Betriebsanleitung
Edition 03/2016
Release F

Automation des AKD® Drives Über Modbus mit einer Siemens S7-1200/1500 PLC

Programmiert im TIA Portal V13

Largo Brughetti 1/B2 • 20813 Bovisio Masciago (MB) • Italia • Tel: +39 0362 594260 • www.kollmorgen.com • P.IVA 02965490960 • Cod. Fisc. 02491750234

Bisher erschienene Ausgaben:

Ausgabe	*Bemerkungen*
A 11/2014	Erste Version
B 12/2014	Überarbeitete Version, Modbusdatenzugriff direkt im Datenblock anstelle des allgemeinen Merker Bereichs
C 02/2015	Überarbeitete Version, AKD_MBComm Anpassungen Software V0.8
D 02/2015	Skalierung korrigiert Neue Funktion: Lesen eines Fahrauftrages
E 09/2015	Skalierung Stromwert korrigiert Konvertierung nach TIA V13 SP1
F 03/2016	Überarbeitung der Kommunikationsüberwachung Hinzufügen der S7-1200 mit Firmware 4.1

Bemerkungen
Zur Lesbarkeit des Programmcode haben die Variablen einen Vorgestellten Kleinbuchstaben entsprechend der folgenden Tabelle.

Prefix	Grösse	Bedeutung
i	Unbekannt	Eingang
o	Unbekannt	Ausgang
b	1 Bit	Bool
by	8 Bit	Byte
w	16 Bit	Word
dw	32 Bit	Double Word
	Unbekannt	Variablenstruktur

Inhaltsverzeichnis

Einleitung

Das hier behandelte System beinhaltet ein AKD-P Servoverstärker und ein S7-1200 PLC. Alle Automatisierungsfunktionen werden von der PLC aus angesteuert, der Funktionsumfang ist wie folgt:

- Software Ein-/Ausschalten
- Not halt
- Rücksetzen von Alarmmeldungen
- Tippen
- Referenzfahrt der Achse (mit AKD Referenzfahrt oder direkt auf Position)
- Starten eines definierten Fahrauftrags
- Aufsetzen von Fahraufträgen
- Lesen von Fahraufträgen
- Wechseln des Operationsmodus
- Direktes Schreiben oder Lesen von Parametern

Jede Funktion wird mit spezifischen Bits und Variablen angesteuert und mit den nötigen Werten eingestellt, aktuelle Informationen vom AKD werden zurück gelesen.

Dieses Manual soll dazu dienen die nötigen Befehle zu verstehen und mit wenigen Schritten Bewegungen zu realisieren.

	**Verwenden Sie das TIA-Projekt „AKD_MBComm_Modules.zap13, AKD_MBCommS1500_Modules.zap13“ niemals unverändert in einer Anwendung. Die Anwendung ist ein Beispiel wie der AKD in ein TIA-Projekt integriert werden kann. Dieses Projektbeispiel muss immer an die vorhandene Anwendung angepasst werden.**
	**KOLLMORGEN srl haftet nicht für Schäden und schliesst alle Ansprüche aus, die sich durch den Einsatz des TIA-Projekts „AKD_MBComm_Modules.zap13, AKD_MBCommS1500_Modules.zap13“ oder Programmteilen daraus ergeben könnten.**

Konfiguration AKD

Die Verbindung zwischen dem AKD-P (Stecker X11) und der PLC (Stecker X1P1) wird mit Modbus TCP realisiert. Grundeinstellungen des Reglers sollten über die Workbench erfolgen. Die Modbusverbindung nutzt den Port 502, Workbench Telnet mit Port 23, beiden Kommunikationen können parallel aufgebaut sein.

Mit der Funktion MB_Client werden die benötigten Daten zwischen den Geräten ausgetauscht. Damit das Lesen auf dem AKD-P schneller funktioniert, werden die benötigten Daten in ein fortlaufendes Register geschrieben. Dafür wird das dynamische Modbus-Mapping verwendet.

Im Terminal kann folgendes Makro „MBCommSet.macro“ ausgeführt werden. Damit werden die Grundeinstellungen der Kommunikation gemacht. Die nötigen Befehle können auch einzeln von Hand eingegeben werden.

