AMKASYN
Application interface API
ACC-bus / CANopen interface

Version: 2023/25
Part no.: 200335

Translation of the "Original Dokumentation"

AMKmotion

MEMBER OF THE ARBURG FAMILY

Imprint

Name: PDK_200335_API_en

Version:

Version	Change	Letter symbol
2023/25	AMKmotion Design	STL

Previous version: 2014/51

Product version:

Product Part no.	Firmware Version (Part no.)	Hardware Version (Part no.)
KE	from KE-E03 V3.03 2010/18	

Reservation: We reserve the right to modify the content of the documentation as well as the delivery options for the product.

Publisher: AMKmotion GmbH + Co KG
Gaußstraße 37-39
73230 Kirchheim unter Teck
Germany

Phone +49 7021 50 05-0
Fax +49 7021 50 05-176

E-mail: info@amk-motion.com

Registration court: AG Stuttgart, HRA 230681, Kirchheim unter Teck,
Tax Id no.: DE 145 912 804

Complementary: AMKmotion Verwaltungsgesellschaft mbH, HRB 774646

Service: Phone +49 7021 50 05-190, Fax -193

For fast and reliable troubleshooting, you can help us by informing our Customer Service about the following:

- Type plate data for each unit
- Software version
- Device configuration and application
- Type of fault/problem and suspected cause
- Diagnostic messages (error messages)

E-mail service@amk-motion.com

Internet address: www.amk-motion.com

Content

1 Abbreviations

ACC	AMKASYN CAN COMMUNICATION Hardware synchronised CANopen interface
AMK-PLC	AMKASYN programmable logical controller
AIPEX	AMK start-up and parameter setting explorer
AFP	AMK field bus protocol for drive commanding
API	Application Interface
CAN	Controller Area Network (field bus interface)
CAN-S	Hardware synchronised CANopen Protocol
ccb	Concise configuration binary file type *.ccb
ccf	Concise configuration file *.ccf
CiA	CAN in Automation, e.V
Ident number	(ID No.) Parameter according to SERCOS interface standard for parameterization of the AMKASYN Systems
KE/KW	Compact power supply module/compact inverter module
KU	AMKASYN Digital compact servo drives
Mapping	for data exchange via fieldbus data blocks are available (PDO) the application data entries are called „mapping“
PDO	Process data object
PGT	Periphery system clock
Rx	Receive data
SDO	Service data object
SWQ	setpoint source
Tx	Transmit data

2 What is API? Introduction and Overview

The Application interface API is an application layer which enables simple access to the AMK drive functions. The API is a variable array of the AMK operating software (firmware) and allows to control and analyse the drive. The API is divided into sending variables (e.g. actual values, status variables, configurable cyclical display values, variables for the evaluation of binary inputs...) and receiving variables (e.g. 16/32 bit setpoint sources, control variables, variable for the control of binary outputs...), Transmit / receive direction out of the view of the drive.

AMK devices which are linked to each other through a field bus, can exchange drive data with each other using the API variables.

Both, drive-internal plc controls (AMK-plc option cards, controller cards with integrated plc, AMKAMAC) as well as field bus systems have read and write access to the API variable array.

Centralized PLC systems:

For centralized PLC systems, variable contents (e.g. Setpoints) are mapped from the PLC program in an API variable of the receiving drive so that there is no further intelligence required for data interpretation. “Mapping” means the user message configuration to exchange application data.

Decentralized PLC systems:

Different AMK plc´s can exchange data via field bus variables. The plc´s have to evaluate the data in the plc programs.

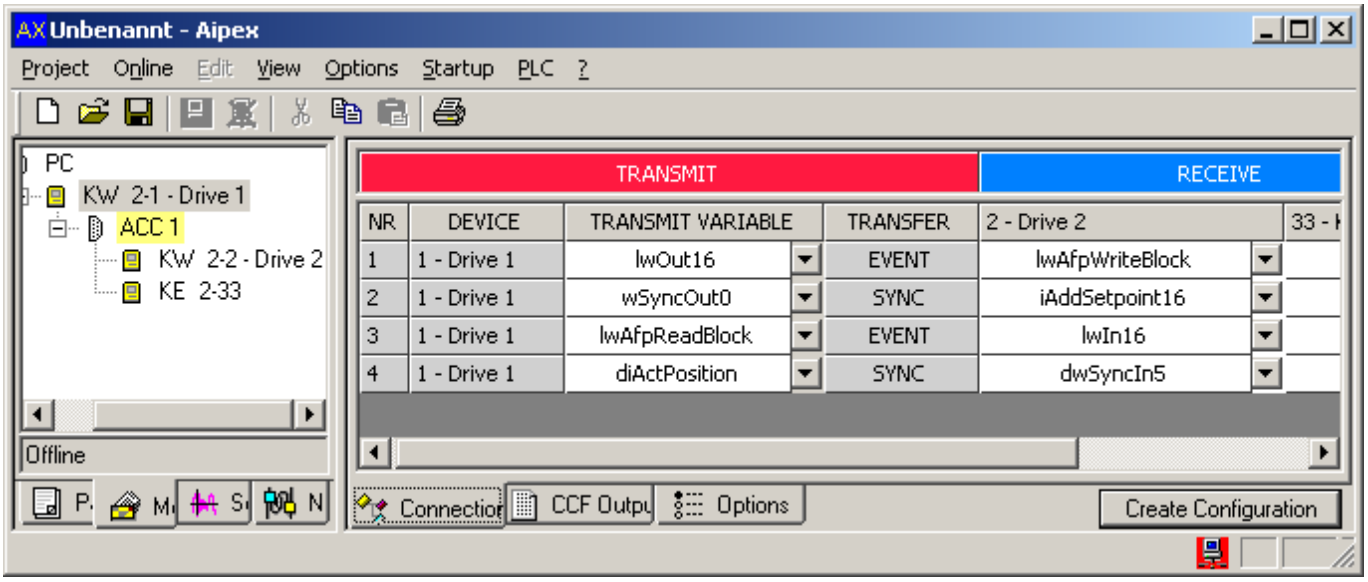
Working with API variables allows an efficient and goal-oriented synchronous data transmission between network stations in synchronous CANopen bus systems (ACC-Bus).

The data exchange has to be configured in the PC Software AIPEX in the message configurator.

For every node transmit data can be selected out of a list. Via link list every transmit data will be assigned to a receiver.

AIPEX creates a ”CAN Configuration File“ (*.ccf, *.ccb) out from the link list. The configuration file is stored in a parameter of the system and must be transferred into the ACC-Bus master.

Message configuration with AIPEX



The screen shot shows the link list in the AIPEX message configurator. With the link list the transmit and receive data are assigned.

Example:

Transmit		**Receive**	
Variable	**Type**	**1-Drive1 SPS**	**2-Drive2**
2-Drive2.diActPosition	SYNC	dwSyncIn5	-

The allocation effects that the actual position value (diActPosition) of drive-2 will be send to drive-1-SPS cyclically within the ID2 cycle time (clock signal of the position loop) and can be evaluated in the synchronous plc program task. Drive 1 receive the actual position value in the variable dwSyncIn5.

The field bus variables are symbolic addressing to the device memory where the plc has read and write access.

3 Field Bus Variables and Address Area

The field bus variables are used to transmit and receive data with an AMK-PLC platform. Address ranges with different significance and intended use are assigned to the field bus variables. There is a difference between an asynchronous and a synchronous variable array. The synchronous range differs from the asynchronous one in that variables are processed in a synchronized manner to a hardware synchronization signal (jitter smaller 1ms). These variables are used for the transmission of, for example, set points, actual values ... The asynchronous range serves as transmission of non-time-critical data and of the AFP protocol.

Addressing of the field bus variable is carried out through the replacement of "X" by a number which corresponds to an address in the address range. The following diagram shows the address ranges in the various AMK hardware components.

Note: The „reserved“ marked address areas are limited through the plc target system.

For addressing, it must be ensured that not overlapping addressing is carried out. A byte with the address 0 and 1 occupies the same address range as a WORD with the address 0.

The address areas shown in the following figures are available as input area to receive variables and additional as output area to transmit variables.

Addressing field bus variables

Field bus variables		**Memory organisation Addressing**																				**Example**	
Typ	Syntax																						
Long word	lwInx / lwOutx	0								1												lwIn1	lwOut1
Double word	dwInx / dwOutx	0				1				2				3								lwIn2	lwOut2
Word	wInx / wOutx	0		1		2		3		4		5		6		7						wIn4	wOut4
Byte	byInx / byOutx	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17			byIn8	byOut8

Note: The data exchange of the asynchronous address area is only byte consistent.
The data exchange of the synchronous area variables is always consistent.

Field bus variables for KW- / KU-R03P

Transmission type	**ASYNCHRONOUS**			
Byte address	**0 ... 7**	**8 ... 63**	**64...127**	**128 ... 255**
Use	Reserved for Hardware	Binary I/O	Res.	AFP commands
Field bus variables	-	lwIn1…lwIn7 / lwOut1…lwOut7 dwIn2…dwIn15 / dwOut2…dwOut15 wIn4...wIn31 / wOut4...wOut31 byIn8...byIn63 / byOut8...byOut63	-	lwIn16…lwIn31 / lwOut16…lwOut31 dwIn32...dwIn63 / dwOut32...dwOut63 wIn64…wIn127 / wOut64…wOut127 byIn128…byIn255 / byOut128..byOut255

Transmission type	**SYNCHRONOUS**
Byte address	**0 ... 31**
Use	cyclic synchronous data
Field bus variables	wSynchIn0…lwSyncIn3 / lwSyncOut0…lwSyncOut3 dwSyncIn0…dwSyncIn7 / dwSyncOut0…dwSyncOut7 wSyncIn0…wSyncIn15 / wSyncOut0…wSyncOut15 bySyncIn0…bySyncIn31 / bySyncOut0…bySyncOut 31

Field bus variables for KW-/KU-PLC,KW-/KU-PLC1

Transmission type	ASYNCHRONOUS			
Byte address	**0 ... 7**	**8 ... 63**	**64...127**	**128 ... 255**
Use	Reserved for Hardware	Binary I/O	Res.	AFP commands
Field bus variables	-	lwIn1…lwIn7 / lwOut1…lwOut7 dwIn2…dwIn15 / dwOut2…dwOut15 wIn4...wIn31 / wOut4...wOut31 byIn8...byIn63 / byOut8...byOut63	-	lwIn16…lwIn31 / lwOut16…lwOut31 dwIn32...dwIn63 / dwOut32...dwOut63 wIn64…wIn127 / wOut64…wOut127 byIn128…byIn255 / byOut128..byOut255

Transmission type	SYNCHRONOUS
Byte address	**0 ... 63**
Use	cyclic synchronous data
Field bus variables	lwSynchIn0…lwSyncIn7 / lwSyncOut0…lwSyncOut7 dwSyncIn0…dwSyncIn15 / dwSyncOut0…dwSyncOut15 wSyncIn0…wSyncIn31 / wSyncOut0…wSyncOut31 bySyncIn0…bySyncIn63 / bySyncOut0…bySyncOut 63

Field bus variables for KW-/KU-PLC2

Transmission type	ASYNCHRONOUS		
Byte address	**0 ... 7**	**8 ... 127**	**128 ...255**
Use	Reserved for Hardware	Binary I/O	AFP commands
Field bus variables	-	lwIn1…lwIn15 / lwOut1…lwOut15 dwIn2…dwIn31 / dwOut2…dwOut31 wIn4...wIn63 / wOut4...wOut63 byIn8...byIn127 / byOut8...byOut127	lwIn16…lwIn31 / lwOut16…lwOut31 dwIn32...dwIn63 / dwOut32...dwOut63 wIn64…wIn127 / wOut64…wOut127 byIn128…byIn255 / byOut128..byOut255

Transmission type	SYNCHRONOUS
Byte address	**0 ... 127**
Use	cyclic synchronous data
Field bus variables	lwSynchIn0…lwSyncIn15 / lwSyncOut0…lwSyncOut15 dwSyncIn0…dwSyncIn31 / dwSyncOut0…dwSyncOut31 wSyncIn0…wSyncIn63 / wSyncOut0…wSyncOut63 bySyncIn0…bySyncIn127 / bySyncOut0…bySyncOut127

Field bus variables for AMKAMAC Controller AS-PL12 and integrated AS-FCT1 assembly

Transmission type	**ASYNCHRONOUS**	
Byte address	**0 ... 255**	**256 ... 511**
Use	Binary I/O	AFP commands
Fieldbus-variables	lwIn0…lwIn63 / lwOut…lwOut63 dwIn0…dwIn127 / dwOut0…dwOut127 wIn0...wIn255 / wOut0…wOut255 byIn0...byIn511 / byOut0…byOut511	

Transmission type	**SYNCHRONOUS**
Byte address	**0 ... 255**
Use	cyclic synchronous data
Field bus variables	lwSynchIn0…lwSyncIn31 / lwSyncOut0…lwSyncOut31 dwSyncIn0…dwSyncIn63 / dwSyncOut0…dwSyncOut63 wSyncIn0…wSyncIn127 / wSyncOut0…wSyncOut127 bySyncIn0…bySyncIn255 / bySyncOut0…bySyncOut255

Field bus variables for AMKAMAC Controller AS-PL14 and integrated AS-FCT2 assembly

