

Application Note

MP2000iec Application Design Guideline

Rev 2.00

Applicable Product: MP2000iec, MotionWorksIEC

Relevant Firmware/Software/Function Block Library

Item	Details
Controller Firmware version	1.2.2.9
IDE (Software version)	1.2.2.7
Function Block Library	PLCOpenPlus v-2.2.a

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

- 1. Summary and Scope 4
- 2. Motion Functionality 5
 - 2.1 Motion Function Blocks 5
 - 2.1.1 ENABLE AXIS:..... 5
 - 2.1.2 JOGGING: 5
 - 2.1.3 HOMING: 5
 - 2.1.4 POINT TO POINT MOTION:..... 8
 - 2.1.5 HIGH SPEED REGISTRATION: 12
 - 2.1.6 GEARING: 14
 - 2.1.7 PHASING:..... 15
 - 2.1.8 CAMMING:..... 15
 - 2.1.9 MONITORING:..... 19
 - 2.1.10 MAINTENANCE:..... 19
 - 2.2 Function Block Performance..... 23
 - 2.3 MECHATROLINK Update Rate 26
 - 2.4 Typical PLC Scan Times 26
- 3. Communication: MODBUS/TCP, Ethernet/IP & OPC..... 29
 - 3.1. MP2000iec as MODBUS TCP Client (Master)..... 30
 - 3.2 MP2000iec as MODBUS TCP Server (Slave) 32
 - 3.3 MP2000iec as EtherNet/IP scanner..... 33
 - 3.4 MP2000iec as EtherNet/IP adapter 34
 - 3.5 OPC Communication..... 35
 - 3.6 Heartbeat and Watchdog implementations for ensuring healthy communication when the MP2000iec controller is an adapter/server..... 35
- 4. Supported Devices..... 38
 - 4.1 MECHATROLINK 38
 - 4.1.1 MECHATROLINK Axes 38
 - 4.1.2 MECHATROLINK I/O 39
 - 4.2 Other I/O Devices 39
 - 4.2.1 Onboard devices..... 39
- 5. Memory Capacity 40
- 6. Applications List..... 40
- 7. Solution Packages and Template Code..... 41
 - 7.1 Tool Boxes..... 41
- 8. References 43
- 9. MP2600iec..... 44
 - 9.1 Differences: MP2300Siec/MP2310iec vs. MP2600iec..... 44

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

9.2	Hardware combinations	45
9.3	MP2600iec Performance	46
9.4	Sample Applications for the MP2600iec	46
9.5	Sample application performance metrics for MP2600iec	47
9.6	MP940 with MotionWorks+ vs MP2600iec with MotionWorksIEC: Execution comparison	48
9.7	MP940 → MP2600iec field conversion benchmarks	49
10	Revision History	50

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

1. Summary and Scope

The purpose of this design guideline is to recommend best practices for applying the MP2000iec and to aid in determining if the MP2000iec product is a good fit for an application during the sales process. If you feel your requirements are outside the metrics specified in this guideline, please call to discuss other possible solutions or recommendations we may have to solve the application. This guideline is not a replacement for the user manuals. Its purpose is to provide insight as to the differences between the standard MP controller behavior and the MP2000iec, and give the reader expectations regarding product performance based on the application criteria.

As new firmware and software versions are released, this design guideline will be updated accordingly. Revision history will always include the firmware and software version applicable.

Terms used in this document

Wherever possible we have conformed to the terms specified by the governing body responsible for such technology. Below is a list of definitions, and their source.

Term	Definition	Source
Drive	Generic Term used to identify any type of motor power source, which could be servo, VFD, stepper, etc.	Yaskawa
Servopack	Name given to Yaskawa Sigma Series servo amplifiers	Yaskawa
FMK	Flexible Motion Kernel (Motion Trajectory engine) SVB Counterpart)	Yaskawa
SVB	MECHATROLINK Motion Master (FMK counterpart)	Yaskawa
Scanner	Describes a master or initiator of communications using the EtherNet/IP protocol	ODVA
Adapter	Describes a slave or receiver of communications requests using the EtherNet/IP protocol.	ODVA
Client	Describes a master of communications using the MODBUS protocol	Modicon
Server	Describes the communications slave to a client using the MODBUS protocol	Modicon

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

2 Motion Functionality

2.1 Motion Function Blocks

2.1.1 ENABLE AXIS:

MC_Power: This Function Block enables or disables the servopack. On enabling the axis, the axis state changes from 'disabled' state to 'standstill' with a commanded velocity of zero. This function block does not check to verify if the servo is ready or if the servopack is alarm free. If the function block is unable to switch states between disabled and standstill, an error is reported as an output of the function block. A safe enable routine will have to be incorporated by the programmer depending on the application. Enable_Positive and Enable_Negative inputs are not active.

Note: If the servopack is made to go into hardware base block state (HBB), it is recommended that the user manually clears the HBB state and re-enables power using MC_Power. It is not recommended to have only a normally closed contact for the status of the Hardware Base Block (HBB) circuit to enable the MC_Power block. If the servopack goes into an HBB state in the same scan a 'servo on' command is issued over MECHATROLINK, the axis will get stuck in an error state mode and will be unresponsive.

2.1.2 JOGGING:

MC_MoveVelocity: This Function Block commands controlled motion at the specified velocity. The axis controlled by the MC_MoveVelocity will be in position mode although a continuous velocity move is being executed. The InVelocity output bit stays on as long as long as the axis is commanded the specified velocity by the controller.

MC_Stop: This Function Block commands a controlled motion stop and transitions the axis to the 'Stopping' state. If the execute bit of an MC_Stop for an axis is in a high state, any other move (discrete or continuous) on that axis is prevented.

2.1.3 HOMING:

Among homing methods, the most popular three methods are listed below. I) Home to c-pulse, ii) Home to limit switch, iii) Home to hard stop

MC_StepRefPulse: This function Block performs homing by searching for Zero pulse (also called Marker or

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

reference pulse) in the encoder on all Sigma-2, 3 and 5 Series rotary servos.

Note: *Torque Limit input on MC_StepRefPulse is not supported. A slow velocity input (1 rev/sec) for MC_StepRefPulse is recommended.*

MC_StepLimitSwitch: This function Block performs homing by searching for the POT (positive over travel) and NOT (negative over travel) switches that are wired to the CN1 channel on the servopack.

Note: *Turn off controller parameter 1310 (feed forward velocity) using MC_WriteBoolParameter before exiting an over travel limit. Once the axis is clear of the over travel limit, turn on controller parameter 1310.*

Note: *For vertical axes, make sure Pn001.1 is set to hold torque on limit switch trigger*

MC_TorqueControl: This function block continuously exerts a torque or force of the specified magnitude. Use Y_HoldPosition and MC_Stop to finish torque mode operation.

Y_HoldPosition: This function block brings an axis from any mode to position mode instantaneously. The function block uses the position at the beginning of the scan and executes it at the next MECHATROLINK scan. Therefore it is recommended that the Y_HoldPosition be used only when the axis is stationary. Any use of the Y_HoldPosition while the axis is in motion can cause mechanical shudder on the axis.

MC_SetPosition: This Function Block shifts the coordinate system of an axis by changing both the commanded position as well as the actual position of an axis with the same value without any movement caused. If there is position error on the axis (commanded position – actual position), the error is retained. In such a case, MC_SetPosition sets the commanded position to the user defined position and actual position will be offset by the position error that was present before the set position was implemented.

Note: To home to a hard stop, the two recommended methods are:

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

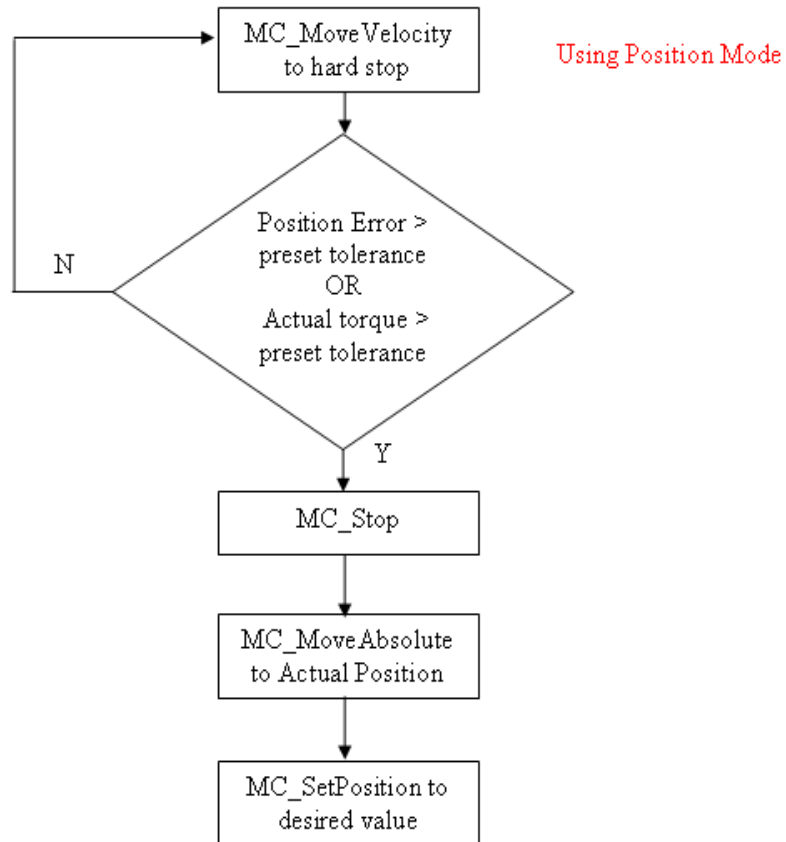


Figure 1: Home to hard stop in position mode

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

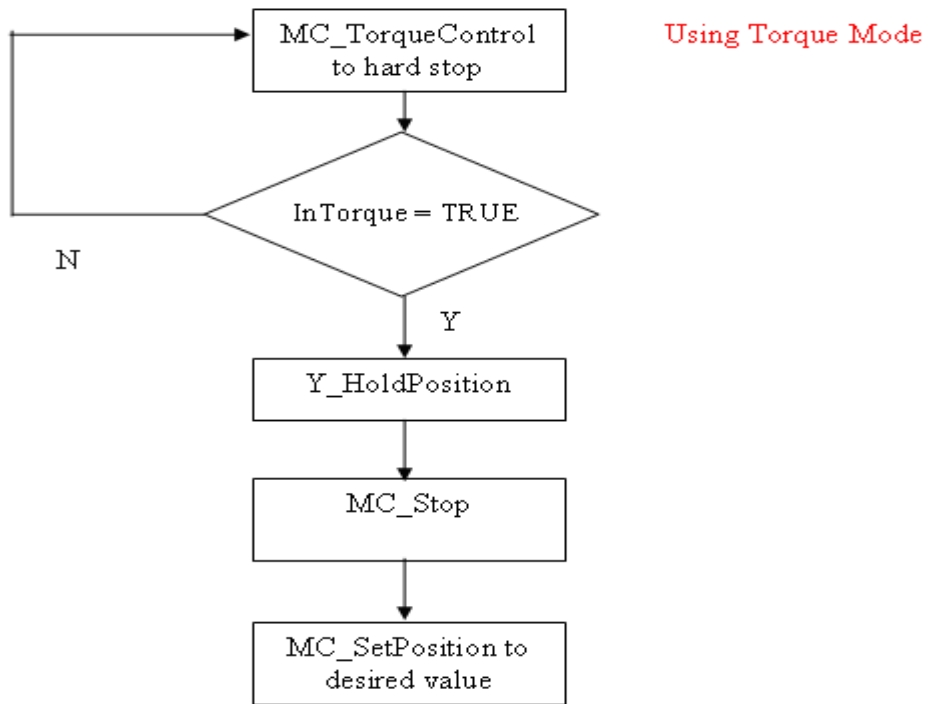


Figure 2: Home to hard stop in torque mode

2.1.4 POINT TO POINT MOTION:

Motion in linear and rotary modes can be obtained using the following blocks

MC_MoveRelative: This Function Block commands a controlled motion of the specified distance relative to the commanded position at the time of the execution.

MC_MoveAbsolute: This Function Block commands a controlled motion to the specified absolute position.

Note:

The done output bit on a motion function block turns on when the motion profiler in the controller has completed its last command. The motion blocks do not wait for a signal from the servopack saying the servo is within the 'In Position' (COIN) window for setting the Done output bit on a motion function block high. This can affect untuned systems or gantry systems with parallel axes where the two axes are tuned to different levels. If the user wishes to ensure 'motion done' based on feedback, the position error parameter 1130 on MC_ReadParameter will have to be used in conjunction with logic in the application.

