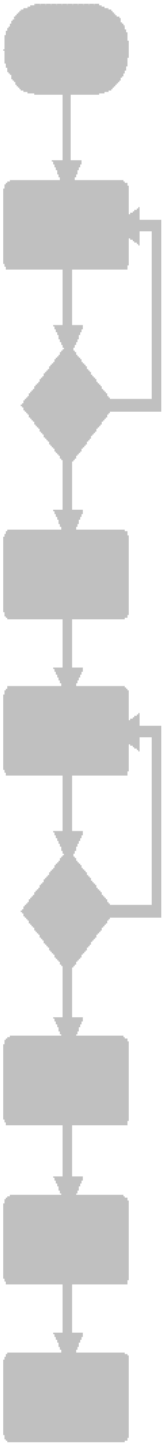


Nano-D
Technical Information
Version Update
from V1.01 to V2.00

Jetter

Edition 1.00

April 17, 2002 / Printed in Germany

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.

Contents

1 Introduction

Overview of Version Update			
Version	**Function**	**expanded**	**corrected**
V2.00	System bus	✓	✓
	Special registers	✓	
	DISPLAY instructions	✓	
	Special functions	✓	

By updating the operating version to V2.00, the NANO-D is provided with a number of new functions.

Important !

After having updated the operating system version to V2.00, it is mandatory to check register 2032 for switch-on delay. The user must write a value into this register in accordance with the respective application.
It is recommended to write value “60” in register 2032 for a start-up time of 6 seconds.

Important !

During the update to operating version V2.00, the power supply of the NANO-D must not be interrupted.

2 Expansions

2.1 DISPLAY Instructions for JX2-PRN1 and JX2-SER1 Modules

The instructions to display texts or registers on user interfaces were expanded by the output on JX2-PRN1 and JX2-SER1 modules.
Basically, the representation of texts or registers remains unchanged. However, device # 8 must be selected for the JX2-PRN1 module, and device # 11 for the JX2-SER1 module. JetSym is required to redirect the output of DISPLAY instructions to a JX2-SER1/JX2-PRN1 module.
Output on a JX2-SER1/JX2-PRN1 module is supported by the following instructions:

Supported Instructions	
JX2-PRN1	**JX2-SER1**
`DISPLAY_TEXT (8, 1, "Hello")`	`DISPLAY_TEXT (11, 1, "Hello")`
`DISPLAY_TEXT (8, 1, @1400)`	`DISPLAY_TEXT (11, 1, @1400)`
`DISPLAY_TEXT (8, 1, @@1400)`	`DISPLAY_TEXT (11, 1, @@1400)`
`DISPLAY_REG (8, 1, 1400)`	`DISPLAY_REG (11, 1, 1400)`
`DISPLAY_REG (8, 1, @1400)`	`DISPLAY_REG (11, 1, @1400)`

2.1.1 Overview of Registers

Register 2837: Module Number of JX2-PRN1	
Function	**Description**
Read	Module number of JX2-PRN1
Write	New module number
Value range	2 – 24
Value after reset	2

This register allows to select the JX2-PRN1 module for which the DISPLAY instructions are intended.

Register 2838: Module Number of JX2-SER1	
Function	**Description**
Read	Module number of JX2-SER1
Write	New module number
Value range	2 – 24
Value after reset	2

This register allows to select the JX2-SER1 module for which the DISPLAY instructions are intended.

2.1.2 Implementation

Output on a JX2-PRN1 Module

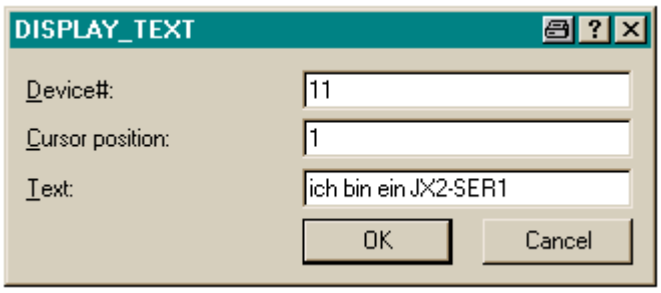
In the case of output to a JX2-PRN1 module, characters will be output as long as the connected printer is ready. If the printer is not ready, output will be interrupted and the task is switched.