Transmission type	**ASYNCHRONOUS**	
Byte address	**0 ... 255**	**256 ... 511**
Use	Binary I/O	AFP commands
Fieldbus-variables	lwIn0…lwIn511 / lwOut…lwOut511 dwIn0…dwIn1023 / dwOut0…dwOut1023 wIn0...wIn2047 / wOut0…wOut2047 byIn0...byIn4095 / byOut0…byOut4095	

Transmission type	**SYNCHRONOUS**
Byte address	**0 ... 511**
Use	cyclic synchronous data
Field bus variables	lwSynchIn0…lwSyncIn63 / lwSyncOut0…lwSyncOut63 dwSyncIn0…dwSyncIn127 / dwSyncOut0…dwSyncOut127 wSyncIn0…wSyncIn255 / wSyncOut0…wSyncOut255 bySyncIn0…bySyncIn511 / bySyncOut0…bySyncOut511

Field bus variables for AMKAMAC Controller AS-C and integrated AS-FCT1 assembly

Transmission type	**ASYNCHRONOUS**	
Byte address	**0 ... 255**	**256 ... 511**
Use	Binary I/O	AFP commands
Fieldbus-variables	lwIn0…lwIn63 / lwOut…lwOut63 dwIn0…dwIn127 / dwOut0…dwOut127 wIn0...wIn255 / wOut0…wOut255 byIn0...byIn51 byOut0…byOut511	

Transmission type	**SYNCHRONOUS**
Byte address	**0 ... 127**
Use	cyclic synchronous data
Field bus variables	lwSynchIn0…lwSyncIn15 / lwSyncOut0…lwSyncOut15 dwSyncIn0…dwSyncIn31 / dwSyncOut0…dwSyncOut31 wSyncIn0…wSyncIn63 / wSyncOut0…wSyncOut63 bySyncIn0…bySyncIn127 / bySyncOut0…bySyncOut127

4 Conventions

The API variable names start with a PREFIX, identifying the data type:

Data type-	Type name	Keyword	Prefix	Example
Bit sequence 8 Bit	Byte	BYTE	**by**	byAxisState
Bit sequence 16 Bit	Word	WORD	**w**	wRealTimeBits
Bit sequence 32 Bit	Double word	DWORD	**dw**	dwDummy
Bit sequence 64 Bit	Long word	LWORD	**lw**	lwOutX
integer	Integer	INT	**i**	iActValue0
double integer	Double integer	DINT	**di**	diActPosition

5 API variables for KW and KU modules

5.1 Receive of setpoints

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
diMainSetpoint	2030	01	Rx	32 bit set point, relevant in the operation modes ID32800...
diReserve1	2030	02	Rx	reserved
iAddSetpoint16	2030	03	Rx	16Bit incremental position setpoint, relevant in the operation modes ID32800…
diReserve2	2030	04	Rx	reserved
diAddSetpoint32	2030	05	Rx	32Bit incremental set point for synchronous mode with angle alignment (SWQW), relevant in the operation modes ID32800…

5.2 Transmit of actual values

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
iMessage16	2040	01	Tx	Configurable cyclic 16bit actual value message, see.ID32785
iMessage32	2040	02	Tx	Configurable cyclic 32bit actual value message, see.ID32785
diActPosition	2040	03	Tx	32Bit actual position value

5.3 Status and control variables

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
wDeviceState	2048	00	Tx	Status bits

BIT	SYNTAX	MEANING
0	SBM	System ready message
1	QUE	Acknowledgement inverter on (only KU)
2 ...3	-	Reserved
4	-	Reserved
5		Acknowledgement pulse width measurement
6	PR_POS	Acknowledgement probe pos. edge
7	PR_NEG	Acknowledgement probe neg. edge
8	QRF	Controller enable
9 ...15	-	Reserved

<table>
<tr><th>API variable name</th><th>CAN Index [hex]</th><th>CAN Sub-Index [hex]</th><th>Copy-direction</th><th>Application</th></tr>
<tr><td>wDeviceControl</td><td>2049</td><td>00</td><td>Rx</td><td>
<table>
<tr><td colspan="3">Control bits</td></tr>
<tr><th>BIT</th><th>SYNTAX</th><th>MEANING</th></tr>
<tr><td>0</td><td>FL</td><td>Delete error</td></tr>
<tr><td>1</td><td>UE</td><td>Inverter on
(only KU)</td></tr>
<tr><td>2 ...4</td><td>-</td><td>Reserved</td></tr>
<tr><td>5</td><td></td><td>Start pulse width measurement</td></tr>
<tr><td>6</td><td>CLR-I</td><td>Clears the integral component of the speed controller</td></tr>
<tr><td>7</td><td>PR_EN</td><td>Enable measuring function</td></tr>
<tr><td>8</td><td>RF</td><td>Controller enable [1)]</td></tr>
<tr><td>9</td><td></td><td>Delete residual distance</td></tr>
<tr><td>10...15</td><td>-</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>wRealTimeBits</td><td>204A</td><td>00</td><td>Tx</td><td>
<table>
<tr><td colspan="2">Real time bit messages</td></tr>
<tr><th>BIT</th><th>MEANING</th></tr>
<tr><td>0</td><td>$| n_{set} - n_{act} | < n_{window}$ ($n_{act} = n_{set}$), window according ID157</td></tr>
<tr><td>1</td><td>$| n_{act} | < n_{min}$, threshold acc. ID124</td></tr>
<tr><td>2</td><td>$| n_{act} | < n_x$, threshold acc. ID125</td></tr>
<tr><td>3</td><td>$| M_{act} | > M_x$, threshold acc. ID126</td></tr>
<tr><td>4</td><td>$| M_{act} | > M_{limit}$, limit acc. ID82, ID83</td></tr>
<tr><td>5</td><td>$| n_{set} | > | n_{limit} |$, limit acc. ID38, ID39</td></tr>
<tr><td>6</td><td>In position window acc. ID57</td></tr>
<tr><td>7</td><td>$| P_{act} | \geq P_x$, threshold acc. ID158</td></tr>
<tr><td>8</td><td>Negative position limit acc. ID50</td></tr>
<tr><td>9</td><td>Drive angle synchronous, window acc. ID228</td></tr>
<tr><td>10</td><td>Drive speed synchronous, windows acc. ID32952</td></tr>
<tr><td>11</td><td>$N_{act} \geq 0$</td></tr>
<tr><td>12</td><td>Acknowledgement actual value scaled</td></tr>
<tr><td>13</td><td>ID32922 residual distance window reset</td></tr>
<tr><td>14</td><td>Overcurrent message: capacity > ID32999 (50%)</td></tr>
<tr><td>15</td><td>Positive position limit ID49</td></tr>
<tr><td colspan="2">caption:
N speed, M torque,
P power X position</td></tr>
</table>
</td></tr>
<tr><td>byAxisState</td><td>204D</td><td>00</td><td>Tx</td><td>
<table>
<tr><td colspan="3">Status bits</td></tr>
<tr><th>BIT</th><th>SYNTAX</th><th>MEANING</th></tr>
<tr><td>0</td><td>REF_ACK</td><td>Reference point known</td></tr>
<tr><td>1 ...7</td><td></td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>wStatusBitsId144</td><td>204E</td><td>00</td><td>Tx</td><td>Free configurable status word for displaying real time bits out of the code list for binary output assignment (See also documentation Parameter, ID144 status bits). The content of ID144 will be defined via ID26 "configuration list status bits"</td></tr>
</table>

