AG25, AG26

B&R Automation Studio Library

Software description

V1.1

Table of contents

1 General notes

The library and its function were tested on a B&R X20 CP 1583. The module was programmed using Automation Studio V4.0.20.56.

1.1 Requirements

- Basic knowledge of handling and programming B&R systems.
- Familiarity with Ethernet Powerlink.

1.2 Liability

SIKO GmbH assumes no warranty whatsoever regarding topicality, correctness, completeness or quality of the information or software products provided. All liability claims against SIKO GmbH referring to material or immaterial damages caused by using or not using the information or software provided or by using erroneous or incomplete information or software are always excluded.

1.3 Versions overview

V1.1 valid for library SIKODrvLib Version 1.01.4

1.4 List of abbreviations

EPL	Ethernet Powerlink®
SW	Status Word
CW	Control Word

2 Hardware configuration

NOTICE The AG2x could not be identified online. The module must be added to the configuration manually

2.1 Create a new Automation Studio project by using New Project Wizard.

Name the project "SIKO_example".

Choose "Identify hardware configuration online".

2.2 Setup connection to the PLC

NOTICE Ask your administrator for allowed IP address settings.

Precondition: The PLC is powered and connected to the programming PC.

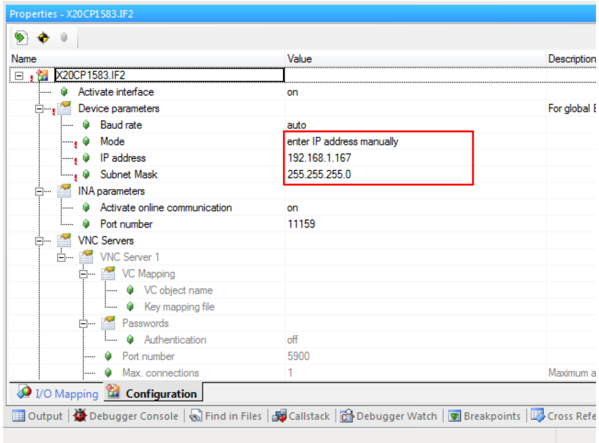
After identification you must set up the IP parameters of the PLC inside the project.

In this example we assign an IP address manually.

Go to tab "Physical View".

Right click on folder "ETH" and choose "Properties...".

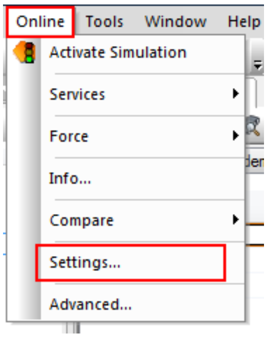
Go to tab "Configuration"



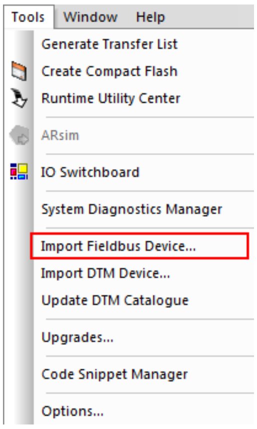
Name	Value	Description
X20CP1583.IF2		
Activate interface	on	
Device parameters		For global
Baud rate	auto	
Mode	enter IP address manually	
IP address	192.168.1.167	
Subnet Mask	255.255.255.0	
INA parameters		
Activate online communication	on	
Port number	11159	
VNC Servers		
VNC Server 1		
VC Mapping		
VC object name		
Key mapping file		
Passwords		
Authentication	off	
Port number	5900	
Max. connections	1	Maximum

2.3 Connect to the PLC

Establish a connection with the PLC via "Online" > "Settings".



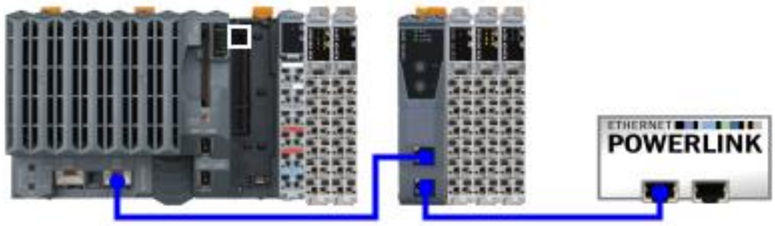
2.4 Import the XDD Device Description File for AG2X



2.5 Add new module to your hardware configuration

After import of the XDD-file you can search for "SIKO" in the hardware catalog. Choose the module "SIKO DriveLine AG2x EPL" and add it to the System Designer.

