Betriebsanleitung

SICK
AFS60/AFM60 EtherNet/IP
Add-On Instruction

SICK_AFX60 Add-On Instruction für
Rockwell Automation Steuerungen

Ausgabestand: Juli 2012

SICK
Sensor Intelligence.

Inhaltsverzeichnis

1 Zu diesem Dokument 3
1.1 Funktion dieses Dokuments 3
1.2 Zielgruppe 3
2 Allgemeines 4
2.1 Bausteinspezifikationen 5
3 Einbinden der AOI in RSLogix5000 6
3.1 Hardwarekonfiguration 6
3.2 AOI Import 6
3.3 Bausteinaufruf 7
3.3.1 Konfiguration des GetMessage Parameters 8
3.3.2 Konfiguration des SetMessage Parameters 9
3.4 Lesen von Geräteparametern 10
3.5 Schreiben von Geräteparametern 12
3.6 Verhalten im Fehlerfall 12
4 Parameterübersicht 13
5 Fehlercodes 15
6 Beispiel 17
6.1 Programmaufruf 17
6.2 Auslesen von Geräteparametern 18
6.3 Schreiben von Geräteparametern 19

1 Zu diesem Dokument

Bitte lesen Sie dieses Kapitel sorgfältig, bevor Sie mit dieser Betriebsanleitung und der SICK_AFX60 Add-On Instruction arbeiten.

1.1 Funktion dieses Dokuments

Diese Betriebsanleitung beschreibt den Umgang mit der SICK_AFX60 Add-On Instruction. Sie leitet das technische Personal des Maschinenherstellers bzw. Maschinenbetreibers zur Projektierung und Inbetriebnahme der Add-On Instructions an.

1.2 Zielgruppe

Diese Betriebsanleitung richtet sich an fachkundiges Personal wie z.B. Techniker oder Ingenieure.

2 Allgemeines

Die Add-On Instruction (AOI) wird zur Kommunikation zwischen einer Rockwell Steuerung und dem SICK AFS60/AFM60 EtherNet/IP Encoder verwendet. Der Encoder muss hierfür in das EtherNet/IP Umfeld des Steuerungsprojekts eingebunden werden. Die Kommunikation erfolgt mit dem Encoder azyklisch über das CIP-Protokoll.

Mit der AOI ist es möglich auf Anforderung ausgewählte Sensordaten aus dem Gerät auszulesen und zu beschreiben

Mit dem Baustein können die folgenden Geräteparameter ausgelesen werden:
- Seriennummer
- Skalierung Flag
- Anzahl der Schritte pro Umdrehung
- Gesamtauflösung
- Preset-Wert
- Unteres Limit der Position
- Oberes Limit der Position
- Geschwindigkeitsformat
- Geschwindigkeitsuntergrenze
- Geschwindigkeitsobergrenze
- Warnungen
- Warnungs-Flag
- Encoder Temperatur
- Format der Encodertemperatur
- Bewegungszeit des Encoders
- Betriebszeit des Encoders
- Höchste Geschwindigkeit seit Inbetriebnahme

Mit dem Baustein können die folgenden Geräteparameter beschrieben werden:
- Preset-Wert
- Unteres Limit der Position
- Oberes Limit der Position
- Geschwindigkeitsuntergrenze
- Geschwindigkeitsobergrenze
- Format der Encodertemperatur

2.1 Bausteinspezifikationen

AOI Name:	SICK_AFX60
Version:	1.0
Routinenname:	Logic
Verwendete UDTs:	SICK_AFX60_DATA SICK_AFX60_READ_SEL SICK_AFX60_READ SICK_AFX60_WRITE_SEL SICK_AFX60_WRITE
Bausteinaufruf:	Zyklisch
Erstellsprache:	Strukturierter Text (ST)

Die AFX60 AOI ist eine asynchron arbeitende Routine d.h. die Bearbeitung erstreckt sich über mehrere SPS Zyklen. Die Routine muss solange im Anwenderprogramm aufgerufen werden, bis die Bearbeitung abgeschlossen ist.

Die folgende Abbildung zeigt die Darstellung der AOI in der Relay Ladder Logik.

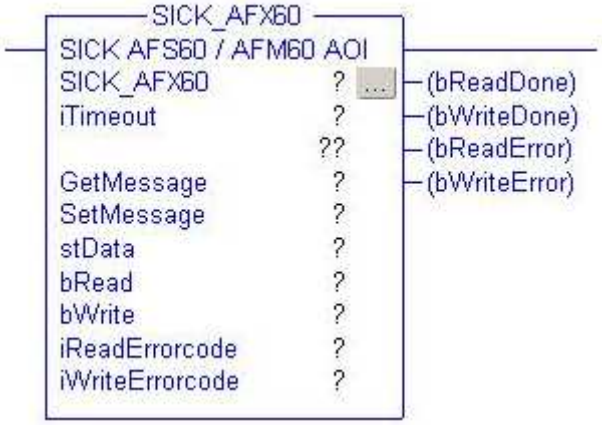


Abbildung 1: Darstellung der SICK_AFX60 AOI

3 Einbinden der AOI in RSLogix5000

Die Implementierung des AFS60/AFS60 Encoders wird über eine Add-On Instruction (AOI) gehandhabt. Diese AOI beinhaltet eine Programmroutine und mehrere Datenstrukturen (UDTs). Die Programmroutine kann an einer beliebigen Stelle des Anwenderprogramms aufgerufen werden und ermöglicht einen azyklischen Datenaustausch mit dem Sensor (CIP-Generic).

3.1 Hardwarekonfiguration

Bitte entnehmen Sie die Hardwarekonfiguration der Bedienungsanleitung des AFS60/AFM60 Encoders. Sobald der Sensor in das LSLogix5000 eingebunden ist, stehen die zyklischen Prozessdaten des Encoders zur Verfügung. Die AOI unterstützt zusätzlich ein Zugriff auf die azyklischen Servicedaten des Encoders.

3.2 AOI Import

Um die SICK_AFX60 AOI im Anwenderprogramm benutzten zu können, muss diese zunächst über „File → Import Component → Add-On Instruction“ in ein bestehendes Projekt importiert werden.