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

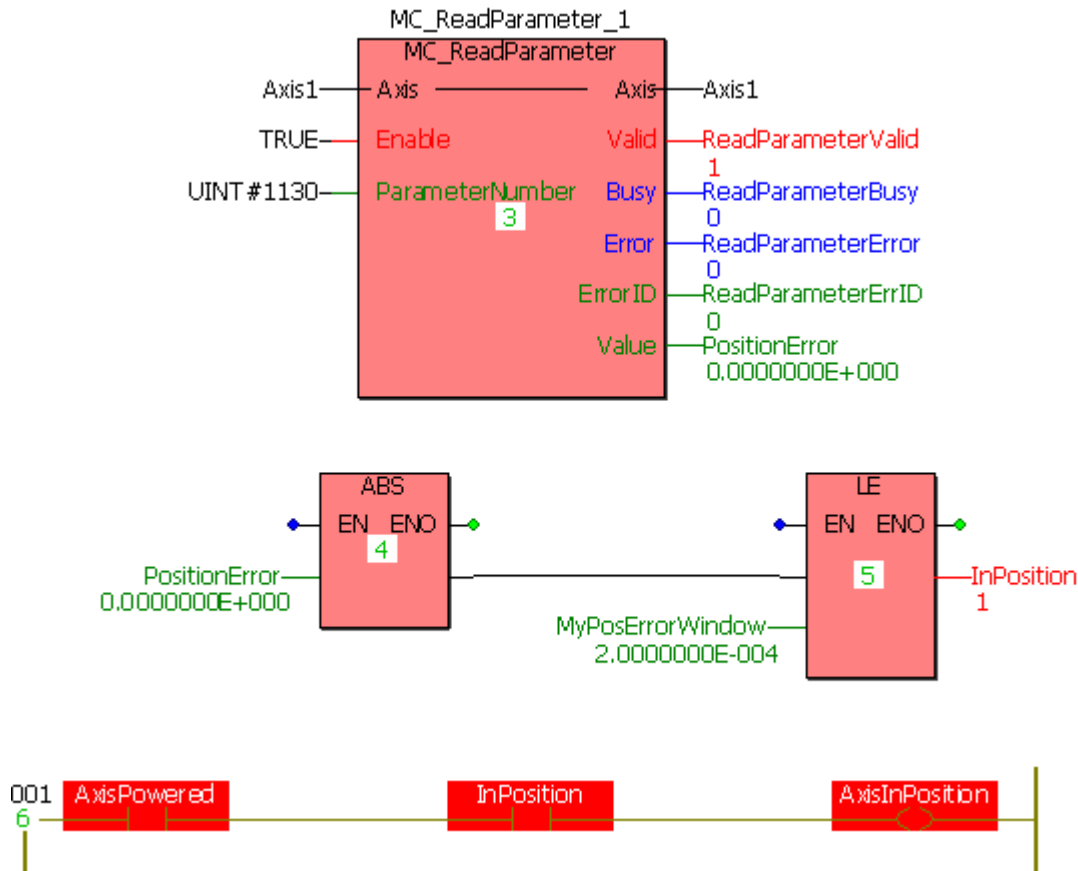


Figure 3: Recommended 'InPosition' bit implementation

S-curve functionality

S-curve functionality for discrete motion profiles is available in firmware version 1.1.2.5 by implementing a moving average filter. To use the moving average filter, the user will have to enable it through the Configuration Tool under the configuration tab for the MECHATROLINK axis. The filter time constant can also be specified in the same tab. (Figure 4a). The filter is implemented as a lag filter as shown in Figure 4b.

Note: If rotary shortest path is being used to command motion on a rotary axis, and the moving average filter (parameter 1300) is being used, MC_Stop issued while the axis is at the rotary rollover position will cause unexpected motion on the axis.

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

Parameter #	Parameters	Current Value	Units	Min	Max	Default Value
1007	Load Type	Rotary		0	1	Linear
1031	Logical Axis Number	1		1	512	
1300	Moving Average Filter 1 Enable	False				False
1301	Moving Average Filter 1 Time Constant	0.1	s	0	1	0.1

← Set to TRUE

Figure 4a: Moving average filter for s curve functionality

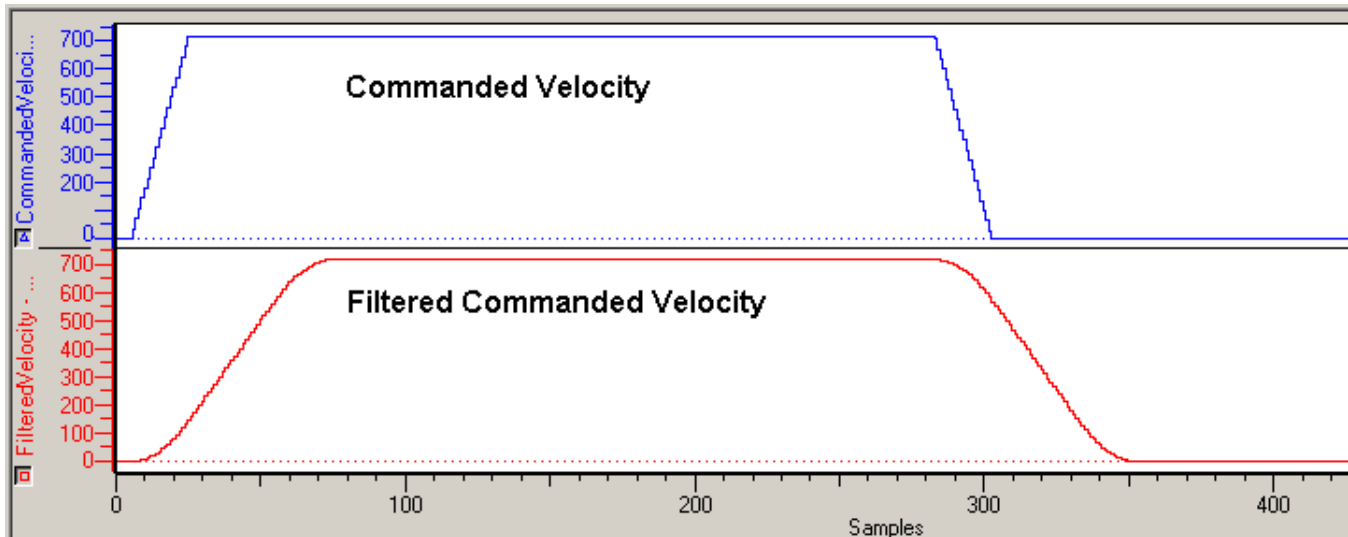


Figure 4b: Commanded velocity profiles

Motion moves can be aborted, buffered or blended. Sixteen moves can be buffered in the FMK memory. Moves can also be blended without the axis coming to a stop. Blending can be performed in four different ways depending on the demands of the application. Acceleration of the second function block is used to transition from one velocity to another even if the second velocity is smaller than the first.

On the MP2300Siec and the MP2310iec controllers, Pns 811 and 812 filter the command position calculated by the drive. The effect of Pns 811 and/or 812 can be seen by logging Position Reference Speed in Sigma Win+.

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

With Pn 812=0, Position Reference Speed is a stair step at the MECHATROLINK update. Pn 812 = MECHATROLINK update, the drive interpolates and the stair step is now at the drive position loop frequency. New configuration feature property in the Configuration Tool: "1311: Drive Motion Command OPTION ACCFIL Value (0, 1, and 2)". By default, for backward compatibility, this parameter is off. Also, it makes no attempt to set Pn810, Pn811, or Pn812. In order to use the exponential filter, set controller parameter 1311 to 1 and Pn 811 to the MECHATROLINK update rate. In order to use the s curve filter, set controller parameter 1311 to 2 and Pn 812 to the MECHATROLINK update rate.

riD4370

MyMachine

- Mechatrolink-II
 - SGDV Rotary - 1
 - SGDV Rotary - 2
- TCP/IP Settings
- EtherNet/IP
- Modbus/TCP
- LIO-01
- Counter

Parameter #	Parameters	Current Value	Units	Min	Max	Default Value
1014	Commanded Torque/Thrust	0	%			
1015	Output Ratio	1		1	21474	1
1016	In Position Window	50	mm	1E-05	5000C	50
1017	At Speed Window	50	mm/s	1E-05	5000C	50
1018	Node Number	1				
1019	Amplifier Model	SGDV-R90F11A				
1020	Version	32				0x00
1021	Amplifier Type					
1022	Amplifier Power		Watts			
1023	Motor Model	SGMJV-01A3A61				
1024	Motor Type					
1025	Motor Power		Watts			
1026	Encoder Type					
1027	Encoder Resolution	0	Encoder l	1024	53687	
1028	Network Number	1				1
1031	Logical Axis Number	1		1	512	
1033	Machine Cycle	1	mm	1	1000C	1
1034	Input Ratio	1		1	21474	1
1035	Acceleration/Deceleration	0	mm/s ²			0
1036	Motor Rated Torque		Nm			
1037	Motor Rated Speed		RPM			
1048	Drive Type	0				0
1200	Reverse Position Limit	-1.797693E+308	mm	-1.797	1.797	-1.797693E+308
1201	Forward Position Limit	1.797693E+308	mm	-1.797	1.797	1.797693E+308
1300	Moving Average Filter 1 Enable	False				False
1301	Moving Average Filter 1 Time Constant	0.1	s	0	1	0.1
1310	Velocity Feed Forward Enabled/Disabled	Enabled				Enabled
1311	Drive Motion Command OPTION ACCFIL	0: No acceleration/decele				0: No acceleration/decele

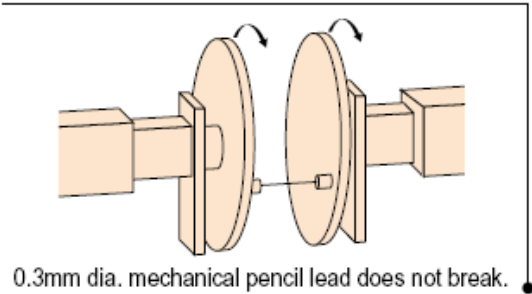
Figure 4c: Enabling acceleration and deceleration filters at drive level

The various control modes available are shown in Figure 4d: Position, Velocity, Phasing and Torque. MC_MoveVelocity is implemented in position mode. Phase offset is performed by MC_MoveSuperImposed. MC_TorqueControl implements torque control in true torque mode. It is possible to switch between all modes on the fly via the application program.

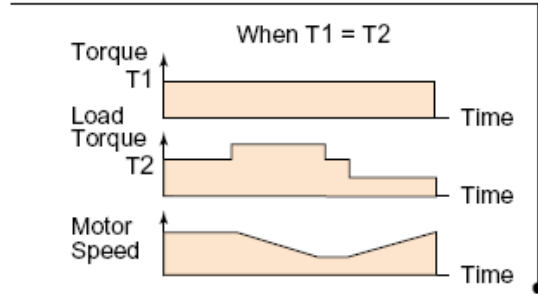
Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

Four Control Modes All-in-one

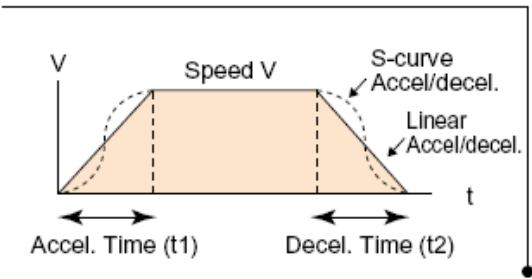
- **Synchronous Phase Control**
 Not applicable for applications using motion control from a PC.
 Speed control with position compensation (electronic shaft) or position control with 100% speed feed forward (electronic cam). Multi-axis servomotors can be controlled synchronously.



- **Torque Control**
 Generates a constant torque, regardless of speed.



- **Position Control**
 Advances to the target position, and stops or holds.



- **Speed Control**
 Turns the motor at the specified speed, with user-defined acceleration/deceleration slopes.

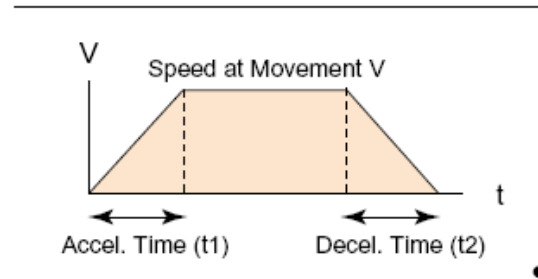


Figure 4d: Control modes available

2.1.5 HIGH SPEED REGISTRATION:

MC_TouchProbe: The function block will output the axis position when a trigger event occurs. The response time of the input depends on the hardware

Note:



Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

The MC_TouchProbe function block can capture position of servo axes using the EXT1, EXT2, or EXT3 switches on the CN1 channel of the servopack at much faster rates compared to the program scan. Positions of external encoders can be captured if the external encoder is wired to the LIO card. Latch speeds are shown below

SGDV (M-II)

5 us to 15 us at constant speed *

SGDS, SGDH

(Rising)	Typical	Worst
+24V	2.67us	15.23us
+12V	5.34us	31.84us

Measuring example: 10us

(Falling)	Typical	Worst
+24V	46.72us	266.51us
+12V	93.45us	557.25us

Measuring example: 100us

MP2000 LIO-01

	ON	OFF
DI	60us	0.5ms
Z (5V, 12V)	1us	1us

MP2000 LIO-06

	ON	OFF
DI	60us	0.5ms
Z (5V, 12V)	1us	1us

**these numbers indicate total throughput. Although the hardware may be able to accomplish 2 microseconds, full throughput to latch position is 5 microseconds at normal temperature and constant speed. If speed is varying greatly or ambient temperature rises to max of 55C, then latch throughput time could be up to 15 microseconds.*

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

In order to capture rising and falling edges (to capture position of the servo motor) on one sensor, the sensor will have to be wired to EXT1 and EXT2 on the CN1 channel of the servopack. The servopack will have to be configured to use rising edge detection on one input and falling edge detection on the other input. It should be remembered that there is only one memory location for the latched position. So, if the falling edge detection takes place before the MECHATROLINK data transfer for the first position takes place, the first position gets overwritten. The controller will not get the first rising edge position. It is important to design this application based on MECHATROLINK update rate, speed of the axis and time between the two trigger conditions. (Registration detection, read and rearm cycle time = 2 Application scans max. [This can be reduced to one application scan. Contact Motion Applications for further details]). Continuous latching feature of Sigma-5 is not supported.

