

1. GENERAL INTRODUCTION 6

2. OVERVIEW OF THE DEFINED EXTENDED FUNCTION BLOCKS: 7

2.1. GENERAL REMARKS TO THE FUNCTION BLOCK BEHAVIOR 7
2.1.1. Response Time 7

3. STATE DIAGRAM 8

4. DEFINED USER DERIVED DATATYPES 9

4.1. ABORTING VERSUS BUFFERED MODES 9
4.1.1. Buffered versus Non-buffered modes 10

5. FUNCTION BLOCKS – EXTENSIONS FOR MOTION CONTROL 11

5.1. MC_LOADCONTROL 11
5.2. MC_LIMITLOAD 13
5.3. MC_LIMITMOTION 16
5.4. MC_LOADSUPERIMPOSED 18
5.5. MC_LOADPROFILE 20

APPENDIX A COMPLIANCE PROCEDURE AND COMPLIANCE LIST 22

APPENDIX A 1. STATEMENT OF SUPPLIER 23
APPENDIX A 2. SUPPORTED DATATYPES 24
APPENDIX A 3. OVERVIEW OF THE FUNCTION BLOCKS 24
Appendix A 3.1 MC_LoadControl 24
Appendix A 3.2 MC_LimitLoad 25
Appendix A 3.3 MC_LimitMotion 25
Appendix A 3.4 MC_LoadSuperImposed 25
Appendix A 3.5 MC_LoadProfile 26
APPENDIX A 4. THE PLCOPEN MOTION CONTROL LOGO AND ITS USAGE 27

Table of Figures

FIGURE 1: THE STATE DIAGRAM....... 8

FIGURE 2: MC_LOADCONTROL TIMING DIAGRAM....... 12

FIGURE 3: MC_LIMITLOAD TIMING DIAGRAM 14

FIGURE 4: MC_LIMITLOAD USED IN SFC 15

FIGURE 5: MC_LIMITMOTION TIMING DIAGRAM....... 17

FIGURE 6: MC_LOADSUPERIMPOSED TIMING DIAGRAM 19

FIGURE 7: MC_LOADPROFILE TIMING DIAGRAM....... 21

Table of Tables

TABLE 1: OVERVIEW OF THE DEFINED FUNCTION BLOCKS 7

TABLE 2: SUPPORTED DEFINED DATATYPES 9

TABLE 3: THE ENUM TYPE MC_BUFFER_MODE 9

TABLE 4: OVERVIEW OF BUFFERED VERSUS NON-BUFFERED MODES 10

TABLE 5: SUPPORTED DERIVED DATATYPES 24

TABLE 6: SHORT OVERVIEW OF THE FUNCTION BLOCKS 24

1. General Introduction

At the end of 2001, PLCopen released the first version of the specification of an independent library of function blocks for motion control. It included motion functionality for single axes and multiple axes, several administrative tasks, as well as a state diagram. This specification provides the user with a standard command set and structure independent of the underlying architecture, while merging the logic and motion control in one development tool.
This structure can be used on many platforms and architectures. In this way one can decide which architecture will be used at a later stage of the development cycle. Advantages for the machine builder are, amongst others, lower costs for supporting the different platforms and the freedom to develop application software in a more independent way, without limiting the productivity of the machine. In addition to those benefits, system maintenance is easier and the education period is shorter. This is a major step forward, and is more and more accepted by users as well as suppliers.

With the release of Part 1, it was understood that additional functionality was needed. Part 1 provides the basis for a set of inter-related specifications:

Part 1 - PLCopen Function Blocks for Motion Control
Part 2 - PLCopen Motion Control - Extensions, which in the new release 2.0 is merged with Part 1
Part 3 - PLCopen Motion Control - User Guidelines
Part 4 - PLCopen Motion Control - Coordinated Motion
Part 5 - PLCopen Motion Control - Homing Procedures
Part 6 - PLCopen Motion Control - Fluid Power Extensions

Parts 2 through 6 of the standard are additions to the PLCopen Function Blocks for Motion Control (Part 1), and should not be seen as stand alone documents.

The objective of this specification "Part 6 – PLCopen Motion Control – Fluid Power Extensions" is:

> *Optimizing programming and integration of fluid power devices and systems by defining function blocks employing PLCopen standardization and modular methodology.*

2. Overview of the defined extended Function Blocks:

Administrative		*Motion*	
Single Axis	***Multiple Axis***	***Single Axis***	***Multiple Axis***
MC_LimitLoad		MC_LoadControl	
MC_LimitMotion		MC_LoadSuperImposed	
		MC_LoadProfile	

Table 1: Overview of the defined Function Blocks

2.1. General Remarks to the Function Block Behavior

2.1.1. Response Time

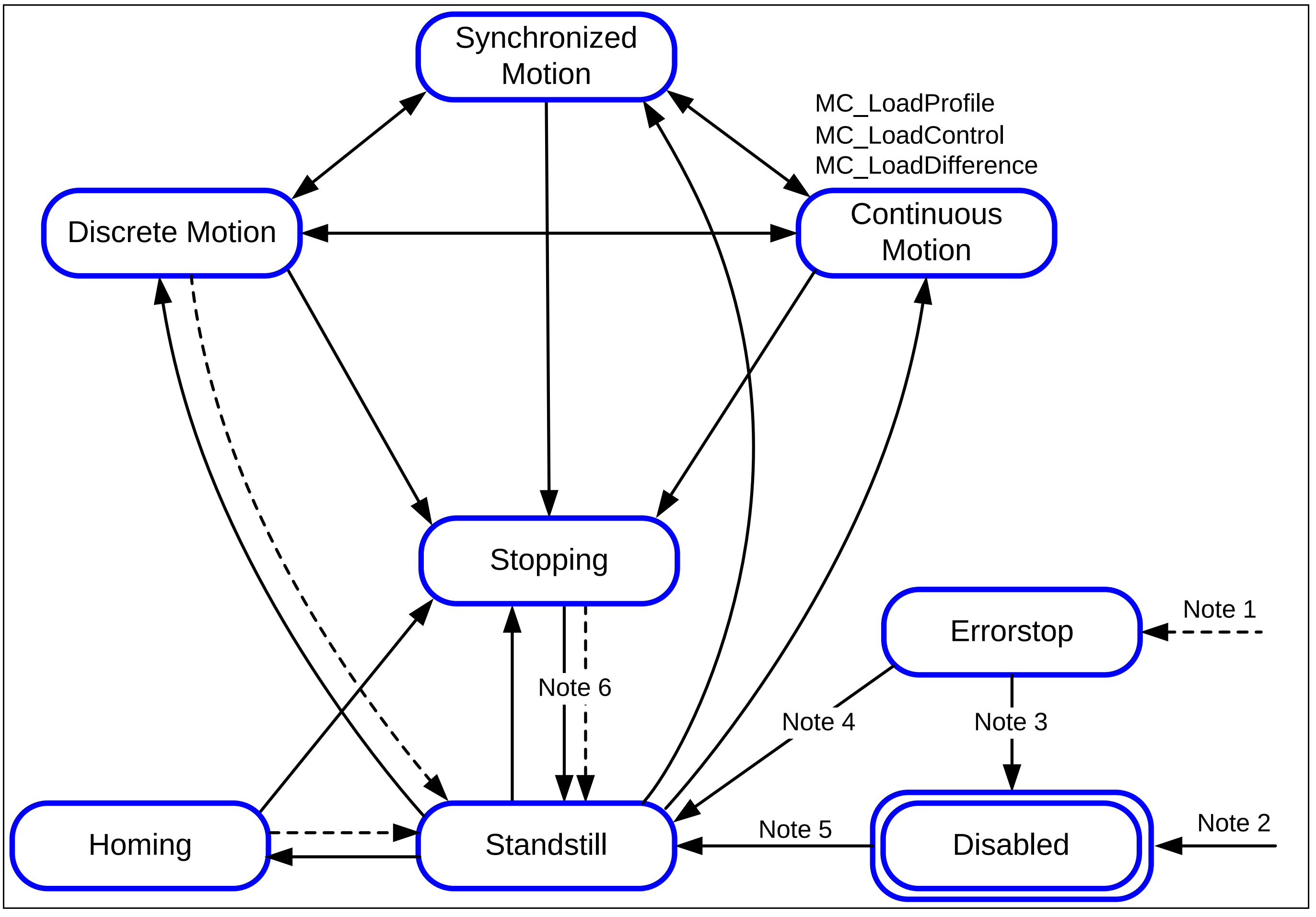
The time taken by the system to respond to a command may vary widely by manufacturer / vendor / supplier and product.
For example in one implementation, writing a variable may result in commanding a request via a serial connection and waiting for a response, while another implementation might not have a serial connection, resulting in a faster response.

3. State Diagram

The following state diagram is based on the version as defined in 'Part 1 – Function Blocks for Motion Control', Version 2.0 This specification adds three Load Function Blocks to the State Diagram:

- MC_LoadControl
- MC_LoadSuperImposed
- MC_LoadProfile

Function Blocks not listed in the state diagram do not affect the State Diagram, meaning that whenever they are called the state does not change.



Note 1:	From any state. An error in the axis occurred.
Note 2:	From any state. MC_Power.Enable = FALSE and there is no error in the axis.
Note 3:	MC_Reset AND MC_Power.Status = FALSE
Note 4:	MC_Reset AND MC_Power.Status = TRUE AND MC_Power.Enable = TRUE
Note 5:	MC_Power.Enable = TRUE AND MC_Power.Status = TRUE
Note 6:	MC_Stop.Done = TRUE AND MC_Stop.Execute = FALSE

Figure 1: The State Diagram

4. Defined User Derived Datatypes

The objective of this PLCopen Task Force Motion Control Extensions is defined in chapter 1 General Introduction.
To reach this objective, it is necessary to define additional reference types. These references are a representation of the ‘objects’ or devices, which are not necessarily a part of the process image.
As a general rule, these new reference datatypes are intended to be used in the same way as the AXIS_REF datatype, meaning that parameters can be read with similar Function Blocks having for instance an INPUT_REF instead of AXIS_REF and using the corresponding I/O parameters.
With the definition of these reference structures (or datatypes), there are Function Blocks defined which give access to the referenced data.

The following reference datatypes are defined within this document:

Defined datatype(s)	**Comment**	**Relevant Function Block(s)**
MC_DIRECTION	Enum type (1-of-4 values: mcPositiveDirection, mcShortestWay, mcNegativeDirection, mcCurrentDirection)	MC_LoadControl MC_LimitLoad MC_LimitMotion
MC_BUFFER_MODE	Enum type (one of 6 values) See below	MC_LoadControl MC_LoadProfile
MC_TL_REF	Supplier specific datatype. Example: The content of Time/Load pair may be expressed in DeltaTime/Load, where Delta could be the difference in time between two successive points.	MC_LoadProfile

Table 2: Supported defined datatypes

4.1. Aborting versus Buffered Modes

Some of the FBs have an input called ‘BufferMode’. With this input, the FB can either work in a ‘Non-buffered mode’ (default behavior) or in a ‘Buffered mode’. The difference between those modes is when they should start their action:

- A command in a non-buffered mode acts immediately, even if this interrupts another motion
- A command in a buffered mode waits till the current FB sets its ‘Done’ output (or ‘InPosition’ or ‘InVelocity’, ...).

There are several options for the buffered mode. For this reason, this input is an ENUM of type MC_BUFFER_MODE. The following modes have been identified:

- Aborting — Default mode without buffering. The next FB aborts an ongoing motion and the command affects the axis immediately.
- Buffered — The next FB affects the axis as soon as the previous movement is ‘Done’. There is no blending.
- BlendingLow — The next FB controls the axis after the previous FB has finished (equivalent to buffered), but the axis will not stop between the movements. The velocity is blended with the lowest velocity of both commands (1 and 2) at the first end-position (1).
- BlendingPrevious — blending with the velocity of FB 1 at end-position of FB 1
- BlendingNext — blending with velocity of FB 2 at end-position of FB1
- BlendingHigh — blending with highest velocity of FB 1 and FB 2 at end-position of FB1

The ENUM has been defined as follows:

No.	MC_BUFFER_MODE	Description
0	mcAborting	Start FB immediately (default mode)
1	mcBuffered	Start FB after current motion has finished
2	mcBlendingLow	The velocity is blended with the lowest velocity of both FBs
3	mcBlendingPrevious	The velocity is blended with the velocity of the first FB
4	mcBlendingNext	The velocity is blended with velocity of the second FB
5	mcBlendingHigh	The velocity is blended with highest velocity of both FBs

Table 3: The ENUM type MC_BUFFER_MODE

Supplier specific extensions are allowed after these defined Enums.

4.1.1. Buffered versus Non-buffered modes

Function block	**Can be specified as a buffered command**	**Can be followed by a buffered command**
MC_LoadControl	Yes	Yes
MC_LoadSuperImposed	No	No
MC_LoadProfile	Yes	Yes

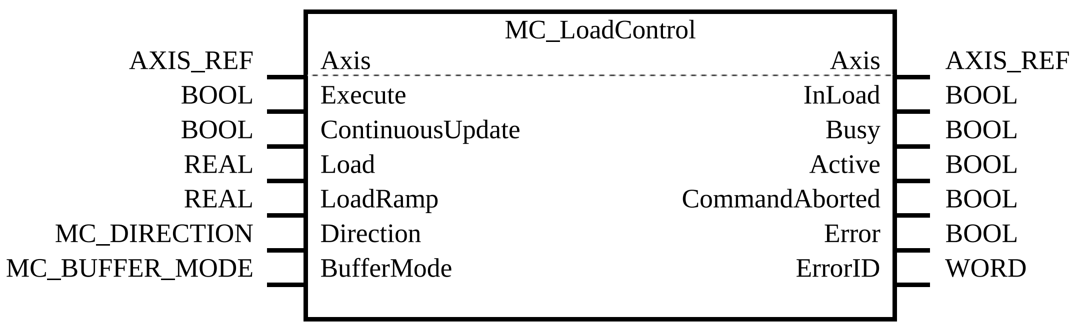
Table 4: Overview of buffered versus non-buffered modes

MC_LoadSuperImposed can not be followed by a buffered command since, if there is an underlying motion and a superimposed motion, it can not be determined which motion gives the start condition of the following buffered command.

5. Function Blocks – Extensions for Motion Control

5.1. MC_LoadControl

FB-Name		**MC_LoadControl**	
This function block continuously exerts a torque or force or pressure of the specified magnitude. This magnitude is approached using a defined ramp (‘LoadRamp’), and the Function Block sets the ‘InLoad’ output if the commanded load level is reached. Positive torque, force and differential pressure is in the positive direction of velocity, pressure is physically unsigned.			
VAR_IN_OUT			
B	Axis	AXIS_REF	Reference to the axis
VAR_INPUT			
B	Execute	BOOL	Start the motion at rising edge
E	ContinuousUpdate	BOOL	See Part 1 section 2.4.6 The input ‘ContinuousUpdate’
B	Load	REAL	Value of the load (Torque, force or pressure in technical unit [u])
E	LoadRamp	REAL	The maximum time derivative of the set value of torque, force or pressure ([u] per sec)
E	Direction	MC_DIRECTION	Enum type (1 of 3 values: mcPositiveDirection, mcNegativeDirection and mcCurrentDirection) Note: shortest way not applicable.
E	BufferMode	MC_BUFFER_MODE	Defines the chronological sequence of the FB. See 4.1.1 Buffered versus Non-buffered modes
VAR_OUTPUT			
B	InLoad	BOOL	Setpoint value of torque, force or pressure equals the commanded value and respective controller is active
E	Busy	BOOL	The FB is not finished and new output values are to be expected
E	Active	BOOL	Indicates that the Function Block has control on the axis
E	CommandAborted	BOOL	Command is aborted by another command
B	Error	BOOL	Signals that an error has occurred within the Function Block
E	ErrorID	WORD	Error identification
Note: • Using this command leads to undefined motion of the axis, unless other (motion controlled) axes or mechanical structures (arrester) are involved.			



Use Case Rational

The Function Block LoadControl provides base functionality for any application in which the devolution of forces, torque or pressures provided by an axis to a process has to be actively defined and controlled (e.g. in presses).

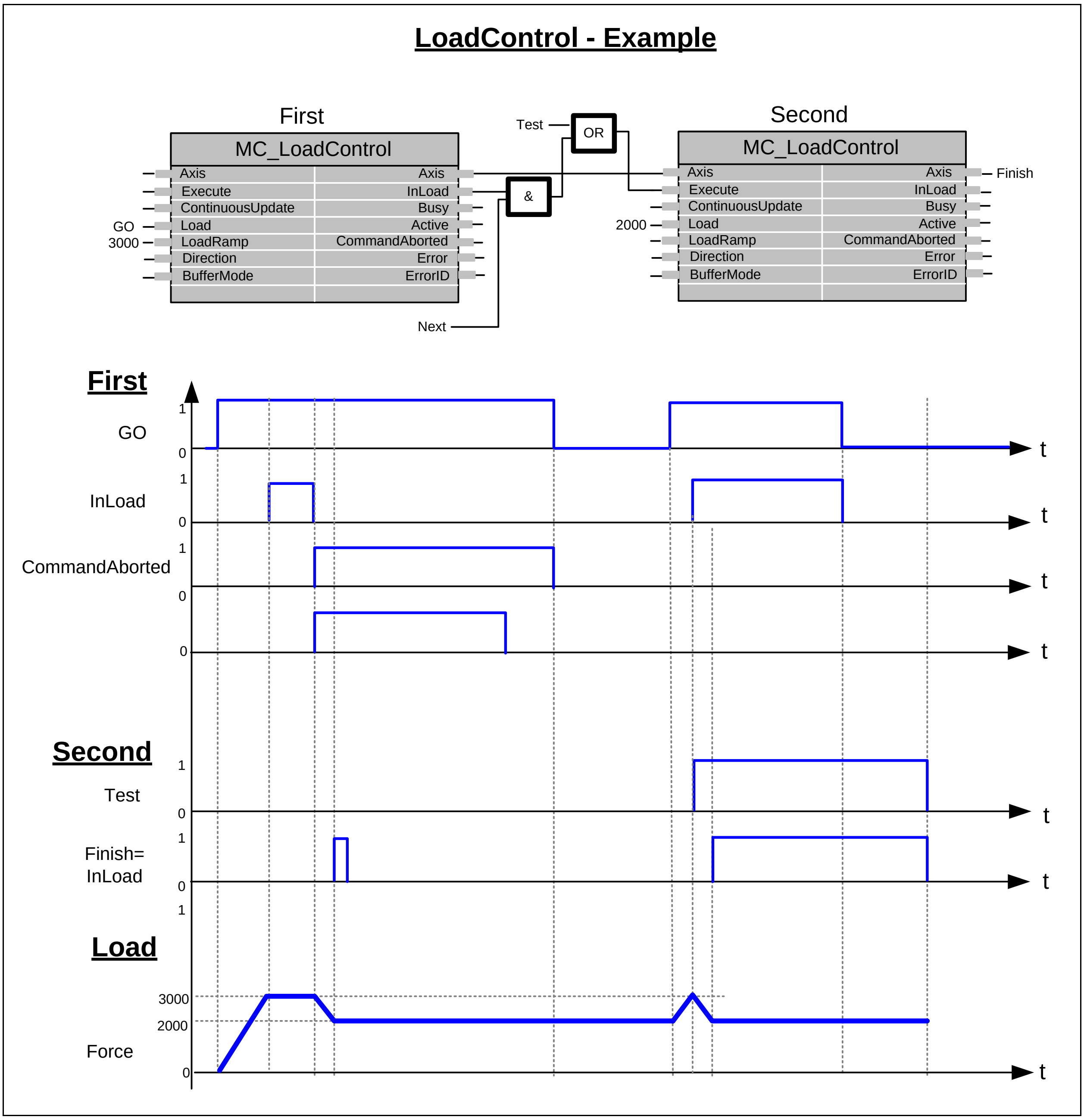
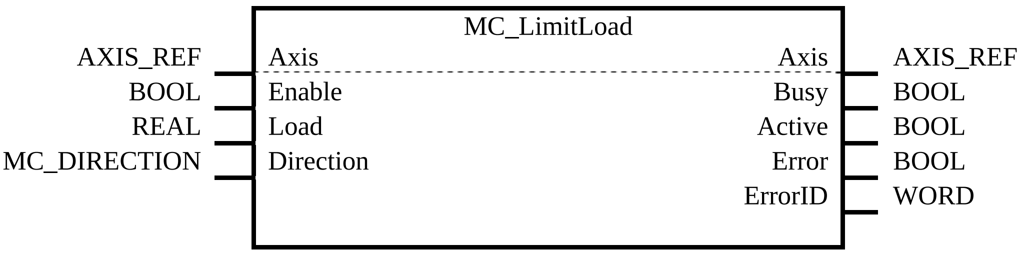


Figure 2: MC_LoadControl Timing Diagram

5.2. MC_LimitLoad

FB-Name			**MC_LimitLoad**	
This Function Block activates a limitation of the load values provided by an axis. This may be torque, force, pressure or differential pressure. The measures taken to keep the limits are vendor specific; switching between load and motion control depends on the external load conditions of the axis. The FB sets the 'Busy' output when the limiting measures are stand-by on the axis. The 'Active' output is set, when the limiting measures are active on the axis.				
VAR_IN_OUT				
	B	Axis	AXIS_REF	Reference to the axis
VAR_INPUT				
	B	Enable	BOOL	Allows function block to modify (clamp) a motion command
	B	Load	REAL	Value of the maximum applicable load on the axis (Torque, force or pressure in technical unit [u])
	E	Direction	MC_DIRECTION	Enum type (1 of 3 values: mcPositiveDirection, mcNegativeDirection and mcCurrentDirection) Note: shortest way not applicable.
VAR_OUTPUT				
	E	Busy	BOOL	The FB is not finished and new output values are to be expected
	B	Active	BOOL	Indicates that this FB is influencing the motion on the axis
	B	Error	BOOL	Signals that an error has occurred within the Function Block
	E	ErrorID	WORD	Error identification
Note: • Issuing MC_LimitLoad does not cause a motion of the axis itself. It is meant to work in parallel to a motion command. • It is not guaranteed that activity of the limiting measures will be seen by the FB: a short pulse of the limited quantities could be over before the next Function Block cycle occurs.				



Use Case Rational

The Function Block MC_LimitLoad is intended to provide overload protection for a *process* in terms of driving forces, torque or pressures during motion (e.g. mould protection in injection moulding machines).
If load values on the axis exceed the given limit, appropriate measures are taken to keep this limit, implying that the motion will not be following the programmed path but now depends on the load conditions. However, the 'Active' output of the MC_MoveXXX will stay TRUE in this case, following the modified PLCopen definition "The 'Active' output indicates that the FB has control of the path generation for the axis". This is despite the fact that, physically, only the load-conditions or the movement of an axis can be independently controlled with set values. With actual load below programmed limit, the programmed motion will proceed. The Function block can be applied in different scenarios which could be e.g.

- A more centralized application in terms of a "protection mode", where the complete motion is load limited. In this case the function block would be enabled independently from the motion program itself.
- A more decentralized application in terms of additional functionality during the motion program. In this case the function block would be activated by and within the motion program itself. An application example is the mould protection scenario mentioned above, restricting the limiter activity to a certain phase of the programmed motion. Ensuring that limits are only supervised e.g. while one certain MC_MoveXXX has primary control on the axis can be achieved by enabling MC_LimitLoad by the 'Active' output of the MC_MoveXXX. In this way the limitation is only activated when the MC_MoveXXX takes control on the axis for the first time and is deactivated when the MC_MoveXXX looses control on the axis by 'Done', 'CommandAborted' or 'Error'.

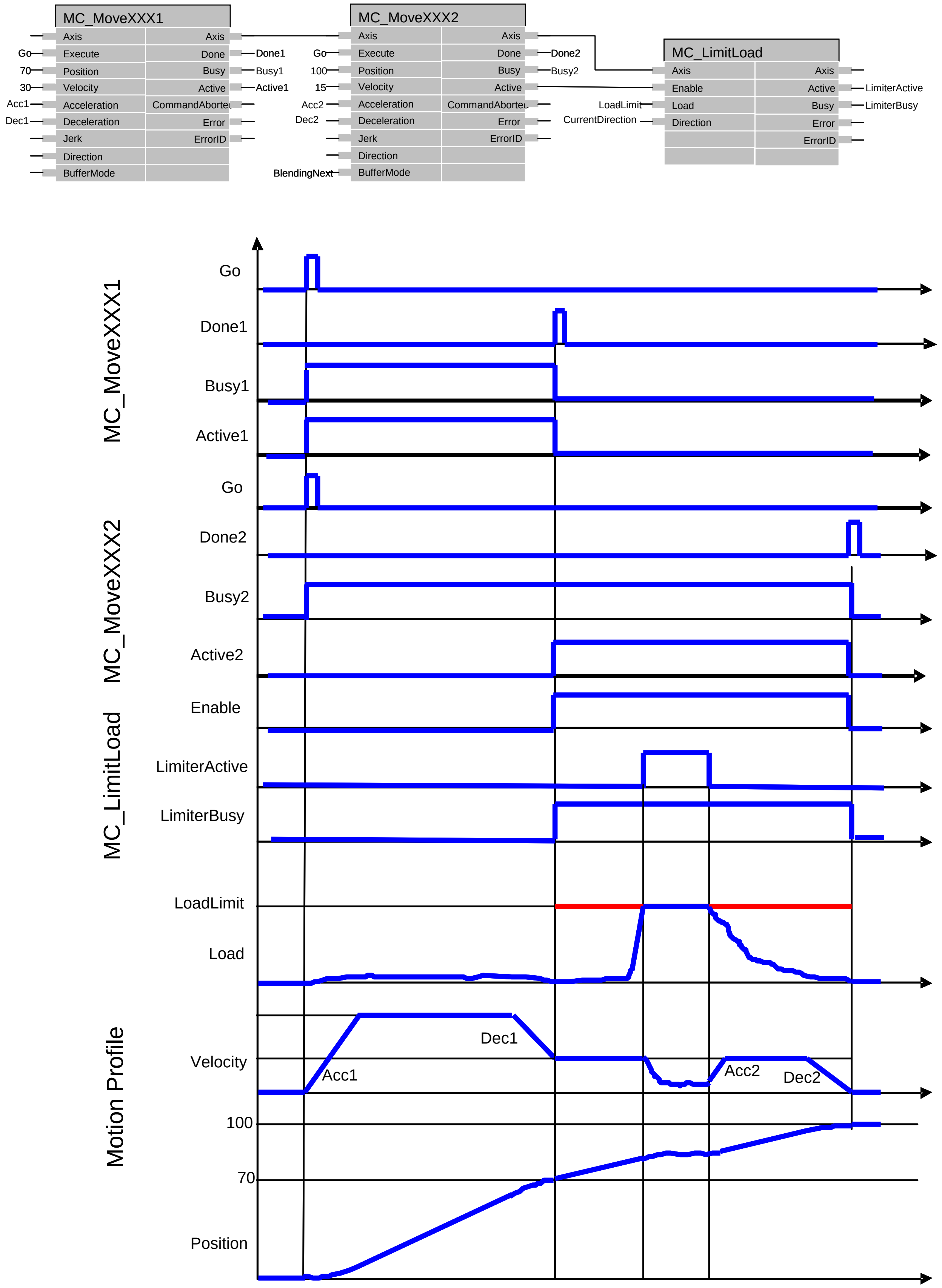


Figure 3: MC_LimitLoad Timing Diagram

Example: The combination with SFC

In the diagram below, an example is explained. SFC is used here to distinguish between a movement where the MC_LimitLoad functionality has become 'Active' or not. In Step 2 there is a movement like 'MoveAbsolute', which is limited by the MC_LimitLoad functionality. If the absolute position is reached without MC_LimitLoad becoming active, the transition via done to step 3 is applicable. However, if the MC_LimitLoad becomes 'Active', the transition to the 'Halt' step is applicable, issuing a MC_Halt.

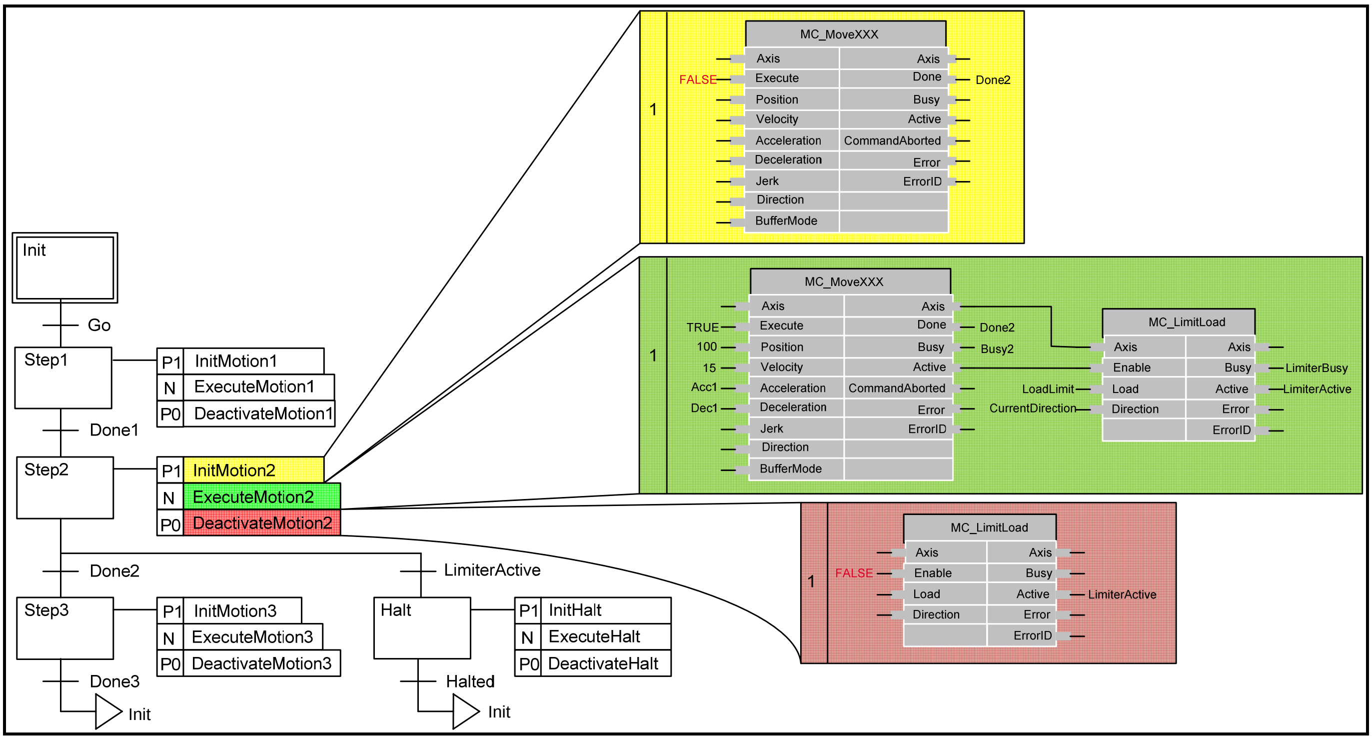
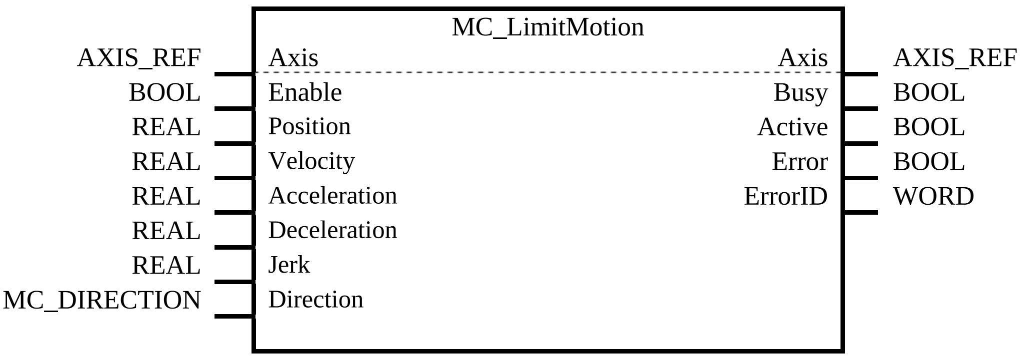


Figure 4: MC_LimitLoad used in SFC

5.3. MC_LimitMotion

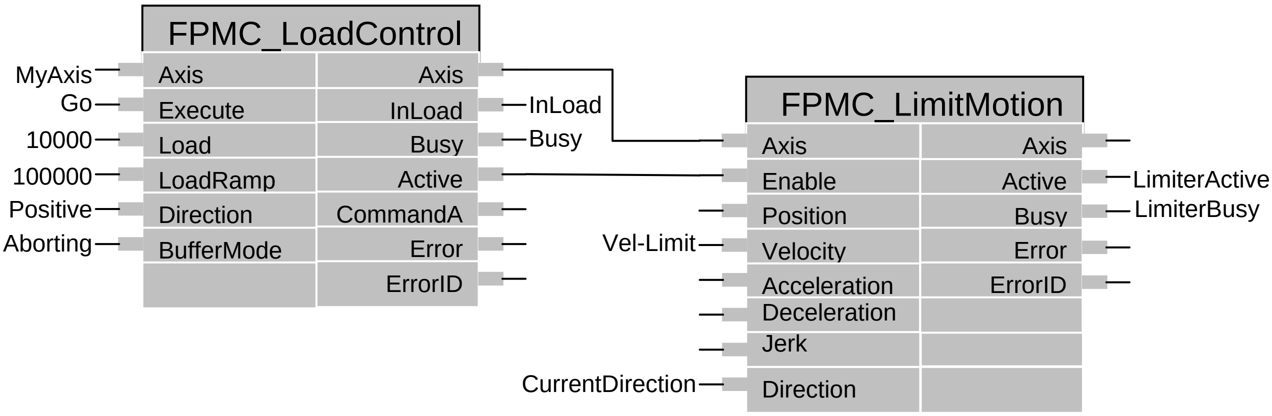
FB-Name		**MC_LimitMotion**	
This Function Block activates a limitation of the motion values of an axis. These are ‘Position’, ‘Velocity’, ‘Acceleration’, ‘Deceleration’ and ‘Jerk’. The measures taken to keep the limits are vendor specific; switching between motion and load control depends on the external load conditions of the axis. The FB sets the ‘Busy’ output when the limiting measures are stand-by on the axis. The ‘Active’ output is set, when the limiting measures are active on the axis.			
VAR_IN_OUT			
B	Axis	AXIS_REF	Reference to the axis
VAR_INPUT			
B	Enable	BOOL	Allows function block to modify (clamp) a load command
E	Position	REAL	Absolute position not to be crossed during load control
B	Velocity	REAL	Absolute value of the maximum velocity
E	Acceleration	REAL	Value of the maximum acceleration (acceleration is applicable with same sign of torque and velocity)
E	Deceleration	REAL	Value of the maximum deceleration (deceleration is applicable with opposite signs of torque and velocity)
E	Jerk	REAL	Value of the maximum jerk
E	Direction	MC_DIRECTION	Enum type (1 of 3 values: mcPositiveDirection, mcNegativeDirection and mcCurrentDirection) Note: shortest way not applicable.
VAR_OUTPUT			
E	Busy	BOOL	The FB is not finished and new output values are to be expected
B	Active	BOOL	Shows that limits in motion are reached and the Axis is out of load control and controls the commanded maximum motion values.
B	Error	BOOL	Signals that an error has occurred within the Function Block
E	ErrorID	WORD	Error identification
Note: • It is not guaranteed that activity of the limiting measures will be seen by the FB: a short pulse of the limited quantities could be over before the next Function Block cycle occurs.			



Use Case Rational

The Function Block MC_LimitMotion is intended to protect a *process* from undefined movements during load/torque control
Possible application: e.g. force fitting.
The FB is intended to be used in conjunction with a MC_LoadControl or MC_TorqueControl having primary control on the axis. The MC_LimitMotion should be enabled by the ‘Active’ output of the MC_LoadControl / MC_TorqueControl. If motion values on the axis exceed the given limit, appropriate measures are taken to keep to these limits, implying that the load/torque will not follow the programmed trajectory but depend on the external load conditions. However, the ‘Active’ output of the MC_LoadControl/MC_TorqueControl will stay TRUE in this case, following the modified PLCopen definition “The ‘Active’ output indicates, that the FB has control on *the set-value generation of* the axis”. This is despite the fact, that physically only the load-conditions *or* the movement of an axis can be controlled. With actual motion states below programmed limits, the programmed load/torque trajectory will proceed. Enabling the limiter block with activation of the MC_LoadControl/MC_TorqueControl ensures that limits are only supervised when the MC_LoadControl/MC_TorqueControl takes control on the axis for the first time. Disabling the limiter block with de-activation of the

MC_LoadControl/MC_TorqueControl ensures that limits are no more supervised when the MC_LoadControl/MC_TorqueControl looses control on the axis by ‘CommandAborted’ or ‘Error’



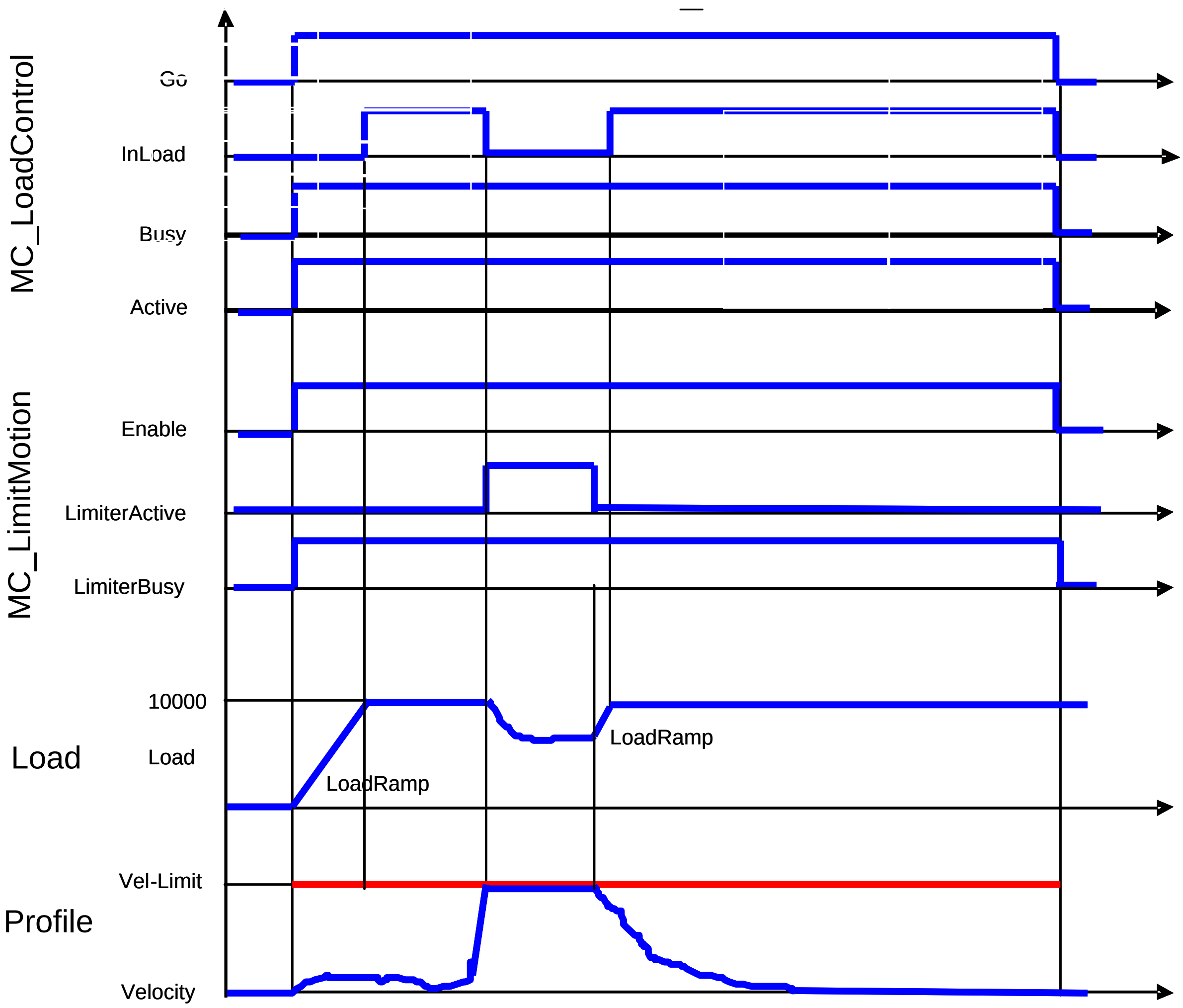


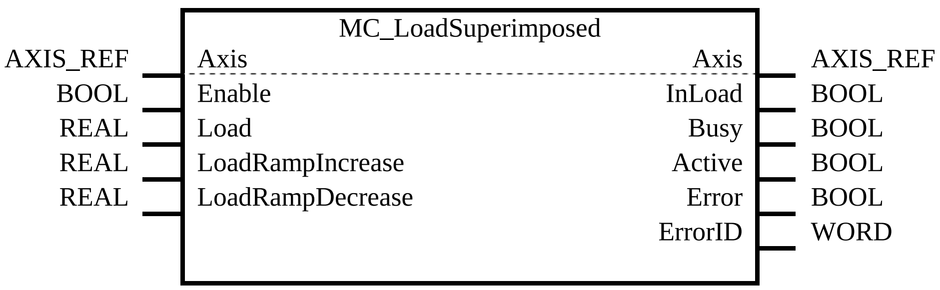
Figure 5: MC_LimitMotion Timing Diagram

5.4. MC_LoadSuperImposed

FB-Name		**MC_LoadSuperImposed**	
This Function Block commands a controlled load update (increase/decrease) of a specified relative value additional to an existing load. The existing load control operation is not interrupted, but is superimposed by the additional load.			
VAR_IN_OUT			
B	Axis	AXIS_REF	Reference to the axis
VAR_INPUT			
B	Enable	BOOL	Activate the motion while enabled
B	Load	REAL	Load that is to be superimposed (in technical unit [u])
E	LoadRampIncrease	REAL	Value of the load ramp increase of the additional load([u] per sec)
E	LoadRampDecrease	REAL	Value of the load ramp decrease of the additional load([u] per sec)
VAR_OUTPUT			
B	InLoad	BOOL	Additional superimposed load has been achieved
E	Busy	BOOL	The FB is not finished and new output values are to be expected
E	Active	BOOL	Indicates that this FB is contributing to the motion on the axis
B	Error	BOOL	Signals that an error has occurred within the Function Block
E	ErrorID	WORD	Error identification

Note:

- If MC_LoadSuperImposed is 'Active', then any other command in aborting mode except MC_LoadSuperImposed will abort both load commands: both the MC_LoadSuperImposed and the underlying load command. In any other mode, the underlying load command is not aborted
- If MC_LoadSuperImposed is 'Active' and another MC_LoadSuperImposed is commanded, only the on-going MC_LoadSuperImposed command is aborted, and replaced by the new MC_LoadSuperImposed command, but not the underlying load command
- The values of 'LoadRampIncrease' and 'LoadRampDecrease' are additional values to the on-going load control, and not absolute ones. With this, the underlying FB always finishes its job in the same period of time regardless of whether a MC_LoadSuperImposed FB takes place concurrently.
- The output 'Active' has a different behavior as in buffered FBs.



Use Case Rational
The function block MC_LoadSuperImposed is intended to allow a superimposed load command to be issued on top of an existing load command without superceding the original load command.

Possible Application: Actuator: hydraulic cylinder with fluid pressure sensor actuates the press of plastic injection molding machine in a continuous load operation.
Request: prior to MC_LoadSuperImposed call, a MC_LoadControl block is 'Active' with a command of 7,500 kPa to press melted plastic into the mold. Once the MC_LoadControl 'InLoad' condition is achieved a superimposed pressure of 5,000 kPa is added several times to cause a hammering effect to relieve stresses in the plastic.
Result: the MC_LoadControl pressure command of 7,500 kPa is superimposed with a discrete pressure command of 5,000 kPa. Once the 'LoadSuperImposed' command is active the system pressure rises to 12,500 kPa.
When the superimposed pressure command has been achieved the MC_LoadSuperImposed block is done and the original command given by the MC_LoadControl resumes the original pressure command.
The MC_LoadSuperImposed block is executed several times without affecting the original pressure command given by the MC_LoadControl block.

