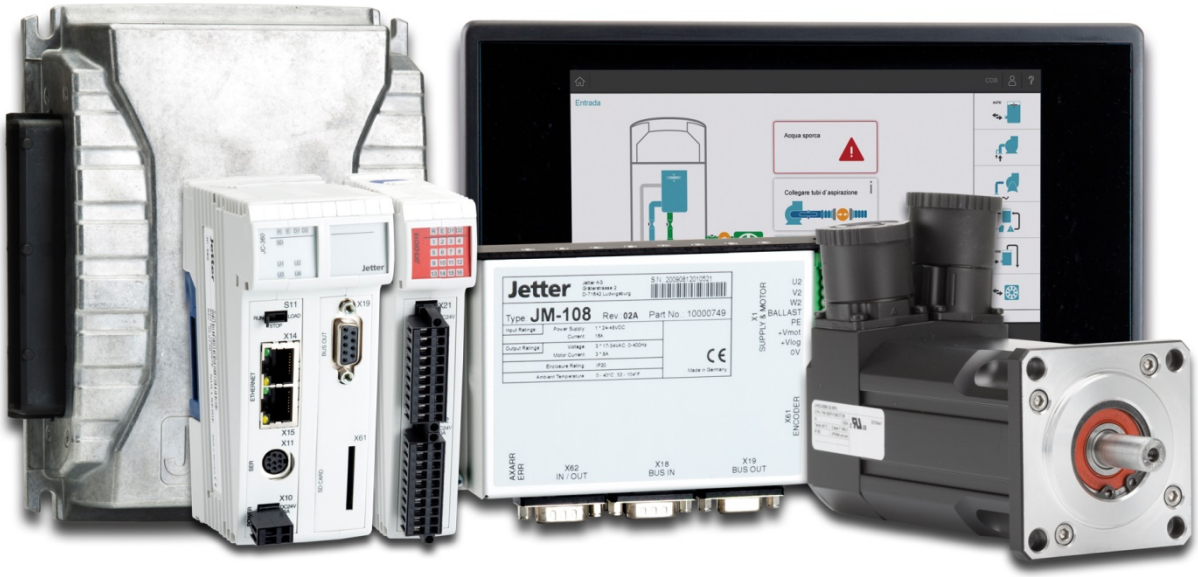
Jetter automation



MCX – Diagnose

Application Note 055

60886067_00

We automate your success.

Dieses Dokument wurde von der Jetter AG mit der gebotenen Sorgfalt und basierend auf dem ihr bekannten Stand der Technik erstellt. Änderungen und technische Weiterentwicklungen an unseren Produkten werden nicht automatisch in einem überarbeiteten Dokument zur Verfügung gestellt. Die Jetter AG übernimmt keine Haftung und Verantwortung für inhaltliche oder formale Fehler, fehlende Aktualisierungen sowie daraus eventuell entstehende Schäden oder Nachteile.

Jetter AG
Gräterstraße 2
71642 Ludwigsburg
Deutschland

www.jetter.de

Telefon:

Zentrale	+49 7141 2550-0
Vertrieb	+49 7141 2550-531
Technische Hotline	+49 7141 2550-444

E-Mail:	info@jetter.de
Technische Hotline	hotline@jetter.de
Vertrieb	vertrieb@jetter.de

Produktname	MCX – Diagnose
Dokumentart	Application Note 055
Originaldokument	
Dokumentenversion	1.00
Artikelnummer	60886067_00
Ausgabedatum	21.05.2021

Inhaltsverzeichnis

1 Einleitung

Zur Programmierung, Inbetriebnahme, Tests und Betrieb einer Anwendung mit MCX gehört auch der Umgang mit den Diagnosemitteln, die die MCX und die Programmierumgebung JetSym zur Verfügung stellen.
Hierzu zählt aber nicht nur die Auswertung von Fehlern, sondern auch die Prüfung des gewünschten Verhaltens bzw. Programmablaufs.
In dieser Application Note soll dargestellt werden, welche Diagnosewerkzeuge von JetSym angeboten und wie sie angewendet werden. Des Weiteren werden die Auswertemöglichkeiten innerhalb eines Applikationsprogramms vorgestellt.

2 Fehlerarten

2.1 Benutzerfehler

Fehlbedienung der MCX: z.B. Vorgabe von ungültigen Werten, Verfahrbefehle auf inaktive Objekte, …

2.2 Achsfehler

Fehler, die am Drive aufgetreten sind, z.B. Schleppfehler, Zwischenkreisüberspannung, Geberfehler, …

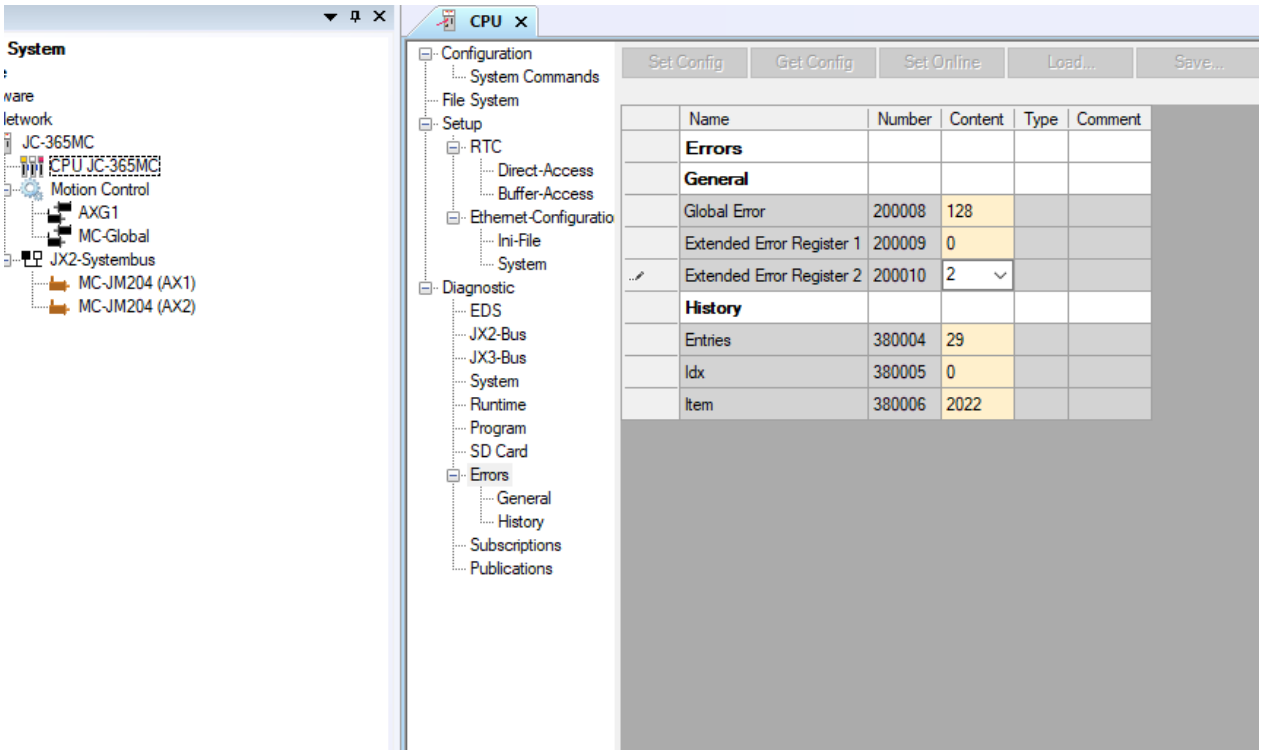
2.3 Verbundfehler

Tritt in einem aktiven Verbund mindestens ein Achsfehler auf, so wird dies ein Verbundfehler.