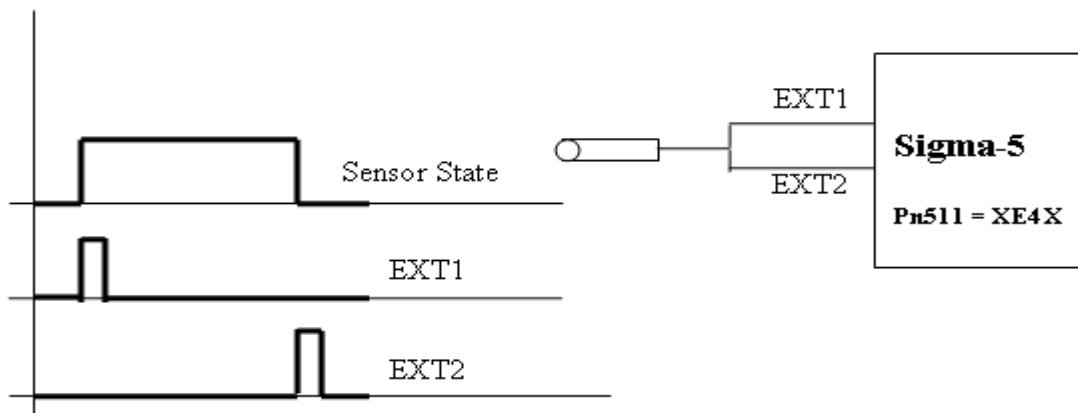


Figure 5: Rising/Falling edge detection for registration

MC_AbortTrigger: The Function Block aborts function blocks which are associated with trigger events

Note: If MC_AbortTrigger is used in a high speed task (= 4ms) it can cause a watchdog time out

2.1.6 GEARING:

Velocity gearing (non-rigid) and position gearing (rigid) are available by using the following function blocks.

MC_GearIn: This Function Block commands a ratio between the VELOCITY of the master and slave axes. MC_GearIn gears to commanded position when in motion and actual position when not in motion

Note: To ensure that all slaves are commanded the same target velocity and have no position offset relative to each other, execute MC_GearIn for all slaves before the master starts motion. If the master is in motion and the slaves are geared in, the slaves may have an offset relative to each other even though all the slaves were

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

geared in the same application scan. This is explained in section 2.4 of this document

MC_GearInPos: This Function Block commands a gear ratio between the POSITION of the master and slave axes. Synchronization is achieved over a defined region of travel for both master and slave.

Note: *MC_GearInPos will work only if it is being used to accelerate a slave to make it catch up to a master that has moved past the slave's position. MC_GearInPos does not support axes (master or slave) in rotary mode. Contact Yaskawa to enquire if camming can be used to accomplish the required motion.*

MC_GearOut: This Function Block disengages the Slave axis from the Master axis. The slave will continue to move at the last commanded velocity. MC_Stop has to be used to bring the slave axis to a standstill.

2.1.7 PHASING:

MC_MoveSuperImposed: This Function Block commands a controlled motion of the specified relative distance additional to an existing motion. The existing Motion is not interrupted, but is superimposed by the additional motion. This function block is valid for all motion blocks since this is implemented in position mode.

2.1.8 CAMMING:

Note: *Cam command profiles are generated at MECHATROLINK scan rates. Linear Interpolation is used to calculate cam slave positions at the FMK level for motion profile generation. Multiple cam tables can be held in the FMK memory and switching between these tables can be done on the fly. The memory size is specified in section 5 of this document.*

Y_CamFileSelect: This function block loads a cam table from a file. This function block returns a cam table ID which can be used to engage a slave axis to a master. Switching can happen as fast as one scan. However, care should be taken in the application program to ensure jerk does not happen by switching when slave positions are identical in the tables. The cam file (eg. 'Camdata.csv') can be saved (downloaded) in the controller from the PC using MotionWorksIEC using download file feature. An example is shown in Figure 6. Six significant digits for each point in the position array are recommended for a smooth cam profile. Any less number of significant digits can cause rough cam motion.

Y_CamStructSelect: This function block loads a cam table from the application memory area to the motion memory area and returns a CamTableID to be referenced when activating the Cam function

Y_CamIn: This Function Block engages the axis in camming mode with the cam profile specified by CamTableID.

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

Table 1: Cam Engage Options

Cam Engage Options	Status*
StartMode: Immediate	S
StartMode: InPosition	S
StartMode: Linked	S
Master Relative	S
SlaveAbsolute	S
Rampln	U

*S: Supported, U: Unsupported, F: Future

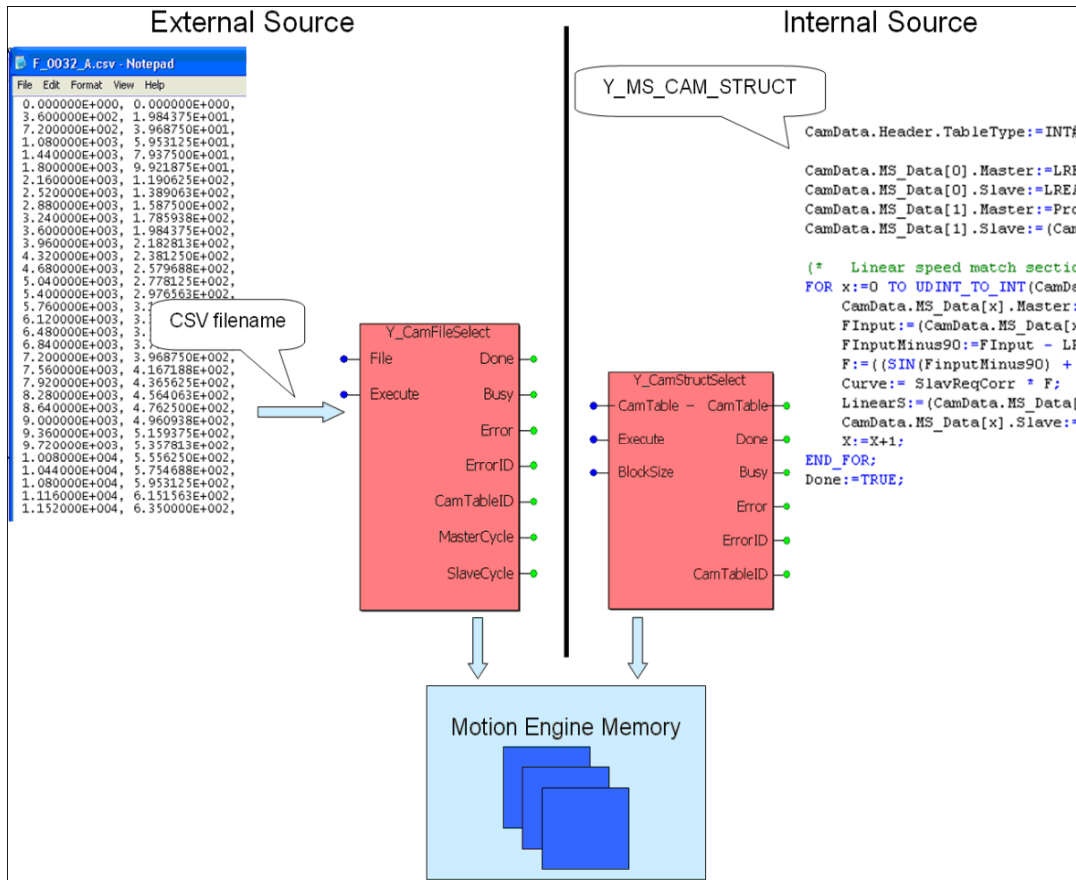


Figure 6: Two methods of loading cam profiles

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

Note: *MotionWorksIEC uses csv formatted files to import cam tables into the controller. Yaskawa recommends 'Cam Tool' or Microsoft Excel to create cam profiles and save in csv format.*

YCamOut: This Function Block disengages a Slave axis from its Master axis.

Table 2: Cam Disengage Options

Cam Disengage Options	Status*
EndMode: Immediate	S
EndMode: InPosition	S
EndMode: End of Profile	S
Ramp Out	U

*S: Supported, U: Unsupported, F: Future

Y_CamScale: This Function Block multiplies cam slave position data derived from the cam table by a scale factor. This can be used on reciprocating cam applications where stroke length is variable.

Y_CamShift: This Function Block dynamically modifies the master - slave relationship by adding a perceived offset to the master position, effectively causing the slave to advance or retard from the originally specified synchronization data in the cam data table.

Table 3: Cam Adjust Modes

Adjust Mode Options	Status*
Master Distance	S
Elapsed Time	S
Range	S

*S: Supported, U: Unsupported, F: Future

Y_SlaveOffset: This Function Block applies an offset to the slave position. This can be used when individual slave positions need phase adjustment. Eg. Slave axis set up to adjust wear and tear.

Y_WriteCamTable: This Function Block copies cam data from the application program memory into the motion memory.

Y_ReleaseCamTable: This Function Block frees memory in the motion area currently allocated for a cam table

Y_ReadCamTable: This Function Block copies a cam table from the motion memory into the application program memory

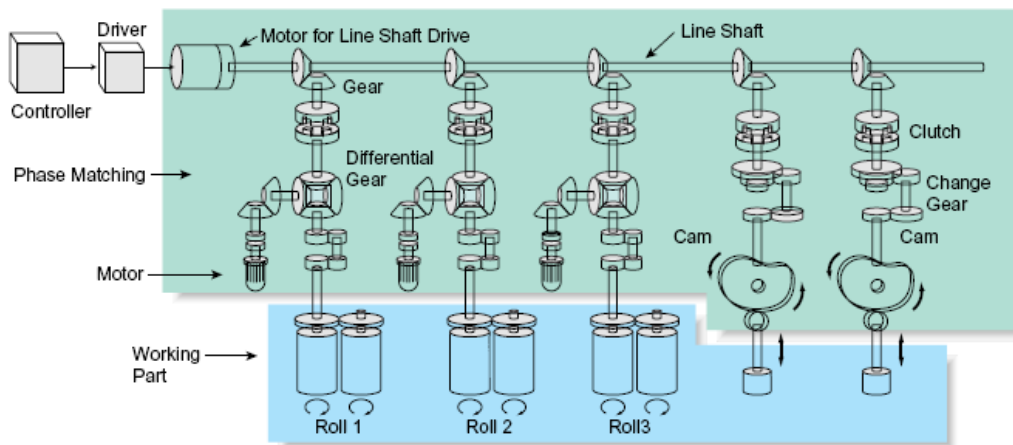
Note: *A manual on how to create a cam profile in Cam Tool and how to create a simple camming application in MotionWorksIEC is given on the website under the literature section for MP2000iec[4]: EM.MCD.09.043, EC.MCD.09.044. An introductory camming technical webinar has been posted under partner login on*

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

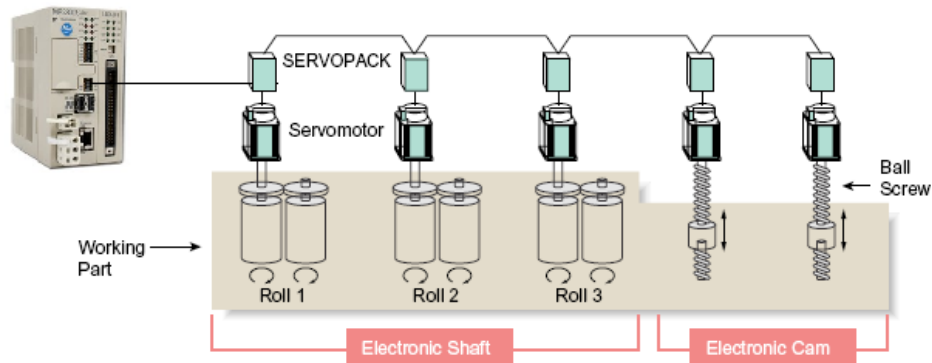
Yaskawa.com. The link to the document is
[https://partner.yaskawa.com/site/dmsearch.nsf/sitesearch?openagent&query=\(camming%20presentation\)](https://partner.yaskawa.com/site/dmsearch.nsf/sitesearch?openagent&query=(camming%20presentation))

Electronic Shaft and Electronic Cam for Synchronous Phase Control

Conventional Method



Synchronized Operation by MP2300Siec



AC servomotors move the same way as with rollers on the line shaft.

AC servomotors move the same way as with the mechanical cam.

