Programming IEC applications for Maintenance

Introduction

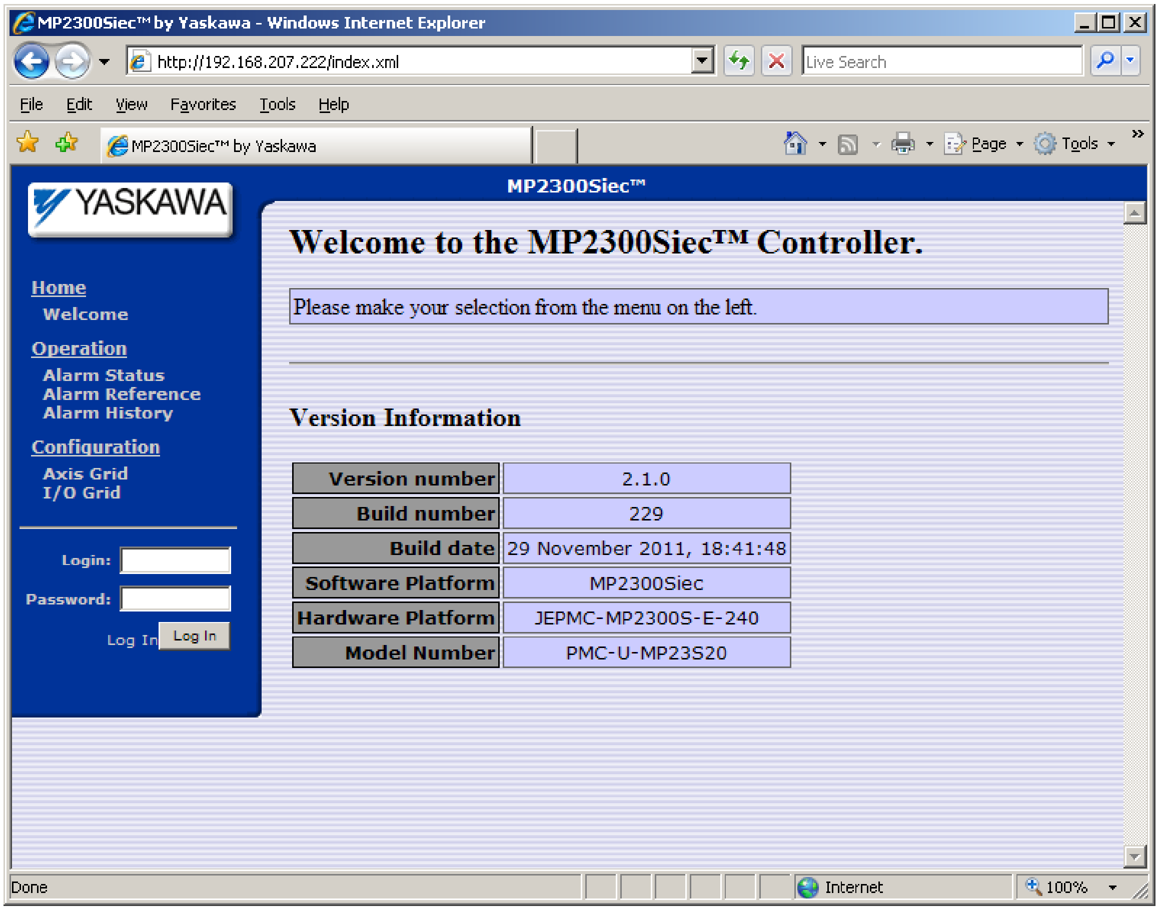
Yaskawa recommends that all IEC applications be programmed for maintenance by including routines that can be run in the field when an amplifier, motor, or controller needs to be replaced. Industrial machines have a long life and it makes sense to include pre programmed maintenance routines to help maintenance personnel restore the machine to production after an issue causes downtime. In an ICE application there are three components that should be considered when designing the program: the controller, amplifier, or motor. Each of these components would require some software setup if they needed to be replaced, while things like I/O cards or other network devices could be replaced by setting hardware switches and power cycling the system.

Programming for Controller Replacement

When replacing a controller in the field, the controller must contain the application program, configuration settings and amplifier parameters that are stored in the existing application. Fortunately, the controller contains the tools required to accomplish this. After the application has been commissioned, create an archive of the controller which contains all of the items mentioned above. Note: commissioning the system requires that the application program is started when the controller is rebooted. This means that the programmer must include the boot project when sending the project to the controller.

Use the following procedure to create the archive, or duplicate the application on a new controller. The steps are identical up to step 4, where 4 a) is used to create the archive and 4 b) to duplicate the controller. When creating the archive the procedure ends at step 4. When duplicating the system (controller and amplifiers are new, or being replaced), continue the procedure until the end.

1. To use the web browser enter the controller’s IP address in Internet explorer.



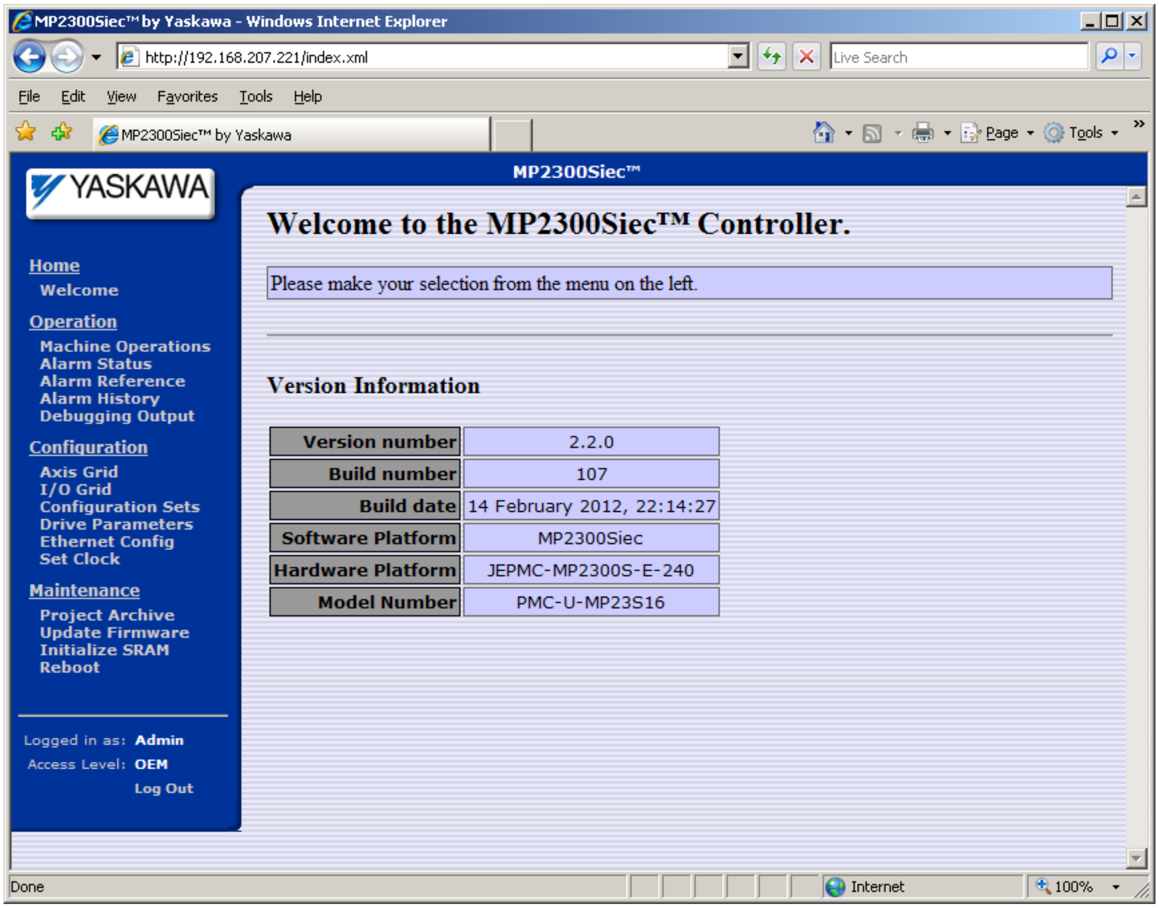
Welcome to the MP2300Siec™ Controller.

