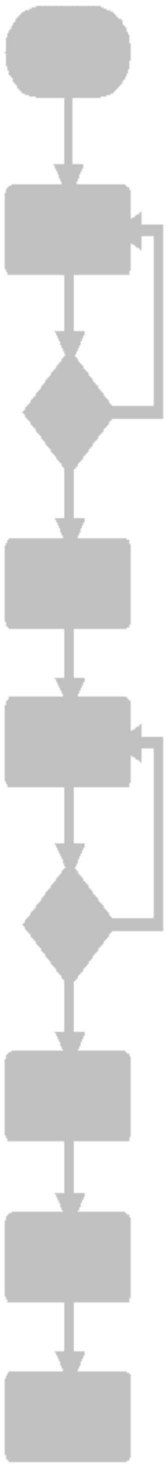


JetControl 24x

Version Update
from V3.22 to V3.23

Jetter
Automation. Made easy.

Rev. 1.00

April 2007 / Printed in Germany

Table of Contents

1 Introduction

Overview of Version Updates			
Version	**Description**	**New**	**Fixed**
3.23	Operating System Update	✓	
	Modbus/TCP	✓	
	IP address setting		✓
	System bus	✓	✓
3.22	Flags overlaid over registers	✓	
	E-Mail	✓	
	Operating System Update		✓
	Flash disk		✓
3.21	Serial interfaces	✓	✓
	System command	✓	
	Timer instructions	✓	
	Special register(s)	✓	
	Ethernet / Networking	✓	✓
	System bus		✓

1.1 Update Information

The programming tool JetSym version 3.0 or higher is required to fully support all functions of the JX3 I/O system.

2 New Features

2.1 Modbus/TCP

The instruction header of a Modbus frame contains a "Unit ID", which is not evaluated by Modbus/TCP devices, as they can be addressed without ambiguity by their IP address. So far, the Modbus/TCP client implementation used the fixed value "1" as "Unit ID". Converters from Modbus/TCP to Modbus RTU use the "Unit ID" for addressing the Modbus RTU servers. Therefore, we have provided new special functions for reading and writing registers, as well as for initializing the RemoteScan. These special functions can be used to set the "Unit ID".

2.1.1 Function 67: Reading registers

This function corresponds to function 65 with additional "Unit ID". Function 65 can be used as before.

JetSym	`SPECIALFUNCTION (67, Parameter1, Parameter2)`
JetSym ST	`SYSTEMFUNCTION (67, Parameter1, Parameter2);`

Operating Principle: Using this function, a register block from a Modbus/TCP server is copied to the registers/variables of the local memory.

Parameter 1: Number of the first register/address of the first variable of a descriptive block.

Offset	Description	
0	IP address	
1	Port number	502
2	Timeout	in ms
3	Number of the source register	remote
4	Number of the designation register	local
5	Quantity of registers	1 .. 125
6	Unit ID	0 .. 255
7	Internal usage	0
8	Internal usage	0

Parameter 2: Number of the register / address of the variable resulting from the function.

Potential errors:

Error code	Description
0	No error
-1 or -2	Error during connection set-up

-4	Error during data transfer
-5	Error message from server
-8	Timeout
-10	No Modbus/TCP license

2.1.2 Function 68: Writing to Registers

This function corresponds to function 66 with additional "Unit ID". Function 66 can be used as before.

JetSym	`SPECIALFUNCTION (68, Parameter1, Parameter2)`
JetSym ST	`SYSTEMFUNCTION (68, Parameter1, Parameter2);`

Operating Principle: By using this function, the content of registers/variables of the local memory is copied to a register block a Modbus/TCP server.

Parameter 1: Number of the first register/address of the first variable of a descriptive block.

Offset	Description	
0	IP address	
1	Port number	502
2	Timeout	in ms
3	Number of the source register	local
4	Number of the designation register	remote
5	Quantity of registers	1 .. 125
6	Unit ID	0 .. 255
7	Internal usage	0
8	Internal usage	0

Parameter 2: Number of the register / address of the variable resulting from the function.

Potential errors:

Error code	Description
0	No error
-1 or -2	Error during connection set-up
-3	Error during data transfer
-5	Error message from server
-8	Timeout
-10	No Modbus/TCP license

2.1.3 Function 85: Configuring RemoteScan

This function corresponds to function 80 with additional "Unit ID". Function 80 can be used as before.

JetSym	`SPECIALFUNCTION (85, Parameter1, Parameter2)`
JetSym ST	`SYSTEMFUNCTION (85, Parameter1, Parameter2);`

Operating Principle: To configure RemoteScan special/system function 85 is used. However, this function does not initiate communication.