Add the Powerlink connection.



2.6 Declare global variables

Go to tab "Logical View".

Right click on folder "SIKO_example\Global.var".

Choose Open > Open As Text.

Type in or copy the declarations into the window "Variable Declaration".

```
VAR
  ParameterAg2x_1 : ParameterAg2x_type;
  gDrive1ActualValue : DINT;
  gDrive1StatusWord : UINT;
  gDrive1diStatus : USINT;
  gDrive1TargetValue : DINT;
  gDrive1ControlWord : UINT;
  gDrive1doControl : USINT;
END_VAR
```

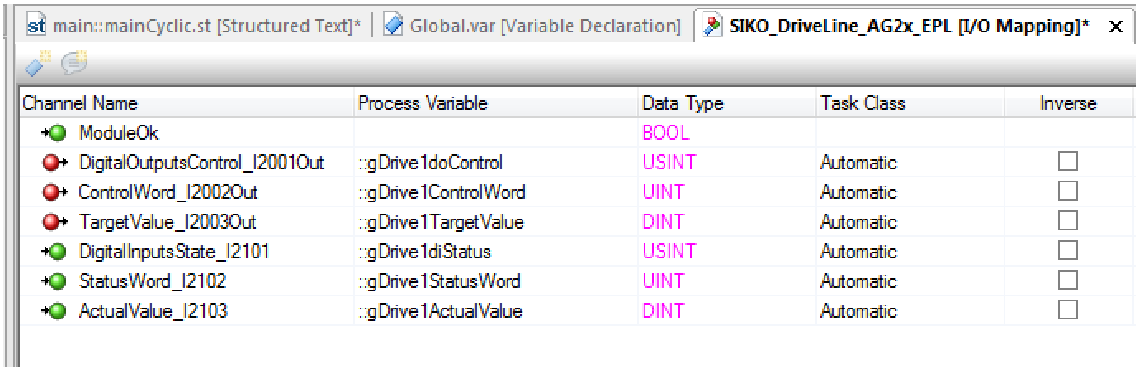
2.7 Add Mapping

Go to tab "Physical View".

Double click on the folder "..\SIKO_DriveLine_AG2x_EPL" to open the "I/O Mapping" window.

Go to window "I/O Mapping" and right click in row "Process Variable" choose "Add Mapping".

After setup of the mapping the window "I/O Mapping" should look like this:


main::mainCyclic.st [Structured Text]*
Global.var [Variable Declaration]
SIKO_DriveLine_AG2x_EPL [I/O Mapping]*

Channel Name	Process Variable	Data Type	Task Class	Inverse
ModuleOk		BOOL		
DigitalOutputsControl_I2001Out	::gDrive1doControl	USINT	Automatic	☐
ControlWord_I2002Out	::gDrive1ControlWord	UINT	Automatic	☐
TargetValue_I2003Out	::gDrive1TargetValue	DINT	Automatic	☐
DigitalInputsState_I2101	::gDrive1diStatus	USINT	Automatic	☐
StatusWord_I2102	::gDrive1StatusWord	UINT	Automatic	☐
ActualValue_I2103	::gDrive1ActualValue	DINT	Automatic	☐

3 Software configuration

3.1 Import package SIKODrive

Go to tab "Logical View".

Import package "SIKODriveVx-xx-x.zip" with File > Import.

Export "SIKODrvLib" to Library.

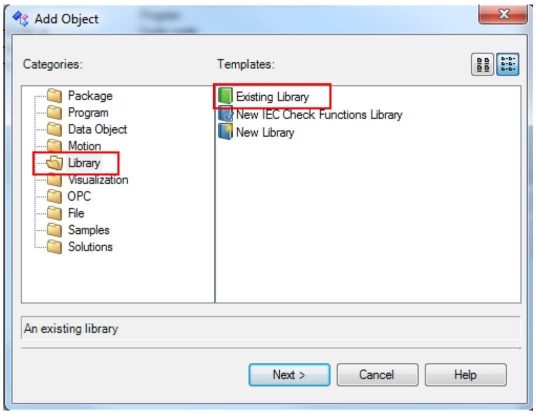
Delete the folder "SIKO_example\SIKODrive".

3.2 Library import

Go to tab "Logical View".

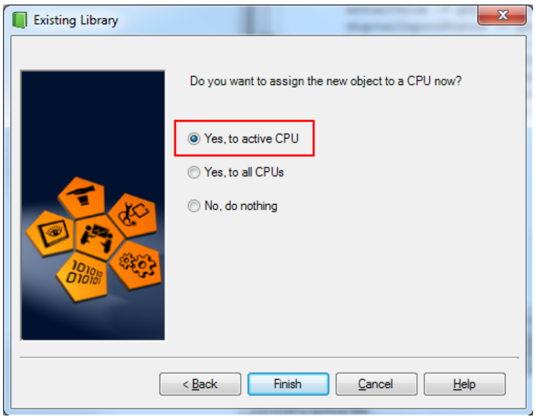
Right click on folder "SIKO_example\Libraries" and choose "Add Object...".

Add libraries "AsEPL", "DataObj" and "standard" to the project.



An existing library

Assign new object to active CPU



Do you want to assign the new object to a CPU now?

● Yes, to active CPU

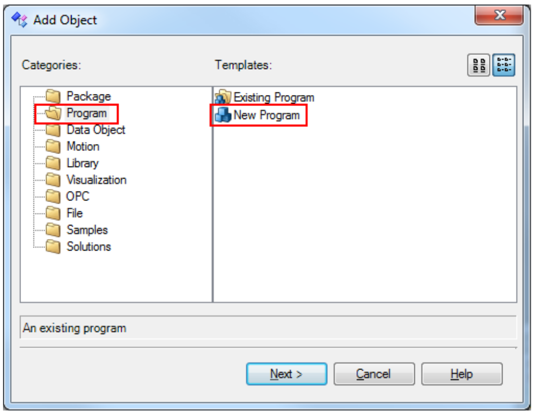
○ Yes, to all CPUs

○ No, do nothing

3.3 Add programm to project

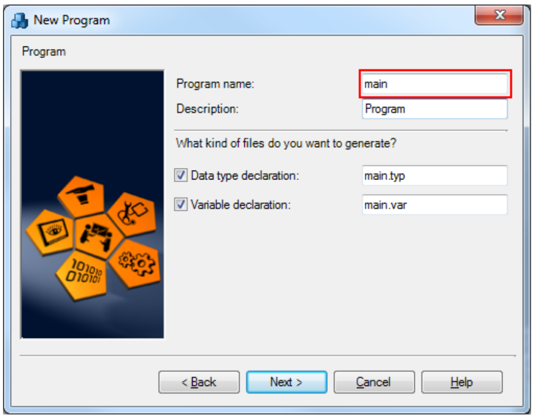
Go to tab "Logical View".

Right click on folder "SIKO_example" and choose "Add Object...".



An existing program

Save programm as "main".



Program name: main

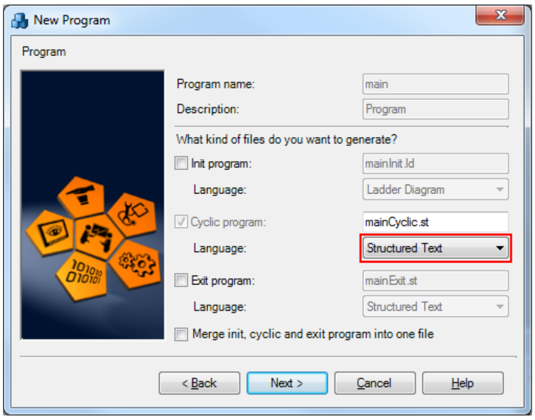
Description: Program

What kind of files do you want to generate?

☑ Data type declaration: main.typ

☑ Variable declaration: main.var

Choose language "Structured Text"



Program name: main

Description: Program

What kind of files do you want to generate?

☐ Init program: mainInit.ld

Language: Ladder Diagram

☑ Cyclic program: mainCyclic.st

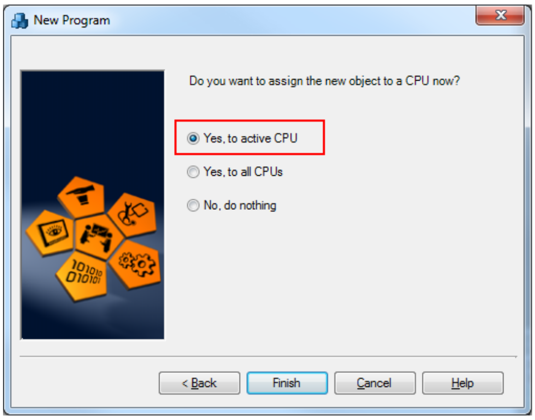
Language: Structured Text

☐ Exit program: mainExit.st

Language: Structured Text

☐ Merge init, cyclic and exit program into one file

Assign program to active CPU



Do you want to assign the new object to a CPU now?

- ● Yes, to active CPU
- ○ Yes, to all CPUs
- ○ No, do nothing

3.4 Create instances of fbSikoAg2x, ParameterAg2x, ST_Ag2xStatus, ST_Ag2xControl

Go to tab "Logical View".

Right click on folder "SIKO_example\main\main.var" and choose Open > Open As Text.

Type in or copy the declarations into the window "Variable Declaration".

```
VAR
  fbSikoAg2x_1 : fbSikoAg2x;
  ParameterAg2x_1 : ParameterAg2x_type;
  PdInAg2x_1 : PdInAg2x_type;
  PdOutAg2x_1 : PdOutAg2x_type;
END_VAR
```

3.5 Funktion block call

Go to tab "Logical View".

Double click on folder "SIKO_example\main\mainCyclic.st"

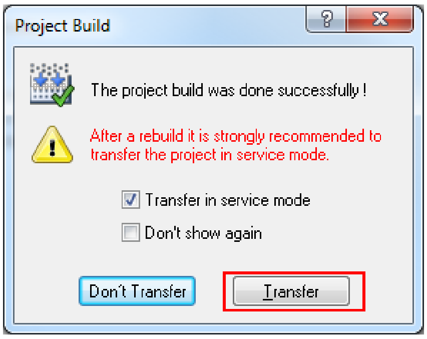
Type in or copy the program into the window "Structured Text".

```
PROGRAM _CYCLIC
  fbSikoAg2x_1(
            modulename := 'Drive1',
            nodeId := 2,
            statusWord := gDrive1StatusWord,
            actualValue := gDrive1ActualValue,
            digitalInputsStatus := gDrive1diStatus,
            PdOutAg2x := PdOutAg2x_1,
            ParameterAg2x := ADR(ParameterAg2x_1));
  PdInAg2x_1:= fbSikoAg2x_1.PdInAg2x;
  gDrive1ControlWord:= fbSikoAg2x_1.controlWord;
  gDrive1TargetValue:= fbSikoAg2x_1.targetValue;
  gDrive1doControl:= fbSikoAg2x_1.digitalOutputsControl;
END_PROGRAM
```

3.6 Built project

After project is built successfully a message window to transfer the project is shown.

Confirm "Transfer".



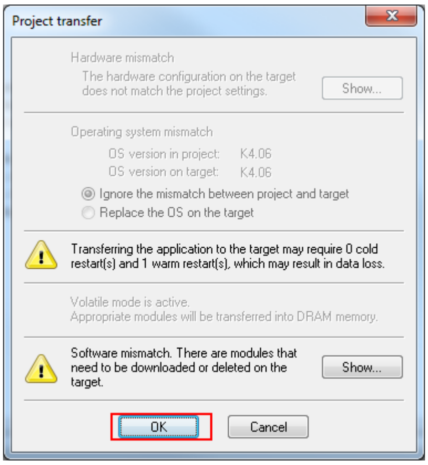
The project build was done successfully !

After a rebuild it is strongly recommended to transfer the project in service mode.

☑ Transfer in service mode

☐ Don't show again

A second window appears, confirm ok.



Hardware mismatch

The hardware configuration on the target does not match the project settings.

Operating system mismatch

OS version in project: K4.06
OS version on target: K4.06

● Ignore the mismatch between project and target
○ Replace the OS on the target

Transferring the application to the target may require 0 cold restart(s) and 1 warm restart(s), which may result in data loss.