Output on a JX2-SER1 Module

In the case of output to an JX2-SER1 module, the DISPLAY instructions first check the sending buffer of the JX2-SER1 module. If the buffer is full, the task is switched.
As soon as the sending buffer is empty, the text or register is written into the sending buffer one by one. If an error occurs during communication, the special flag 2048 “Timeout during IO access” is set and the instruction is aborted. The special flag must be reset by the user.

2.1.3 DISPLAY_TEXT on a JX2-SER1 Module

The DISPLAY_TEXT instruction will output the text on a JX2-SER1 module, if device # 11 is selected. The module number itself must have been specified in register 2838. Cursor position and control characters in the text are irrelevant. The DISPLAY_TEXT instruction for a JX2-SER1 module only works with direct, indirect and double-indirect register addressing.



Dialog window of the DISPLAY_TEXT instruction

Example 1: Output with DISLAY_TEXT on a JX2-SER1 Module

In the following example, the text „Hello" will be output on a JX2-SER1 module. The following characters are sent one after the other on the JX2-SER1:

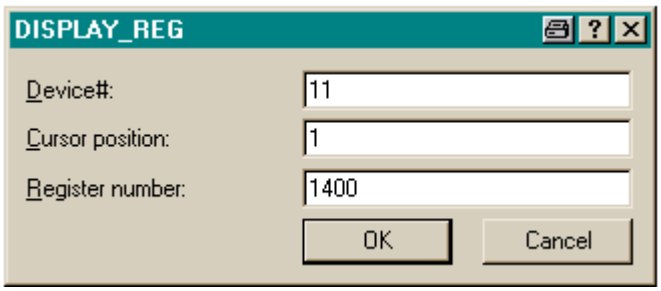
48_{hex}	ASCII-Code for „H"
65_{hex}	ASCII-Code for „e"
$6C_{hex}$	ASCII-Code for „l"
$6C_{hex}$	ASCII-Code for „l"
$6F_{hex}$	ASCII-Code for „o"

Program

```
REGISTER_LOAD (2838, 2)                 // Module # of JX2-SER1
DISPLAY_TEXT (11, 1, "Hello")           // Output text on JX2-SER1
```

2.1.4 DISPLAY_REG on a JX2-SER1 Module

The DISPLAY_REG instruction will output a register in ASCII format on a JX2-SER1 module, if device # 11 is selected. The module number itself must have been specified in register 2838. The cursor position is irrelevant. All formatting options for the output to a user interface still apply.
The DISPLAY_REG instruction for a JX2-SER1 module works with direct and indirect register addressing. Double indirect register addressing is not possible.



Dialog window of the DISPLAY_REG instruction

Example 2: Output with DISPLAY_REG on a JX2-SER1 Module

In the following example, value 7623 is loaded to register 1400. Next, the register is output on a JX2-SER1 module with a field width of 8 characters and without decimal positions. The following characters are sent one after the other on the JX2-SER1:

20_{hex}	Space characters
20_{hex}	Space characters
20_{hex}	Space characters
20_{hex}	Space characters
37_{hex}	ASCII-Code for „7“
36_{hex}	ASCII-Code for „6“
32_{hex}	ASCII-Code for „2“
33_{hex}	ASCII-Code for „3“

Program

```
REGISTER_LOAD (2810, 0)            // No decimal positions
REGISTER_LOAD (2812, 8)            // Set field width to 8
REGISTER_LOAD (2838, 2)            // Module number for JX2-SER1
REGISTER_LOAD (1400, 7623)         // Output register
DISPLAY_REG (11, 1, 1400)          // Display register on JX2-SER1
```

2.2 Switch-On Delay

Register 2032: Switch-On Delay	
Function	**Description**
Read	Current switch-on delay, resolution 100 ms Value after reset: latest set value
Write	New switch-on delay
Value range	0 – 600

The switch-on delay causes the NANO-D to wait after being switched on before the system bus is initialized and the application program is started. This delay time can be defined in register 2032 in increments of 100 ms. For example, a value of 60 in register 2032 means that the NANO-D will wait 6 s before initializing the system bus.
The delay time for the switch-on delay is useful when modules connected to the system bus require a long time from power-on until they are ready for operation. Smart IOs JX-SIO, for example, require a certain time after power-on until they can be initialized via the system bus. In this case, register 2032 should contain value 60.
As long as the delay time is active during switch-on delay, the RUN-LED will be flashing at increasingly short intervals.