3 JetSym

3.1 Steuerung - Überprüfung der Systemfehler

Im Hardware-Setupfenster der eingestellten Steuerung lassen sich spezifische Systemregister anzeigen.

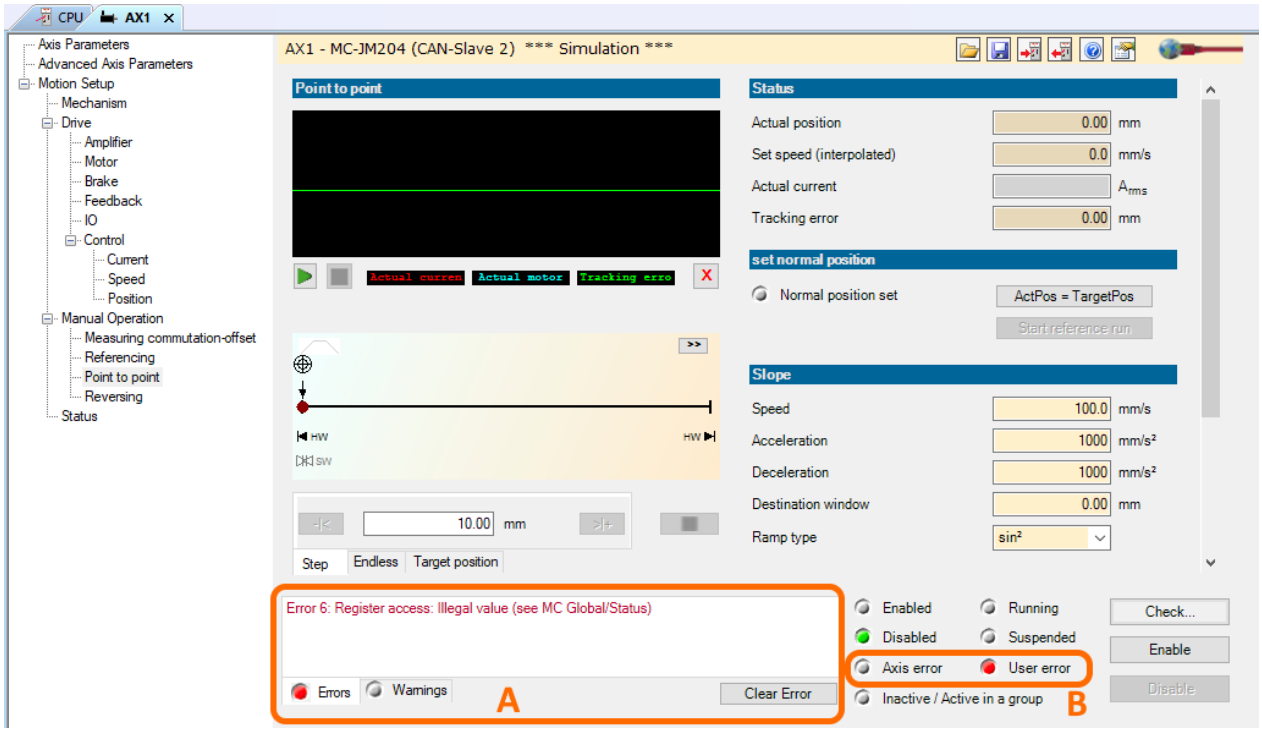


Diese umfassen auch die erkannten Systemfehler:
Register 200008: Fehlerregister
Bit 7: Fehler im erweiterten Fehlerregister
Register 200010: Erweitertes Fehlerregister 2
Bit 1: Fehler am MCX-Objekt

Eine ausführliche Beschreibung der weiteren Bits in diesen Registern kann in den Handbüchern der jeweiligen Steuerung nachgelesen werden.
Bit 7 in Register 200008 beschreibt nur, dass Fehler in den erweiterten Fehlerregistern angezeigt werden, nicht dass ein Fehler in der MCX festgestellt wurde.
Bit 1 in Register wird gesetzt, sobald die MCX einen Fehler festgestellt hat. Ein Löschen des Fehlers mit ClearErrors() setzt dieses Bit nicht zurück. Es muss gezielt zurückgesetzt werden. Das zugehörige Bit 7 im Register 200008 muss ebenfalls separat zurückgesetzt werden. Werden in den erweiterten Fehlern noch andere Fehler angezeigt, so bleibt Bit 7 im Register 200008 gesetzt.

3.2 MotionSetup – Einzelachse

Auf jeder Inbetriebnahme-Seite befindet sich eine Übersicht über anstehende Fehler oder Warnungen.