Please make your selection from the menu on the left.

Version Information

Version number	2.1.0
Build number	229
Build date	29 November 2011, 18:41:48
Software Platform	MP2300Siec
Hardware Platform	JEPMC-MP2300S-E-240
Model Number	PMC-U-MP23S20

2. Click on login and enter “Admin” for the user name and “MP2300S”, “MP2600”, or “MP3200” for the password.



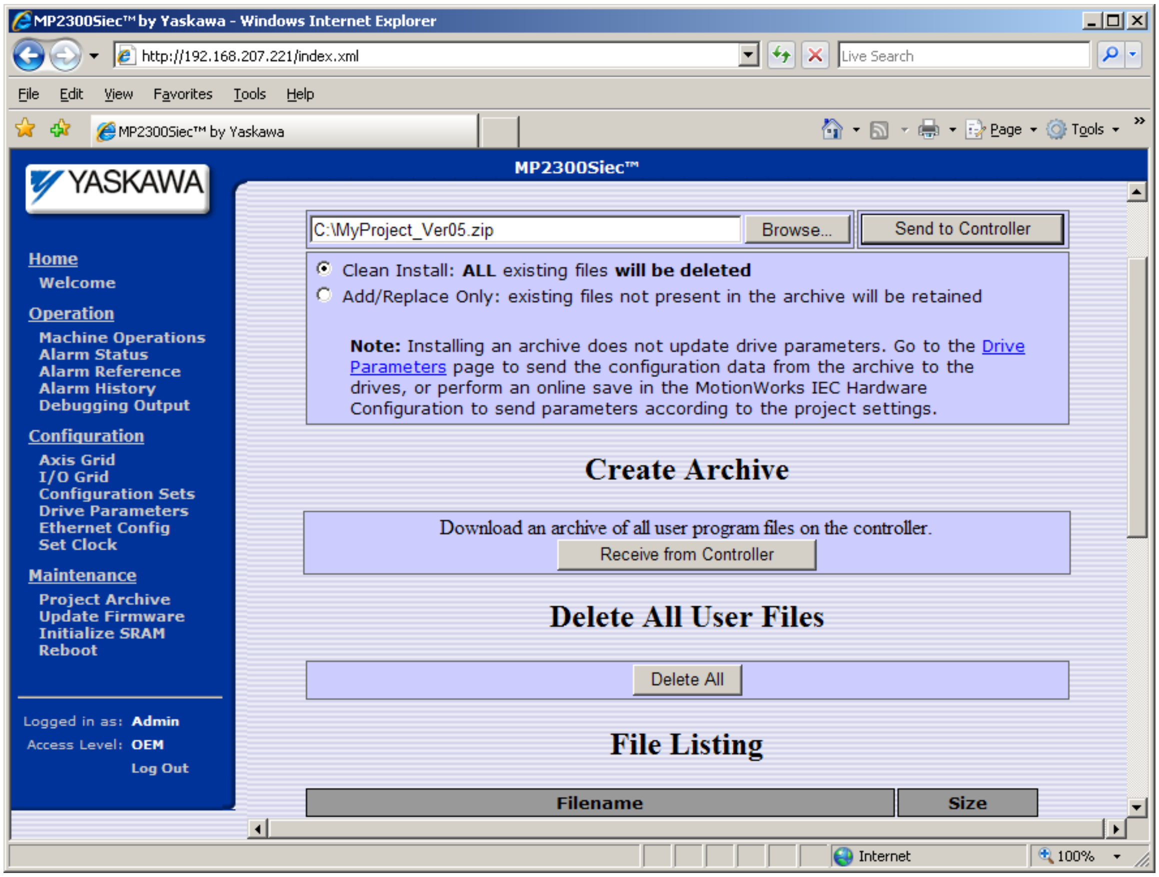
Welcome to the MP2300Siec™ Controller.

Please make your selection from the menu on the left.

Version Information

Version number	2.2.0
Build number	107
Build date	14 February 2012, 22:14:27
Software Platform	MP2300Siec
Hardware Platform	JEPMC-MP2300S-E-240
Model Number	PMC-U-MP23S16

3. Click on Project archive on the lower left side of the screen.



Logged in as: **Admin**

Access Level: **OEM**

Log Out

C:\MyProject_Ver05.zip

● Clean Install: **ALL** existing files **will be deleted**

○ Add/Replace Only: existing files not present in the archive will be retained

Note: Installing an archive does not update drive parameters. Go to the Drive Parameters page to send the configuration data from the archive to the drives, or perform an online save in the MotionWorks IEC Hardware Configuration to send parameters according to the project settings.

Create Archive

Download an archive of all user program files on the controller.