Important !

After completed operating system update of NANO-D V1.01 or lower to V2.00, it is mandatory to check register 2032. The user must write a value in accordance with the application.

2.3 Size of the Application Program

Register 2035: Size of the Application Program in the RAM	
Function	**Description**
Read	Current size of the application program in bytes Value after reset: Size of the application program
Write	Illegal
Value range	0 – 65535

Register 2036: Size of the Application Program in the FLASH	
Function	**Description**
Read	Current size of the application program in bytes Value after reset: Size of the application program
Write	Illegal
Value range	0 – 65535

The memory size required by the application program in the NANO-D can be determined by reading registers 2035 and 2036. The max. available memory for the NANO-D is 65535 bytes.

2.4 Millisecond-Timer

Register 2037: Millisecond-Timer	
Function	**Description**
Read	Current value of the millisecond-timer Value after reset: 0
Write	Illegal
Value range	0 – 65535

The NANO-D increments the millisecond-timer by the value 1 each millisecond. The timer starts automatically when switching on the NANO-D. It is not possible to stop the timer.

2.5 System Bus

Apart from the non-intelligent and intelligent JX2-expansion modules, the system bus of the NANO-D allows to connect Smart I/Os JX.SIO and other modules, such as the Festo valve terminals CPV-Direct. For more detailed information, please refer to the relevant product descriptions.
Due to the max. permissible number of I/Os of the NANO-D, up to 10 Smart-IOs can be connected to the system bus. The table below lists the I/O numbers as well as the registers of the JX-SIOs. The parameter „y“ stands for the module number of the individual devices.
The other modules with module info 65 through 67 are basically treated like the Smart IOs JX-SIO.

2.5.1 Smart IOs JX-SIO

The Smart IOs are a new product line by Jetter AG. They offer a wide spectrum of digital and analog inputs and outputs.

Overview of Smart IO JX-SIO	
Function	**Description**
Digital inputs	IN 7y01 ... IN 7y64
Digital outputs	OUT 7y01 ... OUT 7y64
Analog inputs Configuration for analog inputs	Register 5y60 ... Register 5y71 Register 7y10 ... Register 7y21
Analog outputs Configuration for analog outputs	Register 6y60 ... Register 6y71 Register 7y30 ... Register 7y51
Error, status, serial number, etc.	Register 7y90 ... Register 7y99
Terminal array	Register 7y02 ... Register 7y03

2.5.2 Festo CPV-Direct Valve Terminal

Festo valve terminals of the type CPV..-GE-CO2-8 can directly be connected to the system bus of the NANO-D. Unlike the formerly used Festo CP-modules, these valve terminals do not require check number (PN number) handling.
It is not allowed to combine the valve terminals of the type CPV..-GE-CO2 and Festo CP-modules of the type CPV..-FB / CPA..-FB / CP-E..-FB / CP-A..-FB. These Festo CP-modules can be connected to the CP-expansion port of the CPV Direct.

Overview of Smart IO JX-SIO	
Function	**Description**
Digital inputs	IN 7y01 ... IN 7y16
Digital outputs	OUT 7y01 ... OUT 7y32
Error, status, serial number, etc.	Register 7y90 ... Register 7y99
Condition monitoring	Register 7y04 ... Register 7y25 Register 7y50 ... Register 7y65

2.5.3 Technical Data – System Bus

The max. configuration of the system bus depends on max. marginal data. On the one hand, the NANO-D only allows a certain quantity of inputs and outputs. Besides, configuration of a JX-SIO is limited and only a certain number of JX-SIOs can be connected.