Figure 7a: Electronic cam functionality

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

2.1.9 MONITORING:

MC_ReadStatus: This Function Block returns in detail the status of the axis, *i.e.* PLCOpen states. (Standstill, Discrete motion, Continuous motion, Accelerating, Decelerating, Error Stop, Homing, Synchronous motion, Disabled)

MC_ReadActualPosition: This Function Block returns the actual position of the axis (feedback) in user units.

MC_ReadActualVelocity: This Function Block returns the feedback velocity

MC_ReadActualTorque: This Function Block returns the feedback torque

MC_ReadParameters: This Function Block returns the value of an axis-specific parameter. Some popular parameters are position error, commanded position, commanded velocity, commanded torque, commanded acceleration, deceleration, torque limit, etc

MC_ReadBoolParameter: This Function Block returns the value of axis-specific Boolean parameters like InPosition, At Velocity etc

MC_ReadAxisError: This Function Block reports axis errors not related to Function Blocks, servopack warnings and alarms. Error Class output designates the source of the alarm or warning. The AxisErrorID output contains the error code.

Y_ReadDriveParameter: This Function Block reads the specified parameter from the servopack of the specified axis.

Y_ReadAlarm: This Function Block reports controller-specific alarms that are not axis related.

2.1.10 MAINTENANCE:

Note:

Machines that are modular (groups or individual axes missing at times) can be programmed with the MP2000iec. The controller warns the user if all configured axes are not discovered on the network on boot up. This alarm can be cleared from within the application itself. For example:

If there are two configured axes on a controller and the system is powered up with only one axis (Axis1). The user will get an error code from the MC_ReadAxisError block of 0004 and an error class of 340A as shown below

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

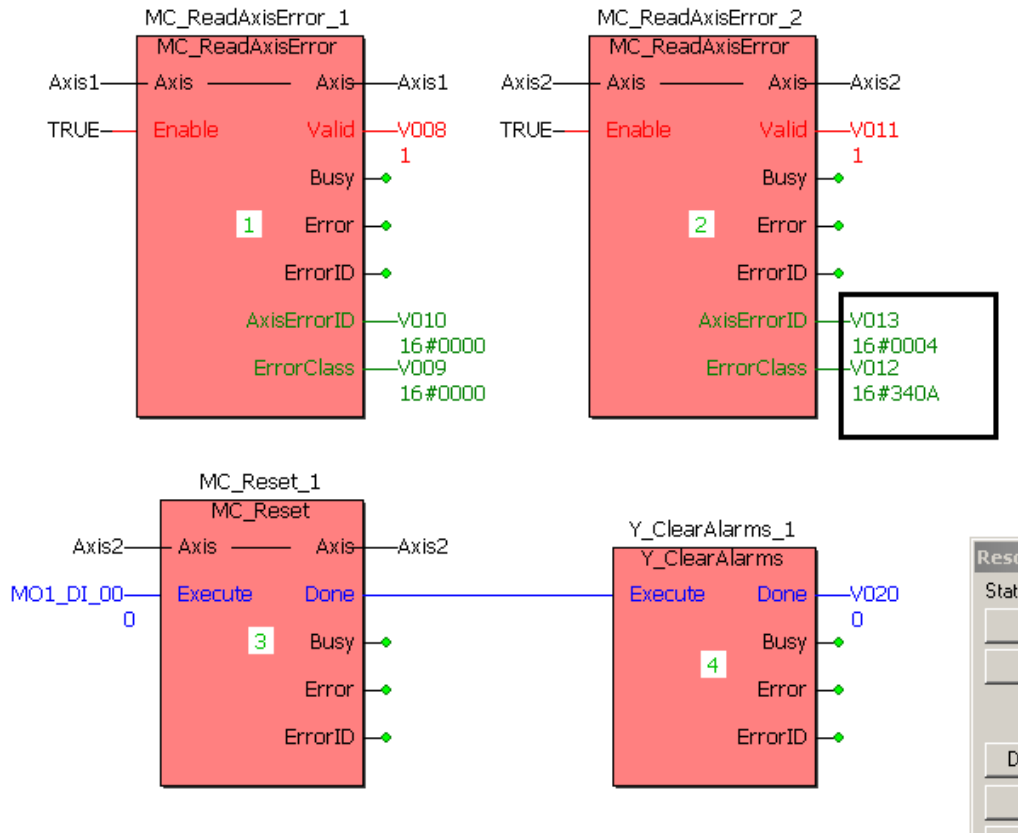


Figure 7b: Error and ErrorID on MC_ReadAxisError

The description of the error ID is as shown in figure 7c

Alarm Code	Source	Description
340a 0004	AXIS2	Configured servo network axis node was not found [more...]
340a 0001	I/O Config	Source for logical input not found. [more...]
340a 0002	I/O Config	Source for logical output not found. [more...]

Clear Alarms Save...

Figure 7c: Error list from web page

The first error says that the configured axis is not found. Notice the IO configuration alarms. That corresponds to the inputs and outputs on the servo node that was not found. The first alarm can be cleared using MC_Reset for that axis and the second two alarms can be cleared using the Y_ClearAlarms block as shown above.

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

MC_WriteBoolParam: This Function Block writes the value of an axis specific Boolean parameter like enable software limit.

MC_WriteParameter: This Function Block writes the value of an axis-specific parameter like Maximum Acceleration limit, Max Deceleration limit etc.

Y_ClearAlarms: This Function Block clears controller-based alarms that are not axis specific

MC_Reset: This Function Block makes the transition from the ErrorStop to StandStill state by resetting axis-related errors.

Note: *Emergency Stop by taking away servopack power will cause the axis to be disabled unexpectedly. This can cause a MECHATROLINK warning (A95). If the servopack displays any error or warning while the controller project is running, the controller CPU load increases and can lead to a watchdog time out condition. It is recommended to clear servopack and controller alarms and warnings as part of emergency stop recovery code.*

Y_ResetMechatrolink: This function block resets the MECHATROLINK network. Nodes can be temporarily disconnected from network and rediscovered afterward. New MECHATROLINK axes can be synchronized without cycling power on the controller using Y_ResetMechatrolink.

Y_WriteDriveParameter: This Function Block writes the specified parameter to the servopack of the specified axis.

Y_ResetAbsoluteEncoder: This Function Block clears absolute encoder alarms caused by battery power loss, cable disconnection, etc. This function block is equivalent to the Fn008 servopack function. After performing this function, the motor position will be cleared and must be re-established (see MC_SetPosition) to avoid mechanical damage to the machine.

Y_VerifyParameters: This function block is used to compare the current servopack parameters with the servopack parameters that are in the controller with the project. If the two parameter files match, a Boolean output on the function block is turned on. If there is even one parameter that does not match, the parameter number, the parameter as read from the servopack and the parameter value stored in the controller are displayed. If multiple parameters do not match, the first parameter that did not match is displayed. The servopack parameters are stored in the controller when the configuration is saved using the Configuration Tool.

Y_WriteParameters: The Y_WriteParameter function block is used to write a servopack parameter file stored in the controller into the servopack replacing the actual parameters in the servopack. This is useful in maintenance scenarios where a servopack needs to be swapped. Servopack parameter files are stored in the controller when the configuration is saved using the Configuration Tool.

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

Note: *Downloading a program while any of the following blocks are busy (even if the PLC was stopped), might lead to undetermined consequences:*

- MC_Reset*
- MC_StepRefPulse*
- MC_StepLimitSwitch*
- Y_CamFileSelect*
- Y_ClearAlarms*
- Y_ResetAbsoluteEncoder*
- Y_ResetMechatrolink*
- Y_VerifyParameters*
- Y_ReadParameters*

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

2.2 Function Block Performance

Table 4 shows the typical and worst case measurements for each function block. For the motion function blocks (MC_MoveAbsolute, MC_MoveRelative, etc.) the worst case situation occurs when the function block aborts an active function block. For, MC_Stop, MC_Power, MC_Reset, MC_TouchProbe, MC_AbortTrigger, MC_StepRefPulse, and MC_StepLimitSwitch, the worst case occurs on the rising edge of Execute.

Table 4: Function block execution time

Function Block	MP2300Siec/MP2310iec		MP2600iec	
	Typical (usec)	Worst Case (usec)	Typical (usec)	Worst Case (usec)
MC_AbortTrigger	40	350	20	400
MC_FinishHoming	30	700	50	1600
MC_GearIn	35	500	70	1200
MC_GearInPos	35	700	70	1900
MC_GearOut	35	300	30	600
MC_MoveAbsolute	35	700	30	1200
MC_MoveRelative	35	700	30	1200
MC_MoveSuperimpose	35	350	30	800
MC_MoveVelocity	35	400	30	850
MC_Power	40	400	60	950
MC_ReadActualPosition	16	25	35	40
MC_ReadActualTorque	16	25	35	40
MC_ReadActualVelocity	16	25	35	40
MC_ReadAxisError	55	120	90	200
MC_ReadParameter	30	65	65	110
MC_ReadBoolParameter	30	65	55	65
MC_ReadStatus	40	70	75	85
MC_Reset	30	150	30	250
MC_SetPosition	30	400	30	1000
MC_StepRefPulse	35	150	40	350
MC_StepLimitSwitch	35	150	40	350
MC_Stop	35	600	35	1900
MC_TorqueControl	40	450	60	850

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

MC_TouchProbe	60	300	20	900
MC_WriteParameter	30	40	20	60
MC_WriteBoolParameter	30	40	20	60
Y_CamFileSelect	25	150	20	300
Y_CamIn	40	700	90	2000
Y_CamOut	25	450	20	950
Y_CamScale	35	200	50	550
Y_CamShift	35	200	50	550
Y_CamStructSelect	25	233	50	700
Y_ClearAlarms	25	150	20	200
Y_HoldPosition	30	530	70	1200
Y_ReadAlarm	15	50	20	150
Y_ReadCamTable	25	115	20	550
Y_ReadDriveParameter	25	300	20	750
Y_ResetAbsoluteEncoder	35	150	40	350
Y_ResetMechatrolink	25	150	N/A	N/A
Y_SlaveOffset	35	200	65	550
Y_VerifyParameters	35	150	60	400
Y_WriteCamTable	25	115	60	550
Y_WriteDriveParameter	25	300	20	850
Y_WriteParameters	35	150	30	350

Note:

On an application with multiple axes (> 4), when using function blocks like MC_Reset, MC_Power, or MC_SetPosition which take more time and resources to execute, it is recommended to execute them in a stair stepped manner as shown in figure 8 below. This will not affect motion performance since these function blocks are not related to motion.

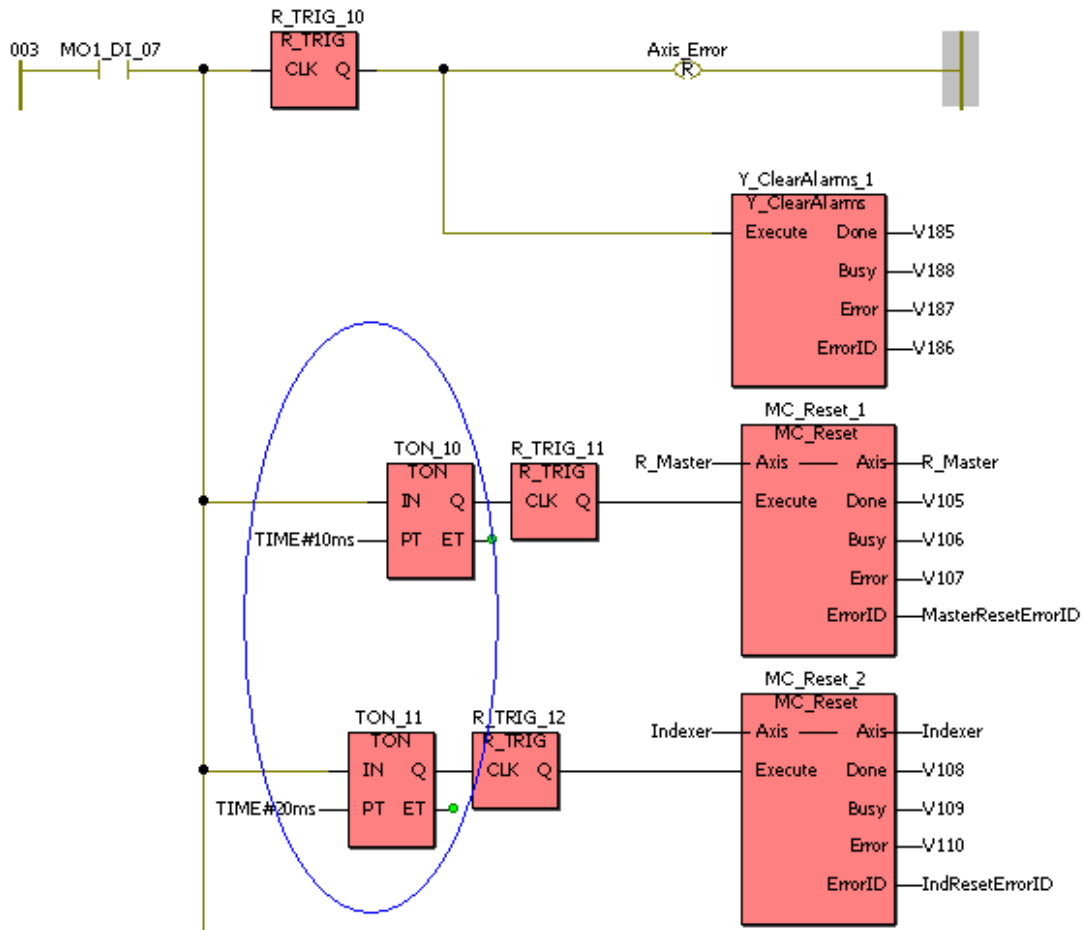


Figure 8: Recommended staggered execution of function blocks

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

2.3 MECHATROLINK Update Rate

Table 5 gives the maximum axis count that should be used with a particular MECHATROLINK communication rate.