Parameter 1: Number of the first register/address of the first variable of a descriptive block.

Offset	Description	
0	Protocol	5 = Modbus/TCP
1	Number of communication units to follow	1 .. 10
	Communication unit 1	
2	Address	Modbus/TCP: IP address
3	Port number	Modbus/TCP: 502
4	Update rate	10 .. 65535 ms
5	Quantity of output registers	0 .. 125
6	Output source register number	local
7	Output destination register number	remote
8	Quantity of input registers	0 .. 125
9	Input source register number	remote
10	Input destination register number	local
11	Number of the first register of the status register block	
12	Timeout	in ms
13	Unit ID	0 .. 255
14	Internal usage	0
15	Internal usage	0
	Communication unit 2	

16	Address	Modbus/TCP: IP address
..	..	

Parameter 2: Number of the register / address of the variable resulting from the function.

Potential errors:

Error code	Description
0	No error
-1	Protocol not supported
-2	Set quantity of communication units > 10
-3	Invalid address or port number
-4	Invalid register number
-10	RemoteScan is already running

Status register block

The number of the first register of the status register block, consisting of 3 consecutive registers, has to be specified in the description block of each communication unit (Offset 11, 25). Error messages of this communication unit are stored to this block when RemoteScan is running.

The status register block has the following structure:

Reg. Offset	Description			
0	Status (bit-coded)	Bit #	Description	
		0	Scan is running	Is set after each update cycle
		1	An error has occurred	Is set each time an error has occurred
1	Error code	The code of the last error is displayed.		
		Value	Description	
		0	No error	
		< 0	Application-specific error	
			Modbus/TCP:	
			Value	Description
			-1	Error in the network driver
			-2	Error in the connection management
			-3	Error when sending output registers
			-4	Error when reading input registers
			-5	Exception response
			-6	Error when receiving the response
			-7	Wrong transaction ID
			-8	Timeout
		101	Timeout	
		102	Error when reading/writing local registers	
		103/ 104	Error in the lower-level communication layer	
2	Number of errors	The number is incremented each time an error occurs.		

Note!

For starting and stopping communication functions 81 and 82 are used as before.

2.2 Operating System Update

Special function **70** allows to update the OS of a module on the system bus.
The OS files must be stored to the root directory. Access to OS files is carried out with administrator rights and cannot be restricted.
The file names consist of two constant parts and a register content. So, files can be selected by different register values.

2.2.1 Special/System Functions

Implementation

Since OS updates may take considerable long time, other application tasks are processed while the update is running. However, only one update can be processed at a time. Tasks which invoke the update function while an update initiated by another task is running are therefore blocked until the update is completed.

During an OS update any register access to modules on the system bus is blocked. Write access will not be carried out; read access will return the value "0". Communication via serial interface or Ethernet will return "Timeout". Bit 3 and/or bit 4 in error register 2008 will be set which causes the red LED (ERR) to light up.

The state of the currently running operation can be polled through the registers specified below.

Note!

If you intend to update the OS of several modules, it is advisable to start updating with the last module and to proceed from here "towards the CPU".
Once all updates have been completed, the whole system must be restarted.

Updating the Operating System

JetSym `SPECIALFUNCTION (70, Parameter1, Parameter2)`

JetSym ST `SYSTEMFUNCTION (70, Parameter1, Parameter2);`

Operating Principle	Special function 70 allows to update the OS of a module on the system bus.
Parameter 1:	Number of the first register/address of the first variable of a descriptive block.

Offset	Description	
0	File Name	
1	Slot	0
2	Submodule	0
3	Slave	2 .. 9

Parameter 2:	Number of the register / address of the variable resulting from the function.

Potential errors:

Error code	Description
0	No error
-1	Invalid parameter
-2	Error when opening file (e.g. file not found)
-3	Error while reading file
-4	Error when closing file
-5	Invalid file type (no OS file)
-6	Wrong module type
-7 / -8	Error while transmitting data to the module
< -9	Internal errors

Functional parameters for Offset 1 and 2 (slot, submodule) are only used for other controllers of the JetControl series and should be set to "0" in the case of JetControl 24x.

2.2.2 File Names

File names always start with "osfile_" followed by a numerical value and the extension "os". The numerical value for drawing a distinction between various files is acquired from the first parameter register of the special function.

Example 1: Name of an OS file

```
osfile_123456789.os
osfile_0.os
```

Note!

