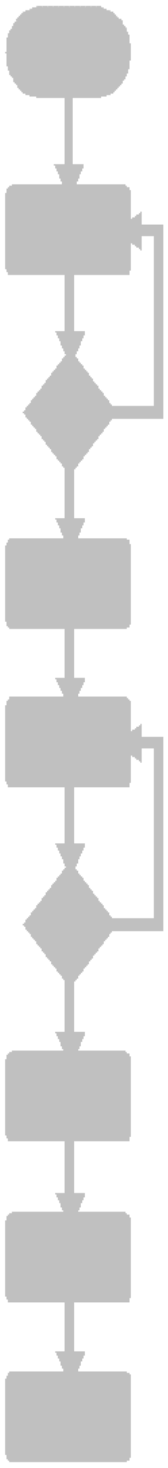


JX2-IO16

Version Update

from V. 1.04 to V. 2.00

Edition 1.03

December 2005 / Printed in Germany

Table of Contents

1 Introduction

Version Updates - Survey			
Version	**Function**	**upgraded**	**corrected**
V 2.00	Register interface	✓	
	Pulse stretching	✓	
	Counting function	✓	
	Diagnostics and adminstration registers	✓	

1.1 System Requirements

Software Versions of the Controllers and the JX6-SB(-I) Submodule	
Control	**Minimum Software Version**
JC-241, JC-243, JC-246	3.14
NANO-B, NANO-C, NANO-D	3.51
JX6-SB(-I) (for JC-647, DELTA, JC-800)	2.12

2 Expansions

2.1 Register Interface

As of software version 2.00, registers have been supplied to the user of JX2-IO16. By these registers, new module functions can be configured and new diagnose information can be read out.

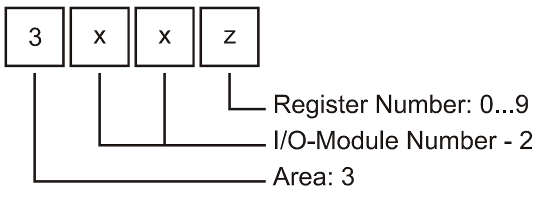
2.1.1 Register addressing

Coding of the register numbers for NANO-B/C/D and JC-24x

Each register number starts with the area number 3.

Coding of register numbers: 3xxz

Comment:



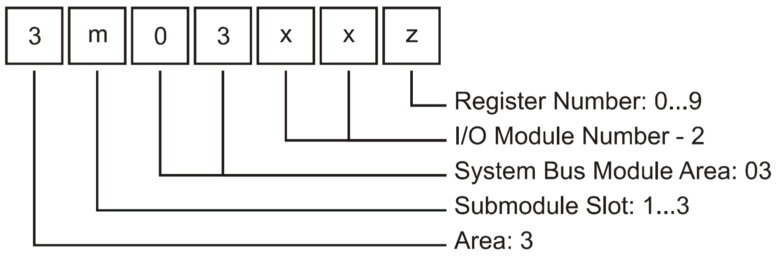
Coding of the register numbers for DELTA and JC-647

In the register numbers always consist of area number 3 and system bus module area 03.

Coding of register numbers: 3m0 3xxz

Comment:

Comment:

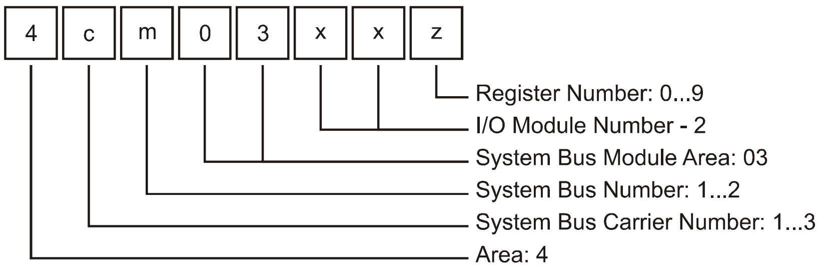


Coding of the register numbers for JC-800

In the register numbers always consist of area number 4 and system bus module area 03.

Coding of register numbers: 4cm0 3xxz

Comment:



The last four digits are the same for every address.

Note!

For determination of the I/O module number, only the non-intelligent modules will be counted. Intelligent modules, such as JM-2xx, JM-6xx, JX2-SV1, JX2-SM2, JX2-PID1, etc., located among the modules, are not being taken into consideration.

I/O module number 1 has been reserved for the control unit, respectively for the system bus module, of the JC-800. Starting from there, the I/O module numbers are counted from left to right. This means that the first I/O module after the controller, respectively after the system bus module, has got I/O module number 2.

The following applies to the addressing of the last four digits:

```
Register number = 3000 + (I/O-module number - 2) * 10 + local register
number
```

Example: Determination of the register numbers

Determining the register number of the third expansion module at a JetControl 246:

I/O module number = 4
Local register number = 9

Register number = 3000 + (4 - 2) * 10 + 9 = 3029

2.1.2 Register survey

For communication with the controller, 10 registers have been provided by the JX2-IO16 module. Further, there is a register array, which is addressed via register 8 and register 9. For this, register 8 provides the index of the array element, while register 9 contains the value of the element.

In the column "R/W", the possibility of access to the register, respectively to the array element, has been defined:

R	=	Read
W	=	Write

Registers

Register Number	Name	R/W	Description
3xx0	Status / Control	R/W	Unit: - Default value: 0b 00000000 00000001 00000000
3xx1	Manual Pulse Stretching - Set	R/W	Unit: - Default value: 0
3xx2	Manual Pulse Stretching - Reset	R/W	Unit: - Default value: 0
3xx7	Counter Value of Input 8	R/W	Unit: - Default value: 0
3xx8	Register Array: Index	R/W	Unit: - