

**JetWeb**

**JX6-SB / JX6-SB-I**

**User Information**



This document is to be used together with the appurtenant safety instructions.

Jetter AG reserves the right to make alterations to its products in the interest of technical progress. These alterations need not be documented in every single case.

This manual and the information contained herein have been compiled with due diligence. However, Jetter AG assumes no liability for printing or other errors or damages arising from such errors.

The brand names and product names used in this document are trademarks or registered trademarks of the respective title owner.

---

## Table of Contents

<b>1</b>	<b>Description</b>	<b>1</b>
1.1	Compatibility with the D-CAN2 Submodule	1
1.2	Quick Start	1
1.3	Product Description – JX6-SB Module	1
1.4	Product Description – JX6-SB-I Module	2
1.5	System Requirements	3
1.6	Technical Data	4
1.7	Third-party Modules	5
<b>2</b>	<b>Installation</b>	<b>6</b>
2.1	Connector Pin Assignment	6
2.2	Front Panel	7
2.3	Diagnostic Indicators (LEDs)	7
2.4	System Bus Cable	8
2.5	System Bus - Baud Rates	9
<b>3</b>	<b>Operating System Update</b>	<b>10</b>
3.1	OS Update - JX6-SB(-I) Module	10
3.2	OS Update – JX2-Slave Module	12
<b>4</b>	<b>Topology of the Jetter System Bus</b>	<b>14</b>
4.1	System Bus Topology for Master-Slave Operation	14
4.1.1	Distributed Arrangement on the System Bus	15
4.1.2	Connecting JX-SIO Modules to the System Bus	15
4.1.3	Assigning System Bus Module Numbers	16
4.2	System Bus Topology for Master-Master Operation	17
<b>5</b>	<b>Operating Modes</b>	<b>18</b>
<b>6</b>	<b>Getting Started</b>	<b>19</b>

---

6.1	Operating Mode "Master-Slave JX6-SB"	19
6.2	Operating Mode "Master-Master"	24
<b>7</b>	<b>Operating Mode „Master-Slave D-CAN2“</b>	<b>29</b>
7.1	Register Areas	29
7.1.1	Configuration and Status Registers	29
7.1.2	Register Area for JX2-I/O Modules	30
7.1.3	Register Area for JX2-Slave Modules	30
7.2	I/O Range	31
7.2.1	Access to Individual Inputs and Outputs	31
7.2.2	Access to Inputs and Outputs	32
7.2.3	Access to Inputs and Outputs by means of Register Overlaying	32
7.2.4	Addressing of the JX2-I/O Modules and of the Festo-CP-FB Modules	33
7.3	Description of Registers	35
7.4	Overview of Registers	54
<b>8</b>	<b>Operating Mode "Master-Slave JX6-SB"</b>	<b>56</b>
8.1	Overview of Registers	56
8.1.1	Configuration and Status Registers	57
8.1.2	System Bus Special Registers	57
8.1.3	Register Area for JX2-I/O Modules	58
8.1.4	Register Area for JX-SIO Modules	58
8.1.5	Register Area for JX2-Slave Modules	59
8.1.6	Register Area for Third-party Modules	59
8.2	Numbering of Modules	60
8.3	I/O Range	62
8.3.1	Access to Inputs and Outputs	62
8.3.2	Access to Inputs and Outputs by means of Register Overlaying	62
8.4	Description of Registers	73
8.4.1	Initialization and Diagnostics	73
8.4.2	Short-circuit in one of the JX2-I/O modules	79
8.4.3	Timeout Monitoring	79
8.4.4	Connected Expansion Modules	81
8.4.5	Dummy Module Configuration	85
8.4.6	Access to 32-bit JX-SIO Registers	88
8.5	I/O Data and Module Registers	90
8.5.1	JX2-I/O Modules	90

---

---

8.5.2	JX-SIO	92
8.5.3	JX2-Slave Modules	93
8.6	Timeout Adjustment for JX-SIO Modules	94
8.6.1	Commands 31 and 32	95
8.7	Special Flags	96
8.8	Overview of Registers	97
<b>9</b>	<b>JX2-Slave Modules connected to the JX6-SB-I Module</b>	<b>101</b>
9.1	Oscilloscope Mode in JetSym	101
9.2	Axis Programming	104
<b>10</b>	<b>Operating Mode "Master-Master"</b>	<b>105</b>
10.1	Description of Registers	105
10.2	Sample Programs	112
10.3	Signal Waveform when Receiving Data	118
10.4	Buffer Structure	120
10.5	Overview of Registers	121
<b>A</b>	<b>Explanation of Terms</b>	<b>i</b>
<b>B</b>	<b>Index of Illustrations</b>	<b>ii</b>
<b>C</b>	<b>Sample Programs</b>	<b>iii</b>
<b>D</b>	<b>Calculating the I/O Sum</b>	<b>iv</b>

## History

- |                        |   |
|------------------------|---|
| <b>Revision 1.00</b>   | First edition   |
| <b>Revision 1.10</b>   | Master-Master operation has been added<br>Sample programs have been expanded  |
| <b>Revision 2.01</b>   | JX2-Slave module has been added   |
| <b>Revision 2.10</b>   | Chapter "Getting Started" has been added<br>Product description, 3 <sup>rd</sup> party modules and system requirements have been supplemented to the chapter "Description"<br>JetSym dialog boxes have been added to OS update description<br>Mode "Master-Slave JX6-SB" has been supplemented<br>System bus configuration tables have been added to appendix |
| <b>Revision 2.11.1</b> | Chapter "JX2-Slave Module on the JX6-SB-I" has been added   |

# 1 Description

This user information gives a description of functions offered by the JX6-SB-I module with software version 2.11.



This user information is to be used together with the safety instructions and description of operating parameters belonging to the higher-level control system (D-CPU, D-CPU 200 or JetControl 647).

In future, this function description will be replaced by a supplemented and revised operator's manual.

## 1.1 Compatibility with the D-CAN2 Submodule

D-CAN2 submodules for D-CPU and D-CPU200 controllers are fully compatible with the JX6-SB submodule described in this manual. All OS versions for the JX6-SB submodule can also be transferred to a D-CAN2 submodule.

It is also possible to transfer all OS versions of the D-CAN2 submodule to a JX6-SB or JX6-SB-I submodule.

## 1.2 Quick Start

For experienced users we recommend to start with chapter 1 and then to continue with chapter 6 "Getting Started". In case of additional questions, this user information can assist you in commissioning a JX6-SB(-I) module.

The Jetter AG wish all users of the JX6-SB(-I) module success. We hope that this product will assist you in solving your technical problems.

## 1.3 Product Description – JX6-SB Module

The JX6-SB expansion module allows to connect all system bus modules produced by Jetter AG to Delta and JetControl 647 controllers. Its design is to Modulbus standard.

The JX6-SB module detects and supports all JX2-I/O and JX-SIO modules produced by Jetter AG. Modules other manufacturers, such as valve terminals of Festo AG & Co., can be connected as well.

The JX6-SB module can be initialized by the user in three different modes. The modes "Master-Slave D-CAN2" and "Master-Slave JX6-SB" are for connecting expansion modules.

The third mode which can be selected by the user during initialization is for networking several Delta and JetControl 647 controllers via JX6-SB modules. To do so, the JX6-SB module has to be initialized in "Master-Master" mode.

JX2-I/O and JX-SIO modules, as well as third-party modules can completely be integrated into Jetter control systems via a JX6-SB module. Via register interface the application program has transparent access to the modules connected to the system bus.

**The JX6-SB module allows the following maximum system bus configuration:**

- 31 JX2-I/O expansion modules
  - 248 digital inputs
  - 248 digital outputs
  - 124 analog inputs
  - 124 analog outputs
  - 62 hardware counters
  - 31 parallel interfaces
  - 31 serial interfaces
- 10 JX-SIO or third-party modules

The JX6-SB(-I) module can be plugged into one of the submodule sockets of D-CPU, D-CPU 200, D-CPU 2 and JetControl 647 controllers. It is also possible to equip all submodule sockets of a CPU with JX6-SB(-I) modules. This option allows to build up several independent system busses.

## **1.4 Product Description – JX6-SB-I Module**

In addition to the features of the JX6-SB expansion module, the JX6-SB-I module supports all JX2-Slave modules produced by Jetter AG. Up to 8 JX2-Slave and JetMove modules can be connected to one JX6-SB-I module. Synchronization of several axes connected to the JX6-SB-I is not supported.

All features and functions of JX6-SB modules described in this manual apply to JX6-SB-I modules with no limitations. Above that, the JX6-SB-I module has additional functions which are indicated separately.

**The JX6-SB-I module allows the following maximum system bus configuration:**

- 31 JX2-I/O expansion modules
  - 248 digital inputs
  - 248 digital outputs
  - 124 analog inputs
  - 124 analog outputs
  - 62 hardware counters
  - 31 parallel interfaces
  - 31 serial interfaces
- 10 JX-SIO or third-party modules
- 8 JX2-Slave / JetMove expansion modules
  - 16 stepper motor axes
  - 8 servo axes
  - 32 PID controllers



## 1.5 System Requirements



This user information gives a description of functions offered by the JX6-SB-I module with software version 2.11.

This expansion module can only be operated together with the following controllers or devices. In the following, these controllers are referred to as CPU.

<b>System Requirements</b>	
<b>System Bus Module JX6-SB</b>	
<b>Controller</b>	<b>Minimum CPU Version</b>
D-CPU, D-CPU 200	V2.28
JetControl 647	V3.00
External Modulbus Carrier	
<b>System Bus Module JX6-SB-I</b>	
<b>Controller</b>	<b>Minimum CPU Version</b>
D-CPU, D-CPU 200	V2.28
JetControl 647	V3.00
External Modulbus Carrier	
<b>Functions of System Bus Module JX6-SB</b>	
<b>Function / Expansion Modules</b>	<b>Minimum JX6-SB Version</b>
JX2-I/O Modules	V1.00
JX-SIO	V2.10
Master-Master Mode	V1.10
JX2-Slave / JetMove Modules	V2.01 (JX6-SB-I required)
Disconnectible bus terminating resistor	Rev02





## 1.6 Technical Data

<b>Technical Data – JX6-SB(-I) Module</b>	
<b>Operating Mode "Master-Slave JX6-SB"</b>	
Total amount of I/Os When combining JX2-I/O and JX-SIO modules attention must be paid not to exceed the maximum amount of I/Os. A description on how to calculate the amount of I/Os is given in Appendix D "I/O Sum of Smart I/O JX-SIO Modules"	496
JX6-SB-I only Maximum amount of JX2-Slave / JetMove modules	8
Limited by the maximum amount of I/Os of the JX6-SB module: Maximum amount of JX2-I/O Modules Maximum amount of JX-SIO modules	31 10
<b>Operating Mode "Master-Master"</b>	
Maximum amount of JX6-SB(-I) modules	12
Maximum amount of register data per JX6-SB(-I) module	64
<b>Physical Data</b>	
Power Supply	+5 V –4 % / +4 %
Connections	Jetter system bus 125 kBaud up to 1 MBaud
Dimensions (H x W x D in mm)	17 mm x 54.51 mm x 120 mm
Power consumption	approx. 2 W
Weight	60 g

## 1.7 Third-party Modules

The Jetter system bus allows Jetter AG brand modules, as well as third-party modules to be connected to it. Such modules are, for example, valve terminals produced by Festo AG & Co. Basically, these modules are treated as JX-SIO modules.

The JX6-SB(-I) module with software version V2.10 and higher supports in "Master-Slave JX6-SB" mode the following modules:

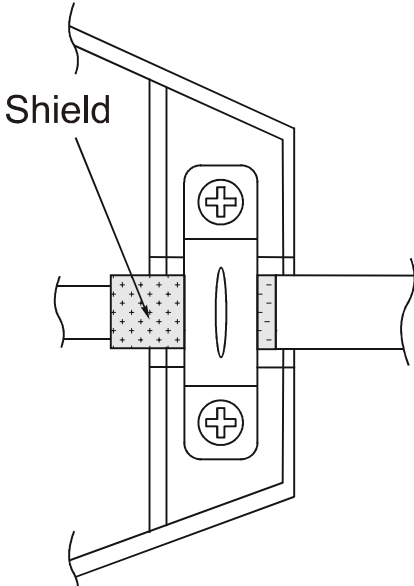
Third-party Modules Connected to the System Bus	
Manufacturer	Product Designation
Bürkert GmbH & Co. KG	Valve terminal type 8640  Bürkert_BI_100_Benutzerinformation
Festo AG & Co.	CPV10-GE-CO2-8 CPV14-GE-CO2-8 CPV18-GE-CO2-8 CPX-FB14  Festo_BI_100_BenutzerInformation
SMC Pneumatik GmbH	SI unit EX120 - SCA1 SI unit EX121 - SCA1 SI unit EX122 - SCA1  SMC_BI_100_Benutzerinformation
Lenze GmbH & Co. KG	Fieldbus Function Module Type 2175, SW Version 1.0 Supported basic devices 8201 – 8204 8211 – 8218 8221 – 8227 8241 – 8246 8200 vector 8200 vector, Cold plate  Lenze_BI_101_Benutzerinformation

These third-party modules are automatically detected and commissioned. Additional commissioning software is not required. When connecting such modules, refer to the manuals provided by the corresponding manufacturer.

Additional information is available from Jetter AG describing how to operate these modules on the Jetter system bus.

## 2 Installation

### 2.1 Connector Pin Assignment

Connector Pin Assignment		
JX6-SB(-I)	Shield	Specification
SUB-D, 9-pin female connector	 <p>Connect shield with the greatest possible surface area! Use metallised housings!</p>	<ul style="list-style-type: none"> <li>• maximum transfer rate: 1 MBit/s</li> <li>• max. cable length: 30 m at 1 MBaud</li> </ul>
Signal	Pin (male)	Pin (female)
CMODE0	1	1 <sup>1</sup>
CL	2	2
GND	3	3
CMODE1	4	4 <sup>1</sup>
TERM	5	5 <sup>1</sup>
Unassigned	6	6
CH	7	7
Unassigned	8	8
Do not connect	9	9

The shield must be connected to the connector housings on both ends of the cable with the greatest possible surface area.

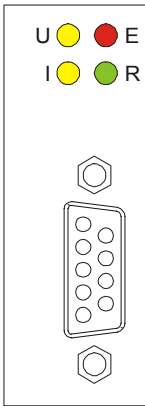
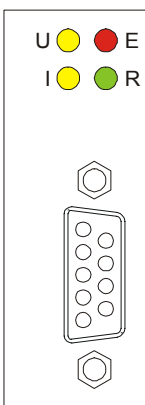
<sup>1</sup> will not be required for master-master operation

## 2.2 Front Panel

The front panel of the JX6-SB(-I) module is equipped with a 9-pin Sub-D female connector for connecting the system bus, as well as four LEDs to indicate various operating statuses of the module.

## 2.3 Diagnostic Indicators (LEDs)

The four LEDs indicate certain operating statuses of the firmware.

Description of LEDs			
After power-up			
	LED	State	Meaning
	'R' (green)	Slowly blinking	The JX6-SB module is ready for initialization commands.
	'R' (green)	Blinking twice, then long interval	The JX6-SB-I module is ready for initialization commands.
	'R' (green) 'E' (red) 'I' (yellow)	Blinking fast	The JX6-SB(-I) module has no valid firmware.
Master-Slave Mode			
	LED	State	Meaning
	'R' (green)	On	This LED indicates the successful initialization (command 1 or command 30) of the JX6-SB(-I) module.
	'E' (red)	On	When the operating system firmware of the JX6-SB(-I) module is running, the red LED indicates the state of the error bits in the status register (11m100). When an error has occurred, the E-LED is lit.
	'I' (yellow)	On	The yellow LED will remain lit, as long as the initialization of the system bus is active.
'U' (yellow)	On	This yellow status LED is used for indicating the scan cycle of all input modules. After each reading of the input conditions of all connected modules the LED changes its state. Thus, the blinking frequency indicates the duration of the input module updating process.	

Description of LEDs			
Master-Master Mode			
	LED	State	Meaning
	'R' (green)	On	This LED indicates the successful initialization (command 10) of the connected JX6-SB(-I) module.
	'E' (red)	On	A system bus error has occurred.
	'I' (yellow)	On	An accumulation of errors has been detected on the system bus. (the warning limit has been reached)
	'U' (yellow)	On	An overflow is reported by at least one receiving buffer.

## 2.4 System Bus Cable

To the manufacture of the system bus cable the following minimum requirements apply:

System Bus Cable - Technical Data		
Core cross-sectional area	1 MBaud	0.25 -0.34 mm <sup>2</sup>
	500 kBaud	0.34 -0.50 mm <sup>2</sup>
	250 kBaud	0.34 - 0.60 mm <sup>2</sup>
	125 kBaud	0.50 - 0.60 mm <sup>2</sup>
Cable capacitance per unit length	maximum 60 pF / m	
Resistivity	1 MBaud	maximum 70 Ω / km
	500 kBaud	maximum 60 Ω / km
	250 kBaud	maximum 60 Ω / km
	125 kBaud	maximum 60 Ω / km
Number of cores	5	
Shield	Complete shielding, no paired shielding	

The maximum cable length depends on the baud rate used and on the number of modules connected to the bus. The following rule of thumb applies to the system bus cable: Each JX2-I/O module connected to the bus reduces the maximum cable length by approx. 1 m.

Allowed Line Lengths			
Baud Rate	Max. line length	Max. tap line length	Max. overall tap line length
1000 kBaud	30 m	0.3 m	3 m
500 kBaud	100 m	1 m	39 m
250 kBaud	200 m	3 m	78 m
125 kBaud	200 m	-	-

The potential difference between the JX6-SB(-I) module and all expansion modules must not exceed 0.5 Volt. A constant ground potential must always be guaranteed. If there are long lines between two modules on the system bus, the line shield must be connected with functional earth (FE) approximately every 10 m because of EMI precautions. This connection should be laid out with the greatest possible surface area.

## 2.5 System Bus - Baud Rates

The system bus of Jetter AG can be operated at baud rates between 125 kBaud and 1 MBaud. Generally, the max. permissible line length of the system bus becomes shorter with increasing baud rate. At the same time however, the data transmission speed on the system bus increases with increasing baud rate. Therefore, for every individual application you have to decide whether the system bus should be operated with the maximum transfer rate or the maximum cable length.

The permissible baud rates of the system bus also depend on the modules connected to it.

Permissible Baud Rates					
JX2-I/O Modules JX2-Slave / JetMove Modules	JX-SIO	125 kBaud	250 kBaud	500 kBaud	1000 kBaud
✓		✓	✓	✓	✓
	✓	✓	✓	✓	✓
✓	✓	✓			✓

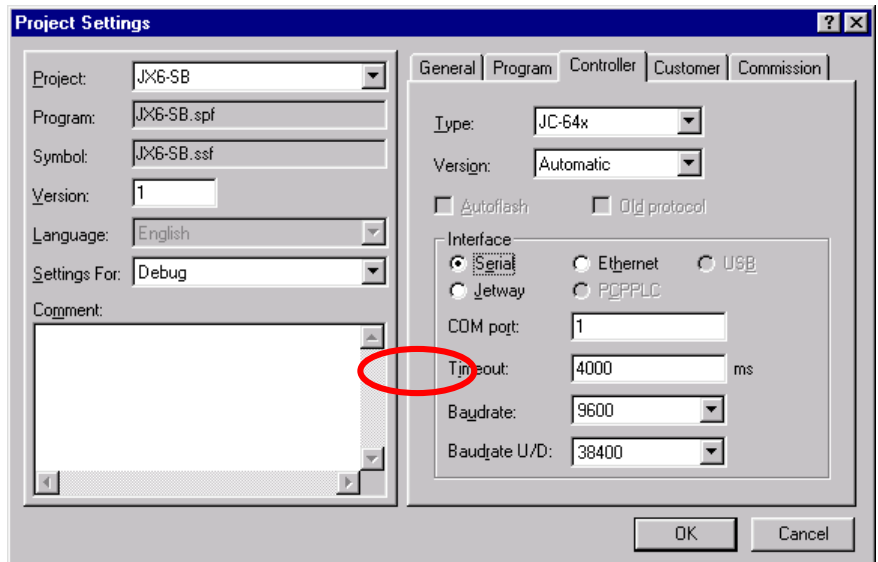
## 3 Operating System Update

### 3.1 OS Update - JX6-SB(-I) Module

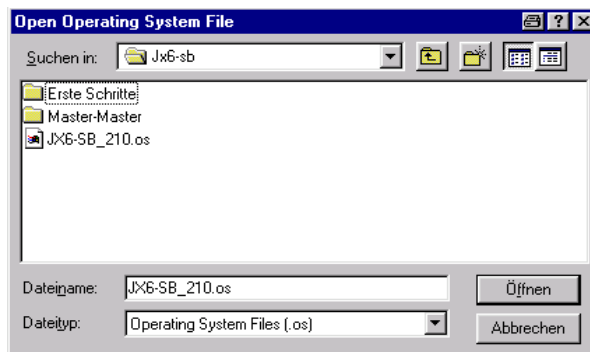
If no valid firmware is found during self-test, the three LEDs 'R', 'E' and 'I' are flashing simultaneously at a frequency of about 2 Hz. In this case, a new operating system must be installed.

An OS update can be carried out to use new functions provided by an updated OS version, or, for example, to update from an older version to version 2.10 being described in this User Information.

- An OS update can be carried out through JetSym. Connect your PC via serial port to the CPU. Move the switch on the JC647 or Delta-CPU to LOAD and power-up the controller.
- Open "Project Settings" through menu item "Project -> Settings". Set the timeout for the serial port to 4000 ms.

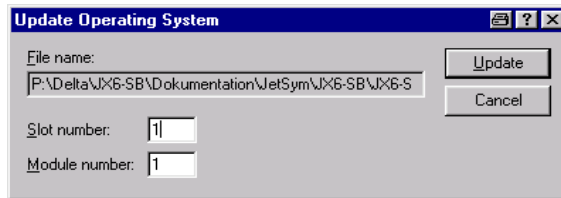


- Select menu item „Build -> Update OS...“. Browse your hard disk for the proper OS file in the dialog box. The OS file name for OS version V2.10 is JX6-SB\_210.os.



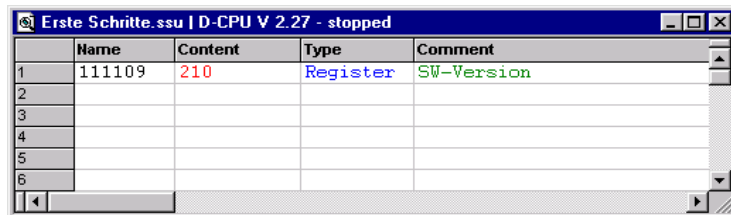


- In the following dialog box you have to enter the slot number and module number of the JX6-SB module. If the JX6-SB(-I) module is located on a CPU, the slot number is "1". Enter the module number, i.e. the number of the submodule socket of the CPU. The module number of the lowest submodule socket of the CPU ist 1. On completion of this process, the OS update starts. The progress of the update process will be indicated.



After having carried out the OS update, restart the CPU so that all changes will be applied.

- The present OS version of the JX6-SB(-I) module can be read out through register 111109. The value 210 in this register stands for OS version V2.10.

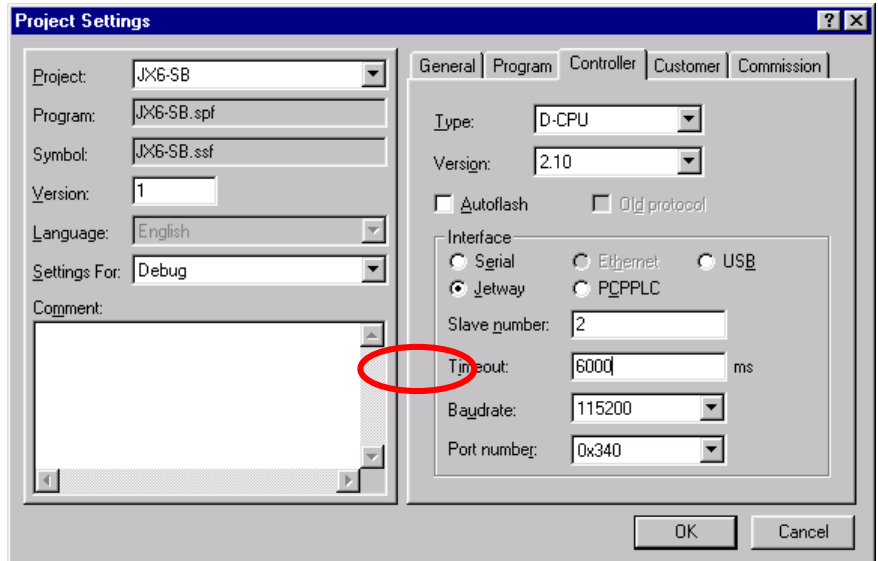


	Name	Content	Type	Comment
1	111109	210	Register	SW-Version
2				
3				
4				
5				
6				

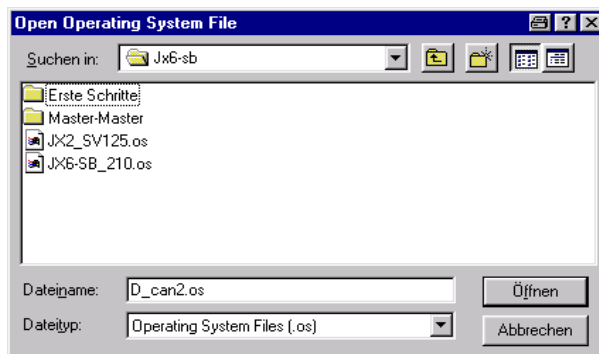
### 3.2 OS Update – JX2-Slave Module

JX2-Slave modules can be updated by analogy with the OS update of JX6-SB(-I) modules. JetSym automatically recognizes what kind of update it has to deal with.

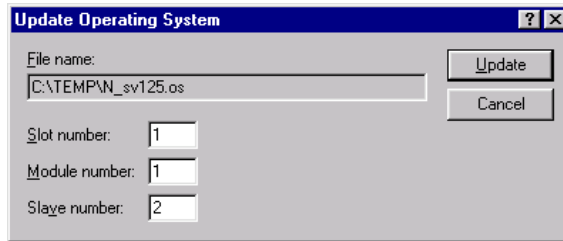
- An OS update can be carried out through JetSym. Connect your PC via serial port to the CPU. Move the switch on the JC647 or Delta-CPU to LOAD and power-up the controller.
- Open "Project Settings" through menu item "Project -> Settings". Set the timeout for the serial port to 6000 ms.



- Select menu item „Build -> Update OS...“. Browse your hard disk for the proper OS file in the dialog box. The OS file name of OS version V1.25 for a JX2-SV1 module is JX2-SV125.os.



- In the following dialog box you have to enter the slot number, the module number and slave number. If the JX6-SB(-I) module is located on a CPU, the slot number is "1". Enter the module number, i.e. the number of the submodule socket of the CPU. The module number of the lowest submodule socket of the CPU ist 1. Slave number "2" is assigned to the first JX2-Slave module connected to the JX6-SB(-I) module.



- After having carried out the OS update, restart the CPU and the JX2-Slave module so that all changes will be applied.
- The present OS version of the first JX2-Slave module can be read out through register 3112199. The value 125 in this register stands for OS version V1.25.

The screenshot shows a window titled "Erste Schritte.ssu | D-CPU V 2.27 - stopped". It contains a table with the following data:

	Name	Content	Type	Comment
1	3112199	125	Register	SW-Version JX2-Slave
2				
3				
4				
5				
6				

## 4 Topology of the Jetter System Bus

### 4.1 System Bus Topology for Master-Slave Operation

#### Decentralized arrangement of expansion modules

The DELTA and JetControl 64x CPUs can be expanded by digital and analog modules. For this purpose, the JX6-SB(-I) submodule is installed in the CPU of the JX6 family. The expansion modules are arranged in de-centralised manner and in a total distance of up to 30 meters from the JX6-SB(-I) module by 1 MBaud.

#### Note

In order to ensure flawless functioning of the decentralised arrangement, the following conditions as regards configuration must be met.

Failure to meet these conditions will result in malfunctions of individual modules or breakdown of the entire system configuration.

- 31 JX-I/O modules as a maximum, the JX2-PS1 modules not being counted, can be connected.
  - For each de-centralised module group, at least one JX2-PS1 module or one JX2-IO16 module will be needed. Even with intelligent modules, the JX2-PS1 modules must be located at the beginning of the JX2-Slave module set, in order to reduce electromagnetic interferences.
  - One JX2-PS1 module is to supply 5 JX2-I/O modules as a maximum.
  - One JX2-IO16 module is to supply 3 JX2-I/O modules as a maximum.
  - 8 JX2-Slave modules as a maximum can be connected.
  - Power supply of JX2-Slave modules (such as JetMove200, JX2-PID1, JX2-SM1D, etc.) is made through an individual power supply unit (DC 24 V), and not through a JX2-PS1 module.
  - JX2-Slave modules are not capable of supplying JX2-I/O modules with power.
  - JX-SIO modules are supplied with DC 24 V by a PSU of their own. Therefore, they do not require a JX2-PS1 module.
  - The last JX-SIO module has to be equipped with a terminating resistor.
  - The maximum line length and tap line length depending on the baud rate must not be exceeded.

If these prerequisites are not met, this will lead to malfunctioning of a module or to failure of the entire system.

### 4.1.1 Distributed Arrangement on the System Bus

By using the Jetter system bus, several expansion modules can be arranged in decentralized position at a distance of up to 200 m from the CPU. When installing expansion modules, the instructions given in chapter 4 have to be observed.

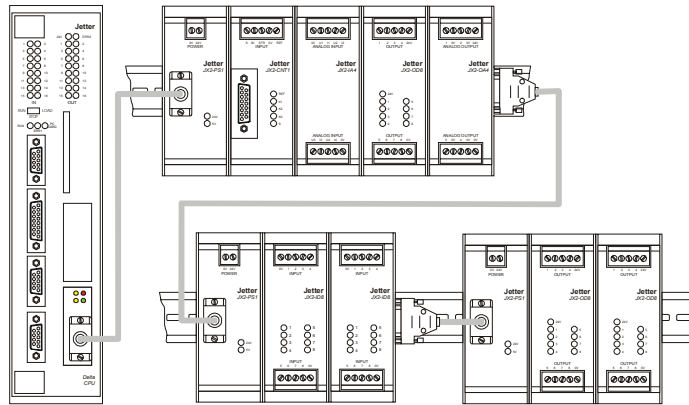


Fig. 1: Distributed Arrangement on the System Bus

### 4.1.2 Connecting JX-SIO Modules to the System Bus

When connecting JX-SIO modules to the system bus, the installation instructions given in the JX-SIO manual have to be observed. Unlike JX2-I/O and JX2-Slave modules, the system bus line is connected to the JX-SIO module through terminal blocks, and not via SUB-D connectors.