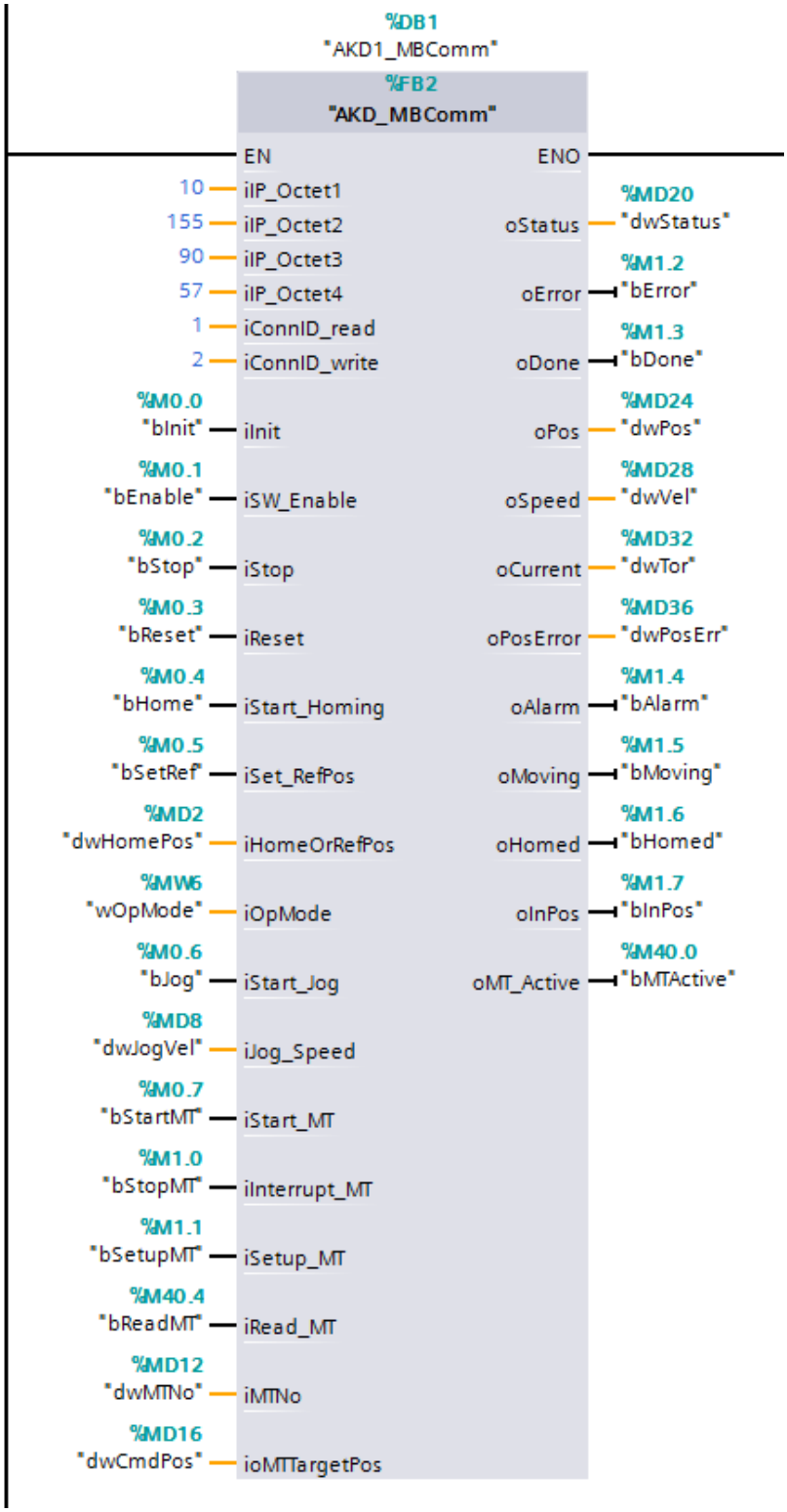
Liste der Parameter welche für die Kommunikation eingestellt werden müssen, dies kann mit dem „MBCommSet.macro“ gemacht werden.

Terminal Kommando	**Beschreibung**
MODBUS.DYNMAP 1	Einschalten des dynamischen Mapping
MODBUS.CLRDYNMAP	Löschen des dynamischen Mapping-Bereichs
MODBUS.ADDR8192 945	Parameter 945 auf die Modbusadresse 8192 schreiben: MODBUS.DRVSTAT
MODBUS.ADDR8193 268	DRV.MOTIONSTAT High Byte
MODBUS.ADDR8194 269	DRV.MOTIONSTAT Low Byte
MODBUS.ADDR8195 2072	PL.FB_32 High Byte
MODBUS.ADDR8196 2073	PL.FB_32 Low Byte
MODBUS.ADDR8197 2066	PL.ERR_32 High Byte
MODBUS.ADDR8198 2067	PL.ERR_32 Low Byte
MODBUS.ADDR8199 954	MODBUS.FAULT1 High Byte
MODBUS.ADDR8200 955	MODBUS.FAULT1 Low Byte
MODBUS.ADDR8201 856	VL.FB High Byte
MODBUS.ADDR8202 857	VL.FB Low Byte
MODBUS.ADDR8203 432	IL.FB High Byte
MODBUS.ADDR8204 433	IL.FB Low Byte
MODBUS.ADDR8300 2060	MT.P_32 High Byte
MODBUS.ADDR8301 2061	MT.P_32 Low Byte
MODBUS.ADDR8302 566	MT.V High Byte
MODBUS.ADDR8303 567	MT.V Low Byte
MODBUS.ADDR8304 2056	MT.ACC_32 High Byte
MODBUS.ADDR8305 2057	MT.ACC_32 Low Byte
MODBUS.ADDR8306 2058	MT.DEC_32 High Byte
MODBUS.ADDR8307 2059	MT.DEC_32 Low Byte
MODBUS.ADDR8308 532	MT.CNTL High Byte
MODBUS.ADDR8309 533	MT.CNTL Low Byte
MODBUS.ADDR8310 546	MT.NEXT High Byte
MODBUS.ADDR8311 547	MT.NEXT Low Byte
MODBUS.DYNMAP 0	Ausschalten des dynamischen Mapping
MODBUS.PSCALE 20	Positionsauflösung für PL.FB_32
SM.MODE 0	Servicefunktionsmodus 0
SM.T1 0	Servicefunktionszeit T1 0ms
SM.T2 0	Servicefunktionszeit T2 0ms
DIN.LCMD1 DRV.STOP;DRV.CMDDELAY 500;DRV.DIS	Befehlsspeicher 1: Stopp, Verzögerung von 500ms, Ausschalten.
DIN3.MODE 9	Digitaler Eingang 3 auf Befehlsausführung
DIN3.PARAM 1	Befehlsspeicher 1
DRV.NVSAVE	Speichern der Änderungen im Nichtflüchtigen Bereich des AKD.

AKD - PLC Verbindung

Konfiguration PLC

Der Funktionsblock **AKD_MBComm** dient als Interface für den Anwender in der PLC. In diesem Baustein wird das Lesen wie auch das Schreiben mit dem MB_Client Funktionsblock ausgeführt. Dazu wird der MBR Datensatz zum Lesen und MBW Datensatz zum Schreiben genützt.