[1)] To control the signals UE and RF via API variables, the following parameter setting is necessary:

ID32795=5 Source UE
ID32796=5 Source RF

5.4 Binary inputs / outputs

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
bySystemIn	2060	00	Tx	Sends image of the binary inputs on the controller card
bySystemOut	2070	00	Rx	Receives image for the binary outputs on the controller card [1)]
byInp1Byte0	6000	01	Tx	Each sent byte makes the image of 8 binary input bits available (e.g. binary I/O option card) (e.g. I/O option card KU- or KW-EA2 in slot 1: I1...I8 are imaged by the variable byInp1Byte0, I9...I12 by the variable byInp1Byte1
byInp1Byte1	6000	02	Tx	
byInp1Byte2	6000	03	Tx	
byInp1Byte3	6000	04	Tx	
byInp1Byte4	6000	05	Tx	
byInp1Byte5	6000	06	Tx	
byInp1Byte6	6000	07	Tx	
byInp1Byte7	6000	08	Tx	
byOutp1Byte0	6200	01	Rx	Incoming bytes write on the image of the binary outputs (e.g. binary I/O option card) Option card KU- oder KW-EA2 in slot 1 O1...O8 are controlled by the variable byOutp1Byte0 [2)]
byOutp1Byte1	6200	02	Rx	
byOutp1Byte2	6200	03	Rx	
byOutp1Byte3	6200	04	Rx	
byOutp1Byte4	6200	05	Rx	
byOutp1Byte5	6200	06	Rx	
byOutp1Byte6	6200	07	Rx	
byOutp1Byte7	6200	08	Rx	

[1)] Binary outputs can be controlled with API variables if they are not used by the basic system. Please set following parameter: ID32864=0 Output port address 3

[2)] Binary outputs can be controlled with API variables if they are not used by the basic system. Please set following parameter: ID32846=0 Output port address 1

5.5 Diagnostic messages

<table>
<tr><th>API variable name</th><th>CAN Index [hex]</th><th>CAN Sub-Index [hex]</th><th>Copy-direction</th><th>Application</th></tr>
<tr><td>wSysWarn</td><td>2048</td><td>00</td><td>Tx</td><td>
<table>
<tr><td colspan="2">Status bits</td></tr>
<tr><td>BIT</td><td>MEANING</td></tr>
<tr><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>Overtemperature external components
R0x: Overtemperature motor (Motor PTC resistor)
KE, KU: Overtemperature brake resistor</td></tr>
<tr><td>2 ...5</td><td>Reserved</td></tr>
<tr><td>6</td><td>R0x: The measuring impulse on the input TPI (Touch probe input) of the option card KW-EN1or BE3on the controller card R0x is not detected within the window ID34155</td></tr>
<tr><td>7...15</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>wAxisError</td><td>2105</td><td>00</td><td>Tx</td><td>
<table>
<tr><td colspan="2">Valid for R0x</td></tr>
<tr><td>BIT</td><td>MEANING</td></tr>
<tr><td>0</td><td>Current higher than nominal current (I²t-monitoring)</td></tr>
<tr><td>1</td><td>Reserved</td></tr>
<tr><td>2</td><td>Overtemperature motor (Motor PTC resistor)</td></tr>
<tr><td>3</td><td>Reserved</td></tr>
<tr><td>4</td><td>Error logic voltage in the power supply</td></tr>
<tr><td>5</td><td>Encoder error</td></tr>
<tr><td>6</td><td>Error software commutation</td></tr>
<tr><td>7</td><td>Short circuit / earth fault motor- / motor terminal</td></tr>
<tr><td>8</td><td>Overvoltage DC circuit</td></tr>
<tr><td>9</td><td>Undervoltage</td></tr>
<tr><td>10</td><td>Reserved</td></tr>
<tr><td>11</td><td>Error positioning control (Excessive error)</td></tr>
<tr><td>12</td><td>Communication error SERCOSinterface</td></tr>
<tr><td>13</td><td>Position software limit value (software limit switch) exceeded</td></tr>
<tr><td>14</td><td>Reserved</td></tr>
<tr><td>15</td><td>Manufacturer specific error (see diagnosis documentation)</td></tr>
</table>
</td></tr>
<tr><td>byErrorSys</td><td>204C</td><td>01</td><td>Tx</td><td rowspan="2">The evaluation of this two bytes are used to discern error and warning status of the system. There is an error if one of this two bytes is not equal zero and the "System Ready" bit (SBM) is FALSE. There is a warning if one of this two bytes is not equal zero and the "System Ready" bit (SBM) is TRUE. The evaluation can be programmed e.g. in the AMK-PLC By the way of reading ID390 diagnosis number or ID32840 diagnosis list the diagnosis number according to the documentation "diagnosis messages" can be read out.</td></tr>
<tr><td>byErrorOpt</td><td>204C</td><td>02</td><td>Tx</td></tr>
</table>

5.6 Analog outputs write

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
iAnalogOutp1	6411	01	Rx	Analog output value 1
iAnalogOutp2	6411	02	Rx	Analog output value 2
iAnalogOutp3	6411	03	Rx	Analog output value 3
iAnalogOutp4	6411	04	Rx	Analog output value 4

Note: Analog outputs become available with the option KW-SM1 part-no.: 29509.

5.7 Configurable 4x32Bit Message

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
diSetpointScr1	2050	01	Tx	Content 1. Message from the configurable 4x32Bit message (ID32948)
diSetpointScr2	2050	02	Tx	Content 2. Message from the configurable 4x32Bit message (ID32948)
diSetpointScr3	2050	03	Tx	Content 3. Message from the configurable 4x32Bit message (ID32948)
diSetpointScr4	2050	04	Tx	Content 4. Message from the configurable 4x32Bit message (ID32948)

5.8 AFP Protocol for drive commanding

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
lwAFPReadBlock	2020	01-08	Tx	8Byte AFP status data
lwAFPWriteBlock	2021	01-08	Rx	8Byte AFP control data

5.9 Feed-forward control variables setpoint list - and actual value list

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
iSetpoint0	2080	01	Rx	ID32838 setpoint list
iSetpoint1	2080	02	Rx	ID32838 setpoint list
iSetpoint2	2080	03	Rx	ID32838 setpoint list
iSetpoint3	2080	04	Rx	ID32838 setpoint list
diSetpoint0	2081	01	Rx	ID32838 setpoint list
diSetpoint1	2081	02	Rx	ID32838 setpoint list
iActValue0	2090	01	Tx	ID32839 actual value list
iActValue1	2090	02	Tx	ID32839 actual value list
diActValue0	2091	01	Tx	ID32839 actual value list
diActValue1	2091	02	Tx	ID32839 actual value list

5.10 Field bus variables

Field bus variables are used by the plc to exchange data with other plc´s or drives. The plc program writes values to the transmit variables and evaluates receive data.