Table 5: Guidelines for maximum axis count for MECHATROLINK update rate.

	0.5 ms	1.0 ms	1.5 ms	2.0 ms	2.5 ms	3.0 ms	4.0 ms
Maximum Axis Count	1	2	4	6	7	8	16

There is a trade off between motion performance and application performance; the PLC application can run faster if the MECHATROLINK update rate is made to be slower. So, if the response time to a change in an input is more important than the motion performance, then the application programmer should consider slowing down the MECHATROLINK update rate.

Also, there are circumstances when an application developer can exceed the guidelines in Table 5. For example, an idle axis (no trajectory computation) takes 50 usec, so if the application developer is certain that only 8 of the 16 axes will every move at the same time, a 2.5 ms MECHATROLINK update rate may work. It is recommended that the application programmer documents this information for program and application scalability purposes as a note for future program maintenance.

2.4 Typical PLC Scan Times

Based on performance measurements, the follow PLC scan times are typical:

Table 6: Guidelines for application scan times

	Case 1	Case 2	Case 3	Case 4
Number of servopacks	4	4	8	8
MECHATROLINK period (ms)	1	2	2	3
PLCopen Function blocks	40	40	40	40
Number of axes moving at once.	4	4	8	8
Ethernet/IP connections	1	1	1	1
Scan time average (ms)	7	4	7	5
Scan time worst case (ms)	17	9	26	17

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

The relation between an application scan time, MECHATROLINK scan, input reads and outputs write functionality is illustrated below. It is recommended that the application scan time be a multiple of the MECHATROLINK scan. Motion commands are sent to the servopack at the MECHATROLINK scan. Since MECHATROLINK scans have higher priority, they interrupt the application scan and perform motion commands based on the application program scanned till then.

Recommendation: Application scan time = X * MECHATROLINK scan time

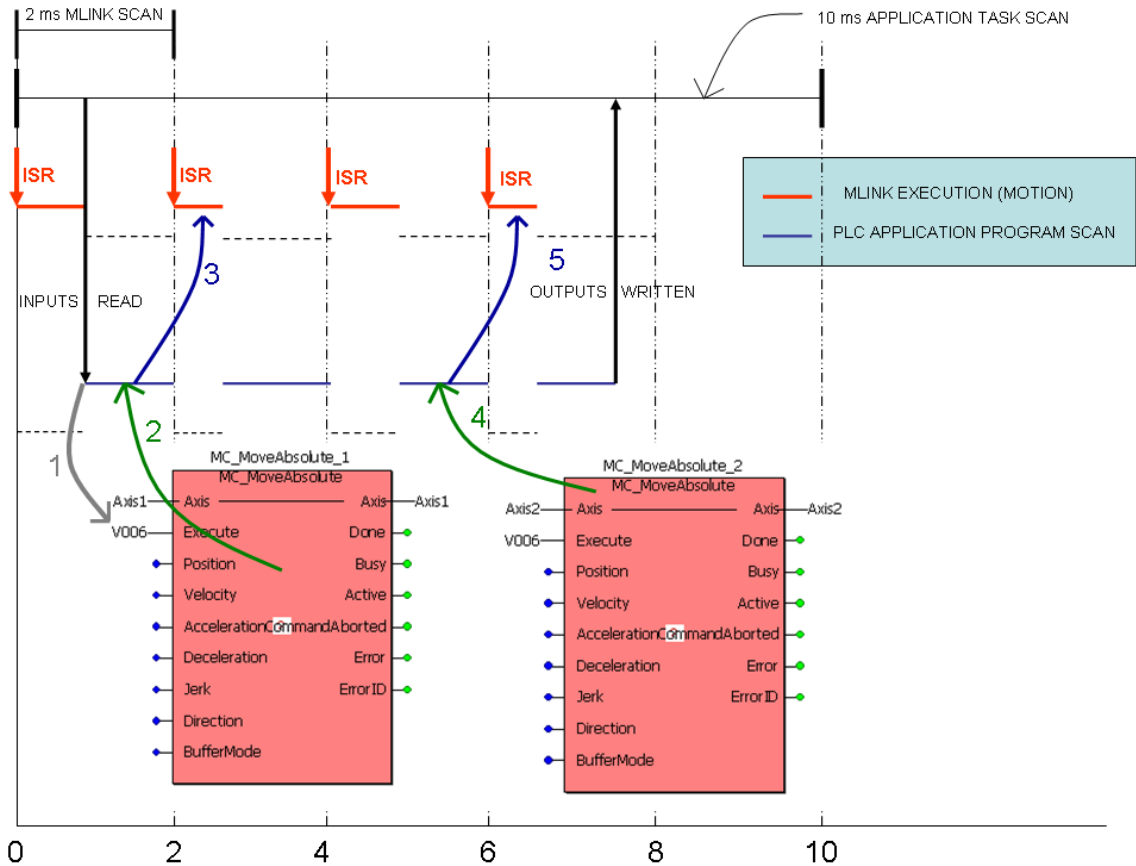


Figure 9: MECHATROLINK and application scan sequence

In the above illustration, the MECHATROLINK scan is 2 ms and the application scan is 10 ms. It can be seen that the first input read happens after the first MECHATROLINK scan within the PLC application scan. This is when input 'V006' is read (process 1). In the application scan, the first MC_MoveAbsolute block is scanned

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

(**process 2**). At the next MECHATROLINK scan, the motion command corresponding to the move is sent to the servopack (**process 3**). MC_MoveAbsolute_2 could be located further down in the application program. In this example it is scanned between 4 and 6 ms (**process 4**). Therefore, the command for MC_MoveAbsolute_2 will be sent over MECHATROLINK in the interrupt scan after the 6 ms mark (**process 5**). In short, two function blocks scanned and executed by the same input variable (V006), in the same application scan may not start at the same time depending on their positions in the application program due to the MECHATROLINK interrupt priority. There can be a position difference between the two axes = speed * number of MECHATROLINK scans.

All the outputs are written when the PLC application scan is finished. This means that in some scans if there is logic that uses CPU resources more than in adjacent scans, outputs will be written later in the PLC scan. Output writing does not happen always at the same point in time in each scan.

Note:

To ensure synchronous motion on multiple axes, gear the axes together before the master axis starts motion, and then start motion on the master axis. This will ensure synchronous motion of multiple axes

<p><u>MP2000 Standard Execution Methodology</u> In the standard MP2000 product, the SVB module is the controller's motion engine, responsible for control modes, sending and receiving motion data to each axis, and calculating interpolation data when an axis is operated in interpolate mode. The High Scan Ladder of the CPU can be programmed to perform any type of motion, ranging from simple homing or jogging to camming and gearing. Scan rates are typically 1- 4 ms based on the number of axes in the system. In addition to the High Scan, the user can program lower priority activity in the Low Scan ladder to optimize the overall performance of the application.</p>	<p><u>MP2300Siec Execution Methodology</u> In the MP2300Siec controller, the Flexible Motion Kernel, or FMK runs at the same update rate as the MECHATROLINK network. Because all motion functionality is integrated into the firmware and programmable via PLCOpen function blocks, the FMK / MECHATROLINK scan is equivalent in performance to the High scan of the standard MP controller. The user application task in MotionWorksIEC can be compared to the low speed scan of the standard MP controller. See Figure 10 for details.</p> <p>For this reason, MP2300Siec motion profiles are generated at the same rate of performance as the MP2000 standard.</p>
--	--

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

In MotionWorksIEC Pro, 16 application tasks may be configured. This greatly increases the performance options available to the programmer. One task may be set to run at the same interval as the FMK for the most critical operation, while other tasks are at slower rates based on their level of importance to the machine operation. Priority levels on each task can be set.

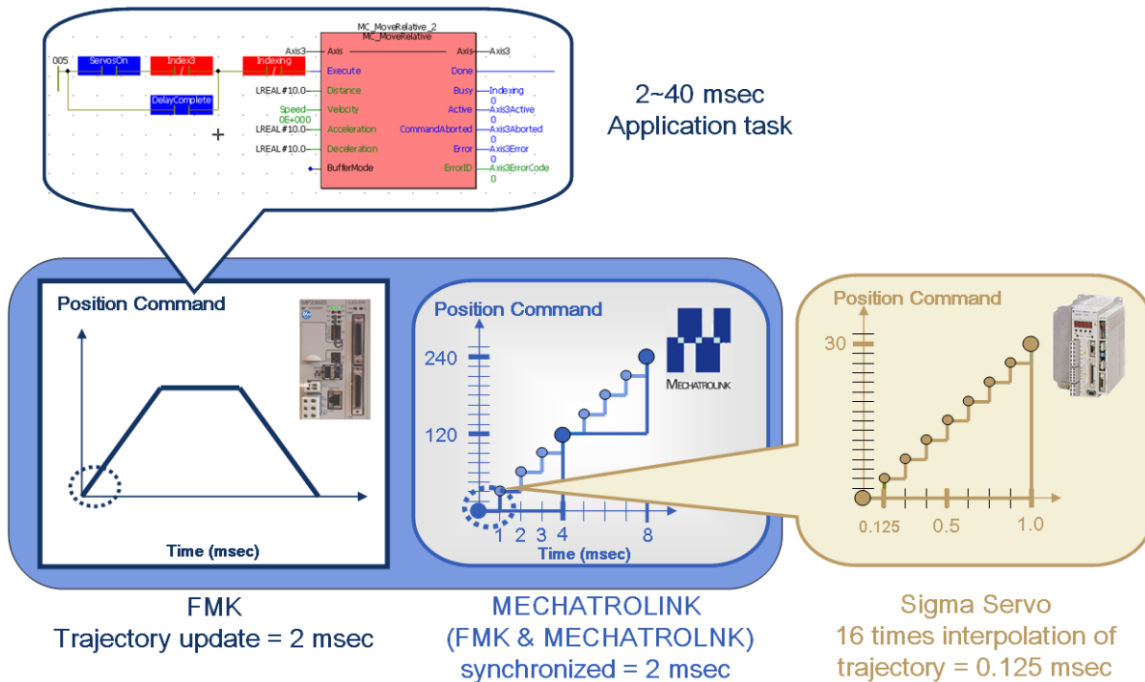


Figure 10: Performance of FMK motion engine

3. Communication: MODBUS/TCP, Ethernet/IP & OPC

Note: Motion based on MODBUS TCP or EtherNet/IP based master commands for gearing or camming are not recommended. This can be done only if the application does not call for tight synchronization and positioning. The master for synchronous motion has to be a servopack or a virtual axis. MECHATROLINK based Phoenix bus coupler masters are also not recommended for applications that demand high precision

Documentation and quick start guides for configuring devices, establishing communication can be obtained in the Configuration Tool manual [4]. The various supported Function codes for MODBUS TCP and assembly instances for EtherNet/IP are listed in the manual. App notes on communication with third party devices are listed in [4].

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

Until version 1.0.5 build 2 of MotionWorksIEC, the new program template had assembly instances and data blocks enabled which made the MP2000iec an EtherNet/IP adapter and MODBUS TCP server by default. These communication drivers used to load the CPU even if communication was not being used. In order to reduce CPU load, the new program template in MotionWorksIEC (from version 1.1.1 build 4) will not have the assembly instances or data blocks enabled by default. To increase performance, the required assembly instances and data blocks will now have to be manually enabled.

The recommended steps to enable communications on EtherNet/IP or MODBUS TCP are:

1. Launch the configuration tool and connect to the controller
2. Choose the auto-discovered configuration (if first time configuration)
4. Configure communication (for either client/server or scanner/adapter)
8. Save and cycle power on the machine

3.1. MP2000iec as MODBUS TCP Client (Master)

- Maximum number of configurable servers(slaves) : 20
- Maximum number of data blocks per server: (40 tested)
- Minimum poll period: 4 ms (+4 ms tolerance).

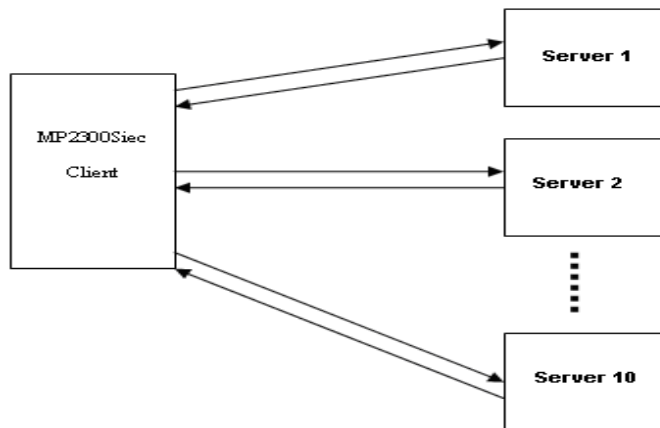


Figure 11: MP2000iec as a MODBUS client (master)

Note: Care should be taken to set the timeout period on the server (slave) driver code. The recommended time out period for server driver code is $TO = \text{poll period} * \# \text{ of data blocks}$.