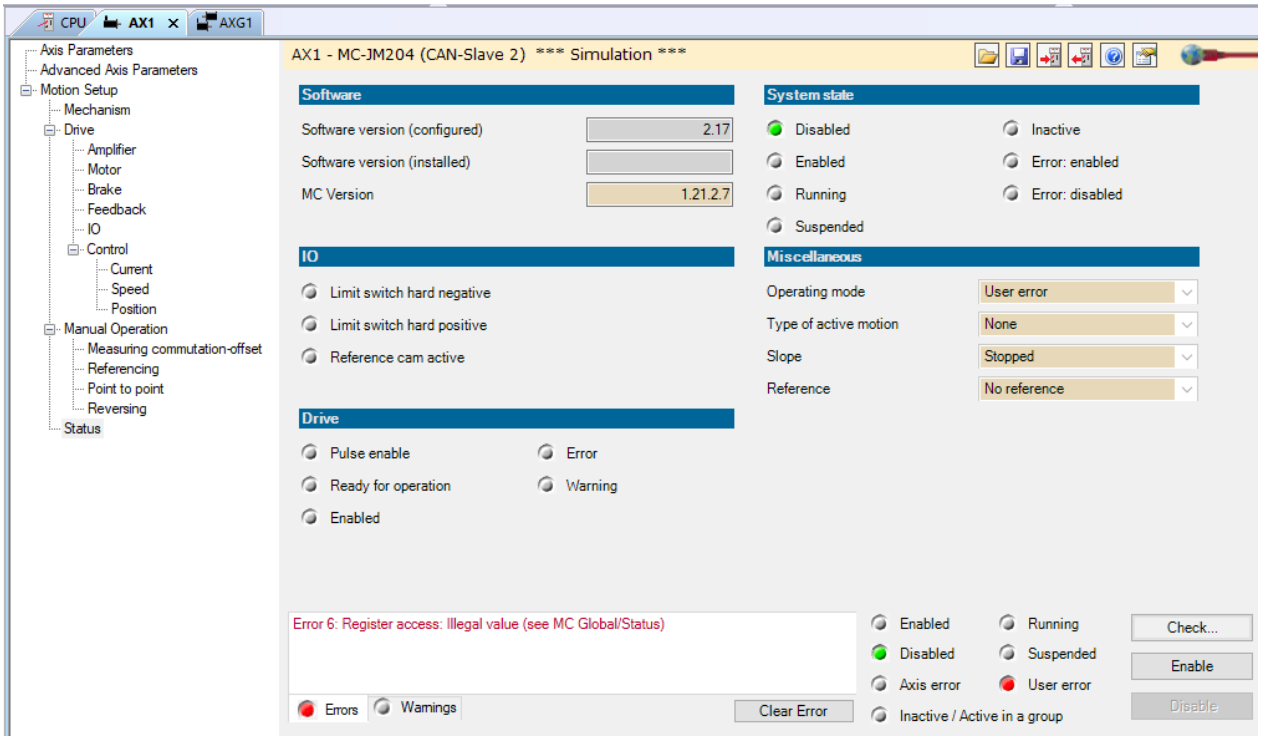


Stehen Fehler oder Warnungen an, werden diese durch die rote Anzeige des jeweiligen Reiters signalisiert.
Ist der Reiter "Errors" aktiv, werden hier die Fehler, die diese Achse betreffen, in Feld A aufgelistet.
Analog wird bei Auswahl des Reiters "Warnings" die anstehenden Warnungen angezeigt.
Zusätzlich wird im Feld B noch angezeigt, ob ein Achs- oder Benutzerfehler aktiv ist.
Mit der Schaltfläche "Clear Error" lassen sich anstehende Fehler quittieren.

3.2.1 Statusübersicht - Einzelachse

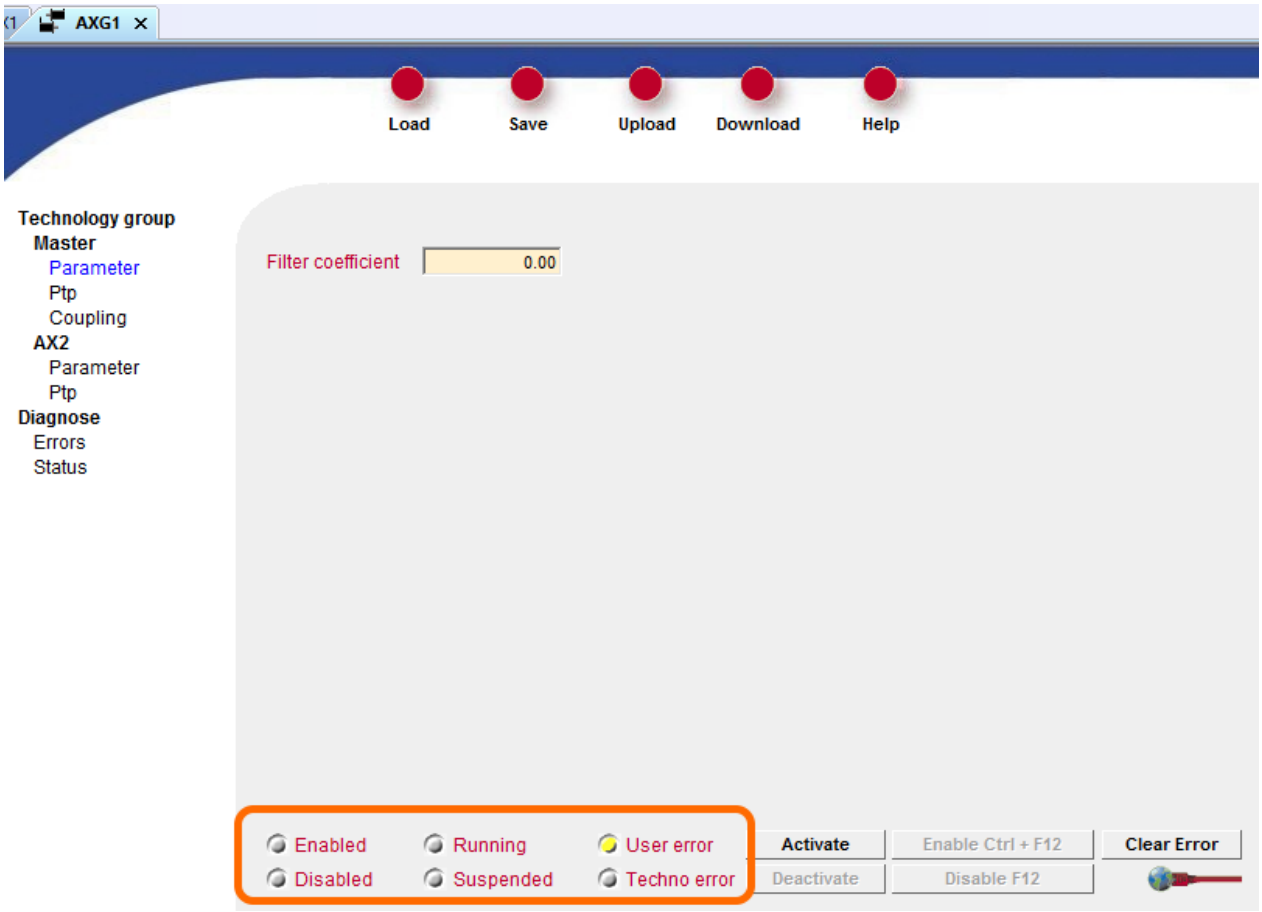
Auf dieser Übersichtsseite werden wichtige Statusinformationen über eine Einzelachse angezeigt.

- Betriebssystemversion des Drives
- MCX-Version
- Schalter
- Hardware-Status des Drives
- Anzeige der MCX-Zustände
- Weitere
    - Betriebsart
    - Aktive Bewegungsart
    - Rampenstatus
    - Referenzstatus


CPU
AX1
AXG1
Axis Parameters
Advanced Axis Parameters
Motion Setup
Mechanism
Drive
Amplifier
Motor
Brake
Feedback
IO
Control
Current
Speed
Position
Manual Operation
Measuring commutation-offset
Referencing
Point to point
Reversing
Status
AX1 - MC-JM204 (CAN-Slave 2) *** Simulation ***
Software
Software version (configured)
2.17
Software version (installed)
MC Version
1.21.2.7
System state
Disabled
Inactive
Enabled
Error: enabled
Running
Error: disabled
Suspended
IO
Limit switch hard negative
Limit switch hard positive
Reference cam active
Miscellaneous
Operating mode
User error
Type of active motion
None
Slope
Stopped
Reference
No reference
Drive
Pulse enable
Error
Ready for operation
Warning
Enabled
Error 6: Register access: Illegal value (see MC Global/Status)
Enabled
Running
Disabled
Suspended
Axis error
User error
Inactive / Active in a group
Check...
Enable
Disable
Errors
Warnings
Clear Error

