



ETHERNET/IP SET UP GUIDE

Riccardo Piccinini
Mini Motor S.P.A.
FBS-EIP-EN
Firmware Revision: >4.044

Version 4, 07-2026

Table of Contents

- 1. Ethernet/IP 1
 - 1.1. CIP Objects 1
 - 1.2. Process Data 5
- 2. Application Example 13
 - 2.1. Tools 13
 - 2.2. Overview 13
 - 2.3. DBS Basic Application 13
 - 2.4. AOI CanOpenDS402 Explanation 21
 - 2.5. AOI Rockwell-Like Explanation 25
 - 2.6. Writing Parameters with Async Messaging 28

1. Ethernet/IP



Note: available only for drivers equipped with Ethernet IP optional board (---/---/EIP).

This document outlines the conformance of the Ethernet/IP and CIP stack implemented in the servodrive; the following documents apply unless otherwise stated:

- CIP Network library – Volume 1 (edition 3.3, november 2007)
- CIP Network library – Volume 2; Ethernet/IP adaptation of CIP (edition 1.4, november 2007)
- CiA DS402 v3.0.1.15

Furthermore the CiA DS402 adaptation to CIP are described in the relevant section.

1.1. CIP Objects

1.1.1. Overview

The servodrive implements the following objects of the CIP object model:

- CIP common objects
 - Identity object – 01h
 - Assembly object – 04h
 - TCP object – F5h
 - Ethernet object – F6h;
- Manufacturer specific objects
 - COM object – 0x64
 - PAR object – 0x65
 - MISC object – 0x66
 - CIA402 objects – 0x67

Furthermore the following ASSEMBLIES contain process data:

- O>>T: Assembly 0x64

- T>>O: Assembly 0x65
- CFG: Assembly 0x66

1.1.2. COM object - 0x64

Instance	Attribute ID	Name	Data Type	Unit meas.	Access
0x0	0x01	Revision	UINT		RO
0x1	0x01	Software edition	UINT		RO
	0x02	Software CRC	UDINT		RO
	0x10	Save parameters	UDINT		RO

1.1.3. COM object - 0x65

Instance	Attribute ID	Name	Data Type	Unit meas.	Access
0x0	0x01	Revision	UINT		RO
0x1	0x00–0xFF	P000–P255	UINT		RW
0x2	0x00–0xFF	P256–P511	UINT		RW

Special parameters All 32 Bit parameter are divided in two 16 bits parameters of contiguous numbering for the most significant and least significant bits.

These parameters are:

- P170 - Ip Addr
- P172 - Ip Mask
- P174 - Ip Gateway
- P120-135 - Multipos Ref 1 - 8

1.1.4. Misc objects - 0x66

Instance	Attribute ID	Name	Data Type	Unit meas.	Access
0x0	0x01	Revision	UINT		RO
0x30 (48)	0x00	Analog input 0	UINT		RO
	0x01	Digital inputs	UINT		RO
	0x02	Digital outputs	UINT		RW
	0x03	Heatsink temperature	UINT	0.1 °C	RO
	0x05	Board temperature	UINT	0.1 °C	RO
	0x08	Actual Alarm	UINT	N/A	RO
	0x10	Dclink voltage	UINT	0.1 V	RO
	0x11	Current actual value	UINT	mA	RO
	0x12	Current limit for PP/PV/PT	UINT	mA	RO

1.1.5. CiA 402 Objects - 0x67

Instance	Attr. ID	Name	Data Type	Unit meas.	Access
0x0	0x01	Revision	UINT		RO
0x60 (96)	0x40	Control word	UINT		RW
	0x41	Status word	UINT		RO
	0x5A	Quick-stop option code	INT		RW
	0x5C	Disable operation option code	INT		RW
	0x5E	Fault reaction code	INT		RW
	0x60	Mode of operation	SINT		RW
	0x64	Position actual value	DINT	pulses	RO
	0x65	Following error window	DINT	pulses	RW
	0x66	Following error time	INT	ms	RW
	0x67	Position window	DINT	pulses	RW
	0x68	Position window time	INT	ms	RW
	0x6C	Velocity actual value	DINT	rpm	RO
	0x6D	Velocity window	DINT	rpm	RW
	0x6E	Velocity window time	INT	ms	RW
	0x6F	Velocity threshold	DINT	rpm	RW
	0x70	Velocity threshold time	INT	ms	RW
	0x71	Target torque	INT	thousand-th of rated	RW
	0x72	Max torque	INT	thousand-th of rated	RW
	0x73	Max current	INT	thousand-th of rated	RW
	0x75	Motor rated current	DINT	milliA	RO
	0x76	Motor rated torque	DINT	milliNm	RO
	0x77	Torque actual value	INT	thousand-th of rated	RO
	0x78	Current actual value	INT	thousand-th of rated	RO
	0x7A	Target position	DINT	pulses	RW
	0x7B	Position Range index1: min limit index2: max limit	ARRAY[1..2] OF DINT	pulses	RW
	0x7C	Home offset	DINT	pulses	RW
	0x7D	Position limits index1: min limit index2: max limit	ARRAY[1..2] OF DINT	pulses	RW

Instance	Attr. ID	Name	Data Type	Unit meas.	Access
	0x7E	Polarity	USINT		RW
	0x80	Max motor speed	DINT	rpm	RO
	0x81	Profile velocity	DINT	rpm	RW
	0x83	Profile acceleration	DINT	rpm/s	RW
	0x84	Profile deceleration	DINT	rpm/s	RW
	0x85	Quick stop deceleration	DINT	rpm/s	RW
	0x87	Torque slope	INT	thousandth of rated / s	RW
	0x98	Homing method	SINT		RW
	0x99	Homing speed index1: switch speed index2: index speed	ARRAY[1..2] OF DINT	rpm	RW
	0x9A	Homing acceleration	DINT	rpm/s	RW
	0xF2	Profile Option Code	INT		RW
	0xFF	Target velocity	DINT	rpm	RW

1.2. Process Data

_Valid from Firmware ≥ v4.036 _ There are four possible configurations of the Ethernet IP process data.

- **Default (DFL):** the standard process data implementing the profile velocity/torque/position/homing
- **A:** Standard process data with the possibility to read Digital and Analog inputs and command via fieldbus the Digital Out. The latter needs the parameter P015 on 10-Fieldbus to receive commands.
- **B:** A process data with the CiA 402 Touch probe functionality
- **C:** A process data with our manufacturer custom Touch Probe implementation
- **D:** A process data with acceleration three axis rms values

You must select the corresponding device description and set the parameter P176 to the desired value.

1.2.1. Layout Default (DFL)

The configuration for process data is the following:

Assembly 0x64 (Originator to Target, DBS/FC inputs - PLC outputs)

Assembly Length : 50 Bytes

Offset	Type	Attribute	Class	Instance	Attr. ID	Default	Meaning
0x00	U16	ControlWord	0x67	0x60	0x40	0000h	see DSP402
0x02	S32	TargetPosition	0x67	0x60	0x7A	+0	position units
0x06	U16	TorqueLimit	0x67	0x60	0x72	1000	× 0.1% Trq

Offset	Type	Attribute	Class	Instance	Attr. ID	Default	Meaning
0x08	S32	TargetVelocity	0x67	0x60	0xFF	+0	rpm
0x0C	S16	TargetTorque	0x67	0x60	0x71	+0	× 0.1% Trq
0x0E	U8	ModeOfOperation	0x67	0x60	0x60	1	Profile ^[1]
0x0F	U8	Padding Byte					
0x10	U32	ProfileVelocity	0x67	0x60	0x81	+3000	rpm
0x14	U32	ProfileAccel	0x67	0x60	0x83	+3000	rpm/s
0x18	U32	ProfileDecel	0x67	0x60	0x84	+3000	rpm/s
0x1C	S32	PositionLimit.Neg	0x67	0x60	0x7D.1	-2147483648	position units
0x20	S32	PositionLimit.Pos	0x67	0x60	0x7D.2	+2147483647	position units
0x24	S32	HomingOffset	0x67	0x60	0x7C	+0	position units
0x28	S8	HomingMethod	0x67	0x60	0x98	37	SetQuota ^[1]

Assembly 0x65 (Target to Originator, DBS/FC outputs - PLC inputs)

Assembly Length : 20 Bytes

Offset	Type	Attribute	Class	Instance	Attr. ID	Default	Meaning
0x00	U16	StatusWord	0x67	0x60	0x41	0250h	Motor Status ^[1]
0x02	S32	PositionActualValue	0x67	0x60	0x64	+0	position units
0x06	S16	TorqueActualValue	0x67	0x60	0x77	+0	× 0.1% Trq
0x08	S32	VelocityActualValue	0x67	0x60	0x6C	+0	rpm
0x0C	U16	VDclink	0x67	0x30	0x10	+0	× 0.1 V
0x0E	U8	ModeOfOperation	0x67	0x60	0x60	1	Profile ^[1]
0x0F	U8	Padding Byte					
0x10	U16	ActualAlarm	0x67	0x30	0x08	0	see DBS Manual

1.2.2. Layout A

The configuration for process data is the following:

Assembly 0x64 (Originator to Target, DBS/FC inputs - PLC outputs)

Assembly Length : 50 Bytes

Offset	Type	Attribute	Class	Instance	Attr. ID	Default	Meaning
0x00	U16	ControlWord	0x67	0x60	0x40	0000h	see DSP402
0x02	S32	TargetPosition	0x67	0x60	0x7A	+0	position units
0x06	U16	TorqueLimit	0x67	0x60	0x72	1000	× 0.1% Trq
0x08	S32	TargetVelocity	0x67	0x60	0xFF	+0	rpm
0x0C	S16	TargetTorque	0x67	0x60	0x71	+0	× 0.1% Trq
0x0E	U8	ModeOfOperation	0x67	0x60	0x60	1	Profile ^[1]
0x0F	U8	Padding Byte					
0x10	U32	ProfileVelocity	0x67	0x60	0x81	+3000	rpm
0x14	U32	ProfileAccel	0x67	0x60	0x83	+3000	rpm/s
0x18	U32	ProfileDecel	0x67	0x60	0x84	+3000	rpm/s
0x1C	S32	PositionLimit.Neg	0x67	0x60	0x7D.1	-2147483648	position units
0x20	S32	PositionLimit.Pos	0x67	0x60	0x7D.2	+2147483647	position units
0x24	S32	HomingOffset	0x67	0x60	0x7C	+0	position units
0x28	S8	HomingMethod	0x67	0x60	0x98	37	SetQuota ^[1]
0x29	U8	Padding Byte					
0x2A	U16	Digital output	0x67	0x30	0x02	0	bit0: DOUT1 bit1..15: not used ^[2]

Assembly 0x65 (Target to Originator, DBS/FC outputs - PLC inputs)

Assembly Length : 30 Bytes

Offset	Type	Attribute	Class	Instance	Attr. ID	Default	Meaning
0x00	U16	StatusWord	0x67	0x60	0x41	0250h	Motor Status ^[1]
0x02	S32	PositionActualValue	0x67	0x60	0x64	+0	position units
0x06	S16	TorqueActualValue	0x67	0x60	0x77	+0	× 0.1% Trq
0x08	S32	VelocityActualValue	0x67	0x60	0x6C	+0	rpm
0x0C	U16	VDclink	0x67	0x30	0x10	+0	× 0.1 V
0x0E	U8	ModeOfOperation	0x67	0x60	0x60	1	Profile ^[1]

Offset	Type	Attribute	Class	Instance	Attr. ID	Default	Meaning
0x0F	U8	Padding Byte					
0x10	U16	ActualAlarm	0x67	0x30	0x08	0	see DBS Manual
0x12	S16	Analog Input	0x67	0x30	0x00	0	-32768 ... +32767
0x14	U16	Digital Input	0x67	0x30	0x01	0	Bit0: DIN1 Bit1: DIN2 Bit2: DIN3 Bit3: DIN4 Bit4: DIN5 Bit5..15: not used

1.2.3. Layout B

The configuration for process data is the following:

Assembly 0x64 (Originator to Target, DBS/FC inputs - PLC outputs)

Assembly Length : 50 Bytes

Offset	Type	Attribute	Class	Instance	Attr. ID	Default	Meaning
0x00	U16	ControlWord	0x67	0x60	0x40	0000h	see DSP402
0x02	S32	TargetPosition	0x67	0x60	0x7A	+0	position units
0x06	U16	TorqueLimit	0x67	0x60	0x72	1000	× 0.1% Trq
0x08	S32	TargetVelocity	0x67	0x60	0xFF	+0	rpm
0x0C	S16	TargetTorque	0x67	0x60	0x71	+0	× 0.1% Trq
0x0E	U8	ModeOfOperation	0x67	0x60	0x60	1	Profile ^[1]
0x0F	U8	Padding Byte					
0x10	U32	ProfileVelocity	0x67	0x60	0x81	+3000	rpm
0x14	U32	ProfileAccel	0x67	0x60	0x83	+3000	rpm/s
0x18	U32	ProfileDecel	0x67	0x60	0x84	+3000	rpm/s
0x1C	S32	PositionLimit.Neg	0x67	0x60	0x7D.1	-2147483648	position units
0x20	S32	PositionLimit.Pos	0x67	0x60	0x7D.2	+2147483647	position units
0x24	S32	HomingOffset	0x67	0x60	0x7C	+0	position units
0x28	S8	HomingMethod	0x67	0x60	0x98	37	SetQuota ^[1]

Offset	Type	Attribute	Class	Instance	Attr. ID	Default	Meaning
0x29	U8	Padding Byte					
0x2A	U16	Digital output	0x67	0x30	0x02	0	bit0: DOUT1 bit1..15: not used ^[2]
0x2C	U16	Touch Probe Function	0x67	0x60	0xB8	0	^[1]

Assembly 0x65 (Target to Originator, DBS/FC outputs - PLC inputs)

Assembly Length : 40 Bytes

Offset	Type	Attribute	Class	Instance	Attr. ID	Default	Meaning
0x00	U16	StatusWord	0x67	0x60	0x41	0250h	Motor Status ^[1]
0x02	S32	PositionActualValue	0x67	0x60	0x64	+0	position units
0x06	S16	TorqueActualValue	0x67	0x60	0x77	+0	× 0.1% Trq
0x08	S32	VelocityActualValue	0x67	0x60	0x6C	+0	rpm
0x0C	U16	VDclink	0x67	0x30	0x10	+0	× 0.1 V
0x0E	U8	ModeOfOperation	0x67	0x60	0x60	1	Profile ^[1]
0x0F	U8	Padding Byte					
0x10	U16	ActualAlarm	0x67	0x30	0x08	0	see DBS Manual
0x12	S16	Analog Input	0x67	0x30	0x00	0	-32768 ... +32767
0x14	U16	Digital Input	0x67	0x30	0x01	0	Bit0: DIN1 Bit1: DIN2 Bit2: DIN3 Bit3: DIN4 Bit4: DIN5 Bit5..15: not used
0x16	U16	Touch Probe Status	0x67	0x60	0xB9	0	^[1]
0x18	S32	Touch Probe Rising Edge position	0x67	0x60	0xBA	0	position units
0x1C	S32	Touch Probe Falling Edge position	0x67	0x60	0xBB	0	position units

1.2.4. Layout C

The configuration for process data is the following:

Assembly 0x64 (Originator to Target, DBS/FC inputs - PLC outputs)

Assembly Length : 60 Bytes

Offset	Type	Attribute	Class	Instance	Attr. ID	Default	Meaning
0x00	U16	ControlWord	0x67	0x60	0x40	0000h	see DSP402

Offset	Type	Attribute	Class	Instance	Attr. ID	Default	Meaning
0x02	S32	TargetPosition	0x67	0x60	0x7A	+0	position units
0x06	U16	TorqueLimit	0x67	0x60	0x72	1000	× 0.1% Trq
0x08	S32	TargetVelocity	0x67	0x60	0xFF	+0	rpm
0x0C	S16	TargetTorque	0x67	0x60	0x71	+0	× 0.1% Trq
0x0E	U8	ModeOfOperation	0x67	0x60	0x60	1	Profile ^[1]
0x0F	U8	Padding Byte					
0x10	U32	ProfileVelocity	0x67	0x60	0x81	+3000	rpm
0x14	U32	ProfileAccel	0x67	0x60	0x83	+3000	rpm/s
0x18	U32	ProfileDecel	0x67	0x60	0x84	+3000	rpm/s
0x1C	S32	PositionLimit.Neg	0x67	0x60	0x7D.1	-2147483648	position units
0x20	S32	PositionLimit.Pos	0x67	0x60	0x7D.2	+2147483647	position units
0x24	S32	HomingOffset	0x67	0x60	0x7C	+0	position units
0x28	S8	HomingMethod	0x67	0x60	0x98	37	SetQuota ^[1]
0x29	U8	Padding Byte					
0x2A	U16	Digital output	0x67	0x30	0x02	0	bit0: DOUT1 bit1..15: not used ^[2]
0x2C	S32	Touch Probe positioning offset	0x67	0x30	0x20	0	Position Units
0x2C	S32	Touch Probe positioning modulo	0x67	0x30	0x21	0	Position Units

Assembly 0x65 (Target to Originator, DBS/FC outputs - PLC inputs)

Assembly Length : 30 Bytes

Offset	Type	Attribute	Class	Instance	Attr. ID	Default	Meaning
0x00	U16	StatusWord	0x67	0x60	0x41	0250h	Motor Status ^[1]
0x02	S32	PositionActualValue	0x67	0x60	0x64	+0	position units
0x06	S16	TorqueActualValue	0x67	0x60	0x77	+0	× 0.1% Trq