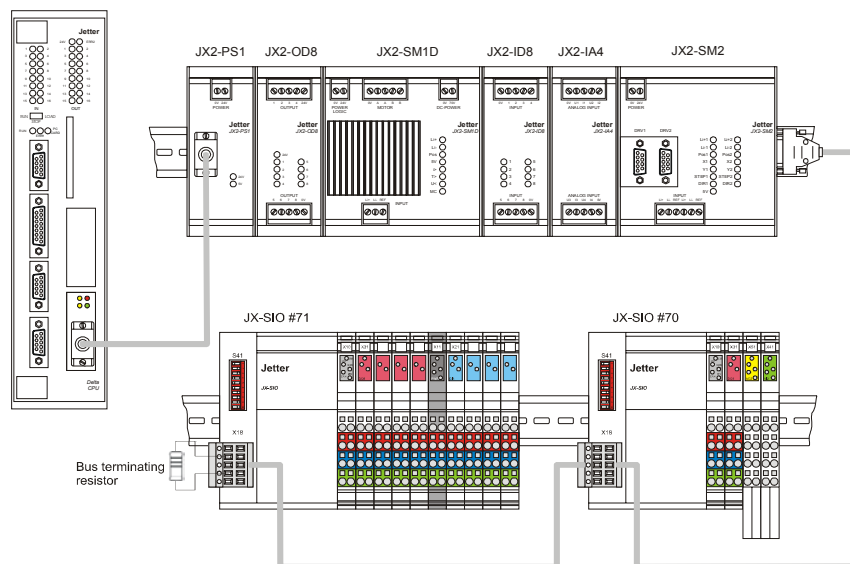


Fig. 2: Distributed Arrangement on the System Bus with JX-SIO Modules

### 4.1.3 Assigning System Bus Module Numbers

When assigning system bus module numbers, JX2-I/O modules, JX2-Slave modules and JX-SIO modules have to be counted separately. In this connection, the following rules apply:

- all JX2-I/O modules are counted up, beginning from the JX6-SB(-I) module. When doing so, all JX2-Slave modules, JX-SIO and JX2-PS1 modules are being ignored.
- the number "2" is assigned to the first JX2-I/O module connected to the JX6-SB(-I) module.
- all JX2-Slave modules are counted up, beginning from the JX6-SB(-I) module. When doing so, all JX2-I/O modules, JX-SIO and JX2-PS1 modules are being ignored.
- the number "2" is assigned to the first JX2-Slave module connected to the JX6-SB(-I) module.
- to all JX-SIO modules the module number set by switch S41 is assigned, irrespective of their position on the system bus.

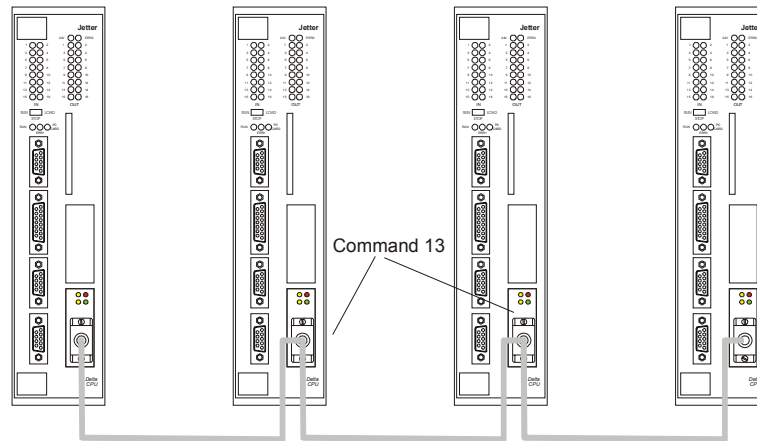
#### Example 1: Assigning System Bus Module Numbers

The module numbers for system bus configuration as given in chapter 4.1.3 are assigned as follows:

System Bus Module Numbers		
Module	I/O Module Number	Slave Module Number
JX6-SB(-I)	-	-
JX2-PS1	-	-
JX2-OD8	2	-
JX2-SM1D	-	2
JX2-ID8	3	-
JX2-IA4	4	-
JX2-SM2	-	3
JX-SIO	70	-
JX-SIO	71	-

## 4.2 System Bus Topology for Master-Master Operation

In master-master mode, up to 12 CPUs can be connected via the system bus. The permitted line and tap line lengths have to be maintained by all means. The bus terminating resistor at both ends of the system bus must be activated in the JX6-SB(-I) modules. In any other module, it must be deactivated by issuing command 13. A bus terminating resistor which can be turned off has not been available until hardware version Rev. 02.



**Fig. 3: Networking of Several CPUs in Master-Master Operating Mode**

## 5 Operating Modes

The JX6-SB(-I) module can be operated in three different modes.

- „**Master-Slave JX6-SB**“ for connecting JX2-I/O, JX2-Slave, and JX-SIO modules
- „**Master-Slave D-CAN2**“ for connecting JX2-I/O, JX2-Slave modules
- „**Master-Master**“ for connecting additional JX6-SB(-I) modules

<b>Operating Modes - Overview</b>			
<b>Function</b>	<b>Master-Slave D-CAN2</b>	<b>Master-Slave JX6-SB(-I)</b>	<b>Master-Master</b>
Minimum software version	V1.00	V2.10	V1.10
Initialization command	1	30	10
<b>Expansion Modules</b>			
Additional JX6-SB(-I) modules			✓
JX2-I/O Modules	✓	✓	
JX2-Slave modules (JX6-SB-I only)	✓	✓	
JX-SIO		✓	
FESTO CP-FB modules	✓		
Third-party Modules		✓	
<b>IO Numbering and Register Numbers</b>			
IO numbers of JX2-I/O modules	1m001 ... 1m999	m <sub>1</sub> 0201 ... m <sub>1</sub> 3216	
IO numbers of JX-SIO modules		m <sub>1</sub> 7001 ... m <sub>1</sub> 7964	
Registers of JX2-I/O modules	3m03000 ... 3m03309	3m03000 ... 3m03309	
Registers of JX-SIO modules		3m07001 ... 3m07964	
Registers of JX2-Slave modules (JX6-SB-I only)	3m12101 ... 3m19999	3m12101 ... 3m19999	

### Note

For new applications we recommend not to use "Master-Slave D-CAN2" mode.



# 6 Getting Started

This chapter will guide you through the main steps required to commission a JX6-SB(-I) module.

## 6.1 Operating Mode "Master-Slave JX6-SB"

In this operating mode Jetter AG expansion modules can be commissioned and directly accessed by the CPU.

To get familiar with the JX6-SB(-I) module, create a new project in JetSym and name it "Getting Started". The project "Getting Started" is independent of the CPU used and applies to a Delta-CPU, as well as to a JC 647.

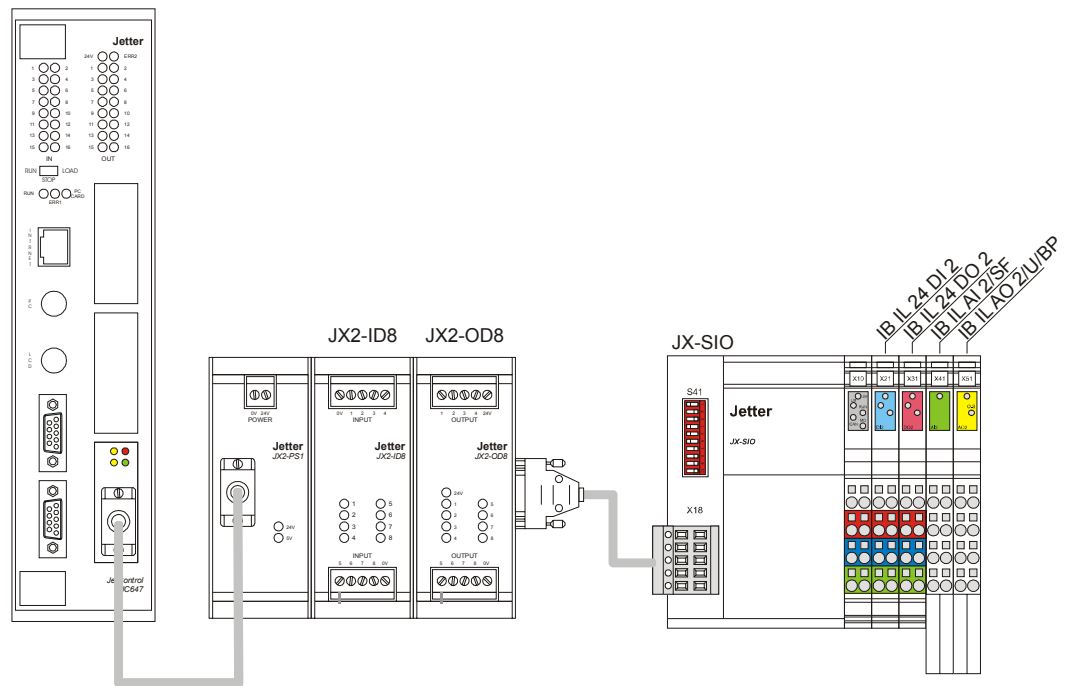
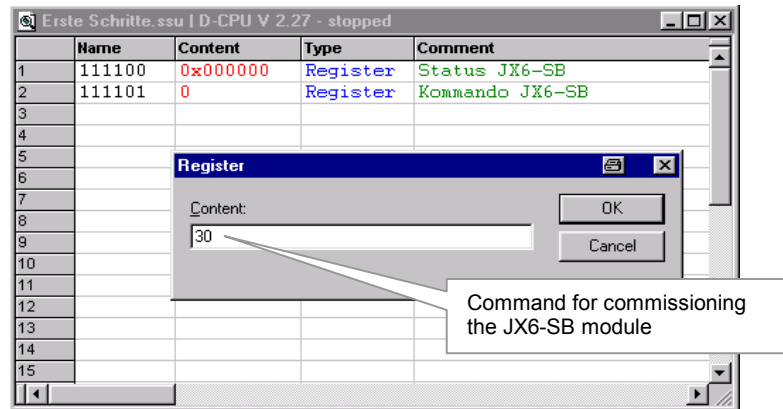


Fig. 4: Master-Slave JX6-SB(-I) Configuration

In this chapter the system bus configuration shown in the above illustration will be commissioned. A JX2-ID8, a JX2-OD8, as well as a JX-SIO with four terminals are connected to the system bus.

## Initializing the JX6-SB Module



**Fig. 5: Issuing Command 30 via Setup Window**

After the Delta CPU has been powered up, the green LED on the JX6-SB(-I) module is blinking. To initialize the connected system bus, command 30 has to be entered into the command register of the JX6-SB(-I) module. This command will cause the JX6-SB(-I) module to start in "Master-Slave JX6-SB" mode.

### Register 111101 / Command Register

Once command 30 has been issued, the lower yellow LED on the JX6-SB(-I) module turns on and signals that the system bus is being initialized. After all modules have been detected and initialized, the green LED is permanently lit and the upper yellow LED is blinking.

### Register 111100 / JX6-SB(-I) Status Register

By means of the status register you can check whether initialization has been successful. The JetSym setup window is best suited for this purpose. The obvious thing would be to watch the status register in hexadecimal representation.

After successful initialization the status register contains the value 0x000300. This means that bit 8 "Modules have been initialized" and bit 9 "Inputs are valid" are set.

## Accessing the Module Array

	Name	Content	Type	Comment
1	111100	0x000300	Register	Status JX6-SB
2	111101	30	Register	Kommando JX6-SB
3				
4	3102015	3	Register	Index Modularray
5	3102016	64	Register	Modularray
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				

**Fig. 6: Accessing the Module Array**

The JX6-SB(-I) module creates a module array which lists all modules connected to the system bus.

### Register 3102015 / Module Array Index

When this register contains the value "0", the amount of modules connected to the system bus is displayed in register 3102016.

### Register 3102016 / Module Array

In the module array, an unambiguous module code is assigned to each connected module. For example, the JX-SIO has the module code 64.

## Timeout of the JX-SIO Module

	Name	Content	Type	Comment
1	111100	0x000301	Register	Status JX6-SB
2	111101	30	Register	Kommando JX6-SB
3				
4	3102015	3	Register	
5	3102016	64	Register	
6				
7	3102008	8	Register	
8	3102011	70	Register	
9				
10				
11				
12				
13				
14				
15				

**Fig. 7: Detecting a Timeout**

The JX6-SB(-I) module automatically monitors all connected I/O modules. The JX6-SB(-I) module detects if an I/O module fails to answer, for example, due to a cable breakage or an error of the I/O module itself.

On interruption of the system bus line leading to the JX-SIO module, the individual registers contain the above mentioned values.

### **Register 111100 / JX6-SB(-I) Status Register**

When an I/O module timeout has occurred, bit 0 is set.

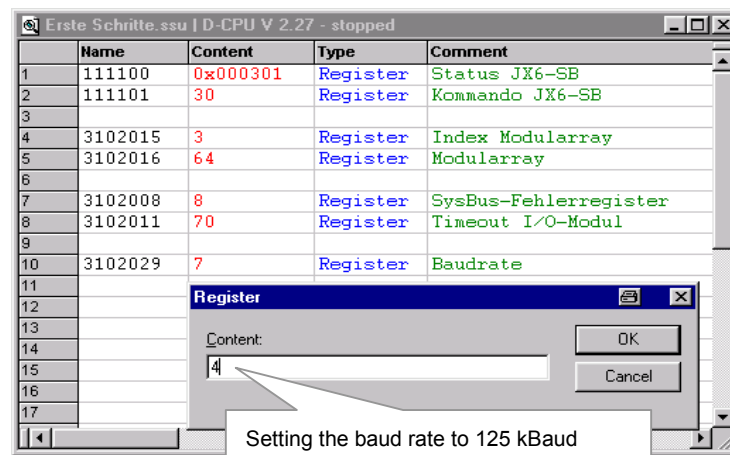
### **Register 3102008 / System Bus Error Register**

When an I/O module timeout has occurred, bit 3 is set.

### **Register 3102011 / I/O-Module Timeout**

When an I/O module timeout has occurred, the number of the I/O module is entered here.

## Changing the Baud Rate



**Fig. 8: Changing the Baud Rate**

After the JX6-SB(-I) module has been powered up, it sets the system bus baud rate to 1 MBaud. With this baud rate, a system bus with a total length of up to 30 m can be built up. For a longer bus length, the baud rate has to be reduced.

### Register 3102029 / System Bus Baud Rate

For the configuration, given in our example, the baud rate can be set to 125 kBaud. To do so, switch S18 on the JX-SIO module has to be switched over first. Then, the JX-SIO module has to be switched off and powered on again. Next, enter the value "4" (= 125 kBaud) into register 3102029 and start the JX6-SB(-I) module by issuing command 30. As regards functions, there is no difference between individual baud rates. Solely the cycle time of the system bus changes.

### Example 2: Initializing the JX6-SB(-I) Module

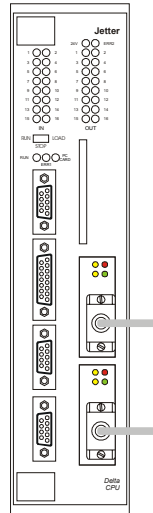
This program is for commissioning the JX6-SB(-I) module in "Master-Slave JX6-SB" mode. Once the program has been executed, output 22305 on the JX2-OD8 module is blinking.

#### Program File

```
TASK 0
    REGISTER_LOAD (3102029, 7)           // Setting the baud rate if required
    REGISTER_LOAD (111101, 30)          // Command for starting
    WHEN
        BIT_CLEAR (111100, 13)         // Setting the busy bit to zero
        BIT_SET (111100, 9)           // Enabling inputs
    THEN
    LABEL lblink
        OUT 22305                       // Flashing light
        DELAY 10
        -OUT 22305
        DELAY 10
        GOTO lblink
```

## 6.2 Operating Mode "Master-Master"

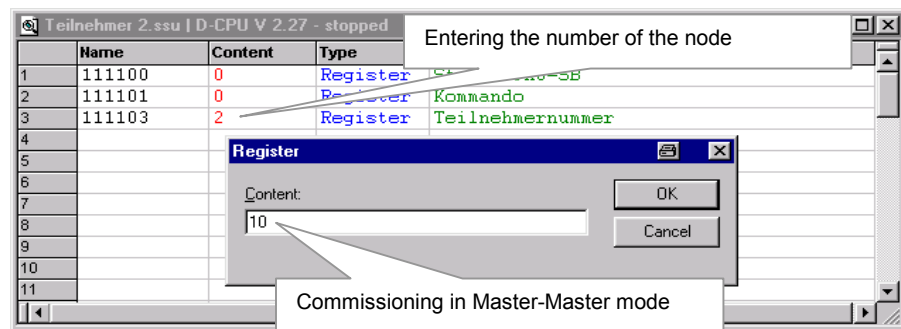
This operating mode allows register data to be exchanged between several CPUs. To familiarize with the Master-Master operating mode, create in JetSymb a new project and name it "Master-Master".



**Fig. 9: Master-Master Configuration**

For this purpose, you can install two JX6-SB(-I) modules on one CPU. Create two setup windows for station 2 and station 3.

### Configuring the JX6-SB(-I) module located in socket 1



**Fig. 10: Configuring the lower JX6-SB(-I) Module**

After the Delta CPU has been powered up, the green LED on both JX6-SB(-I) modules is blinking.

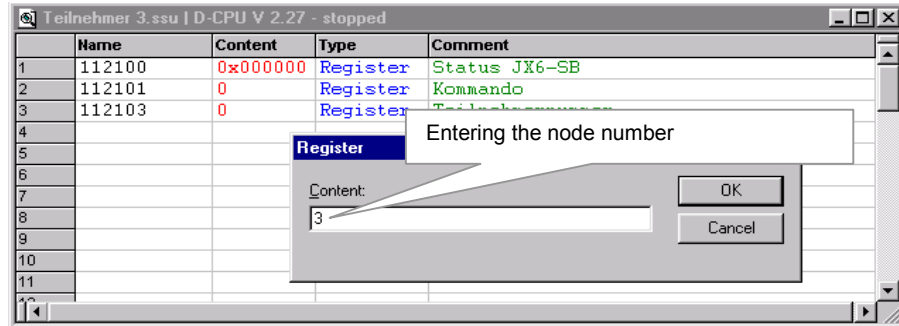
#### Register 111103 / Station Number

To initialize the lower module, first, assign station number "2" to it.

**Register 111101 / Command**

Then, enter command "10" into register 111101. After successful initialization, the green LED is permanently lit and the status register 111100 contains the value 800000<sub>HEX</sub>.

**Configuring the JX6-SB(-I) module located in socket 2**



**Fig. 11: Configuring the upper JX6-SB(-I) Module**

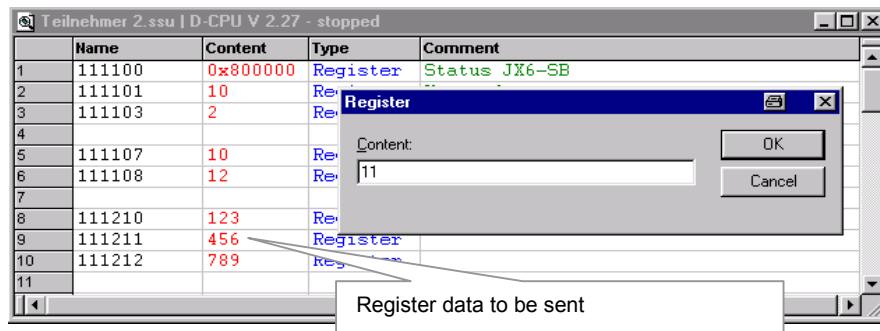
**Register 112103 / Station Number**

First of all, a station number has to be assigned to the upper module to be able to initialize it, too. Since station # 2 already exists, station number 3 is assigned to the upper module.

**Register 112101 / Command**

Then, enter command "10" into register 112101. After successful initialization, the green LED is permanently lit and the status register 112100 contains the value 800000<sub>HEX</sub>.

**Sending two register data from the lower JX6-SB(-I) module**



**Fig. 12: Sending Register Data**

When sending register data, the number of the first and the last register to be sent have to be specified only. Register 11m2zz contains the data to be sent. In our example, registers 111210 through 111212 are to be sent. Consequently, the first register data number is 10 and the last register data number is 12. These two numbers have to be entered into the registers 111107 and 111108. Then, command 11 has to be issued. Once the busy bit in the status register will be reset to zero, the register data will have been sent.

### Receiving register data on the upper JX6-SB(-I) module

	Name	Content	Type	Comment
1	112100	0x800001	Register	Status JX6-SB
2	112101	10	Register	Kommando
3	112103	3	Register	Teilnehmernummer
4				
5	112110	10	Register	erstes Empfangsregister
6	112111	12	Register	letztes Empfangsregister
7				
8				
9				
10				
11				

Bit 0 = 1, receive register data

**Fig. 13: Detecting Received Register Data**

The JX6-SB(-I) module receives the sent register data without assistance and stores them to an internal receiving buffer. Each JX6-SB(-I) module creates two internal receiving buffers for every other JX6-SB(-I) module.

Once the JX6-SB(-I) module has received register data from station 2, it sets bit 0 in the status register. In addition, registers 112110 and 112111 contain the number of the first and the last received register.



## Reading Received Register Data

	Name	Content	Type	Comment
1	112100	0x800001	Register	Status JX6-SB
2	112101	12	Register	Kommando
3	112103	3	Register	Teilnehmernummer
4				
5	112110	10	Register	erstes Empfangsregister
6	112111	12	Register	letztes Empfangsregister
7				
8	112105	2	Register	Teilnehmernummer
9	112107	10	Register	erstes Register
10	112108	12	Register	letztes Register
11				
12	112210	123	Register	Register-Daten
13	112211	456	Register	
14	112212	789	Register	
15				
16				

Fig. 14: Reading Register Data

By means of command "12" the content of an internal receiving buffer can be copied into register 1122zz. Then, the CPU can read out this register.

### Register 112105 / Station Number for Command 12

The contents of receiving buffer of station 2 are to be copied into register 1122zz. Therefore, a "2" has to be entered here.

### Register 112107 / First Register Datum

As specified in register 112110, number "10" is assigned to the first register to be copied.

### Register 112108 / Last Register Datum

As specified in register 112111, number "12" is assigned to the last register to be copied.

### Register 112101 / Command

By issuing command 12 the JX6-SB(-I) module copies 3 registers from the internal receiving buffer of station 2 into registers 112210 through 112212.

### Register 112210 through 112212 / Register Data

The register data sent by station 2 now appear in these registers.

### Receipt Acknowledgement

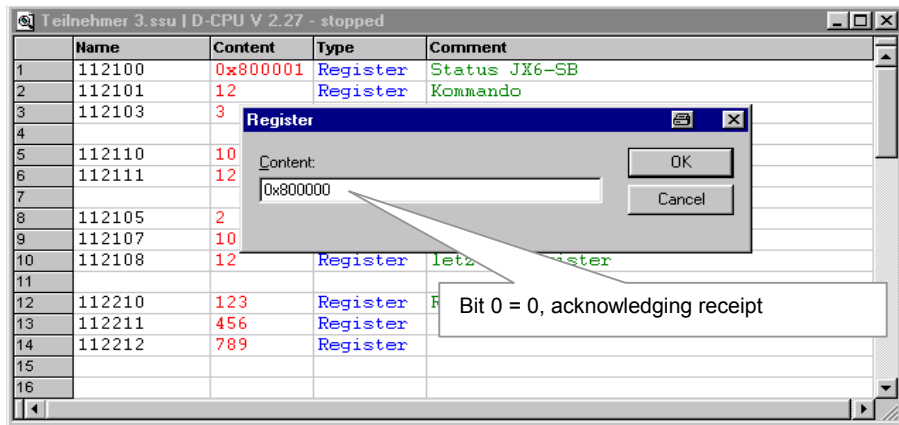


Fig. 15: Receipt Acknowledgement

If station 2 now sends additional register data, these data can be read out only if receipt of previously sent register data is acknowledged by deleting bit 0 in the status register. Without such an acknowledgement, the JX6-SB(-I) module copies new received register data into the second internal receiving buffer.

#### Register 112100 / Status Register

Receipt of register data is acknowledged by deleting bit 0.

# 7 Operating Mode „Master-Slave D-CAN2“

JX2-I/O and JX2-Slave modules can fully be integrated into Jetter control systems via a JX6-SB(-I) module. The application program has transparent access to the modules connected to the system bus.

The interface between the D\_CAN2 and the user program consists of several registers and an I/O range.

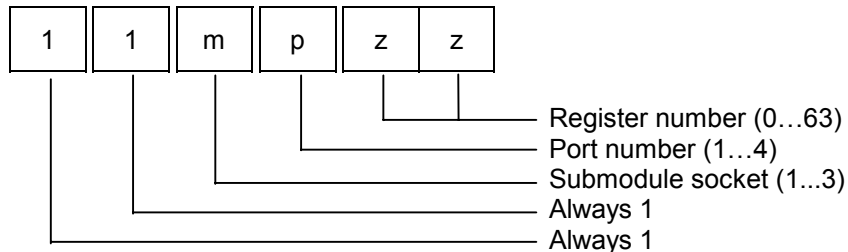
## 7.1 Register Areas

Register Areas - Master-Slave JX6-SB(-I)		
Register Area	Description	Remanent register
11m100 ... 11m163	Configuration and status registers of the JX6-SB(-I) module	no
11m300 ... 11m315	Overlaid registers for digital inputs of JX2-I/O modules	no
11m400 ... 11m415	Overlaid registers for digital outputs of JX2-I/O modules	no
3m0 3000 ... 3m0 3299	Registers of JX2-I/O modules Delta-CPU from V2.28 on JetControl JC647 from V3.00 on	no
3m 12100 ... 3m 19999	JX6-SB-I only Configuration and status registers of the JX2-Slave module Delta-CPU from V2.28 on JetControl JC647 from V3.00 on	no

### 7.1.1 Configuration and Status Registers

The numbers of registers belonging to this register area consist of a six-digit code.

#### Code of configuration and status registers



The submodule socket code “m” indicates the number of the socket on the CPU module. The port number “p” is to distinguish the four register ranges on the respective submodule. Finally, one of the 64 possible registers, belonging to the corresponding port, will be designated by the register number “zz”.

### Example 3: Command Register of the JX6-SB(-I) Submodule

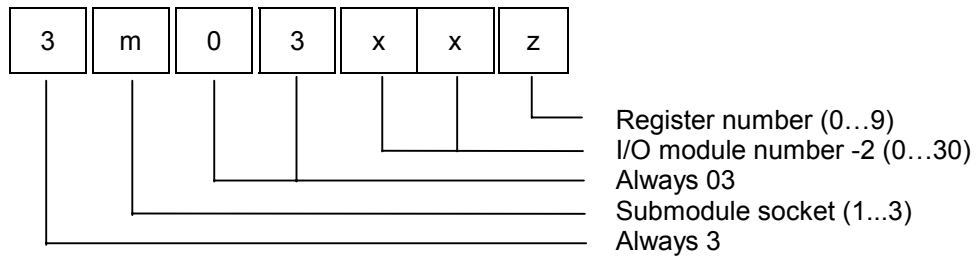
In this example, command 1 is to be entered into register 1 (command register) of port 1 located on the JX6-SB(-I) submodule residing in socket 2 of the CPU module.

```
REGISTER_LOAD (112101, 1)
```

## 7.1.2 Register Area for JX2-I/O Modules

The numbers of registers belonging to this register area consist of a seven-digit code.

### Register code for JX2-I/O Modules



The socket number “m” indicates in which socket of the CPU module the submodule is located. The I/O module number “xx” is to distinguish between several JX2-I/O modules. Finally, one of the 10 possible registers will be designated by the register number “z”.

### Example 4: Version register of a JX2-I/O module

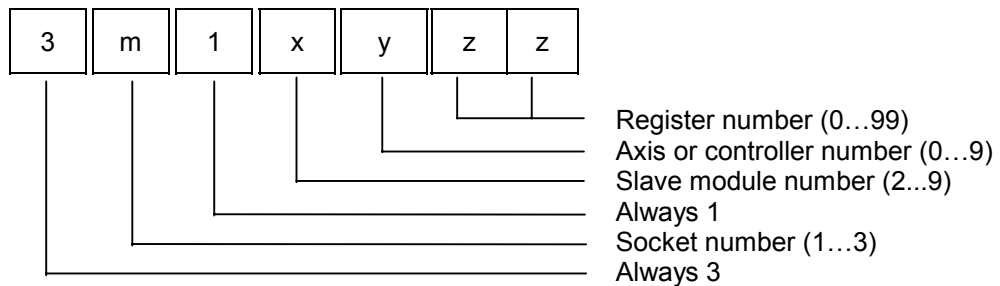
The version value (register 9) of the third JX2—I/O module on the system bus is to be entered into register 1234 of the CPU. The JX6-SB(-I) module is located in socket # 2 on the CPU module.

```
REGISTER_LOAD (1234, @3203029)
```

## 7.1.3 Register Area for JX2-Slave Modules

The numbers of registers belonging to this register area consist of a seven-digit code. This register area is available only on the JX6-SB-I module.

### Register code for JX2-Slave Modules



The socket number "m" indicates in which socket of the CPU module the submodule is located. The slave module number "x" is to distinguish between several JX2-slave modules. The axis or controller number "y" is to access individual functions of the JX2-Slave, and, finally, the register number "zz" is to select a register from the available amount of 100 registers.

### Example 5: Actual Position Register of a JX2-Slave Module

The actual position (register 9, axis 1) of the fourth JX2-Slave module on the system bus is to be entered into register 1234. The JX6-SB(-I) module is located in socket # 2 on the CPU module.

```
REGISTER_LOAD (1234, @3215109)
```

## 7.2 I/O Range

In the I/O range, the transferred inputs and outputs are combined. Following system bus initialization, all inputs and outputs of JX2-I/O modules are consecutively numbered. Each JX2-I/O module has 8 inputs and/or 8 outputs.

Access to inputs and outputs is possible by means of I/O instructions of the application program, or by overlaying of registers (in groups of 16 each).

### 7.2.1 Access to Individual Inputs and Outputs

Access to the inputs and outputs of expansion modules connected to the JX6-SB(-I) module can be made as usual with the help of the I/O instructions in the SYMPAS language. Distinction between expansion module I/Os and local I/Os is made by the numbers.

### Example 6: Input and Output Instructions

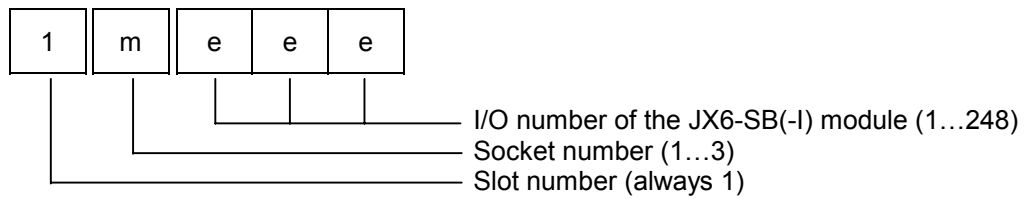
In the following example the JX6-SB(-I) module is plugged into the second submodule socket of the CPU module. Five modules are connected to the system bus. On activation of input 1 of the second JX2-I/O module, output 2 on the first JX2-I/O module is to be set.

#### Program File:

```
TASK 0
  WHEN
    IN 12009          // Waiting until input is set
  THEN
    OUT 12002        // Setting the output
```

## 7.2.2 Access to Inputs and Outputs

### Code of I/O numbers for JX2-I/O Modules

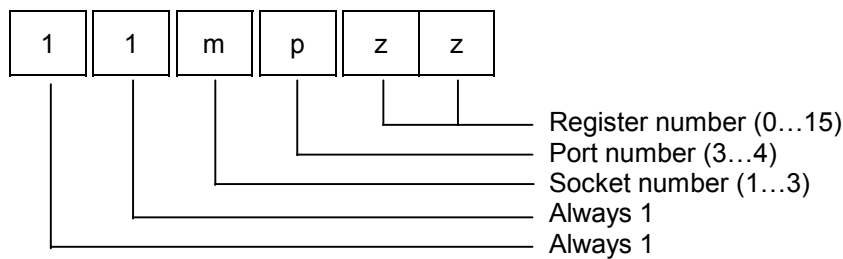


The socket number “m” indicates in which socket of the CPU module the submodule is located. The I/O numbers of the JX6-SB(-I) module range between 1 and 999.

## 7.2.3 Access to Inputs and Outputs by means of Register Overlaying

The values of the registers that are overlaid by the input and output ranges are 16-bit values. As in the registers mentioned under 0, the register numbers are six-digit numbers which are composed as follows:

### Code of I/O register overlaying for JX2-I/O modules



The submodule socket code “m” indicates the number of the socket on the CPU module. Port number “p” is to distinguish between input and output ranges. The inputs have been summarized in port 3, the outputs in port 4. Finally, one of the groups of sixteen will be selected by the register number “zz”. The register ranges of the inputs and outputs are transferred from, respectively to the JX6-SB(-I) without handshake, and for this reason have only a data consistency of 8 bit.