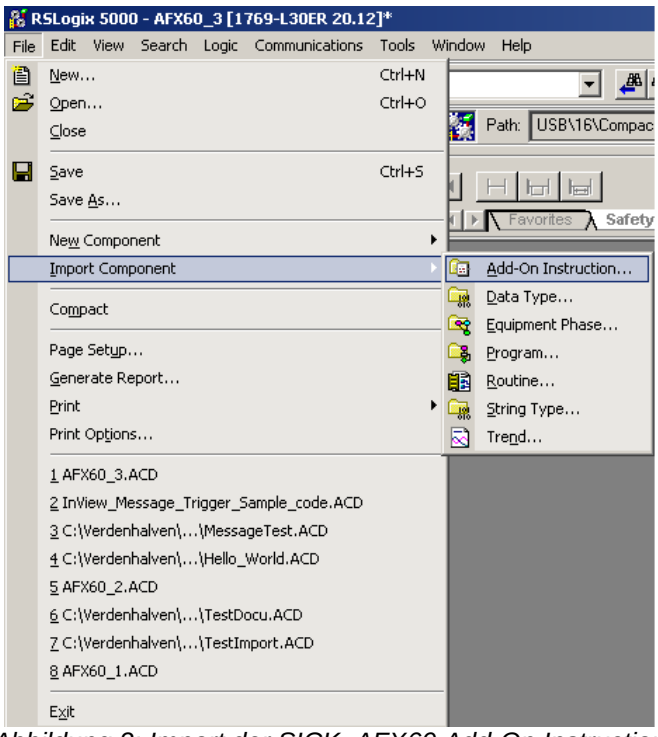


Abbildung 2: Import der SICK_AFX60 Add-On Instruction

3.3 Bausteinaufruf

Um die SICK_AFX60 AOI einsetzten zu können, muss diese zunächst im Anwenderprogramm aufgerufen und mit gültigen Parametern beschaltet werden. Nur mit einer gültigen Beschaltung ist es möglich Sensorparameter auszulesen bzw. zu beschreiben.

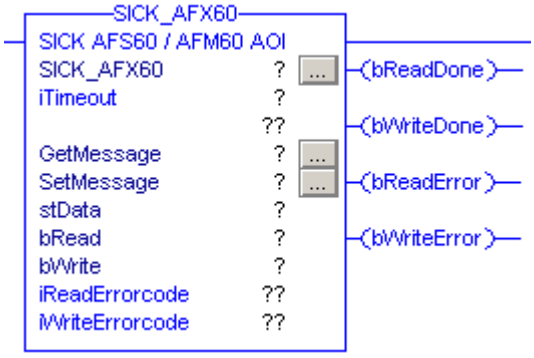


Abbildung 3: Ansicht der SICK_AFX60 Routine im Anwenderprogramm

SICK_AFX60: Name der Bausteininstanz. Über die Bausteininstanz ist es möglich auf Input und Output Parameter der Routine zuzugreifen.

iTimeout: Zeit nachdem ein Timeout-Fehler ausgelöst wird, sollte die Bearbeitungszeit diese überschreiten. Die Timeout Zeit wird in Millisekunden angegeben (üblicher Wert: 5000ms).

GetMessage: Zusätzlich zur Beschaltung des Parameters ist es erforderlich diesen über die rechtsstehende Schaltfläche "…" zu konfigurieren.

SetMessage: Zusätzlich zur Beschaltung des Parameters ist es erforderlich diesen über die rechtsstehende Schaltfläche "…" zu konfigurieren.

stData: Der „stData" Parameter verwendet die Struktur des mitgelieferten SICK_AFX_DATA UDTs. Hier werden die gelesenen und die zu schreibenden Parameterwerte des Encoders abgelegt. Der UDT muss als Controller-Tag instanziiert und der Routine übergeben werden.

bRead: Über eine steigende Flanke werden alle von der AOI unterstützten Sensorparameter ausgelesen.

bWrite: Über eine steigende Flanke werden alle in den Datentyp „stData" ausgewählten Parameterwerte in den Encoder geschrieben.

Abbildung 4 zeigt eine Beispielbeschaltung der Routine im Anwenderprogramm (Ladder Logik).

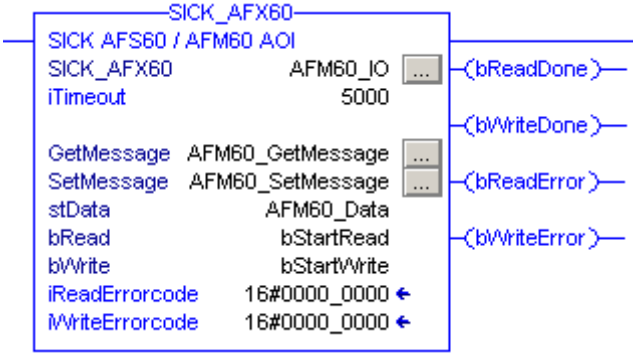


Abbildung 4: Beispielbeschaltung der SICK_AFX60 Routine

3.3.1 Konfiguration des GetMessage Parameters

Der GetMessage Parameter muss über die Schaltfläche "…" (neben der Zuweisung) einmalig wie folgt konfiguriert werden.

Message Type:	CIP Generic
Service Type:	Get Attribute Single
Service Code:	Wird durch den Service Typen auf 16#E automatisch voreingestellt
Instance:	1
Class:	1
Attribute:	1
Destination Element:	Verweis auf das Array „arrReadRecord" aus der Datenstruktur die dem Eingangsparameter „stData" übergeben wird.
Path:	Hardwarename des anzusprechenden Encoders, der in der RSLogix5000 Projektiert ist. Der Encoder kann über die „Browse…" Schaltfläche ausgewählt werden.
Connected:	Nicht aktiv: Verbindung wird auf Anfrage automatisch auf- und wieder abgebaut. Aktiv: Verbindung bleibt dauerhaft bestehen (Bearbeitungszeit für Auf- und Abbau der Verbindung entfällt).

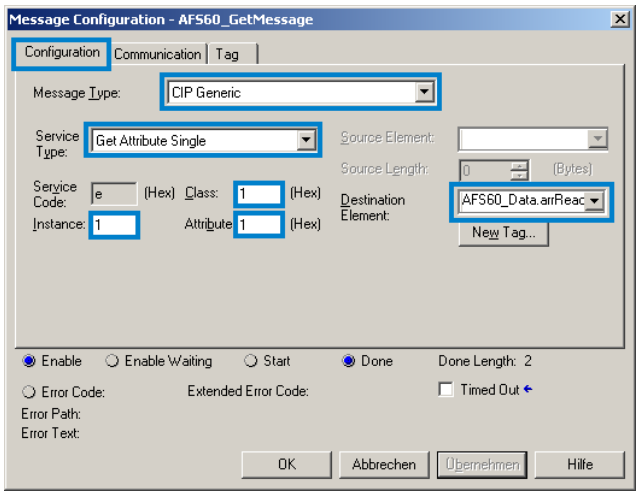


Abbildung 5: Konfiguration des GetMessage Parameters (Configuration)

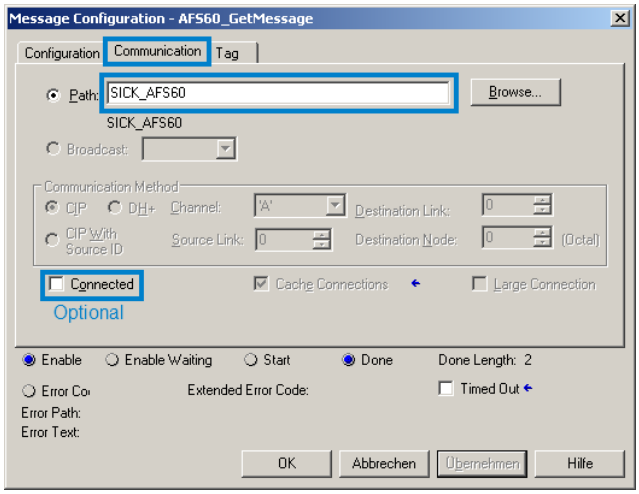


Abbildung 6: Konfiguration des GetMessage Parameters (Communication)

3.3.2 Konfiguration des SetMessage Parameters

Der SetMessage Parameter muss über die Schaltfläche "…" (neben der Zuweisung) einmalig wie folgt konfiguriert werden.

Message Type:	CIP Generic
Service Type:	Set Attribute Single
Service Code:	Wird durch den Service Typen auf 16#10 automatisch voreingestellt
Instance:	1
Class:	1
Attribute:	1
Source Element:	Verweis auf das Array „arrWriteRecord" aus der Datenstruktur die dem Eingangsparameter „stData" übergeben wird.
Source Length:	4
Path:	Hardwarename des anzusprechenden Encoders, der in der RSLogix5000 Projektiert ist. Der Encoder kann über die „Browse…" Schaltfläche ausgewählt werden.
Connected:	Nicht aktiv: Verbindung wird auf Anfrage automatisch auf- und wieder abgebaut. Aktiv: Verbindung bleibt dauerhaft bestehen (Bearbeitungszeit für Auf- und Abbau der Verbindung entfällt).

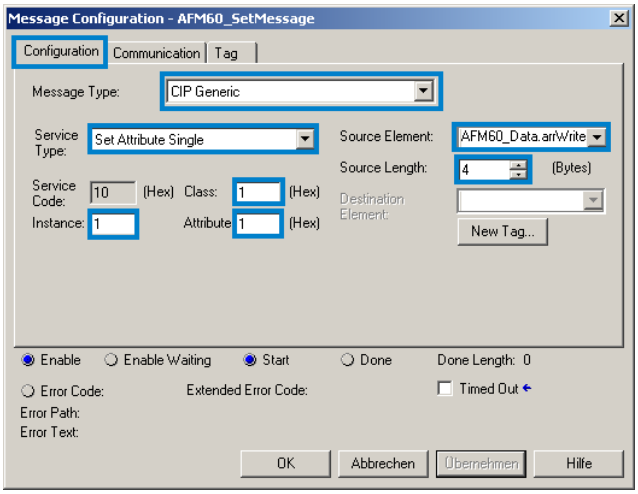


Abbildung 7: Konfiguration des SetMessage Parameters (Configuration)

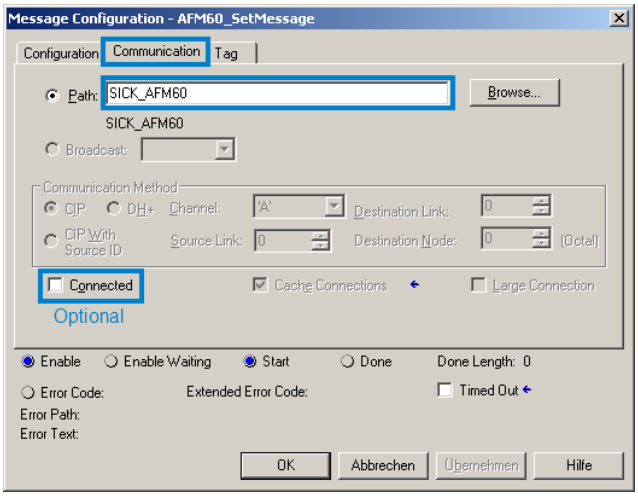


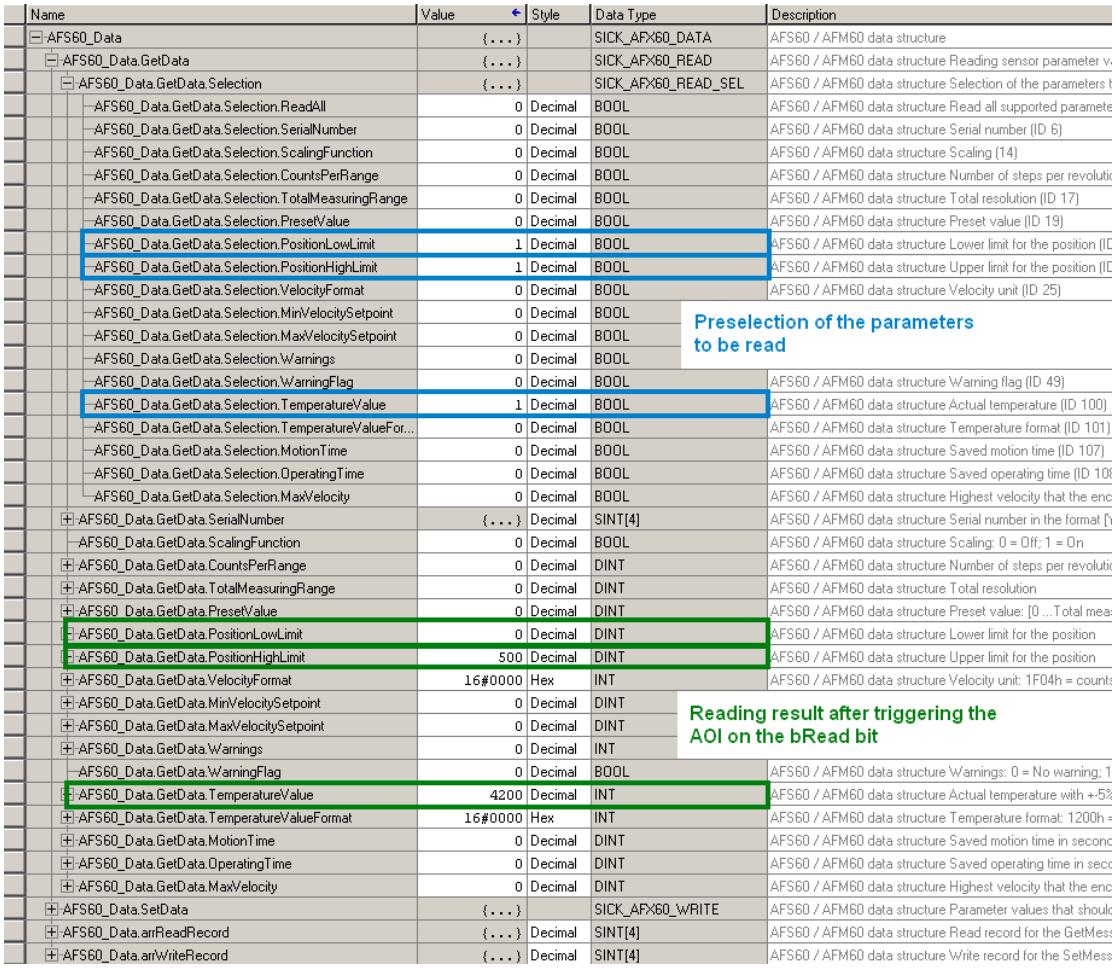
Abbildung 8: Konfiguration des SetMessage Parameters (Communication)

3.4 Lesen von Geräteparametern

Bevor die Bausteinaktion „Parameter lesen“ ausgeführt werden kann, muss angegeben werden, welche Geräteparameter ausgelesen werden sollen. Diese Auswahl wird in der zugeordneten Datenstruktur „stData“ in der Unterstruktur „GetData.Selection“ vorgenommen.

Ist in der Unterstruktur „GetData.Selection“ ein Parameter mit dem Wert “1“ belegt, bedeutet dies, dass der gewählte Parameter beim Ausführen der Bausteinaktion ausgelesen wird. Ist das Bit „ReadAll“ gesetzt, werden alle in der Unterstruktur „GetData“ aufgeführten Geräteparameter zum auslesen vorgewählt.

Abbildung 9 zeigt exemplarisch das Vorgehen beim auslesen der Geräteparameter „PositionLowLimit“, „PositionHighLimit“ und „TemperatureValue“.



Name	Value	Style	Data Type	Description
AFS60_Data	{...}		SICK_AFX60_DATA	AFS60 / AFM60 data structure
AFS60_Data.GetData	{...}		SICK_AFX60_READ	AFS60 / AFM60 data structure Reading sensor parameter v
AFS60_Data.GetData.Selection	{...}		SICK_AFX60_READ_SEL	AFS60 / AFM60 data structure Selection of the parameters t
AFS60_Data.GetData.Selection.ReadAll	0	Decimal	BOOL	AFS60 / AFM60 data structure Read all supported paramete
AFS60_Data.GetData.Selection.SerialNumber	0	Decimal	BOOL	AFS60 / AFM60 data structure Serial number (ID 6)
AFS60_Data.GetData.Selection.ScalingFunction	0	Decimal	BOOL	AFS60 / AFM60 data structure Scaling (14)
AFS60_Data.GetData.Selection.CountsPerRange	0	Decimal	BOOL	AFS60 / AFM60 data structure Number of steps per revolutio
AFS60_Data.GetData.Selection.TotalMeasuringRange	0	Decimal	BOOL	AFS60 / AFM60 data structure Total resolution (ID 17)
AFS60_Data.GetData.Selection.PresetValue	0	Decimal	BOOL	AFS60 / AFM60 data structure Preset value (ID 19)
AFS60_Data.GetData.Selection.PositionLowLimit	1	Decimal	BOOL	AFS60 / AFM60 data structure Lower limit for the position (ID
AFS60_Data.GetData.Selection.PositionHighLimit	1	Decimal	BOOL	AFS60 / AFM60 data structure Upper limit for the position (ID
AFS60_Data.GetData.Selection.VelocityFormat	0	Decimal	BOOL	AFS60 / AFM60 data structure Velocity unit (ID 25)
AFS60_Data.GetData.Selection.MinVelocitySetpoint	0	Decimal	BOOL	
AFS60_Data.GetData.Selection.MaxVelocitySetpoint	0	Decimal	BOOL	
AFS60_Data.GetData.Selection.Warnings	0	Decimal	BOOL	
AFS60_Data.GetData.Selection.WarningFlag	0	Decimal	BOOL	AFS60 / AFM60 data structure Warning flag (ID 49)
AFS60_Data.GetData.Selection.TemperatureValue	1	Decimal	BOOL	AFS60 / AFM60 data structure Actual temperature (ID 100)
AFS60_Data.GetData.Selection.TemperatureValueFor...	0	Decimal	BOOL	AFS60 / AFM60 data structure Temperature format (ID 101)
AFS60_Data.GetData.Selection.MotionTime	0	Decimal	BOOL	AFS60 / AFM60 data structure Saved motion time (ID 107)
AFS60_Data.GetData.Selection.OperatingTime	0	Decimal	BOOL	AFS60 / AFM60 data structure Saved operating time (ID 10
AFS60_Data.GetData.Selection.MaxVelocity	0	Decimal	BOOL	AFS60 / AFM60 data structure Highest velocity that the enc
AFS60_Data.GetData.SerialNumber	{...}	Decimal	SINT[4]	AFS60 / AFM60 data structure Serial number in the format [Y
AFS60_Data.GetData.ScalingFunction	0	Decimal	BOOL	AFS60 / AFM60 data structure Scaling: 0 = Off; 1 = On
AFS60_Data.GetData.CountsPerRange	0	Decimal	DINT	AFS60 / AFM60 data structure Number of steps per revolutio
AFS60_Data.GetData.TotalMeasuringRange	0	Decimal	DINT	AFS60 / AFM60 data structure Total resolution
AFS60_Data.GetData.PresetValue	0	Decimal	DINT	AFS60 / AFM60 data structure Preset value: [0 ...Total mea
AFS60_Data.GetData.PositionLowLimit	0	Decimal	DINT	AFS60 / AFM60 data structure Lower limit for the position
AFS60_Data.GetData.PositionHighLimit	500	Decimal	DINT	AFS60 / AFM60 data structure Upper limit for the position
AFS60_Data.GetData.VelocityFormat	16#0000	Hex	INT	AFS60 / AFM60 data structure Velocity unit: 1F04h = counts
AFS60_Data.GetData.MinVelocitySetpoint	0	Decimal	DINT	
AFS60_Data.GetData.MaxVelocitySetpoint	0	Decimal	DINT	
AFS60_Data.GetData.Warnings	0	Decimal	INT	
AFS60_Data.GetData.WarningFlag	0	Decimal	BOOL	AFS60 / AFM60 data structure Warnings: 0 = No warning; 1
AFS60_Data.GetData.TemperatureValue	4200	Decimal	INT	AFS60 / AFM60 data structure Actual temperature with +-5%
AFS60_Data.GetData.TemperatureValueFormat	16#0000	Hex	INT	AFS60 / AFM60 data structure Temperature format: 1200h =
AFS60_Data.GetData.MotionTime	0	Decimal	DINT	AFS60 / AFM60 data structure Saved motion time in seconc
AFS60_Data.GetData.OperatingTime	0	Decimal	DINT	AFS60 / AFM60 data structure Saved operating time in secc
AFS60_Data.GetData.MaxVelocity	0	Decimal	DINT	AFS60 / AFM60 data structure Highest velocity that the enc
AFS60_Data.SetData	{...}		SICK_AFX60_WRITE	AFS60 / AFM60 data structure Parameter values that should
AFS60_Data.arrReadRecord	{...}	Decimal	SINT[4]	AFS60 / AFM60 data structure Read record for the GetMess
AFS60_Data.arrWriteRecord	{...}	Decimal	SINT[4]	AFS60 / AFM60 data structure Write record for the SetMess

Abbildung 9: Auslesen von Geräteparametern

Zu jeder Vorauswahl in der Unterstruktur „GetData.Selection“ gehört ein entsprechender Wert in der Unterstruktur „GetData“, wo die ausgelesenen Werte gespeichert werden.

Um die Bausteinaktion auszuführen muss der Eingangsparameter bRead mit einer positiven Flanke (Signalwechsel von logisch null auf eins) angetriggert werden. Wenn die Routine am Ausgangsparameter bReadDone = TRUE signalisiert, wurde die Aktion erfolgreich durchgeführt. Die Leseergebnisse wird in den am Eingang „stData“ übergebenden TAG abgespeichert (Unterstruktur „GetData“). Eine Beschreibung der Geräteparameter finden Sie in der entsprechenden Gerätedokumentation.

3.5 Schreiben von Geräteparametern

Bevor die Bausteinaktion „Parameter schreiben“ auszuführt werden kann, muss angegeben werden, welche Geräteparameter geschrieben werden sollen. Diese Auswahl wird in der zugeordneten Datenstruktur „stData“ in der Unterstruktur „SetData.Selection“ vorgenommen.

Ist in der Unterstruktur „SetData.Selection“ ein Parameter mit dem Wert “1“ belegt, bedeutet dies, dass beim Ausführen der Bausteinaktion der zugehörige Parameterwert „SetData“ beschrieben wird. Es kann ein oder mehrere Parameter vorgewählt werden.

Zu jeder Vorauswahl in der Unterstruktur „SetData.Selection“ gehört ein entsprechender Wert in der Unterstruktur „SetData“. Eine Beschreibung der Geräteparameter finden Sie in der entsprechenden Gerätedokumentation.

Abbildung 10 zeigt exemplarisch das Vorgehen beim schreiben der Geräteparameter „PresetValue“ und „PositionHighLimit“.

Name	Value	Style	Data Type	Description
AFS60_Data	{...}		SICK_AFX60_DATA	AFS60 / AFM60 data structure
AFS60_Data.GetData	{...}		Preselection of the parameters to be written	
AFS60_Data.SetData	{...}			
AFS60_Data.SetData.Selection	{...}		SICK_AFX60_WRITE_SEL	AFS60 / AFM60 data structure Selection of the
AFS60_Data.SetData.Selection.PresetValue	1	Decimal	BOOL	AFS60 / AFM60 data structure Preset value (ID
AFS60_Data.SetData.Selection.PositionLowLimit	0	Decimal	BOOL	AFS60 / AFM60 data structure Lower limit for th
AFS60_Data.SetData.Selection.PositionHighLimit	1	Decimal	BOOL	AFS60 / AFM60 data structure Upper limit for th
AFS60_Data.SetData.Selection.MinVelocitySetpoint	0	Decimal	BOOL	AFS60 / AFM60 data structure Minimum velocit
AFS60_Data.SetData.Selection.MaxVelocitySetpoint	0	Decimal	BOOL	AFS60 / AFM60 data structure Maximum veloci
AFS60_Data.SetData.Selection.TemperatureValueFor...	0	Decimal	BOOL	AFS60 / AFM60 data structure Temperature for
AFS60_Data.SetData.PresetValue	123	Decimal	DINT	AFS60 / AFM60 data structure Preset value: [0
AFS60_Data.SetData.PositionLowLimit	0	Decimal	DINT	AFS60 / AFM60 data structure Lower limit for th
AFS60_Data.SetData.PositionHighLimit	800	Decimal	DINT	AFS60 / AFM60 data structure Upper limit for th
AFS60_Data.SetData.MinVelocitySetpoint	0	Decimal	DINT	AFS60 / AFM60 data structure Minimum velocit
AFS60_Data.SetData.MaxVelocitySetpoint	0	Decimal	Values to be written	
AFS60_Data.SetData.TemperatureValueFormat	16#0000	Hex		
AFS60_Data.arrReadRecord	{...}	Decimal	SINT[4]	AFS60 / AFM60 data structure Read record for
AFS60_Data.arrWriteRecord	{...}	Decimal	SINT[4]	AFS60 / AFM60 data structure Write record for

Abbildung 10: Schreiben von Geräteparametern

Um das schreiben eines Geräteparameters anzustoßen muss der Eingangsparameter bWrite mit einer positiven Flanke (Signalwechsel von logisch null auf eins) angetriggert werden. Der Ausgangsparameter bWriteDone = TRUE signalisiert, dass die Aktion erfolgreich durchgeführt wurde.

3.6 Verhalten im Fehlerfall

Bei einem fehlerhaften Eingabewert oder einer fehlerhaften Eingangsbeschaltung der Programmroutine, wird ein Errorbit (bReadError, bWriteError) gesetzt und ein Fehlercode (iReadErrorcode, iWriteErrorcode) ausgegeben. In diesem Fall wird keine weitere Bearbeitung durchgeführt. Die Diagnoseparameter (bReadError, bWriteError, iReadErrorcode, iWriteErrorcode) der Routine behalten solange ihren Wert, bis ein neuer Auftrag gestartet wird.

4 Parameterübersicht

Parameter	Deklaration	Datentyp	Beschreibung
EnableIn	Input	BOOL	Enable Input (nur für Ladder Logik)
iTimeout	Input	DINT	Zeit nachdem ein Timeout-Fehler ausgelöst wird, sollte die Bearbeitung des Bausteins diese überschreiten. Die Timeout Zeit wird in Millisekunden angegeben. Üblicher Wert: 5000 [ms]
GetMessage	InOut	MESSAGE	Instanz für den intern verwendeten MSG-Befehl zum auslesen eines Sensorparameters. Dieser Parameter muss wie in Kapitel 3.3.1 beschrieben konfiguriert werden.
SetMessage	InOut	MESSAGE	Instanz für den intern verwendeten MSG-Befehl zum schreiben eines Sensorparameters. Dieser Parameter muss wie in Kapitel 3.3.2 beschrieben konfiguriert werden.
stData	InOut	SICK_AFX60_DATA	Dieser Parameter verwendet die Struktur des mitgelieferten SICK_AFX_DATA UDTs. Hier werden die gelesenen und die zu schreibenden Parameterwerte abgelegt. Der UDT muss als Controller-Tag instanziiert und der Routine übergeben werden.
bRead	InOut	BOOL	Über eine steigende Flanke werden alle in dem Datentyp „stData“ (Unterstruktur GetData.Selection) ausgewählten Geräteparameter ausgelesen.
bWrite	InOut	BOOL	Über eine steigende Flanke werden alle in dem Datentyp „stData“ (Unterstruktur SetData.Selection) ausgewählten Parameterwerte in den Encoder geschrieben.
bReadDone	Output	BOOL	Zeigt an, ob die Leseaktion fehlerfrei durchgeführt wurde. TRUE: Bearbeitung abgeschlossen FALSE: Bearbeitung nicht abgeschlossen
bWriteDone	Output	BOOL	Zeigt an, ob die Schreibaktion fehlerfrei durchgeführt wurde. TRUE: Bearbeitung abgeschlossen FALSE: Bearbeitung nicht abgeschlossen

Parameter	Deklaration	Datentyp	Beschreibung
bReadError	Output	BOOL	Das Fehlerbit zeigt an, ob bei der Durchführung der Leseaktion ein Bausteinfehler aufgetreten ist. TRUE: Kein Fehler FALSE: Abbruch mit Fehler
bWriteError	Output	BOOL	Das Fehlerbit zeigt an, ob bei der Durchführung der Schreibaktion ein Bausteinfehler aufgetreten ist. TRUE: Kein Fehler FALSE: Abbruch mit Fehler
iReadErrorcode	Output	DINT	Fehlerstatus der Leseaktion (siehe Fehlercodes)
iWriteErrorcode	Output	DINT	Fehlerstatus Schreibaktion (siehe Fehlercodes)
EnableOut	Output	BOOL	Enable Output (nur für Ladder Logik)

Tabelle 1: Parameterübersicht der SICK_AFX60 AOI

5 Fehlercodes

Die Ausgangsparameter iReadErrorcode und iWriteErrorcode enthalten die folgenden Fehlerinformationen jeweils im Bezug auf die Lese- oder Schreibaktion:

- Bausteinfehlercodes
- Message Fehlercodes
- Erweiterte Message Fehlercode

Fehlercode (32Bit)	
Erweiterte Message Fehlercodes (16Bit)	Message Fehlercode (16Bit) Bausteinfehlercodes (16Bit)

Fehlercode	Kurzbeschreibung	Beschreibung
16#**0000_0000**	Kein Fehler	Kein Fehler
16#**0000_F001**	Timeout Fehler	Die gewählte Bausteinaktion konnte innerhalb der Timeout Zeit nicht ausgeführt werden.
16#**0000_F002**	Interner Fehler	Interner Bausteinfehler
16#**0000_F003**	Ungültiges Attribut ausgelesen	Geräteanfrage passt nicht mit Geräteantwort überein. Mögliche Ursache ist eine Mehrfachverwendung der verwendeten MSG-Instanz.
16#**0000_F004**	Ungültige Klasse ausgelesen	Geräteanfrage passt nicht mit Geräteantwort überein. Mögliche Ursache ist eine Mehrfachverwendung der verwendeten MSG-Instanz.
16#**0000_F005**	Ungültige Parameterauswahl	Keine Parameter zum lesen/schreiben ausgewählt. Bitte wählen sie einen, oder mehrere Parameter in der jeweiligen „Selection“ Struktur aus.
16#**0000_F006**	Ungültiger Eingabewert: PositionLowLimit	Ungültiger Eingabewert beim schreiben des Parameters „PositionLowLimit“. Gültiger Wertebereich: [0 .. 1073741823]
16#**0000_F007**	Ungültiger Eingabewert: PositionHighLimit	Ungültiger Eingabewert beim schreiben des Parameters „PositionHighLimit“. Gültiger Wertebereich: [0 .. 1073741823]
16#**0000_F008**	Ungültiger Eingabewert: MinVelocitySetpoint	Ungültiger Eingabewert beim schreiben des Parameters „MinVelocitySetpoint“. Gültiger Wertebereich: [0 .. 1073741823]

Fehlercode	Kurzbeschreibung	Beschreibung
16#**0000_F009**	Ungültiger Eingabewert: MaxVelocitySetpoint	Ungültiger Eingabewert beim schreiben des Parameters „MaxVelocitySetpoint“. Gültiger Wertebereich: [0 .. 1073741823]
16#**0000_F00A**	Ungültiger Eingabewert: TemperaturValueFormat	Ungültiger Eingabewert beim schreiben des Parameters „TemperaturValueFormat“. Gültige Werte: [4608, 4609]
16#xxxx_**0001** - 16#xxxx_**7000**	Message Fehler	Fehlercode, der von den verwendeten MSG-Boxen generiert wird. Die vollständige Beschreibung dieser Fehlercodes entnehmen Sie bitte dem RSLogix5000 Hilfesystem (Stichwort: Error codes, message).
16#**0001**_xxxx - 16#**00FF**_xxxx	Erweiterter Message Fehler	Erweiterter Fehlercode, der von den verwendeten MSG-Boxen generiert wird. Die vollständige Beschreibung dieser Fehlercodes entnehmen Sie bitte dem RSLogix5000 Hilfesystem (Stichwort: Error codes, message).

Tabelle 2: Fehlercodes der SICK_AFX60 AOI

6 Beispiel

Das folgende Beispiel zeigt den Umgang mit der SICK_AFX60 AOI. Vorraussetzung für die Kommunikation zwischen Encoder und SPS ist eine korrekte Hardwareprojektierung sowie eine gültige Parameterbeschaltung der aufzurufenden AOI.

6.1 Programmaufruf

Beim Programmaufruf muss darauf geachtet werden, dass die Parameter „GetMessage“ und „SetMessage“ zusätzlich konfiguriert werden müssen (siehe Kapitel 3.3.1 und 3.3.2).

Der Timeout ist mit 5000ms angegeben. Dies bedeutet, dass die Routine einen Timeout-Fehler auslöst, sobald die Bearbeitungszeit des Bausteins länger als 5 Sekunden beträgt.

Eine Lese- / Schreibfunktion wird ausgeführt, wenn das jeweilige Bit „bStartRead“ / „bStartWrite“ mit einer positiven Flanke angetriggert wird. Lese- und Schreibaktionen sind voneinander unabhängig und können auch Zeitgleich gestartet werden.

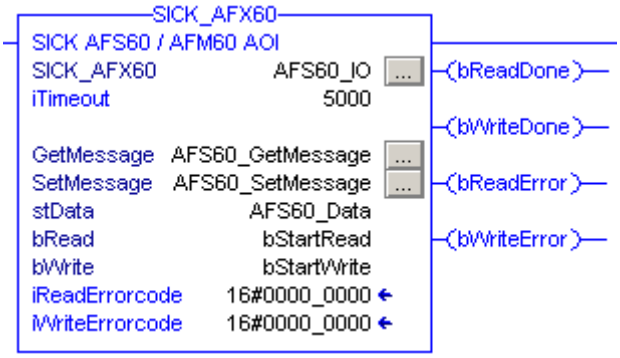
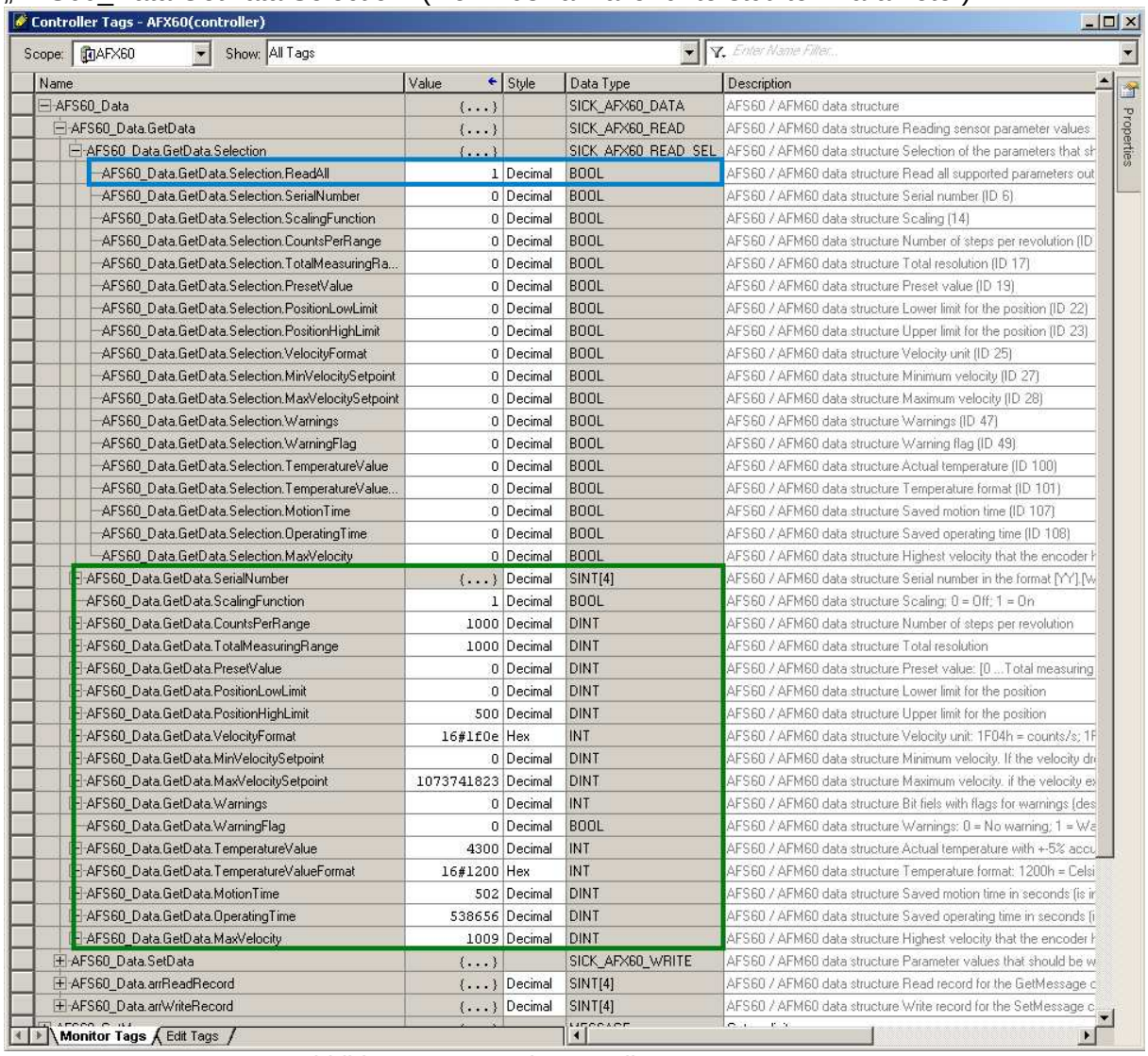


Abbildung 11: Programmaufruf der SICK_AFX60 Routine

6.2 Auslesen von Geräteparametern

Über eine steigende Flanke (bStartRead) werden alle vorausgewählten Sensorparameter ausgelesen. Die Auswahl der zu lesenden Geräteparameter erfolgt über die Datenstruktur „AFS60_Data.GetData.Selection“ (hier: Auswahl aller unterstützten Parameter).