Observe capitalization. The file system is case sensitive.

2.2.3 Registers

The processing state of an OS update and the number of the task carrying out the operation can be read from two registers.

Register 2975: Processing State	
Function	**Description**
Read	0: No OS update in progress 1: Processing transferred to update module 2: OS is being sent to module 3: OS update completed Value following reset: 0
Write	Illegal
Value range	0 .. 255

Register 2976: Task number	
Function	**Description**
Read	Number of the task performing an OS update 0 .. 99: Task number 255: No task Value following reset: 255
Write	Illegal
Value range	0 .. 255

2.3 System Bus

Starting from this OS version firmware version 1.35 of the Jetter system bus driver is integrated into the OS.

2.3.1 Special Register 2008: Error Messages

Furthermore, bit 13 in error register 2008 has been added: System bus initialization error. Possible reasons: Excess configuration (too many modules connected) or error when commissioning modules.

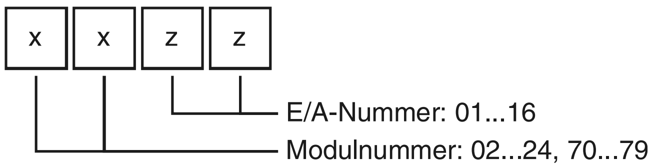
2.3.2 SDO Upload

So far, an error message was issued only in the case of a timeout. Now, on read access to registers which are not supported (e.g. analog configuration while no analog modules are connected) bit 4 in error status register 2008 is set, too, and the red LED "ERR" is lit. In the case of an access via communication interface (e.g. in JetSym) the status byte indicates "Timeout".

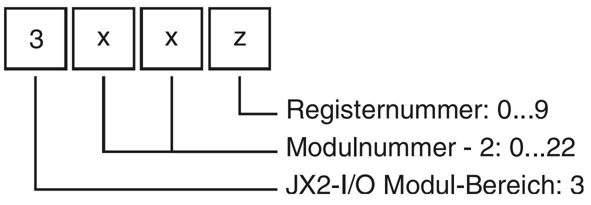
2.3.3 Compactbox LJX7-CSL

Apart from I/O module numbers 70 through 79, Compactbox modules can be set to module numbers 2 .. 24 (JC-243: 16; JC-241: 8). Thus, the number of modules that can be connected is no longer limited to 10 (JC-241: 7) and no addresses for JX-SIO modules or third party modules are occupied.

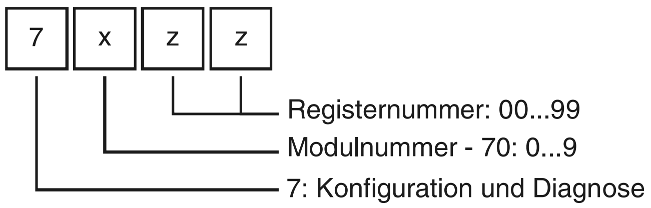
Coding of the Input / Output Number:



Coding register numbers for addresses ranging from 02 to 24



Coding register numbers for addresses ranging from 02 to 79



For more information on the Compactbox LJX7-CSL on the Jetter system bus refer to: ljx7_csl_bi_xxxx[a]_user_information.pdf

a.xxxx: current document revision

2.3.4 Milan Drives:

Starting from this OS version the Advanced models of Milan drives by Werner Riester GmbH & Co. KG are directly supported.

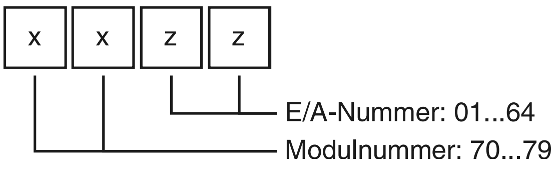
For more information on Milan Drives on the Jetter system bus refer to: AUMA_BI_xxxx[a]_user_information.pdf

a.xxxx: current document revision

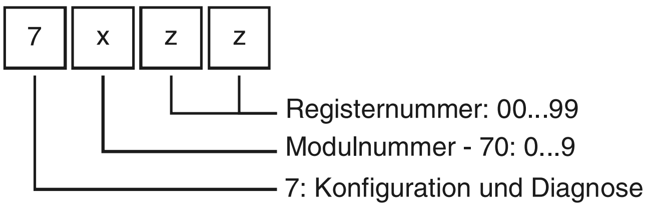
2.3.5 BWU 1821

The CANopen / AS-i gateway BWU 1821 by Bihl+Wiedemann GmbH is for connecting AS-i I/O modules to the Jetter system bus.