One data block is polled per poll period set in the client (master). For example if the client has four data blocks configured for a particular server with a 50 ms poll period, the first data block is polled first. After 50 ms

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

the second data block is polled. 50 ms after the second data block, the third data block is polled and so on. The first data block is polled a second time 200 ms after it was polled the previous time.

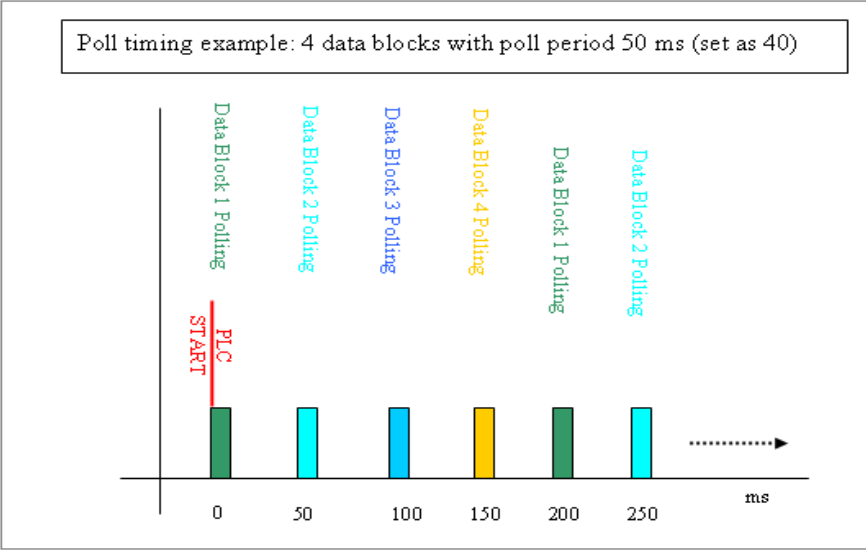


Figure 12: MODBUS poll period timing in MP2000iec

➤ Function codes supported:

Function Code	Description
1	Read Coils
2	Read Inputs
3	Read Holding Registers
4	Read Input Registers
5	Write Single Coil
6	Write Single Register
16	Write Multiple Registers

Figure 13: MODBUS function codes supported

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

3.2 MP2000iec as MODBUS TCP Server (Slave)

- Maximum number of clients on network: Unlimited (Theoretically) , (2 tested)
- Update rate: task scan rate
- Maximum number of data blocks: 5

Table 7: MODBUS Function Codes supported (MP as a server)

Traffic	Function Code	Size
Server to Client	FC 2	128 input bits (%QX)
Server to Client	FC 4	1024 registers (%QB)
Client to Server	FC 5	128 coils (%IX)
Client to Server	FC 6, 16	1024 registers (%IB)
Server to Client	FC 3	1024 registers (%QB)

Note: The application program in MP2000iec can now write into the holding register area (40000) in the MODBUS memory. This is done by enabling the Holding register output data block through the configuration tool.

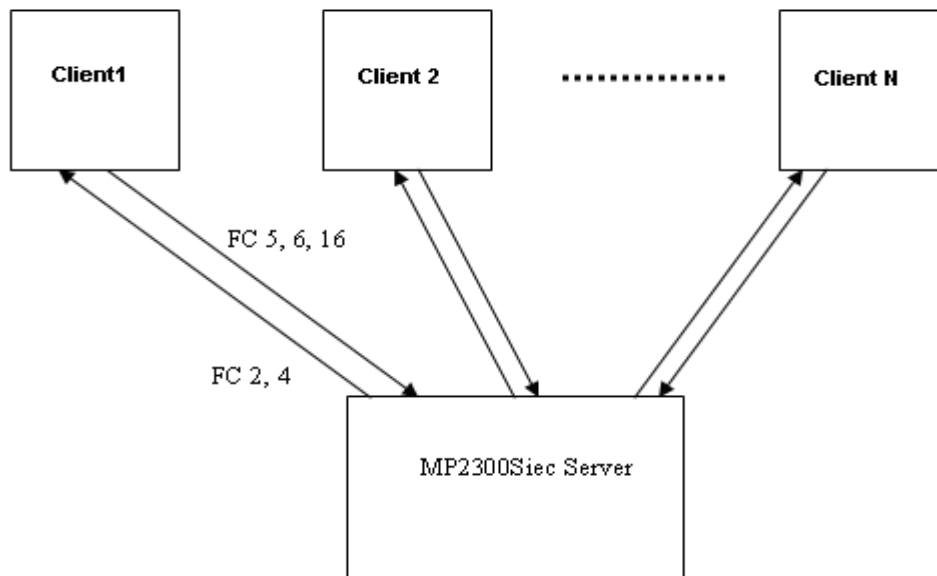


Figure 14: MP2000iec as a MODBUS server (slave)

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

Note: MODBUS outputs are **not** retained on power cycle. This issue is fairly easy to workaround by first storing the output value in a global variable that is retained and then copying the global variable to the output. MODBUS inputs are configurable for retain ability.

Table 8: Devices tested for MODBUS TCP communication with the MP2000iec [4]

Device	Client/Server with MP2000iec*	App Note
RedLion HMI G3-10	C/S	AN.MCD.08.122
Wago I/O Module	C/S	eLV.MotionWorksIEC.01.ModbusSlave
ASi Controller *YEG	C	
Beijer/Cimrex	C	
Kepware	C	
Maple HMI	C	
MP2300Siec	C/S	
Modicon	C/S	
MP2300	C/S	
AB Panelview	C	
Phoenix MODBUS IO	S	AN.MCD.09.045
DigiOne IAP serial to Ethernet converter		AN.MCD.09.093
Pro-face HMI	C	AN.MCD.09.124

*C: Client (Master), S: Server (Slave)

Note: MODBUS outputs can be configured to change state to OFF on PLC Stop or to retain the last state on PLC Stop. This is done by configuring the outputs when MODBUS functionality is configured in the hardware configuration.

3.3 MP200iec as EtherNet/IP scanner

- Maximum number of configurable adapters(slaves) : 20
- Maximum number of assembly instances: There can be multiple instances communicating with one physical device. User can configure 20 adapters with one instance in and one instance out.
- Minimum poll period: 2 ms

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

3.4 MP2000iec as EtherNet/IP adapter

- Maximum number of scanners on network: Unlimited (theoretically) , (2 tested)
- Update rate: task scan rate
- Maximum number of assembly instances: 6 input and 6 output

The preconfigured instances are listed below. The instances can be enabled or disabled based on which ones the user plans on using. The feature to enable and disable these assembly instances helps reduce CPU load during operation.

Input Assembly Instances (Originator to Target)			Output Assembly Instances (Target to Originator)		
Enabled	Instance	Size (bytes)	Enabled	Instance	Size (bytes)
<input type="checkbox"/>	111	128	<input type="checkbox"/>	101	128
<input type="checkbox"/>	112	256	<input type="checkbox"/>	102	256
<input type="checkbox"/>	113	128	<input type="checkbox"/>	103	128
<input type="checkbox"/>	114	256	<input type="checkbox"/>	104	256
<input type="checkbox"/>	115	128	<input type="checkbox"/>	105	128
<input type="checkbox"/>	116	256	<input type="checkbox"/>	106	256

Figure 15: Assembly instances when MP2000iec is an EtherNet/IP adapter

Note: EtherNet/IP inputs are retained on power cycle. EtherNet/IP outputs are **not** retained on power cycle. This issue is fairly easy to workaround by first storing the output value in a global variable that is retained and then copying the global variable to the output.

Table 9: Devices tested for EtherNet/IP communication with the MP2000iec

Device	Scanner/Adapter with MP2000iec*	App Note
AB ControlLogix EIP module v 13.27	S	AN.MCD.08.107
AB CompactLogix L32E v 13, v 16	S	AN.MCD.08.110
AB MicroLogix 1100, 1200, 1400, v 8	S	AN.MCD.08.108
MP2xxx (263IF-01)	S/A	
MP2300Siec	S/A	
Cognex Vision	S	
Yaskawa VFD (V7)	A	MWIEC Conf Man
Motoman NX100 controller with PCI card	A	
Numatics Valve	A	AN.MCD.09.092

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

SMC EX250-SEN1	A	AN.MP2000iec.03
Beckhoff I/O	A	

*S: Scanner, A: Adapter

Note: *EtherNet/IP outputs can be configured to change state to OFF on PLC Stop or to retain the last state on PLC Stop. This is done by configuring the outputs when EtherNet/IP functionality is configured in the hardware configuration.*

3.5 OPC Communication

- Communication speed: 1000 words back and forth in 40 ms (tested) [Also tested 90 Tags @ 10ms]. [Part PDE-U-OPCPA]
- YEA OPC server works with Windows Vista
- YEA OPC server does **not** work with Windows CE
 - Third party versions are available to support CE: Kepware
- OPC variables can be chosen during application creation (variable declaration)

3.6 Heartbeat and Watchdog implementations for ensuring healthy communication when the MP2000iec controller is an adapter/server

The following figures illustrate how a combination of heartbeat and watchdog techniques can be used to monitor communication loss between the MP2000iec as a server/adapter over MODBUS TCP/EtherNetIP. As a server or adapter, the MP2000iec does not have control over communication with clients or scanners. Therefore the status of communication with a particular client/scanner can be ensured only by means of a heart beat in combination with a watchdog.

The client/scanner device should have the following logic. 'From_Slave' is a Boolean modbus variable that is an input to the client and 'To_Slave' is a Boolean modbus variable output from the client.

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		



Figure 16a: Heartbeat logic in Client device

[-] <Modbus Slave> 'ogrp2' Address Range: %QB4 - %QB5 (* Do Not Modify Group Name or Status Variable!! *)					
To_Slave	FALSE	BOOL	VAR_GLOBAL		%QX4.0
[-] <Modbus Slave> 'igrp3' Address Range: %IB4 - %IB5 (* Do Not Modify Group Name or Status Variable!! *)					
Stat	16#1000	WORD	VAR_GLOBAL	(* Do Not Mo...	%IW6
From_Slave	FALSE	BOOL	VAR_GLOBAL		%IX4.0

Figure 16b: Addressing on Client device

The server/adaptor program on the MP2000iec should contain the following logic.



Figure 16c: HeartBeat Logic in server device

[-] Modbus FC#05 Qty: 128 Coils, Address Range: %IX0.0 - %IX15.7					
From_Master	TRUE	BOOL	VAR_GLOBAL		%IX0.0
[-] Modbus FC#06,16 Qty: 1024 Registers, Address Range: %IB16 - %IB2063					
[-] Modbus FC#02 Qty: 128 Inputs, Address Range: %QX0.0 - %QX15.7					
To_Master	FALSE	BOOL	VAR_GLOBAL		%QX0.0
[-] Modbus FC#04 Qty: 1024 Input Registers, Address Range: %QB16 - %QB2063					

Figure 16d: Addressing on server device

Variable 'To_Slave' is mapped to 'From_Master' on the MP2000iec. Variable 'To_Master' on the server is mapped to 'From_Slave' on the client.

Table 10a: Server Client variable mapping

Server		Client
From_Master	←	To_Slave
To_Master	→	From_Slave

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

On losing communication, the heartbeat part of the communication process will get stuck. This can be used with watchdog logic to warn the MP2000iec program that communication has ceased and safety routines need to get activated. Normal operation involving motion triggered by communication should be stopped at this juncture because communication has been broken. The following is watchdog logic that should be incorporated on the MP2000iec server. 'Comm_Fail' is the variable that becomes TRUE when the heartbeat is stuck for one second. The time for the watchdog can be set on the two timer blocks. This is application specific and is dependent upon how long the application can run with failed communication.