3.3 MotionSetup – Verbund

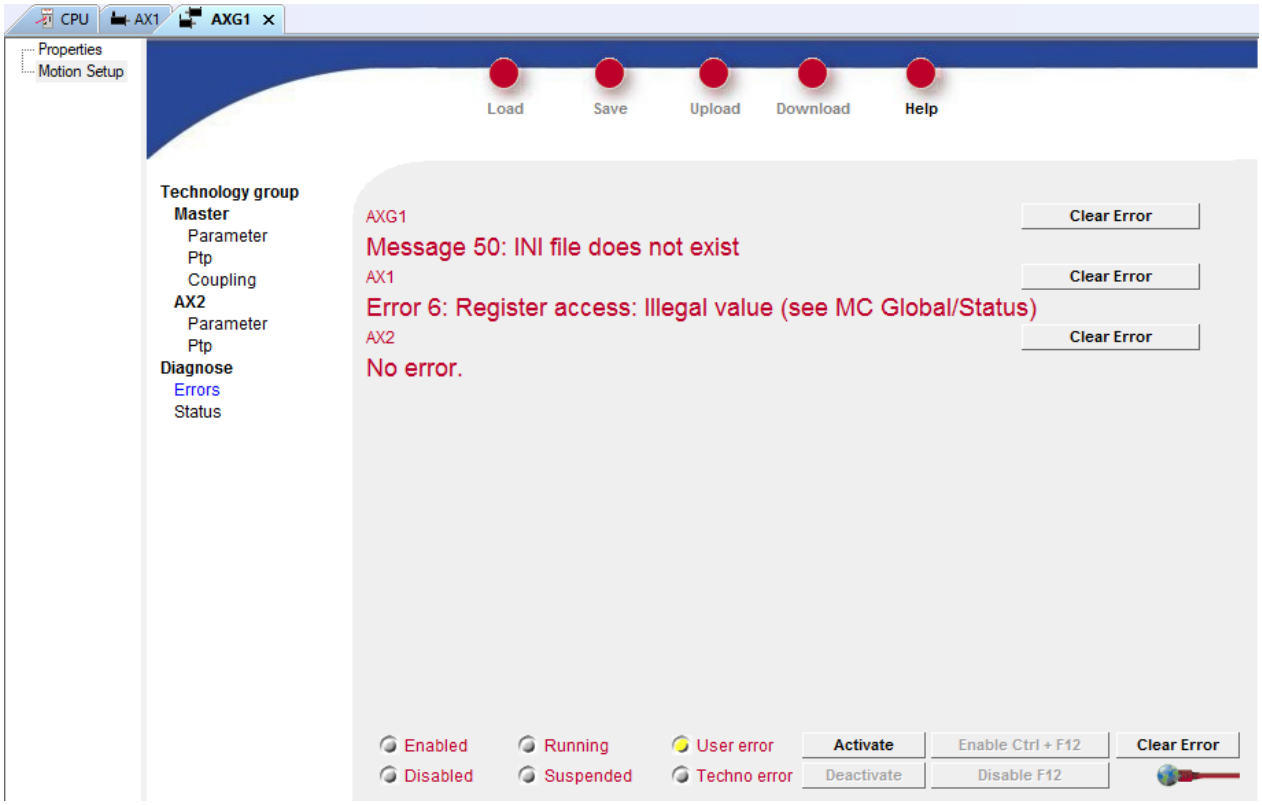
Auf der Inbetriebnahme-Seite eines Achsverbunds wird neben dem aktiven Zustand des Verbunds auch angezeigt, ob ein Benutzer- oder Verbundfehler vorliegt.



Mit der Schaltfläche "Clear Error" lassen sich anstehende Fehler quittieren.

3.3.1 Fehlerübersicht - Verbund

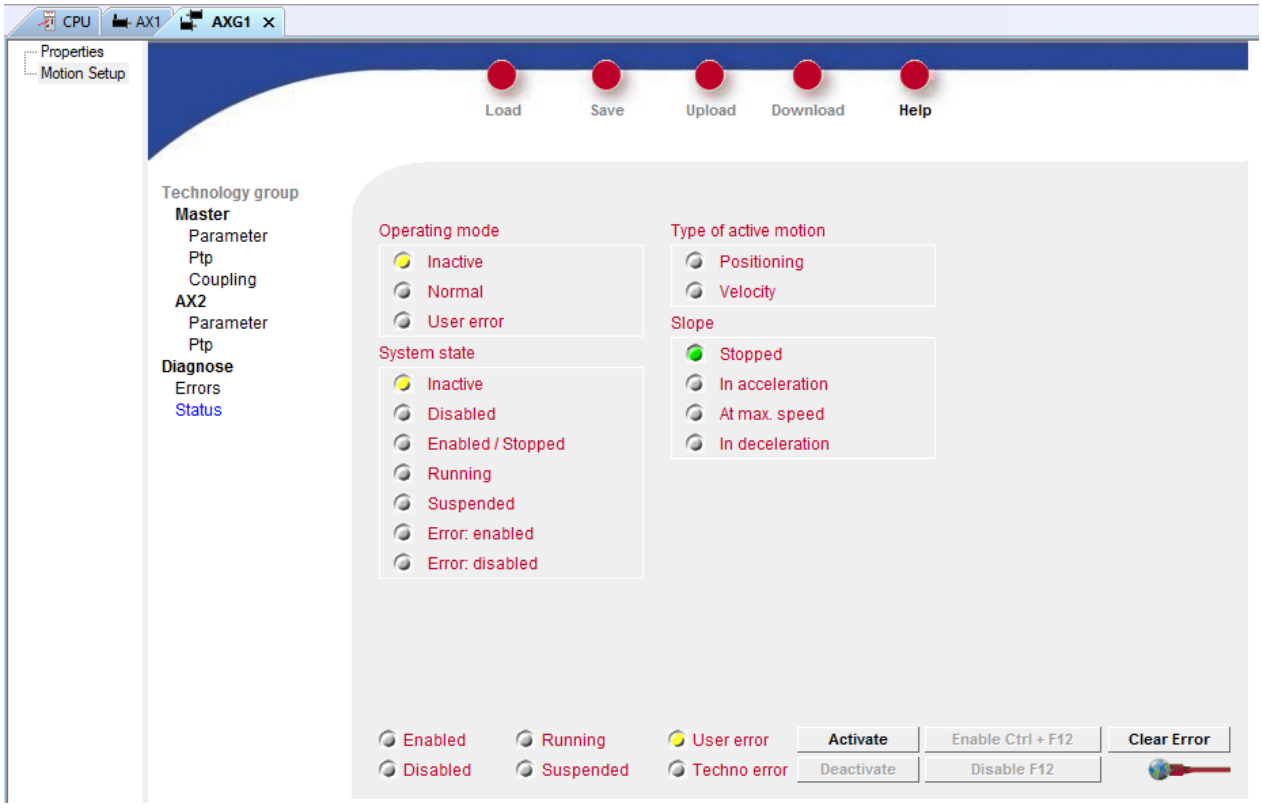
In dieser Übersicht werden die aktuellen Fehler, Warnungen und Meldungen des Verbunds und seiner verbundenen Objekte aufgelistet.
Mit der Schaltfläche "Clear Error" lassen sich anstehende Fehler quittieren. Wird "Clear Error" auf den Verbund angewandt, löscht dies auch die Fehler der Verbundmitglieder.