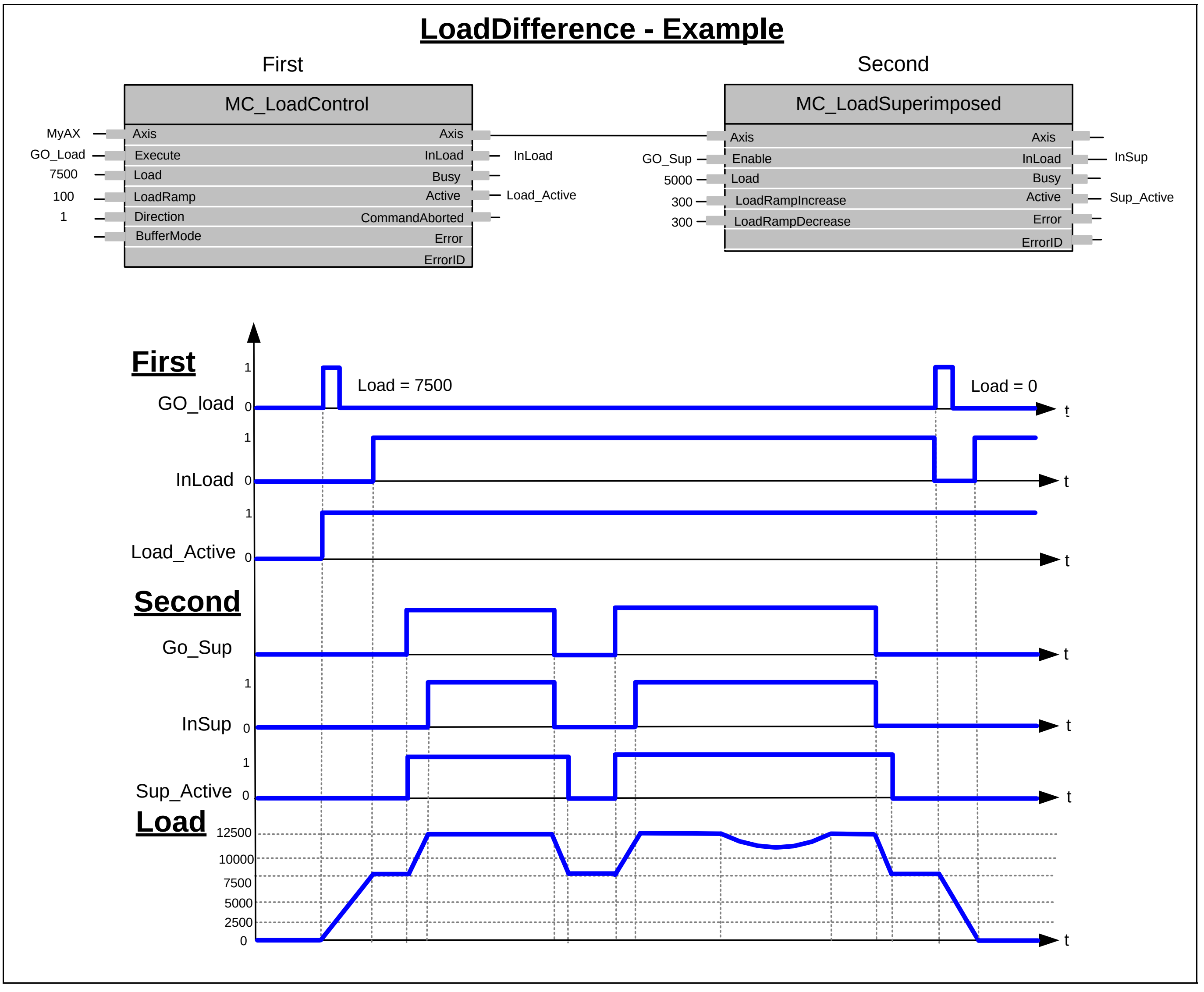


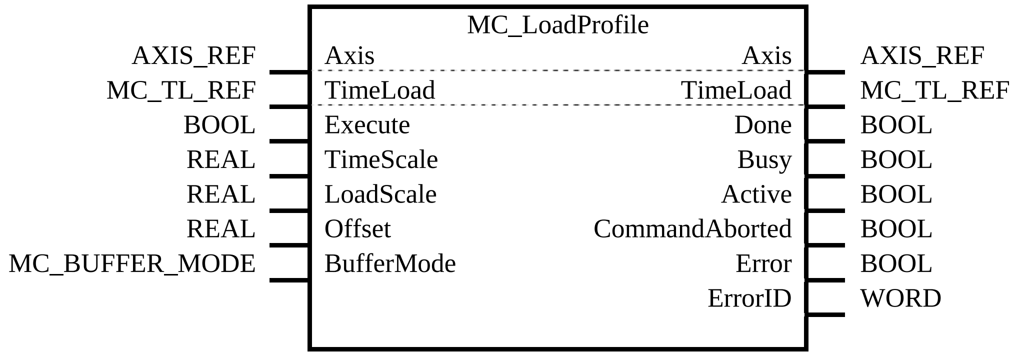
Figure 6: MC_LoadSuperImposed Timing Diagram

5.5. MC_LoadProfile

FB-Name		**MC_LoadProfile**	
This Function Block commands a time-load locked profile. The load in the final element in the profile should be maintained. The state remains 'ContinuousMotion'.			
VAR_IN_OUT			
B	Axis	AXIS_REF	Reference to the axis
B	TimeLoad	MC_TL_REF	Reference to Time / Load. Description - see note below
VAR_INPUT			
B	Execute	BOOL	Start the motion at rising edge
E	TimeScale	REAL	Overall time scaling factor of the profile
E	LoadScale	REAL	Overall load scaling factor of the profile
E	Offset	REAL	Overall offset for torque, force or pressure profile (in technical unit [u])
E	BufferMode	MC_BUFFER_MODE	Defines the chronological sequence of the FB. See 4.1.1 Buffered versus Non-buffered modes
VAR_OUTPUT			
B	ProfileCompleted	BOOL	Profile completed
E	Busy	BOOL	The FB is not finished and new output values are to be expected
E	Active	BOOL	Indicates that the FB has control on the axis
E	CommandAborted	BOOL	Command is aborted by another command
B	Error	BOOL	Signals that an error has occurred within the Function Block
E	ErrorID	WORD	Error identification

Note:

- MC_TL_REF is a supplier specific datatype. An example for this datatype is given here below:
  - The content of Time/Load pair may be expressed in DeltaTime/Load, where Delta could be the difference in time between two successive points.
  - **TYPE**
    MC_TL : STRUCT
    DeltaTime : TIME;
    Load : REAL;
    END_STRUCT
    END_TYPE
  - **TYPE**
    MC_TL_REF ; STRUCT
    NumberOfPairs : INT;
    MC_TL _Array : ARRAY [1..N] OF MC_TL;
    END_STRUCT
    END_TYPE
- This functionality does not mean it runs one profile over and over again: it can switch between different profiles.



Use Case Rational
This function block would be used to set a load profile curve in applications requiring a distinct load profile to effectively perform an operation. (e.g. plastic injection molding, metal forming, press, and other applications requiring time at load profiles for proper processing,)

MC_LoadProfile - Example

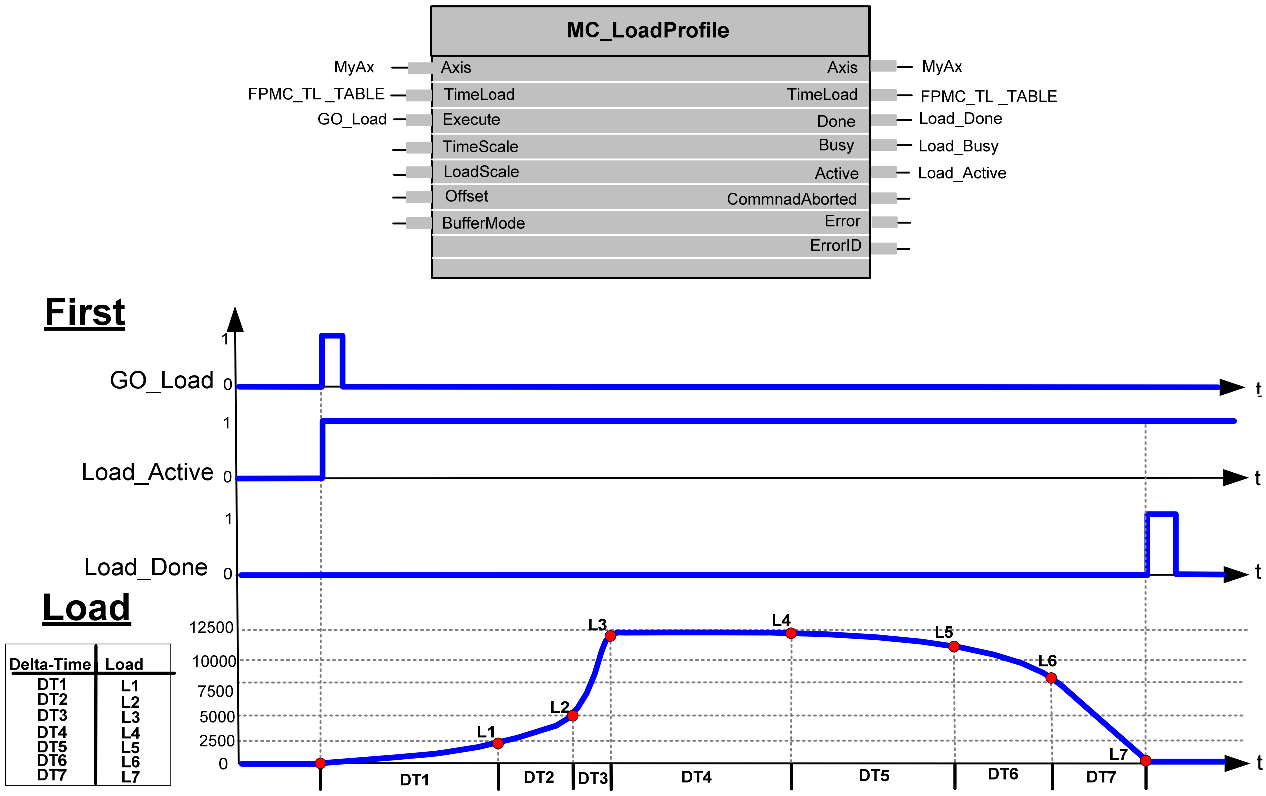


Figure 7: MC_LoadProfile Timing Diagram

Appendix A Compliance Procedure and Compliance List

Listed in this Appendix are the requirements for the compliance statement from the supplier of the Motion Control Function Blocks. This part should be seen as integral to Part 1 – Function Blocks for Motion Control.
The compliance statement consists of two main groups: supported datatypes and supported Function Blocks, in combination with the applicable inputs and outputs. The supplier has to fill out the tables for the used datatypes and Function Blocks, according to their product, committing their support to the specification.

By submitting these tables to PLCopen, as well as those from Part 1, and after approval by PLCopen, the list will be published on the PLCopen website, www.plcopen.org , as well as a short form overview, as specified in Appendix A 2. Supported Derived Datatypes and Appendix A 3. Overview of the Function Blocks as below.

In addition to this approval, the supplier is permitted access and usage rights to the PLCopen Motion Control logo, as described in Part 1, chapter Appendix A 4. - The PLCopen Motion Control Logo and Its Usage.

Appendix A 1. Statement of Supplier

Supplier name	
Supplier address	
City	
Country	
Telephone	
Fax	
Email address	
Product Name	
Product version	
Release date	

I hereby state that the following tables as filled out and submitted do match our product as well as the accompanying user manual, as stated above.

Name of representation (person):

Date of signature (dd/mm/yyyy):

Signature:

Appendix A 2. Supported Datatypes

Defined datatypes with MC library:	Supported	If not supported, which datatype used
BOOL		
WORD		
REAL		

Within the specification the following derived datatypes are defined. Define which of these structures are used in this system:

Derived datatypes:	Where used	Supported	Which structure
AXIS_REF	All FBs		
MC_DIRECTION	MC_LoadControl MC_LimitLoad MC_LimitMotion		
MC_BUFFER_MODE	MC_LoadControl MC_LoadProfile		
MC_TL_REF	MC_LoadProfile		

Table 5: Supported derived datatypes

Appendix A 3. Overview of the Function Blocks

Single Axis Function Blocks	Supported Yes / No	Comments (<= 48 char.)
MC_LoadControl		
MC_LimitLoad		
MC_LimitMotion		
MC_LoadSuperImposed		
MC_LoadProfile		

Table 6: Short overview of the Function Blocks

Appendix A 3.1 MC_LoadControl

If Supported	**MC_LoadControl**	Sup.Y/N	Comments
VAR_IN_OUT			
B	Axis		
VAR_INPUT			
B	Execute		
E	ContinuousUpdate		
B	Load		
E	LoadRamp		
E	Direction		
E	BufferMode		
VAR_OUTPUT			
B	InLoad		
E	Busy		
E	Active		
E	CommandAborted		
B	Error		
E	ErrorID		

Appendix A 3.2 MC_LimitLoad

If Supported	**MC_LimitLoad**	Sup.Y/N	Comments
VAR_IN_OUT			
B	Axis		
VAR_INPUT			
B	Enable		
B	Load		
E	Direction		
VAR_OUTPUT			
B	Busy		
E	Active		
B	Error		
E	ErrorID		

Appendix A 3.3 MC_LimitMotion

If Supported	**MC_LimitMotion**	Sup.Y/N	Comments
VAR_IN_OUT			
B	Axis		
VAR_INPUT			
B	Enable		
E	Position		
B	Velocity		
E	Acceleration		
E	Deceleration		
E	Jerk		
E	Direction		
VAR_OUTPUT			
B	Busy		
E	Active		
B	Error		
E	ErrorID		

Appendix A 3.4 MC_LoadSuperImposed

If Supported	**MC_LoadSuperImposed**	Sup.Y/N	Comments
VAR_IN_OUT			
B	Axis		
VAR_INPUT			
B	Enable		
B	Load		
E	LoadRampIncrease		
E	LoadRampDecrease		
VAR_OUTPUT			
B	InLoad		
E	Busy		
E	Active		
B	Error		
E	ErrorID		

Appendix A 3.5 MC_LoadProfile

If Supported	**MC_LoadProfile**	Sup.Y/N	Comments
VAR_IN_OUT			
B	Axis		
B	TimeLoad		
VAR_INPUT			
B	Execute		
E	TimeScale		
E	LoadScale		
E	Offset		
E	BufferMode		
VAR_OUTPUT			
B	ProfileCompleted		
E	Busy		
E	Active		
E	CommandAborted		
B	Error		
E	ErrorID		

Appendix A 4. The PLCopen Motion Control Logo and Its Usage

For quick identification of compliant products, PLCopen has developed a logo for the Motion Control Function Blocks:

Figure 8: The PLCopen Motion Control Logo

This motion control logo is owned and trademarked by PLCopen.

In order to use this logo free-of-charge, the relevant company has to fulfill all the following requirements:

1. the company has to be a voting member of PLCopen;
2. the company has to comply with the existing specification, as specified by the PLCopen Task Force Motion Control, and as published by PLCopen, and of which this statement is a part;
3. this compliance application is provided in written form by the company to PLCopen, clearly stating the applicable software package and the supporting elements of all the specified tables, as specified in the document itself;
4. in case of non-fulfillment, which has to be decided by PLCopen, the company will receive a written statement concerning this from PLCopen. The company will have a one month period to either adopt their software package in such a way that it complies, represented by the issuing of a new compliance statement, or remove all reference to the specification, including the use of the logo, from all their specification, be it technical or promotional material;
5. the logo has to be used as is - meaning the full logo. It may be altered in size providing the original scale and color setting is kept.
6. the logo has to be used in the context of Motion Control.