<table>
<tr><th>API variable name</th><th>CAN Index [hex]</th><th>CAN Sub-Index [hex]</th><th>Copy-direction</th><th>Application</th></tr>
<tr><td>lwInX
lwOutX</td><td>1)</td><td></td><td>Rx
Tx</td><td>
<table>
<tr><td colspan="2">Longword asynchronous (64Bit)</td></tr>
<tr><td>Hardware</td><td>Index X [LWORD]</td></tr>
<tr><td>KU-/KW-R03P</td><td rowspan="4">1 ...31</td></tr>
<tr><td>KU-/KW-PLC</td></tr>
<tr><td>KU-/KW-PLC1</td></tr>
<tr><td>KU-/KW-PLC2</td></tr>
</table>
</td></tr>
<tr><td>dwInX
dwOutX</td><td>2000
200C</td><td>01+X
01+X</td><td>Rx
Tx</td><td>
<table>
<tr><td colspan="2">Doubleword asynchronous (32Bit)</td></tr>
<tr><td>Hardware</td><td>Index X [DWORD]</td></tr>
<tr><td>KU-/KW-R03P</td><td rowspan="4">2 ...63</td></tr>
<tr><td>KU-/KW-PLC</td></tr>
<tr><td>KU-/KW-PLC1</td></tr>
<tr><td>KU-/KW-PLC2</td></tr>
</table>
</td></tr>
<tr><td>wInX
wOutX</td><td>2001
200D</td><td>01+X
01+X</td><td>Rx
Tx</td><td>
<table>
<tr><td colspan="2">Word asynchronous (16Bit)</td></tr>
<tr><td>Hardware</td><td>Index X [WORD]</td></tr>
<tr><td>KU-/KW-R03P</td><td rowspan="4">4 ...127</td></tr>
<tr><td>KU-/KW-PLC</td></tr>
<tr><td>KU-/KW-PLC1</td></tr>
<tr><td>KU-/KW-PLC2</td></tr>
</table>
</td></tr>
<tr><td>byInX
byOutX</td><td>2002
200E</td><td>01+X
01+X</td><td>Rx
Tx</td><td>
<table>
<tr><td colspan="2">Byte asynchronous (8Bit)</td></tr>
<tr><td>Hardware</td><td>Index X [BYTE]</td></tr>
<tr><td>KU-/KW-R03P</td><td rowspan="4">8 ...255</td></tr>
<tr><td>KU-/KW-PLC</td></tr>
<tr><td>KU-/KW-PLC1</td></tr>
<tr><td>KU-/KW-PLC2</td></tr>
</table>
</td></tr>
</table>

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
lwSyncInX lwSyncOutX	1)		Rx Tx	Longword synchronous (64Bit))

Hardware	Index X [LWORD]
KU-/KW-R03P	0 ...3
KU-/KW-PLC	0 ...7
KU-/KW-PLC1	0 ...7
KU-/KW-PLC2	0 ...15

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
dwSyncInX dwSyncOutX	2003 200F	01+X 01+X	Rx Tx	Doubleword synchronous

Hardware	Index X [DWORD]
KU-/KW-R03P	0 ...7
KU-/KW-PLC	0 ...15
KU-/KW-PLC1	0 ...15
KU-/KW-PLC2	0 ...31

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
wSyncInX wSyncOutX	2004 2010	01+X 01+X	Rx Tx	Word synchronous

Hardware	Index X [WORD]
KU-/KW-R03P	0 ...15
KU-/KW-PLC	0 ...31
KU-/KW-PLC1	0 ...31
KU-/KW-PLC2	0 ...63

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
bySyncInX bySyncOutX	2005 2011	01+X 01+X	Rx Tx	Byte synchronous

Hardware	Index X [BYTE]
KU-/KW-R03P	0 ...31
KU-/KW-PLC	0 ...63
KU-/KW-PLC1	0 ...63
KU-/KW-PLC2	0 ...127

1)lw (longword) is AMK internally composed out of bytesà byte mapping, such as lwOut10 starts with byOut80

5.11 Dummy variables

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
dwDummy	0004	00	Rx	Receive PDO Mapping without data content, dummy variables
wDummy	0003	00	Rx	
byDummy	0002	00	Rx	

5.12 Parallel connection servo controller

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
uwFieldAngleMaster	2112	00	Tx	Field angle master
swIsqMaster	2113	00	Tx	ISQ Master (Current setpoint)

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
swIsdMaster	2114	00	TX	ISD Master (current setpoint)
uwFieldAngleSlave	2115	00	Rx	Field angle slave
swIsqSlave	0003	00	Rx	ISQ Slave (Current setpoint)
swIsdSlave	0002	00	Rx	ISD Slave (Current setpoint)

6 API variables for KWZ two-axis inverter module

6.1 Receive of setpoints

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
diMainSetpoint	2030	01	Rx	32 bit set point, relevant in the operation modes ID32800...
diReserve1	2030	02	Rx	reserved
iAddSetpoint16	2030	03	Rx	16Bit incremental position setpoint, relevant in the operation modes ID32800…
diReserve2	2030	04	Rx	reserved
diAddSetpoint32	2030	05	Rx	32Bit incremental set point for synchronous mode with angle alignment (SWQW), relevant in the operation modes ID32800…
iSetSpeed	2030	06	Rx	Speed setpoint 16 Bit in (rpm)

6.2 Transmit of actual values

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
iMessage16	2040	01	Tx	Configurable cyclic 16bit actual value message, see.ID32785
iMessage32	2040	02	Tx	Configurable cyclic 32bit actual value message, see.ID32785
diActPosition	2040	03	Tx	32Bit actual position value

6.3 Status and control variables

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
wDeviceState	2048	00	Tx	Status bits

BIT	SYNTAX	MEANING
0	SBM	System ready message
1	QUE	Acknowledgement inverter on (only KU)
2 ...3	-	Reserved
4	-	Reserved
5		Acknowledgement pulse width measurement
6	PR_POS	Acknowledgement probe pos. edge
7	PR_NEG	Acknowledgement probe neg. edge
8	QRF	Controller enable
9 ...15	-	Reserved