3.3.2 Statusübersicht - Verbund

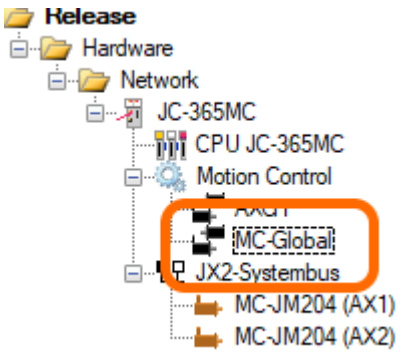
Analog zur Statusübersicht der Einzelachse werden hier die aktuellen Zustände des Achsverbunds angezeigt.
Verfügbar sind 4 Kategorien:

- Betriebsart
- Betriebszustand
- Aktive Bewegungsart
- Rampenstatus


CPU
AX1
AXG1
Properties
Motion Setup
Load
Save
Upload
Download
Help
Technology group
Master
Parameter
Ptp
Coupling
AX2
Parameter
Ptp
Diagnose
Errors
Status
Operating mode
Inactive
Normal
User error
System state
Inactive
Disabled
Enabled / Stopped
Running
Suspended
Error: enabled
Error: disabled
Type of active motion
Positioning
Velocity
Slope
Stopped
In acceleration
At max. speed
In deceleration
Enabled
Disabled
Running
Suspended
User error
Techno error
Activate
Deactivate
Enable Ctrl + F12
Disable F12
Clear Error

3.4 MotionSetup – MC-Global

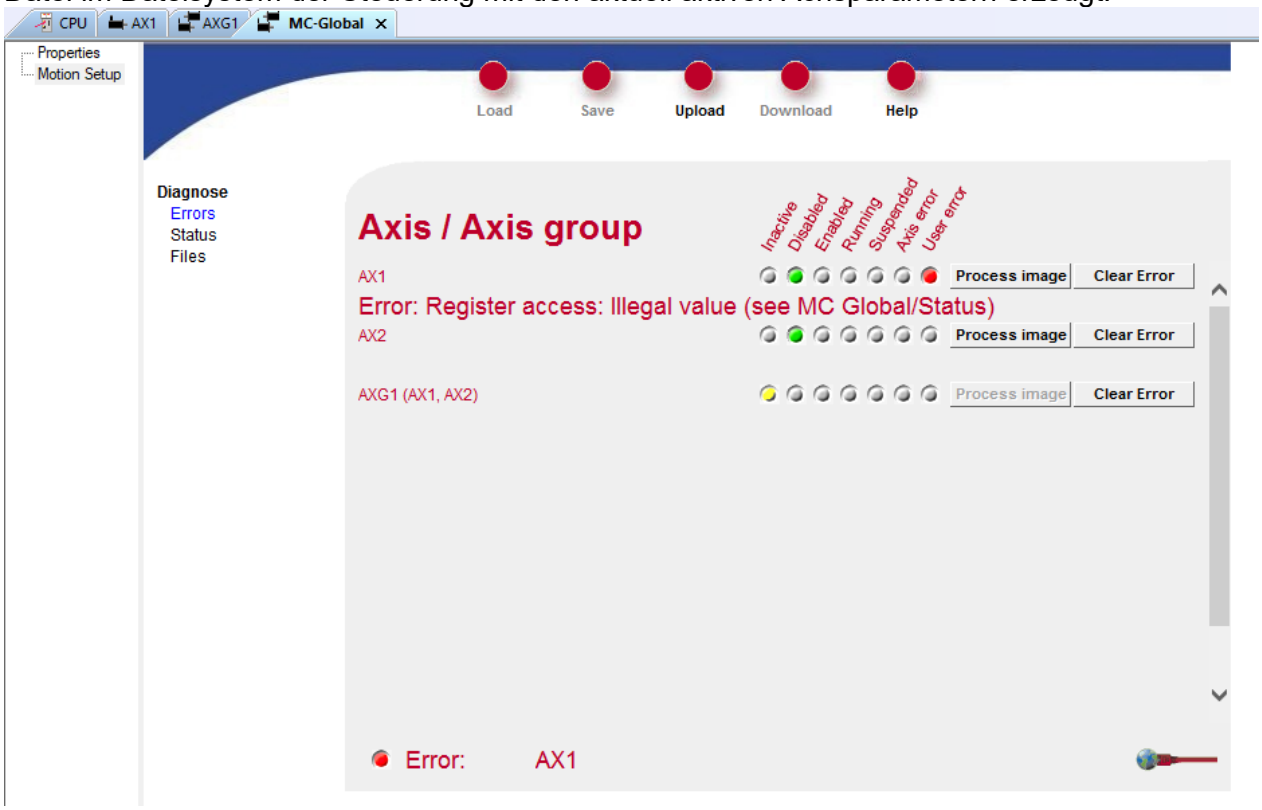
MC-Global liefert eine Übersicht über das gesamte MCX-System.



Im Hardwarebaum wird der Knoten "MC-Global" erzeugt, sobald ein MCX-Objekt angelegt wird.