Offset	Type	Attribute	Class	Instance	Attr. ID	Default	Meaning
0x08	S32	VelocityActualValue	0x67	0x60	0x6C	+0	rpm
0x0C	U16	VDclink	0x67	0x30	0x10	+0	× 0.1 V
0x0E	U8	ModeOfOperation	0x67	0x60	0x60	1	Profile ^[1]
0x0F	U8	Padding Byte					
0x10	U16	ActualAlarm	0x67	0x30	0x08	0	see DBS Manual
0x12	S16	Analog Input	0x67	0x30	0x00	0	-32768 ... +32767
0x14	U16	Digital Input	0x67	0x30	0x01	0	Bit0: DIN1 Bit1: DIN2 Bit2: DIN3 Bit3: DIN4 Bit4: DIN5 Bit5..15: not used

1.2.5. Layout D

The configuration for process data is the following:

Assembly 0x64 (Originator to Target, DBS/FC inputs - PLC outputs)

Assembly Length : 50 Bytes

Offset	Type	Attribute	Class	Instance	Attr. ID	Default	Meaning
0x00	U16	ControlWord	0x67	0x60	0x40	0000h	see DSP402
0x02	S32	TargetPosition	0x67	0x60	0x7A	+0	position units
0x06	U16	TorqueLimit	0x67	0x60	0x72	1000	× 0.1% Trq
0x08	S32	TargetVelocity	0x67	0x60	0xFF	+0	rpm
0x0C	S16	TargetTorque	0x67	0x60	0x71	+0	× 0.1% Trq
0x0E	U8	ModeOfOperation	0x67	0x60	0x60	1	Profile ^[1]
0x0F	U8	Padding Byte					
0x10	U32	ProfileVelocity	0x67	0x60	0x81	+3000	rpm
0x14	U32	ProfileAccel	0x67	0x60	0x83	+3000	rpm/s
0x18	U32	ProfileDecel	0x67	0x60	0x84	+3000	rpm/s
0x1C	S32	PositionLimit.Neg	0x67	0x60	0x7D.1	-2147483648	position units
0x20	S32	PositionLimit.Pos	0x67	0x60	0x7D.2	+2147483647	position units

Offset	Type	Attribute	Class	Instance	Attr. ID	Default	Meaning
0x24	S32	HomingOffset	0x67	0x60	0x7C	+0	position units
0x28	S8	HomingMethod	0x67	0x60	0x98	37	SetQuota ^[1]
0x29	U8	Padding Byte					
0x2A	U16	Digital output	0x67	0x30	0x02	0	bit0: DOUT1 bit1..15: not used ^[2]

Assembly 0x65 (Target to Originator, DBS/FC outputs - PLC inputs)

Assembly Length : 30 Bytes

Offset	Type	Attribute	Class	Instance	Attr. ID	Default	Meaning
0x00	U16	StatusWord	0x67	0x60	0x41	0250h	Motor Status ^[1]
0x02	S32	PositionActualValue	0x67	0x60	0x64	+0	position units
0x06	S16	TorqueActualValue	0x67	0x60	0x77	+0	× 0.1% Trq
0x08	S32	VelocityActualValue	0x67	0x60	0x6C	+0	rpm
0x0C	U16	VDclink	0x67	0x30	0x10	+0	× 0.1 V
0x0E	U8	ModeOfOperation	0x67	0x60	0x60	1	Profile ^[1]
0x0F	U8	Padding Byte					
0x10	U16	ActualAlarm	0x67	0x30	0x08	0	see DBS Manual
0x12	S16	Analog Input	0x67	0x30	0x00	0	-32768 ... +32767
0x14	U16	Digital Input	0x67	0x30	0x01	0	Bit0: DIN1 Bit1: DIN2 Bit2: DIN3 Bit3: DIN4 Bit4: DIN5 Bit5..15: not used
0x16	U16	Acceleration rms	0x67	0x30	0x31		
0x19	U16	Acceleration rms	0x67	0x30	0x32		
0x1A	U16	Acceleration rms	0x67	0x30	0x33		

[1] see DSP402

[2] P015=10 required for fieldbus operation

2. Application Example

2.1. Tools

Program Example (no AOI)

Program Example (AOI, DS402)

Program Example (AOI, Rockwell-like)

DBS/FC BSI Interface Software

AOI Download (DS402)

AOI Download (Rockwell-like)

2.2. Overview

This document shows the creation of a DBS55 smart motor application using the Ethernet/IP communication interface; a Rockwell CompactLogix controller is configured and an application is written into the Studio 5000 development environment.

References:

- CompactLogix L18ERM
- DBS/FC Firmware version 4.044
- Studio 5000, version 31.01.00

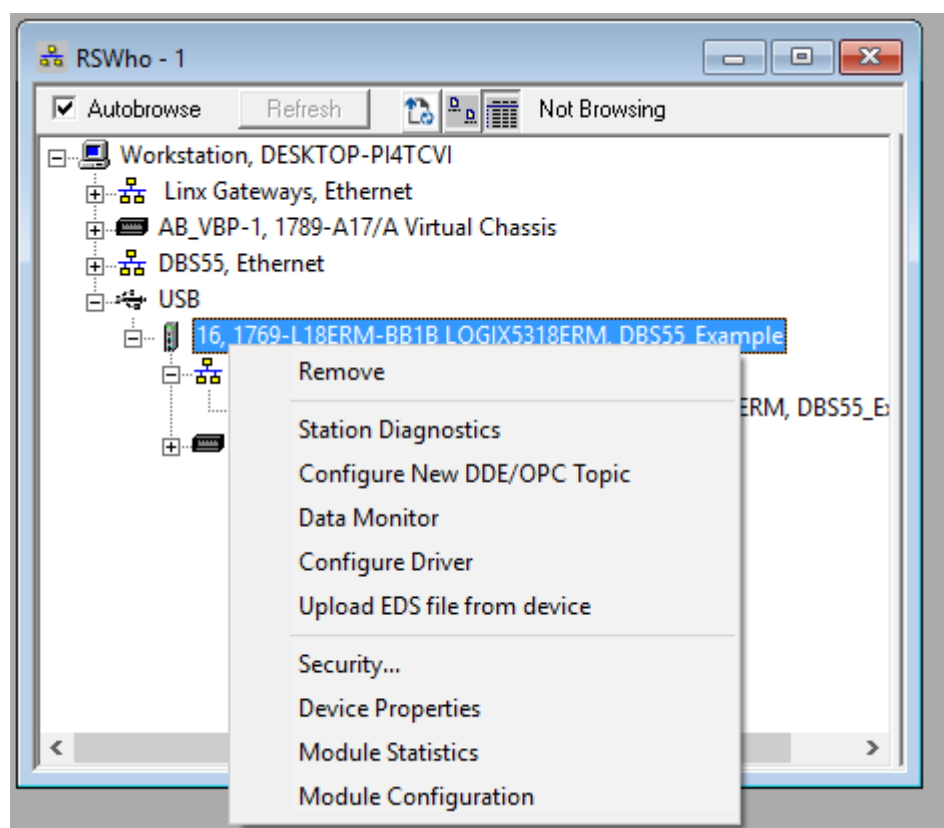


2.3. DBS Basic Application

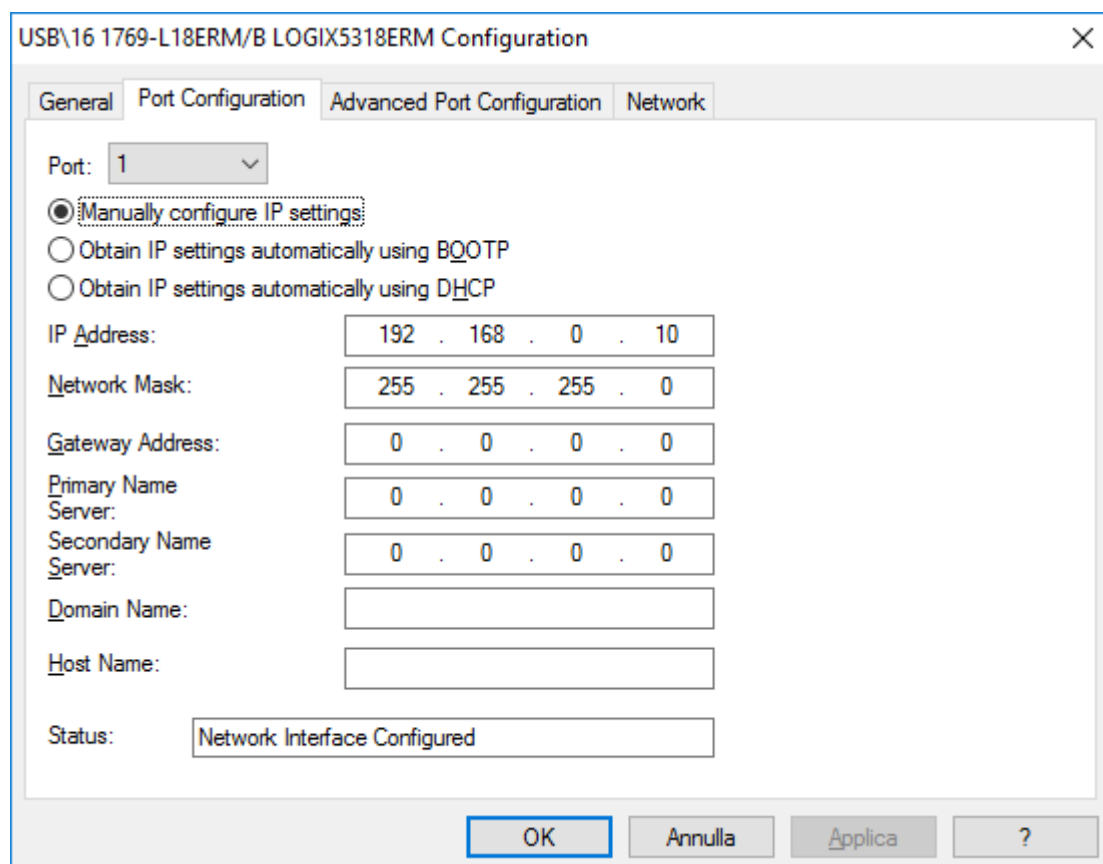
2.3.1. Configure Controller Address

- To set the controller address, start RSLinx

- Select Communication → RSWho and open the controller node (in the example the controller is connected via USB); then right click and select *Module Configuration*

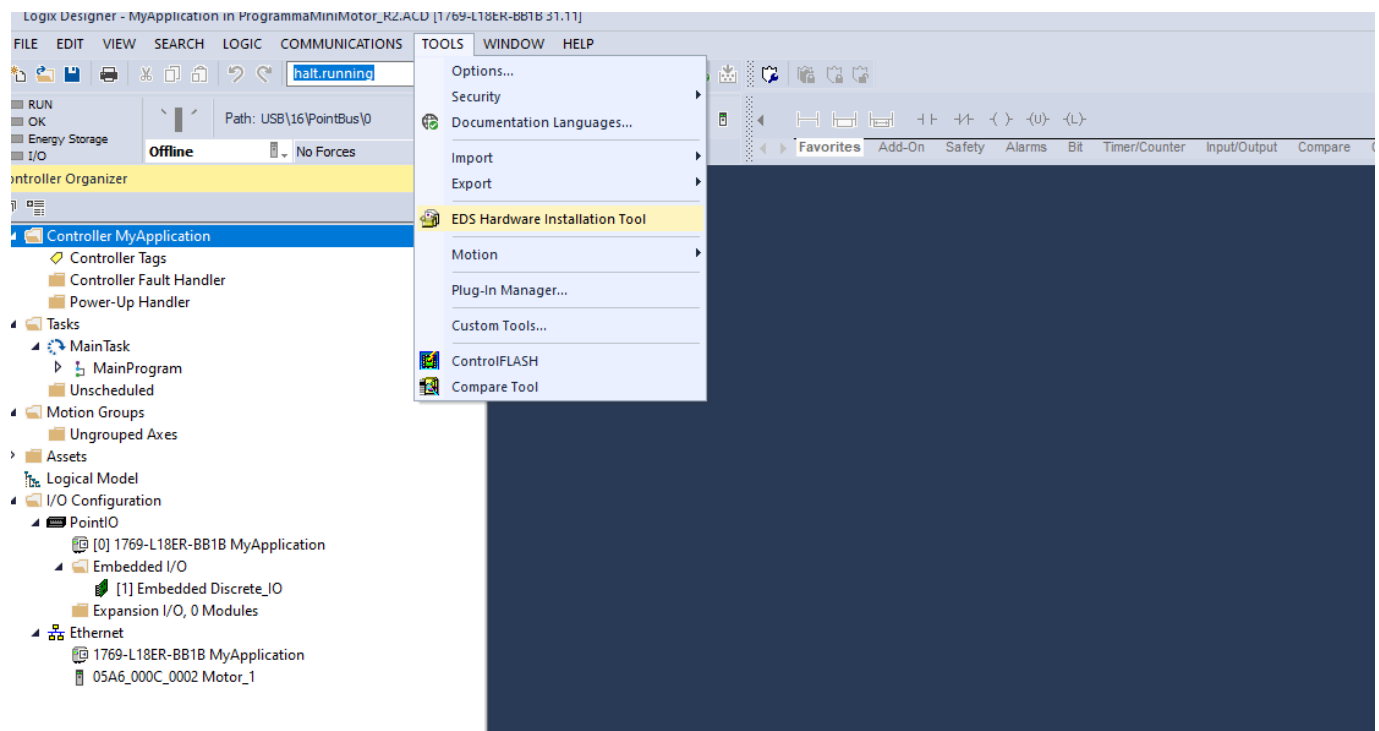


- In the configuration dialog, select *Port Configuration* (1), then *Manually configure IP settings* (2) and insert address and mask (3); in the example 192.168.0.10 is set
- Confirm with OK.

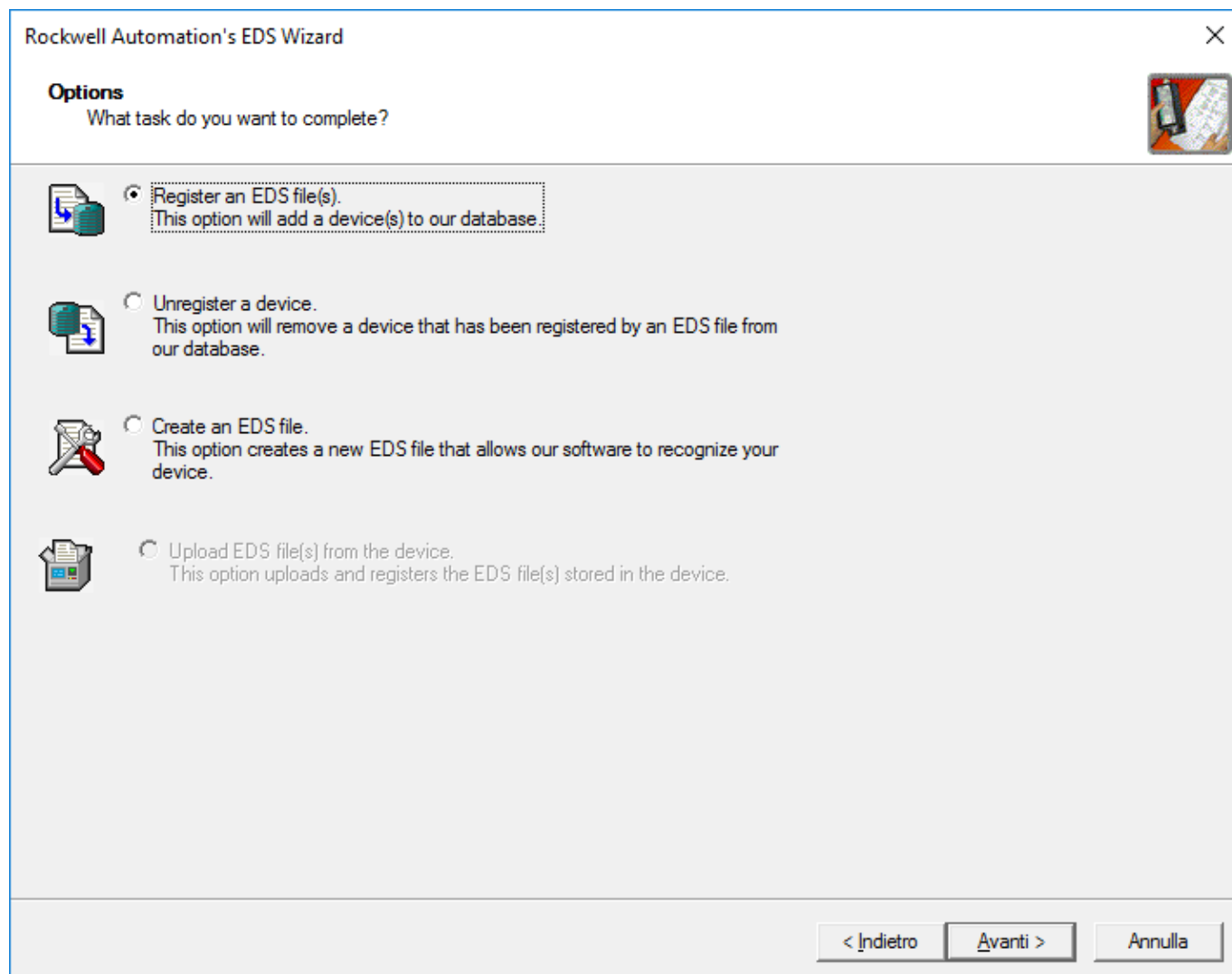


2.3.2. Install EDS

- In Studio 5000 click on Tools → EDS Hardware Installation Tool



- Register an EDS file



- Browse
- Open the BSI software revision\DeviceDesc\EthernetIP folder and select the EIP layout you desire. For this example we will select DBS55_CIP_dfl.eds. Install only the one you want to use.

Condividi Visualizza

MM-DBS_v4.041 > DeviceDesc > EthernetIP

Nome	Ultima modifica	Tipo	Dimensione
DBS55_CIP.eds	21/11/2022 16:31	File EDS	6 KB
DBS55_CIP_dfl.eds	21/11/2022 16:31	File EDS	21 KB
DBS55_CIP_ext-a.eds	21/11/2022 16:31	File EDS	23 KB
DBS55_CIP_ext-b.eds	21/11/2022 16:31	File EDS	26 KB
DBS55_CIP_ext-c.eds	21/11/2022 16:31	File EDS	27 KB
DBS55_CIP_ext-d.eds	21/11/2022 16:31	File EDS	30 KB
Readme.txt	21/11/2022 16:31	Documento di testo	1 KB

Rockwell Automation's EDS Wizard
✕

Registration


Electronic Data Sheet file(s) will be added to your system for use in Rockwell Automation applications.

☒ Register a single file
☐ Register a directory of EDS files
 ☐ Look in subfolders

Named:

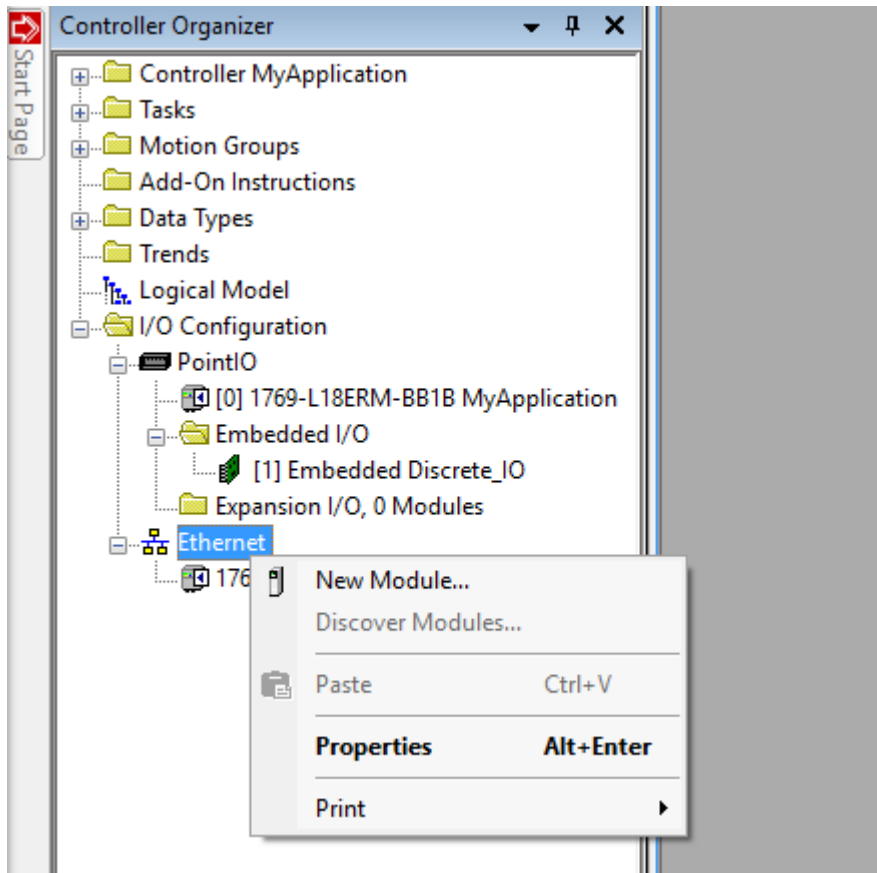

 * If there is an icon file (.ico) with the same name as the file(s) you are registering then this image will be associated with the device.

To perform an installation test on the file(s), click Next

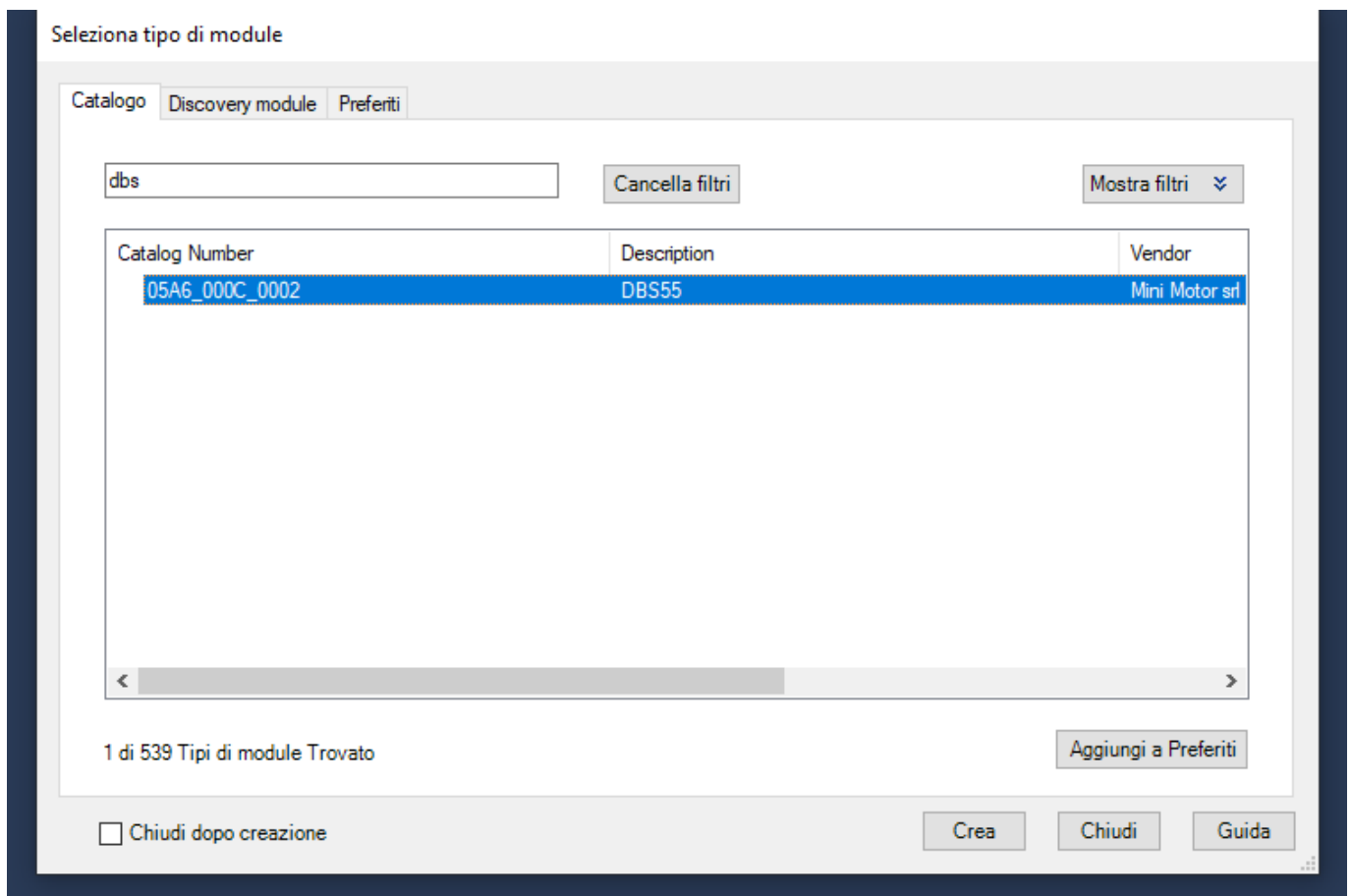
- Next →, Next →, Next →; click on Finish to complete the insertion.

2.3.3. Add DBS Slave to a Project

- Right click on *I/O Configuration > Ethernet* and select *New Module*



- Type `db5` in the filter and select `05A6_000C_0002 – DBS55`, then click on Create



- In the *New Module* window enter the device name and the IP address of the device, then finally click on

Change.



The IP address of the device should be in the same subnet as the master IP address; the IP address of the slave is configured by the BSI configuration software (see DBS/FC manual).

New Module

General

Type: DBS55
Vendor: Mini Motor srl
Parent: Local
Name: Motor_1
Description:

Ethernet Address
☐ Private Network: 192.168.1.
☒ IP Address: 192 . 168 . 0 . 11
☐ Host Name:

Module Definition
Revision: 2.001
Electronic Keying: Compatible Module
Connections: Exclusive Owner
Change ...