System Bus Configuration	
Total number of I/Os **If the total number of I/Os is reached, all other specifications in this table are limited by the total number of I/Os.**	200
Max. number of non-intelligent JX2-modules	23
Max. number of intelligent JX2-modules	4
Max. number of JX-SIO and Festo CPV-Direct	10
Max. number of digital inputs per JX-SIO	64
Max. number of digital outputs per JX-SIO	64
Max. number of analog inputs per JX-SIO	12
Max. number of analog outputs per JX-SIO	12
Max. number of terminals per JX-SIO	63

2.5.4 Systembus Baud Rate

Baud rates for the system bus can be set between 125 kBaud und 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. Whether the system bus is operated with max. data transmission speed or with long line length must be decided individually depending on the application.
The permissible baud rates of the system bus also depend on the connected modules.

Permissible Baud Rates					
Non-intelligent JX2-modules **Intelligent JX2-modules**	**Smart-IOs JX-SIO**	**125 kBaud**	**250 kBaud**	**500 kBaud**	**1000 kBaud**
✓		✓	✓	✓	✓
	✓	✓	✓	✓	✓
✓	✓	✓			✓

Register 2029: Systembus Baud Rate	
Function	**Description**
Read	Current value of the set baud rate Value after reset = baud rate set last
Write	New value for baud rate A new baud rate only becomes active after switching the controller off and on again. 4 125 KBaud 5 250 KBaud 6 500 KBaud 7 1000 KBaud
Value range	4-7

2.5.5 System Bus Diagnosis

Register 2016: Module Array	
Function	**Description**
Read	Module info
	0 – 63 non-intelligent expansion modules
	64 JX-SIO
	65 Festo CPV Direct
	66 Festo Terminal CPX
	67 Buerkert valve terminal type 8640
	128 – 250 Intelligent expansion modules
	252 JX-SIO dummy-module
	253 Intelligent dummy-module
	254 Non-intelligent dummy-module
	255 not identified
Write	Illegal
Value range	0 to 255
Value after reset	Quantity of expansion modules

The modules Smart IO JX-SIO and other modules, such as Festo valve terminals, were added to the entries in the module array. The other modules with module info 65 through 67 are basically treated like the Smart IOs JX-SIO.

Register 2070: Quantity of Plugged JX-SIO Modules	
Function	**Description**
Read	Max. quantity of JX-SIO modules
Write	Illegal
Value range	0 ... 10

The number of plugged modules entered in register 2070 depends on the module with the highest module number. For instance, if two JX-SIOs with module numbers through 71 and 74 are connected, then value 5 is written to register 2070. The NANO-D regards the missing modules with the numbers 70, 72 and 73 as dummy-modules.

Register 2071: I/O Size of the System Bus	
Function	**Description**
Read	Current size of the system bus
Write	Illegal
Value range	0 – 200

The max. quantity of modules that can be connected to the Jetter system bus not only depends on the quantity of modules but also on their I/O size. Special tables can be used to calculate the I/O size. The current I/O size can be read from this register.

2.6 Special Functions 60 and 61

For serial communication between devices of automation technology, communcation according to the Modbus protocol has established itself among others.
Responsible for the Modbus protocol is MODICON Inc. who also publish it. This protocol does not depend on a certain transmission technique like RS485 or Ethernet.
There are two special functions to check the CRC-checksum according to MODBUS. Additional programming in the application program is not necessary.

2.6.1 Calculation Method

The CRC-checksum according to Modbus is calculated according to the algorithm published in the „Modbus Protocol“ of June 1996.

Modbus Protocol / Reference Guide

PI-MBUS-300 Rev. J
June 1996
MODICON, Inc., Industrial Automation Systems

2.6.2 Programming

Special functions 60 and 61 serve to check the CRC-checksum according to MODBUS. The special function can be called using both direct and indirect register transfer.