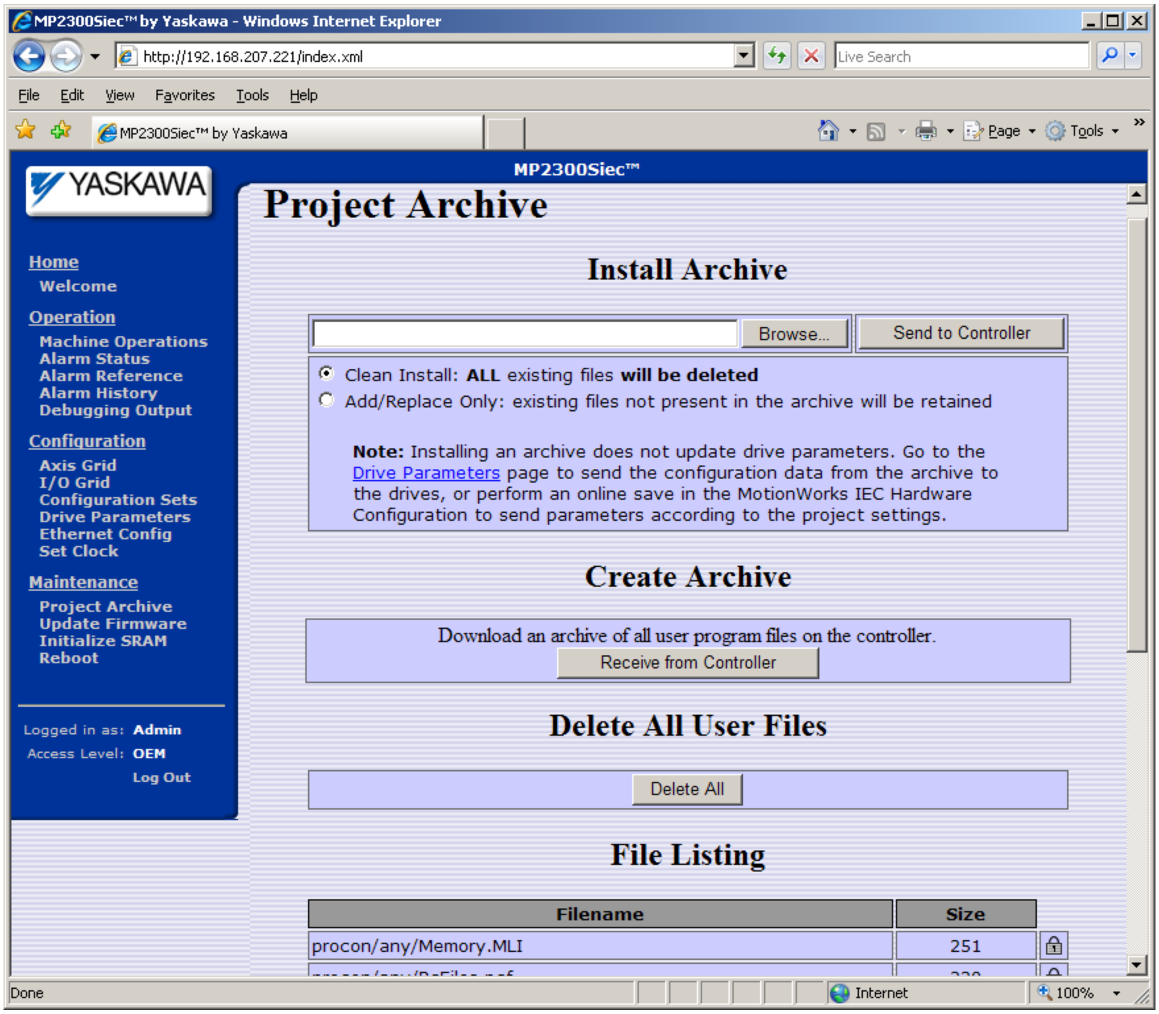
Delete All User Files

File Listing

Filename	Size

4. a) To create the archive, Press the “Receive from Controller” button and select the location to store the archive file.

b) To duplicate the controller, browse to the folder containing the archive file and press the “Send to Controller” button.



Project Archive

Install Archive

Browse... Send to Controller

● Clean Install: **ALL** existing files **will be deleted**

○ Add/Replace Only: existing files not present in the archive will be retained

Note: Installing an archive does not update drive parameters. Go to the Drive Parameters page to send the configuration data from the archive to the drives, or perform an online save in the MotionWorks IEC Hardware Configuration to send parameters according to the project settings.

Create Archive

Download an archive of all user program files on the controller.

Receive from Controller

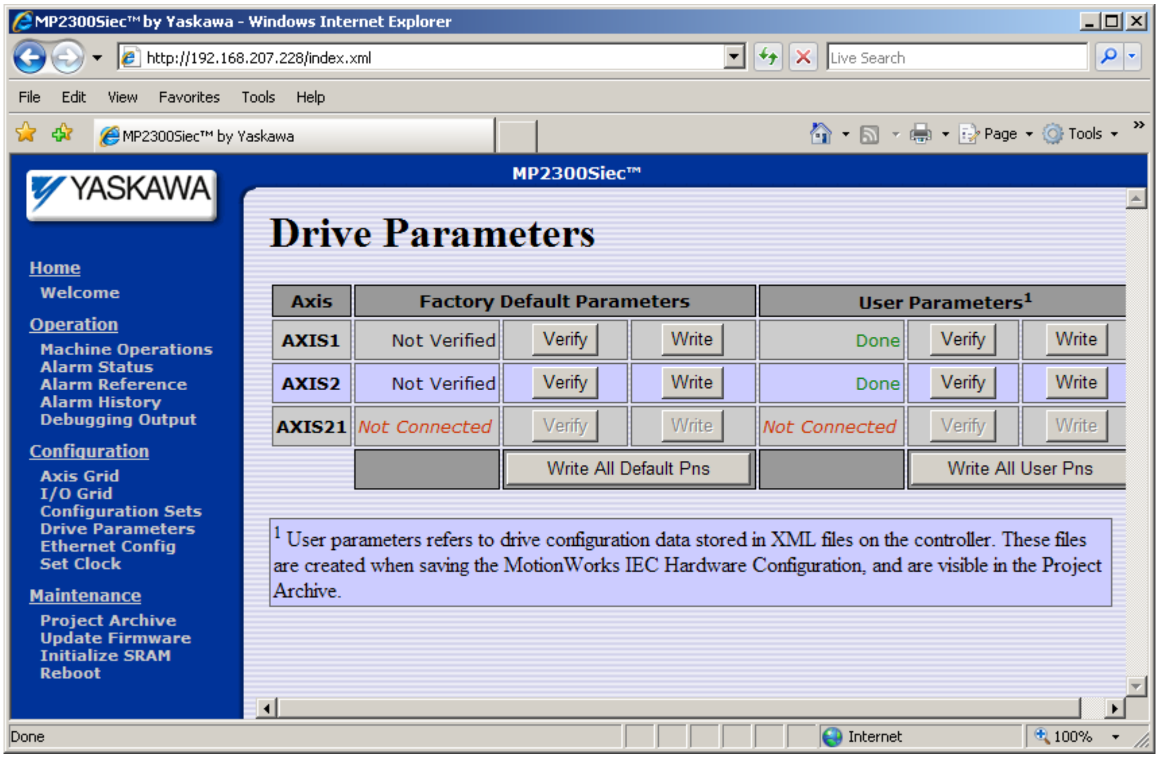
Delete All User Files

Delete All

File Listing

Filename	Size
procon/any/Memory.MLI	251

5. Click on the Drive Parameters link on the archive page shown above, or on the left side of the screen to send the amp parameters