<table>
<tr><th>API variable name</th><th>CAN Index [hex]</th><th>CAN Sub-Index [hex]</th><th>Copy-direction</th><th>Application</th></tr>
<tr><td>wDeviceControl</td><td>2049</td><td>00</td><td>Rx</td><td>
<table>
<tr><td colspan="3">Control bits</td></tr>
<tr><th>BIT</th><th>SYNTAX</th><th>MEANING</th></tr>
<tr><td>0</td><td>FL</td><td>Delete error</td></tr>
<tr><td>1</td><td>UE</td><td>Inverter on [1)]
(only KU)</td></tr>
<tr><td>2 ...4</td><td>-</td><td>Reserved</td></tr>
<tr><td>5</td><td></td><td>Start pulse width measurement</td></tr>
<tr><td>6</td><td>CLR-I</td><td>Clears the integral component of the speed controller</td></tr>
<tr><td>7</td><td>PR_EN</td><td>Enable measuring function</td></tr>
<tr><td>8</td><td>RF</td><td>Controller enable [1)]</td></tr>
<tr><td>9</td><td></td><td>Delete residual distance</td></tr>
<tr><td>10...15</td><td>-</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>wRealTimeBits</td><td>204A</td><td>00</td><td>Tx</td><td>
<table>
<tr><td colspan="2">Real time bit messages</td></tr>
<tr><th>BIT</th><th>MEANING</th></tr>
<tr><td>0</td><td>$| n_{set} - n_{act} | < n_{window}$ ($n_{act} = n_{set}$), window according ID157</td></tr>
<tr><td>1</td><td>$| n_{act} | < n_{min}$, threshold acc. ID124</td></tr>
<tr><td>2</td><td>$| n_{act} | < n_x$, threshold acc. ID125</td></tr>
<tr><td>3</td><td>$| M_{act} | > M_x$, threshold acc. ID126</td></tr>
<tr><td>4</td><td>$| M_{act} | > M_{limit}$, limit acc. ID82, ID83</td></tr>
<tr><td>5</td><td>$| n_{set} | > | n_{limit} |$, limit acc. ID38, ID39</td></tr>
<tr><td>6</td><td>In position window acc. ID57</td></tr>
<tr><td>7</td><td>$| P_{act} | ^3 P_x$, threshold acc. ID158</td></tr>
<tr><td>8</td><td>Negative position limit acc. ID50</td></tr>
<tr><td>9</td><td>Drive angle synchronous, window acc. ID228</td></tr>
<tr><td>10</td><td>Drive speed synchronous, windows acc. ID32952</td></tr>
<tr><td>11</td><td>$N_{act} \geq 0$</td></tr>
<tr><td>12</td><td>Acknowledgement actual value scaled</td></tr>
<tr><td>13</td><td>ID32922 residual distance window reset</td></tr>
<tr><td>14</td><td>Overcurrent message: capacity > ID32999 (50%)</td></tr>
<tr><td>15</td><td>Positive position limit ID49</td></tr>
<tr><td colspan="2">Caution:
N speed, M torque,
P power, X position</td></tr>
</table>
</td></tr>
<tr><td>byAxisState</td><td>204D</td><td>00</td><td>Tx</td><td>
<table>
<tr><td colspan="3">Status bits</td></tr>
<tr><th>BIT</th><th>SYNTAX</th><th>MEANING</th></tr>
<tr><td>0</td><td>REF_ACK</td><td>Reference point known</td></tr>
<tr><td>1 ...7</td><td>-</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>wStatusBitsId144</td><td>204E</td><td>00</td><td>Tx</td><td>Free configurable status word for displaying real time bits out of the code list for binary output assignment (See also documentation Parameter, ID144 status bits). The content of ID144 will be defined via ID26 "configuration list status bits"</td></tr>
</table>

[1)]To control the signals UE and RF via API variables, the following parameter setting is necessary:

ID32795=5 Source UE
ID32796=5 Source RF

6.4 Binary inputs / outputs

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
bySystemIn	2060	00	Tx	Sends image of the binary inputs BE1, BE2 and BE3 on the controller card
bySystemOut	2070	00	Rx	Receives image for the binary outputs BA1, BA2 and BA3 on the controller card [1)]
byInp1Byte0	6000	01	Tx	Each sent byte makes the image of 8 binary input bits available
byInp1Byte1	6000	02	Tx	
byOutp1Byte0	6200	01	Rx	Incoming bytes write on the image of the binary outputs [2)]
byOutp1Byte1	6200	02	Rx	

[1)] Binary outputs can be controlled with API variables if they are not used by the basic system. Please set following parameter: ID32864=0 Output port address 3

[2)] Binary outputs can be controlled with API variables if they are not used by the basic system. Please set following parameter: ID32846=0 Output port address 1

6.5 Diagnostic messages

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
byErrorSys	204C	01	Tx	The evaluation of this two bytes are used to discern error and warning status of the system. There is an error if this byte is not equal zero and the "System Ready" bit (SBM) is FALSE. There is a warning if this byte is not equal zero and the "System Ready" bit (SBM) is TRUE. The evaluation can be programmed e.g. in the AMK-PLC By the way of reading ID390 diagnosis number or ID32840 diagnosis list the diagnosis number according to the documentation "diagnosis messages" can be read out.

6.6 Configurable 4x32Bit Message

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
diSetpointScr1	2050	01	Tx	Content 1. Message from the configurable 4x32Bit message (ID32948)

6.7 AFP Protocol for drive commanding

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
lwAFPReadBlock	2020	01-08	Tx	8Byte AFP status data
lwAFPWriteBlock	2021	01-08	Rx	8Byte AFP control data

6.8 Feed-forward control variables setpoint list - and actual value list

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
iSetpoint0*	2080	01	Rx	ID32838 setpoint list
iSetpoint1*	2080	02	Rx	ID32838 setpoint list
iSetpoint2*	2080	03	Rx	ID32838 setpoint list
iSetpoint3*	2080	04	Rx	ID32838 setpoint list
diSetpoint0*	2081	01	Rx	ID32838 setpoint list
diSetpoint1*	2081	02	Rx	ID32838 setpoint list
iActValue0*	2090	01	Tx	ID32839 actual value list
iActValue1*	2090	02	Tx	ID32839 actual value list
diActValue0*	2091	01	Tx	ID32839 actual value list
diActValue1*	2091	02	Tx	ID32839 actual value list

* In preparation

7 API variables for KWF Double Frequency Inverters

Note: At the KWF module both transmit variables (diMainSetpoint and wDeviceControl) must be mapped in one message. It is not allowed to map only one of the transmit variables.