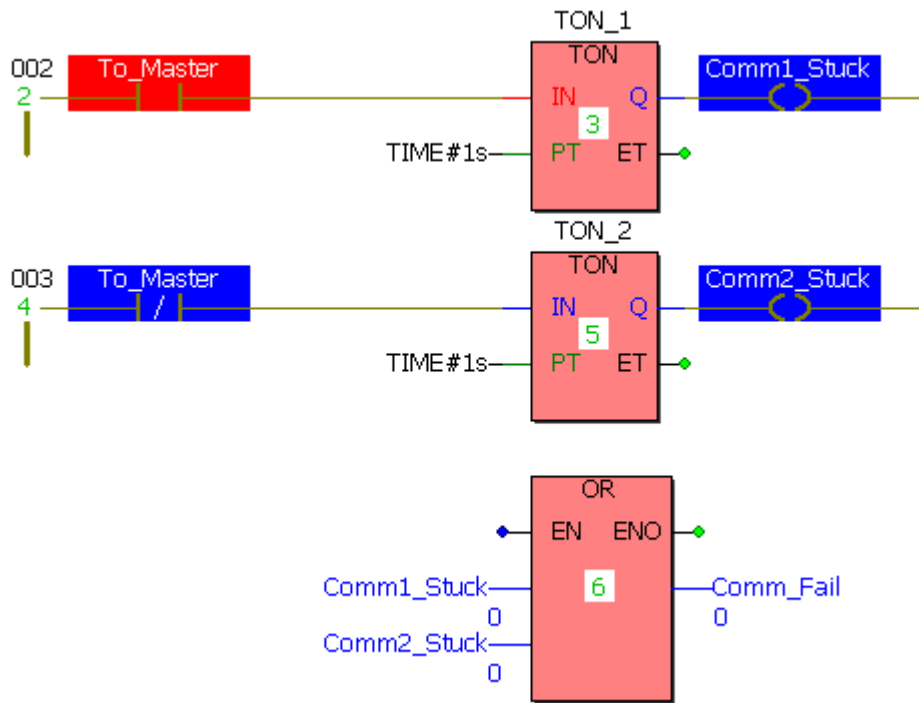


Figure 16e: Watchdog logic on server

Variable 'Comm_Fail' will have to be interlocked with an MC_Stop block which controls the axis in question. If the execute of MC_Stop is held high, no motion block can get executed. This can be used to prevent untoward motion in case of communication failure between devices.

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

4 Supported Devices

4.1 MECHATROLINK

4.1.1 MECHATROLINK Axes

- External encoder supported only through LIO-01/02 and LIO-6, External non-quadrature Absolute encoders not supported (Grey Code, SSI, Serial, etc.)

Table 10b: MECHATROLINK Axes

Supported Features/Hardware	Status*
Self Configuration	S
Sigma-5 Linear	S
Sigma-5 Rotary	S
Servopack Inputs Sigma-5 (seven inputs)	S
Servopack Outputs Sigma-5 (Three outputs)	S
Sigma-2, Sigma-3 Rotary	S
Sigma-3 Linear	S
Servopack Inputs Sigma-2 and Sigma-3	S
Direct Drive motors	S
Virtual Axes (6 per controller)	S
Yaskawa Mechatrolink I/O	F (See table 11)
Exlar motors (absolute encoder)	S
Sigma-2 Linear	U
Servopack Outputs Sigma-2 and Sigma-3	U
Sigma-1	U

*S: Supported, U: Unsupported, F: Future

Note:

Virtual axes use 2/3 of the resources used by a servo axis. There is a one MECHATROLINK scan delay between the commanded position and actual position of a virtual axis. It is recommended that any position following be executed using the actual position of the virtual master because the commanded position on the virtual master is unmodulated (actual position is modulated). Virtual axis needs to be configured while the controller is offline. Once virtual axis configuration is saved in an offline mode in the project, make connection with the controller by choosing the offline configuration.

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

4.1.2 MECHATROLINK I/O

Table 11: MECHATROLINK I/O Devices

Device	Description	Status*
Phoenix Bus Coupler	8 in , 4 out + additional slices	S
IO2310/IO2330	(64 in, 64 out)	S
PL2900	(2 channels)	U
PL2910	(2 channels)	U
AN2900	(4 channels)	U
AN2910	(2 channels)	U

*S: Supported, U: Unsupported (please contact apps for direction), F: Future

4.2 Other I/O Devices

4.2.1 Onboard devices

Table 12: Onboard I/O Devices

Device	Description	Status*
LIO-01, LIO-02	(16 in, 16 out, 1 ext enc)	S
LIO-04, LIO-05	(32 in, 32 out)	S
LIO-06	8DI/8DO, 1AI/1AO, 1 ext en	S
AI-01^	(8 channels)	S
AO-01^	(4 channels)	S
CNTR-01	(2 channels)	F (see note below)
260-IF	DeviceNet	U
PO-01	(4 axes)	F
SVB-01	(2 ports)	U
SVA-01	(2 axes)	U

*S: Supported, U: Unsupported, F: Future

^ Multiple AI or AO cards on the MP2310iec controller not supported

Note: For applications that require future (F) or unsupported devices (U), please contact Yaskawa Applications Group for options/workarounds or development schedule.

Digital outputs can be configured to change state to OFF on PLC Stop or to retain the last state on PLC Stop. This is done by configuring outputs in the hardware configuration.

If an application requires a CNTR-01 card, an MP2310iec controller with two LIO-01 or 02 cards can serve the purpose. Each LIO-01/02 card has one external encoder input built in. Make sure that there is an LIO-01 or 02 card in the first slot of the MP2310.

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

5 Memory Capacity

Flash (Battery backed)

- MP2300Siec Total Size: 8MB
- Firmware Image & Netboot monitor: 4 MB
- Flash File System: 4MB
- Firmware files (Web interface, default config, drive parameters, etc.): 0.9MB
- Available for program files: 3.1MB

Not all of the 3.1 MB available for program files can be used for cam files because some will be used for the boot project and application configuration files. As this amount depends on the size of the application and download settings, there's no hard rule to determine the space available for cam files. However, flash usage can be monitored in the Configuration Tool.

Ramdisk (Volatile)

- Size: 4 MB
- Web interface: 0.9MB
- Available for application use: 3.1MB

In addition to cam files, the controller may also use the ramdisk to store log files (as large as 1 MB) and to temporarily store configuration files (typically less than 100k). So, the application can safely use 2 MB for cam files and 3 MB if logging is not needed.

6 Applications List

The MP2000iec is perfect for applications requiring multi-axis point-to-point operation. Industries served include Packaging, Assembly, Converting and Material Handling, as well as Machine Tool, Semiconductor, Medical and Automotive.

Applications well-suited for MP2000iec control are those involving Point-to-Point movement, Feed-to-Length, Electronic Line Shaft, Absolute Rotary Indexing, Torque/Force Control, Camming involving four axes.

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

Table 13: Applications

Motion Type	Application	Motion Function	Applicable Toolbox
Point-to-Point	<ul style="list-style-type: none"> ▪ Pick and Place ▪ Case Packing ▪ Assembly ▪ Gantry Control ▪ Machining 	Discrete Motion blocks (MC_MoveAbsolute)	Gantry
Feed-to-Length	<ul style="list-style-type: none"> ▪ Labeling ▪ Material Cutoff ▪ Intermittent Form/Fill/Seal ▪ Position Registration 	Registration using MC_TouchProbe and MC_MoveAbsolute	Feed To Length, CamSlave_FeedToLength
Electronic Line Shaft	<ul style="list-style-type: none"> ▪ Mechanical line shaft replacement ▪ Conveyor control 	MC_GearIn, MC_SuperImposed	-
Absolute Rotary Indexing	<ul style="list-style-type: none"> ▪ Dial plate indexer ▪ Parts magazine ▪ Valve control ▪ Lane diverter ▪ Antenna positioning 	MC_MoveAbsolute	EtherNet Positioner
Torque/Force control	<ul style="list-style-type: none"> ▪ Winding ▪ Pick and Place 	MC_TorqueControl	Winding
Camming	<ul style="list-style-type: none"> ▪ Flying shear ▪ Rotary knife ▪ Packaging machines ▪ Continuous Form/Fill/Seal 	Y_CamIn with YCamShift, Y_CamGenerator, Y_CamBlend	Camming Toolbox

7 Solution Packages and Template Code

7.1 Tool Boxes

One of the key strengths of the IEC61131-3 programming environment is the ability to develop libraries of re-usable code. Yaskawa has leveraged this ability to create Application Code Toolboxes designed for use in certain applications using MP2000iec machine controllers and MotionWorksIEC software. These toolboxes may be imported into user programs as a *User Library* to form the foundations of more complete, customized solutions and will **save time** for developers who would otherwise have to start from scratch. [4]

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

Table 14: Solutions Package

ToolBoxes	Detail
Gantry	XYor XYZ coordinate system positioning tools. Used for general point-to-point applications such as pick-and-place and parts transfer. Also provides functions for X-X' parallel axis control.
PLCOpenToolBox	This toolbox builds off the industry standard PLCOpen Motion Function Blocks by combining certain basic blocks into easier-to-use higher-level blocks. Examples are : <ul style="list-style-type: none"> • AxisStatus • AxisControl • Home_LS • ReadAxisParameters • EnableServo • Jog • MoveRelative_ByTime
VFD Control	This toolbox contains functions for communicating with Yaskawa Variable Frequency Drives (VFDs) over Ethernet/IP connection.
Feed to Length	This toolbox represents a nearly complete Solution for Feed-to-length applications with or without sensor registration. The core code is essentially distilled into a single function block for maximum ease of use.
Ethernet Positioner	This toolbox represents a nearly complete Solution for PLC-centric control systems that need motion. The toolbox contains ready-made instances for Ethernet/IP or Ethernet MODBUS/TCP communications with a host PLC sequencer. Motion code is pre-written to activate motion and send axis status based on sequencing commands from the PLC.
Winding	This toolbox contains functions specific to winding and tension control applications.
Cam Toolbox	Contains functions that provide enhanced capabilities for camming applications, such as profile generation, registration, and E-Stop recovery. Examples are: <ul style="list-style-type: none"> • CamGenerator • CamBlend • Cam Slave_FeedToLength • UpdateCamTable

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

8 References

- 1: MP2000iec Landing page:
<http://yaskawa.com/site/Products.nsf/products/Multi-Axis%20Motion%20Controllers~MP2300Siec.html>
- 2: MP2000iec Literature:
<http://yaskawa.com/site/Products.nsf/products/Multi-Axis%20Motion%20Controllers~MP2300Siec.html?openDocument&seq=1>
3. Sigma-5 Manuals:
<http://yaskawa.com/site/Products.nsf/products/Servo%20Amplifiers~SGDVSigma5.html?openDocument&seq=1>
4. App notes and example code:
http://yaskawa.com/site/DMControl.nsf/Productline_NewWindow.html?readform&pg=Servo%20Systems%20and%20Motion%20Controllers&product=MP2300Siec&restricttcategory=MP2300Siec
5. Camming webinar:
[https://partner.yaskawa.com/site/dmsearch.nsf/sitesearch?openagent&query=\(camming%20presentation\)](https://partner.yaskawa.com/site/dmsearch.nsf/sitesearch?openagent&query=(camming%20presentation))
6. ToolBox Manual:
http://yaskawa.com/site/products.nsf/ProductDetailPages/Multi-Axis%20Motion%20Controllers~MP2000iec%20Series~MP2000iec_Application_Toolboxes.html

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

9 MP2600iec

The MP2600iec is a 1.5 axes machine controller module fully integrated into the Sigma-5 series servopack. MotionWorks IEC is the standard programming environment that allows the user to scale up (1.5 axes to multi axes) and scale down (with some restrictions: mentioned in sub section 9.1) projects. All controllers come with EtherNet/IP and MODBUS TCP communication built in.

9.1 Differences: MP2300Siec/MP2310iec vs. MP2600iec

1. EN/ENO not supported on function blocks with the MP2600iec. The work around is:
 - Write conditional logic in text programs using if/then logic.
 - Contact Yaskawa to enquire about function blocks replacements
 - Use Jumps in ladder / function block programs
2. Code created for MP2300Siec or MP2310iec will not port over to the MP2600iec if EN/ENO blocks are present. Conversion will be required. Code generated for MP2600iec (single axis) can be ported over for MP2300Siec and MP2310iec (multi-axis controllers)
3. Resource window looks different. Download bootproject is handled differently. Check 'Permanent as boot project' to download boot project every time the download button is pressed. Regular 'Download' downloads changes.
4. Cam files need to be downloaded using Configuration Tool. Use path 'data/cam/xxxxxxx.csv' in Y_CamFileSelect to load cam table from controller memory to FMK memory.
Online>Controller Configuration Utilities> Send CAM data file to data/Cam directory on controller.
5. Outputs cannot be mapped to same address using two variables of different data types. Example: If %OW 4096 named 'output_word' is a variable used, %OX4096.0 cannot be accessed within the program using another bit type variable name.
6. Outputs are 50 mA (MP2300Siec is 100 mA)
7. Task interval resolution is in microseconds. This helps code optimization.
8. DPRAM update can be reduced to 1ms and not 0.5 ms MECHATROLINK update as on the MP2300Siec/MP2310iec.
9. Output relay (available on the face of the MP2300Siec/MP2310) not available on the MP2600iec hardware.
10. Two Ethernet ports are available on the MP2600iec. They should be assigned IP addresses in two different sub networks
11. Function block execution takes longer time on the MP2600iec controller. A comparison with the MP2300Siec or MP2310iec controller is given in Table 15. Detailed function block execution times are provided in table 4.
12. 1 MB RAM memory is available for variables in the MP2600iec compared to 1.5 MB available for variables on the MP2300Siec

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

Please refer to the document available at the link below for details on programming differences between the MP2300Siec/MP2310iec and the single axis MP2600iec controllers

<https://partner.yaskawa.com/site/dmcontrol.nsf/SearchV/86256EC30069B6348625770E007923EA?OpenDocument&Source=SearchResultPage>

Code running on an MP2600iec takes on average 1.7 times longer than what it takes to run the same code on an MP2300Siec or MP2310iec controller. Detailed function block execution times for the MP2600iec are provided in Table 4. A comparison of average function block execution times between MP2300Siec/MP2310iec and MP2600iec (compiled from Table 4) is given in Table 15 below.