Status: Creating

OK Cancel Help

- In the module definition window, in the *Electronic Keying* field select *Disable Keying*
- Confirm the module definition dialog and finally click on Create, then close the dialog

Module Definition

Revision: 2.001
Electronic Keying: Disable Keying
Connections: Exclusive Owner
Change ...

Module Definition

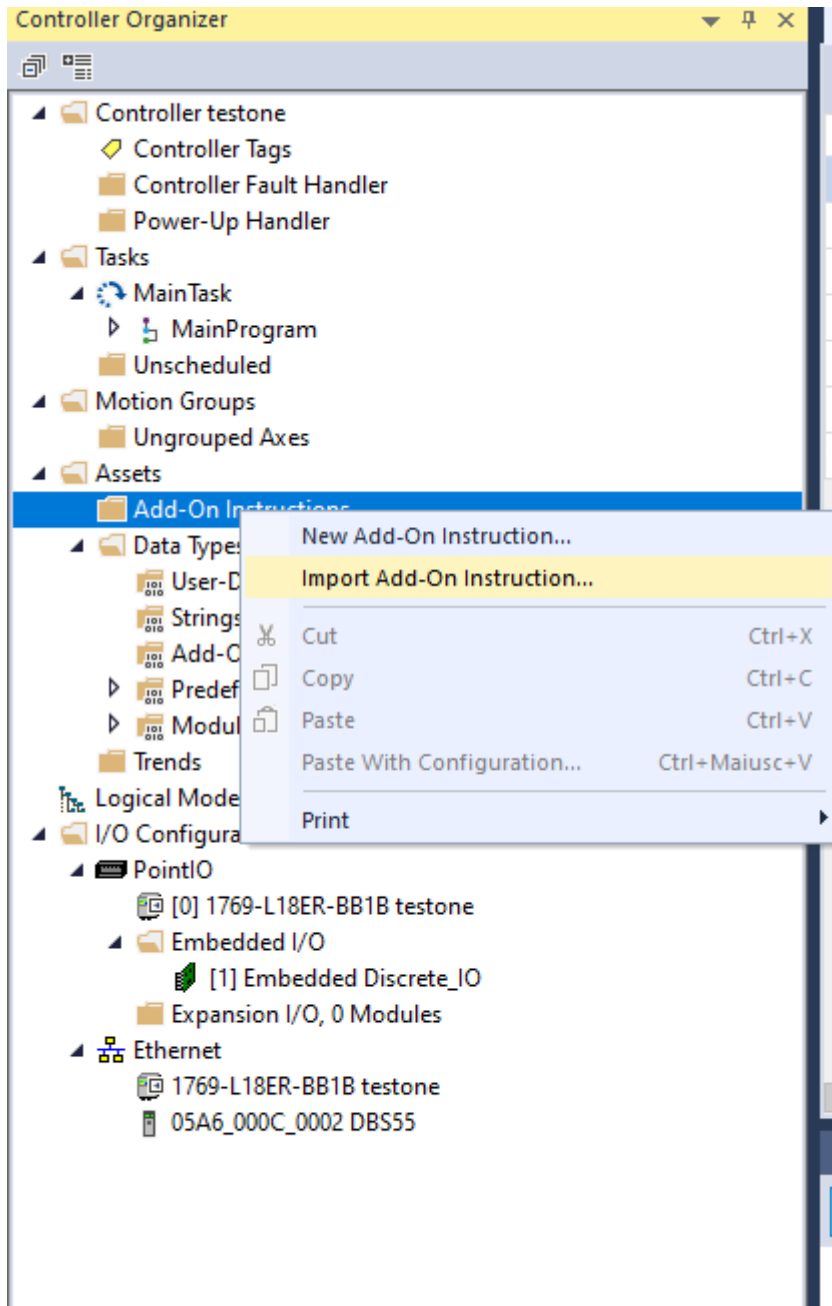
Revision: 2 001
Electronic Keying: Disable Keying
Connections:

Name	Size
Exclusive Owner	Input: 20 Output: 50 SINT

OK Cancel Help

2.3.4. AOI Import

- Import the DBS AOI you need for your project. Go to *Assets* → *Add-On Instructions* → *Right Click* → *select which AOI to import*.



- After having imported an AOI you should find two User Defined Data Types.
- Create for each motor a variable of both types.

Controller Tags - testone(controller) x						
Scope:  testone		Show: All Tags				
Name	Alias For	Base Tag	Data Type	Description		
▶ Local:1:O			AB:Embedded_DiscreteIO:O:0			R
▶ Local:1:I			AB:Embedded_DiscreteIO:I:0			R
▶ Local:1:C			AB:Embedded_DiscreteIO:C:0			R
▶ DBS1_In			DBS55_In	PdoRX		R
▶ DBS1_out			DBS55_Out	PdoTX		R

2.4. AOI CanOpenDS402 Explanation

There are 7 AOI that can be used for DBS and FC systems. We will look at them here.

2.4.1. MM_ReadData_Dfl

This instruction should be the first and its job is to take the motor assembly data and assign it to a DBS55_Out structure. Each Mini Motor Integrated Drive system creates a DB13776:I:0 data type that must be assigned to a specific MM_ReadData_Dfl AOI. Then each data is passed to a specific DBS55_Out variable. This structure will be used in all operative blocks.

LocalStructure

Motor_ReadData

Motor_ReadData DBS_In

MotorRead Motor_1:I

MotorInStruc

Enter Name Filter...

Show: All Tags

Name	Data Type	Usage	Description
▶ Motor_1:I	_05A6:000C_0002_DB13776:I:0	<controller>	

MotorInStructure Motor_1_Out

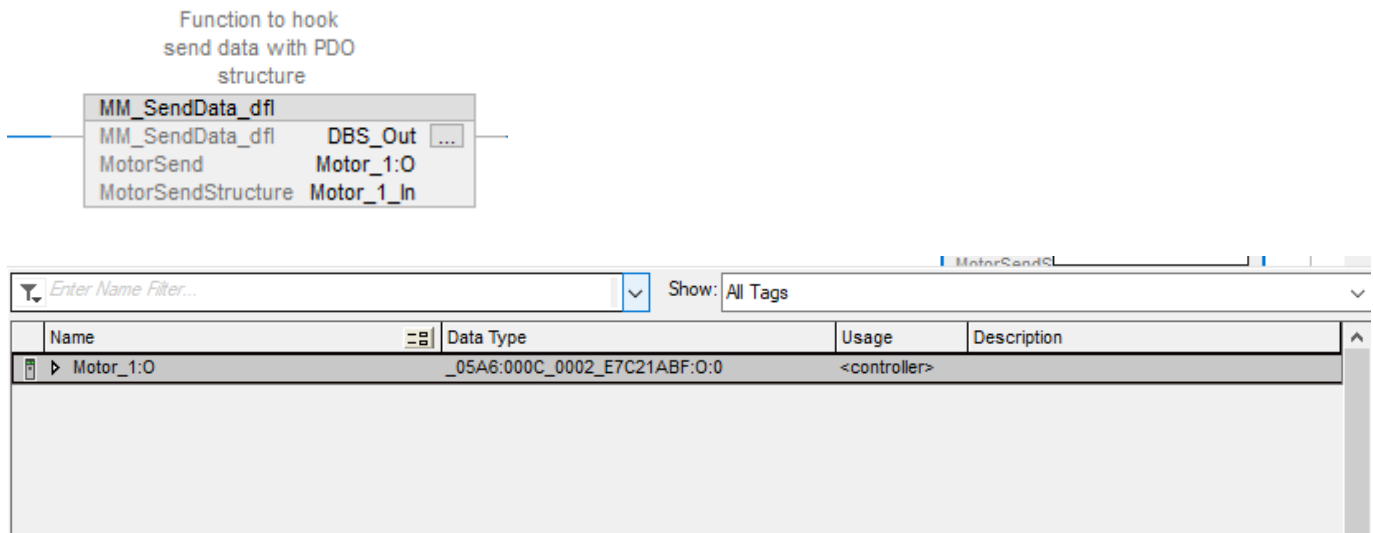
Enter Name Filter...

Show: All Tags

Name	Data Type	Usage	Description
▶ Motor_1_Out	DBS55_Out	<controller>	PdoTX

2.4.2. MM_SendData_Dfl

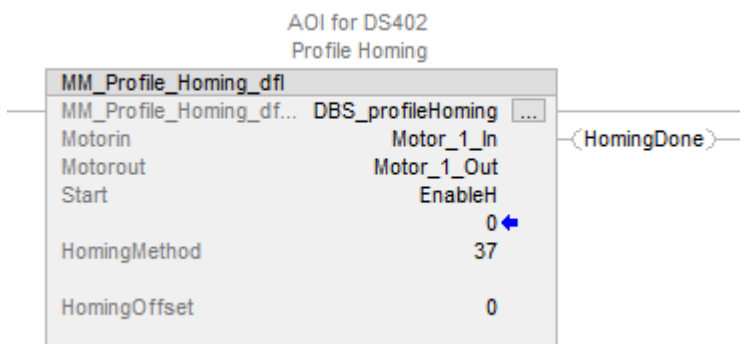
This instruction should be the last and its job is to move data from a DBS55_In data type to the motor data inlet. Each Mini Motor Integrated Drive system creates an E7C21ABF:O:0 data type that must be assigned to a specific MM_SendData_Dfl AOI. In this way the data manipulated by the rest of the blocks is passed from the DBS55_In to the motor.