Volatile mode is active.
Appropriate modules will be transferred into DRAM memory.

Software mismatch. There are modules that need to be downloaded or deleted on the target.

After transfer of the project the CPU is in service mode. To start the program carry out a warm start.

3.7 Create watch configuration

Go to tab "Logical View".

Click on folder "SIKO_example\main".

Then click "Open" > "Watch".

Right click inside window "Watch" and choose "Insert Variable".

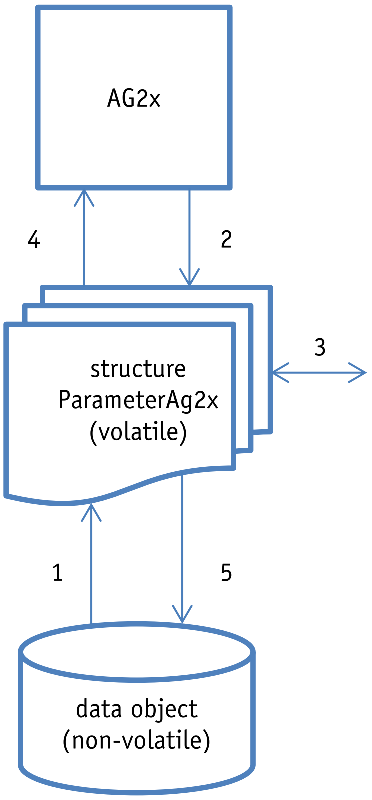
3.8 Software example

3.8.1 Parameter access

NOTICE	**Danger of data loss** All parameters are stored non-volatile in a data object specified by input "modulename". To access parameters from the PLC program it is necessary to load them from the data object via command "startReadDataObj" once after startup into the structure ParameterAg2x. It is the responsibility of the programmer to store changes within the structure ParameterAg2x to the data object via command "startWriteDataObj" to avoid data loss.

The present module contains the parameter data in addition to the process data (CW/SW). Parameters that can be changed (read/write) exist in programming as actual value (_r) and as target value (_w) as well. Furthermore, it is differentiated between pure read parameters (only indicated as actual value) and pure write parameters (only indicated as target value).

3.8.2 Storage model for drive parameters



1: read data object to structure ParameterAg2x
(command "startReadDataObj")

2: read parameter from drive to _r values of
structure ParameterAg2x
(command "startRead")

3: access to actual values (_r) and
target values (_w) from PLC program.
Parameters are only valid if data object was
…read once.

4: write parameters to drive
(command "startWrite")

5: write data object
(command "startWriteDataObj")

3.8.3 Read parameters from data object

If a rising edge is applied to the "startReadDataObj" input, then all parameter values are loaded from the data object specified at input "modulename" into the structure ParameterAg2x. If the specified data object does not exist a data object with default values will be created for further use.

3.8.4 Write parameters to data object

If a rising edge is applied to the "startWriteDataObj" input, then the structure ParameterAg2x is stored to the data object specified at input "modulename".

3.8.5 Read parameters from drive

Precondition: Data object is loaded via command "startReadDataObj"

If a rising edge is applied to the "startRead" input, then all parameters will be read to the structure ParameterAg2x and can be used for further programming.

3.8.6 Diagnosis Reading "counterRead"

If counter value is not reset to "0" the read cycle was interrupted by read failure. This indicates to a communication failure. For detailed information about the cause of failure there is an error code available at the output "status".

3.8.7 Write parameters to drive

Precondition: Data object is loaded via command "startReadDataObj"

If a rising edge is applied to the "startWrite" input of the module, then all actual values (_r) and their corresponding target values (_w) of the structure ParameterAg2x are compared. If they are unequal the target value will be transferred to the drive.

3.8.8 Diagnosis Reading "counterWrite"

If counter value is not reset to "0" the write cycle was interrupted by write failure. This indicates to a communication failure or a parameter value is beyond range of value accepted by drive. For detailed information about the cause of failure there is an error code available at the output "status".

3.8.9 Copy parameters from read to write

Precondition: Data object is loaded via command "startReadDataObj"

If a rising edge is applied to the "startCopy" input of the module, then all actual values (_r) are copied to their corresponding target values (_w).