Name der Variable	Datentyp	Richtung	Beschreibung
EN	Bool	In	Aufruf des DB
iIP_OctetX	USInt	In	IP-Adresse des anzusprechenden AKD
iConnID_read	UInt	In	Verbindungsnummer für den Lesebefehl **(Eineindeutig)**
iConnID_write	UInt	In	Verbindungsnummer für den Schreibbefehl **(Eineindeutig)**
iInit	Bool	In	Abbrechen der Modbus Verbindungen
iSW_Enable	Bool	In	Schaltet die Softwarefreigabe Ein/Aus
iStop	Bool	In	Stoppt alle Bewegungen
iReset	Bool	In	Setzt die Fehler auf dem AKD zurück
iStart_Homing	Bool	In	Startet die Referenzierung via AKD Referenzfahrt
iSet_RefPos	Bool	In	Setzt die aktuelle Position als Referenz
iHomeOrRefPos	DWord	In	End Positionswert für Set_RefPos oder Start_Homing
iOpMode	Int	In	Operationsmodus des AKD
iStart_Jog	Bool	In	Startet eine Tippbewegung
iJog_Speed	DWord	In	Tippgeschwindigkeit
iStart_MT	Bool	In	Startet den im AKD abgespeicherten Fahrauftrag
iInterrupt_MT	Bool	In	Unterbricht einen aktiven Fahrauftrag
iSetup_Mt	Bool	In	Einrichten eines Fahrauftrags im AKD
iRead_MT	Bool	In	Lesen eines kompletten Fahrauftrages im AKD
iMTNo	DWord	In	Nummer des Fahrauftrages für Setup_MT oder Start_MT
ioMTTargetPos	DWord	In/Out	Zielposition für einen Fahrauftrag Setup_MT
ENO	Bool	Out	DB Aufruf erfolgreich
oStatus	DWord	Out	Aktueller Status des AKD
oError	Bool	Out	Fehler in der Verbindung oder dem Funktionsblock
oDone	Bool	Out	Modbusschreibbefehl wurde erfolgreich ausgeführt
oPos	DWord	Out	Aktuelle Position
oSpeed	DWord	Out	Aktuelle Geschwindigkeit
oCurrent	DWord	Out	Aktueller Strom
oPosError	DWord	Out	Aktueller Positionsfehler
oError	Bool	Out	Anzeigen eines AKD Fehlers
oMoving	Bool	Out	Achse Bewegt sich
oHomed	Bool	Out	Achse ist Referenziert
oInPos	Bool	Out	Achse ist innerhalb des Positionsfensters des Fahrauftrags
oMT_Active	Bool	Out	Fahrauftrag aktiv
wMBR_Data[0..15]	Word	Static	Datenfeld für die zu lesenden Modbus Daten
dwMBW_Data[0..5]	DWord	In/Out	Datenfeld für die zuschreibenden Modbus Daten
bMBR_Req	Bool	Static	Startet eine Leseanfrage
dwMBR_Addr	DWorrd	Static	Gibt die Leseadresse
bMBR_Busy	Bool	Static	Zeigt eine aktives Lesen an
bMBR_Done	Bool	Static	Zeigt eine Erfolgreiches Lesen an
bMBR_Error	Bool	Static	Zeigt ein Verbindungsfehler an
wMBR_Status	Word	Static	Zeigt den Status der Leseverbindung an
bMBW_Req	Bool	Static	Startet eine Schreibanfrage
dwMBW_Addr	DWord	Static	Gibt die Schreibadresse
bMBW_Busy	Bool	Static	Zeigt eine aktives Schreiben an
bMBW_Done	Bool	Static	Zeigt eine Erfolgreiches Schreiben an
bMBW_Error	Bool	Static	Zeigt ein Verbindungsfehler an
wMBW_Status	Word	Static	Zeigt den Status der Schreibverbindung an
dwMBW_ChechForWrite	DWord	Static	Sucht den nächsten Befehl welcher geschrieben werden muss
wMBW_DataLenght	Word	Static	Länge des zu beschreibenden Datenregisters
bDriveNeedsDisable	Bool	Static	Zeigt das Software ein des AKD an
bJogIsActive	Bool	Static	Zeigt den Start einer Tippbewegung an
bHomingIsActive	Bool	Static	Zeigt den Start eines Referenzierungsvorgangs an
bSetRefPosIsActive	Bool	Static	Zeigt den Start einer direkten Referenzierung an.
bStartMTIsActive	Bool	Static	Zeigt den Start eines Fahrauftrags an
bOldResetState	Bool	Static	Vergleich mit dem Eingang Reset
wOldOpMode	Word	Static	Vergleich mit dem Eingang OpMode

dwOldJogSpeed	DWord	Static	Vergleich mit dem Eingang JogSpeed
dwOldHomeOrRefPos	DWord	Static	Vergleich mit dem Eingang RefPos
dwSetupMT	DWord	Staic	Lokaler Status beim Schreiben eines neuen Fahrauftrages
bMBWInProgress	Bool	Static	Schreibbefehl in Ausführung
MBR	MB_Client	Static	Modbuskommunikationsblock für das Lesen
MBW	MB_Client	Static	Modbuskommunikationsblock für das Schreiben
bSetup_MTDone	Bool	Static	Ein Fahrsatz wurde fertig in den AKD geschrieben
bMTInterrupted	Bool	Static	Ein aktiver Fahrauftrag wurde unterbrochen und wartet auf weder Aufnahme
bResetInPos	Bool	Static	Setzt beim Start eines Fahrauftrages das InPos für Minimum einen Zyklus auf Null
bMBR_Init	Bool	Static	Stoppen und zurücksetzten der Leseverbindung
bMBW_Init	Bool	Static	Stoppen und zurücksetzten der Schreibbindung
TimeStamp	DTL	Static	Systemzeit
dwMillSec	DWord	Static	Millisekunden der Systemzeit
dwMBRTimeStamp	DWord	Static	Zeitstempel beim Aufruf der Leseverbindung
wLocalWDTime	Word	Static	Überwachungszeitlimit für die Leseverbindung
bWDError	Bool	Static	Zeitüberschreitung der Leseverbindung
dwTempRunTime	DWord	Static	Laufzeit der aktuellen Leseverbindung
bReadMTDone	Bool	Static	Lesen eines Fahrauftrages abgeschlossen
dwReadMT	DWord	Static	Lokaler Status beim Lesen eines Fahrauftrages
bWDActive	Bool	Static	Lokaler Status der Kommunikationstimeoutüberwachung

Das wMBR_Data Wort Feld wird für die über Modbus gelesenen Daten verwendet. Für die über Modbus zu schreibenden Daten wird das dwMBW_Data Doppel Wort Feld genutzt. Das dwMBW_Data[0] wird immer genutzt und vom Funktionsblock beschrieben. dwMBW_Data[1] – dwMBW_Data[5] wird nur genutzt um Fahraufträge aufzusetzen. Dazu müssen die jeweiligen Parameter direkt im entsprechenden Feld eingetragen werden. Eine Auflistung der Funktion der einzelnen Variablen gibt die nachfolgende Tabelle.

Status Daten

Variable	***Richtung***	***Grösse***	***Beschreibung***	***Bit***	***Beschreibung***	***Bit***	***Beschreibung***
wMBR_Data[0]	IN	Word	MODBUS.DRVSTAT	3	Negatives HW Limit	0	Regler aktiv
				4	Positives SW Limit	1	STO Status
				5	Negatives SW Limit	2	Positives HW Limit
wMBR_Data[1]	IN	Word	DRV.MOTIONSTAT	0	Not Stopp Fehler im Regler		
				1	Servicefunktion aktiv		
				2	Ungültiger Fahrauftrag		
				3	Fahrauftrag in Position		
wMBR_Data[2]	IN	Word	DRV.MOTIONSTAT	4	Referenzfahrtfehler	0	Fahrauftrag aktiv
				5	Getriebemodus synch	1	Referenz gefunden
				6	Getriebemodus aktiv	2	Referenzfahrt fertig
				7	Not Stopp aktiv	3	Referenzfahrt aktiv
wMBR_Data[3]	IN	Word	PL.FB_32 high byte				
wMBR_Data[4]	IN	Word	PL.FB_32 low byte				
wMBR_Data[5]	IN	Word	PL.ERR_32 high byte				
wMBR_Data[6]	IN	Word	PL.ERR_32 low byte				
wMBR_Data[7]	IN	Word	MODBUS.FAULT1 high byte				

wMBR_Data[8]	IN	Word	MODBUS.FAULT1 low byte
wMBR_Data[9]	IN	Word	VL.FB high byte
wMBR_Data[10]	IN	Word	VL.FB low byte
wMBR_Data[11]	IN	Word	IL.FB high byte
wMBR_Data[12]	IN	Word	IL.FB low byte

Kontroll Daten

Variable	***Richtung***	***Grösse***	***Beschreibung***	***Zugriff***
dwMBW_Data[0]	OUT	DWord	Daten für den nächsten Schreibbefehl	Intern
dwMBW_Data[1]	OUT	DWord	MT.V, Fahrauftragsgeschwindigkeit	Extern
dwMBW_Data[2]	OUT	DWord	MT.ACC, Fahrauftragsbeschleunigung	Extern
dwMBW_Data[3]	OUT	DWord	MT.DEC, Fahrauftragsbremsbeschleunigung	Extern
dwMBW_Data[4]	OUT	DWord	MT.CNTL, Fahrauftragskontorollwort	Extern
dwMBW_Data[5]	OUT	DWord	MT.NEXT, Nummer des folge Fahrauftrages	Extern

Zu beachten

Die ConnID_read und ConnID_write dürfen nicht den gleichen Wert besitzen. Werden mehrere Achsen angesprochen so müssen diese Nummern immer Eineindeutig sein!

Die Eingangsfunktionen werden, sollten mehrere Signale gelichzeitig anstehen, nach Prioritäten abgehandelt. Es gilt eins als höchste Priorität.

Wenn die **Kommunikation abbricht** oder die PLC Ausgeschaltet wird so wird dies auf dem AKD **nicht erkannt**. Mit einem digitalen Eingang kann eine Überwachung eingerichtet werden, dazu einen digitalen Eingang auf 9-Command Buffer einstellen. Mit der negative Flanke können so verschiedene Funktionen wie zum Beispiel ein DRV.STOP / DRV.DIS mit oder ohne Verzögerung ausgeführt werden. Im MBCommSet.macro ist dazu ein Beispiel angefügt welches den Eingang drei mit dem Befehlsspeicher eins benützt.
Auf der PLC Seite wird die Leseverbindung mit einem Timeout von 500ms Überwacht. Ein Timeout wird im *oError* Bit angezeigt. Um eine Verbindung neu aufzubauen muss das *iInit* für eine Sekunde angesteuert werden.

Mit dem *iRead_MT* können die Werte eines Fahrauftrages aus dem AKD zurück gelesen werden. Dazu wird die MBR Funktion genützt. Das heisst die aktuellen Daten werden für einen Lesezyklus nicht aktualisiert. Die gelesenen Werte werden in *iMTTaregtPos* sowie in das dwMBW_Data[1..5] Feld geschrieben, gemäss *iMTTargetPos*.

Ist das TIA-Portal online auf der Steuerung so kann es zu inkonstanten Verzögerungen zwischen zwei Lesebefehlen kommen (9ms - 80ms). Im Testaufbau wurden ohne TIA-Portal 9ms als Laufzeit zwischen zwei Lesebefehlen gemessen. Bei der Verbindung mit zwei AKD ist diese Zeit bereits auf 18ms angestiegen, bei drei AKD auf 26ms. Diese Zeitmessungen wurden alleine, sprich ohne zusätzlichen Code ausgeführt und sind daher als Best möglich anzusehen.

Der AKD-P00306-NAEC wurde mit der Firmware M_01-14-02-000 betrieben.

Dieses Project wurde mit einer Siemens CPU1214 DC/DC/DC getestet und kann bei anderen Konfigurationen Änderungen benötigen welche hier nicht aufgeführt sind.

Detail Beschreibung der Eingaben

iIP_OctetX
Hier wird die IP-Adresse des zu verbindenden AKD's angegeben.

iConnID_read/write
Gibt die Verbindungsnummer der Modbusverbindung an. Jede Modbusverbindung in der PLC muss eine eineindeutige Verbindungsnummer haben. Die *iConnID_read* und die *iConnID_write* müssen unterschiedlich sein!

iInit
Wenn die Initialisierung aktive ist wird die Modbus Kommunikation abgebrochen. Bei Fehler mit der Kommunikation (*oError*) kann damit ein Reset erfolgen. Das *iInit* muss dazu aktive sein, bis der *oError* Null wird, Minimum aber eine Zeit von ca. 300ms. Fällt das *oError* direkt mit dem Setzten des *iInit* ab, so kann es sein, dass der MB_Client noch nicht sauber gestoppt ist und sich bei Aufbau der Verbindung aufhängt.

iSW_ENABLE
0 = Ausschalten des AKD, Priorität 1.
1= Einschalten des AKD, Priorität 5.

iReset
Einmaliges Rücksetzten von Fehlern auf dem AKD. Soll der Befehl nochmals ausgeführt werden muss wieder eine Positive Flanke auftreten, Priorität 2.

iStop
Anhalten der Achse ohne die Softwarefreigabe auszuschalten, Priorität 3.

iStart_Homing
Startet den Referenzierungsvorgang nach den Einstellungen in der Workbench. Soll der Befehl nochmals ausgeführt werden muss wieder eine Positive Flanke auftreten, Priorität 7.

iSet_RefPos
Setzt die aktuelle Position als Referenz. Der AKD ist damit referenziert. Für diesen Befehl muss die Leistung nicht aktiv sein. Soll der Befehl nochmals ausgeführt werden muss wieder eine Positive Flanke auftreten, Priorität 9.

iHomeOrRefPos
Position welche die Achse nach einem Referenzierungsvorgang hat. Dieser Wert wird von *iStart_Homing* und *iSetRefPos* genutzt und als Nullposition gesetzt. Die Skalierung ist im Abschnitt „Skalierung der Werte“ beschrieben, Priorität 6.

iOpMode
Operationsmodus des AKD umschalten. Priorität 4.
0 = Torque Mode, Stromregelung
1 = Veloctiy Mode, Geschwindigkeitsregelung
2 = Position Mode, Positionsregelung

iStart_Jog
Bewegt den Motor mit der fixen Geschwindigkeit *iJog_Speed*, Priorität 12.
Wenn *iOpMode* = 2 ist muss für einen Geschwindigkeitswechsel die Achse stoppen, das Tippen wird mit der Servicefunktion realisiert. Wenn *iOpMode* = 1 ist ein Geschwindigkeitswechsel direkt an einer Bewegenden Achse möglich, das Tippen nutzt den VL.CMDU Befehl.

iJog_Speed
Geschwindigkeit mit der die Tippbewegung ausgeführt wird. Die Skalierung ist im Abschnitt „Skalierung der Werte“ beschrieben, Priorität 11.

iStart_MT
Starten des Fahrauftrag welcher im AKD auf der Position *iMTNo* abgelegt ist. Soll der Befehl nochmals ausgeführt werden muss wieder eine Positive Flanke auftreten, Priorität 14.

iInterrupt_MT
Stoppt einen aktiven Fahrauftrag. Beim Wechsel auf Null wird der Fahrauftrag fortgeführt. Wird ein Bewegung mit einem *iStop* gehalten, die Softwarefreigabe deaktiviert oder ein *iReset* ausgeführt, so kann kein Fortführung mehr stattfinden, siehe Grafik Ablauf Fahrauftragsunterbruch, Priorität 15.

iSetup_MT
Schreibt einen Fahrauftrag mit den Werten (siehe *iMTTargetPos*) in den AKD mit der Fahrsatznummer *iMTNo*. Die *iMTNo* darf nicht Null betragen! Dieser Vorgang kann bis zu 300ms dauern. Es ist jedoch möglich einen Fahrauftrag der aktuell ausgeführt wird zu ändern und diese neue Bewegung auszuführen, wenn die alte Bewegung abgeschlossen ist. Soll der Befehl nochmals ausgeführt werden muss wieder eine Positive Flanke auftreten, Priorität 13.

iRead_MT
Liest die Daten des entsprechenden Fahrauftrages *iMTNo* und schreibt diese gemäss *iMTTargetPos* in das dwMBW_Data Feld, Priorität 17.

iMTNo
Auswahl des Fahrsatzes welcher gestartet, bearbeitet oder gelesen werden soll. Um ein *iSetup_MT* ausführen zu können muss die *iMTNo* ungleich Null sein.

iMTTargetPos
Setzt die Zielposition für den aufzusetzenden Fahrauftrag.Ein Fahrauftrag besteht aus der Zielposition, Geschwindigkeit, Beschleunigung, Bremsbeschleunigung, Bewegungsart und dem folge Auftrag. Alle Parameter ausser der Zielposition müssen direkt in den jeweiligen Pointer geschrieben werden. Die Skalierung ist im Abschnitt „Skalierung der Werte“ beschrieben.
dwMBW_Data[1] = Geschwindigkeit, MT.V
dwMBW_Data[2] = Beschleunigung, MT.ACC
dwMBW_Data[3] = Bremsbeschleunigung, MT.DEC
dwMBW_Data[4] = Bewegungsart, MT.CNTL 0 = absolut, 1= relativ
dwMBW_Data[5] = Folge Fahrauftragsnummer, (wird nur beachtet wenn MT.CNTL Bit 4 = WAHR)

Für genauere Beschreibungen kann der jeweilige Befehl im AKD Workbench Handbuch nach geschlagen werden.