2.4.3. MM_Profile_Homing_Dfl

With the MM_Profile_Homing_Dfl you can home the Integrated Drive family following the DS402 homing procedure as explained in the main DBS/FC manual.

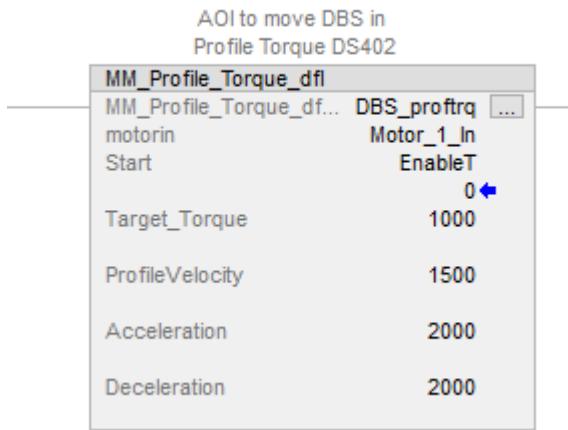
The *Motorin* and *Motorout* are hooked to the DBS55_In and DBS55_Out data types of the specific motor, while the *Homing Method* is the homing procedure selector and *Homing Offset* is the offset to add once the procedure is done. A single bit signals the completion of the homing procedure. This procedure can be started with a *Start* BOOLEAN value.



2.4.4. MM_Profile_Torque

The motor can be used in DS402 Profile Torque mode, useful when you want to have better torque control in applications where speed is secondary.

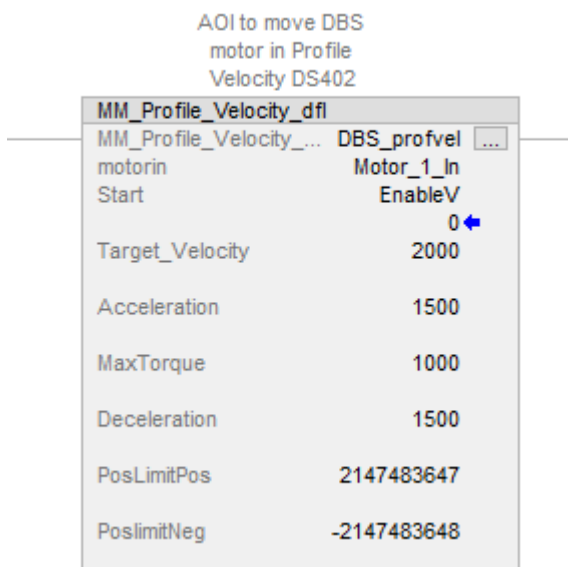
The AOI must be hooked to a DBS55_In data type of the motor to send commands. A BOOL *Start* makes the motor turn. You can set the maximum torque the motor can use with *Target_Torque* and constrain the trajectory with the acceleration and deceleration.



2.4.5. MM_Profile_Velocity_Dfl

The motor can be used in DS402 Profile Velocity mode, enabling the motor to reach and maintain a specific target speed.

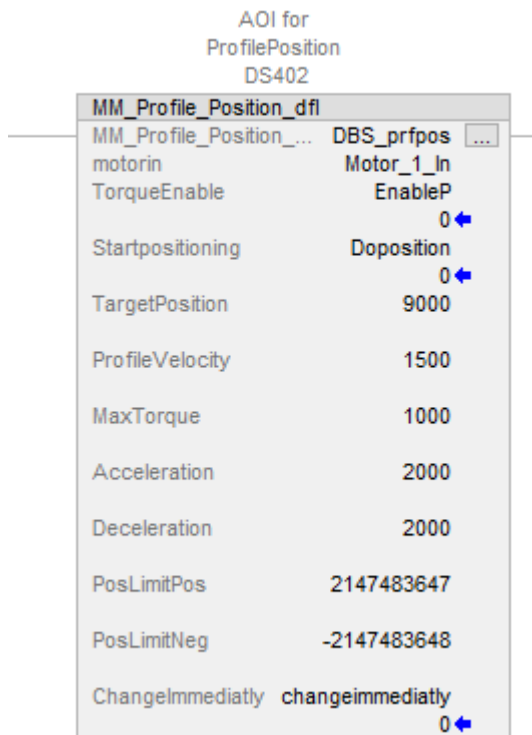
The AOI must be hooked to a DBS55_In data type of the motor to send commands. A BOOL *Start* makes the motor turn. You can set the target speed and constrain the trajectory with the other parameters. If the position limits are at their maximum value (2147483647 and -2147483648 respectively) then they are disabled. If they are both at 0 the motor will not move.



2.4.6. MM_Profile_Position_Dfl

The motor can be commanded to reach specific positions using its integrated absolute multiturn encoder.

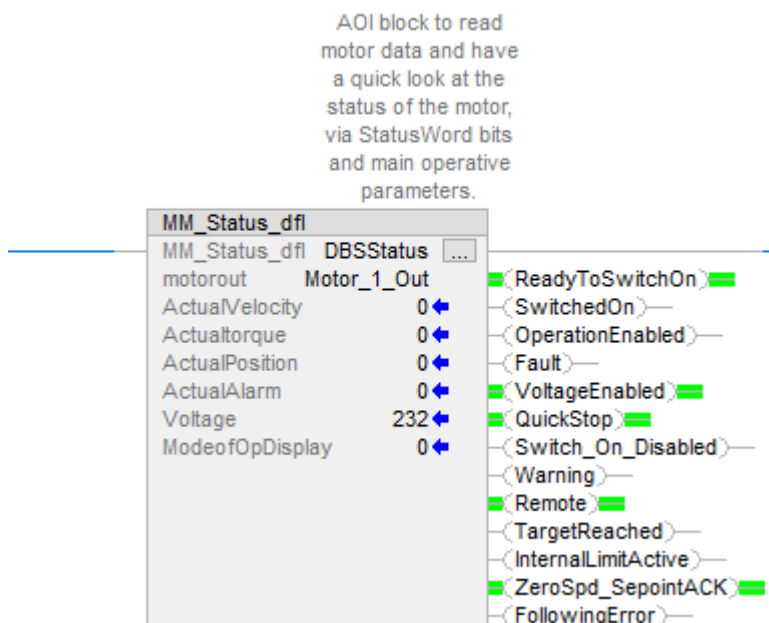
The AOI must be hooked to a DBS55_In data type of the motor to send commands. The AOI has two separate commands to start the motor. *TorqueEnable* makes the motor enabled in standstill, keeping the actual position. *Startpositioning* is a rising edge command that makes the motor start its positioning. The trajectory can be constrained via the parameters shown (Velocity, Deceleration, Acceleration, Position limits and max torque). The *ChangeImmediately* command is used to make the new position command immediately execute, discarding the current motion.



2.4.7. MM_Status_Dfl

This block can be used to have quick access to the motor status, hooking a DBS55_Out structure to it.

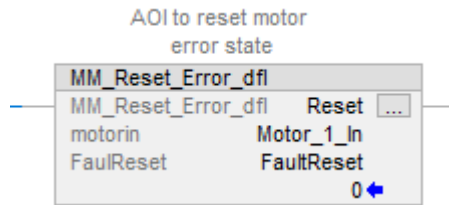
It shows the motor speed, actual torque, actual position, actual alarm and voltage. It also shows the bits of the StatusWord that can be used to power internal logics or diagnostics. It gives the feedback for the Target Reached and other behaviours for the various operating modes.



2.4.8. MM_Reset_Error_Dfl

This AOI is used to reset the error status on the motor. It is hooked to a DBS55_In instance to send the

command. The Fault status can be seen on the MM_Status_Dfl block.

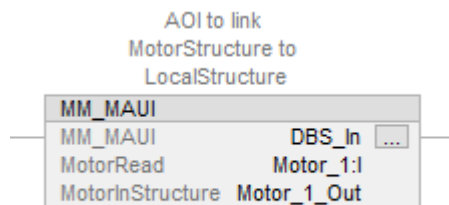


2.5. AOI Rockwell-Like Explanation

These AOIs follow a different nomenclature and set of triggers, translating the front end of the DS402 command protocol to those more familiar with Rockwell operated drives.