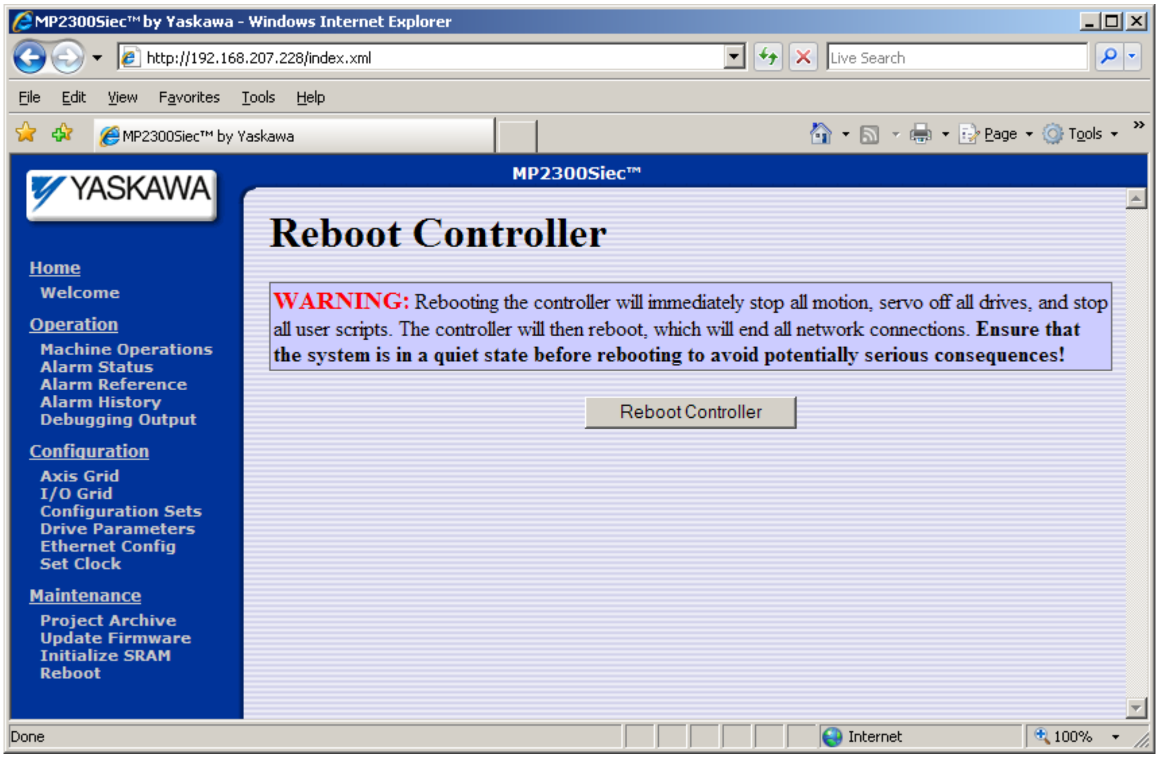


Drive Parameters

Axis	Factory Default Parameters			User Parameters[1]		
AXIS1	Not Verified	Verify	Write	Done	Verify	Write
AXIS2	Not Verified	Verify	Write	Done	Verify	Write
AXIS21	Not Connected	Verify	Write	Not Connected	Verify	Write
		Write All Default Pns			Write All User Pns	

[1] User parameters refers to drive configuration data stored in XML files on the controller. These files are created when saving the MotionWorks IEC Hardware Configuration, and are visible in the Project Archive.

6. Either click on the “Write All User Pns” button to write all of the drive parameters, or click on the “Write” button under User Parameters to write individual drives.



Reboot Controller

WARNING: Rebooting the controller will immediately stop all motion, servo off all drives, and stop all user scripts. The controller will then reboot, which will end all network connections. **Ensure that the system is in a quiet state before rebooting to avoid potentially serious consequences!**

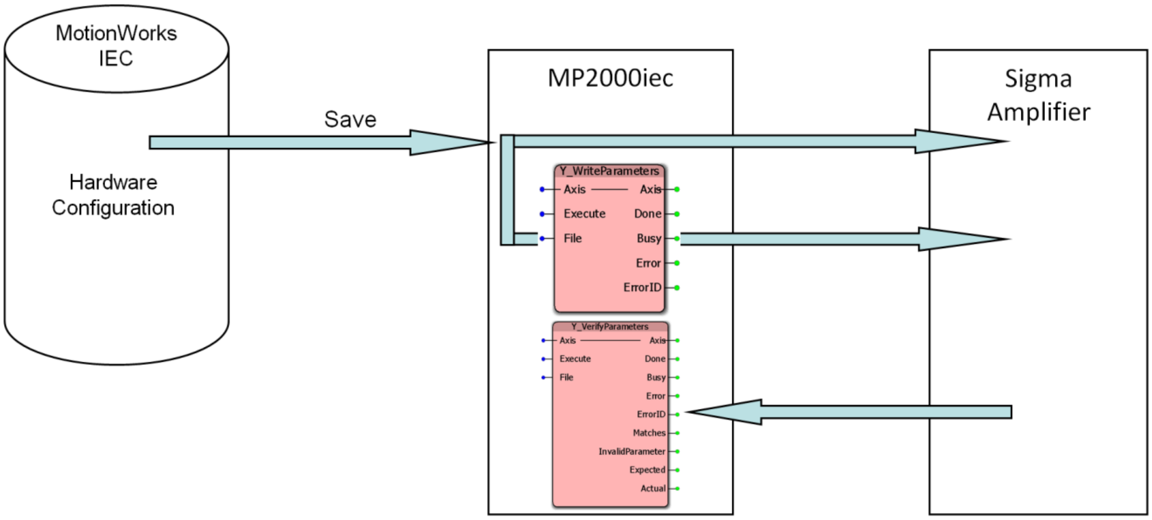
Reboot Controller

7. Reboot the controller or power cycle the system.

Programming for Amplifier Replacement

When an amplifier is replaced, the parameters in the old amplifier must be copied into the new amplifier. The IEC controller contains a copy of the parameters stored in the amplifier when the project was saved while online with the controller.

The following diagram represents how parameters are managed during program or configuration operations.



Amplifier parameters are stored in three places:

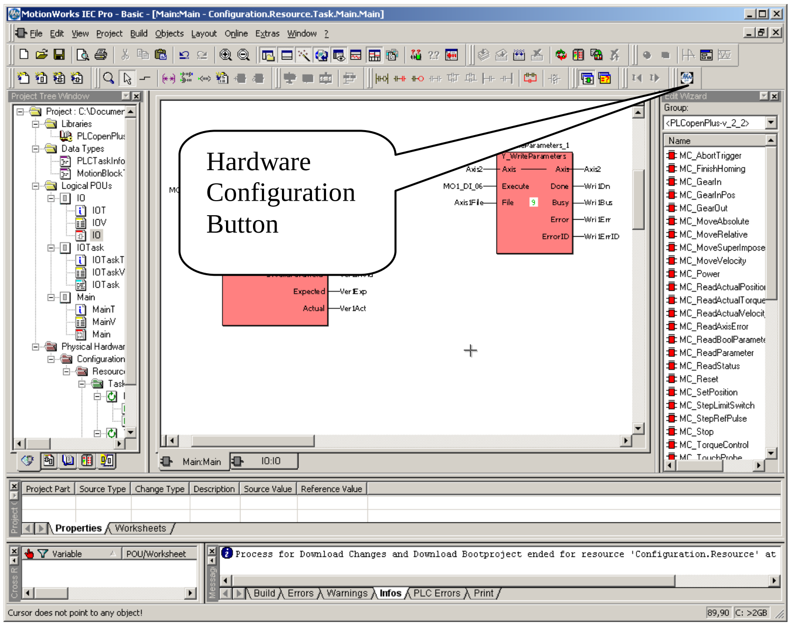
- The computer's hard drive (IEC Project subdirectory)
- The MPxxxxiec controller (As an XML file as seen in the Project Archive)
- The Amplifier

When the programmer is online and saves the Hardware Configuration, parameter files are saved to the hard drive, written to the controller, and the individual parameters are sent to the amplifier, so that all three locations contain the same data. The Y_WriteParameters function block sends the data from the parameter file stored on the controller to the amplifier. The Y_VerifyParameters function block reads the parameters from the drive and then compares them with the parameter file stored on the controller.

Procedure

The success of this procedure requires the programmer to make sure that the controller contains an identical copy of the parameters used in the amplifier. After the system is commissioned (everything is tuned and running correctly after a power cycle), be sure to connect with the Hardware Configuration and select the StartUp configuration, which will read all the amplifier parameters. Press ‘save’ which will then write the XML files into the controller project archive.