7.1 Receive of setpoints

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
diMainSetpoint	2030	01	Rx	speed setpoint in 1/10000 rpm

7.2 Status and control variables

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
wDeviceState	2048	00	Tx	Status bits

BIT	SYNTAX	MEANING
0	SBM	System ready message
1...3	-	Reserved
4	WARN	Warning
5	ERR	Error
6...7	-	Reserved
8	QRF	Controller enable
9 ...15	-	Reserved

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
wDeviceControl	2049	00	Rx	Control bits

BIT	SYNTAX	MEANING
0	FL	Delete error
1...7	-	Reserved
8	RF	Controller enable [1)]
9...15		Reserved

[1)] To control the signal RF via API variable, the following parameter setting is necessary:
ID32796=5 Source RF

8 API variables for the KE modules

Note: At the KE module all three transmit variables (iMessage16, diMessage32 und wDeviceState) must be mapped in one message. It is not allowed to map only one or two of the transmit variables.The PDO is event driven transmission type EVENT). Smalest possible cycle time is 10 ms for this message.

8.1 Transmit of actual values

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
iMessage16	2040	01	Tx	Configurable cyclic 16bit actual value message, see ID32785 Configurable values: 32836 DC-bus voltage 33101 Inverter overload indication 33116 Temperature internal 34144 Nominal voltage effective 34145 Line current effective
iMessage32	2040	02	Tx	Configurable cyclic 32bit actual value message, see ID32786

8.2 Status and control variables

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
wDeviceState	2048	00	Tx	Status bits

BIT	SYNTAX	MEANING
0	SBM	System ready message
1	QUE	Acknowledgement inverter on
2	QFL	Acknowledgement Clear Error
3	-	Reserved
4	WRN	KE warning message
5	ERR	KE error message
6...15	-	Reserved

wDeviceControl	2049	00	Rx	Control bits

BIT	SYNTAX	MEANING
0	FL	Delete error
1	UE	Inverter on
2 ...15	-	Reserved

9 API variables for AMKASMART IDT motor with integrated servo controller

9.1 Receive of setpoints

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
diSetSpeed	2030	01	Rx	32bit speed setpoint
iSetTorque	2030	03	Rx	16bit torque setpoint
iSetPosition	2030	04	Rx	16bit position setpoint
diSetPosition	2030	05	Rx	32bit position setpoint
iSetSpeed	2030	06	RX	16bit speed setpoint

9.2 Transmit of actual values

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
iMessage16	2040	01	Tx	Configurable cyclic 16bit actual value message, see ID32785
iMessage32	2040	02	Tx	Configurable cyclic 32bit actual value message, see ID32786
diActPosition	2040	03	Tx	32Bit actual position value

9.3 Status and control variables

<table>
<tr><th>API variable name</th><th>CAN Index [hex]</th><th>CAN Sub-Index [hex]</th><th>Copy-direction</th><th>Application</th></tr>
<tr><td>wDeviceState</td><td>2048</td><td>00</td><td>Tx</td><td>
<table>
<tr><td colspan="3">Status bits</td></tr>
<tr><th>BIT</th><th>SYNTAX</th><th>MEANING</th></tr>
<tr><td>0</td><td>SBM</td><td>System ready message</td></tr>
<tr><td>1</td><td>QUE</td><td>Acknowledgement
48V supply ON</td></tr>
<tr><td>2 ...7</td><td>-</td><td>Reserved</td></tr>
<tr><td>8</td><td>QRF</td><td>Controller enable</td></tr>
<tr><td>9 ...15</td><td>-</td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>wDeviceControl</td><td>2049</td><td>00</td><td>Rx</td><td>
<table>
<tr><td colspan="3">Control bits</td></tr>
<tr><th>BIT</th><th>SYNTAX</th><th>MEANING</th></tr>
<tr><td>0</td><td>FL</td><td>Delete error</td></tr>
<tr><td>1...7</td><td>-</td><td>Reserved</td></tr>
<tr><td>8</td><td>RF</td><td>Controller enable 1)</td></tr>
<tr><td>9...15</td><td></td><td>Reserved</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><th>API variable name</th><th>CAN Index [hex]</th><th>CAN Sub-Index [hex]</th><th>Copy-direction</th><th>Application</th></tr>
<tr><td>wRealTimeBits</td><td>204A</td><td>00</td><td>Tx</td><td>
<table>
<tr><td colspan="2">Real time bit messages</td></tr>
<tr><td>BIT</td><td>MEANING</td></tr>
<tr><td>0</td><td>$| n_{set} - n_{act} | < n_{window}$ ($n_{act} = n_{set}$), window according ID157</td></tr>
<tr><td>1</td><td>$| n_{act} | < n_{min}$, threshold acc. ID124</td></tr>
<tr><td>2</td><td>$| n_{act} | < n_x$, threshold acc. ID125</td></tr>
<tr><td>3</td><td>$| M_{act} | > M_x$, threshold acc. ID126</td></tr>
<tr><td>4</td><td>$| M_{act} | > M_{limit}$, limit acc. ID82, ID83</td></tr>
<tr><td>5</td><td>$| n_{set} | > | n_{limit} |$, limit acc. ID38, ID39</td></tr>
<tr><td>6</td><td>In position window acc. ID57</td></tr>
<tr><td>7</td><td>$| P_{act} | \geq P_x$, threshold acc. ID158</td></tr>
<tr><td>8</td><td>Negative position limit acc. ID50</td></tr>
<tr><td>9</td><td>Drive angle synchronous, window acc. ID228</td></tr>
<tr><td>10</td><td>Drive speed synchronous, windows acc. ID32952</td></tr>
<tr><td>11</td><td>$N_{act} \geq 0$</td></tr>
<tr><td>12</td><td>Acknowledgement actual value scaled</td></tr>
<tr><td>13</td><td>Reserved</td></tr>
<tr><td>14</td><td>Overcurrent message: capacity > ID32999 (50%)</td></tr>
<tr><td>15</td><td>Positive position limit ID49</td></tr>
<tr><td colspan="2">caption:
N speed, M torque,
P power X position</td></tr>
</table>
</td></tr>
<tr><td>byAxisState</td><td>204D</td><td>00</td><td>Tx</td><td>
<table>
<tr><td colspan="3">Status bits</td></tr>
<tr><td>BIT</td><td>SYNTAX</td><td>MEANING</td></tr>
<tr><td>0</td><td>REF_ACK</td><td>Reference point known</td></tr>
<tr><td>1 ...7</td><td></td><td>Reserved</td></tr>
</table>
</td></tr>
<tr><td>wStatusBitsId144</td><td>204E</td><td>00</td><td>Tx</td><td>Free configurable status word for displaying real time bits out of the code list for binary output assignment (See also documentation Parameter, ID144 status bits). The content of ID144 will be defined via ID26 "configuration list status bits"</td></tr>
</table>

[1)] To control the signal RF via API variable, the following parameter setting is necessary:

ID32795=5 Source UE

ID32796=5 Source RF

9.4 Binary inputs/outputs

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
bySystemIn	2060	00	Tx	The image of “binary inputs BE1, BE2 and BE3 is sent
bySystemOut	2070	00	Rx	The image of “binary outputs” is received to issue on binary outputs BA1, BA2 and BA3 of the controller card [1)]
byInp1Byte0	6000	01	Tx	The image of 1 byte „”binary inputs“ is copied to the binary input port 1 (ID32873-ID32881)