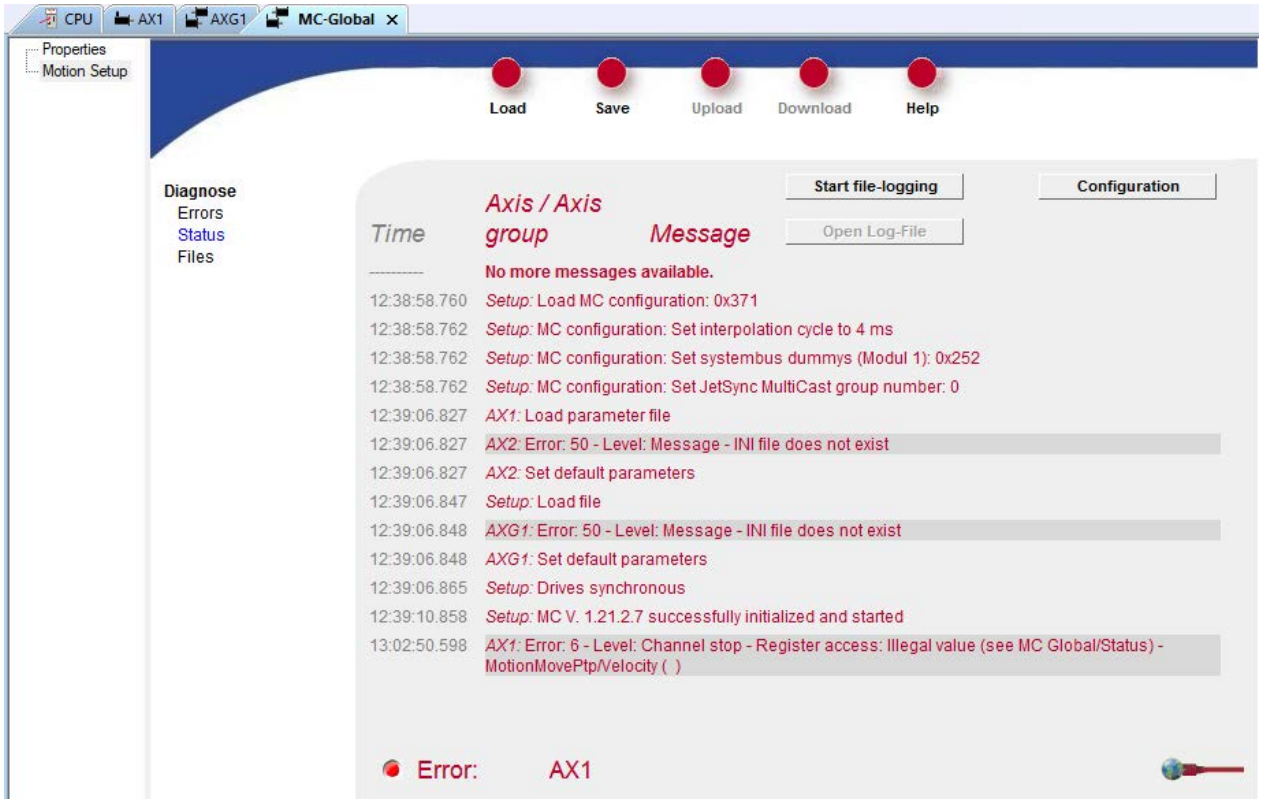
3.4.1 Fehler

In dieser Ansicht werden alle im Hardware-Manager angelegten Achsobjekte aufgelistet.
Der Betriebszustand jedes Objekts wird dargestellt.
Fehler und Warnungen werden für jedes Objekt aufgelistet.
Mit "Clear Error" lassen sich anstehende Fehler quittieren. Wird "Clear Error" auf den Verbund angewandt, löscht dies auch die Fehler der Verbundmitglieder.
Für spätere Analysen kann ein Prozessabbild erzeugt werden. Hierbei wird eine "*_Tmp99.ini"-Datei im Dateisystem der Steuerung mit den aktuell aktiven Achsparametern erzeugt.



3.4.1.1 Status

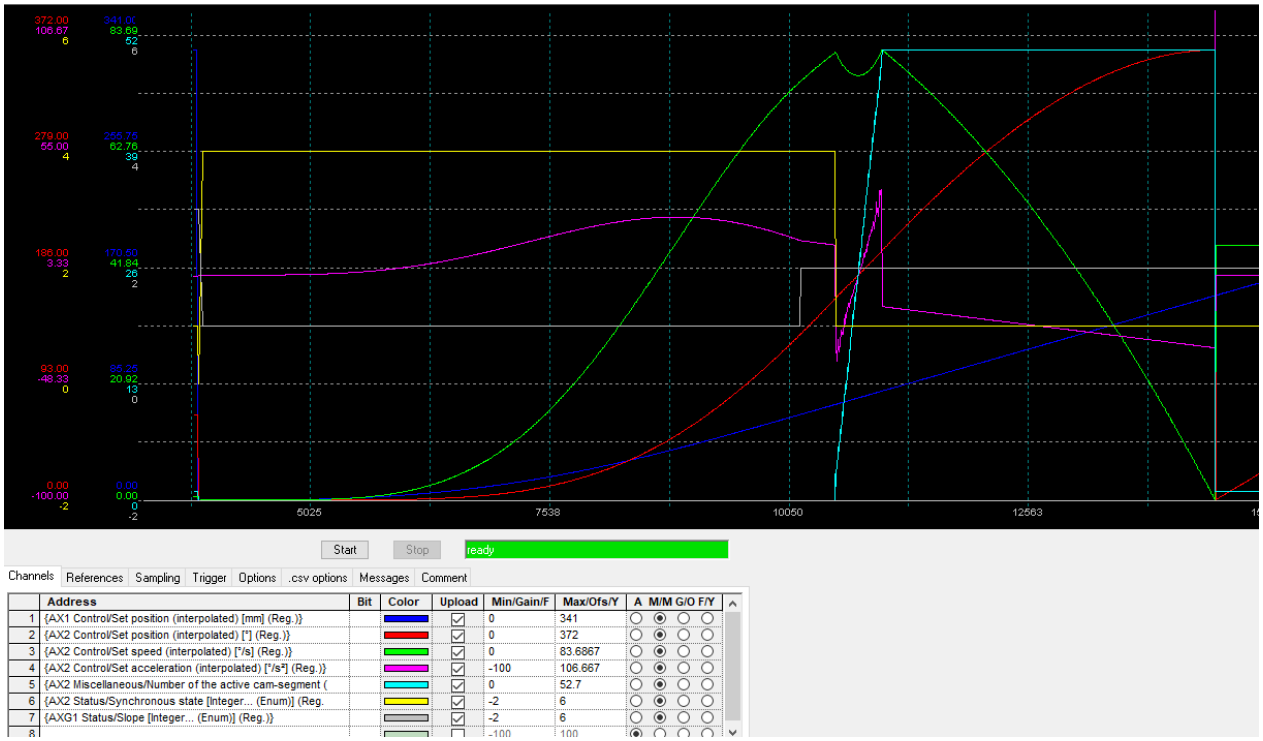
Die MCX speichert in einem Ringpuffer die Zugriffe bzw. Befehle.
Somit kann der zeitliche Verlauf von MCX-Ereignissen nachvollzogen werden.



In diesem Beispiel ist der Bootverlauf dargestellt, welcher mit "…successfully initialized and started" endet. Sie sehen aber auch, dass beim Aufruf eines MovePtp an der Achse 1 ein ungültiger Wert eingetragen wurde. Es zeigt zwar nicht an, welcher Parameter mit welchem ungültigen Wert beschrieben wird, allerdings kann aufgrund dieser Meldung im Applikationsprogramm nachgeforscht werden.
Auf diese Art und Weise kann nachvollzogen werden, welcher Befehl zu einem Fehler geführt hat oder in welcher Phase Fehler aufgetreten sind.

3.5 Oszilloskop

Im JetSym-Oszilloskop lassen sich nicht nur Positionen und Geschwindigkeiten aufzeichnen. MCX stellt viele weitere Parameter zu Verfügung, die zur Analyse eines Ablaufs interessant sein können. Des Weiteren können auch Register/Variablen aus dem Applikationsprogramm aufgezeichnet werden.
Dank dieser Funktionen ist das Oszilloskop nicht nur ein Werkzeug, um Achsen in Betrieb zunehmen, sondern auch zum Debuggen und Visualisieren von Abläufen des Applikationsprogramms.