Input and Output Overlay			
Register number	Inputs	Register number	Outputs
11m300	1m001 .. 1m016	11m400	1m001 .. 1m016
11m301	1m017 .. 1m032	11m401	1m017 .. 1m032
11m302	1m033 .. 1m048	11m402	1m033 .. 1m048
11m303	1m049 .. 1m064	11m403	1m049 .. 1m064
11m304	1m065 .. 1m080	11m404	1m065 .. 1m080
11m305	1m081 .. 1m096	11m405	1m081 .. 1m096
11m306	1m097 .. 1m112	11m406	1m097 .. 1m112
11m307	1m113 .. 1m128	11m407	1m113 .. 1m128

Input and Output Overlay			
11m308	1m129 .. 1m144	11m408	1m129 .. 1m144
11m309	1m145 .. 1m160	11m409	1m145 .. 1m160
11m310	1m161 .. 1m176	11m410	1m161 .. 1m176
11m311	1m177 .. 1m192	11m411	1m177 .. 1m192
11m312	1m193 .. 1m208	11m412	1m193 .. 1m208
11m313	1m209 .. 1m224	11m413	1m209 .. 1m224
11m314	1m225 .. 1m240	11m414	1m225 .. 1m240
11m315	1m241 .. 1m248	11m415	1m241 .. 1m248

### 7.2.4 Addressing of the JX2-I/O Modules and of the Festo-CP-FB Modules

The following possibilities of addressing exist for any available module type:

- all JX2-I/O modules are addressed in 8 Bit width
- all FESTO-CP-FB modules are addressed in 16 Bit width

Technical Data - JX2-I/O Modules			
Module	Inputs	Outputs	Registers
JX2-ID8	8 dig. inputs		9 = version number
JX2-OD8, JX2-OD4		8 dig. outputs	9 = version number
JX2-IO16	8 dig. inputs	8 dig. outputs	
JX2-IA4, JX2-OA4, JX2-CNT1, JX2-SER, JX2-PRN1		8 virtual configuration outputs	0 – 9 = module specific registers
FESTO Input Terminal	16 dig. inputs		
FESTO Output Terminal		16 dig. outputs	

All inputs and outputs are strung together consecutively.

This means that each JX2-I/O module has 8 inputs and/or 8 outputs. All FESTO-CP-FB valve terminals have 16 inputs or 16 outputs. The FESTO-CP-FB modules are always addressed after all the JX2-I/O modules. During system bus initialization all JX2-I/O and all JX2-Slave modules are separately considered. Thus, they are addressed separately.

**Example 7: System Bus Configuration with 5 JX2 Modules**

System bus with one JX2-IO16, two JX2-ID8 modules, one JX2-OD8 and one JX2-OA4 module. The modules of the following system bus configuration have the input and output addresses shown in the table below. Power supply modules are not displayed.

<b>System Bus Configuration</b>			
<b>Module number</b>	<b>Type</b>	<b>Inputs</b>	<b>Outputs</b>
2	JX2-ID8 Input module	1m001 ... 1m008	-
3	JX2-OD8 Output module	-	1m001 ... 1m008
4	JX2-ID8 Input module	1m009 ... 1m016	-
5	JX2-IO16 Input/output Module	1m017 ... 1m024	1m009 ... 1m016
6	JX2-OA4 Analog outputs	-	1m017 ... 1m024

**Example 8: System Bus Configuration with JX2 Modules and 1 Festo-CP-FB Module**

System bus with two digital output modules JX2-OD8, one FESTO-CP-FB input module, and one digital input module JX2-ID8. Power supply modules are not displayed.

<b>System Bus Configuration</b>			
<b>Module number</b>	<b>Type</b>	<b>Inputs</b>	<b>Outputs</b>
2	JX2-OD8 Output module	-	1m001 .. 1m008
3	JX2-OD8 Output module	-	1m009 .. 1m016
5	FESTO-CP-FB Input Module	1m009 .. 1m016	-
6		1m017 .. 1m024	
4	JX2-ID8 Input module	1m001 .. 1m008	-

Irrespective of their position on the system bus, FESTO-CP-FB modules are addressed after all JX2-I/O modules.



## 7.3 Description of Registers

Register 11m100: JX6-SB(-I) Status	
Function	Description
Read	Present interface state
Write	Acknowledge the status
Value range	Bit-oriented
Value after reset	0 with JX6-SB 2048 with JX6-SB-I

### Meaning of the individual status register bits:

- Bit 0 : 1 = Timeout of JX2-I/O module
- Bit 1 : 1 = Output driver error with one of the JX2-I/O modules
- Bit 2 : 1 = Fatal system bus error
- Bit 3 : 1 = Diagnostic message of a Festo CP-FB module
- Bit 4 : 1 = Timeout of JX2-Slave module
- Bit 8 : 1 = all expansion modules connected to the system bus have been initialized
- Bit 9 : 1 = all input data of expansion modules connected to the system bus are valid
- Bit 13 : 1 = Busy; the module is in the initialization state
- Bit 15 : 0 = Hardware configuration: JX6-SB  
1 = Hardware configuration: JX6-SB-I

Register 11m101: Commands	
Function	Description
Read	Command issued last
Write	Command issued to the interface
Value range	0, 1 - 2
Value after reset	0

By writing into this register, certain actions are triggered on the module.

### Commands for Master-Slave-D-CAN2 Mode:

- 1 Initializing the JX2-Slave, JX2-I/O and FESTO-CP-FB modules.  
This command can also be used to re-initialize the system bus.
- 2 Re-programming the baud rate of the Festo-CP-FB modules

---

**Example 9: Initialization in Master-Slave-D-CAN2 Mode:**

In this example, 3 modules of 1 MBaud each are to be initialized.

```
REGISTER_LOAD (11m101, 1) // Initializing the bus system
WHEN
  BIT_CLEAR (11m100, 13)
THEN
  // three modules are to be detected:
  IF
    REG 11m110 // Checking whether initialization
    # // has found all modules.
    3
  THEN
    // Error message
  THEN
    // Bus system has been initialized with 1 MBaud
```

For initialization by issuing command 1, the description of register 11m113 must also be considered. When issuing command 2, registers 11m104 and 11m162 are relevant.

<b>Register 11m102: Timeout with a JX2-I/O Module</b>	
<b>Function</b>	<b>Description</b>
Read	Module number of the JX2-I/O module, where a timeout has occurred.
Write	Deleting the module number
Value range	0, 2 - 32
Value after reset	0

If in register 11m100 timeout of an JX2-I/O module is signaled by bit 0, the number of the I/O module can be read out of this register. The JX2-I/O modules are counted as one module, Festo-CP-FB modules are counted as two modules. The first module has module number 2.

<b>Register 11m103: Output Driver Error</b>	
<b>Function</b>	<b>Description</b>
Read	Module number of the JX2-I/O module, where an error of the output driver has occurred.
Write	Deleting the module number
Value range	0, 2 - 32
Value after reset	0

If in register 11m100 an output driver error of an JX2-I/O module is signaled by bit 1, the number of this JX2-I/O module can be read out of this register. When reading out the module number, the way modules are counted has to be taken into account: The JX2-I/O modules are counted as one module each, Festo-CP-FB modules are counted as two modules each. The first module has module number 2. If bit 3 has been set in register 11m100, the number of the Festo-CP-FB module that signals a diagnostic message can be read out of this register.

Register 11m104: Baud Rate	
Function	Description
Read	Present baud rate
Write	New baud rate
Value range	4 - 7
Value after reset	7 (1 Mbaud)

### Adjustable baud rates:

<b>4</b>	125 kBaud (not available for FESTO-CP-FB modules)
<b>5</b>	250 kBaud
<b>6</b>	500 kBaud
<b>7</b>	1 MBaud

In bus systems of a total length of more than 30 meters, communication at the standard baud rate of 1 Mbaud is not possible any more. Thus, the baud rate must be reduced.

### Example 10: Setting the Baud Rates in JX2-I/O Modules

The JX2 modules can communicate at 125 KBaud, 250 KBaud, 500 KBaud and 1 MBaud. Changes in baud rates apply only after the system bus has been re-initialized. All JX2-I/O and JX2-Slave modules will recognize the baud rate automatically.

```

REGISTER_LOAD (11m104, 5) // Baud rate 250 kBaud
REGISTER_LOAD (11m101, 1) // Initializing the bus system
WHEN
  BIT_CLEAR (11m100, 13)
THEN
  // Three modules are to be recognized:
  IF
    REG 11m110 // Check, whether initialization
    # // has found all modules.
    3
  THEN
    // Error message
  THEN
    // Bus system has been initialized at 250 kBaud

```

### Example 11: Setting the baud rate of Festo-CP-FB modules

The Festo-CP-FB modules can communicate at 250 Kbaud, 500 Kbaud and 1 Mbaud. The factory setting for all Festo-CP-FB modules is 1 Mbaud.

To change the baud rate, all Festo-CP-FB modules have to be re-programmed via JX6-SB(-I) module.

All Festo-CP-FB modules that have been connected and recognized are always programmed for the baud rate written in register 11m104. After re-powering the new baud rate will be active; thus, the Festo-CP-FB modules can be used in a longer bus system.

```
// The baud rate has to be re-programmed only once
// The baud rate is stored by Festo-CP-FB modules in a non-volatile mode
// FESTO-CP modules, connected at 1 Mbaud
REGISTER_LOAD (11m104, 7) // Baud rate 1 Mbaud
REGISTER_LOAD (11m101, 1) // Initializing the bus system
WHEN
  BIT_CLEAR (11m100, 13)
THEN // Three modules are to be detected:
  IF
    REG 11m110 // Check, whether initializing
    # // has found all modules.
    3
  THEN
    //Error message
  THEN
    // Bus system has been initialized at 1 Mbaud
    REGISTER_LOAD (11m104, 6) // Baud rate 500 kbaud
    REGISTER_LOAD (11m162, 1234567) // Code for re-programming
    REGISTER_LOAD (11m101, 2) // Re-programming the baud rate
  WHEN
    BIT_CLEAR (11m100, 13)
  THEN

// FESTO-CP-FB modules re-programmed to 500 kbaud
// Turn CPU and all FESTO-CP-FB modules off and on again
```

```
// The system bus has to be initialized once the CPU has been powered up
REGISTER_LOAD (11m104, 6) // Baud rate 500 kBaud
REGISTER_LOAD (11m101, 1) // Initializing the bus system
WHEN
    BIT_CLEAR (11m100, 13)
THEN // Three modules are to be detected:
IF
    REG 11m110 // Check, whether initialization
    # // has found all modules.
    3
THEN
    // Error message
THEN
    // Bus system has been initialized at 500 kBaud
```

Mixed bus systems (JX2- and FESTO-CP-FB modules) may only be run at a baud rate of 1 MBaud. This means, a baud rate reduction is only possible in bus systems that consist either of JX2 modules or Festo-CP-FB modules exclusively.

<b>Register 11m105: Input Module Dummy List 2 - 17</b>	
Function	Description
Read	Bit-coded list of configured JX2-I/O dummy input modules # 2 through 17.
Write	Configuration of JX2-I/O dummy input modules Bit 0: 1 = JX2-I/O module # 2 exists 0 = JX2-I/O module # 2 is a dummy input module  Bit 1: 1 = JX2-I/O module # 3 exists 0 = JX2-I/O module # 3 is a dummy input module  Bit 2: 1 = JX2-I/O module # 4 exists 0 = JX2-I/O module # 4 is a dummy input module  etc.
Value range	0 - 65535
Value after reset	65535

<b>Register 11m106: Input Module Dummy List 18 -32</b>	
Function	Description
Read	Bit-coded list of configured JX2-I/O dummy input modules # 18 through 32.
Write	Configuration of JX2-I/O dummy input modules Bit 0: 1 = JX2-I/O module # 18 exists 0 = JX2-I/O module # 18 is a dummy input module  Bit 1: 1 = JX2-I/O module # 19 exists 0 = JX2-I/O module # 19 is a dummy input module  Bit 2: 1 = JX2-I/O module # 20 exists 0 = JX2-I/O module # 20 is a dummy input module  etc.
Value range	0 - 65535
Value after reset	65535

A deleted bit means, that the respective module is only virtually available as an input module. The modules that are following the virtual output module will receive the consecutive addressing number. Festo-CP-FB modules cannot be defined as dummy modules. Changes in dummy module configuration apply only after the system bus has been re-initialized.

<b>Register 11m107: Output Module Dummy List 2 - 17</b>	
Function	Description
Read	Bit-coded list of configured JX2-I/O dummy output modules # 2 through 17.
Write	Configuration of JX2-I/O dummy output modules Bit 0: 1 = JX2-I/O module # 2 exists 0 = JX2-I/O module # 2 is a dummy output module  Bit 1: 1 = JX2-I/O module # 3 exists 0 = JX2-I/O module # 3 is a dummy output module  Bit 2: 1 = JX2-I/O module # 4 exists 0 = JX2-I/O module # 4 is a dummy output module  etc.
Value range	0 - 65535
Value after reset	65535

<b>Register 11m108: Output Module Dummy List 18 -32</b>	
Function	Description
Read	Bit-coded list of configured JX2-I/O dummy output modules # 18 through 32.
Write	Configuration of JX2-I/O dummy output modules Bit 0: 1 = JX2-I/O module # 18 exists 0 = JX2-I/O module # 18 is a dummy output module  Bit 1: 1 = JX2-I/O module # 19 exists 0 = JX2-I/O module # 19 is a dummy output module  Bit 2: 1 = JX2-I/O module # 20 exists 0 = JX2-I/O module # 20 is a dummy output module
Value range	0 - 65535
Value after reset	65535

A deleted bit means, that the respective module is only virtually available as an output module. The modules that are following the virtual output module will receive the consecutive addressing number. Festo-CP-FB modules cannot be defined as dummy modules. Changes in dummy module configuration apply only after the system bus has been re-initialized.



<b>Register 11m109: Firmware Version Number</b>	
<b>Function</b>	<b>Description</b>
Read	Present firmware version
Write	Illegal
Value range	23-bit signed integer
Value after reset	Version number * 100

From this register the firmware version number of the JX6-SB(-I) module can be read out. The value that has been read equals the product of the version number times a hundred. Thus, value 210, for example, refers to version 2.10.

When the JX6-SB(-I) module is in the self test routine after switching on, the version number of the self test routine plus one thousand is displayed by this register.

$$\text{Register 11m109} = 1103$$

If the JX6-SB(-I) module is in the loading routine during the operating system update, the version number of the loading routine plus two thousand is displayed by this register.

$$\text{Register 11m109} = 2103$$

<b>Register 11m110: Number of detected JX2-I/O and JX2-Slave modules</b>	
<b>Function</b>	<b>Description</b>
Read	Number of detected JX2-I/O and JX2-Slave modules
Write	Makes no sense
Value range	0 - 39
Value after reset	0

After the initialization phase, the number of recognized modules can be read out of this register. Each JX2-I/O module or FESTO-CP-FB module is considered as one module.

<b>Register 11m111: Pointer to a module array</b>	
<b>Function</b>	<b>Description</b>
Read	Present pointer to a certain module
Write	Setting the pointer to a certain module
Value range	0 - 39
Value after reset	0

<b>Register 11m112: Module array</b>																																									
<b>Function</b>	<b>Description</b>																																								
Read	<p>Module array</p> <p>11m111 = 0 -&gt; 11m112= Amount of modules</p> <p>11m111= 1 -&gt; 11m112= Code of the first module</p> <p>11m111= 2 -&gt; 11m112= Code of the second module</p> <p>Code #:</p> <p><b>JX2-I/O Modules</b></p> <table> <tr><td>0</td><td>JX2-OD8</td></tr> <tr><td>1</td><td>JX2-ID8</td></tr> <tr><td>2</td><td>JX2-IO16</td></tr> <tr><td>3</td><td>JX2-IA4</td></tr> <tr><td>4</td><td>JX2-OA4</td></tr> <tr><td>5</td><td>JX2-CNT1</td></tr> <tr><td>6</td><td>JX2-PRN1</td></tr> <tr><td>7</td><td>JX2-SER1</td></tr> <tr><td>32</td><td>Festo-CP-FB output module</td></tr> <tr><td>33</td><td>Festo-CP-FB input module</td></tr> </table> <p><b>JX2-Slave Modules</b></p> <table> <tr><td>128</td><td>JX2-SV1</td></tr> <tr><td>129</td><td>CAN-DIMA</td></tr> <tr><td>130</td><td>JX2-SM2</td></tr> <tr><td>131</td><td>JX2-SM1D</td></tr> <tr><td>132</td><td>JX2-PID1</td></tr> <tr><td>133</td><td>JX2-PROF1</td></tr> <tr><td>146</td><td>JetMove 600 Series</td></tr> </table> <p><b>Dummy modules</b></p> <table> <tr><td>253</td><td>JX2-Slave Dummy Module</td></tr> <tr><td>254</td><td>JX2-I/O Dummy Module</td></tr> <tr><td>255</td><td>Not identified</td></tr> </table>	0	JX2-OD8	1	JX2-ID8	2	JX2-IO16	3	JX2-IA4	4	JX2-OA4	5	JX2-CNT1	6	JX2-PRN1	7	JX2-SER1	32	Festo-CP-FB output module	33	Festo-CP-FB input module	128	JX2-SV1	129	CAN-DIMA	130	JX2-SM2	131	JX2-SM1D	132	JX2-PID1	133	JX2-PROF1	146	JetMove 600 Series	253	JX2-Slave Dummy Module	254	JX2-I/O Dummy Module	255	Not identified
0	JX2-OD8																																								
1	JX2-ID8																																								
2	JX2-IO16																																								
3	JX2-IA4																																								
4	JX2-OA4																																								
5	JX2-CNT1																																								
6	JX2-PRN1																																								
7	JX2-SER1																																								
32	Festo-CP-FB output module																																								
33	Festo-CP-FB input module																																								
128	JX2-SV1																																								
129	CAN-DIMA																																								
130	JX2-SM2																																								
131	JX2-SM1D																																								
132	JX2-PID1																																								
133	JX2-PROF1																																								
146	JetMove 600 Series																																								
253	JX2-Slave Dummy Module																																								
254	JX2-I/O Dummy Module																																								
255	Not identified																																								
Write	Disabled																																								
Value range	0 - 255																																								
Value after reset	0																																								

<b>Register 11m113: Amount of Festo-CP-FB modules</b>	
<b>Function</b>	<b>Description</b>
Read	Present amount of Festo-CP-FB modules
Write	Illegal
Value range	0 - 15
Value after reset	0

There are two possibilities of initializing the Festo-CP-FB modules:

### **Example 12: Initialization by the JX6-SB(-I) Module (sorting by check numbers):**

All Festo-CP-FB modules are automatically initialized by the JX6-SB(-I) module once command 1 has been issued. The CP modules are entered into the CP list according to their check number. First the lower, then the higher check numbers are entered into the list. Addressing of the CP modules corresponds to the order in the CP list. After initializing, the amount of CP modules (11m113) and the module ID (11m115-11m117) can be read.

```

REGISTER_LOAD (11m101, 1)          // Initializing the bus system
WHEN
REGISTER_LOAD (11m100, 13)
THEN
IF // 3 modules are to be detected
REGISTER_LOAD (11m110, 1)          // Check, whether initializing
#                                  // has found all modules.
3
THEN
// Error message
THEN
// bus system has been initialized at 1 Mbaud
REGISTER_LOAD (11m114, 1)          // First CP module
REGISTER_LOAD (100, @11m115)       // Checking number # 1
REGISTER_LOAD (101, @11m116)       // Type # 1
REGISTER_LOAD (102, @11m117)       // I/O configuration # 1
REGISTER_LOAD (11m114, 2)          // Second CP module
REGISTER_LOAD (103, @11m115)       // Checking number # 2
REGISTER_LOAD (104, @11m116)       // Type # 2
REGISTER_LOAD (105, @11m117)       // I/O configuration # 2
REGISTER_LOAD (11m114, 3)          // Third CP module
REGISTER_LOAD (106, @11m115)       // Checking number # 3
REGISTER_LOAD (107, @11m116)       // Type # 3
REGISTER_LOAD (108, @11m117)       // I/O configuration # 3

```

### Example 13: Initializing according to the settings of the user program (logical order):

The application program is to enter the amount of CP modules, their configuration, check number and type into the register of the JX6-SB(-I) module before command 1 is issued. The CP modules are entered into the CP list according to this order.

If exactly one CP module that has been entered is missing, and, in return, exactly one new CP module of the same type is found, then, the check number of the former CP module will be replaced by the number of the new module.

If the defined CP module cannot be initialized, no CP module will be displayed as found (11m113 = 0), thus, no CP module can be addressed.

If additional modules are found, they will be set to the end of the CP list.

After this, the application program can read out the actual condition and save a possible exchange of a CP module.

#### Note:

Starting from Firmware version 1.10, the type of the CP module is to be transferred to the JX6-SB(-I) module as specified by the application program.

```
// Enter FESTO-CP modules into the list
REGISTER_LOAD (11m113, 3) // Initializing 3 FESTO-CP modules
REGISTER_LOAD (11m114, 1) // First CP module
REGISTER_LOAD (11m115, @100) // Checking number # 1
REGISTER_LOAD (11m116, @101) // Type # 1
REGISTER_LOAD (11m117, @102) // I/O configuration # 1
REGISTER_LOAD (11m114, 2) // Second CP module
REGISTER_LOAD (11m115, @103) // Checking number # 2
REGISTER_LOAD (11m116, @104) // Type # 2
REGISTER_LOAD (11m117, @105) // I/O configuration # 2
REGISTER_LOAD (11m114, 3) // Third CP module
REGISTER_LOAD (11m115, @106) // Checking number # 3
REGISTER_LOAD (11m116, @107) // Type # 3
REGISTER_LOAD (11m117, @108) // I/O configuration # 3
REGISTER_LOAD (11m101, 1) // Initializing the bus system

WHEN
  BIT_CLEAR (11m100, 13)
THEN
  IF // Have 3 modules been recognized?
    REG 11m110 // Check, whether initializing
    # // has found all modules.
    3
  THEN
    // Error message
  THEN
    // Bus system has been initialized with 1 Mbaud
    // Reading back FESTO CP-FB modules
    REGISTER_LOAD (11m114, 1) // First CP module
    REGISTER_LOAD (100, @11m115) // Checking number # 1
```

```

REGISTER_LOAD (101, @11m116) // Type # 1
REGISTER_LOAD (102, @11m117) // I/O configuration # 1
REGISTER_LOAD (11m114, 2) // Second CP module
REGISTER_LOAD (103, @11m115) // Checking number # 2
REGISTER_LOAD (104, @11m116) // Type # 2
REGISTER_LOAD (105, @11m117) // I/O configuration # 2
REGISTER_LOAD (11m114, 3) // Third CP module
REGISTER_LOAD (106, @11m115) // Checking number # 3
REGISTER_LOAD (107, @11m116) // Type # 3
REGISTER_LOAD (108, @11m117) // I/O configuration # 3
    
```

<b>Register 11m114: Pointer to a Festo-CP-FB Array</b>	
<b>Function</b>	<b>Description</b>
Read	Present pointer to a certain Festo-CP-FB module
Write	Setting the pointer to a certain Festo-CP-FB module
Value range	1 – 15
Value after reset	0

The CP module, the configuration of which is to be read out or written (11m115 through 11m117) is selected by this pointer.

**Meaning of the Pointer to the FESTO-CP-FB Array:**

- 1**        11m115 = Check number of the first module  
             11m116 = Type of the first module  
             11m117 = I/O configuration of the first module
- 2**        11m115 = Check number of the second module  
             11m116 = Type of the second module  
             11m117 = I/O configuration of the second module
- etc.**     Additional modules

<b>Register 11m115: Festo-CP-FB check number</b>	
<b>Function</b>	<b>Description</b>
Read	Check number of the Festo-CP-FB module
Write	New check number
Value range	0 – 1048575
Value after reset	0

The check numbers are written into the table either by the user or by the controller (see register 11m113: Amount of Festo-CP-FB modules). The type together with the check number provide unambiguous access to a Festo-CP-FB module.

<b>Register 11m116: Type of the Festo-CP-FB module</b>	
<b>Function</b>	<b>Description</b>
Read	Type of the Festo-CP-FB module
Write	New type
Value range	0 – 4095
Value after reset	0

<b>Data of Festo-CP-FB Modules</b>			
<b>Type</b>	<b>Part number</b>	<b>Type number</b>	<b>Entry for I/O Configuration</b>
CPV10-GE-FB-4	18250	100	32
CPV10-GE-FB-6	18251	101	32
CPV10-GE-FB-8	18252	102	32
CPV14-GE-FB-4	18260	110	32
CPV14-GE-FB-6	18261	111	32
CPV14-GE-FB-8	18262	112	32
CPV18-GE-FB-4	18201	125	32
CPV18-GE-FB-6	18202	126	32
CPV18-GE-FB-8	18203	127	32
CPA10/14-IFB-CP	173510	150	32
CPA18-IFB-CP		152	32
CP-A8	18207	200	32
CP-A8N	18234	208	32
CP-E16-M8	18205	240	2
CP-E16-M12x2	18206	241	2
CP-E16-KL-IP20-Z	197983	242	2
CP-E16-M8-Z	189670	244	2
CP-E16-M8N	18243	248	2
CP-E16-M12x2N	18244	249	2

<b>Register 11m117: I/O configuration of the Festo-CP-FB module</b>	
Function	Description
Read	I/O configuration of the Festo-CP-FB module 2 = CP input module 32 = CP output module
Write	New I/O configuration
Value range	0 – 32
Value after reset	0

<b>Register 11m118: Diagnostic Byte of the FESTO-CP-FB Module</b>	
Function	Description
Read	Present diagnostic byte
Write	Illegal
Value range	0 - 255
Value after reset	0

When cyclically exchanging data, Festo-CP-FB modules respond with a diagnostic byte indicating the present state of the module. Short circuits and undervoltage are reported by this diagnostic byte.

<b>Diagnostic Byte of the FESTO-CP-FB Module</b>								
CP Module	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Valve terminal			$V_{tol}$					
Input module				$E_{USen}$				
Output module					$A_{UL}$	$A_{SC/O}$		
Input/Output Module				$E_{USen}$	$A_{UL}$	$A_{SC/O}$		

**Meaning of diagnostic bits:**

- $V_{tol}$       The voltage of Festo-CP-FB valves (< 20.4 V) is below tolerance
- $E_{USen}$      Short-circuit in the power supply of CP input module sensors
- $A_{UL}$         Load voltage loss of the CP output module
- $A_{SC/O}$      Short circuit/overload of the CP output module

<b>Register 11m132: Timeout of JX2-Slave Module</b>	
<b>Function</b>	<b>Description</b>
Read	Module number of the JX2-Slave module, where a timeout has occurred.
Write	Deleting the module number
Value range	0, 2 - 9
Value after reset	0

If in register 11m100 timeout of an JX2-Slave module is signaled by bit 4, the number of the I/O module can be read out of this register. The first module has module number 2.

<b>Register 11m133: Amount of Recognized JX2-Slave Modules</b>	
<b>Function</b>	<b>Description</b>
Read	Amount of recognized JX2-Slave modules
Write	Makes no sense
Value range	0 - 8
Value after reset	0

After the initialization phase, the number of recognized modules can be read out of this register. Each JX2-Slave module is taken into account as 1 module.

<b>Register 11m158: Delay during system bus initialization</b>	
<b>Function</b>	<b>Description</b>
Read	Present delay value
Write	New delay value
Value range	0 - 255
Value after reset	10 (1000 ms)

When re-initializing the system bus a delay is to be maintained to ensure that all modules are in their initial state. If the delay is too short, it may happen that individual modules will not be recognized.

For JX2-I/O and FESTO-CP-FB modules a delay is not required.

For JX2-Slave modules a delay of 1 second is required.



<b>Register 11m159: Timeout of Festo-CP-FB modules</b>	
Function	Description
Read	Timeout of Festo-CP-FB modules
Write	New timeout in steps of 8 ms
Value range	0 - 255
Value after reset	40 (320 ms)

All Festo-CP-FB modules have got a timeout. On expiration of this timeout new data are to be received by the JX6-SB(-I) module. This timeout has to be longer than the monitoring interval (register 11m160). When the timeout has been exceeded, all outputs of the module are switched off and the green status LED goes out.

<b>Register 11m160: Monitoring interval of JX2 outputs and Festo-CP-FB modules</b>	
Function	Description
Read	Monitoring interval of JX2 outputs and Festo-CP-FB modules
Write	New monitoring interval
Value range	0 - 255
Value after reset	20 (200 ms)

All digital JX2 input modules and all Festo-CP-FB input modules are cyclically scanned and data are transferred to them in shortest possible intervals. The JX2 output modules and all additional JX2 modules (JX2-IA4, JX2-SER, ...), as well as all Festo-CP-FB output modules are only addressed, if a corresponding output changes, or if a module register is addressed. In order to recognize failure of a module, an additional monitoring interval has been installed. If after the last access to a module the time of the interval has elapsed without a further access, a monitoring access will be triggered.

<b>Register 11m162: Code Register</b>	
<b>Function</b>	<b>Description</b>
Read	Present code
Write	Enter the code
Value range	-8388608 - +8388607
Value after reset	0

In order to prevent wrong operation during a change of baud rates with Festo-CP-FB modules, the code register must be set to the value "1234567" before command "2".

<b>Registers 11m300 through 11m363: Input registers</b>	
<b>Function</b>	<b>Description</b>
Read	Conditions of the inputs 1 through 999
Write	Makes no sense
Value range	0 - 65535
Value after reset	0

By each of these registers 16 inputs can be addressed; they are cyclically read out by the corresponding JX2 and CP modules.

The following assignment applies:

Register 11m300      Inputs 1 through 16  
 Register 11m301      Inputs 17 through 32  
 etc.

In this case, bit 0 refers to input 1, while bit 15 refers to input 16.

<b>Registers 11m400 through 11m463: Output registers</b>	
<b>Function</b>	<b>Description</b>
Read	Conditions of the outputs 1 through 999
Write	Activating one or several outputs
Value range	0 - 65535
Value after reset	0

By each of these registers 16 outputs can be addressed; they are read out by the corresponding JX2 and CP modules.

The following assignment applies:

Register 11m400      Outputs 1 through 16  
 Register 11m401      Outputs 17 through 32  
 etc.

In this case, bit 0 refers to output 1, while bit 15 refers to output 16.

<b>Register 3m03xxz: Register access to JX2-I/O modules</b>	
<b>Function</b>	<b>Description</b>
Read	Depending on the function of the JX2-I/O module
Write	Depending on the function of the JX2-I/O module
Value range	Depending on the function of the JX2-I/O module
Value after reset	Depending on the function of the JX2-I/O module

These registers serve the purpose of directly accessing registers of JX2-I/O modules. With D-CPU or D-CPU 200 modules, for this type of addressing revision V2.28 will be needed.

<b>Register 3m1xyzz: Register Access to JX2-Slave Modules</b>	
<b>Function</b>	<b>Description</b>
Read	Depending on the function of the JX2-Slave module
Write	Depending on the function of the JX2-Slave module
Value range	Depending on the function of the JX2-Slave module
Value after reset	Depending on the function of the JX2-Slave module

These registers serve the purpose of directly accessing registers of JX2-Slave modules. With D-CPU or D-CPU 200 modules, for this type of addressing revision V2.28 will be needed. These registers are available only on the JX6-SB-I module.