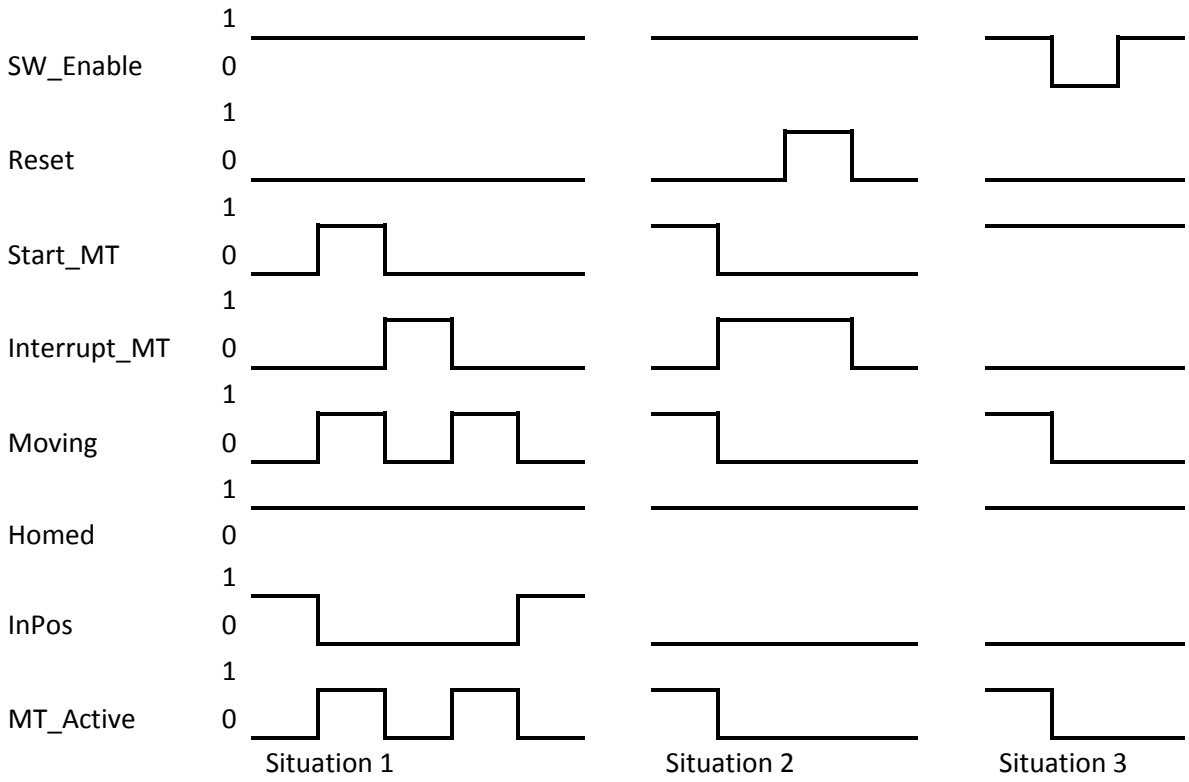
Skalierung der Werte

Auflösung der Position
Die Position entspricht der Modbusskalierung in Umdrehung.
Position im AKD x MODBUS.PSCALE / (MODBUS.PIN / MODBUS.POUT) = Wert in der PLC.
Die Umrechnung mit UNIT.PROTARY = 3 sieht wie folgt aus.
Position im AKD x MODBUS.PSCALE / (MODBUS.PIN / MODBUS.POUT) ^2 = Wert in der PLC.

Auflösung der Geschwindigkeit
Die Geschwindigkeit entspricht der Modbusskalirung in Umdrehungen pro Sekunde.
Geschwindigkeit im AKD x MODBUS.PSCALE / 60 = Wert in der PLC.

Auflösung der Beschleunigung
Die Beschleunigung entspricht der Modbusskalirung in Umdrehungen pro Minute pro Sekunde.
Beschleunigung im AKD x MODBUS.PSCALE / 60 = Wert in der PLC.

Ablauf Fahrauftragsunterbruch



1: Ein *iInterrupt_MT* unterbricht ein Fahrauftrag, dieser wird weitergeführt wenn der *iInterrupt_MT* wieder wegfällt. Sind eine *iInterrupt_MT* fallende Flanke und eine *iStart_MT* steigende Flanke zeitgleich so wird der neue Fahrauftrag ausgeführt und nicht der alte fortgesetzt.

2: Ein *iInterrupt_MT* unterbricht ein Fahrauftrag, dieser wird nicht mehr weitergeführt wenn vor dem wegfallen des *iInterrupt_MT* ein *iReset* oder ein *iStop* ausgeführt worden ist.

3: Ein *iStop* oder ein *iSW_Enable* wird ausgelöst und unterbricht den Fahrauftrag, dieser wird in keinem Fall weitergeführt. Dasselbe passiert wenn der AKD einen Fehler hat oder der STO ausgeschaltete wird.

Detail Beschreibung der Ausgabe

oStatus
Zeigt den aktuellen Status des AKD an.

Bit 0	Regler Aktive
Bit 1	STO Status
Bit 2	Positives Hardwarelimit erreicht
Bit 3	Negatives Hardwarelimit erreicht
Bit 4	Positives Softwarelimit erreicht
Bit 5	Negatives Softwarelimit erreicht
Bit 6	Ungültiger Fahrauftrag aktiviert
Bit 8 - 19	Fehlercode AKD (identisch mit dem Display)

oError
Anzeige von Fehler in der Verbindung oder dem Funktionsblock. Es wird die Lese- wie auch die Schreibverbindung auf Fehler überwacht. Die Leseverbindung hat zusätzlich noch eine aktiv Zeitüberwachung, diese Kontrolliert dass eine einzelne Leseanfrage nicht länger als 500ms dauert. Das Error Bit steht solange an bis der Fehler nicht mehr vorhanden ist oder ein *iInit* die Verbindung unterbrochen hat.

oDone
Zeigt das erfolgreiche ausführen der Schreibverbindung an. Das *oDone* Bit ist während der Befehlsübertragung Null und wird Eins bei erfolgter Übertragung des Befehls.

oPosition
Aktuelle Position der Achse als 32 Bit DWord, PL.FB_32. Die Skalierung ist im Abschnitt „Skalierung der Werte“ beschrieben.

oSpeed
Aktuelle Geschwindigkeit der Achse, als 32 Bit DWord, VL.FB. Die Skalierung ist im Abschnitt „Skalierung der Werte“ beschrieben.

oCurrent
Anzeige des aktuellen Stroms, als 32 Bit DWord, IL.FB in mA. Aufgelöst in Spitzenstrom des AKD / 20130 [Arms].

oPositionError
Aktueller Positionsfehler der Achse als 32 Bit DWord, PL.ERR_32. Die Skalierung ist im Abschnitt „Skalierung der Werte“ beschrieben.

oAlarm
Anzeige eines Fehlers auf der Achse. Es werden nur die Fehler des AKD angezeigt.

oMoving
Anzeige eines aktiven Fahrsatzes oder einer aktiven Tippbewegung.

oHomed
Anzeigen wenn die Referenzierung erfolgreich abgeschlossen wurde. Dies muss WAHR sein um eine Fahrsatzausführen zu können.