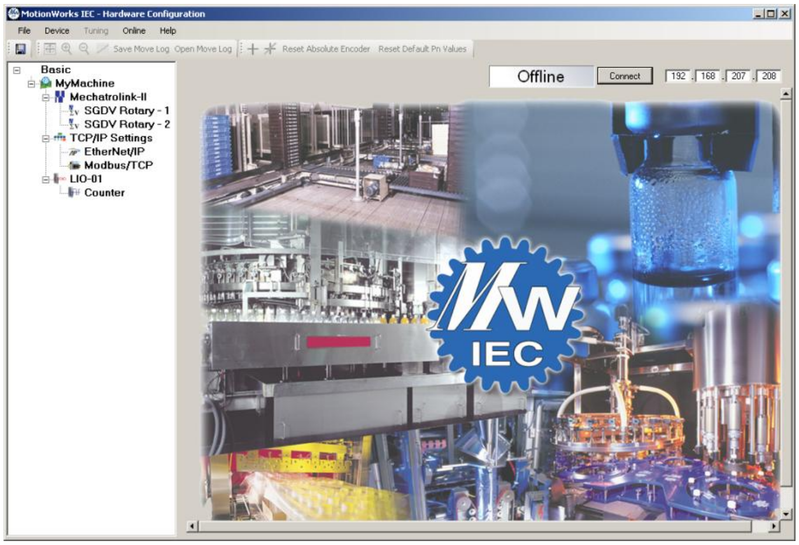
Saving amplifier parameters in the controller


MotionWorks IEC Pro - Basic - [Main:Main - Configuration.Resource.Task.Main.Main]
File Edit View Project Build Objects Layout Online Extras Window ?
Project Tree Window
Edit Wizard
Hardware Configuration Button
Main:Main
IO:IO
Project Part | Source Type | Change Type | Description | Source Value | Reference Value
Properties
Worksheets
Build Errors Warnings Infos PLC Errors Print

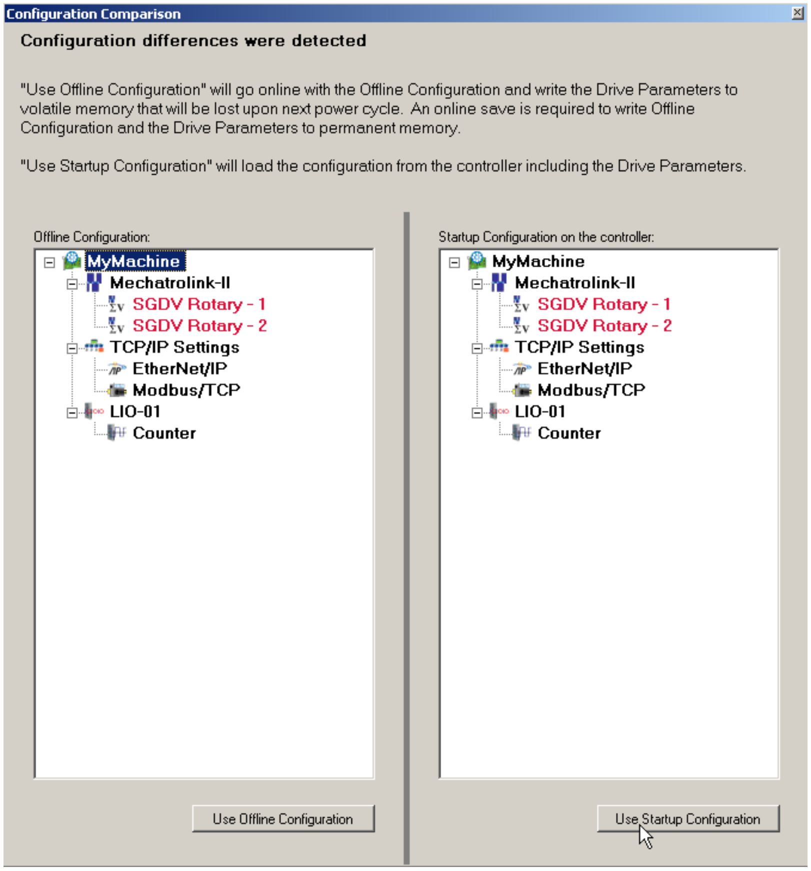
Process for Download Changes and Download Bootproject ended for resource 'Configuration.Resource' at

Cursor does not point to any object!

Select Hardware Configuration.


MotionWorks IEC - Hardware Configuration
File
Device
Tuning
Online
Help
Save Move Log
Open Move Log
Reset Absolute Encoder
Reset Default Pn Values
Basic
MyMachine
Mechatrolink-II
SGDV Rotary - 1
SGDV Rotary - 2
TCP/IP Settings
EtherNet/IP
Modbus/TCP
LIO-01
Counter
Offline
Connect
192
168
207
208
MW
IEC

Go online.