## 7.4 Overview of Registers

Registers	Description	1) Value Range 2) Reset Value
11m100	Status Register	1) bit-oriented 2) 0
11m101	Command register	1) 1 ... 2 2) 0
11m102	Timeout with a JX2-I/O Module	1) 0 , 2 ... 32 2) 0
11m103	Output Driver Error	1) 0 , 2 ... 32 2) 0
11m104	Baud Rate	1) 4 ... 7 2) 7
11m105	Input module: Dummy list 2 -17	1) 0 ... 65535 2) 65535
11m106	Input module: Dummy list 18 - 32	1) 0 ... 65535 2) 65535
11m107	Output module: Dummy list 2 -17	1) 0 ... 65535 2) 65535
11m108	Output module: Dummy list 18 - 32	1) 0 ... 65535 2) 65535
11m109	Firmware Version Number	1) 0 ... 2999 2) Firmware version
11m110	Number of recognized JX2-I/O and JX2-Slave modules	1) 0 ... 39 2) 0
11m111	Pointer to a module array	1) 0 ... 39 2) 0
11m112	Module array	1) 0 ... 255 2) 0
11m113	Amount of Festo-CP-FB modules	1) 0 ... 15 2) 0
11m114	Pointer to Festo-CP-FB array	1) 0 ... 15 2) 0
11m115	Festo-CP-FB check number	1) 0 ... 1048575 2) 0
11m116	Type of the Festo-CP-FB module	1) 0 ... 4095 2) 0

Registers	Description	1) Value Range 2) Reset Value
11m117	I/O configuration of the Festo-CP-FB module	1) 0, 2, 32 2) 0
11m118	Diagnostic byte of the Festo-CP-FB module	1) 0 ... 255 2) 0
11m132	Timeout for a JX2-Slave module	1) 0, 2 ... 9 2) 0
11m133	Amount of recognized JX2-Slave modules	1) 0 ... 8 2) 0
11m158	Delay during system bus initialization	1) 0 ... 255 2) 10 = 1000 ms
11m159	Timeout of Festo-CP-FB modules	1) 0 ... 255 2) 40 = 320 ms
11m160	Monitoring interval of JX2 outputs and Festo-CP-FB modules	1) 0 ... 255 2) 20 = 200 ms
11m3zz	Input registers	1) 0 ... 65535 2) 0
11m4zz	Output registers	1) 0 ... 65535 2) 0
3m03xxz	Register access to JX2-I/O modules	1) 24 bit 2) 0
3m1xyzz	Register access to JX2-Slave modules	1) 24 bit 2) 0

# 8 Operating Mode "Master-Slave JX6-SB"

## 8.1 Overview of Registers

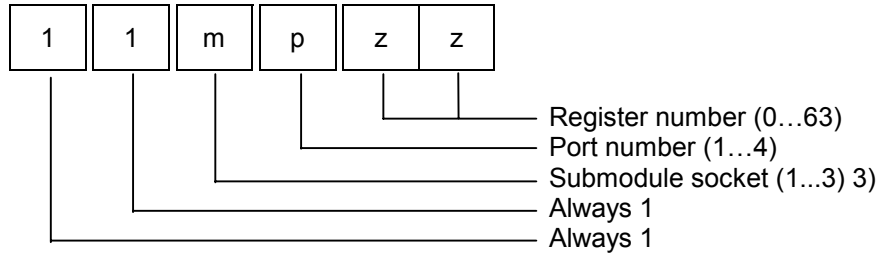
The JX6-SB(-I) module knows a host of registers which can be divided into several areas. The number of each register area results from an unambiguous syntax depending on criteria such as socket number, system bus module number and axis number. All registers of the JX6-SB(-I) module are non-remanent registers. In order to address the register area "3m" the Delta CPU requires version V2.28 or higher, and the Delta CPU2 and JetControl 647 require version V2.19 or higher.

<b>Register Areas - Master-Slave JX6-SB(-I)</b>		
<b>Register Area</b>	<b>Description</b>	<b>Remanent register</b>
11m100 ... 11m163	Configuration and status registers of the JX6-SB(-I) module  - Commissioning the JX6-SB(-I) Module - Reading out the software version, etc.	-
<b>Delta-CPU from V2.28 on, or JetControl JC647 from V3.00 on</b>		
3m0 2000 ... 3m0 2999	System Bus Special Registers  - Diagnosing the JX6-SB(-I) Module - Setting the Baud Rate - Configuring Dummy Modules - Reading out the modules presently connected to the bus - Detecting timeouts, etc.	-
3m0 3000 ... 3m0 3299	Registers of JX2-I/O modules  - Reading and writing analog values of JX2-I/O modules - Reading the software version of JX2-I/O modules	-
3m0 4000 ... 3m0 4399	Overlaid I/O registers for digital inputs and outputs of JX2-I/O modules.	-
3m0 5000 ... 3m0 5999	Overlaid I/O registers for digital inputs and registers for digital and analog input data of JX-SIO modules	-
3m0 6000 ... 3m0 6999	Overlaid I/O registers for digital outputs and registers for digital and analog output data of JX-SIO modules	-
3m0 7000 ... 3m0 7999	Configuration and status registers of the JX-SIO module	partially
3m 12100 ... 3m 19999	Configuration and status registers of the JX2-Slave module	-

### 8.1.1 Configuration and Status Registers

The numbers of registers belonging to this register area consist of a six-digit code.

#### Code of configuration and status registers



The submodule socket code "m" indicates the number of the socket on the CPU module. The port number "p" is to distinguish the four register ranges on the respective submodule. Finally, one of the 64 possible registers, belonging to the corresponding port, will be designated by the register number "zz".

#### Example 14: Command Register of the JX6-SB(-I) Submodule

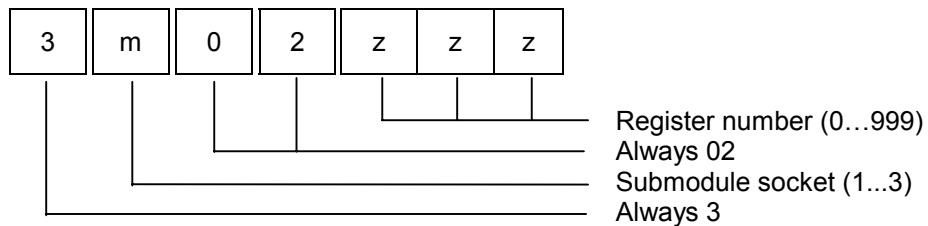
Command 30 is to be written into register 1 (command register) of port 1 located on the JX6-SB(-I) module. The JX6-SB(-I) module is located in socket # 2 on the CPU module.

```
REGISTER_LOAD (112101, 30)
```

### 8.1.2 System Bus Special Registers

The numbers of registers belonging to this register area consist of a seven-digit code.

#### Code of system bus special registers

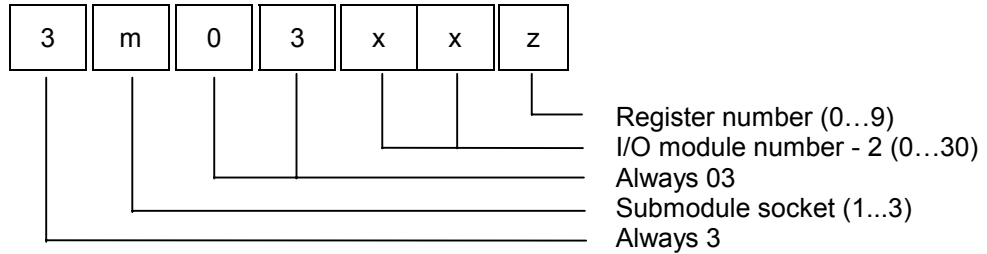


The socket number "m" indicates in which socket of the CPU module the submodule is located. Finally, one of the 100 possible registers will be designated by the register number "zzz".

### 8.1.3 Register Area for JX2-I/O Modules

The numbers of registers belonging to this register area consist of a seven-digit code.

#### Register code for JX2-I/O Modules



The socket number "m" indicates in which socket of the CPU module the submodule is located. The system bus I/O module number "xx" is to distinguish between several JX2-I/O modules. Finally, one of the 10 possible registers will be designated by the register number "z".

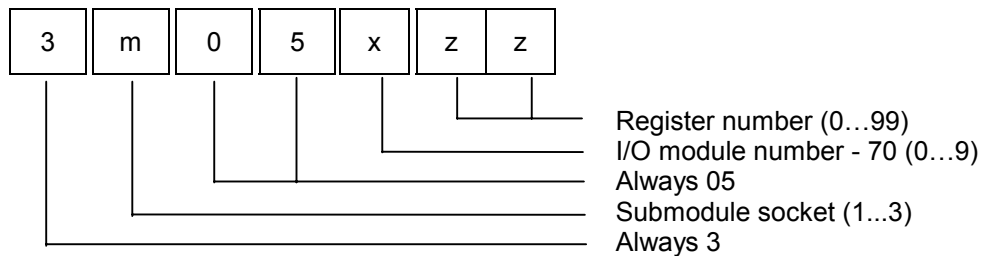
### 8.1.4 Register Area for JX-SIO Modules

The numbers of registers belonging to this register area consist of a seven-digit code.

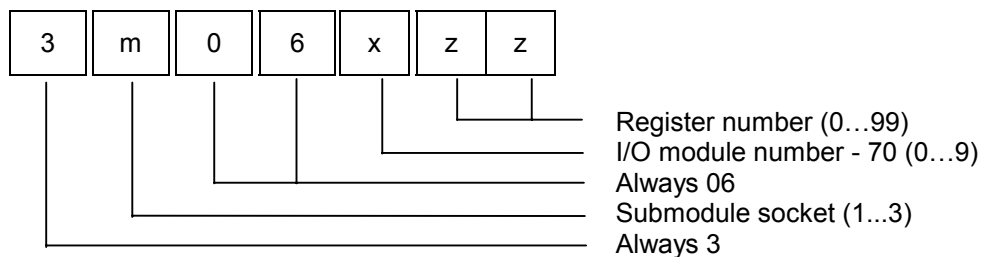
#### Note

When accessing these registers, the information given in chapter 8.6 "Timeout Adjustment for JX-SIO Modules" has to be take into account.

#### Register code for digital and analog input data

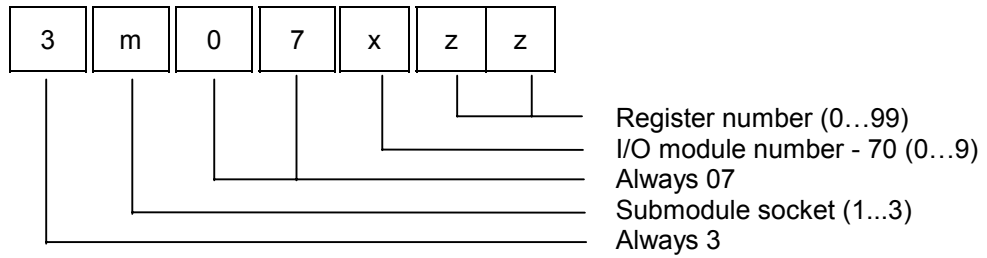


#### Register code for digital and analog output data



#### Register code for configuration and diagnostic registers



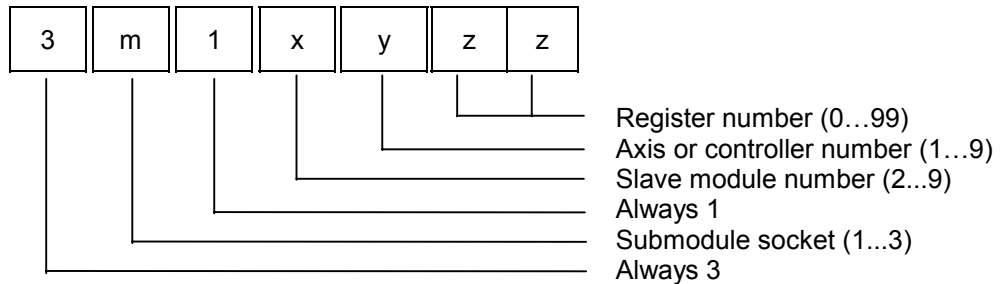


The socket number “m” indicates in which socket of the CPU module the submodule is located. The system bus I/O module number “x” is to distinguish between several JX-SIO modules. Finally, one of the 100 possible registers will be designated by the register number “zz”.

### 8.1.5 Register Area for JX2-Slave Modules

The numbers of registers belonging to this register area consist of a seven-digit code. This register area is available only on the JX6-SB-I module.

#### Register code for JX2-Slave Modules



The socket number “m” indicates in which socket of the CPU module the submodule is located. The system bus slave module number “x” is to distinguish between several JX2-slave modules. The axis or controller number “y” is to access individual functions of the JX2-Slave, and, finally, the register number “zz” is to select a register from the available amount of 100 registers.

### 8.1.6 Register Area for Third-party Modules

The Jetter system bus allows Jetter AG brand modules, as well as third-party modules to be connected to it. Such modules are, for example, valve terminals produced by Festo AG & Co.

Addressing of registers and IO numbers is made by analogy with JX-SIO modules. This applies to all special registers 3m0 2zzz, too.

The meaning of the registers 7xzz is device-specific. For information on these registers please refer to the documentation provided by the corresponding manufacturer.

## 8.2 Numbering of Modules

When assigning system bus module numbers, JX2-I/O modules, JX2-Slave modules and JX-SIO modules have to be counted separately. In this connection, the following rules apply:

- all JX2-I/O modules are counted up, beginning from the JX6-SB(-I) module. When doing so, all JX2-Slave modules, JX-SIO and JX2-PS1 modules are being ignored.
- the number "2" is assigned to the first JX2-I/O module connected to the JX6-SB(-I) module.
- all JX2-Slave modules are counted up, beginning from the JX6-SB(-I) module. When doing so, all JX2-I/O modules, JX-SIO and JX2-PS1 modules are being ignored.
- the number "2" is assigned to the first JX2-Slave module connected to the JX6-SB(-I) module.
- to all JX-SIO modules the module number set by switch S41 is assigned, irrespective of their position on the system bus.

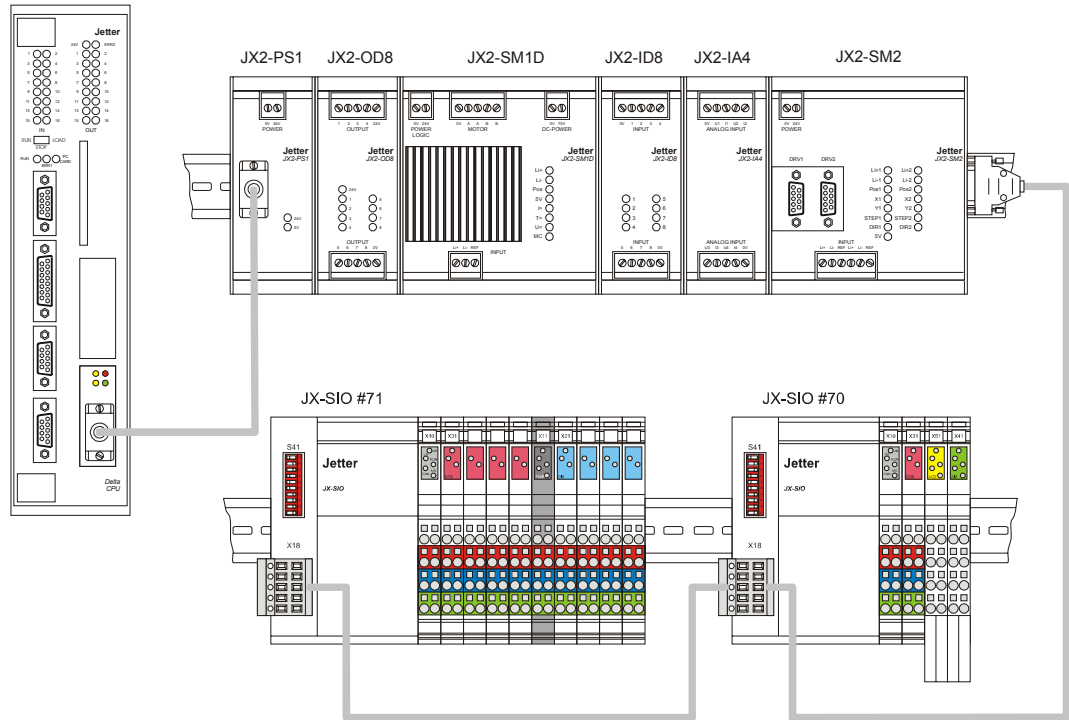


Fig. 16: Numbering: Sample Configuration

Assigning System Bus Module Numbers				
Module	I/O-Module Number	Slave Module Number	Register Number	IO Number
JX6-SB(-I)	-	-	111100 ... 111163	-
JX2-PS1	To be ignored			
JX2-OD8	2	-	3103000 ... 3103009	20201 ... 20208
JX2-SM1D	-	2	3112100 ... 3112199	-
JX2-ID8	3	-	3103010 ... 3103019	20301 ... 20308
JX2-IA4	4	-	3103020 ... 3103029	20401 ... 20408
JX2-SM2	-	4	3113100 ... 3113999	-
JX-SIO 70	70	-	3105000 ... 3107099	27001 ... 27064
JX-SIO 71	71	-	3105100 ... 3107199	27101 ... 27164

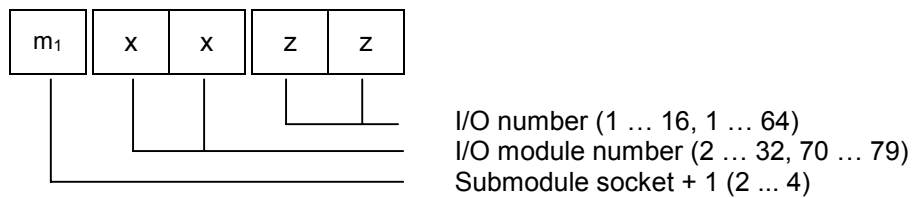
## 8.3 I/O Range

In the I/O range, the transferred inputs and outputs are combined. After system bus initialization, 16 inputs and outputs are assigned to each recognized JX2-I/O module, resp. 64 inputs and outputs to each recognized JX-SIO module. All inputs and outputs can either be accessed individually via I/O instructions of the application program or with the help of the SYMPAS setup screen, or else with the help of register overlay.

### 8.3.1 Access to Inputs and Outputs

Access to inputs and outputs of system bus modules is possible, in the usual way, through I/O instructions of the JetSym programming language. Distinction between system bus I/Os and local I/Os is made by numbering of inputs, and outputs.

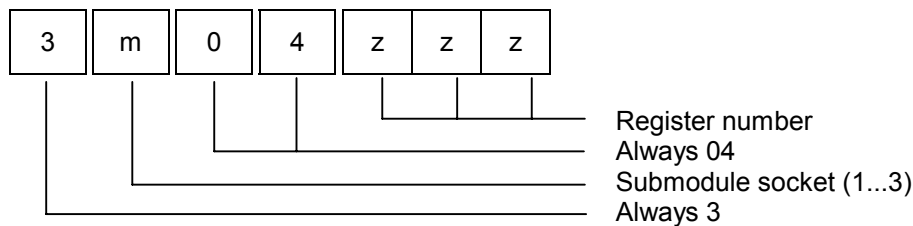
#### Code of I/O numbers for JX-SIO, and JX2-I/O modules



### 8.3.2 Access to Inputs and Outputs by means of Register Overlaying

Besides individual access to digital inputs and outputs, several inputs and outputs can jointly be accessed by means of register overlaying. This approach allows, for example, to simultaneously delete several outputs using a single REG\_ZERO instruction. In the register area 3m04zzz, only a part of the input and output data of JX2-I/O expansion modules # 31, and 32 can be accessed.

#### Code of I/O register overlaying for JX2-I/O modules



Overlying Inputs and Outputs of JX2-I/O modules		
16 Combined Inputs		
I/O Module Number	Registers	Input Numbers
1,2	3m0 4061	- m <sub>1</sub> 0201 ... m <sub>1</sub> 0208
2	3m0 4062	m <sub>1</sub> 0201 ... m <sub>1</sub> 0208 m <sub>1</sub> 0209 ... m <sub>1</sub> 0216
2, 3	3m0 4063	m <sub>1</sub> 0209 ... m <sub>1</sub> 0216 m <sub>1</sub> 0301 ... m <sub>1</sub> 0308
3	3m0 4064	m <sub>1</sub> 0301 ... m <sub>1</sub> 0308 m <sub>1</sub> 0309 ... m <sub>1</sub> 0316
3, 4	3m0 4065	m <sub>1</sub> 0309 ... m <sub>1</sub> 0316 m <sub>1</sub> 0401 ... m <sub>1</sub> 0408
4	3m0 4066	m <sub>1</sub> 0401 ... m <sub>1</sub> 0408 m <sub>1</sub> 0409 ... m <sub>1</sub> 0416
4, 5	3m0 4067	m <sub>1</sub> 0409 ... m <sub>1</sub> 0416 m <sub>1</sub> 0501 ... m <sub>1</sub> 0508
5	3m0 4068	m <sub>1</sub> 0501 ... m <sub>1</sub> 0508 m <sub>1</sub> 0509 ... m <sub>1</sub> 0516
5, 6	3m0 4069	m <sub>1</sub> 0509 ... m <sub>1</sub> 0516 m <sub>1</sub> 0601 ... m <sub>1</sub> 0608
6	3m0 4070	m <sub>1</sub> 0601 ... m <sub>1</sub> 0608 m <sub>1</sub> 0609 ... m <sub>1</sub> 0616
6, 7	3m0 4071	m <sub>1</sub> 0609 ... m <sub>1</sub> 0616 m <sub>1</sub> 0701 ... m <sub>1</sub> 0708
7	3m0 4072	m <sub>1</sub> 0701 ... m <sub>1</sub> 0708 m <sub>1</sub> 0709 ... m <sub>1</sub> 0716
7, 8	3m0 4073	m <sub>1</sub> 0709 ... m <sub>1</sub> 0716 m <sub>1</sub> 0801 ... m <sub>1</sub> 0808
8	3m0 4074	m <sub>1</sub> 0801 ... m <sub>1</sub> 0808 m <sub>1</sub> 0809 ... m <sub>1</sub> 0816
8, 9	3m0 4075	m <sub>1</sub> 0809 ... m <sub>1</sub> 0816 m <sub>1</sub> 0901 ... m <sub>1</sub> 0908
9	3m0 4076	m <sub>1</sub> 0901 ... m <sub>1</sub> 0908 m <sub>1</sub> 0909 ... m <sub>1</sub> 0916
9, 10	3m0 4077	m <sub>1</sub> 0909 ... m <sub>1</sub> 0916 m <sub>1</sub> 1001 ... m <sub>1</sub> 1008
10	3m0 4078	m <sub>1</sub> 1001 ... m <sub>1</sub> 1008 m <sub>1</sub> 1009 ... m <sub>1</sub> 1016
10, 11	3m0 4079	m <sub>1</sub> 1009 ... m <sub>1</sub> 1016 m <sub>1</sub> 1101 ... m <sub>1</sub> 1108
11	3m0 4080	m <sub>1</sub> 1101 ... m <sub>1</sub> 1108 m <sub>1</sub> 1109 ... m <sub>1</sub> 1116
11, 12	3m0 4081	m <sub>1</sub> 1109 ... m <sub>1</sub> 1116 m <sub>1</sub> 1201 ... m <sub>1</sub> 1208
12	3m0 4082	m <sub>1</sub> 1201 ... m <sub>1</sub> 1208 m <sub>1</sub> 1209 ... m <sub>1</sub> 1216
12, 13	3m0 4083	m <sub>1</sub> 1209 ... m <sub>1</sub> 1216 m <sub>1</sub> 1301 ... m <sub>1</sub> 1308
13	3m0 4084	m <sub>1</sub> 1301 ... m <sub>1</sub> 1308 m <sub>1</sub> 1309 ... m <sub>1</sub> 1316
13, 14	3m0 4085	m <sub>1</sub> 1309 ... m <sub>1</sub> 1316 m <sub>1</sub> 1401 ... m <sub>1</sub> 1408
14	3m0 4086	m <sub>1</sub> 1401 ... m <sub>1</sub> 1408 m <sub>1</sub> 1409 ... m <sub>1</sub> 1416
14, 15	3m0 4087	m <sub>1</sub> 1409 ... m <sub>1</sub> 1416 m <sub>1</sub> 1501 ... m <sub>1</sub> 1508
15	3m0 4088	m <sub>1</sub> 1501 ... m <sub>1</sub> 1508 m <sub>1</sub> 1509 ... m <sub>1</sub> 1516
15, 16	3m0 4089	m <sub>1</sub> 1509 ... m <sub>1</sub> 1516 m <sub>1</sub> 1601 ... m <sub>1</sub> 1608
16	3m0 4090	m <sub>1</sub> 1601 ... m <sub>1</sub> 1608 m <sub>1</sub> 1609 ... m <sub>1</sub> 1616
16, 17	3m0 4091	m <sub>1</sub> 1609 ... m <sub>1</sub> 1616 m <sub>1</sub> 1701 ... m <sub>1</sub> 1708
17	3m0 4092	m <sub>1</sub> 1701 ... m <sub>1</sub> 1708 m <sub>1</sub> 1709 ... m <sub>1</sub> 1716
17, 18	3m0 4093	m <sub>1</sub> 1709 ... m <sub>1</sub> 1716 m <sub>1</sub> 1801 ... m <sub>1</sub> 1808
18	3m0 4094	m <sub>1</sub> 1801 ... m <sub>1</sub> 1808 m <sub>1</sub> 1809 ... m <sub>1</sub> 1816
18, 19	3m0 4095	m <sub>1</sub> 1809 ... m <sub>1</sub> 1816 m <sub>1</sub> 1901 ... m <sub>1</sub> 1908
19	3m0 4096	m <sub>1</sub> 1901 ... m <sub>1</sub> 1908 m <sub>1</sub> 1909 ... m <sub>1</sub> 1916

Overlaying Inputs and Outputs of JX2-I/O modules		
19, 20	3m0 4097	m <sub>1</sub> 1909 ... m <sub>1</sub> 1916 m <sub>1</sub> 2001 ... m <sub>1</sub> 2008
20	3m0 4098	m <sub>1</sub> 2001 ... m <sub>1</sub> 2008 m <sub>1</sub> 2009 ... m <sub>1</sub> 2016
20, 21	3m0 4099	m <sub>1</sub> 2009 ... m <sub>1</sub> 2016 m <sub>1</sub> 2101 ... m <sub>1</sub> 2108
21	3m0 4100	m <sub>1</sub> 2101 ... m <sub>1</sub> 2108 m <sub>1</sub> 2109 ... m <sub>1</sub> 2116
21, 22	3m0 4101	m <sub>1</sub> 2109 ... m <sub>1</sub> 2116 m <sub>1</sub> 2201 ... m <sub>1</sub> 2208
22	3m0 4102	m <sub>1</sub> 2201 ... m <sub>1</sub> 2208 m <sub>1</sub> 2209 ... m <sub>1</sub> 2216
22, 23	3m0 4103	m <sub>1</sub> 2209 ... m <sub>1</sub> 2216 m <sub>1</sub> 2301 ... m <sub>1</sub> 2308
23	3m0 4104	m <sub>1</sub> 2301 ... m <sub>1</sub> 2308 m <sub>1</sub> 2309 ... m <sub>1</sub> 2316
23, 24	3m0 4105	m <sub>1</sub> 2309 ... m <sub>1</sub> 2316 m <sub>1</sub> 2401 ... m <sub>1</sub> 2408
24	3m0 4106	m <sub>1</sub> 2401 ... m <sub>1</sub> 2408 m <sub>1</sub> 2409 ... m <sub>1</sub> 2416
24, 25	3m0 4107	m <sub>1</sub> 2409 ... m <sub>1</sub> 2416 m <sub>1</sub> 2501 ... m <sub>1</sub> 2508
25	3m0 4108	m <sub>1</sub> 2501 ... m <sub>1</sub> 2508 m <sub>1</sub> 2509 ... m <sub>1</sub> 2516
25, 26	3m0 4109	m <sub>1</sub> 2509 ... m <sub>1</sub> 2516 m <sub>1</sub> 2601 ... m <sub>1</sub> 2608
26	3m0 4110	m <sub>1</sub> 2601 ... m <sub>1</sub> 2608 m <sub>1</sub> 2609 ... m <sub>1</sub> 2616
26, 27	3m0 4111	m <sub>1</sub> 2609 ... m <sub>1</sub> 2616 m <sub>1</sub> 2701 ... m <sub>1</sub> 2708
27	3m0 4112	m <sub>1</sub> 2701 ... m <sub>1</sub> 2708 m <sub>1</sub> 2709 ... m <sub>1</sub> 2716
27, 28	3m0 4113	m <sub>1</sub> 2709 ... m <sub>1</sub> 2716 m <sub>1</sub> 2801 ... m <sub>1</sub> 2808
28	3m0 4114	m <sub>1</sub> 2801 ... m <sub>1</sub> 2808 m <sub>1</sub> 2809 ... m <sub>1</sub> 2816
28, 29	3m0 4115	m <sub>1</sub> 2809 ... m <sub>1</sub> 2816 m <sub>1</sub> 2901 ... m <sub>1</sub> 2908
29	3m0 4116	m <sub>1</sub> 2901 ... m <sub>1</sub> 2908 m <sub>1</sub> 2909 ... m <sub>1</sub> 2916
29, 30	3m0 4117	m <sub>1</sub> 2909 ... m <sub>1</sub> 2916 m <sub>1</sub> 3001 ... m <sub>1</sub> 3008
30	3m0 4118	m <sub>1</sub> 3001 ... m <sub>1</sub> 3008 m <sub>1</sub> 3009 ... m <sub>1</sub> 3016
30, 31	3m0 4119	m <sub>1</sub> 3009 ... m <sub>1</sub> 3016 m <sub>1</sub> 3101 ... m <sub>1</sub> 3109

<b>Overlaying Inputs and Outputs of JX2-I/O modules</b>		
<b>8 Combined Inputs</b>		
<b>I/O Module Number</b>	<b>Registers</b>	<b>Input Numbers</b>
2	3m0 4122	m <sub>1</sub> 0201 ... m <sub>1</sub> 0208
2	3m0 4123	m <sub>1</sub> 0209 ... m <sub>1</sub> 0216
3	3m0 4124	m <sub>1</sub> 0301 ... m <sub>1</sub> 0308
3	3m0 4125	m <sub>1</sub> 0309 ... m <sub>1</sub> 0316
4	3m0 4126	m <sub>1</sub> 0401 ... m <sub>1</sub> 0408
4	3m0 4127	m <sub>1</sub> 0409 ... m <sub>1</sub> 0416
5	3m0 4128	m <sub>1</sub> 0501 ... m <sub>1</sub> 0508
5	3m0 4129	m <sub>1</sub> 0509 ... m <sub>1</sub> 0516
6	3m0 4130	m <sub>1</sub> 0601 ... m <sub>1</sub> 0608
6	3m0 4131	m <sub>1</sub> 0609 ... m <sub>1</sub> 0616
7	3m0 4132	m <sub>1</sub> 0701 ... m <sub>1</sub> 0708
7	3m0 4133	m <sub>1</sub> 0709 ... m <sub>1</sub> 0716
8	3m0 4134	m <sub>1</sub> 0801 ... m <sub>1</sub> 0808
8	3m0 4135	m <sub>1</sub> 0809 ... m <sub>1</sub> 0816
9	3m0 4136	m <sub>1</sub> 0901 ... m <sub>1</sub> 0908
9	3m0 4137	m <sub>1</sub> 0909 ... m <sub>1</sub> 0916
10	3m0 4138	m <sub>1</sub> 1001 ... m <sub>1</sub> 1008
10	3m0 4139	m <sub>1</sub> 1009 ... m <sub>1</sub> 1016
11	3m0 4140	m <sub>1</sub> 1101 ... m <sub>1</sub> 1108
11	3m0 4141	m <sub>1</sub> 1109 ... m <sub>1</sub> 1116
12	3m0 4142	m <sub>1</sub> 1201 ... m <sub>1</sub> 1208
12	3m0 4143	m <sub>1</sub> 1209 ... m <sub>1</sub> 1216
13	3m0 4144	m <sub>1</sub> 1301 ... m <sub>1</sub> 1308
13	3m0 4145	m <sub>1</sub> 1309 ... m <sub>1</sub> 1316
14	3m0 4146	m <sub>1</sub> 1401 ... m <sub>1</sub> 1408
14	3m0 4147	m <sub>1</sub> 1409 ... m <sub>1</sub> 1416
15	3m0 4148	m <sub>1</sub> 1501 ... m <sub>1</sub> 1508
15	3m0 4149	m <sub>1</sub> 1509 ... m <sub>1</sub> 1516
16	3m0 4150	m <sub>1</sub> 1601 ... m <sub>1</sub> 1608
16	3m0 4151	m <sub>1</sub> 1609 ... m <sub>1</sub> 1616
17	3m0 4152	m <sub>1</sub> 1701 ... m <sub>1</sub> 1708
17	3m0 4153	m <sub>1</sub> 1709 ... m <sub>1</sub> 1716
18	3m0 4154	m <sub>1</sub> 1801 ... m <sub>1</sub> 1808
18	3m0 4155	m <sub>1</sub> 1809 ... m <sub>1</sub> 1816
19	3m0 4156	m <sub>1</sub> 1901 ... m <sub>1</sub> 1908
19	3m0 4157	m <sub>1</sub> 1909 ... m <sub>1</sub> 1916

<b>Overlaying Inputs and Outputs of JX2-I/O modules</b>		
20	3m0 4158	m <sub>1</sub> 2001 ... m <sub>1</sub> 2008
20	3m0 4159	m <sub>1</sub> 2009 ... m <sub>1</sub> 2016
21	3m0 4160	m <sub>1</sub> 2101 ... m <sub>1</sub> 2108
21	3m0 4161	m <sub>1</sub> 2109 ... m <sub>1</sub> 2116
22	3m0 4162	m <sub>1</sub> 2201 ... m <sub>1</sub> 2208
22	3m0 4163	m <sub>1</sub> 2209 ... m <sub>1</sub> 2216
23	3m0 4164	m <sub>1</sub> 2301 ... m <sub>1</sub> 2308
23	3m0 4165	m <sub>1</sub> 2309 ... m <sub>1</sub> 2316
24	3m0 4166	m <sub>1</sub> 2401 ... m <sub>1</sub> 2408
24	3m0 4167	m <sub>1</sub> 2409 ... m <sub>1</sub> 2416
25	3m0 4168	m <sub>1</sub> 2501 ... m <sub>1</sub> 2508
25	3m0 4169	m <sub>1</sub> 2509 ... m <sub>1</sub> 2516
26	3m0 4170	m <sub>1</sub> 2601 ... m <sub>1</sub> 2608
26	3m0 4171	m <sub>1</sub> 2609 ... m <sub>1</sub> 2616
27	3m0 4172	m <sub>1</sub> 2701 ... m <sub>1</sub> 2708
27	3m0 4173	m <sub>1</sub> 2709 ... m <sub>1</sub> 2716
28	3m0 4174	m <sub>1</sub> 2801 ... m <sub>1</sub> 2808
28	3m0 4175	m <sub>1</sub> 2809 ... m <sub>1</sub> 2816
29	3m0 4176	m <sub>1</sub> 2901 ... m <sub>1</sub> 2908
29	3m0 4177	m <sub>1</sub> 2909 ... m <sub>1</sub> 2916
30	3m0 4178	m <sub>1</sub> 3001 ... m <sub>1</sub> 3008
30	3m0 4179	m <sub>1</sub> 3009 ... m <sub>1</sub> 3016
31	3m0 4180	m <sub>1</sub> 3101 ... m <sub>1</sub> 3108
31	3m0 4181	m <sub>1</sub> 3109 ... m <sub>1</sub> 3116
32	3m0 4182	m <sub>1</sub> 3201 ... m <sub>1</sub> 3208
32	3m0 4183	m <sub>1</sub> 3209 ... m <sub>1</sub> 3216



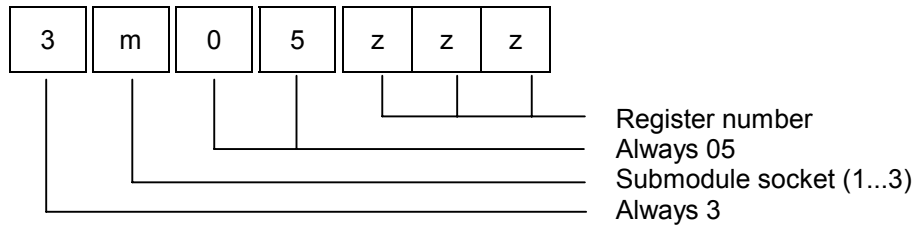
Overlaying Inputs and Outputs of JX2-I/O modules		
16 Combined Outputs		
I/O Module Number	Registers	Output Numbers
1,2	3m0 4261	- m <sub>1</sub> 0201 ... m <sub>1</sub> 0208
2	3m0 4262	m <sub>1</sub> 0201 ... m <sub>1</sub> 0208 m <sub>1</sub> 0209 ... m <sub>1</sub> 0216
2, 3	3m0 4263	m <sub>1</sub> 0209 ... m <sub>1</sub> 0216 m <sub>1</sub> 0301 ... m <sub>1</sub> 0308
3	3m0 4264	m <sub>1</sub> 0301 ... m <sub>1</sub> 0308 m <sub>1</sub> 0309 ... m <sub>1</sub> 0316
3, 4	3m0 4265	m <sub>1</sub> 0309 ... m <sub>1</sub> 0316 m <sub>1</sub> 0401 ... m <sub>1</sub> 0408
4	3m0 4266	m <sub>1</sub> 0401 ... m <sub>1</sub> 0408 m <sub>1</sub> 0409 ... m <sub>1</sub> 0416
4, 5	3m0 4267	m <sub>1</sub> 0409 ... m <sub>1</sub> 0416 m <sub>1</sub> 0501 ... m <sub>1</sub> 0508
5	3m0 4268	m <sub>1</sub> 0501 ... m <sub>1</sub> 0508 m <sub>1</sub> 0509 ... m <sub>1</sub> 0516
5, 6	3m0 4269	m <sub>1</sub> 0509 ... m <sub>1</sub> 0516 m <sub>1</sub> 0601 ... m <sub>1</sub> 0608
6	3m0 4270	m <sub>1</sub> 0601 ... m <sub>1</sub> 0608 m <sub>1</sub> 0609 ... m <sub>1</sub> 0616
6, 7	3m0 4271	m <sub>1</sub> 0609 ... m <sub>1</sub> 0616 m <sub>1</sub> 0701 ... m <sub>1</sub> 0708
7	3m0 4272	m <sub>1</sub> 0701 ... m <sub>1</sub> 0708 m <sub>1</sub> 0709 ... m <sub>1</sub> 0716
7, 8	3m0 4273	m <sub>1</sub> 0709 ... m <sub>1</sub> 0716 m <sub>1</sub> 0801 ... m <sub>1</sub> 0808
8	3m0 4274	m <sub>1</sub> 0801 ... m <sub>1</sub> 0808 m <sub>1</sub> 0809 ... m <sub>1</sub> 0816
8, 9	3m0 4275	m <sub>1</sub> 0809 ... m <sub>1</sub> 0816 m <sub>1</sub> 0901 ... m <sub>1</sub> 0908
9	3m0 4276	m <sub>1</sub> 0901 ... m <sub>1</sub> 0908 m <sub>1</sub> 0909 ... m <sub>1</sub> 0916
9, 10	3m0 4277	m <sub>1</sub> 0909 ... m <sub>1</sub> 0916 m <sub>1</sub> 1001 ... m <sub>1</sub> 1008
10	3m0 4278	m <sub>1</sub> 1001 ... m <sub>1</sub> 1008 m <sub>1</sub> 1009 ... m <sub>1</sub> 1016
10, 11	3m0 4279	m <sub>1</sub> 1009 ... m <sub>1</sub> 1016 m <sub>1</sub> 1101 ... m <sub>1</sub> 1108
11	3m0 4280	m <sub>1</sub> 1101 ... m <sub>1</sub> 1108 m <sub>1</sub> 1109 ... m <sub>1</sub> 1116
11, 12	3m0 4281	m <sub>1</sub> 1109 ... m <sub>1</sub> 1116 m <sub>1</sub> 1201 ... m <sub>1</sub> 1208
12	3m0 4282	m <sub>1</sub> 1201 ... m <sub>1</sub> 1208 m <sub>1</sub> 1209 ... m <sub>1</sub> 1216
12, 13	3m0 4283	m <sub>1</sub> 1209 ... m <sub>1</sub> 1216 m <sub>1</sub> 1301 ... m <sub>1</sub> 1308
13	3m0 4284	m <sub>1</sub> 1301 ... m <sub>1</sub> 1308 m <sub>1</sub> 1309 ... m <sub>1</sub> 1316
13, 14	3m0 4285	m <sub>1</sub> 1309 ... m <sub>1</sub> 1316 m <sub>1</sub> 1401 ... m <sub>1</sub> 1408
14	3m0 4286	m <sub>1</sub> 1401 ... m <sub>1</sub> 1408 m <sub>1</sub> 1409 ... m <sub>1</sub> 1416
14, 15	3m0 4287	m <sub>1</sub> 1409 ... m <sub>1</sub> 1416 m <sub>1</sub> 1501 ... m <sub>1</sub> 1508
15	3m0 4288	m <sub>1</sub> 1501 ... m <sub>1</sub> 1508 m <sub>1</sub> 1509 ... m <sub>1</sub> 1516
15, 16	3m0 4289	m <sub>1</sub> 1509 ... m <sub>1</sub> 1516 m <sub>1</sub> 1601 ... m <sub>1</sub> 1608
16	3m0 4290	m <sub>1</sub> 1601 ... m <sub>1</sub> 1608 m <sub>1</sub> 1609 ... m <sub>1</sub> 1616
16, 17	3m0 4291	m <sub>1</sub> 1609 ... m <sub>1</sub> 1616 m <sub>1</sub> 1701 ... m <sub>1</sub> 1708
17	3m0 4292	m <sub>1</sub> 1701 ... m <sub>1</sub> 1708 m <sub>1</sub> 1709 ... m <sub>1</sub> 1716
17, 18	3m0 4293	m <sub>1</sub> 1709 ... m <sub>1</sub> 1716 m <sub>1</sub> 1801 ... m <sub>1</sub> 1808
18	3m0 4294	m <sub>1</sub> 1801 ... m <sub>1</sub> 1808 m <sub>1</sub> 1809 ... m <sub>1</sub> 1816
18, 19	3m0 4295	m <sub>1</sub> 1809 ... m <sub>1</sub> 1816 m <sub>1</sub> 1901 ... m <sub>1</sub> 1908
19	3m0 4296	m <sub>1</sub> 1901 ... m <sub>1</sub> 1908 m <sub>1</sub> 1909 ... m <sub>1</sub> 1916

<b>Overlaying Inputs and Outputs of JX2-I/O modules</b>		
19, 20	3m0 4297	m <sub>1</sub> 1909 ... m <sub>1</sub> 1916 m <sub>1</sub> 2001 ... m <sub>1</sub> 2008
20	3m0 4298	m <sub>1</sub> 2001 ... m <sub>1</sub> 2008 m <sub>1</sub> 2009 ... m <sub>1</sub> 2016
20, 21	3m0 4299	m <sub>1</sub> 2009 ... m <sub>1</sub> 2016 m <sub>1</sub> 2101 ... m <sub>1</sub> 2108
21	3m0 4300	m <sub>1</sub> 2101 ... m <sub>1</sub> 2108 m <sub>1</sub> 2109 ... m <sub>1</sub> 2116
21, 22	3m0 4301	m <sub>1</sub> 2109 ... m <sub>1</sub> 2116 m <sub>1</sub> 2201 ... m <sub>1</sub> 2208
22	3m0 4302	m <sub>1</sub> 2201 ... m <sub>1</sub> 2208 m <sub>1</sub> 2209 ... m <sub>1</sub> 2216
22, 23	3m0 4303	m <sub>1</sub> 2209 ... m <sub>1</sub> 2216 m <sub>1</sub> 2301 ... m <sub>1</sub> 2308
23	3m0 4304	m <sub>1</sub> 2301 ... m <sub>1</sub> 2308 m <sub>1</sub> 2309 ... m <sub>1</sub> 2316
23, 24	3m0 4305	m <sub>1</sub> 2309 ... m <sub>1</sub> 2316 m <sub>1</sub> 2401 ... m <sub>1</sub> 2408
24	3m0 4306	m <sub>1</sub> 2401 ... m <sub>1</sub> 2408 m <sub>1</sub> 2409 ... m <sub>1</sub> 2416
24, 25	3m0 4307	m <sub>1</sub> 2409 ... m <sub>1</sub> 2416 m <sub>1</sub> 2501 ... m <sub>1</sub> 2508
25	3m0 4308	m <sub>1</sub> 2501 ... m <sub>1</sub> 2508 m <sub>1</sub> 2509 ... m <sub>1</sub> 2516
25, 26	3m0 4309	m <sub>1</sub> 2509 ... m <sub>1</sub> 2516 m <sub>1</sub> 2601 ... m <sub>1</sub> 2608
26	3m0 4310	m <sub>1</sub> 2601 ... m <sub>1</sub> 2608 m <sub>1</sub> 2609 ... m <sub>1</sub> 2616
26, 27	3m0 4311	m <sub>1</sub> 2609 ... m <sub>1</sub> 2616 m <sub>1</sub> 2701 ... m <sub>1</sub> 2708
27	3m0 4312	m <sub>1</sub> 2701 ... m <sub>1</sub> 2708 m <sub>1</sub> 2709 ... m <sub>1</sub> 2716
27, 28	3m0 4313	m <sub>1</sub> 2709 ... m <sub>1</sub> 2716 m <sub>1</sub> 2801 ... m <sub>1</sub> 2808
28	3m0 4314	m <sub>1</sub> 2801 ... m <sub>1</sub> 2808 m <sub>1</sub> 2809 ... m <sub>1</sub> 2816
28, 29	3m0 4315	m <sub>1</sub> 2809 ... m <sub>1</sub> 2816 m <sub>1</sub> 2901 ... m <sub>1</sub> 2908
29	3m0 4316	m <sub>1</sub> 2901 ... m <sub>1</sub> 2908 m <sub>1</sub> 2909 ... m <sub>1</sub> 2916
29, 30	3m0 4317	m <sub>1</sub> 2909 ... m <sub>1</sub> 2916 m <sub>1</sub> 3001 ... m <sub>1</sub> 3008
30	3m0 4318	m <sub>1</sub> 3001 ... m <sub>1</sub> 3008 m <sub>1</sub> 3009 ... m <sub>1</sub> 3016
30, 31	3m0 4319	m <sub>1</sub> 3009 ... m <sub>1</sub> 3016 m <sub>1</sub> 3101 ... m <sub>1</sub> 3109

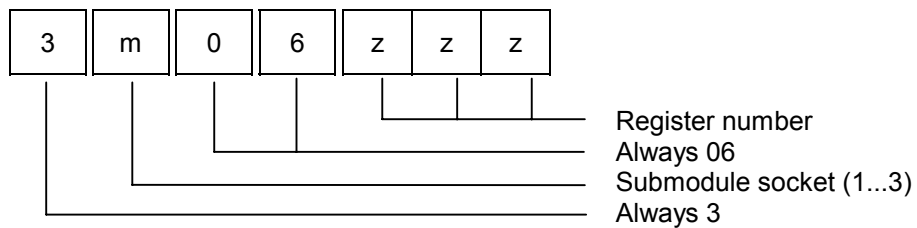
<b>Overlaying Inputs and Outputs of JX2-I/O modules</b>		
<b>8 Combined Outputs</b>		
<b>I/O Module Number</b>	<b>Registers</b>	<b>Output Numbers</b>
2	3m0 4322	m <sub>1</sub> 0201 ... m <sub>1</sub> 0208
2	3m0 4323	m <sub>1</sub> 0209 ... m <sub>1</sub> 0216
3	3m0 4324	m <sub>1</sub> 0301 ... m <sub>1</sub> 0308
3	3m0 4325	m <sub>1</sub> 0309 ... m <sub>1</sub> 0316
4	3m0 4326	m <sub>1</sub> 0401 ... m <sub>1</sub> 0408
4	3m0 4327	m <sub>1</sub> 0409 ... m <sub>1</sub> 0416
5	3m0 4328	m <sub>1</sub> 0501 ... m <sub>1</sub> 0508
5	3m0 4329	m <sub>1</sub> 0509 ... m <sub>1</sub> 0516
6	3m0 4330	m <sub>1</sub> 0601 ... m <sub>1</sub> 0608
6	3m0 4331	m <sub>1</sub> 0609 ... m <sub>1</sub> 0616
7	3m0 4332	m <sub>1</sub> 0701 ... m <sub>1</sub> 0708
7	3m0 4333	m <sub>1</sub> 0709 ... m <sub>1</sub> 0716
8	3m0 4334	m <sub>1</sub> 0801 ... m <sub>1</sub> 0808
8	3m0 4335	m <sub>1</sub> 0809 ... m <sub>1</sub> 0816
9	3m0 4336	m <sub>1</sub> 0901 ... m <sub>1</sub> 0908
9	3m0 4337	m <sub>1</sub> 0909 ... m <sub>1</sub> 0916
10	3m0 4338	m <sub>1</sub> 1001 ... m <sub>1</sub> 1008
10	3m0 4339	m <sub>1</sub> 1009 ... m <sub>1</sub> 1016
11	3m0 4340	m <sub>1</sub> 1101 ... m <sub>1</sub> 1108
11	3m0 4341	m <sub>1</sub> 1109 ... m <sub>1</sub> 1116
12	3m0 4342	m <sub>1</sub> 1201 ... m <sub>1</sub> 1208
12	3m0 4343	m <sub>1</sub> 1209 ... m <sub>1</sub> 1216
13	3m0 4344	m <sub>1</sub> 1301 ... m <sub>1</sub> 1308
13	3m0 4345	m <sub>1</sub> 1309 ... m <sub>1</sub> 1316
14	3m0 4346	m <sub>1</sub> 1401 ... m <sub>1</sub> 1408
14	3m0 4347	m <sub>1</sub> 1409 ... m <sub>1</sub> 1416
15	3m0 4348	m <sub>1</sub> 1501 ... m <sub>1</sub> 1508
15	3m0 4349	m <sub>1</sub> 1509 ... m <sub>1</sub> 1516
16	3m0 4350	m <sub>1</sub> 1601 ... m <sub>1</sub> 1608
16	3m0 4351	m <sub>1</sub> 1609 ... m <sub>1</sub> 1616
17	3m0 4352	m <sub>1</sub> 1701 ... m <sub>1</sub> 1708
17	3m0 4353	m <sub>1</sub> 1709 ... m <sub>1</sub> 1716
18	3m0 4354	m <sub>1</sub> 1801 ... m <sub>1</sub> 1808
18	3m0 4355	m <sub>1</sub> 1809 ... m <sub>1</sub> 1816
19	3m0 4356	m <sub>1</sub> 1901 ... m <sub>1</sub> 1908
19	3m0 4357	m <sub>1</sub> 1909 ... m <sub>1</sub> 1916

<b>Overlaying Inputs and Outputs of JX2-I/O modules</b>		
20	3m0 4358	m <sub>1</sub> 2001 ... m <sub>1</sub> 2008
20	3m0 4359	m <sub>1</sub> 2009 ... m <sub>1</sub> 2016
21	3m0 4360	m <sub>1</sub> 2101 ... m <sub>1</sub> 2108
21	3m0 4361	m <sub>1</sub> 2109 ... m <sub>1</sub> 2116
22	3m0 4362	m <sub>1</sub> 2201 ... m <sub>1</sub> 2208
22	3m0 4363	m <sub>1</sub> 2209 ... m <sub>1</sub> 2216
23	3m0 4364	m <sub>1</sub> 2301 ... m <sub>1</sub> 2308
23	3m0 4365	m <sub>1</sub> 2309 ... m <sub>1</sub> 2316
24	3m0 4366	m <sub>1</sub> 2401 ... m <sub>1</sub> 2408
24	3m0 4367	m <sub>1</sub> 2409 ... m <sub>1</sub> 2416
25	3m0 4368	m <sub>1</sub> 2501 ... m <sub>1</sub> 2508
25	3m0 4369	m <sub>1</sub> 2509 ... m <sub>1</sub> 2516
26	3m0 4370	m <sub>1</sub> 2601 ... m <sub>1</sub> 2608
26	3m0 4371	m <sub>1</sub> 2609 ... m <sub>1</sub> 2616
27	3m0 4372	m <sub>1</sub> 2701 ... m <sub>1</sub> 2708
27	3m0 4373	m <sub>1</sub> 2709 ... m <sub>1</sub> 2716
28	3m0 4374	m <sub>1</sub> 2801 ... m <sub>1</sub> 2808
28	3m0 4375	m <sub>1</sub> 2809 ... m <sub>1</sub> 2816
29	3m0 4376	m <sub>1</sub> 2901 ... m <sub>1</sub> 2908
29	3m0 4377	m <sub>1</sub> 2909 ... m <sub>1</sub> 2916
30	3m0 4378	m <sub>1</sub> 3001 ... m <sub>1</sub> 3008
30	3m0 4379	m <sub>1</sub> 3009 ... m <sub>1</sub> 3016
31	3m0 4380	m <sub>1</sub> 3101 ... m <sub>1</sub> 3108
31	3m0 4381	m <sub>1</sub> 3109 ... m <sub>1</sub> 3116
32	3m0 4382	m <sub>1</sub> 3201 ... m <sub>1</sub> 3208
32	3m0 4383	m <sub>1</sub> 3209 ... m <sub>1</sub> 3216

**Code of overlaid input registers of JX-SIO modules**



**Code of overlaid output registers of JX-SIO modules**



Overlaid Inputs and Outputs of JX-SIO Modules		
16 Combined Inputs		
Register Number	System Bus I/O Module Number	Inputs
3m0 5x10	7x (70 through 79)	m <sub>1</sub> 7x01 ... m <sub>1</sub> 7x16
3m0 5x11	7x (70 through 79)	m <sub>1</sub> 7x09 ... m <sub>1</sub> 7x24
3m0 5x12	7x (70 through 79)	m <sub>1</sub> 7x17 ... m <sub>1</sub> 7x32
3m0 5x13	7x (70 through 79)	m <sub>1</sub> 7x25 ... m <sub>1</sub> 7x40
3m0 5x14	7x (70 through 79)	m <sub>1</sub> 7x33 ... m <sub>1</sub> 7x48
3m0 5x15	7x (70 through 79)	m <sub>1</sub> 7x41 ... m <sub>1</sub> 7x56
3m0 5x16	7x (70 through 79)	m <sub>1</sub> 7x49 ... m <sub>1</sub> 7x64
8 Combined Inputs		
Register Number	System Bus I/O Module Number	Inputs
3m0 5x20	7x (70 through 79)	m <sub>1</sub> 7x01 ... m <sub>1</sub> 7x08
3m0 5x21	7x (70 through 79)	m <sub>1</sub> 7x09 ... m <sub>1</sub> 7x16
3m0 5x22	7x (70 through 79)	m <sub>1</sub> 7x17 ... m <sub>1</sub> 7x24
3m0 5x23	7x (70 through 79)	m <sub>1</sub> 7x25 ... m <sub>1</sub> 7x32
3m0 5x24	7x (70 through 79)	m <sub>1</sub> 7x33 ... m <sub>1</sub> 7x40
3m0 5x25	7x (70 through 79)	m <sub>1</sub> 7x41 ... m <sub>1</sub> 7x48
3m0 5x26	7x (70 through 79)	m <sub>1</sub> 7x49 ... m <sub>1</sub> 7x56
3m0 5x27	7x (70 through 79)	m <sub>1</sub> 7x57 ... m <sub>1</sub> 7x64

<b>Overlaid Inputs and Outputs of JX-SIO Modules</b>		
<b>16 Combined Outputs</b>		
<b>Register Number</b>	<b>System Bus I/O Module Number</b>	<b>Outputs</b>
3m0 6x10	7x (70 through 79)	m <sub>1</sub> 7x01 ... m <sub>1</sub> 7x16
3m0 6x11	7x (70 through 79)	m <sub>1</sub> 7x09 ... m <sub>1</sub> 7x24
3m0 6x12	7x (70 through 79)	m <sub>1</sub> 7x17 ... m <sub>1</sub> 7x32
3m0 6x13	7x (70 through 79)	m <sub>1</sub> 7x25 ... m <sub>1</sub> 7x40
3m0 6x14	7x (70 through 79)	m <sub>1</sub> 7x33 ... m <sub>1</sub> 7x48
3m0 6x15	7x (70 through 79)	m <sub>1</sub> 7x41 ... m <sub>1</sub> 7x56
3m0 6x16	7x (70 through 79)	m <sub>1</sub> 7x49 ... m <sub>1</sub> 7x64
<b>8 Combined Outputs</b>		
<b>Register Number</b>	<b>System Bus I/O Module Number</b>	<b>Outputs</b>
3m0 6x20	7x (70 through 79)	m <sub>1</sub> 7x01 ... m <sub>1</sub> 7x08
3m0 6x21	7x (70 through 79)	m <sub>1</sub> 7x09 ... m <sub>1</sub> 7x16
3m0 6x22	7x (70 through 79)	m <sub>1</sub> 7x17 ... m <sub>1</sub> 7x24
3m0 6x23	7x (70 through 79)	m <sub>1</sub> 7x25 ... m <sub>1</sub> 7x32
3m0 6x24	7x (70 through 79)	m <sub>1</sub> 7x33 ... m <sub>1</sub> 7x40
3m0 6x25	7x (70 through 79)	m <sub>1</sub> 7x41 ... m <sub>1</sub> 7x48
3m0 6x26	7x (70 through 79)	m <sub>1</sub> 7x49 ... m <sub>1</sub> 7x56
3m0 6x27	7x (70 through 79)	m <sub>1</sub> 7x57 ... m <sub>1</sub> 7x64

## 8.4 Description of Registers

### 8.4.1 Initialization and Diagnostics

Register 11m100: JX6-SB(-I) Status	
Function	Description
Read	Present interface state
Write	Acknowledging the status
Value range	Bit-oriented
Value after reset	0 with JX6-SB 2048 with JX6-SB-I

#### Meaning of the individual status register bits:

- Bit 0 : 1 = Timeout of JX2-I/O module
- Bit 1 : 1 = Output driver error with one of the JX2-I/O modules
- Bit 2 : 1 = Fatal system bus error
- Bit 4 : 1 = Timeout of JX2-Slave module
- Bit 5 : 1 = During execution of command 31 or 32 access to a register with the number 3m0 7xzz was made.
- Bit 8 : 1 = all expansion modules connected to the system bus have been initialized
- Bit 9 : 1 = all input data of expansion modules connected to the system bus are valid
- Bit 13 : 1 = Busy; the module is in the initialization state
- Bit 15 : 0 = Hardware configuration: JX6-SB  
1 = Hardware configuration: JX6-SB-I

Register 11m101: Commands	
Function	Description
Read	Command issued last
Write	Command to the JX6-SB(-I) module
Value range	0 ... 255
Value after reset	0

By writing into this register, certain actions are triggered on the module.

### Commands in "Master-Slave JX6-SB" mode:

- 30**      Initializing all expansion modules connected to the system bus.  
            This command can also be used to re-initialize the system bus.
- 31**      Reading the register with the number 3m0 7xzz
- 32**      Writing data to registers with the number 3m0 7xzz

### Example 15: Initialization in "Master-Slave JX6-SB" mode:

In this example, a JX6-SB(-I) module located in socket 1 is initialized in master-slave JX6-SB(-I) mode.

#### Program File

```

REGISTER_LOAD (111101, 30)      // Initializing the system bus
WHEN
    BIT_CLEAR (111100, 13)      // until Init is completed
    BIT_SET (111100, 8)        // all modules are initialized
    BIT_SET (111100, 9)        // all inputs are valid
THEN

IF
    REG 3102013 # 3            // three modules are connected
THEN
    // Fault message
ELSE
    // System bus configuration completed

```



<b>Register 3m0 2008: System Bus Status</b>	
<b>Function</b>	<b>Description</b>
Read	Present interface state
Write	Acknowledging the status
Value range	Bit-oriented
Value after reset	0

Apart from register 11m100, the system bus status can also be checked through register 3m02008. However, the information contained here is not as complete as the information contained in register 11m100. This register corresponds to register 2008 with NANO and JetControl 24x controllers.

### **Meaning of the individual system bus status bits:**

- Bit 3 : 1 = Timeout of JX2-I/O module
- Bit 4 : 1 = Timeout of JX2-Slave module

<b>Register 3m0 2029: System Bus Baud Rate</b>	
<b>Function</b>	<b>Description</b>
Read	Present baud rate
Write	New baud rate Changes in baud rates apply only after the system bus has been re-initialized.
Value range	4 ... 7
Value after reset	7 (1 Mbaud)

### Adjustable baud rates:

<b>4</b>	125 kBaud
<b>5</b>	250 kBaud
<b>6</b>	500 kBaud
<b>7</b>	1 MBaud

The maximum baud rate depends on the cable length of the system bus and on the number of modules connected to the bus. When setting the baud rate, please refer to chapter 2.5.

### Example 16: Adjusting the baud rate

The system bus is to be initialized at a baud rate of 125 kBaud. To do so, the baud rate on all connected JX-SIO modules has first to be set to 125 kBaud. Then, the JX-SIO modules have to be switched off and powered up again. The JX2-I/O and JX2-Slave modules will recognize the baud rate automatically.

The JX6-SB(-I) module is located in socket # 1.

#### Program File

```
REGISTER_LOAD (3102029, 4) // Baud rate = 125 kBaud
REGISTER_LOAD (111101, 30) // Initializing the system bus
WHEN
    BIT_CLEAR (111100, 13) // until Init is completed
    BIT_SET (111100, 8) // all modules are initialized
    BIT_SET (111100, 9) // all inputs are valid
THEN
// System bus configuration completed
```

<b>Register 3m0 2071: Present I/O Size on the System Bus</b>	
<b>Function</b>	<b>Description</b>
Read	I/O size of all expansion modules connected to the system bus
Write	Illegal
Value range	0 – 496
Value after reset	0

The maximum amount of modules to be connected to the system bus is limited by their I/O size. This register helps to easily determine the present I/O size and the resulting expandability of the system bus.

<b>Register 11m109: Firmware Version Number</b>	
<b>Function</b>	<b>Description</b>
Read	Present firmware version
Write	Illegal
Value range	0 – 2999
Value after reset	Version number * 100

From this register the firmware version number of the JX6-SB(-I) module can be read out. The value that has been read equals the product of the version number times a hundred. Thus, value 210, for example, refers to version 2.10. When the JX6-SB(-I) module is in the self test routine after switching on, the version number of the self test routine plus one thousand is displayed by this register.

$$\text{Register 11m109} = 1103$$

If the JX6-SB(-I) module is in the loading routine during the operating system update, the version number of the loading routine plus two thousand is displayed by this register.

$$\text{Register 11m109} = 2103$$

<b>Register 11m156: System Bus Cycle Time</b>	
<b>Function</b>	<b>Description</b>
Read	Time required for a complete cycle of the JX6-SB(-I) module. To determine the actual cycle time, the value contained in this register has to be multiplied by 10.24 $\mu$ s.
Write	Illegal
Value range	0 – 65535
Value after reset	0

During a complete cycle of the JX6-SB(-I) module, all input information of the connected expansion modules is read. At the same time, all output data that have changed are sent. In the background, the connected modules are cyclically checked for timeout. Registers of JX2-I/O or JX2-Slave modules, as well as the configuration register 3m07xzz of the JX-SIO module are accessed immediately, thus, increasing the cycle time. Therefore, the value contained in register 11m156 may vary.