Supported Instructions	
Special Function 60	**Special Function 61**
`SPECIALFUNCTION (60, 100, 103)`	`SPECIALFUNCTION (61, 100, 103)`
`SPECIALFUNCTION (60, 100, @103)`	`SPECIALFUNCTION (61, 100, @103)`
`SPECIALFUNCTION (60, @100, 103)`	`SPECIALFUNCTION (61, @100, 103)`
`SPECIALFUNCTION (60, @100, @103)`	`SPECIALFUNCTION (61, @100, @103)`

2.6.3 Calculating the Modbus CRC-Checksum

Special Function 60: Calculating the Modbus CRC-Checksum	
Function	**Description**
Operating principle	This special function calculates a two-byte checksum from the transferred telegram and adds the two bytes to the end of the telegram.
Parameter 1	Number of the register with the first data of the MODBUS protocol.
Parameter 2	Number of the register with the last data of the MODBUS protocol without the two bytes for the CRC-checksum.
Potential errors	The number of the last register is smaller than the number of the first register. The number of the last register increased by two is higher than 2000, or higher than 49999. Each register may contain useful data in the lowest 8 bits only.
Result in case of error	undefined
Computing time	NANO-B: approx. 3520µs when the data length is 100 registers NANO-C: approx. 3230µs when the data length is 100 registers NANO-D: approx. 820µs when the data length is 100 registers

Example 3: Calculating the Modbus CRC-Checksum

In the following example, a CRC-checksum is calculated using special function 60. For this purpose, the register number with the first data and the register number with the last data must be transferred when calling the special function.
Each register contains one byte useful data. The special function adds the CRC-checksum to the last register, starting with the low-byte according to the Modbus specification. These consecutive registers can then be output via a serial interface.

Program

```
REGISTER_LOAD (100, 0x02)          // Slave Adress
REGISTER_LOAD (101, 0x03)          // Function Code
REGISTER_LOAD (102, 0x00)          // Start No.
REGISTER_LOAD (103, 0x20)          // Start No.
REGISTER_LOAD (104, 0x00)          // Quantity
REGISTER_LOAD (105, 0x04)          // Quantity
SPECIALFUNCTION (60, 100, 105)     // Calculate CRC
// CRC-checksum:
// Reg 106 = 0x45
// Reg 107 = 0xF0
```

2.6.4 Checking the Modbus CRC-Checksum

Special Function 61: Checking the Modbus CRC-Checksum	
Function	**Description**
Operating principle	This special function checks the checksum of the transferred telegram and adds the result to the end of the telegram.
Parameter 1	Number of the register with the first data of the MODBUS protocol.
Parameter 2	Number of the register with the last data of the MODBUS protocol including the two bytes for the CRC-checksum.
Potential errors	The number of the last register is smaller than the number of the first register. The number of the last register increased by two is higher than 2000, or higher than 49999. Each register may contain useful data in the lowest 8 bits only.
Result in case of error	undefined
Computing time	NANO-B: approx. 3560µs when the data length is 100 registers NANO-C: approx. 3310µs when the data length is 100 registers NANO-D: approx. 880µs when the data length is 100 registers

Example 4: Checking the Modbus CRC-Checksum

In the following example, a CRC-checksum is checked using special function 61. For this purpose, the register number with the first data and the register number with the last data including the checksum must be transferred when calling the special function.
Each register contains one byte useful data. The special function adds the result of the verification to the last register. If the checksum is correct, a 1 is returned. Else, a 0 is returned.

Program

```
REGISTER_LOAD (100, 0x02)          // Slave address
REGISTER_LOAD (101, 0x03)          // Function code
REGISTER_LOAD (102, 0x00)          // Start no.
REGISTER_LOAD (103, 0x20)          // Start no.
REGISTER_LOAD (104, 0x00)          // Quantity
REGISTER_LOAD (105, 0x04)          // Quantity
REGISTER_LOAD (106, 0x45)          // CRC-checksum
REGISTER_LOAD (107, 0xF0)          // CRC-checksum
SPECIALFUNCTION (61, 100, 107)     // CRC checksum
// the result can be read from
// register 108
IF
    REG 108 = 1
THEN
    // CRC-checksum correct
ELSE
    // CRC-checksum incorrect
```

3 Eliminated Software Bugs

3.1 System Bus

- If a timeout occurs during a register access to a non-intelligent JX2 expansion module, special flag 2048 will be set and the module number is written to register 2011.
- Now it is also possible to read inputs IN 8 through IN 15 of Festo CP-input modules that are directly connected to the system bus (not via a CPV-Direct).