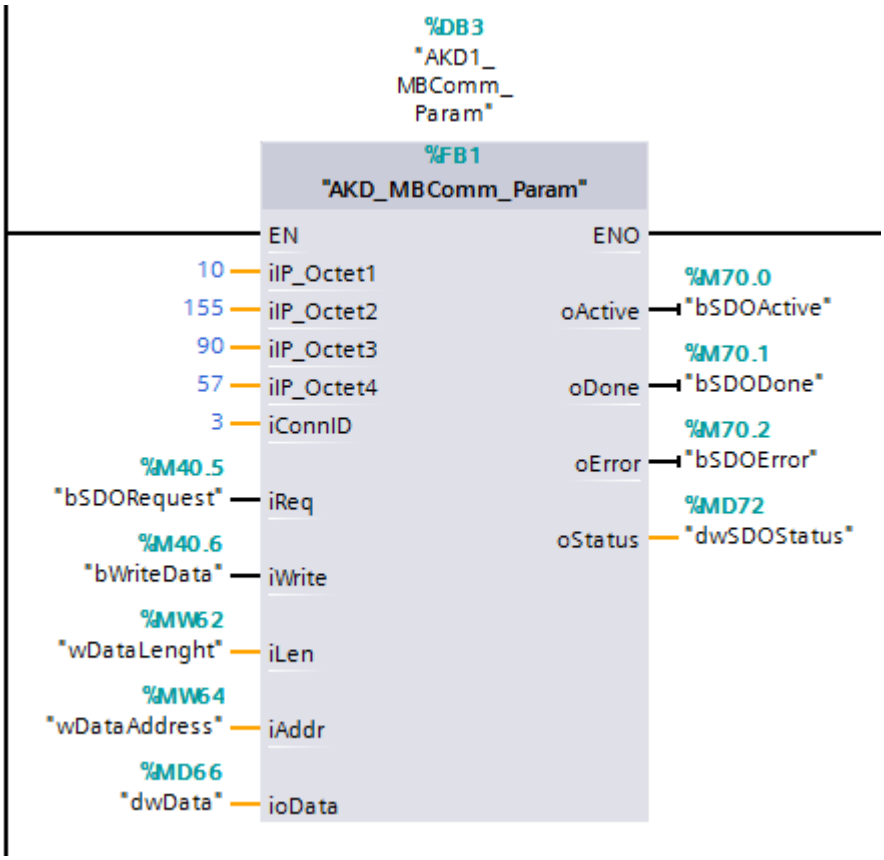
oInPos
Anzeigen das die aktuelle Position innerhalb des MT.POSWND befindet, die Bewegung muss zu diesem Zeitpunkt noch nicht abgeschlossen worden sein. Wird ein Fahrauftrag gestartet oder wieder aufgenommen geht das *oInPos* immer für Minimum einen Zyklus auf null, auch wenn der Fahrauftrag im AKD zu keiner Bewegung oder einem Fehler führt.

oMT_Active
Anzeigen eines aktiven Fahrauftrages.

Direkter Parameterzugriff

Konfiguration PLC

Für den direkten Parameterzugriff wurde ein zweiter Funktionsblock **AKD_MBComm_Param** entwickelt, welcher direkt mit der IP-adresse auf die Daten des verbundenen AKD zugreift. Es ist möglich verschiedene AKD mit demselben Block nacheinander anzusprechen. Dazu muss nur die IP-Adresse geändert werden.



Name der Variable	Datentyp	Richtung	Beschreibung
EN	Bool	In	Aufruf des DB
iIP_OctetX	USInt	In	IP-Adresse des anzusprechenden AKD
iConnID	UInt	In	Verbindungsnummer für den Lesebefehl **(Eineindeutig)**
iReq	Bool	In	Starten eines Lese- oder Schreibbefehls
iWrite	Bool	In	Definiert ob gelesen oder geschrieben wird 0 = Lesen, 1 = Schreiben
iLen	UInt	In	Definiert die Länge des zu bearbeitenden Registers in Word
iAddr	UInt	In	Definiert die Modbusadresse auf dem AKD
ioData	DWord	In/Out	Write = 1, Daten welche geschrieben werden Write = 0, gelesene Daten
ENO	Bool	Out	DB Aufruf erfolgreich
oActive	Bool	Out	MBRW ist in Bearbeitung
oDone	Bool	Out	MBRW wurde erfolgreich abgeschlossen
oError	Bool	Out	MBRW hat einen Fehler oder eine ungültige Eingabe steht an
oStatus	DWord	Out	Status des MBRW Aufrufes
dwDataArray[0..3]	DWord	Static (In/Out)	Datenfeld für die Modbuskommunikation
MBRW	MBClient	Static	MB_Client Funktion zum Lesen oder Schreiben
bMBRW_Req	Bool	Static	Interner Aufruf den nächsten Befehl zu bearbeiten
bMBRW_Discn	Bool	Static	Abbruch der Verbindung
wMBRW_Mode	USInt	Static	Internes setzten des Lese- oder Schreibmodus
dwMBRW_Address	UDInt	Static	Interne Adresse des Parameters
wMBRW_Len	UInt	Static	Interne Länge des zu bearbeitenden Registers
bMBRW_Done	Bool	Static	Interner Status des MBRW: Bearbeitung fertig
bMBRW_Busy	Bool	Static	Interner Status des MBRW: in Bearbeitung
bMBRW_Error	Bool	Static	Interner Status des MBRW: Bearbeitung fehlgeschlagen
wMBRW_Status	Word	Static	Interner Status des MBRW
bDoNextRW	Bool	Static	Befehl in Bearbeitung
bValidFault	Bool	Static	Fehlerhafte Eingabe
wOldIP_OctetX	USInt	Static	Interner Speicher der IP-Adresse des Verbundenen AKD

Zu beachten

Eine Verbindung wird mit dem Einschalten des Datenblockes direkt hergestellt.