3.8.10 Counter value

counter Read	counter Write	Name	Value range (dec)	Default
1	1	LED Functionality	0 … 1	0
2	2	Service Interface Baud Rate	0 … 3	1
3	3	Digital Output 1 Functionality	0 … 3	0
4		Digital Output Functionalities State		-
5	4	Digital Outputs Polarity	0 … 15	0
6	5	Digital Input 1 Functionality	0 … 11	0
7	6	Digital Input 2 Functionality	0 … 11	0
8	7	Digital Input 3 Functionality	0 … 11	0
9	8	Digital Input 4 Functionality	0 … 11	0
10		Digital Input Functionalities State		-
11	9	Digital Inputs Polarity	0 … 15	0
12	10	Controller Parameter P	1 … 500	300
13	11	Controller Parameter I	0 … 500	2
14	12	Controller Parameter D	0 … 500	0
15	13	A-Pos	1 … 100	50
16	14	V-Pos	Gear 66:1 ⇒ 1 … 75 rpm 98:1 ⇒ 1 … 50 rpm 188:1 ⇒ 1 … 30 rpm 368:1 ⇒ 1 … 15 rpm	10
17	15	D-Pos	1 … 101	101
18	16	A-Rot	1 … 100	50
19	17	A-Inch	1 … 100	50
20	18	V-Inch	Gear 66:1 ⇒ 1 … 75 rpm 98:1 ⇒ 1 … 50 rpm 188:1 ⇒ 1 … 30 rpm 368:1 ⇒ 1 … 15 rpm	10
21	19	Pos Window	0 … 1000	10
22	20	Gear Ratio Numerator	1 … 10000	1
23	21	Gear Ratio Denominator	1 … 10000	1
24	22	Spindle Pitch	0 … 1000000	0
25	23	Calibration Value	-999999 … 999999	0
26	24	Software Limit 1	-9999999 … 9999999	99999
27	25	Software Limit 2	-9999999 … 9999999	-19999
28	26	Delta Inch	-1000000 … 1000000	720
29	27	Sense of Rotation	0 … 1	0
30	28	Pos Type	0 … 2	0
31	29	Operating Mode	0 … 1	0
32	30	Inching 2 Stop Mode	0 … 1	0

counter Read	counter Write	Name	Value range (dec)	Default
33	31	Inpos Mode	0 … 2	0
34	32	Loop Length	0 … 30000	360
35	33	Contouring Error Limit	1 … 30000	400
36	34	Current Limiting	25 … 110	110
37	35	Inching 2 Offset	10 … 100	100
38	36	Inching 2 Acceleration Type	0 … 1	0
39	37	Offset Value	-999999 … 999999	0
40	38	PCM Position 1	DINT	0
41	39	PCM Position 2	DINT	0
42	40	PCM Position 3	DINT	0
43	41	PCM Position 4	DINT	0
44	42	PCM Position 5	DINT	0
45	43	PCM Position 6	DINT	0
46	44	PCM Position 7	DINT	0
47	45	PCM Acceleration 1	1 … 100	50
48	46	PCM Acceleration 2	1 … 100	50
49	47	PCM Acceleration 3	1 … 100	50
50	48	PCM Acceleration 4	1 … 100	50
51	49	PCM Acceleration 5	1 … 100	50
52	50	PCM Acceleration 6	1 … 100	50
53	51	PCM Acceleration 7	1 … 100	50
54	52	PCM Velocity 1	Gear 66:1 ⇒ 1 … 75 rpm 98:1 ⇒ 1 … 50 rpm 188:1 ⇒ 1 … 30 rpm 368:1 ⇒ 1 … 15 rpm	10
55	53	PCM Velocity 2	see PCM Velocity 1	10
56	54	PCM Velocity 3	see PCM Velocity 1	10
57	55	PCM Velocity 4	see PCM Velocity 1	10
58	56	PCM Velocity 5	see PCM Velocity 1	10
59	57	PCM Velocity 6	see PCM Velocity 1	10
60	58	PCM Velocity 7	see PCM Velocity 1	10
61	59	PCM Deceleration 1	1 … 101	101
62	60	PCM Deceleration 2	1 … 101	101
63	61	PCM Deceleration 3	1 … 101	101
64	62	PCM Deceleration 4	1 … 101	101
65	63	PCM Deceleration 5	1 … 101	101
66	64	PCM Deceleration 6	1 … 101	101
67	65	PCM Deceleration 7	1 … 101	101
68		Output Stage Temperature		-
69		Voltage of Control		-