Coding of the Input / Output Number:



Coding of Register Numbers



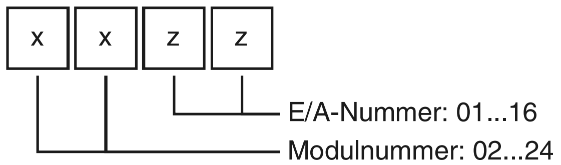
For more information on the AS-i master BWU1821 on the Jetter system bus refer to: BihlWiedemann_BI_xxxx[a]_user_information.pdf

a.xxxx: current document revision

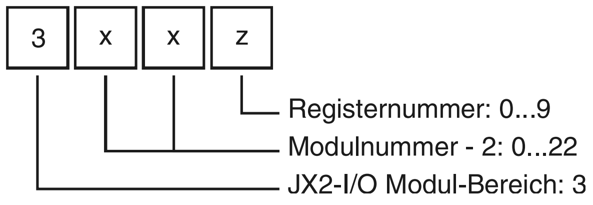
2.3.6 LioN-S

LioN-S modules are digital I/O modules with CANopen interface meeting IP67.

Coding of the Input / Output Number:



Coding of Register Numbers



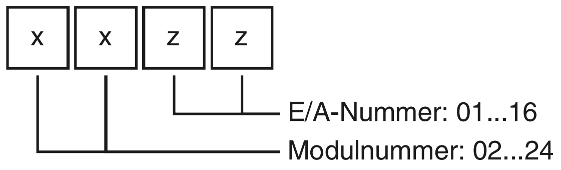
For more information on the LioN-S system on the Jetter system bus refer to: a_LioN_BI_xxxx[a]_user_information.pdf

a.xxxx: current document revision

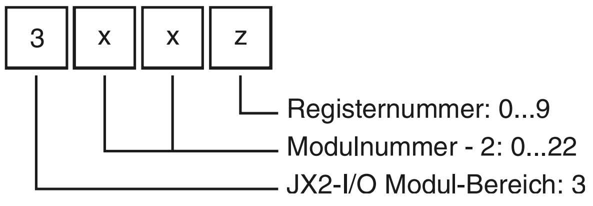
2.3.7 JX3

JX3 modules are compact, digital and analog I/O modules meeting IP20.

Coding of the Input / Output Number:



Coding of Register Numbers



For more information on the Jetter I/O system JX3 refer to:
jx3_io_system_bi_xxxx[a]_user_information.pdf

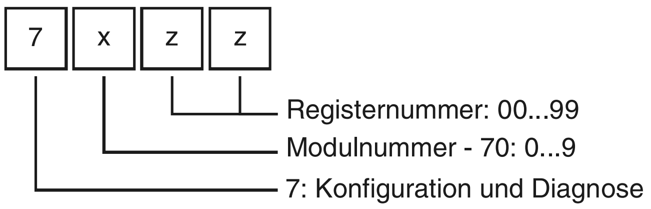
a.xxxx: current document revision

2.3.8 MAXON EPOS

The CAN IDs of the MAXON modules have changed. So far, the CAN IDs ranged from 1 to 10. Now, they have to be set to values from 70 to 79. The register numbers remain unchanged.

This modification is not downwards compatible.

Coding of Register Numbers



For more information on the MAXON EPOS positioning controller on the Jetter system bus refer to: maxon_epos_bi_xxxx[a]_user_information.pdf

a.xxxx: current document revision

3 Fixed Software Bugs

3.1 IP Address Settings

Once the controller is powered up, the Ethernet interface is initialized during the boot process based on the settings in the configuration memory and the position of the address switches. During initialization it is checked whether the IP address of the controller matches the IP of the default gateway considering the subnet mask. In OS versions V3.20 through V3.22 this check was carried out at a point in time when the position of the address switches was not read in yet. In that case it could occur, for example, that a wrong IP address was used for the controller, or that the default gateway was not set.

For more information on how the IP address of the JetControl 24x is set refer to chapter 5.10 in the corresponding manual.

3.2 System Bus

Starting from this OS version firmware version 1.35 of the Jetter system bus the driver is integrated into the OS.

3.2.1 Lenze FU 82xx Vector

When calculating the I/O sum, the Lenze FU 82xx Vector scored 32 I/O points. Starting from this OS version it scores 16 I/O points which is the correct value.

3.2.2 WAGO 750

Reading the data of input or output bytes of special terminals with a length of more than 7 bytes now returns the correct result.