In diesem Beispiel ist neben den Sollpositionen der Leit- und Folgeachse auch die Sollgeschwindigkeit und Sollbeschleunigung dargestellt.
Zusätzlich kann der Kurvenverlauf auch über die Anzeige der Segmentnummer eines Kurvenscheibenprofils kontrolliert werden.
Über die Aufzeichnung des Koppelstatus wird das Verhalten während der unterschiedlichen Koppelphasen überprüft.
Mit Hilfe des Rampenstatus ist zu sehen, dass die Leitachse erst kurz von dem Synchronpunkt die Zielgeschwindigkeit erreicht hat.

4 Applikationsprogramm - MotionAPI

4.1 MCX-BootStatus:

Im Systemregister der MCX ***R59997904*** wird angezeigt, ob der Bootvorgang erfolgreich ist. Ein Statuswert von "***21***" signalisiert, dass der MCX-Kernel gestartet hat und die Objekte (Achsen, Verbund) anlegen und initialisieren konnte.
Es reicht aus, diesen Wert beim Starten der Applikation zu überprüfen, da sich der Wert während der Laufzeit nicht mehr ändert.
Sollte der Wert ungleich "***21***" sein, so konnte die MCX nicht erfolgreich starten.
Das Applikationsprogramm kann hier eine Benutzermeldung generieren.
Dieser Fehler kann nicht quittiert werden. Es ist ein Eingriff des Benutzers notwendig.
Beispiel: Die vorgegebene Hardware stimmt nicht mit der angeschlossenen Hardware überein

- Eingestellte Hardwarekonfiguration überprüfen, ggf. erneut auf Steuerung übertragen
- Sind die angeschlossenen Drives bestromt?
- Stimmen die Adressen?
- Verbindungskabel überprüfen
- …

4.2 MCX-Fehler im Applikationsprogramm

4.2.1 Sammelfehler

Sammelfehler: Tritt bei einem oder mehreren MC-Objekten ein Fehler auf, so wird dies übergeordnet als Sammelfehler angezeigt. Die Anzeigen sind dabei als gleichwertig zu betrachten.

4.2.2 Systemfehlerregister

Anzeige als Systemfehler in den übergeordneten Fehlerregistern der Steuerung:
Fehlerregister1 R200008.Bit7 zeigt an, dass in weiteren Fehlerregistern ein Fehler ansteht.
Fehlerregister3 R200010.Bit1 zeigt an, dass ein Fehler in der MotionControl vorliegt.
Für die Überwachung eines MotionControl-Fehlers reicht es aus, nur R200010.Bit1 zu überwachen. Für eine kaskadierte Überwachung ist es möglich, nur Fehlerregister1 auf ungleich 0 zu überwachen. Teilweise wird hier schon eine entsprechende Fehlerreaktion ausgelöst und im Folgenden dann mit der Evaluierung fortgefahren. Wird eine Abweichung festgestellt, wird erst bei gesetztem Bit7 in diesem Register nach den weiteren Fehlerregistern nachgesehen. Wird dann festgestellt das Fehlerregister3.Bit1 gesetzt ist, wird mit der Fehlerreaktion bzw. der Evaluierung der Fehler fortgesetzt.

4.2.3 Anzahl der fehlerhaften MCX-Objekte

Die Anzahl fehlerhafter MCX-Objekte lässt sich über den MotionAPI-Befehl *mcMgr. FaultyObject-Count* ermitteln. Wird hier ein Wert > 0 zurückgeliefert, ist mindestens ein MoctionControl-Objekt fehlerbehaftet. Es kann dann mit der Fehlerreaktion bzw. Evaluierung fortgefahren werden.

4.2.4 Überprüfung der einzelnen MotionControl-Objekte

Die MotionAPI bietet für jedes angelegte MotionControl-Objekt ein entsprechendes Diagnose-Objekt.

Motion API 1.0	Motion API 2.0	Fehlerort
MCAxis.	*MCAxis.*	
Diagnostics.IsErrorPending	Diagnostics.IsErrorPending	Achsobject
Diagnostics.Software.IsErrorPending()		MCX
Diagnostics.Drive.IsErrorPending()	Drive.Diagnostics.IsErrorPending	Drive

Motion API 1.0	Motion API 2.0	Fehlerort
Diagnostics.Drive.IsErrorPending(ErrorNr)		Spez. Drive *1)
MCGeo.	*MCGeo.*	
Diagnostics.IsErrorPending	Diagnostics.IsErrorPending	Bahnverbund
MCTechno.	*MCTechno.*	
Diagnostics.IsErrorPending	Diagnostics.IsErrorPending	Technologieverbund

1) MCAxis.Diagnostics.Drive.IsErrorPending(ErrorNr): Es liegt genau dieser Fehler an diesem Antriebsverstärker an. Die ErrorNr entspricht genau der Fehlernummer des JetMove 100/ 200.
   Siehe auch dazu die Benutzerinformation des JetMove 2xx:
   jetmove_2xx_an_jetcontrol_bi_2114_benutzerinformation.pdf

Beispiel:

```
// an error occurred on MyAxis
when MyAxis.Diagnostics.IsErrorPending continue;
// execute error handling and create an alarm
```

```
// check which axis has an error
if MyAxis1.Diagnostics.IsErrorPending then
// execute error handling and create an alarm
end_if;
if MyAxis2.Diagnostics.IsErrorPending then
//….
else
//….
end_if;
```

4.2.5 Fehlerevaluation

4.2.5.1 MotionControl-Fehler

Für die angelegten MotionControl-Objekte werden die Fehler bis zum Löschen in einem Puffer abgelegt.
Die MotionAPI bietet die Funktionen, um diesen Puffer auszulesen.

MotionApi 1.0
Das Rücklesen der anstehenden Fehler erfolgt bei allen Objekttypen gleichermaßen.

MotionApi 2.0
Das Rücklesen der anstehenden Fehler erfolgt bei den Achsobjekten direkt im Diagnoseobjekt, während das Rücklesen von Fehlern der Verbundobjekte analog zur MotionApi1.0 erfolgt.

Motion API 1.0	Motion API 2.0	Beschreibung
*MCAxis.**Diagnostics.Software***	*MCAxis.**Diagnostics.***	
ErrorCount	ErrorCount	Anzahl der Fehlereinträge im Puffer
GetErrorCode(BufferIndex)	GetErrorCode(BufferIndex)	Fehlereintrag
MCGeo.Diagnostics.Software		
ErrorCount		Anzahl der Fehlereinträge im Puffer
GetErrorCode(BufferIndex)		
MCTechno.Diagnostics.Software		
ErrorCount		Anzahl der Fehlereinträge im Puffer
GetErrorCode(BufferIndex)		

ErrorCount: Anzahl der Fehlereinträge im Puffer
GetErrorCode(BufferIndex): Rückgabe der anstehenden Fehler. Über den BufferIndex können alle Einträge der Reihe nach ausgelesen werden. Bleibt der Funktionsparameter leer, wird immer der erste Eintrag im Puffer zurückgegeben.