Configuration Comparison

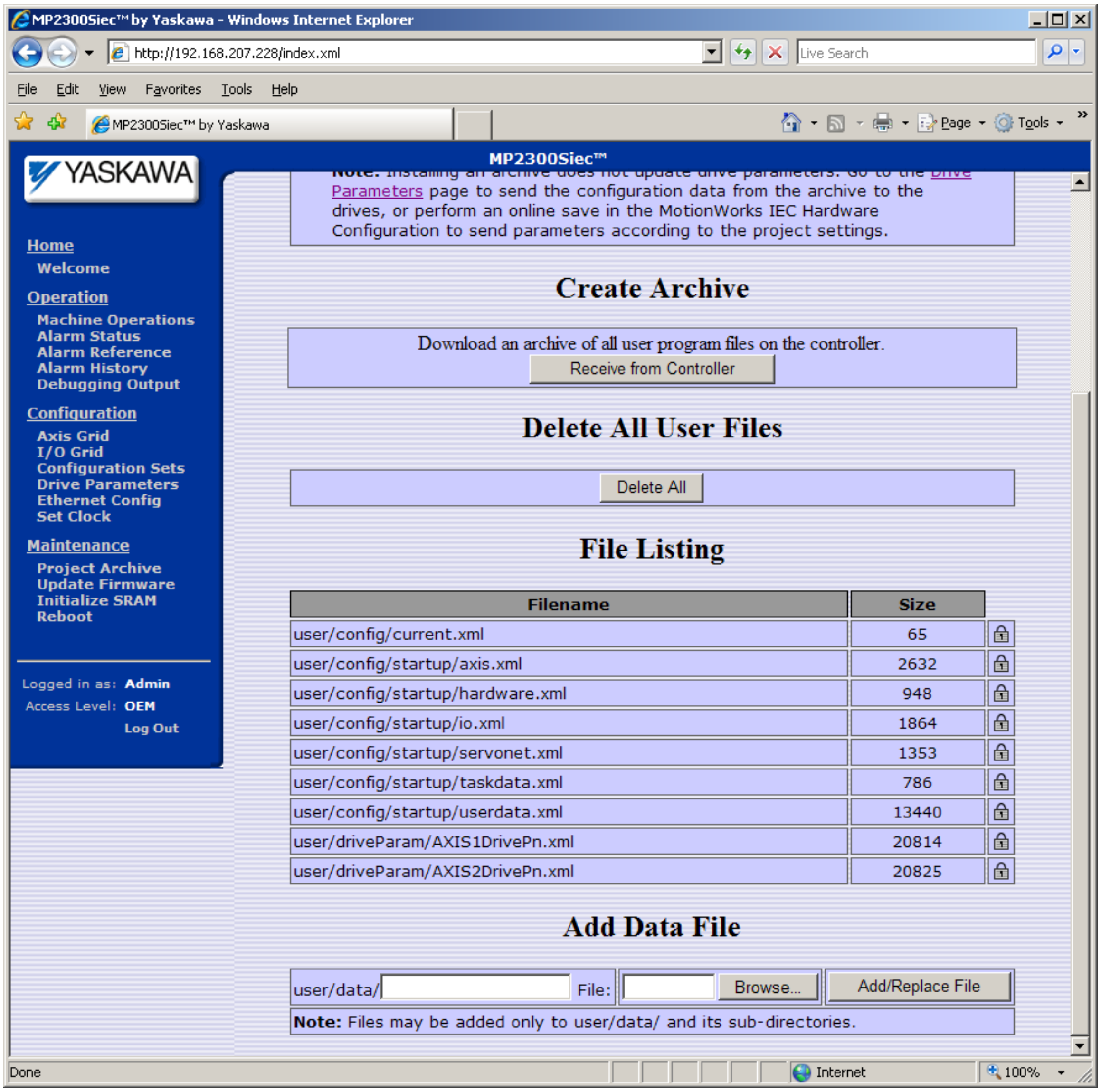
Configuration differences were detected

"Use Offline Configuration" will go online with the Offline Configuration and write the Drive Parameters to volatile memory that will be lost upon next power cycle. An online save is required to write Offline Configuration and the Drive Parameters to permanent memory.

"Use Startup Configuration" will load the configuration from the controller including the Drive Parameters.

If there are parameter differences detected choose to use the startup configuration.
To save the application select File-> Save from the menu or use the save button on the toolbar.

These files can be confirmed and viewed on the controllers Project Archive page from a web browser as shown below.



Note: Installing an archive does not update drive parameters. Go to the Drive Parameters page to send the configuration data from the archive to the drives, or perform an online save in the MotionWorks IEC Hardware Configuration to send parameters according to the project settings.

Create Archive

Download an archive of all user program files on the controller.

Receive from Controller

Delete All User Files

Delete All

File Listing

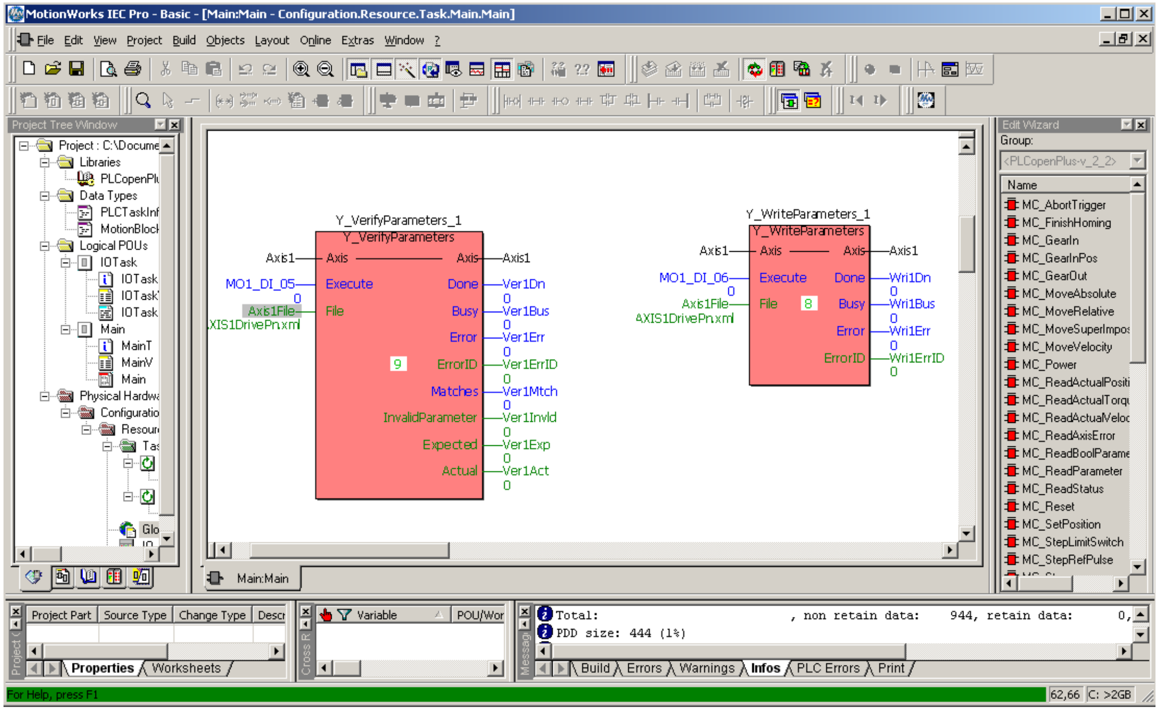
Filename	Size	
user/config/current.xml	65	🔒
user/config/startup/axis.xml	2632	🔒
user/config/startup/hardware.xml	948	🔒
user/config/startup/io.xml	1864	🔒
user/config/startup/servonet.xml	1353	🔒
user/config/startup/taskdata.xml	786	🔒
user/config/startup/userdata.xml	13440	🔒
user/driveParam/AXIS1DrivePn.xml	20814	🔒
user/driveParam/AXIS2DrivePn.xml	20825	🔒

Add Data File

user/data/ ______ File: ______ Browse... Add/Replace File

Note: Files may be added only to user/data/ and its sub-directories.

The next step is to program the amplifier parameter replacement in the application by using the Y_WriteParameters function block.



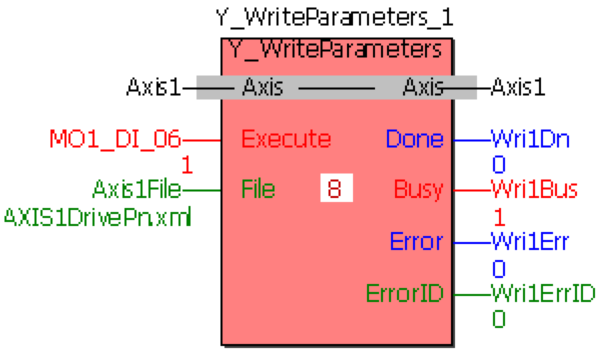
```
Total:                              , non retain data:    944, retain data:      0,
PDD size: 444 (1%)
```

The screenshot above shows a Y_VerifyParameters and a Y_WriteParameters function block. The Y_VerifyParameters function block could be used to automatically detect when an amplifier has been replaced in the field and trigger a Y_WriteParameters function block to run without user intervention. However, this automatic parameter write should be considered carefully. The Y_VerifyParameters function block will show differences under the following conditions:

1. If a parameter is changed using the Y_WriteDriveParameter function block.
2. If a parameter is changed using SigmaWin +.
3. If other motion functions change parameters.