counter Read	counter Write	Name	Value range (dec)	Default
70		Voltage of Output Stage		-
71		Voltage of Battery		-
72		Motor Current		-
73		Actual Position		-
74		Actual Rotational Speed		-
75		Serial Number		-
76		Production Date		-
77		SW Motor Controller		-
78		Gear Reduction		-
79		System Status Word		-
80		Encoder Resolution		-
81		Device ID		-
82		Number of Errors		-
83		Error Number 1		-
84		Error Number 2		-
85		Error Number 3		-
86		Error Number 4		-
87		Error Number 5		-
88		Error Number 6		-
89		Error Number 7		-
90		Error Number 8		-
91		Error Number 9		-
92		Error Number 10		-
93	66	S Command	0 … 8	0

3.8.11 Error codes

If a error occurs, there is an error code present at the output "status" (see Automation Studio Help, Diagnositcs and service, Error numbers, AR System).

3.8.11.1 Manufacturer specific error numbers

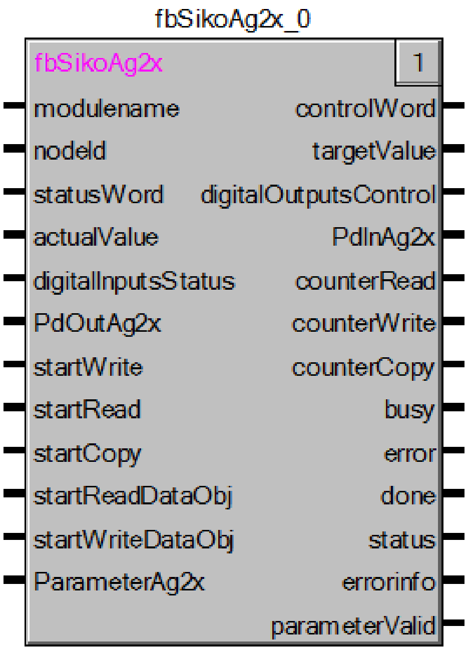
50000	Data object not loaded

4 Function block description

The inputs statusWord, actualValue, digitalInputsStatus are mapped by the function block to a structure of type PdInAg2x_type. This structure is available at output PdInAg2x for further programming.

The function block expects a structure of type PdoutAg2x_type at input PdOutAg2x.
The function block mapps the structure to the outputs controlWord, targetValue and digitalOutputsControl.

The function block provides read and write access to a data object for non-volatile storage of the drive parameters inside the PLC.



4.1 Function block inputs

Input	Data type	Description
modulename	STRING[8]	Name of data object
nodeId	USINT	Address of the node to be accessed (1-255)
statusWord	UINT	Connect to global input variable.
actualValue	DINT	Connect to global input variable.
digitalInputsStatus	USINT	Connect to global input variable.
PdOutAg2x	PdOutAg2x_type	Process data output.
startWrite	BOOL	Start command write to drive.
startRead	BOOL	Start command read from drive.
startCopy	BOOL	Start command copy.
startReadDataObj	BOOL	Start command read data object.
startWriteDataObj	BOOL	Start command write data object.
ParameterAg2x	REFERENCE TO ParameterAG2x_type	Address of the variable of type ParameterAG2x_type

4.2 Function block outputs

Output	Data type	Description
controlWord	UINT	Connect to global output variable.
targetValue	DINT	Connect to global output variable.
digitalOutputsControl	USINT	Connect to global output variable.
PdInAg2x	PdInAg2x_type	Process data input mapped to a structure.
counterRead	USINT	Counter value of read cycle.
counterWrite	USINT	Counter value of write cycle.
counterCopy	USINT	Counter value of copy cycle.
busy	BOOL	Command in processing.
error	BOOL	Command terminated with error.
done	BOOL	Command executed successfully.
status	UINT	Error number (0 = no error)
errorinfo	UDINT	SDO Abort Code (if status = ERR_ASEPL_ACCESS_FAILED)
parameterValid	BOOL	0 = no data object loaded 1 = data object loaded