<b>Register 3m0 2032: Delay during system bus initialization</b>	
<b>Function</b>	<b>Description</b>
Read	Present delay value
Write	New delay value
Value range	0 .. 255
Value after reset	10 (1000 ms)

When re-initializing the system bus a delay is to be maintained – depending on the system bus configuration - to ensure that all modules are in their initial state. If the delay is too short, it may happen that individual modules will not be recognized.

## 8.4.2 Short-circuit in one of the JX2-I/O modules

The JX6-SB(-I) module checks in the background for a short circuit reported by an output driver of a JX2-OD8 module. If this is the case, the module number of the module concerned will be entered into this register. At the same time, bit 1 in the status register of the JX6-SB(-I) module gets set.

Register 3m0 2027: Output Driver Error	
Function	Description
Read	Module number of the JX2-I/O module where an error of the output driver has occurred.
Write	Deleting the I/O module number
Value range	0, 2 .. 32
Value after reset	0

Number 2 is assigned to the first connected JX2-I/O module.

## 8.4.3 Timeout Monitoring

Monitoring telegrams are periodically exchanged between the JX6-SB(-I) module and the JX2-I/O modules, as well as the JX-SIO modules via system bus. By doing so, the JX6-SB(-I) module can detect whether the connection to one of the modules is interrupted. In the case of an interruption, in the JX6-SB(-I) status register bit 0 and in register 3m0 2008 bit 4 "Timeout of IO Module" gets set and the number of the relevant I/O module is entered into register 3m0 2011. Connection to the faulty module can be re-established only after a restart of the JX6-SB(-I) module using command 30.

The interval between two monitoring telegrams can be set in register 3m0 2028.

Register 3m0 2011: Number of I/O-Module with Timeout	
Function	Description
Read	Module number of the I/O module, or JX-SIO module where a timeout has occurred.
Write	Deleting the I/O module number
Value range	0, 2 .. 32, 70 ... 79
Value after reset	0

If in register 11m100 timeout of an JX2-I/O module is signaled by bit 0, the system bus number of the I/O module can be read out of this register.

<b>Register 3m0 2012: Number of a JX2-Slave Module with Timeout</b>	
Function	Description
Read	Slave module number of the JX2-Slave module, where a timeout has occurred.
Write	Deleting the slave module number
Value range	0, 2 .. 9
Value after reset	0

If in register 11m100 timeout of an JX2-Slave module is signaled by bit 4, the number of the slave module can be read out of this register.

<b>Register 3m0 2028: Monitoring Interval for JX2-I/O and JX-SIO Modules</b>	
Function	Description
Read	Monitoring interval for I/O modules in 10 ms increments
Write	New monitoring interval
Value range	0 – 255
Value after reset	20

The interval between two monitoring telegrams can be set in register 3m0 2028.

<b>Register 3m0 2073: Timeout Time for JX-SIO Modules</b>	
Function	Description
Read	Set timeout time
Write	New timeout time in milliseconds
Value range	0 – 255
Value after reset	3

The timeout time for JX-SIO modules is used to adjust the maximum interval between an access to a JX-SIO register and the receipt of the answer by the JX6-SB(-I) module. If the JX6-SB(-I) module does not receive an answer within the specified interval, then bit 3 in the system bus status register 3m0 2008 gets set and the I/O module number of the JX-SIO module is entered into register 3m0 2011 "Number of an I/O module with timeout".

The timeout time which is set after a reset may result in a timeout when accessing registers 3m07xzz of a JX-SIO module equipped with a great number of terminals. For more information refer to chapter 8.6.

### 8.4.4 Connected Expansion Modules

The JX6-SB(-I) module enters all modules connected to the system bus into a module array. In addition, the number of modules can be read directly. After initializing the system bus, the module array can be used to check whether all modules have been recognized.

<b>Register 3m0 2013: Amount of connected JX2-I/O and JX-SIO modules</b>	
<b>Function</b>	<b>Description</b>
Read	Present amount of recognized JX2-I/O and JX2-Slave modules
Write	Illegal
Value range	0 - 41
Value after reset	0

<b>Register 3m0 2014: Amount of connected JX2-Slave modules</b>	
<b>Function</b>	<b>Description</b>
Read	Present amount of recognized JX2-Slave modules
Write	Illegal
Value range	0 - 8
Value after reset	0

<b>Register 3m0 2070: Amount of connected JX-SIO modules</b>	
<b>Function</b>	<b>Description</b>
Read	Present amount of recognized JX-SIO modules  The amount of detected JX-SIO modules is calculated on the basis of the module with the highest module number, and not on the basis of the actual amount of modules.
Write	Illegal
Value range	0 - 10
Value after reset	0

Registers 3m02013; 3m01014, and 3m02070 provide information on the amount of recognized and activated expansion modules connected to the system bus. The module array offers the possibility to read out detailed information on the type of the connected module.

<b>Register 3m0 2015: Pointer to the module array</b>	
<b>Function</b>	<b>Description</b>
Read	Selected module
Write	Selection of a specific module
Value range	0 up to the amount of I/O modules
Value after reset	0

First, all JX2-I/O and JX2-Slave modules are listed in the module array according to their position on the system bus. Then, the JX-SIO modules follow according to their module address.

If there are gaps between JX-SIO module numbers, for every absent JX-SIO module value 252 "JX-SIO dummy module" is displayed.

<b>Register 3m0 2016: Module array</b>	
<b>Function</b>	<b>Description</b>
Read	Module array 3m0 2015 = 0 -> 3m0 2016 = Amount of modules  3m0 2015 = 1 -> 3m0 2016 = Code of the first module  3m0 2015 = 2 -> 3m0 2016 = Code of the second module
Write	Illegal
Value range	0 – 255
Value after reset	Amount of expansion modules

<b>Coding of Modules</b>		
<b>JX2-I/O Modules</b>		
<b>Module Code</b>	<b>Designation</b>	<b>Comment</b>
0	JX2-OD8	8 digital outputs
1	JX2-ID8	8 digital inputs
2	JX2-IO16	8 digital inputs and 8 digital outputs
3	JX2-IA4	4 analog inputs
4	JX2-OA4	4 analog outputs
5	JX2-CNT1	Counter input
6	JX2-PRN1	Module with Centronics interface
7	JX2-SER1	Module with serial interface



<b>Coding of Modules</b>		
<b>JX2-I/O Modules</b>		
<b>Module Code</b>	<b>Designation</b>	<b>Comment</b>
10	LJX7-CSL-ID16	16 digital inputs, IP67
11	LJX7-CSL-ID16-NPN	16 digital inputs (n), IP67
12	LJX7-CSL-OD8-2A	16 digital outputs, IP67
13	LJX7-CSL-OD16	8 digital outputs, IP67
14	LJX7-CSL-ID8-OD8	8 digital inputs and 8 digital outputs, IP67
<b>JX-SIO and Third-party Modules</b>		
<b>Module Code</b>	<b>Designation</b>	<b>Comment</b>
64	JX-SIO	System bus coupler for Smart I/O
65	CPV-Direct Valve Terminal	Festo AG & Co.
66	Terminal CPX	Festo AG & Co.
67	Valve terminal type 8640	Bürkert GmbH & Co. KG
68	SI unit EX12# - SCA1	SMC Pneumatik GmbH
70	Frequency converter 8200 vector	Lenze Drives Systems GmbH
<b>JX2-Slave Modules</b>		
<b>Module Code</b>	<b>Designation</b>	<b>Comment</b>
128	JX2-SV1	Position feedback controller, frequency converter ...
129	CAN-DIMA	Position feedback controller with integrated servo amplifier
130	JX2-SM2	Module for controlling 2 stepper motor amplifiers
131	JX2-SM1D	Module with integrated power unit for controlling a stepper motor
132	JX2-PID1	Module with 4 PID controllers
133	JX2-PROFI1	Slave for Profibus-DP
135	JetMove 200 Series	Position feedback controller with integrated servo amplifier
136	JX2-ProfIM	Master for Profibus-DP
146	JetMove 600 Series	Position feedback controller with integrated servo amplifier

---

<b>Coding of Modules</b>		
<b>Dummy Modules</b>		
<b>Module Code</b>	<b>Designation</b>	<b>Comment</b>
252	JX-SIO dummy-module	
253	JX2-Slave dummy module	
254	JX2-I/O Dummy Module	
255	Not identified	

### 8.4.5 Dummy Module Configuration

Dummy modules are used to configure modules on the system bus which actually do not exist. When assigning system bus module numbers, register and I/O numbers, the JX6-SB(-I) module treats dummy modules as if they were existing modules.

Dummy modules are of great advantage, for example, with series machines which are produced in several versions and configurations with a varying number of expansion modules connected to the system bus. By installing dummy modules, I/O and register numbers in the application program can remain the same.

<b>Register 3m0 2023: JX2-I/O dummy modules 2 ... 25</b>	
<b>Function</b>	<b>Description</b>
Read	Bit-coded list of configured JX2-I/O dummy modules # 2 through 25.
Write	Configuration of JX2-I/O dummy modules Bit 0: 1 = JX2-I/O module # 2 exists 0 = JX2-I/O module # 2 is a dummy module Bit 1: 1 = JX2-I/O module # 3 exists 0 = JX2-I/O module # 3 is a dummy module Bit 2: 1 = JX2-I/O module # 4 exists 0 = JX2-I/O module # 4 is a dummy module etc.
Value range	0 – FFFFFFF <sub>hex</sub>
Value after reset	FFFFFF <sub>hex</sub>

<b>Register 3m0 2030: JX2-I/O dummy modules 26 ... 32</b>	
<b>Function</b>	<b>Description</b>
Read	Bit-coded list of configured JX2-I/O dummy modules # 18 through 32.
Write	Configuration of new JX2-I/O dummy modules Bit 0: 1 = JX2-I/O module # 26 exists 0 = JX2-I/O module # 26 is a dummy module Bit 1: 1 = JX2-I/O module # 27 exists 0 = JX2-I/O module # 27 is a dummy module Bit 2: 1 = JX2-I/O module # 28 exists 0 = JX2-I/O module # 28 is a dummy module etc.
Value range	0 – FF <sub>hex</sub>
Value after reset	FF <sub>hex</sub>

The two registers 3m02023 and 3m02030 are for configuring JX2-I/O dummy modules. While doing so, each bit represents one module. Changes apply only after the system bus has been re-initialized.

Register 3m0 2024: JX2-Slave dummy modules	
Function	Description
Read	Bit-coded list of JX2-Slave dummy modules # 2 through 9. Bit 0: 1 = JX2-Slave module # 2 exists 0 = JX2-Slave module # 2 is a dummy module Bit 1: 1 = JX2-Slave module # 3 exists 0 = JX2-Slave module # 3 is a dummy module Bit 2: 1 = JX2-Slave module # 4 exists 0 = JX2-Slave module # 4 is a dummy module etc.
Write	Configuration of new JX2-Slave dummy modules
Value range	0 – FF <sub>hex</sub>
Value after reset	FF <sub>hex</sub>

Register 3m0 2024 is used to configure JX2-Slave dummy modules. While doing so, each bit represents one module. Changes apply only after the system bus has been re-initialized.

### Example 17: Configuring Dummy Modules

A machine is produced in two different versions. The simple version does not require analog inputs and one stepper motor less than the fully equipped version.

Due to the installation of dummy modules, all I/O and register numbers can be kept. The application program needs not be adjusted.

For the simple version, the JX2-Slave with the slave module number 2 and the JX2-I/O module with the I/O module number 3 have to be configured as dummy modules. The JX2-ID8 module keeps its I/O numbers IN 22401 ... IN 22408, and the JX2-SM2 module the register numbers starting from 13100.

The JX6-SB(-I) module continues to report that it has found three JX2-I/O and two JX2-Slave modules. However, the module code for dummy modules is entered into registers 310215 and 3102016 in the module array.

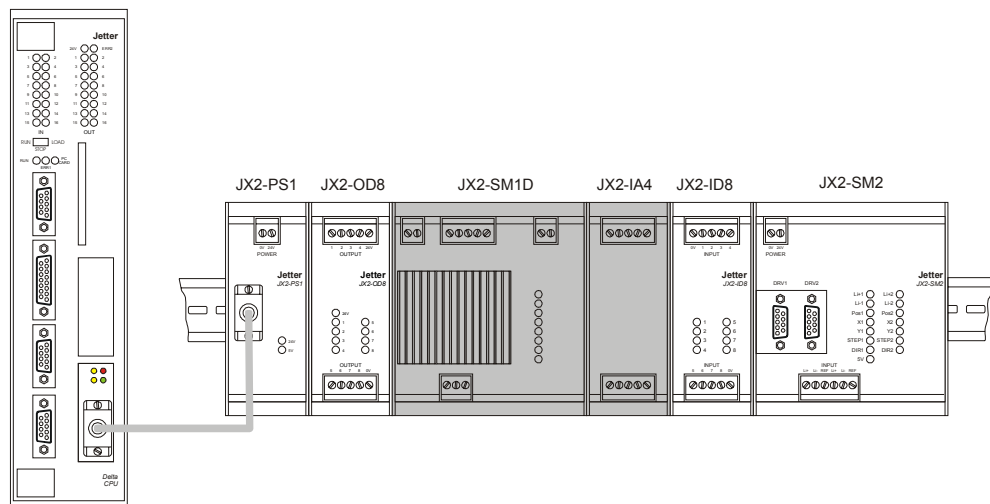


Fig. 17: Configuring Dummy Modules

**Program File**

```
BIT_CLEAR (3102023, 1)           // JX2-I/O module #3 is a dummy
BIT_CLEAR (3102024, 0)           // JX2-Slave module #2 is a dummy
REGISTER_LOAD (111101, 30)       // Initializing the system bus
WHEN
    BIT_CLEAR (111100, 13)       // until Init is completed
    BIT_SET (111100, 8)          // all modules are initialized
    BIT_SET (111100, 9)          // all inputs are valid
THEN
REGINC(3102015)                   // Index to first module
IF REG 3102016 # 0                 // first module is not a JX2-OD8
THEN GOTO lError
ELSE REGINC(3102015)              // Index to next module
IF REG 3102016 # 254             // second module is not a dummy
THEN GOTO lError
ELSE REGINC(3102015)              // Index to next module
IF REG 3102016 # 253             // third module is not a dummy
THEN GOTO lError
ELSE REGINC(3102015)              // Index to next module
IF REG 3102016 # 1               // fourth module is not a JX2-ID8
THEN GOTO lError
ELSE REGINC(3102015)              // Index to next module
IF REG 3102016 # 130             // fifth module is not a JX2-SM2
THEN GOTO lError
THEN
// all modules correctly found
```

## 8.4.6 Access to 32-bit JX-SIO Registers

Some registers of the JX-SIO module have a size of 32 bits. This causes problems to CPUs with 24-bit register architecture.

If the JX-SIO module transfers data in the most significant byte, when 32-bit registers are read, then, CPUs with 24-bit architecture can read the bits 24 through 31 out of register 3m07x95. In addition, special flag 2276 will be set.

When transferring 32-bit registers from a JX-SIO module, special flag 2277 can be used to select whether bits 24 through 31 are to be transferred from register 3m0 7x95 or directly from the register datum.

<b>Register 3m0 7x95: JX-SIO 32-bit Register Extension</b>	
<b>Function</b>	<b>Description</b>
Read	Most significant byte of 32-bit JX-SIO registers for read access
Write	Most significant byte of 32-bit JX-SIO registers for write access
Value range	0 – 255
Value after reset	0

<b>Special Flag 2276: JX-SIO 32-Bit Register Overflow</b>	
<b>Function</b>	<b>Description</b>
Read	If this special flag is set, register 3m07x95 is to be read out. 0 = No register overflow when reading 32-bit JX-SIO register 1 = Register overflow when reading 32-bit JX-SIO register
Write	Resetting the register overflow identifier The special flag must be reset by the user.
Value range	0 – 1
Value after reset	0

<b>Special Flag 2277: JX-SIO 32-Bit Register Selection</b>	
<b>Function</b>	<b>Description</b>
Read	Present state
Write	0 = When transferring 32-bit registers to a JX-SIO, bits 24 through 31 are transferred from the register datum 1 = When transferring 32-bit registers to a JX-SIO, bits 24 through 31 are transferred from register 3m0 7x95
Value range	0 – 1
Value after reset	0

### **Example 18: Transferring a 32-bit value to a function terminal on a D-CPU**

The value 123456789 is to be transferred as input data to the first function terminal of a JX-SIO module with the I/O module number 70. The JX6-SB(-I) module is located in the lower socket on a Delta CPU module.

The register architecture of the Delta CPU is 24 bits wide. Thus, larger values cannot be directly written. Since the value to be written is greater than 24 bits, it has to be split up. The lower 24 bits are transferred to the register itself, while the upper 8 bits are transferred to register 3m0 7x95.

In hexadecimal notation 123456789 is 07 5B CD 15<sub>HEX</sub>

#### **Program File**

```

BIT_SET (3102652, 5)           // Setting special flag 2277
REGISTER_LOAD (3107095, 0x07) // Upper 8 bits
REGISTER_LOAD (3107004, 1)    // First function terminal
REGISTER_LOAD (3107007, 0x5BCD15) // Transfer to JX-SIO register
// now, the value 0x075BCD15 is transferred to register 3107007

```

## 8.5 I/O Data and Module Registers

### 8.5.1 JX2-I/O Modules

16 inputs and outputs are unambiguously assigned to each JX2-I/O module, irrespective of the actual number of I/Os the module is equipped with. In addition, each JX2-I/O module is equipped with 10 registers with unambiguously assigned numbers. The function of these registers depends on the type of the JX2-I/O module.

Numbering and assignment of inputs and outputs, as well as of registers to the individual JX2-I/O modules is shown in the table below. For more information on the wildcard characters "m<sub>1</sub>" and "m" refer to chapter 0.

#### Note

These register numbers are supported by the Delta CPU with software version V2.28 or higher, resp. by the JetControl 647 with V3.00 or higher. The JX6-SB(-I) module has to be initialized in "Master-Slave JX6-SB" mode.

I/O and Register Numbers of JX2-I/O Modules			
System Bus I/O Module #	Input #	Output #	Register #
2	IN m <sub>1</sub> 0201 ... IN m <sub>1</sub> 0216	OUT m <sub>1</sub> 0201 ... OUT m <sub>1</sub> 0216	REG 3m0 3000 ... REG 3m0 3009
3	IN m <sub>1</sub> 0301 ... IN m <sub>1</sub> 0316	OUT m <sub>1</sub> 0301 ... OUT m <sub>1</sub> 0316	REG 3m0 3010 ... REG 3m0 3019
4	IN m <sub>1</sub> 0401 ... IN m <sub>1</sub> 0416	OUT m <sub>1</sub> 0401 ... OUT m <sub>1</sub> 0416	REG 3m0 3020 ... REG 3m0 3029
5	IN m <sub>1</sub> 0501 ... IN m <sub>1</sub> 0516	OUT m <sub>1</sub> 0501 ... OUT m <sub>1</sub> 0516	REG 3m0 3030 ... REG 3m0 3039
6	IN m <sub>1</sub> 0601 ... IN m <sub>1</sub> 0616	OUT m <sub>1</sub> 0601 ... OUT m <sub>1</sub> 0616	REG 3m0 3040 ... REG 3m0 3049
7	IN m <sub>1</sub> 0701 ... IN m <sub>1</sub> 0716	OUT m <sub>1</sub> 0701 ... OUT m <sub>1</sub> 0716	REG 3m0 3050 ... REG 3m0 3059
8	IN m <sub>1</sub> 0801 ... IN m <sub>1</sub> 0816	OUT m <sub>1</sub> 0801 ... OUT m <sub>1</sub> 0816	REG 3m0 3060 ... REG 3m0 3069
9	IN m <sub>1</sub> 0901 ... IN m <sub>1</sub> 0916	OUT m <sub>1</sub> 0901 ... OUT m <sub>1</sub> 0916	REG 3m0 3070 ... REG 3m0 3079
10	IN m <sub>1</sub> 1001 ... IN m <sub>1</sub> 1016	OUT m <sub>1</sub> 1001 ... OUT m <sub>1</sub> 1016	REG 3m0 3080 ... REG 3m0 3089
11	IN m <sub>1</sub> 1101 ... IN m <sub>1</sub> 1116	OUT m <sub>1</sub> 1101 ... OUT m <sub>1</sub> 1116	REG 3m0 3090 ... REG 3m0 3099
12	IN m <sub>1</sub> 1201 ... IN m <sub>1</sub> 1216	OUT m <sub>1</sub> 1201 ... OUT m <sub>1</sub> 1216	REG 3m0 3100 ... REG 3m0 3109
13	IN m <sub>1</sub> 1301 ... IN m <sub>1</sub> 1316	OUT m <sub>1</sub> 1301 ... OUT m <sub>1</sub> 1316	REG 3m0 3110 ... REG 3m0 3119
14	IN m <sub>1</sub> 1401 ... IN m <sub>1</sub> 1416	OUT m <sub>1</sub> 1401 ... OUT m <sub>1</sub> 1416	REG 3m0 3120 ... REG 3m0 3129
15	IN m <sub>1</sub> 1501 ... IN m <sub>1</sub> 1516	OUT m <sub>1</sub> 1501 ... OUT m <sub>1</sub> 1516	REG 3m0 3130 ... REG 3m0 3139



<b>I/O and Register Numbers of JX2-I/O Modules</b>			
16	IN m <sub>1</sub> 1601 ... IN m <sub>1</sub> 1616	OUT m <sub>1</sub> 1601 ... OUT m <sub>1</sub> 1616	REG 3m0 3140 ... REG 3m0 3149
17	IN m <sub>1</sub> 1701 ... IN m <sub>1</sub> 1716	OUT m <sub>1</sub> 1701 ... OUT m <sub>1</sub> 1716	REG 3m0 3150 ... REG 3m0 3159
18	IN m <sub>1</sub> 1801 ... IN m <sub>1</sub> 1816	OUT m <sub>1</sub> 1801 ... OUT m <sub>1</sub> 1816	REG 3m0 3160 ... REG 3m0 3169
19	IN m <sub>1</sub> 1901 ... IN m <sub>1</sub> 1916	OUT m <sub>1</sub> 1901 ... OUT m <sub>1</sub> 1916	REG 3m0 3170 ... REG 3m0 3179
20	IN m <sub>1</sub> 2001 ... IN m <sub>1</sub> 2016	OUT m <sub>1</sub> 2001 ... OUT m <sub>1</sub> 2016	REG 3m0 3180 ... REG 3m0 3189
21	IN m <sub>1</sub> 2101 ... IN m <sub>1</sub> 2116	OUT m <sub>1</sub> 2101 ... OUT m <sub>1</sub> 2116	REG 3m0 3190 ... REG 3m0 3199
22	IN m <sub>1</sub> 2201 ... IN m <sub>1</sub> 2216	OUT m <sub>1</sub> 2201 ... OUT m <sub>1</sub> 2216	REG 3m0 3200 ... REG 3m0 3209
23	IN m <sub>1</sub> 2301 ... IN m <sub>1</sub> 2316	OUT m <sub>1</sub> 2301 ... OUT m <sub>1</sub> 2316	REG 3m0 3210 ... REG 3m0 3219
24	IN m <sub>1</sub> 2401 ... IN m <sub>1</sub> 2416	OUT m <sub>1</sub> 2401 ... OUT m <sub>1</sub> 2416	REG 3m0 3220 ... REG 3m0 3229
25	IN m <sub>1</sub> 2501 ... IN m <sub>1</sub> 2516	OUT m <sub>1</sub> 2501 ... OUT m <sub>1</sub> 2516	REG 3m0 3230 ... REG 3m0 3239
26	IN m <sub>1</sub> 2601 ... IN m <sub>1</sub> 2616	OUT m <sub>1</sub> 2601 ... OUT m <sub>1</sub> 2616	REG 3m0 3240 ... REG 3m0 3249
27	IN m <sub>1</sub> 2701 ... IN m <sub>1</sub> 2716	OUT m <sub>1</sub> 2701 ... OUT m <sub>1</sub> 2716	REG 3m0 3250 ... REG 3m0 3259
28	IN m <sub>1</sub> 2801 ... IN m <sub>1</sub> 2816	OUT m <sub>1</sub> 2801 ... OUT m <sub>1</sub> 2816	REG 3m0 3260 ... REG 3m0 3269
29	IN m <sub>1</sub> 2901 ... IN m <sub>1</sub> 2916	OUT m <sub>1</sub> 2901 ... OUT m <sub>1</sub> 2916	REG 3m0 3270 ... REG 3m0 3279
30	IN m <sub>1</sub> 3001 ... IN m <sub>1</sub> 3016	OUT m <sub>1</sub> 3001 ... OUT m <sub>1</sub> 3016	REG 3m0 3280 ... REG 3m0 3289
31	IN m <sub>1</sub> 3101 ... IN m <sub>1</sub> 3116	OUT m <sub>1</sub> 3101 ... OUT m <sub>1</sub> 3116	REG 3m0 3290 ... REG 3m0 3299
32	IN m <sub>1</sub> 3201 ... IN m <sub>1</sub> 3216	OUT m <sub>1</sub> 3201 ... OUT m <sub>1</sub> 3216	REG 3m0 3300 ... REG 3m0 3309

## 8.5.2 JX-SIO

64 digital inputs and outputs are unambiguously assigned to each JX2-SIO module, irrespective of the actual number of I/Os the module is equipped with. In addition to this, each JX-SIO module is provided with 12 registers for analog inputs and 12 registers for analog outputs. 100 additional registers are available for diagnostic, administration and configuration purposes.

Assignment of I/O and register numbers for all third-party modules which can be connected to the system bus is the same as for JX-SIO modules.

Numbering and assignment of inputs and outputs, as well as of registers to the individual JX2-SIO modules is shown in the table below. For more information on the wildcard characters "m<sub>1</sub>" and "m" refer to chapters 0 and 0.

### Note

These register numbers are supported by the Delta CPU with software version V2.28 or higher, resp. by the JetControl 647 with V3.00 or higher. The JX6-SB(-I) module has to be initialized in "Master-Slave JX6-SB" mode.

<b>I/O and Register Numbers of JX-SIO Modules</b>			
<b>System Bus I/O Module #</b>	<b>Digital Inputs Digital Outputs</b>	<b>Analog Inputs Analog Outputs</b>	<b>Registers</b>
70	IN m <sub>1</sub> 7001... IN m <sub>1</sub> 7064  OUT m <sub>1</sub> 7001... OUT m <sub>1</sub> 7064	REG 3m0 5050... REG 3m0 5071  REG 3m0 6050... REG 3m0 6071	REG 3m0 7000... REG 3m0 7099
71	IN m <sub>1</sub> 7101... IN m <sub>1</sub> 7164  OUT m <sub>1</sub> 7101... OUT m <sub>1</sub> 7164	REG 3m0 5150... REG 3m0 5171  REG 3m0 6150... REG 3m0 6171	REG 3m0 7100... REG 3m0 7199
72	IN m <sub>1</sub> 7201... IN m <sub>1</sub> 7264  OUT m <sub>1</sub> 7201... OUT m <sub>1</sub> 7264	REG 3m0 5250... REG 3m0 5271  REG 3m0 6250... REG 3m0 6271	REG 3m0 7200... REG 3m0 7299
73	IN m <sub>1</sub> 7301... IN m <sub>1</sub> 7364  OUT m <sub>1</sub> 7301... OUT m <sub>1</sub> 7364	REG 3m0 5350... REG 3m0 5371  REG 3m0 6350... REG 3m0 6371	REG 3m0 7300... REG 3m0 7399
74	IN m <sub>1</sub> 7401... IN m <sub>1</sub> 7464  OUT m <sub>1</sub> 7401... OUT m <sub>1</sub> 7464	REG 3m0 5450... REG 3m0 5471  REG 3m0 6450... REG 3m0 6471	REG 3m0 7400... REG 3m0 7499
75	IN m <sub>1</sub> 7501... IN m <sub>1</sub> 7564  OUT m <sub>1</sub> 7501... OUT m <sub>1</sub> 7564	REG 3m0 5550... REG 3m0 5571  REG 3m0 6550... REG 3m0 6571	REG 3m0 7500... REG 3m0 7599
76	IN m <sub>1</sub> 7601... IN m <sub>1</sub> 7664  OUT m <sub>1</sub> 7601... OUT m <sub>1</sub> 7664	REG 3m0 5650... REG 3m0 5671  REG 3m0 6650... REG 3m0 6671	REG 3m0 7600... REG 3m0 7699

<b>I/O and Register Numbers of JX-SIO Modules</b>			
77	IN m <sub>1</sub> 7701...	REG 3m0 5750...	REG 3m0 7700...
	IN m <sub>1</sub> 7764	REG 3m0 5771	REG 3m0 7799
	OUT m <sub>1</sub> 7701...	REG 3m0 6750...	
	OUT m <sub>1</sub> 7764	REG 3m0 6771	
78	IN m <sub>1</sub> 7801...	REG 3m0 5850...	REG 3m0 7800...
	IN m <sub>1</sub> 7864	REG 3m0 5871	REG 3m0 7899
	OUT m <sub>1</sub> 7801...	REG 3m0 6850...	
	OUT m <sub>1</sub> 7864	REG 3m0 6871	
79	IN m <sub>1</sub> 7901...	REG 3m0 5950...	REG 3m0 7900...
	IN m <sub>1</sub> 7964	REG 3m0 5971	REG 3m0 7999
	OUT m <sub>1</sub> 7901...	REG 3m0 6950...	
	OUT m <sub>1</sub> 7964	REG 3m0 6971	

### 8.5.3 JX2-Slave Modules

<b>Register 3m 1xyzz: Registers of JX2-Slave Modules</b>	
<b>Function</b>	<b>Description</b>
Read	Depending on the function of the slave
Write	Depending on the function of the slave
Value range	Depending on the function of the slave
Value after reset	Depending on the function of the slave

Registers of JX2-Slave modules can directly be accessed via register number 3m11xyzz. For more information on the wildcard characters refer to chapter 8.1.5.

**Note**

JX2-Slave modules are supported only by the JX6-SB-I module. These register numbers are supported by the Delta CPU with software version V2.28 or higher, resp. by the JetControl 647 with V3.00 or higher. The JX6-SB(-I) module has to be started in "Master-Slave JX6-SB" mode.

## 8.6 Timeout Adjustment for JX-SIO Modules

The response time required by the JX-SIO module for accesses to registers 3m07xzz depends on the degree of extension of the Smart I/O module. The more terminals are connected, the more computing time is required by the JX-SIO module.

After power-up, the CPU waits up to 5 ms for a response from the JX6-SB(-I) module when accessing registers. Once this time limit is exceeded, special flag 2105 will be set.

When accessing registers, the JX6-SB(-I) module waits up to 3 ms for a response from the JX-SIO module. Once this time limit is exceeded, bit 0 in status register 11m100 will be set and the red LED on the JX6-SB(-I) module will be lit.

Both timeout values can be modified through registers and possibly have to be adjusted.

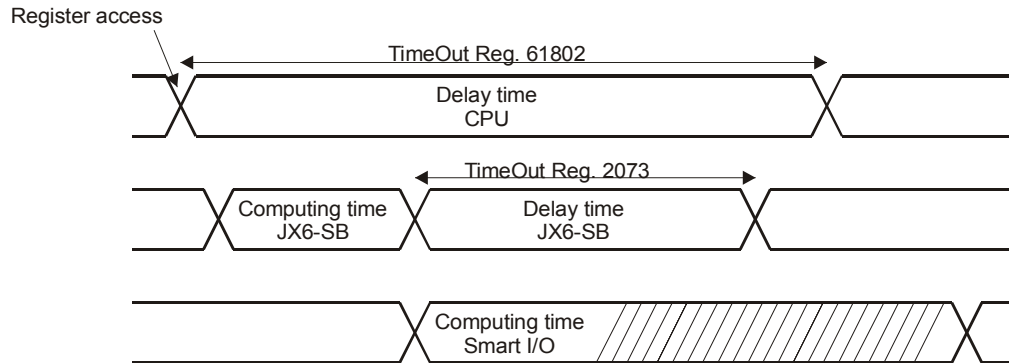


Fig. 18: Access to Registers 3m07xzz

The disadvantage with increasing the timeout through register 61802 is that the CPU waits until it receives a response or until the timeout specified in register 61802 has expired. Also, no task switch is carried out.