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
byOutp1Byte0	6200	01	Rx	The incoming byte “binary outputs image” are issued on the binary outputs (e.g. I/O option card KU- or KW-EA2 in slot 1. A1...A8 are controlled by the variable byOutp1Byte0. 2)

1) binary outputs can be controlled by API variables if they are not used by the basic system. The following parameter setting is necessary: ID32864=0 output port address 3

2) binary outputs can be controlled by API variables if they are not used by the basic system. The following parameter setting is necessary: ID32846=0 output port address 1

9.5 Diagnostic messages

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
byErrorSys	204C	01	Tx	The evaluation of this byte is used to discern error and warning status of the system. There is an error if this byte is not equal zero and the "System Ready" bit (SBM) is FALSE. There is a warning if this byte is not equal zero and the "System Ready" bit (SBM) is TRUE. The evaluation can be programmed e.g. in the AMK-PLC By reading ID390 diagnosis number or ID32840 diagnosis list the diagnosis number according to the documentation "diagnosis messages" can be read out.

9.6 Configurable 4x32Bit message

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
diSetpointScr1	2050	01	Tx	Content 1. Message from the configurable 4x32Bit message (ID32948) (Only the counter value (High word) is transferred, because the rectangular input of the IDT doesn't have a zero pulse)

9.7 AFP protocol for drive commanding

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
lwAFPReadBlock	2020	01-08	Tx	8Byte AFP status data
lwAFPWriteBlock	2021	01-08	Rx	8byte AFP control data to command the drive via AMK field bus protocol (see Documentation AFP T.-Nr.:27872)

9.8 Feed-forward control variables setpoint list - and actual value list

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
iSetpoint0	2080	01	Rx	ID32838 setpoint list
iSetpoint1	2080	02	Rx	ID32838 setpoint list
iSetpoint2	2080	03	Rx	ID32838 setpoint list
iSetpoint3 [1)]	2080	04	Rx	ID32838 setpoint list
diSetpoint0 [1)]	2081	01	Rx	ID32838 setpoint list
diSetpoint1 [1)]	2081	02	Rx	ID32838 setpoint list
iActValue0 [1)]	2090	01	Tx	ID32839 actual value list
iActValue1 [1)]	2090	02	Tx	ID32839 actual value list
diActValue0	2091	01	Tx	ID32839 actual value list
diActValue1	2091	02	Tx	ID32839 actual value list

[1)] under construction

10 API variables for the AMKAMAC controllers AS-PL12, AS-PL14 and AS-C

Field bus variables are used by the plc to exchange data with other plc´s or drives. The plc programs write values to the transmit variables and evaluate receive data.

API variable name	CAN Index [hex]	CAN Sub-Index [hex]	Copy-direction	Application
lwInX lwOutX	1)		Rx Tx	Longword asynchronous (64Bit) **Hardware** / **Index X [LWORD]** AS-PL12 + AS-FCT1: 0 ...63 AS-PL14 + ASFCT2: 0 ...63 AS-C + AS-FCT1: 0 ...63
dwInX dwOutX	2000 200C	01+X 01+X	Rx Tx	Doubleword asynchronous (32Bit) **Hardware** / **Index X [DWORD]** AS-PL12 + AS-FCT1: 0 ...127 AS-PL14 + ASFCT2: 0 ...127 AS-C + AS-FCT1: 0 ...127
wInX wOutX	2001 200D	01+X 01+X	Rx Tx	Word asynchronous (16Bit) **Hardware** / **Index X [DWORD]** AS-PL12 + AS-FCT1: 0 ...254 AS-PL14 + ASFCT2: 0 ...254 AS-C + AS-FCT1: 0 ...254
byInX byOutX	2002 2006 200E 2012	01+X X-254 01+X X-254	Rx Rx Tx Tx	Byte asynchronous (8Bit) **Hardware** / **Index X [DWORD]** AS-PL12 + AS-FCT1: 0 ...254, 255... 509 AS-PL14 + ASFCT2: 0 ...254, 255... 509 AS-C + AS-FCT1: 0 ...254, 255... 509
lwSyncInX lwSyncOutX	1)		Rx Tx	Longword synchronous (64Bit) **Hardware** / **Index X [DWORD]** AS-PL12 + AS-FCT1: 0 ...31 AS-PL14 + ASFCT2: 0 ...63 AS-C + AS-FCT1: 0 ...15
dwSyncInX dwSyncOutX	2003 200F	01+X 01+X	Rx Tx	Doubleword synchronous **Hardware** / **Index X [DWORD]** AS-PL12 + AS-FCT1: 0 ...63 AS-PL14 + ASFCT2: 0 ...127 AS-C + AS-FCT1: 0 ...31
wSyncInX wSyncOutX	2004 2010	01+X 01+X	Rx Tx	Word synchronous **Hardware** / **Index X [DWORD]** AS-PL12 + AS-FCT1: 0 ...127 AS-PL14 + ASFCT2: 0 ....254 AS-C + AS-FCT1: 0 ...63
bySyncInX bySyncOutX	2005 2007 2011 2013 *	01+X X-254 01+X X-254	Rx Rx Tx Tx	Byte synchronous **Hardware** / **Index X [BYTE]** AS-PL12 + AS-FCT1: 0 ...254, *255... 509 AS-PL14 + ASFCT2: 0 ...254, 255... 509 AS-C + AS-FCT1: 0 ...127

1) lw (longword) is AMK internally composed out of bytesà byte mapping, such as lwOut10 starts with byOut80

*not supported

Your opinion is important!

With our documentation we want to offer you the highest quality support in handling the AMKmotion products.
That is why we are now working on optimizing our documentation.
Your comments or suggestions are always of interest to us.
We would be grateful if you take a bit of time and answer our questions. Please return a copy of this page to us.

e-mail: Documentation@amk-motion.com
or
fax no.: +49 7021/50 05-199

Thank you for your assistance.
Your AMKmotion documentation team

1. How would you rate the layout of our AMKmotion documentation?

   (1) very good (2) good (3) satisfactory (4) less than satisfactory (5) poor

2. Is the content structured well?

   (1) very good (2) good (3) moderate (4) hardly (5) not at all

3. How easy is it to understand the documentation?

   (1) very easy (2) easy (3) moderately easy (4) difficult (5) extremely difficult

4. Did you miss any topics in the documentation?

   (1) no (2) if yes, which ones:

5. How would you rate the overall service at AMKmotion?

   (1) very good (2) good (3) satisfactory (4) less than satisfactory (5) poor

AMKmotion GmbH + Co KG
Phone : +49 7021/50 05-0, fax: +49 7021/50 05-199
E-Mail: info@amk-motion.com
Homepage: www.amk-motion.com