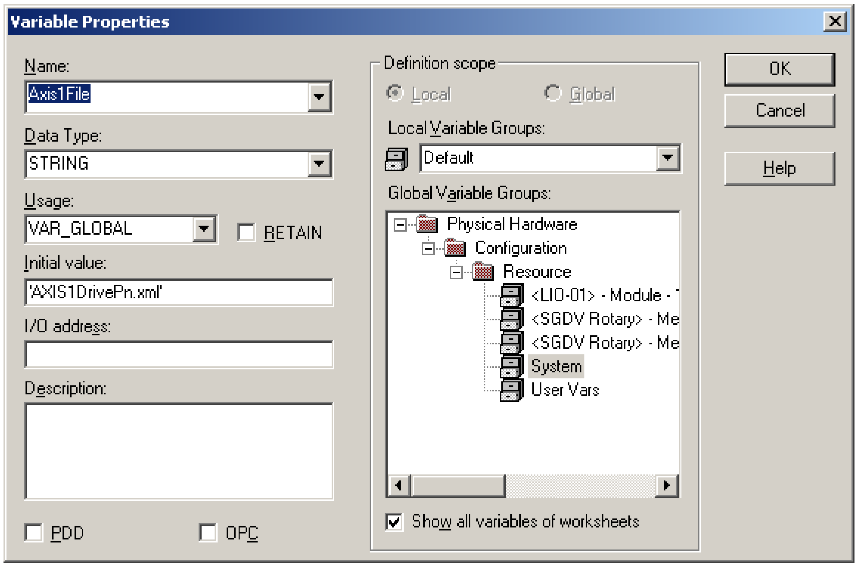
Also, as with most forms of non-volatile memory, the number of times the memory can be written is limited so it is not recommended to simply write drive parameters every time a difference is detected.

The programmer must insert a Y_WriteParameters function block for each amplifier, and determine how the operator or maintenance person will trigger the functions. In the example shown below, digital input 6 was used to trigger the write function block but we could have just as easily used an HMI variable available on a maintenance screen to trigger the write operation.



The file name string type variable must be set to 'AXIS*DrivePn.xml' with the * replaced with the axis number. This is the name of the file that the MotionWorks IEC Hardware Configuration application writes during the save operation. (These file names can be viewed on the Project Archive page from a web browser)

The variable definition of Axis1File is shown below:



Variable Properties

Name: Axis1File

Data Type: STRING

Usage: VAR_GLOBAL □ RETAIN

Initial value: 'AXIS1DrivePn.xml'

I/O address:

Description:

□ PDD □ OPC

Definition scope

● Local ○ Global

Local Variable Groups: Default

Global Variable Groups:

- Physical Hardware
  - Configuration
    - Resource
      - <LIO-01> - Module -
      - <SGDV Rotary> - Me
      - <SGDV Rotary> - Me
      - System
      - User Vars

☑ Show all variables of worksheets

Alternatively, a string literal may be defined as STRING#'AXIS1DrivePn.xml', with the string value after STRING# enclosed in single quotes.

Note About Absolute Encoder Motors

There are two methods to supply battery backup power to the absolute encoder:

1. Use an encoder cable with an inline battery.
2. Connect a battery to the CN1 connector of the amplifier.

If the absolute encoder is powered from a battery through the CN1 connector, the battery will be disconnected from the motor when replacing the amplifier. An amplifier alarm (A.810) will be generated, and must be cleared using the Y_ResetAbsoluteEncoder function block, the toolbar function in the Hardware Configuration or the web server. Next, the axis must be homed and the position must be defined using the MC_SetPosition block. See the next section “Programming applications for absolute encoder motor replacement” for a description of how to do this programmatically.

Programming applications for absolute encoder motor replacement

When an absolute encoder motor must be replaced, extra programming has to be included to allow the operator or maintenance personnel to perform the same operations as the programmer did when first commissioning the machine, namely, resetting the absolute encoder error, homing the axis, defining position, and setting the multi turn limit (if required for the application).

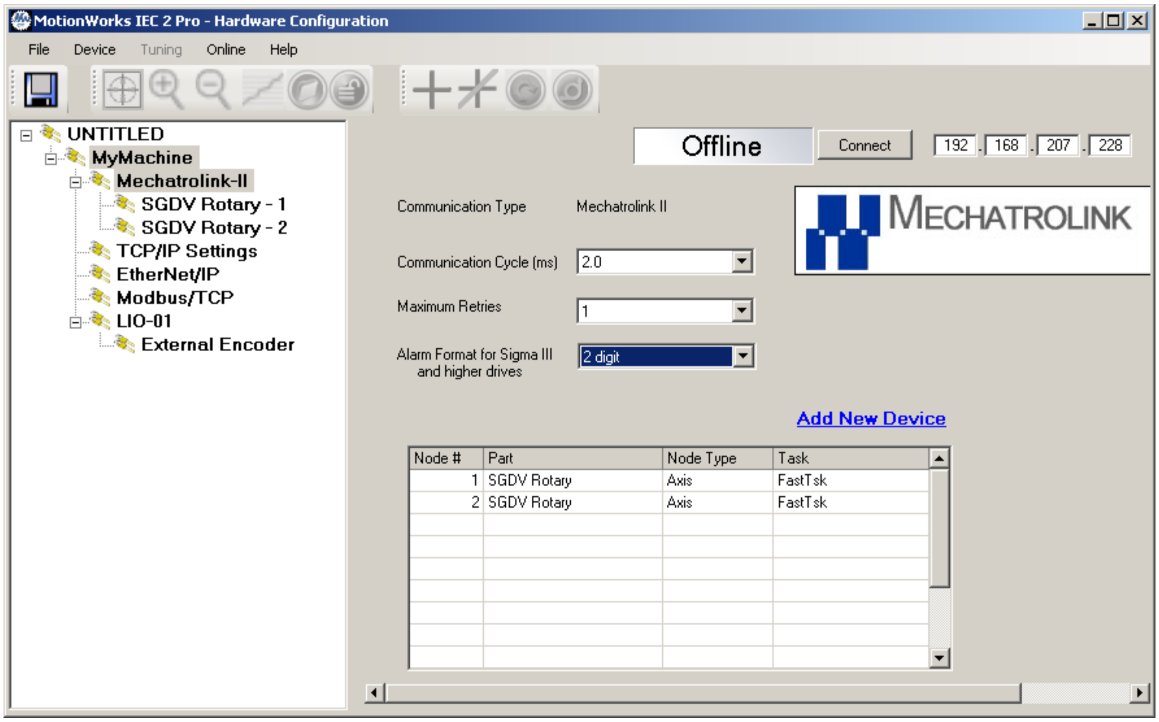
Procedure

After the absolute encoder motor is replaced, the program must clear the absolute encoder alarm and home the axis.

Begin by providing a routine that the maintenance personnel can trigger externally from an input or HMI.

Reading the axis error

When the absolute encoder motor is replaced or the motor is disconnected from the battery while power is off, the axis will have an absolute encoder alarm. The alarm will be an A.810 and the read axis error function block will show the error ID and error class. The error ID and error class will display one of two values. This depends on the setting chosen on the Mechatrolink screen in Hardware Configuration as shown below:



UNTITLED
- MyMachine
  - Mechatrolink-II
    - SGDV Rotary - 1
    - SGDV Rotary - 2
  - TCP/IP Settings
  - EtherNet/IP
  - Modbus/TCP
  - LIO-01
    - External Encoder