When initiating a read and write access through the commands 31 and 32, the CPU can carry out a task switch during the waiting period and process other tasks.

Register 3m0 2073: Timeout Time for JX-SIO Modules	
Function	Description
Read	Set timeout time
Write	New timeout time in milliseconds
Value range	0 – 255
Value after reset	3.

The timeout time for JX-SIO modules is used to adjust the maximum interval between an access to a JX-SIO register and the receipt of the answer by the JX6-SB(-I) module. If the JX6-SB(-I) module does not receive an answer within the specified interval, then bit 3 in the system bus status register 3m0 2008 gets set and the I/O module number of the JX-SIO module is entered into register 3m0 2011 "Number of an I/O module with timeout".

## 8.6.1 Commands 31 and 32

Register 11m201: JX-SIO register index	
Function	Description
Read	Present index to a JX-SIO register
Write	New index to a JX-SIO register The register number corresponds to the JX-SIO registers, but without the prefix "3m0"
Value range	7000 - 7999
Value after reset	0

Register 11m202: JX-SIO register datum	
Function	Description
Read	Present contents of a JX-SIO register
Write	New contents of a JX-SIO register
Value range	24 bits
Value after reset	0

For read and write accesses to JX-SIO registers, the two registers 11m201 and 11m202 form a pair. These two registers are for transferring the register number and contents when using the two commands 31 and 32.

### Example 19: Writing command 32 into register 3207xzz

This sample program is for configuring the analog input terminal of a Smart I/O module. In order not to block the CPU during the configuration phase, command 32 will be used. The JX6-SB(-I) module is located in the first socket of the CPU board. I/O module number 70 is assigned to the JX-SIO module.

#### Program File

```
REGISTER_LOAD (111201, 7010) //Config. reg. - first analog input
REGISTER_LOAD (111202, 0x0000) //Config. data - first analog input
REGISTER_LOAD (111101, 32) // Issuing command 32
WHEN
    BIT_CLEAR(111100, 13)
THEN
//Program continuation
```

## 8.7 Special Flags

The JX6-SB(-I) module indicates certain error conditions through special flags. This feature facilitates detailed error analysis. All special flags of the JX6-SB(-I) module can be reached through overlaid registers only.

The JX6-SB(-I) module deletes all special flags after the CPU has been powered-up. A re-initialization of the system bus will not affect the special flags.

<b>JX6-SB(-I) Special Flags</b>			
<b>Special Flags</b>	<b>Registers</b>	<b>Bit</b>	<b>Description</b>
2048	3m0 2638	0	Timeout during an access to a JX2-I/O module
2049	3m0 2638	1	Timeout during an access to a JX2-Slave module
2050	3m0 2638	2	Timeout during an access to a JX2-I/O register
2065	3m0 2639	1	Activated error signalization for output driver errors
2067	3m0 2639	3	Fatal system bus error
2270	3m0 2651	14	Timeout during an access to an inactive JX-SIO module
2272	3m0 2652	0	Access to an unknown JX-SIO register
2273	3m0 2652	1	Access to a register which is not supported by this JX-SIO module, e.g. configuration of an analog input, although there is no analog input terminal.
2274	3m0 2652	2	Timeout when monitoring a JX-SIO module
2275	3m0 2652	3	JX-SIO has carried out an internal reset, or is not ready
2276	3m0 2652	4	Overflow during a read access to a 32-bit register
2277	3m0 2652	5	Selecting the upper 8 bits for 32-bit write access to JX-SIO registers

## 8.8 Overview of Registers

Registers	Description	1) Value Range 2) Reset Value
<b>Initialization and Diagnostics</b>		
11m100	JX6-SB(-I) Status	1) bit-oriented 2) 0 or 2048
11m101	Commands	1) 30 - 32 2) 0
11m109	Firmware Version Number	1) 0 - 2999 2) Firmware version
11m156	System Bus Cycle Time	1) 0 - 65535 2) 0
3m02008	System Bus Status	1) bit-oriented 2) 0
3m02029	System Bus Baud Rate	1) 4 – 7 2) 7
3m02032	Delay during system bus initialization	1) 0 - 65535 2) 10 = 1000 ms
3m02071	Present I/O Size on the System Bus	1) 0 - 496 2) 0
<b>Short Circuit</b>		
3m02027	Error in the output driver	1) 0, 2-32 2) 0
<b>Timeout Monitoring</b>		
3m02011	Number of I/O-Module with Timeout	1) 0, 2-32, 70-79 2) 0
3m02012	Number of a JX2-Slave Module with Timeout	1) 2 - 8 2) 0
3m02028	Monitoring Interval for JX2-I/O and JX-SIO Modules	1) 0 - 255 2) 20 = 20 ms
3m02073	Timeout Time for JX-SIO Modules	1) 0 – 255 2) 3 = 3 ms

Registers	Description	1) Value Range 2) Reset Value
<b>Connected Expansion Modules</b>		
3m02013	Amount of connected JX2-I/O and JX-SIO modules	1) 0 - 41 2) 0
3m02014	Amount of connected JX2-Slave modules	1) 0 - 8 2) 0
3m02015	Pointer to the module array	1) 0 - 41 2) 0
3m02016	Module array	1) 0 2) 0 – 255
3m02070	Amount of connected JX-SIO modules	1) 0 - 10 2) 0
<b>Dummy Module Configuration</b>		
3m02023	JX2-I/O dummy modules 2 ... 25	1) 0 - FFFFFFFF <sub>HEX</sub> 2) FFFFFFFF <sub>HEX</sub>
3m02024	JX2-Slave dummy modules	1) 0 - 255 2) 255
3m02030	JX2-I/O dummy modules 26 ... 32	1) 0 - 255 2) 255
<b>Access to 32-bit JX-SIO Registers</b>		
3m07x95	JX-SIO 32-bit Register Extension	1) 0 - 255 2) 0
<b>Access to 3m07xzz Registers using Command 31 and 32</b>		
11m201	JX-SIO register index	1) 7000 - 7999 2) 0
11m202	JX-SIO register datum	1) 24 bit 2) 0
<b>Special Flags</b>		
3m02638	Bit 0 : Special flag 2048 Timeout of JX2-I/O module  Bit 1 : Special flag 2049 Timeout of JX2-Slave Module  Bit 2 : Special flag 2048 Timeout during register access to JX2-I/O module	1) 0 - 65535 2) 0



Registers	Description	1) Value Range 2) Reset Value
3m02619	Bit 1 : Special flag 2065 Error signal  Bit 3 : Special flag 2067 Fatal system bus error	
3m02651	Bit 14 : Special flag 2270 Inactive JX-SIO module	
3m02652	Bit 0 : Special flag 2272 Unknown JX-SIO register  Bit 1 : Special flag 2273 Not supported JX-SIO register  Bit 2 : Special flag 2274 Timeout of the JX-SIO Module  Bit 3 : Special flag 2275 JX-SIO module not ready for operation  Bit 4 : Special flag 2276 32-bit register overflow  Bit 5 : Special flag 2277 32-bit register selection	
<b>16 Combined JX2-I/O Inputs</b>		
3m04061	m <sub>1</sub> 0201 ... m <sub>1</sub> 0208	1) 0 – 65535 2) Input map
3m04062	m <sub>1</sub> 0209 ... m <sub>1</sub> 0216, m <sub>1</sub> 0201 ... m <sub>1</sub> 0208	
3m04063	m <sub>1</sub> 0301 ... m <sub>1</sub> 0308, m <sub>1</sub> 0209 ... m <sub>1</sub> 0216	
...		
3m04118	m <sub>1</sub> 3009 ... m <sub>1</sub> 3016, m <sub>1</sub> 3001 ... m <sub>1</sub> 3008	
3m04119	m <sub>1</sub> 3101 ... m <sub>1</sub> 3109, m <sub>1</sub> 3009 ... m <sub>1</sub> 3016	
<b>8 Combined JX2-I/O Inputs</b>		
3m04122	m <sub>1</sub> 0201 ... m <sub>1</sub> 0208	1) 0 – 255 2) Input map
3m04123	m <sub>1</sub> 0209 ... m <sub>1</sub> 0216	
3m04124	m <sub>1</sub> 0301 ... m <sub>1</sub> 0308	
...		
3m04182	m <sub>1</sub> 3201 ... m <sub>1</sub> 3208	
3m04183	m <sub>1</sub> 3209 ... m <sub>1</sub> 3216	
<b>16 Combined JX2-I/O Outputs</b>		
3m04261	m <sub>1</sub> 0201 ... m <sub>1</sub> 0208	1) 0 – 65535 2) 0
3m04262	m <sub>1</sub> 0209 ... m <sub>1</sub> 0216, m <sub>1</sub> 0201 ... m <sub>1</sub> 0208	
3m04263	m <sub>1</sub> 0301 ... m <sub>1</sub> 0308, m <sub>1</sub> 0209 ... m <sub>1</sub> 0216	
...		
3m04318	m <sub>1</sub> 3009 ... m <sub>1</sub> 3016, m <sub>1</sub> 3001 ... m <sub>1</sub> 3008	

Registers	Description	1) Value Range 2) Reset Value
3m04319	m <sub>1</sub> 3101 ... m <sub>1</sub> 3109, m <sub>1</sub> 3009 ... m <sub>1</sub> 3016	
<b>8 Combined JX2-I/O Outputs</b>		
3m04322	m <sub>1</sub> 0201 ... m <sub>1</sub> 0208	1) 0 – 255
3m04323	m <sub>1</sub> 0209 ... m <sub>1</sub> 0216	2) 0
3m04324	m <sub>1</sub> 0301 ... m <sub>1</sub> 0308	
...		
3m04382	m <sub>1</sub> 3201 ... m <sub>1</sub> 3208	
3m04383	m <sub>1</sub> 3209 ... m <sub>1</sub> 3216	
<b>16 Combined JX-SIO Inputs</b>		
3m05x10	m <sub>1</sub> 7x01 ... m <sub>1</sub> 7x16	1) 0 – 65535
...		2) Input map
3m05x16	m <sub>1</sub> 7x49 ... m <sub>1</sub> 7x64	
<b>8 Combined JX-SIO Inputs</b>		
3m05x20	m <sub>1</sub> 7x01 ... m <sub>1</sub> 7x08	1) 0 – 255
...		2) Input map
3m05x27	m <sub>1</sub> 7x57 ... m <sub>1</sub> 7x64	
<b>16 Combined JX-SIO Outputs</b>		
3m0 6x10	m <sub>1</sub> 7x01 ... m <sub>1</sub> 7x16	1) 0 – 65535
...		2) 0
3m0 6x16	m <sub>1</sub> 7x49 ... m <sub>1</sub> 7x64	
<b>8 Combined JX-SIO Outputs</b>		
3m0 6x20	m <sub>1</sub> 7x01 ... m <sub>1</sub> 7x08	1) 0 – 255
...		2) 0
3m0 6x27	m <sub>1</sub> 7x57 ... m <sub>1</sub> 7x64	
<b>Expansion Module Registers</b>		
3m03xxz	Registers of JX2-I/O expansion modules	1) 24-bit 2) module-dependent
3m07xzz	Registers of JX-SIO expansion modules	1) 24-bit / 32-bit 2) module-dependent
3m1xzzz	Registers JX2-Slave expansion modules	1) 24-bit / 32-bit 2) module-dependent

## 9 JX2-Slave Modules connected to the JX6-SB-I Module

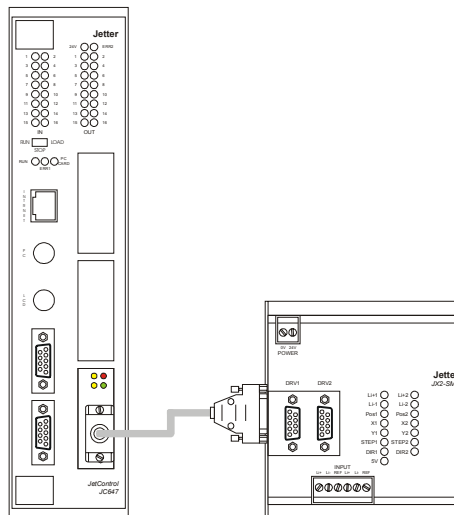
All intelligent JX2-Slave and JetMove modules produced by Jetter AG can be connected to the JX6-SB-I module. JetSym supports commissioning and programming of axes and similar peripheral devices connected to such modules.

### 9.1 Oscilloscope Mode in JetSym

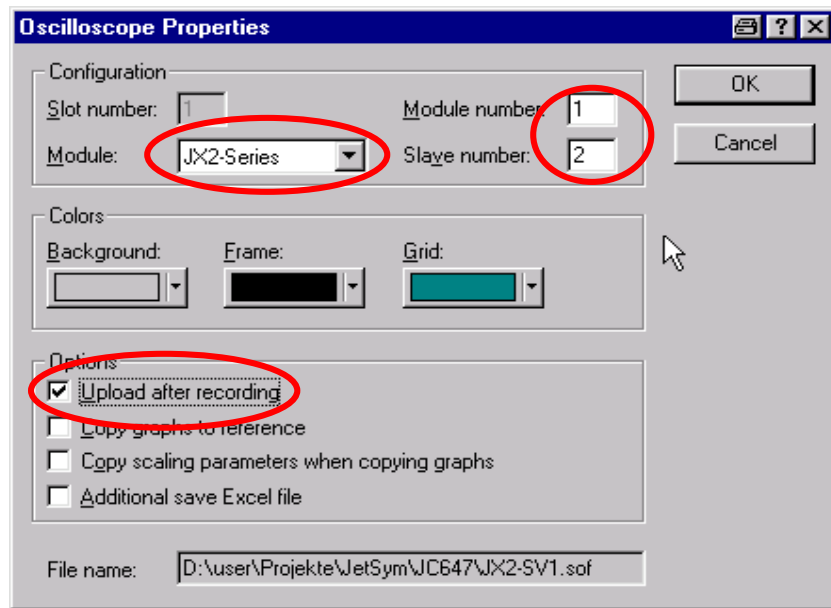
The JetSym oscilloscope mode is for recording JX2-Slave data and visualizing them in a window. After recording data are buffered on the JX2-Slave module and then transferred to the PC for visualization purposes.

For information on working with the oscilloscope mode, please refer to the JetSym online help.

#### Example 20: Recording Data and Visualizing them in JetSym



In this example, the actual position, actual speed and tracking error of an axis connected to a JX2-SV1 module are recorded. Create a new oscilloscope file in JetSym and open the "Properties" dialog.

**Module**

For a JX2-Slave connected to a JX6-SB-I module, select "JX2-Series".

**Module number**

Enter here the number of the socket where the JX6-SB-I module is located. In this example the JX2-SV1 module has been inserted into socket # 1.

**Slave number (JetSym V2.10)**

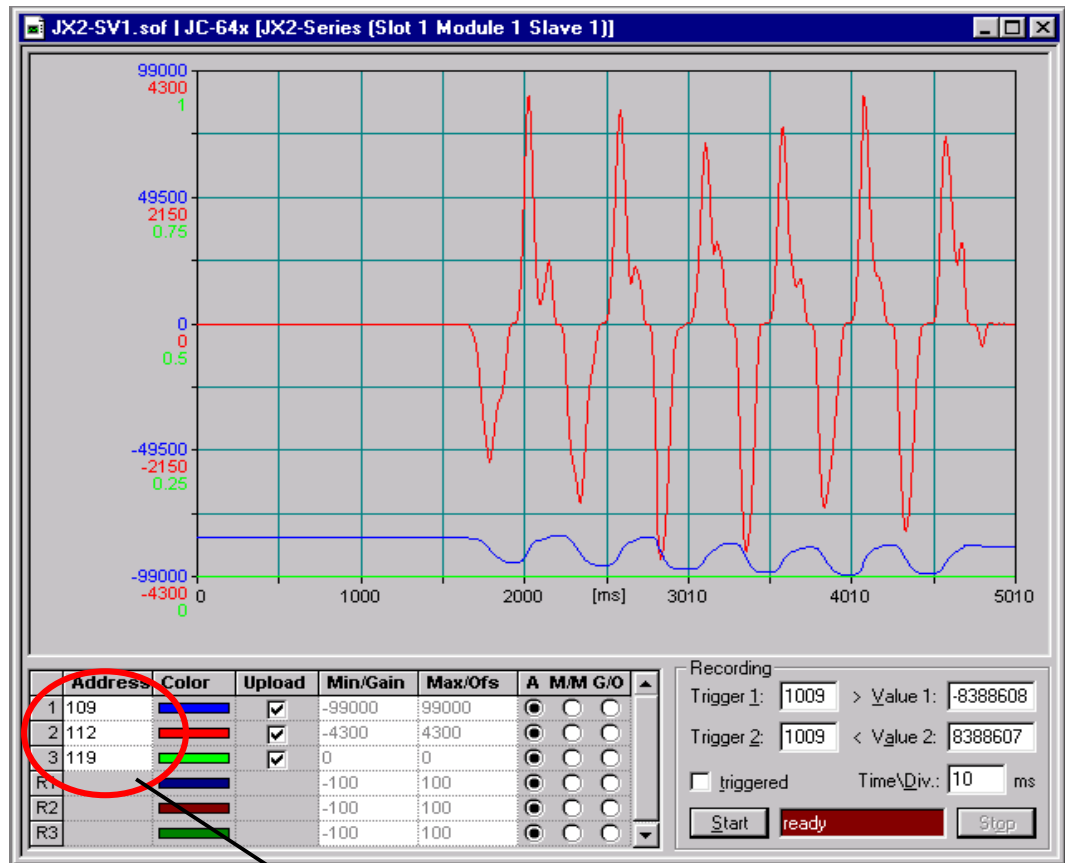
As "Slave number" enter the number of the JX2-SV1 module.

**Slave number (JetSym before V2.10)**

As "Slave number" enter the number of the JX2-SV1 module minus 1.

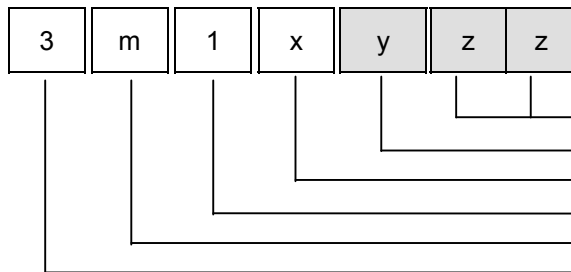
**Upload after recording**

Also check the option "Upload after recording". If this option is checked, the recorded data are automatically uploaded to the PC when recording is stopped.



Register code for JX2-Slave

Modules in oscilloscope mode



- Register number (0...99)
- Axis or controller number (0...9)
- „Slave number“ (2...9)
- Always 1
- „Module number“ (1...3)
- Always 3

Enter the register number "zz" and the axis or controller number "y" into column "Address". JetSVM independently calculates the actual register numbers on the basis of the settings made in the dialog "Properties".  
 In our example, registers 3112**109**, 3112**112**, and 3112**119** are to be recorded.

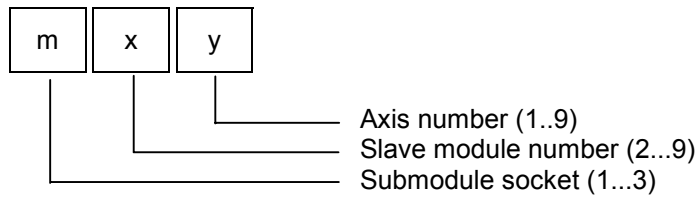
## 9.2 Axis Programming

With the JetControl controller JC647 axes connected to the JX6SB-I module can directly be addressed using the JetSym instructions `POS`, `AXARR` and `Actual_POS`.

### Note

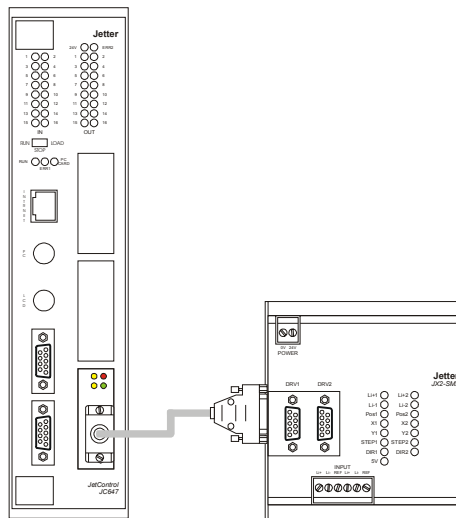
These JetSym instructions for axes connected to the JX-SB-I module are supported by JetControl JC647 starting from version V3.00.

### Coding of the axis number



The socket number "m" indicates in which socket of the CPU module the submodule is located. The system bus slave module number "x" is to distinguish between several JX2-slave modules. The axis number "y" is to distinguish between several axes of the JX2-slave module.

### Example 21: Positioning an axis



In this example the axis connected to the JX2-SV1 module is moved to position 20400 with a speed of 500. The JX6-SB-I module is located in socket # 1. The JX2-SM2 module is the first connected JX2-Slave module. So, this axis number is "121".

### Program File

```

POS (121, 20400, 500) // Starting the positioning
WHEN
    AXARR 121 // Waiting until axis has reached its position
THEN //Program continuation
    
```

# 10 Operating Mode "Master-Master"

## 10.1 Description of Registers

Register 11m100: Status	
Function	Description
Read	Status information on the JX6-SB(-I) module
Write	Only permitted for bits 0 to 11
Value range	24 bits, bit-oriented
Value after reset	0

### Meaning of the individual status register bits:

- Bit 0 : Receive register data from node # 2
- Bit 1 : Receive register data from node # 3
- Bit 2 : Receive register data from node # 4
- Bit 3 : Receive register data from node # 5
- Bit 4 : Receive register data from node # 6
- Bit 5 : Receive register data from node # 7
- Bit 6 : Receive register data from node # 8
- Bit 7 : Receive register data from node # 9
- Bit 8 : Receive register data from node # 10
- Bit 9 : Receive register data from node # 11
- Bit 10 : Receive register data from node # 12
- Bit 11 : Receive register data from node # 13
- Bit 12 : System bus error
- Bit 13 : BUSY
- Bit 14 : One internal receiving buffer overflow as a minimum
- Bit 15 : Parameter error  
This bit is set if the participant number (11m103), the baud rate (11m104), the first register to be sent (11m107) or the number of registers to be sent (11m106) are non-valid values.
- Bit 16 : The system bus alarm limit has been reached
- Bit 22 : The bus terminating resistor has been deactivated
- Bit 23 : The JX6-SB(-I) module is running in master-master mode

<b>Register 11m101: Command Registers</b>	
<b>Function</b>	<b>Description</b>
Read	Command issued last
Write	Command issued to the interface
Value range	0 - 255
Value after reset	0

### **Commands for Master-Master Operation:**

- 10** Initialize the JX6-SB(-I) module in master-master mode
- 11** Transmit register data
- 12** Update the contents of registers 11m2zz
- 13** Deactivate the bus terminating resistor
- 14** Activate the bus terminating resistor.  
After reset, the bus terminating resistor will be activated automatically.

### **Comments:**

- When register data are being transmitted, the contents of registers 11m2zz must not be changed, as long as the BUSY-bit is set in the status register.
- If command 11 "Send Register Data" is issued with invalid parameters contained in registers 11m106 and 11m107, bit 15 will be set in the status register and the function will be stopped.
- The same applies for command 12 "Updating the contents of register 11m2zz".
- If a system bus error occurs when command 11 is issued, the error message will be deleted first, then transmission will be started.
- A bus terminating resistor that can be switched off has not been available until hardware revision 02.



<b>Register 11m102: Internal Receiving Buffer Overflow</b>	
Function	Description
Read	Present overflow condition
Write	Illegal
Value range	0 - 4095
Value after reset	0

When new register data transmitted by another participant are received, the JX6-SB(-I) module sets the corresponding receiving bit in status register 11m100.

If the same participant transmits updated register data, before the receiving bit has been cleared, this will be signalled in register 11m102. At the same time, bit 15 in the status register will get value 1.

Clearing the receiving bit acknowledges the overflow bit in this register. As new register data have been received in the background, the receiving bit will immediately be set to 1.

**The Significance of the Individual Bits in Register 11m102:**

- Bit 0 : Receiving buffer overflow, node # 2
- Bit 1 : Receiving buffer overflow, node # 3
- Bit 2 : Receiving buffer overflow, node # 4
- Bit 3 : Receiving buffer overflow, node # 5
- Bit 4 : Receiving buffer overflow, node # 6
- Bit 5 : Receiving buffer overflow, node # 7
- Bit 6 : Receiving buffer overflow, node # 8
- Bit 7 : Receiving buffer overflow, node # 9
- Bit 8 : Receiving buffer overflow, node # 10
- Bit 9 : Receiving buffer overflow, node # 11
- Bit 10 : Receiving buffer overflow, node # 12
- Bit 11 : Receiving buffer overflow, node # 13

<b>Register 11m103: Node Number</b>	
Function	Description
Read	Set node number
Write	New node number
Value range	2 - 13
Value after reset	0

This register must be written into, before command 10 "Initialize the JX6-SB(-I) Module in Master-Master Mode" is transmitted. If, during initialization, an invalid value is contained in this register, the JX6-SB(-I) module will take on node number 2. In the status register, bit 15 "Parameter Error" will take on value 1.

<b>Register 11m104: Baud Rate</b>	
<b>Function</b>	<b>Description</b>
Read	Presently set baud rate
Write	New baud rate
Value range	4 - 7
Value after reset	7

This register must be written into, before command 10 "Initialize the JX6-SB(-I) Module in Master-Master Mode" is transmitted. Otherwise, the JX6-SB(-I) module will take the system bus interface into operation at a baud rate of 1 MBaud. If during initializing, an invalid value has been written into this register, the system bus interface will also be taken into operation at 1 MBaud. In the status register, bit 15 "Parameter Error" will take on value 1.

#### **Adjustable baud rates:**

<b>4</b>	125 kBaud
<b>5</b>	250 kBaud
<b>6</b>	500 kBaud
<b>7</b>	1 MBaud

<b>Register 11m105: Nodes for register 11m2zz</b>	
<b>Function</b>	<b>Description</b>
Read	Present node number
Write	New node number for command 12
Value range	2 - 13
Value after reset	0

This register contains the number of the node, the register data of which are to be mapped to the registers 11m200 through 11m263 when command 12 is issued.

<b>Register 11m106: Amount of Register Data</b>	
<b>Function</b>	<b>Description</b>
Read	Last specified amount
Write	New number of register data that are to be updated
Value range	1 – 64
Value after reset	0

When commands 11 and 12 are issued, the number of register data that are to be sent to other nodes, respectively, that are to be updated in registers 11m2zz, will be determined in this register.

<b>Register 11m107: Number of First Register Data</b>	
<b>Function</b>	<b>Description</b>
Read	Number of first register data specified at last
Write	New number of first register data
Value range	0 - 63
Value after reset	0

When commands 11 and 12 are issued, the first register data that are to be sent to other nodes, respectively, that are to be updated in registers 11m2zz, will be determined in this register.

<b>Register 11m108: Number of Last Register Data</b>	
<b>Function</b>	<b>Description</b>
Read	Number of last register data specified at last
Write	New number of the last register data
Value range	0 – 63
Value after reset	0

When commands 11 and 12 are issued, the last register data that are to be sent to other nodes, respectively, that are to be updated in registers 11m2zz, will be determined in this register.

<b>Register 11m109: Firmware Version Number</b>	
<b>Function</b>	<b>Description</b>
Read	Present firmware version
Write	Illegal
Value range	23-bit signed integer
Value after reset	Version number * 100

From this register the firmware version number of the JX6-SB(-I) module can be read out. The value that has been read equals the product of the version number times a hundred. Thus, value 210, for example, refers to version 2.10.

When the JX6-SB(-I) module is in the self test routine after switching on, the version number of the self test routine plus one thousand is displayed by this register.

$$\text{Register 11m109} = 1103$$

Version 1.03

Self test routine

If the JX6-SB(-I) module is in the loading routine during the operating system update, the version number of the loading routine plus two thousand is displayed by this register.

$$\text{Register 11m109} = 2103$$

Version 1.03

Loading routine

<b>Registers 11m110 through 11m133: First and Last Receiving Register</b>	
<b>Function</b>	<b>Description</b>
Read	After receiving new register data of a participant, the number of the first and last received register datum will be written here
Write	Illegal
Value range	0 - 63
Value after reset	0

<b>Detailed Assignment of the Receiving Registers</b>		
<b>Participant</b>	<b>First Receiving Register</b>	<b>Last Receiving Register</b>
2	11m110	11m111
3	11m112	11m113
4	11m114	11m115
5	11m116	11m117
6	11m118	11m119
7	11m120	11m121
8	11m122	11m123
9	11m124	11m125
10	11m126	11m127
11	11m128	11m129
12	11m130	11m131
13	11m132	11m133

<b>Registers 11m200 through 11m263: Register Data</b>	
<b>Function</b>	<b>Description</b>
Read	After issuing command 12 "Update Register Data", the register data of the respective participant will be written here
Write	Only permitted before issuing command 11 "Send Register Data"
Value range	32 bits / 24 bits
Value after reset	Undefined

## 10.2 Sample Programs

In the following example, the JX6-SB(-I) module will be taken into operation in the master-master coupling mode. In all examples, the following symbol file will be used:

### Symbol File

```

// --- Register ---
rFirst11      11m128
rStatus       11m100      rLast11      11m129
rCommand      11m101      rFirst12     11m130
rOverflow     11m102      rLast12     11m131
rNodeNo       11m103      rFirst13     11m132
rBaudrate     11m104      rLast13     11m133
rNode2zz      11m105
rAmountReg    11m106      // Bits in the status and overflow reg
rFirstReg     11m107      nbNode2     0
rLastReg      11m108      nbNode3     1
rFirst2       11m110      nbNode4     2
rLast2        11m111      nbNode5     3
rFirst3       11m112      nbNode6     4
rLast3        11m113      nbNode7     5
rFirst4       11m114      nbNode8     6
rLast4        11m115      nbNode9     7
rFirst5       11m116      nbNode10    8
rLast5        11m117      nbNode11    9
rFirst6       11m118      nbNode12    10
rLast6        11m119      nbNode13    11
rFirst7       11m120
rLast7        11m121      // --- Bits in status register ---
rFirst8       11m122      nbCanError  12
rLast8        11m123      nbBusy      13
rFirst9       11m124      nbOverflow  14
rLast9        11m125      nbParaError 15
rFirst10      11m126
rLast10       11m127

```

---

### Example 22: Initialization in Master-Master Mode

In the following example, the JX6-SB(-I) module will be taken into operation in the master-master mode. The module is given node number 2; the system bus is set to 1 MBaud.

#### Program File

```
TASK 0
    REGISTER_LOAD (rNodeNo, 2)      // Setting the node address
    REGISTER_LOAD (rBaudrate, 7)   // Baud rate 1 MBaud
    REGISTER_LOAD (rCommand, 10)   // Starting master-master mode

    WHEN                            // Waiting, until start is ready
        BIT_CLEAR (rStatus, nbBusy)

    THEN
        //Program continuation
```

### Example 23: Processing New Register Data Received from Node # 3

In the following example, the application program will react to receiving new register data from node # 3. Having received the register data, the program will copy them into registers 11m2zz. There, they can be processed customer-specifically. After this, the receiving bit will be reset, in order to enable the internal receiving buffer again.