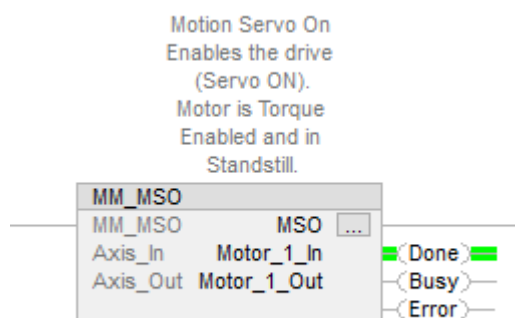
2.5.1. MM_MAUl

This block links the MotorStructure of data exchange to the local structure of data exchange.



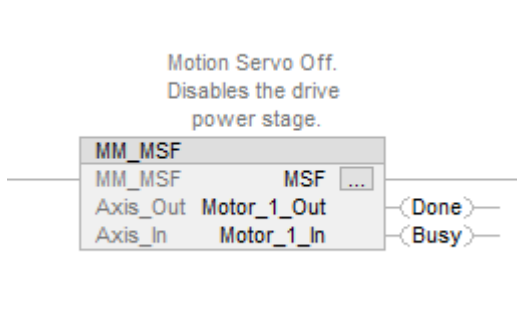
2.5.2. MM_MSO

Motion Servo On. This block puts the motor in standstill with torque enabled.



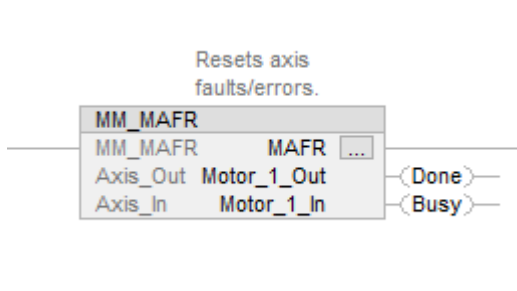
2.5.3. MM_MSF

Motion Servo Off. This block disables the power stage.



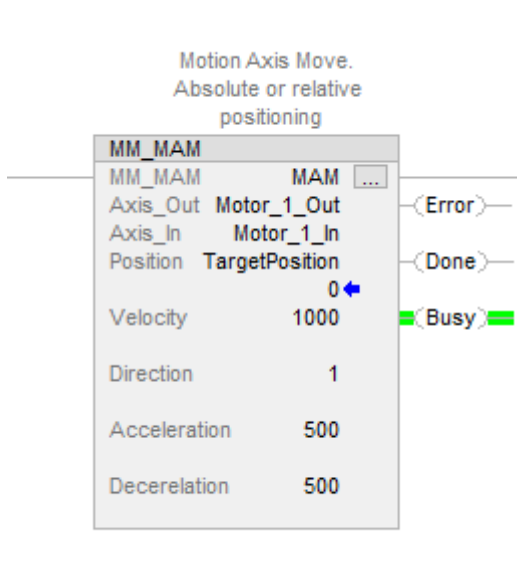
2.5.4. MM_MAFR

This block is used for the reset of errors on the drive. Check the main servo manual to see more information about alarms and conditions.



2.5.5. MM_MAM

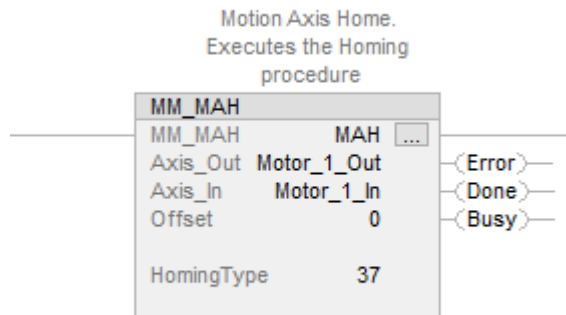
Motion Axis Move. This block moves the motor to a specific position. The trajectory of the motor is built via Acceleration, Deceleration, Velocity and Target Position, and it will always be a trapezoidal trajectory. Velocity is in rpm, acceleration and deceleration are in rpm/s, position is in encoder pulses or whatever scaled position you are using (see the main manual for how to scale). *Direction* changes between absolute (0) and relative (1) positioning.



2.5.6. MM_MAH

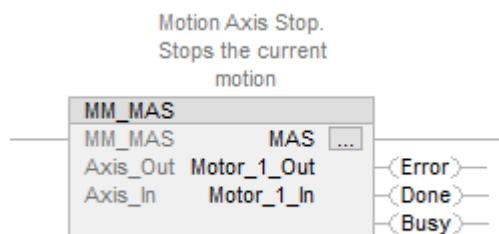
Motion Axis Home. Block used for homing. Check the main manual for all homing modes. Homing mode 37,

the default one, is homing in place.



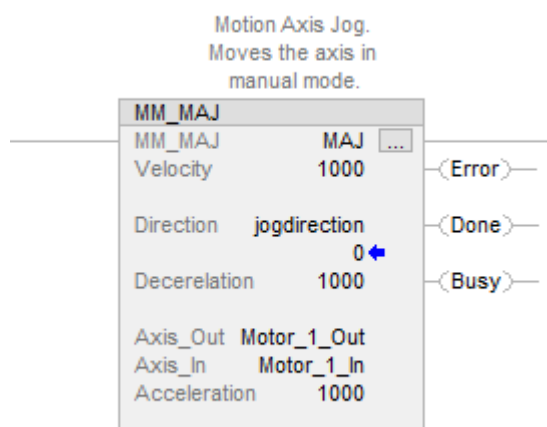
2.5.7. MM_MAS

Motion Axis Stop. Stops the motor.



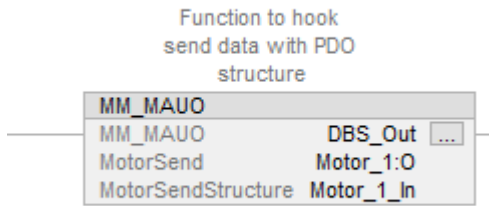
2.5.8. MM_MAJ

Motion Axis Jog. Jogs the motor in either direction chosen with *jogdirection* (0 or 1). The trajectory is trapezoidal, built with target speed, acceleration and deceleration.



2.5.9. MM_MAUO

This block links the local data structure with the cyclic data exchange.



2.6. Writing Parameters with Async Messaging

More complex things are possible with the DBS/FC system.

2.6.1. Async Messaging

The application layer for the DBS55 motor is very close to the CiA402 servodrive definition; the PLC can interact with the motor CiA402 state machine using ControlWord and StatusWord.

Nevertheless, an important part of the CiA402 servodrive model is managed via asynchronous read/write of the CanOPEN object dictionary. In the Ethernet/IP implementation, the asynchronous messaging is implemented as Class 3 GetAttribute/SetAttribute services, where portions of the CanOpen object dictionary are exposed to the Ethernet/IP world via attributes.

There are 4 main Ethernet/IP manufacturer objects that expose the attributes which act as a gateway to the CanOPEN world:

- **COM object – object 0x64:** this object exposes some basic information (e.g. software release) and allows saving parameters to non-volatile storage.
- **PAR object – object 0x65:** this object allows changing the motor parameters via PLC; this allows, for example, changing dynamically the Kp of the speed loop if the load of the motor changes during a machine phase.
- **MISC object – object 0x66:** this object allows reading and writing hardware related information (e.g. analog input value, heatsink temperature etc.).
- **CIA402 objects – object 0x67:** this object allows reading and writing CiA402 objects: for example, reading object 6081h.00 is translated to a GetAttribute on object 0x67, instance 0x60, attribute 0x81.

As an example we will read the BSI parameter P160, PosFactorNum, which is used to scale the motor position value. Using the MSG block, the configuration will be as follows.

As seen in the first chapter of this guide, P160 is part of the Class 0x65, it is in the first instance and its value in hex is A0. Remember to create a Destination element where to write down the value of interest.

Message Configuration - MSG_GetRev

Configuration
Communication
Tag

Message Type: CIP Generic

Service Type: Get Attribute Single

Source Element:

Source Length: 0 (Bytes)

Service Code: e (Hex) Class: 65 (Hex)

Destination Element: APL_Rev

Instance: 1 Attribute: a0 (Hex)

New Tag...

☐ Enable
☐ Enable Waiting
☐ Start
☐ Done
Done Length: 0

☐ Error Code:
Extended Error Code:
☐ Timed Out

Error Path: Motor_1
Error Text:

OK
Annulla
Applica
?

Conversely, to write the same parameter, the MSG must be set in Set Attribute Single.

Message Configuration - MSG_GetRev

Configuration*
Communication
Tag

Message Type: CIP Generic

Service Type: Set Attribute Single

Source Element: Posscaling

Source Length: 2 (Bytes)

Service Code: 10 (Hex) Class: 65 (Hex)

Destination Element:

Instance: 1 Attribute: A0 (Hex)

New Tag...

☐ Enable
☐ Enable Waiting
☐ Start
☐ Done
Done Length: 0

☐ Error Code:
Extended Error Code:
☐ Timed Out

Error Path: Motor_1
Error Text:

OK
Annulla
Applica
?