Tritt ein Fehler auf so wird die Verbindung abgebrochen und erst nach der Behebung des Fehlers wieder aufgebaut. Mögliche Ursachen für einen Fehler sind eine falsche Eingaben, ein Fehler in der Verbindung oder ein Wechsel der IP-Adresse.

Eine falsche Eingaben ist, wenn die Länge des zu bearbeitenden Registers ≤ 0 oder die Adresse < 0 oder > 9999 ist.

ioData entspricht im Lesemodus (*iWrite* = FALSCH) immer dem dwDataArray[0] und kann nicht von aussen beschrieben werden.

Zur Bearbeitung von Registern mit einer Länge von mehr als zwei Wörter muss der Zugriff wie auch die Ausgabe direkt über das *dwDataArray[]* erfolgen. Das Doppel Wort Feld ist auf eine Länge von vier Konfiguriert, bei Bedarf kann diese erweitert werden auf ein Maximum von 123 Wörter, sprich 61 Doppel Wort. Im Schreibmodus wird das *dwDataArray[0]* immer vom Eingang *ioData* überschrieben, daher muss das erste Wort über den Funktionsaufruf gesetzt werden und die folgenden via direkten Zugriff auf die Datenblockstruktur.

Detailbeschreibung der Ein- und Ausgabe

iReq
Starten des Lesen oder Schreiben. Soll der Befehl nochmals ausgeführt werden muss wieder eine Positive Flanke auftreten.

iWrite
Bestimmt ob gelesen oder geschrieben wird.
0 = Lesen
1 = Schreiben

iLen
Länge des zu bearbeitenden Registers. Momentan wird nur eine Ausgabe bis zwei Register (ein DWord) unterstützt. Für längere Befehle muss der Daten Bereich entsprechend angepasst werden.
2 = eine 32-Bit oder kürzere Variable
4 = eine 64-Bit Variable

iAddr
Adresse des Modbusregisters welches bearbeitet werden soll, siehe AKD Hilfedatei. Für Parameter welche als 64-Bit ausgeführt sind gibt es eine spezielle 32-Bit Version (Mapping von 64-Bit-Parametern auf 32-Bit-Parameter).

ioData
Ausgabe der gelesenen Daten, diese werden angezeigt bis eine neue Anfrage in Bearbeitung ist. Während dem Lesen sind die Daten Null.
Eingabe wenn geschrieben wird. Beim Schreiben werden die Daten im Block nicht abgeändert. Soll ein Befehl aktiviert werden welcher im Workbench keinen Wert benötigt, so muss der *ioData* Wert ungleich Null sein.

oActive
Zeigt an, wenn eine Anfrage in Bearbeitung ist.

oDone
Wird mit abgeschlossener Verarbeitung Eins gesetzt. Die Daten können ab diesem Zeitpunkt genutzt werden. *oDone* wird erst wieder rückgesetzt wenn die Verbindung abgebrochen wird oder bei einer neuen Anfrage (*iReq*).

oError
Zeigt ein Fehler im MBRW oder ein Validierungsfehler an. Ein Validierungsfehler kann auftreten bei Falscher Registeradresse, ungültiger Länge des Registers oder beim Wechsel der IP-Adresse. Bei einem Fehler wird die Verbindung abgebrochen.

oStatus
Spiegelt den Status der Modbusverbindung (MBRW). Hilfe zu den einzelnen Fehlernummern finden sich in der TIA-Portal Hilfe, bei der MB_Client Beschreibung.

Änderungen für ein S1500 oder S1200 PLC mit Firmware ≥ 4.1

Das Projekt AKD_MBComm_Modules.zap13 beinhaltet alle Vorlagen für eine Anbindung an eine S7-1200 PLC. Mit dem Projekt AKD_MBCommS1500_Modules.zap13 ist dieselbe Software abgebildet, mit den Änderungen welche mit dem Wechsel zur S7-1500 PLC benötigt werden. Funktionalität wie auch Bedienung bleiben sich identisch, auf die Spezialitäten des zweiten Software Packet wird hier kurz eingegangen.

Der MB_Client Baustein wurde ab der S1200 V4.0 und im S1500 umgebaut. Es funktionieren nun die internen Signale besser und die Verbindung wird über eine eigene Struktur eingestellt. Dazu wurde die Variable MBR_Connect , MBW_Connect und MBRW_Connect eingefügt diese entsprechen dem Datentyp TCON_IP_v4.

Durch diese Änderung ist es möglich, beim am Funktionsblock ein IP Wechsel bei stehender Verbindung vorzunehmen. Neu kann auch die ID der Verbindung an einem aktiven Block geändert werden.

Mit der S1500 PLC sind die Verzögerungszeiten nicht mehr so stark von der Anzahl Achsen abhängig. Bei einer Achse wurde ein Verzögerung von 8ms, bei zwei Achsen 10ms und bei drei Achsen 12ms gemessen. Diese Zeitmessungen wurden alleine, sprich ohne zusätzlichen Code ausgeführt und sind daher als Best möglich anzusehen.

Dieses Project wurde mit einer Siemens 1513-1 PN getestet und kann bei anderen Konfigurationen Änderungen benötigen welche hier nicht aufgeführt sind.

WISSENSWERTES ÜBER KOLLMORGEN
Kollmorgen ist ein führender Anbieter von Antriebssystemen und Komponenten für den Maschinenbau. Dank großem Know-how im Bereich Antriebssysteme, höchster Qualität und umfassender Fachkenntnisse bei der Verknüpfung und Integration von standardisierten und spezifischen Produkten liefert Kollmorgen optimale Lösungen, die mit Leistung, Zuverlässigkeit und Bedienerfreundlichkeit bestechen und Maschinenbauern einen wichtigen Wettbewerbsvorteil bieten.

Besuchen Sie www.kollmorgen.com für Unterstützung bei der Lösung Ihrer Applikationsaufgabe oder kontaktieren Sie uns unter:

KOLLMORGEN srl
Largo Brughetti 1/B2
20813 Bovisio Masciago, Italia
Web www.kollmorgen.com
Email mil-info@kollmorgen.com
Tel.: +39 - 0362 - 594260
Fax: +39 - 0362 - 594263