Beispiel:
In diesem Beispiel für die MotionAPI2.0 werden bei Auftreten eines MCX-Fehlers mit Hilfe des MC-Manager-Objekts "mcMgr" systematisch aus allen fehlerhaften Objekten die anstehenden Fehler ausgelesen.

```
#include ".\GeneralFunctions.stxp"

type
   SystemExtErrorRegsiter2:   bits(
      MCX = 1
   );
end_type;

function superviseAnyError()
var
   nListIndex:                              int32;
   nErrorIndex:                             int32;
   pDiagnostics:      pointer to     MCDiagnosticsSoftware;
end var;

   if mcMgr.FaultyObjectCount > 0 then
      for nListIndex := 0 to mcMgr.FaultyObjectCount by 1 do
         pDiagnostics := 0;
         case mcMgr.FaultyObject[nListIndex].Info.ClassID of
```

```
         MCClassIDs.Axis:
            pDiagnostics :=  mcMgr.AxisList[nListIndex].Diagnostics;
         break;
         MCClassIDs.Techno:
            pDiagnostics :=  mcMgr.TechnoList[nListIndex].Diagnostics.Software;
         break;
         MCClassIDs.Geo:
            pDiagnostics :=  mcMgr.GeoList[nListIndex].Diagnostics.Software;
         break;
         end case;

         if pDiagnostics != 0 then
            for nErrorIndex := 0 to mcMgr.AxisList[nListIndex].Diagnostics.ErrorCount by 1 do
               // Error Evaluation
               trace(mcMgr.FaultyObject[nListIndex].Info.Name + ': '+ IntTo-
Str(pDiagnostics.GetErrorCode(nErrorIndex)) + ';$n');
            end_for;
         end_if;
      end_for;
   end_if;
end function;

var
   g_SystemErrorExt2:         SystemExtErrorRegsiter2 at %vl 200010;
   g bClearError:                    bool;
end_var;

// Task supervises if any MCX-error occured
task supervision autorun

   when g_SystemErrorExt2.MCX continue;
      superviseAnyError();
      g bClearError := false;
      when g_bClearError continue;
      resetTechnoAll();
      resetAxisAll();
      g SystemErrorExt2.MCX := false;

end_task;
```

4.2.5.2 Fehler der Antriebsverstärker

MotionApi 1.0 mit JetMove2xx

Die anliegenden Fehler werden entweder über die direkte Abfrage des Fehlers ausgewertet oder über das Auslesen der überlagerten Fehlerregister, bei denen die Fehler bitcodiert vorliegen.
MCAxis.Diagnostics.Drive.IsErrorPending(ErrorNr): Falls wahr, liegt dieser Fehler an.
MCAxis.Diagnostic.Drive. ErrorFlags1: Bitcodierte Fehler 00 bis 31 des JetMove2xx.
MCAxis.Diagnostic.Drive. ErrorFlags2: Bitcodierte Fehler 32 bis 43 des JetMove2xx.
Die bit-weise Auswertung kann z.B. in einer Zählschleife erfolgen. Die Bedeutung der einzelnen Fehlerbits sind in der Benutzerinformation des JetMove 2xx beschrieben: *jetmove_2xx_an_jetcontrol_bi_2114_benutzerinformation.pdf*

MotionApi 2.0 mit EtherCAT-Antriebsverstärkern

MCAxis.Drive.Diagnostics.IsErrorPending: Falls wahr, liegt ein Fehler an.

MCAxis.Drive.Diagnostics.ActualError: Rückgabe des anliegenden Fehlers. Die Fehlernummern sind dem jeweiligen Benutzerhandbuch zu entnehmen.

Motion API 1.0	Motion API 2.0	Beschreibung
*MCAxis.**Diagnostics.Drive***	*MCAxis.**Drive.Diagnostics***.	
IsErrorPending(ErrorNr)	IsErrorPending	Anzahl der Fehlereinträge im Puffer
ErrorFlags1	*ActualError*	Fehlereintrag
ErrorFlags2		Fehlereintrag

4.2.6 Warnungen und Meldungen: Erkennung und Evaluation

4.2.6.1 MotionControl-Meldungen

Die MotionControl erzeugt neben Fehlermeldungen auch allgemeine Meldungen. Dies sind Informationen, die für die Überprüfung von Abläufen interessant sein können.
Soll z.B. eine Kurvenscheibe zu einer bestimmten Leitachsposition aktiviert werden und wird noch bevor dieser Punkt erreicht ist eine weitere Kurvenscheibe aktiviert, so wird die Meldung erzeugt, dass eine anstehende Kurvenscheibenaktivierung abgebrochen wird.

Motion API 1.0	Motion API 2.0	Beschreibung
*MCAxis.**Diagnostics.***	*MCAxis.**Diagnostics.***	
IsMessagePending		Es liegt eine Meldung an
*MCAxis.**Diagnostics.Software***	*MCAxis.**Diagnostics.***	
IsMessagePending		Es liegt eine Meldung an
MessageCount		Anzahl der Meldungen im Buffer
GetMessageCode(BufferIndex)		Meldungseintrag
MCTechno.**Diagnostics**.		
MCGeo.**Diagnostics**.		
IsMessagePending		Es liegt eine Meldung an
.**Software.**		
IsMessagePending		Es liegt eine Meldung an
MessageCount		Anzahl der Meldungen im Puffer
GetMessageCode(BufferIndex)		Meldungseintrag

<MCObject>.Diagnostics.IsMessagePending und *<MCObject>.Diagnostics.Software.IsMessagePending* sind identische Properties des Diagnostics-Objekts.

4.2.6.2 Antriebsverstärker-Meldungen

Motion API 1.0	Motion API 2.0	Beschreibung
	*MCAxis.Drive.**Diagnostics.***	
-	IsMessagePending	Es liegt eine Meldung an

4.2.6.3 Antriebsverstärker-Warnungen

Im MCX-System werden Warnungen nur vom Antriebsverstärker erzeugt.
Bei MotionAPI 1.0 können neben der Existenz von Warnungen auch der Inhalt des Warnungsregisters des JM2xx ausgelesen werden. Die Bedeutung der einzelnen Warnungsbits sind in der Benutzerinformation des JetMove 2xx beschrieben:
jetmove_2xx_an_jetcontrol_bi_2114_benutzerinformation.pdf

Bei MotionAPI 2.0 kann nur festgestellt werden, dass Warnungen anliegen. Ein evtl. Auslesen der Warnung erfolgt über ein CANopen-Objekt. Diese Objekte sind im Handbuch des jeweiligen Antriebsverstärkers beschrieben.

Motion API 1.0	**Motion API 2.0**	**Beschreibung**
*MCAxis.**Diagnostics.***		
IsWarningPending		Es liegt eine Warnung an
*MCAxis.**Diagnostics.Drive***	*MCAxis.**Drive.Diagnostics***	
IsWarningPending	IsWarningPending	Es liegt eine Warnung an
WarningFlags		JM2xx-Warnungsregister

Jetter automation

Jetter AG
Gräterstraße 2
71642 Ludwigsburg
www.jetter.de

E-Mail info@jetter.de
Telefon +49 7141 2550-0

We automate your success.