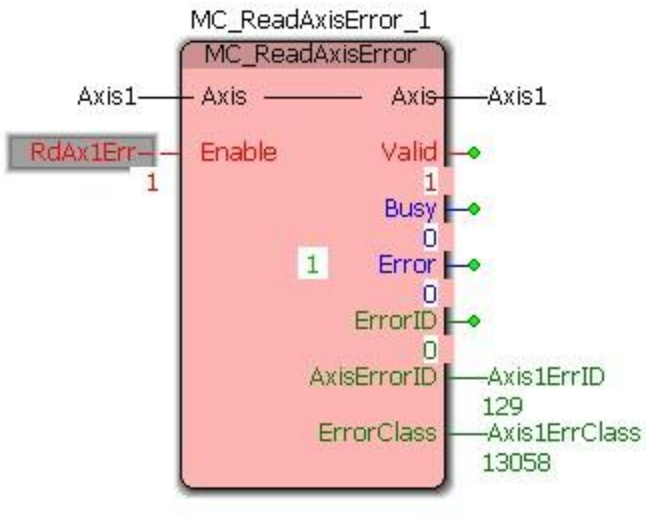
Communication Type: Mechatrolink II

Communication Cycle (ms): 2.0

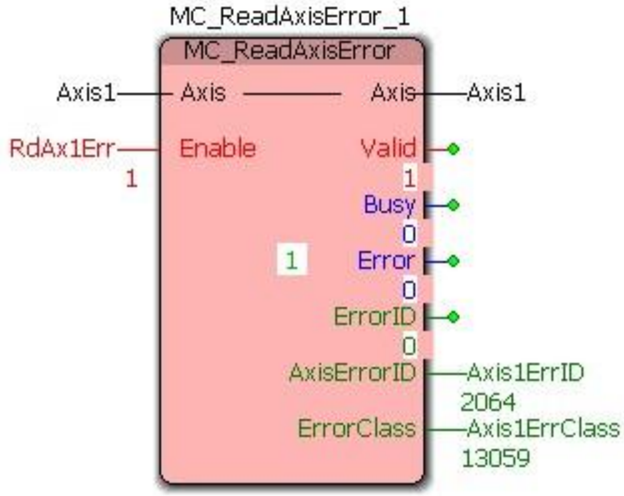
Maximum Retries: 1

Alarm Format for Sigma III and higher drives: 2 digit

Node #	Part	Node Type	Task
1	SGDV Rotary	Axis	FastTsk
2	SGDV Rotary	Axis	FastTsk



Two digit alarm format (default)

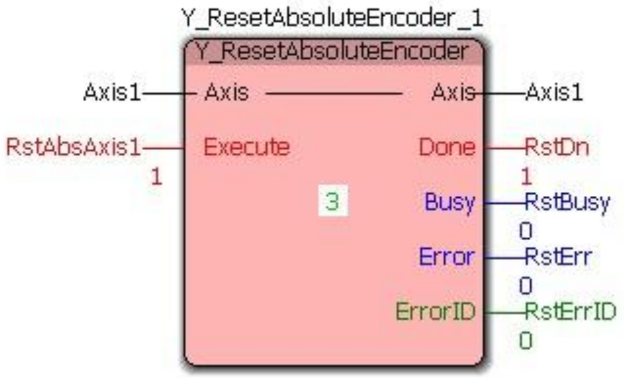


Three digit alarm format

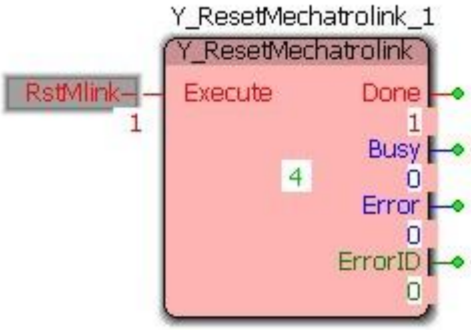
The error class [13058 (0x3302) for 2-digit and 13059 (0x3303) for 3-digit] indicates an amplifier alarm, and the error ID [129 (0x81) for 2-digit and 2064 (0x810) for 3-digit] is the encoder backup error indicating that the encoder was without power and the position is no longer valid.

Clearing the alarm & cycling power

Have the operator trigger a Y_ResetAbsoluteEncoder block and wait for it to complete. The following block uses variable “RstAbsAxis1” to trigger the Y_ResetAbsoluteEncoder block.



After the Y_ResetAbsoluteEncoder block has been run, the amplifier must go through a power cycle to clear the alarm. The alarm will persist until the power has been cycled. The operator can be instructed to power cycle the machine, or you can program a Y-ResetMechatrolink to be run. The Y-ResetMechatrolink function block does a software power cycle of the drives, which will clear all alarms and apply any parameters that require a power cycle.



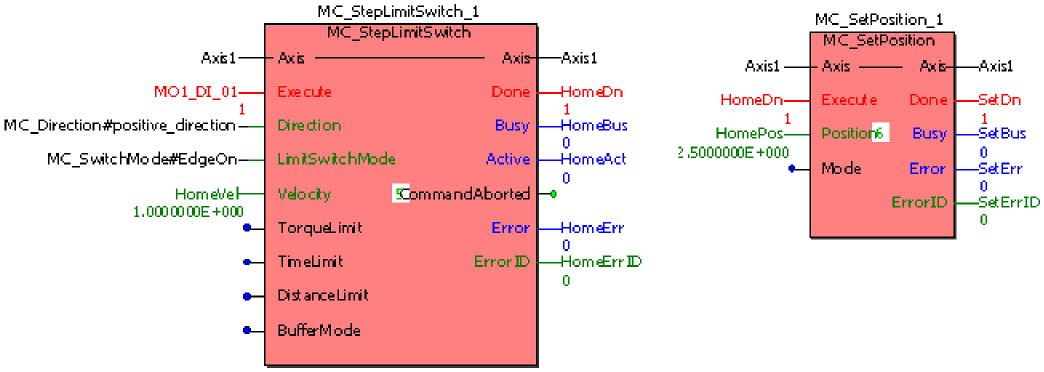
The Y-ResetMechatrolink function block will take several seconds to complete. Make sure the done output of the function block is on before continuing.

Homing the Axis & setting position

When power is restored, it is very important that the machine not go into a running state. The axis position has not been defined and will not be at the correct position for normal machine operation. The machine should be prevented from running until the homing procedure has been completed.

Provide a method to run the homing block used to originally commission the machine and set the position of the axis after the homing block has completed. The set position is required, because this stores the absolute encoder offset in non volatile memory so that the axis will “remember” its position when the machine is power cycled.

In the following example, digital input 1 is used to trigger a home to limit switch block, and when that completes, a MC_SetPosition is run.



Once the set position block has been executed the machine is ready to run.

Note about the Multi Turn Limit

If the amplifier has an absolute encoder and the multi turn limit Pn205 was changed from its default value of 65535, the new multi turn limit must be sent to the motor. In the past, this had to be done by using a hand held operator, SigmaWin+, or the web browser. Now only the Y_ResetAbsoluteEncoder function block needs to be run and the multi turn limit will be set.

Sequence of operations

Follow these steps before the machine is put into production, to allow your system the most flexibility in replacing components:

1. Open the Hardware Configuration and save while online.

2. Create an archive of the controller to be used if a new controller is required, or to duplicate the system, and keep it in a safe place.
3. Program the application to allow a maintenance person to trigger a MC_WriteParameters block for the axis (axes) that must have the parameters re-written.
4. Program the application to run a Y_ResetAbsoluteEncoder function block and perform a homing routine after an absolute encoder motor has been replaced, or has had its battery disconnected.
5. Interlock the machine operation with a MC_VerifyParameters block to insure that the machine is not run with an amplifier that has default parameters. Make sure to test for a subset of parameters as the application may change some parameters during machine operation.

Conclusion

Resuming production after replacing a controller, amplifier, or absolute encoder motor can be very easy if the machine designer incorporates maintenance functions into the program. This allows end users to perform replacement of components on their machines without requiring use of any special programming software.