#### Program File

```
WHEN                                     // Register data from node 3
    BIT_SET (rStatus, nbNode3)
THEN                                     // Display data in 11m2zz
    // Display register data received from node 3 in registers 11m2zz
    REGISTER_LOAD (rNode2zz, 3)         // Node number
    REGISTER_LOAD (rFirstReg, @rFirst3) // First register
    REGISTER_LOAD (rLastReg, @rLast3)   // Last register
    REGISTER_LOAD (rCommand, 12)       // Update 11m2zz
    //
WHEN                                     // Waiting, until data have been copied
    BIT_CLEAR (rStatus, nbBusy)
THEN
    BIT_CLEAR (rStatus, zbNode3)       // Acknowledge receipt
    // Processing the data, user-specifically
```



### Example 24: Process Updated Register Data Received from Node # 3

In this example, on receiving new register data the program checks whether there are more up-to-date register data. This can occur, when acknowledgement of data received will last too long because of clearing the respective bit in the status register. An overflow bit signals that the JX6-SB(-I) module has received new register data during this time. Clearing the reception bits in the status register will cause the former data to be deleted, while reception of the new data will immediately be displayed.

#### Program File

```

WHEN                                                                    // Register data from node 3
    BIT_SET (rStatus, nbNode3)
IF                                                                    // There are more up-to-date data
    BIT_SET (rOverflow, nbNode3)
THEN
    BIT_CLEAR (rStatus, nbNode3)                                        // Deleting the former data

WHEN                                                                    // The new data are ready
    BIT_SET (rStatus, nbNode3)
THEN                                                                    // Displaying data in 11m2zz
    // Register data from node 3 to be displayed in 11m2zz
    REGISTER_LOAD (rNode2zz, 3)                                        // Node number
    REGISTER_LOAD (rFirstReg, @rFirst3)                            // First register
    REGISTER_LOAD (rAmountReg, @rLast3)                            // Last register
    REGISTER_LOAD (rCommand, 12)                                    // Updating 11m2zz
    //
WHEN                                                                    // Waiting, until data have been copied
    BIT_CLEAR (rStatus, nbBusy)
THEN
    BIT_CLEAR (rStatus, nbNode3)                                    // Acknowledging receipt
    // Processing the data, user-specifically

```

### Example 25: Process Register Data Received from Node # 3 Considering an Overflow

Similar to example 11, it is first checked after receiving new register data, whether there are more up-to-date register data. Before the former data are deleted, though, they are first copied into registers 11m2xx and then processed. After this, the receiving bit will be cleared for confirmation; then the new data will be processed.

Basically, it is recommended to control the processing of received data via handshake bits in the register data. Otherwise, more data might be sent than the other nodes could take over for processing. In master-master operation, the system bus represents the fastest constituent; for this reason, it must adjust to the speed of the other constituents.

#### Program File

```

LABEL sReception
WHEN                                // Register data from node 3
    BIT_SET (rStatus, nbNode3)
THEN                                // Displaying data in 11m2zz
    // Register data from node 3 to be displayed in 11m2zz
    REGISTER_LOAD (rNode2zz, 3)     // Node number
    REGISTER_LOAD (rFirstReg, @First3) // First register
    REGISTER_LOAD (rAmountReg, @rLast3) // Last register
    REGISTER_LOAD (rCommand, 12)    // Updating 11m2zz

WHEN                                // Waiting, until data have been copied
    BIT_CLEAR (rStatus, nbBusy)
THEN
    // Processing the data, user-specifically

IF                                  // Are there further data to be processed?
    BIT_SET (rOverflow, nbNode3)
THEN
    BIT_CLEAR (rStatus, nbNode3)    // Acknowledging receipt
    GOTO sReception                // onto a new one
THEN
    //Program continuation

```

### Example 26: Transmit Register Data to the Other Nodes

In the following example, five new register data are transmitted to all the other nodes. The register data can be found in register 200 ff. and must therefore be written into registers 11m2zz first.

The user program is to control registers 11m2zz. The JX6-SB(-I) module will not transmit data to this register by itself.

#### Program File

```
    COPY (5, 200, 11m200)           // Data in the transfer register
    REGISTER_LOAD (rFirstReg, 0)     // First register data
    REGISTER_LOAD (rAmountReg, 5)    // Amount of Register Data
    REGISTER_LOAD (rCommand, 11)     // Sending Data
    WHEN                             // Waiting until data have been sent
        BIT_CLEAR (rStatus, nbBusy)
    THEN
        //Program continuation
```

## 10.3 Signal Waveform when Receiving Data

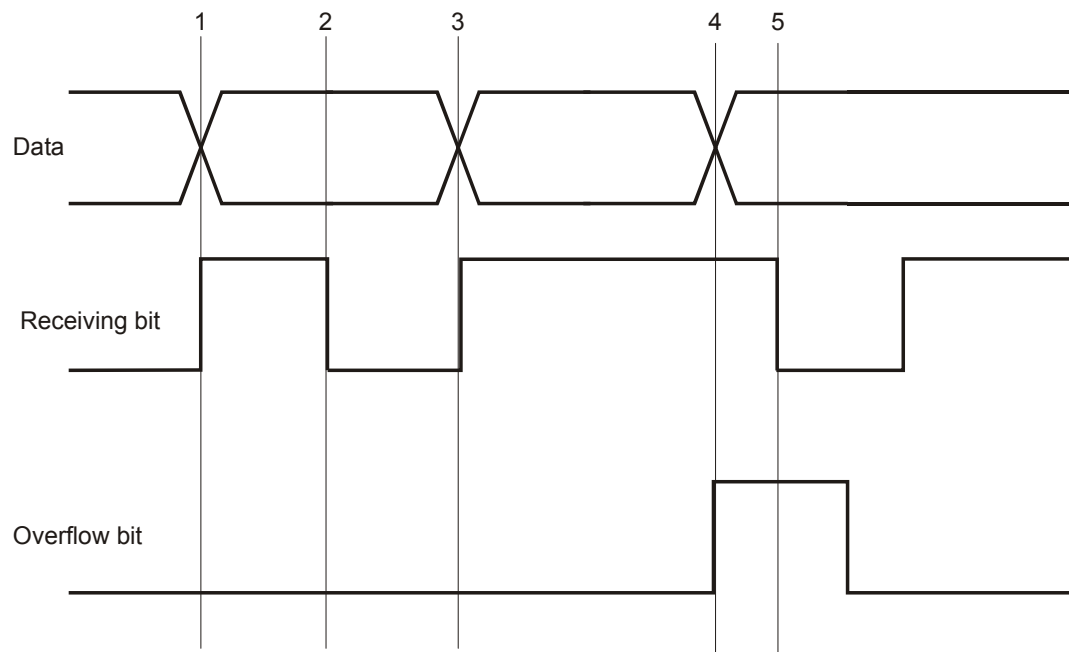


Fig. 19: Timing diagram of the receiving and overflow bit when receiving data

### 1: Receiving register data for the first time

Register data sent by another node have been received. In the status register the receiving bit takes on value 1.

### 2: Acknowledgement of the receiving bit by the user

The user has reacted to data reception and enabled the internal receiving buffer again by clearing the receiving bit.

### 3: Re-receiving register data

Register data sent by another node have been received. In the status register the receiving bit takes on value 1.

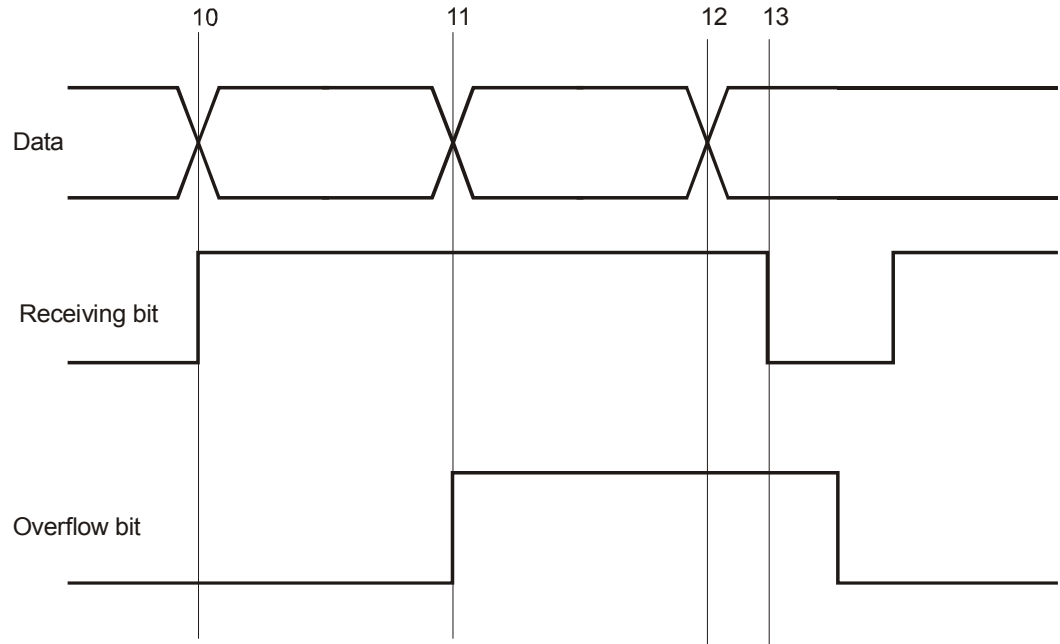
### 4: Re-receiving register data before acknowledging the receipt

Further register data of the same node have been received without setting the receiving bit in the status register to zero. Consequently, the overflow bit will take on value 1.

If, after this, registers 11m2zz are updated by the user issuing command 12, the register data received by node 3 will be copied.

### 5: Acknowledgement of the receiving bit by the user

The user will reset the receiving bit to zero, while the value of the overflow bit is 1. Immediately after this, the receiving bit will take on value 1 again, while the overflow bit will be set to zero. The data received last, in this example at 4, can be picked up.



**Fig. 20: Occurrence of a loss of data**

**10: Re-receiving register data**

Register data sent by another node have been received. In the status register the receiving bit takes on value 1.

**11: Re-receiving register data when the receiving bit is 1**

Further register data of the same node have been received a second time without setting the receiving bit in the status register to zero. Consequently, the overflow bit will take on value 1.

If, after this, registers 11m2zz are updated by the user issuing command 12, the register data received by node 10 will be copied.

**12: Re-receiving register data when the overflow bit is 1**

Register data are being received while the overflow bit is 1. The register data received in 11 will be overwritten. This cannot be ascertained in the user program.

If, after this, registers 11m2zz are updated by the user issuing command 12, the received register data in 10 will be copied, as the receiving bit has not been cleared yet.

**13: Acknowledgement of the receiving bit by the user**

The user will reset the receiving bit to zero, while the value of the overflow bit is 1. Immediately after this, the receiving bit will take on value 1 again, while the overflow bit will be set to zero. The data received last, in this example at 12, can be picked up.

# 10.4 Buffer Structure

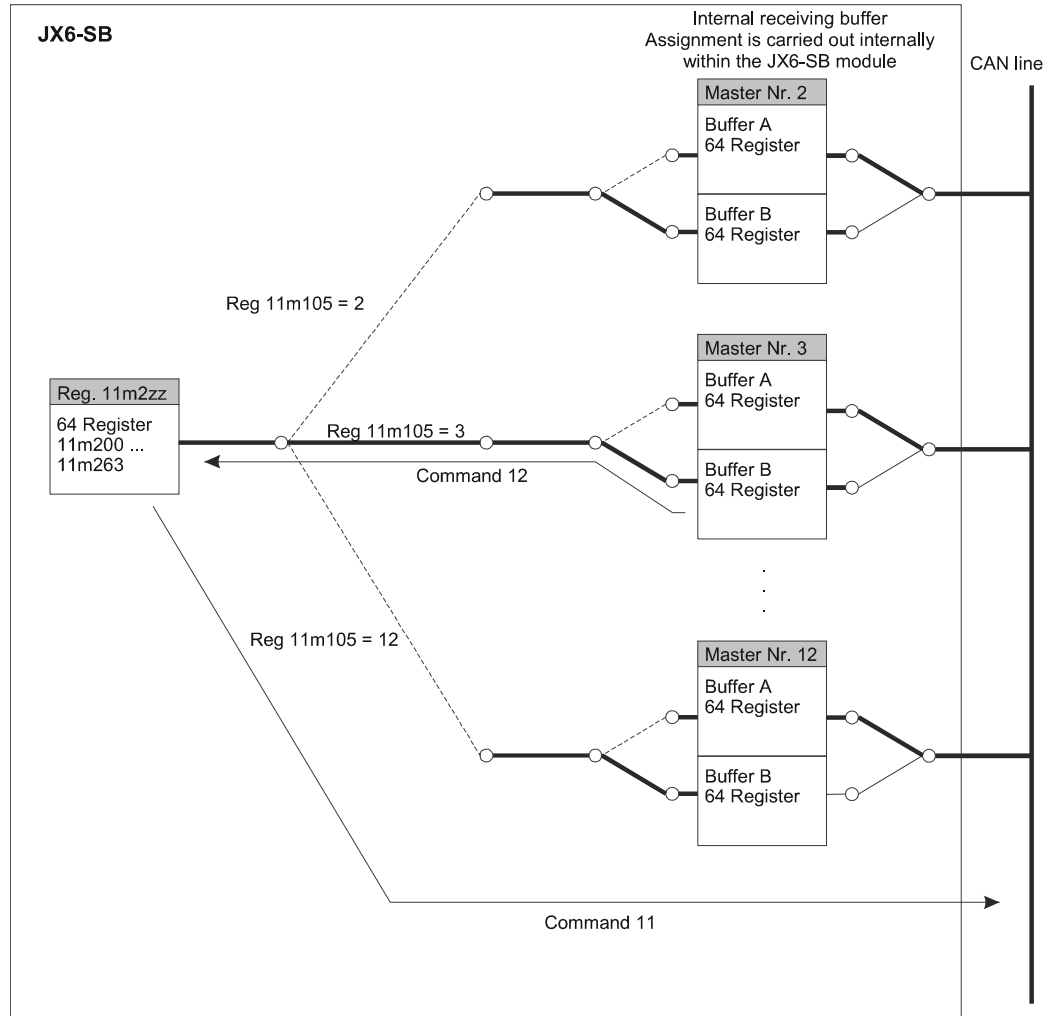


Fig. 21: Buffer Structure of a JX6-SB(-I) Master-Master Connection

## 10.5 Overview of Registers

Registers	Description	1) Value Range 2) Reset Value
11m100	Status Register	1) bit-oriented 2) 0
11m101	Command Register	1) 10 ... 14 2) 0
11m102	Internal receiving buffer overflow	1) 0 ... 4095 2) 0
11m103	Individual participant number	1) 2 ... 13 2) 0
11m104	Baud Rate	1) 4 ... 7 2) 7
11m105	Nodes for register 11m2zz	1) 2 ... 13 2) 0
11m106	Amount of Register Data	1) 1 ... 64 2) 0
11m107	Number of First Register Data	1) 0 ... 63 2) 0
11m109	Firmware Version Number	1) 0 ... 2999 2) Firmware version
11m110	First Receiving Register: Node # 2	1) 0 ... 63 2) 0
11m111	Last Receiving Register: Node # 2	1) 0 ... 63 2) 0
11m112	First Receiving Register: Node # 3	1) 0 ... 63 2) 0
11m113	Last Receiving Register: Node # 3	1) 0 ... 63 2) 0
11m114	First Receiving Register: Node # 4	1) 0 ... 63 2) 0
11m115	Last Receiving Register: Node # 4	1) 0 ... 63 2) 0
11m116	First Receiving Register: Node # 5	1) 0 ... 63 2) 0
11m117	Last Receiving Register: Node # 5	1) 0 ... 63 2) 0

<b>Registers</b>	<b>Description</b>	<b>1) Value Range 2) Reset Value</b>
11m118	First Receiving Register: Node # 6	1) 0 ... 63 2) 0
11m119	Last Receiving Register: Node # 6	1) 0 ... 63 2) 0
11m120	First Receiving Register: Node # 7	1) 0 ... 63 2) 0
11m121	Last Receiving Register: Node # 7	1) 0 ... 63 2) 0
11m122	First Receiving Register: Node # 8	1) 0 ... 63 2) 0
11m123	Last Receiving Register: Node # 8	1) 0 ... 63 2) 0
11m124	First Receiving Register: Node # 9	1) 0 ... 63 2) 0
11m125	Last Receiving Register: Node # 9	1) 0 ... 63 2) 0
11m126	First Receiving Register: Node # 10	1) 0 ... 63 2) 0
11m127	Last Receiving Register: Node # 10	1) 0 ... 63 2) 0
11m128	First Receiving Register: Node # 11	1) 0 ... 63 2) 0
11m129	Last Receiving Register: Node # 11	1) 0 ... 63 2) 0
11m130	First Receiving Register: Node # 12	1) 0 ... 63 2) 0
11m131	Last Receiving Register: Node # 12	1) 0 ... 63 2) 0
11m132	First Receiving Register: Node # 13	1) 0 ... 63 2) 0
11m133	Last Receiving Register: Node # 13	1) 0 ... 63 2) 0
11m2zz	Register array	1) 32-bit / 24-bit 2) 0



---

## A Explanation of Terms

<b>JX2-I/O Module</b>	Non-intelligent system bus expansion module, such as JX2-ID8, JX2-OD8, ...
<b>JX2-Slave Module</b>	Intelligent system bus expansion module, such as JX2-SM1D, JX2-SV1, ...
<b>I/O Module Number</b>	Number of a JX2-I/O module, a JX-SIO or another module connected to the system bus
<b>Slave Module Number</b>	Number of a JX2-Slave module connected to the system bus
<b>Smart I/O Module</b>	A complete expansion unit to be connected to the system bus, consisting of a JX-SIO module and terminals connected to it.
<b>24-bit Register Architecture</b>	Jetter AG PLCs the registers of which are 24 bits wide, e.g. Delta-CPU.
<b>32-bit Register Architecture</b>	Jetter AG PLCs the registers of which are 32 bits wide, e.g. JetControl JC647.

## B Index of Illustrations

Fig. 1:	Distributed Arrangement on the System Bus	15
Fig. 2:	Distributed Arrangement on the System Bus with JX-SIO Modules	15
Fig. 3:	Networking of Several CPUs in Master-Master Operating Mode	17
Fig. 4:	Master-Slave JX6-SB(-I) Configuration	19
Fig. 5:	Issuing Command 30 via Setup Window	20
Fig. 6:	Accessing the Module Array	21
Fig. 7:	Detecting a Timeout	22
Fig. 8:	Changing the Baud Rate	23
Fig. 9:	Master-Master Configuration	24
Fig. 10:	Configuring the lower JX6-SB(-I) Module	24
Fig. 11:	Configuring the upper JX6-SB(-I) Module	25
Fig. 12:	Sending Register Data	25
Fig. 13:	Detecting Received Register Data	26
Fig. 14:	Reading Register Data	27
Fig. 15:	Receipt Acknowledgement	28
Fig. 16:	Numbering: Sample Configuration	61
Fig. 17:	Configuring Dummy Modules	86
Fig. 18:	Access to Registers 3m07xzz	94
Fig. 19:	Timing diagram of the receiving and overflow bit when receiving data	118
Fig. 20:	Occurrence of a loss of data	119
Fig. 21:	Buffer Structure of a JX6-SB(-I) Master-Master Connection	120

## C Sample Programs

Example 1:	Assigning System Bus Module Numbers	16
Example 2:	Initializing the JX6-SB(-I) Module	23
Example 3:	Command Register of the JX6-SB(-I) Submodule	30
Example 4:	Version register of a JX2-I/O module	30
Example 5:	Actual Position Register of a JX2-Slave Module	31
Example 6:	Input and Output Instructions	31
Example 7:	System Bus Configuration with 5 JX2 Modules	34
Example 8:	System Bus Configuration with JX2 Modules and 1 Festo-CP-FB Module	34
Example 9:	Initialization in Master-Slave-D-CAN2 Mode:	36
Example 10:	Setting the Baud Rates in JX2-I/O Modules	38
Example 11:	Setting the baud rate of Festo-CP-FB modules	39
Example 12:	Initialization by the JX6-SB(-I) Module (sorting by check numbers):	45
Example 13:	Initializing according to the settings of the user program (logical order):	46
Example 14:	Command Register of the JX6-SB(-I) Submodule	57
Example 15:	Initialization in "Master-Slave JX6-SB" mode:	74
Example 16:	Adjusting the baud rate	76
Example 17:	Configuring Dummy Modules	86
Example 18:	Transferring a 32-bit value to a function terminal on a D-CPU	89
Example 19:	Writing command 32 into register 3207xzz	95
Example 20:	Recording Data and Visualizing them in JetSym	101
Example 21:	Positioning an axis	104
Example 22:	Initialization in Master-Master Mode	113
Example 23:	Processing New Register Data Received from Node # 3	114
Example 24:	Process Updated Register Data Received from Node # 3	115
Example 25:	Process Register Data Received from Node # 3 Considering an Overflow	116
Example 26:	Transmit Register Data to the Other Nodes	117

## D Calculating the I/O Sum

<b>I/O Sum of Smart I/O JX-SIO Modules</b>			
<b>System Bus Coupler</b>			
<b>Smart I/O Terminal</b>	<b>Amount</b>	<b>Factor</b>	<b>Sum</b>
JX-SIO		* 0	
<b>Digital Input Terminals</b>			
<b>Smart I/O Terminal</b>	<b>Amount</b>	<b>Factor</b>	<b>Sum</b>
IB IL 24 DI 2		* 2	
IB IL 24 DI 2-NPN		* 2	
IB IL 24 EDI 2		* 2	
IB IL 24 EDI 2-DESINA		* 4	
IB IL 24 DI 4		* 4	
IB IL 24 DI 8		* 8	
IB IL 24 DI 8/T2		* 8	
IB IL 24 DI 16		* 16	
IB IL 120 DI 1		* 2	
IB IL 230 DI 1		* 2	
<b>Digital Output Terminals</b>			
<b>Smart I/O Terminal</b>	<b>Amount</b>	<b>Factor</b>	<b>Sum</b>
IB IL 24 DO 2		* 2	
IB IL 24 DO 2-2A		* 2	
IB IL 24 DO 2-NPN		* 2	
IB IL 24 EDO 2		* 8	
IB IL 24 DO 4		* 4	
IB IL 24 DO 8		* 8	
IB IL 24 DO 8-2A		* 8	
IB IL 24 DO 16		* 16	
IB IL DO 1 AC		* 2	
IB IL DO 4 AC-1A		* 4	

<b>I/O Sum of Smart I/O JX-SIO Modules</b>			
<b>Analog Input Terminals</b>			
<b>Smart I/O Terminal</b>	<b>Amount</b>	<b>Factor</b>	<b>Sum</b>
IB IL AI 2/SF		* 4	
IB IL AI 8/SF		* 4	
IB IL AI 8/IS		* 4	
IB IL AI 2/4-20		* 4	
IB IL TEMP 2 RTD		* 4	
IB IL TEMP 2 UTH		* 4	
<b>Analog Output Terminals</b>			
<b>Smart I/O Terminal</b>	<b>Amount</b>	<b>Factor</b>	<b>Sum</b>
IB IL AO 1/SF		* 1	
IB IL AO 1/U/SF		* 1	
IB IL AO 2/U/BP		* 2	
<b>Relay Terminals</b>			
<b>Smart I/O Terminal</b>	<b>Amount</b>	<b>Factor</b>	<b>Sum</b>
IB IL 24/230 DOR 1/W		* 2	
IB IL 24/230 DOR 1/W-PC		* 2	
IB IL 24/230 DOR 4/W		* 4	
IB IL 24/230 DOR 4/W-PC		* 4	
IB IL DOR LV-SET		* 0	
<b>Power-Level Terminals</b>			
<b>Smart I/O Terminal</b>	<b>Amount</b>	<b>Factor</b>	<b>Sum</b>
IB IL 400 ELR 1-3 A		* 16	
IB IL 400 MLR 1-8 A		* 16	
IB IL 400 ELR R-3 A		* 16	
IB IL 24 TC		* 4	
<b>Third-Party Terminals</b>			
<b>Smart I/O Terminal</b>	<b>Amount</b>	<b>Factor</b>	<b>Sum</b>
INLINE CAM		* 32	

<b>I/O Sum of Smart I/O JX-SIO Modules</b>			
<b>Power and Segment Terminals</b>			
<b>Smart I/O Terminal</b>	<b>Amount</b>	<b>Factor</b>	<b>Sum</b>
IB IL 24 PWR IN		*0	
IB IL 24 PWR IN/F		*0	
IB IL 24 PWR IN/F-D		*2	
IB IL 24 PWR IN/2-F		*2	
IB IL 24 PWR IN/2-F-D		*2	
IB IL 24 PWR IN/R		*0	
IB IL 24 SEG		*0	
IB IL 24 SEG/F		*0	
IB IL 24 SEG/F-D		*2	
IB IL 24 SEG/ELF		*2	
IB IL 230 PWR IN		*0	
IB IL 120 PWR IN		*0	
<b>I/O Sum of Smart I/O Terminals</b>			

<b>I/O Sum for Third-Party Modules</b>			
<b>Module</b>	<b>Amount</b>	<b>Factor</b>	<b>Sum</b>
Bürkert Valve terminal type 8640		* 56	
Lenze Frequency converter 82XX vector with NANO-B / NANO-C / NANO-D with JC 24X / JX6-SB / JX6-SB-I		* 16	
		* 8	
SMC SI unit EX12# - SCA1		* 16	
<b>Festo CPV-Direct</b>			
<b>Module</b>	<b>Amount</b>	<b>Factor</b>	<b>Sum</b>
Valve Terminal CP...-GE-CO2		* 16	
Valve Terminal CPV...-GE-FB		* 16	
Valve Terminal CPA...-FB		* 16	
Output Module CP-A...-FB		* 16	
Input Module CP-E...-FB		* 16	
<b>Festo CPX Terminals</b>			
<b>Module</b>	<b>Amount</b>	<b>Factor</b>	<b>Sum</b>
Field Bus Node CPX-FB14		* 0	
Digital Input Module CPX-8DE		* 8	
Digital Input Module CPX-4DE		* 4	
Digital Output Module CPX-4DA		* 4	
Digital Input/Output Module CPX-8DE-8DA		* 16	
Analog Input Module CPX-2AE		* 4	
Analog Output Module CPX-2AA		* 4	
Pneumatic Interface CPX-GP-03-4.0 Depending on the DIP switch configuration		* 8 * 16 * 24 * 32	
Pneumatic Interface CPX-GP-CPA-... Depending on the DIP switch configuration		* 8 * 16 * 24	
<b>I/O Sum for Third-Party Modules</b>			

**Hint about Festo CPX Terminals**

The I/O sum for CPX modules CPX-4DE and CPX-4DA has to be rounded up to the next figure that can be divided by 8.

<b>I/O Sum for JX2 Modules / JC 24X, JX6-SB(-I)</b>			
<b>JX2-I/O Modules</b>			
<b>Module</b>	<b>Amount</b>	<b>Factor</b>	<b>Sum</b>
JX2-CNT1		* 16	
JX2-IA4		* 16	
JX2-ID8		* 16	
JX2-IO16		* 32	
JX2-OA2		* 16	
JX2-OA4		* 16	
JX2-OD2		* 16	
JX2-OD4		* 16	
JX2-OD8		* 16	
JX2-SER1		* 16	
JX2-PRN1		* 16	
<b>I/O Sum for JX2-I/O Modules</b>			
<b>LumbergJetter IP67 Compactbox Module</b>			
<b>Module</b>	<b>Amount</b>	<b>Factor</b>	<b>Sum</b>
LJX7-CSL-ID16		* 16	
LJX7-CSL-ID16-NPN		* 16	
LJX7-CSL-OD8-2A		* 16	
LJX7-CSL-OD16		* 16	
LJX7-CSL-ID8-OD8		* 16	
<b>I/O Sum for LJX7-Compactbox Modules</b>			
<b>JX2-Slave Modules</b>			
<b>Module</b>	<b>Amount</b>	<b>Factor</b>	<b>Sum</b>
JX2-SV1		* 1	
CAN-DIMA		* 1	
JX2-SM2		* 1	
JX2-SM1D		* 1	
JX2-PID		* 1	
JX2-PROFI1		* 1	
JetMove 200 Series		* 1	
JX2-ProfiM		* 1	
JetMove 600 Series		* 1	
<b>Amount of JX2-Slave Modules</b>			



<b>JX Modules</b>			
<b>Module</b>	<b>Amount</b>	<b>Factor</b>	<b>Sum</b>
JX-TP20-R (first JX-TP20-R module)		* 0	
JX-TP20-R (additional JX-TP20-R modules)		* 40	
<b>Amount of JX Modules</b>			

<b>System Bus I/O Sum</b>			
<b>Module Group</b>	<b>Amount</b>	<b>Factor</b>	<b>Sum</b>
I/O Sum for JX2-I/O Modules		* 1	
I/O Sum for LJX-Compactbox Modules		* 1	
I/O Sum of Smart I/O Terminals		* 1	
I/O Sum for LJX7-Compactbox Modules		* 1	
I/O Sum for Third-Party Modules		* 1	
JX6-SB, JX6-SB-I		* 0	
<b>System Bus I/O Sum</b>			
<b>Amount of JX2-Slave Modules</b>			

<b>Allowed I/O Sums</b>			
<b>Controller</b>	<b>Maximum Amount of JX2-I/O Modules</b>	<b>Maximum Amount of JX2-Slave Modules</b>	<b>Maximum I/O Sum</b>
NANO-A	5	0	54
NANO-B	15	3	136
NANO-C	15	3	136
NANO-D	23	4	200
JetControl 241	7	1	136
JetControl 243	15	3	264
JetControl 246	23	6	392
JX6-SB	31	0	496
JX6-SB-I	31	8	496



## **Jetter AG**

Gräterstrasse 2  
D-71642 Ludwigsburg

### **Germany**

Phone: +49 7141 2550-433  
Fax: +49 7141 2550-484  
Internet: <http://www.jetter.de>  
E-Mail: [sales@jetter.de](mailto:sales@jetter.de)

## **Jetter Subsidiaries**

### **Jetter Asia Pte. Ltd.**

32 Ang Mo Kio Industrial Park 2  
#07-03 Sing Industrial Complex  
Singapore 569510

#### **Singapore**

Phone: +65 6483 8200  
Fax: +65 6483 3881  
E-Mail: [sales@jetter.com.sg](mailto:sales@jetter.com.sg)

### **Jetter AG Schweiz**

Münchwilerstrasse 19  
CH-9554 Tägerschen

#### **Switzerland**

Phone: +41 719 1879-50  
Fax: +41 719 1879-69  
E-Mail: [info@jetterag.ch](mailto:info@jetterag.ch)

### **Jetter Automation Inc.**

165 Ken Mar Industrial Parkway  
Broadview Heights  
OH 44147-2950

#### **U.S.A**

Phone: +1 440 8380860  
Fax: +1 440 8380861  
E-Mail: [bernd@jetterus.com](mailto:bernd@jetterus.com)

## **Jetter Branches**

### **Jetter AG Büro Nord**

Am Nordbahnhof 5  
D-59555 Lipstadt

#### **Germany**

Phone: +49 2941 6691-0  
Fax: +49 2941 6691-22  
E-Mail: [dschnelle@jetter.de](mailto:dschnelle@jetter.de)

### **Jetter AG Büro Mitte**

Wohnbacher Strasse 19  
D-61200 Wölfersheim

#### **Germany**

Phone: +49 6036 9843-82  
Fax: +49 6036 9843-83  
E-Mail: [jpommerening@jetter.de](mailto:jpommerening@jetter.de)

### **Jetter AG Büro Ost**

Parkweg 15  
D-99610 Sömmerda

#### **Germany**

Phone: +49 36346 929678  
Fax: +49 36346 929679  
E-Mail: [mjakob@jetter.de](mailto:mjakob@jetter.de)

### **Jetter AG**

#### **Business Unit Ebelt**

Jahnstrasse 7  
D-73760 Ostfildern/Scharnhausen

#### **Germany**

Phone: +49 7185 64069  
Fax: +49 7185 658662  
E-Mail: [info@ebelt.de](mailto:info@ebelt.de)