Name	Value	Style	Data Type	Description
AFS60_Data	{...}		SICK_AFX60_DATA	AFS60 / AFM60 data structure
AFS60_Data.GetData	{...}		SICK_AFX60_READ	AFS60 / AFM60 data structure Reading sensor parameter values
AFS60_Data.GetData.Selection	{...}		SICK_AFX60_READ_SEL	AFS60 / AFM60 data structure Selection of the parameters that sh
AFS60_Data.GetData.Selection.ReadAll	1	Decimal	BOOL	AFS60 / AFM60 data structure Read all supported parameters out
AFS60_Data.GetData.Selection.SerialNumber	0	Decimal	BOOL	AFS60 / AFM60 data structure Serial number (ID 6)
AFS60_Data.GetData.Selection.ScalingFunction	0	Decimal	BOOL	AFS60 / AFM60 data structure Scaling (14)
AFS60_Data.GetData.Selection.CountsPerRange	0	Decimal	BOOL	AFS60 / AFM60 data structure Number of steps per revolution (ID
AFS60_Data.GetData.Selection.TotalMeasuringRa...	0	Decimal	BOOL	AFS60 / AFM60 data structure Total resolution (ID 17)
AFS60_Data.GetData.Selection.PresetValue	0	Decimal	BOOL	AFS60 / AFM60 data structure Preset value (ID 19)
AFS60_Data.GetData.Selection.PositionLowLimit	0	Decimal	BOOL	AFS60 / AFM60 data structure Lower limit for the position (ID 22)
AFS60_Data.GetData.Selection.PositionHighLimit	0	Decimal	BOOL	AFS60 / AFM60 data structure Upper limit for the position (ID 23)
AFS60_Data.GetData.Selection.VelocityFormat	0	Decimal	BOOL	AFS60 / AFM60 data structure Velocity unit (ID 25)
AFS60_Data.GetData.Selection.MinVelocitySetpoint	0	Decimal	BOOL	AFS60 / AFM60 data structure Minimum velocity (ID 27)
AFS60_Data.GetData.Selection.MaxVelocitySetpoint	0	Decimal	BOOL	AFS60 / AFM60 data structure Maximum velocity (ID 28)
AFS60_Data.GetData.Selection.Warnings	0	Decimal	BOOL	AFS60 / AFM60 data structure Warnings (ID 47)
AFS60_Data.GetData.Selection.WarningFlag	0	Decimal	BOOL	AFS60 / AFM60 data structure Warning flag (ID 49)
AFS60_Data.GetData.Selection.TemperatureValue	0	Decimal	BOOL	AFS60 / AFM60 data structure Actual temperature (ID 100)
AFS60_Data.GetData.Selection.TemperatureValue...	0	Decimal	BOOL	AFS60 / AFM60 data structure Temperature format (ID 101)
AFS60_Data.GetData.Selection.MotionTime	0	Decimal	BOOL	AFS60 / AFM60 data structure Saved motion time (ID 107)
AFS60_Data.GetData.Selection.OperatingTime	0	Decimal	BOOL	AFS60 / AFM60 data structure Saved operating time (ID 108)
AFS60_Data.GetData.Selection.MaxVelocity	0	Decimal	BOOL	AFS60 / AFM60 data structure Highest velocity that the encoder h
AFS60_Data.GetData.SerialNumber	{...}	Decimal	SINT[4]	AFS60 / AFM60 data structure Serial number in the format [MY].[W
AFS60_Data.GetData.ScalingFunction	1	Decimal	BOOL	AFS60 / AFM60 data structure Scaling: 0 = Off; 1 = On
AFS60_Data.GetData.CountsPerRange	1000	Decimal	DINT	AFS60 / AFM60 data structure Number of steps per revolution
AFS60_Data.GetData.TotalMeasuringRange	1000	Decimal	DINT	AFS60 / AFM60 data structure Total resolution
AFS60_Data.GetData.PresetValue	0	Decimal	DINT	AFS60 / AFM60 data structure Preset value: [0 ...Total measuring
AFS60_Data.GetData.PositionLowLimit	0	Decimal	DINT	AFS60 / AFM60 data structure Lower limit for the position
AFS60_Data.GetData.PositionHighLimit	500	Decimal	DINT	AFS60 / AFM60 data structure Upper limit for the position
AFS60_Data.GetData.VelocityFormat	16#1f0e	Hex	INT	AFS60 / AFM60 data structure Velocity unit: 1F04h = counts/s; 1F
AFS60_Data.GetData.MinVelocitySetpoint	0	Decimal	DINT	AFS60 / AFM60 data structure Minimum velocity. If the velocity dr
AFS60_Data.GetData.MaxVelocitySetpoint	1073741823	Decimal	DINT	AFS60 / AFM60 data structure Maximum velocity. if the velocity ex
AFS60_Data.GetData.Warnings	0	Decimal	INT	AFS60 / AFM60 data structure Bit fiels with flags for warnings (des
AFS60_Data.GetData.WarningFlag	0	Decimal	BOOL	AFS60 / AFM60 data structure Warnings: 0 = No warning; 1 = Wa
AFS60_Data.GetData.TemperatureValue	4300	Decimal	INT	AFS60 / AFM60 data structure Actual temperature with +-5% accu
AFS60_Data.GetData.TemperatureValueFormat	16#1200	Hex	INT	AFS60 / AFM60 data structure Temperature format: 1200h = Celsi
AFS60_Data.GetData.MotionTime	502	Decimal	DINT	AFS60 / AFM60 data structure Saved motion time in seconds (is ir
AFS60_Data.GetData.OperatingTime	538656	Decimal	DINT	AFS60 / AFM60 data structure Saved operating time in seconds (i
AFS60_Data.GetData.MaxVelocity	1009	Decimal	DINT	AFS60 / AFM60 data structure Highest velocity that the encoder h
AFS60_Data.SetData	{...}		SICK_AFX60_WRITE	AFS60 / AFM60 data structure Parameter values that should be w
AFS60_Data.arrReadRecord	{...}	Decimal	SINT[4]	AFS60 / AFM60 data structure Read record for the GetMessage c
AFS60_Data.arrWriteRecord	{...}	Decimal	SINT[4]	AFS60 / AFM60 data structure Write record for the SetMessage c

Abbildung 12: Ausgelesene aller Geräteparameter

Eine steigende Flanke des Ausgangsparameter „bReadDone“ signalisiert, dass die Leseaktion erfolgreich durchgeführt wurde. Die entsprechenden Werte der Geräteparameter werden in der Datenstruktur „AFS60_Data.GetData“ abgelegt.

6.3 Schreiben von Geräteparametern

Über eine steigende Flanke (bStartWrite) werden alle vorausgewählten Parameter auf das Gerät geschrieben. Die Auswahl der zu schreibenden Geräteparameter erfolgt über die Datenstruktur „AFS60_Data.SetData.Selection“ (hier: PresetValue und PositionHighLimit).

Der jeweilige Wert des Geräteparameters wird in der Struktur „AFS60_Data.SetData“ definiert (hier: PresetValue:= 123 und PositionHighLimit:= 800).

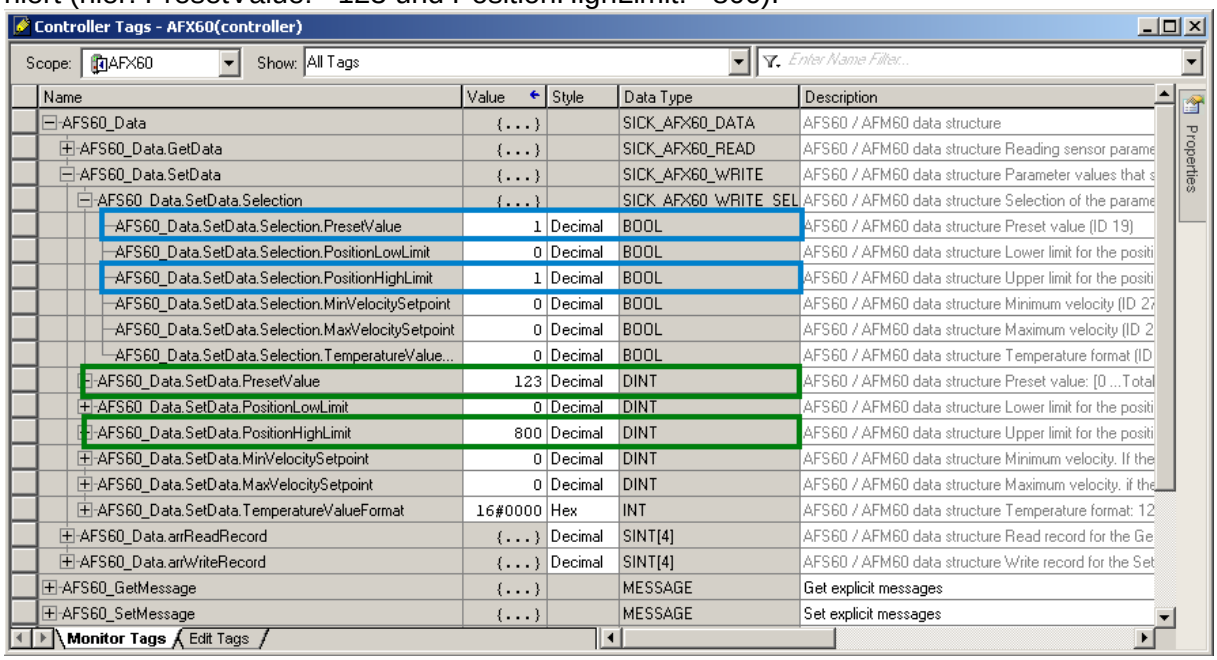


Abbildung 13: Auswahl der zu schreibenden Geräteparameter

Eine steigende Flanke des Ausgangsparameter „bWriteDone“ signalisiert, dass die Schreibaktion erfolgreich durchgeführt wurde.