Table 15: Comparison of average function block execution times between MP2300Siec/MP2310iec and MP2600iec

	MP2300Siec/MP2310iec		MP2600iec	
	Typical	Worst Case	Typical	Worst Case
All function blocks	X	Y	X * 1.35	Y * 2.28

9.2 Hardware combinations

Supported:

- o S5 Option Style Amp 100W - 15kW
- o All 3 Motor technologies:
 - Rotary, DD, Linear
- o Inverter on EtherNet/IP & Modbus TCP
- o HMI (modbus tested)
 - Red Lion, Proface, Exor, AB Panelview
- o UL rated

Unsupported*

- o Full Closed loop option card
- o External safety option card
- o CE rating (in process) - CE achievable inside an enclosure

* Please contact a Yaskawa applications engineer to discuss future availability of the listed unsupported items

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

9.3 MP2600iec Performance

Motion calculation rate

- All motion commands calculated at FMK (Flexible Motion Kernel) level
- FMK scan rate = DPR (Dual Port RAM) update rate (Same as HScan in MP standard). Please refer to Figure 10 (Performance of FMK motion engine)
- DPR rate can be adjusted by the user. Range is 1 to 4msec (2msec is default)
- IEC application tasks do not determine interpolation resolution, DPR rate does.

Camming

- Camming functions are very powerful on the MP2600iec and includes blending (smooth start/stop, switch tables on fly), and easy cam building with on-controller dynamic profile calculation
- Cam position interpolation at FMK update rate

Communications

- Proved/Tested 10 EtherNet/IP and 10 MODBUS TCP connections (20 possible by spec)
- Fastest EtherNet/IP communication 4 ms update when MP2000iec is a scanner
- Fastest MODBUS TCP communication 8 ms update when MP2000iec is a client
- OPC Server is fast (no drivers, direct communication)

9.4 Sample Applications for the MP2600iec

The following single axis applications have been tested on machines controlled by the MP2600iec

- *Point-To-Point Indexing*
- *ABS Rotary Indexing*
- *Registration Applications (most common are indexing and camming with on the fly shifting)*
 - » *Feed-To-Length*
 - » *Linear Flying Shear*
 - » *Random Rotary Placer*
 - » *Rotary Knife*
- *Irrational gearbox ratio*
- *Speed limited torque control*
 - » *Winding*
 - » *Tension Control*

Note: Yaskawa recommends using camming for applications that require rigid gearing (MC_GearInPos: Please refer to the note under the explanation for MC_GearInPos on page 15). Please contact an applications engineer at Yaskawa for details on how camming can be used for a rigid gearing application.

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

9.5 Sample application performance metrics for MP2600iec

1. System A: Complex motion with heavy communication. Core code for Random rotary placer with communication

High speed motion with external encoder latch registration, camming and on the fly shifting	Task 1 @ 6 ms
Sequence functions (Jog, Read Parameters)	Task 2 @ 20 ms
Communication (MODBUS: 100 words in/out) & Maintenance function blocks like (AxisControl, Y_ResetMechatrolink)	Task 3 @ 50 ms

2. System B: Fast scan motion centric (Simple Point to Point)

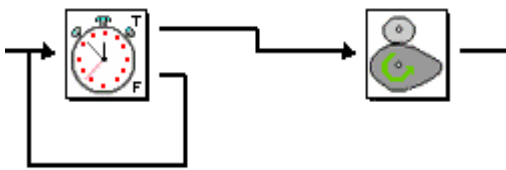
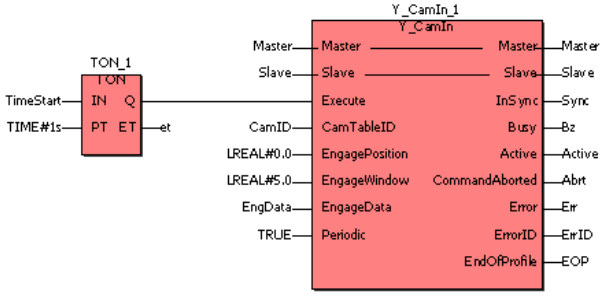
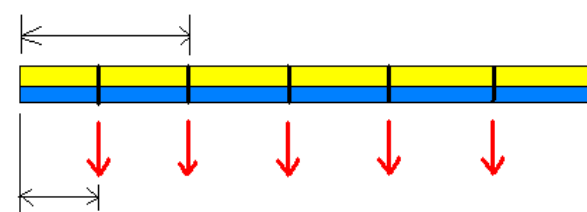
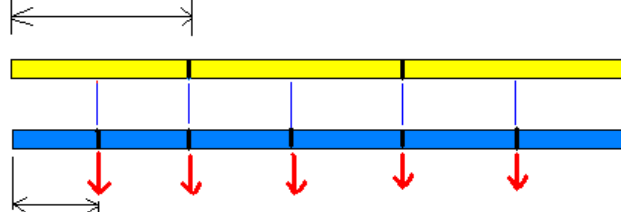
Point to Point Indexing (MC_MoveAbsolutes) with digital I/O	Task 1 @ 2 ms
Communication (EIP: 10 words in/out) and maintenance function blocks	Task 2: @ 20 ms

3. System C: Fast scan motion centric (synchronized motion). Core code for random rotary placer

Cam with registration (MC_TouchProbe)	Task 1 @ 4 ms
Maintenance function blocks like Axis Control, Y_ResetMechatrolink, Y_WriteDriveParameters)	Task 2 @ 20 ms

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

9.6 MP940 with MotionWorks+ vs MP2600iec with MotionWorksIEC: Execution comparison

<u>MP940 with MotionWorks+</u>	<u>MP2600iec with MotionWorks IEC</u>
High Scan = Dual Port Ram (DPRAM) update	Cyclic task interval need not be equal to DPRAM update. DPRAM update can be set by the user
Motion commands need to be in high scan because it determines motion engine update rate (controller command resolution dependent on high scan rate)	Motion commands need not be in a cyclic task that updates at DPRAM rate. (controller commands update at motion engine update rate which is equal to DPRAM rate)
Target position sent to servo every high scan (because high scan update = motion engine update)	Target position sent to servo every DPRAM update
Consider the following example of a cam application	
	
If high scan is set to 2 ms, cam update commands are sent to the servo every 2 ms	If the cyclic task interval is set to 4 ms, and DPRAM update set to 2 ms, cam update commands are sent to servo every 2 ms (good resolution even at slower cyclic task interval settings)
<p>High Scan</p>  <p>Motion Engine update</p>	<p>Cyclic Task update</p>  <p>Motion Engine update</p>

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

9.7 MP940 → MP2600iec field conversion benchmarks

1. Synchronous motion (camming) application involving external encoder master and servo slave

The MP940 → MP2600iec conversion application involved registration based camming with on the fly shifting to track product. Product pitch was 6". The goal was to meet existing throughput of 325 parts/min with an accuracy of +/- 1/16".

- *MP940 Results:*
 - » *Peak performance of 325 parts/min, 6" pitch, +/- 1/16" accuracy*
 - » *Scantimes: 2 mSec High Scan, 20 mSec Low Scan*
- *MP2600iec Results:*
 - » *325 parts/min, 6" pitch, +/- 1/16"*
 - » *Used Maple Systems HMI (modbus)*
 - » *External master*
 - *24V Latch at 60 microsec*
 - » *Scantimes: 4 mSec, 20 mSec, 60 mSec*
- *Key feature that helped results*
 - *Scan Compensation at FMK*

2. Indexing Application

The MP940 → MP2600iec conversion application involved high speed indexing. The indexing move was initiated by a digital input from a PLC and the move completion had to trigger a digital output to signal move completion. The goal was to meet existing performance of the indexing move of 72 degrees in 14 msec. The load ratio at the servopack was 1:1.

- *MP940 Results:*
 - *72 Degree MoveRelative in 14 msec*
- *MP2600iec Bench Test Results:*
 - » *Met basic machine requirements*
 - *PLC interface*
 - *Ext Input GO signal - Index Feed - Set Output COMPLETE signal*
 - *72 Degree MoveRelative in 14 msec*
 - » *Scan time: 2msec FMK, 2msec task, 40 msec task*

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

10 Revision History

#	Revision	Date	Details
1	1.0	3/12/09	Initial Release
2	1.01	3/27/09	Updates on: <ul style="list-style-type: none"> ➤ Camming ➤ Y_VerifyParameters, Y_WriteParameters
3	1.02	6/16/09	Updates on: <ul style="list-style-type: none"> ➤ MC_Power: EnablePositive & Enable Negative (page 4) ➤ MCMoveVelocity: InVelocity bit (page 4) ➤ MC_StepLimitSwitch: Shudder when moving out of over travel limit (page 5) ➤ S curve functionality (page 8) ➤ Y_VerifyParameters & Y_WriteParameters (page 18) ➤ Camming support materials added to partner website (page 15) ➤ MODBUS client supports 4 ms update and 20 servers (page 26) ➤ MODBUS server supports FC 3 (page 28) ➤ MODBUS inputs retained, outputs not controlled on PLC Stop (page 29) ➤ EIP scanner supports 2 ms update and 20 adapters (page 29) ➤ OPC server supports Windows Vista (page 31) ➤ Sample heartbeat and watchdog implementation (page 31) ➤ Hardware output behavior on PLC stop (page 35)
4	1.03	9/21/09	Updates on: <ul style="list-style-type: none"> ➤ Parameter 1310 (feed forward velocity) has to be turned off while exiting an over travel area using motion blocks (Page 5) ➤ To stop an axis controlled by MC_TtorqueControl, use another MC_TorqueControl block to bring down velocity to zero and then once the axis has stopped moving, use MC_Stop (Page 5)

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

			<ul style="list-style-type: none"> ➤ MC_GearIn gears to commanded position when in motion and actual position when not in motion (Page 13) ➤ Modular machine startup and alarm clear without all axes (Page 19) ➤ Ability to use the absolute encoder reset from MotionWorks IEC (Y_ResetAbsoluteEncoder) (Page 20) ➤ If CNTR card functionality is required, use MP2310 with two LIO cards (Page 38) ➤ 6 significant digits needed for cam tables (Page 14) ➤ MC_StepRefPulse has to be commanded a slow velocity when searching for c-channel (Page 4) ➤ MC_GearInPosition bug when decelerating and trying to slow down to match master axis position. (Page 14) ➤ Controller parameter 1311 for acceleration filter (Page 9) ➤ Pro-face HMI application note Page 31) ➤ Work around for retaining MODBUS output (Page 31)
5	1.04	12/28/2009	<ul style="list-style-type: none"> ➤ Do not use moving average filter (parameter 1300) with rotary shortest path motion on rotary axes (page 8) ➤ In a blended motion profile, the acceleration of the second function block is used to change from one velocity to another even if the second velocity is lower than the first one. (Page 9) ➤ Using MC_AbortTrigger in a high speed task may cause watchdog time outs (page 13) ➤ Downloading a program while some maintenance function blocks are busy can cause controller crash (page 20)
6	1.05	2/28/2010	<ul style="list-style-type: none"> ➤ Issuing an MC_Stop with a rotary axis performing shortest path motion with the s-curve filter active can cause unexpected motion (page 8) ➤ Alarms and warnings on the servopack while the controller is running can cause a watchdog timeout condition (page 20) ➤ SMC remote IO tested for EtherNet/IP connectivity

Subject: Application Note	Product: MP2000iec	Doc#: AN.MCD.09.042
Title: MP2000iec Application Design Guideline		

			<p>(page 34)</p> <ul style="list-style-type: none"> ➤ MP2310 with multiple LIO cards can be used instead of the CNTR card. There should be an LIO card in slot 1. (page 40) ➤ Removed rotary placer from list of suitable applications since this application is in the experimental phase (Page 41)
7	2.00	5/30/2010	<ul style="list-style-type: none"> ➤ MC_Power interaction with Hardware Base Block (HBB) (Page 5) ➤ Removed condition to bring axis out of torque control block. With firmware 1.2.1, MC_Stop will bring an axis to position mode from torque mode and the axis will decelerate to a halt at the MC_Stop.Deceleration input (page 6) ➤ MC_StepLimitSwitch for vertical axes. Make sure Pn001.1 is set to hold torque on over travel limit switch trigger (Page 6) ➤ MC_GearInPos unsupported for rotary mode (Page 15) ➤ Removed restriction on Y_CamScale for one way cams. Y_CamScale is now supported for one way cams (Page 17) ➤ Outputs configurable to retain last state or turn OFF on PLC stop (pages 33, 35 and 39) ➤ Beckhoff IO has been tested as an EtherNet/IP adapter to the MP2000iec scanner (Page35) ➤ Reconnect issue with SMC actuator resolved (page 35) ➤ Multiple AI or AO cards on the MP2310iec controller not supported (page 39) ➤ Application details provided in Table 13 (Page 41) ➤ Cam Toolbox added to table 14 (page 43) ➤ Subsection 7.2 (Future toolboxes) deleted (Page 43) ➤ Toolbox manual landing page added (Page 43) ➤ Section 9 added for MP2600iec. Differences between the MP2600iec and the MP2300Siec are detailed. (Page 44)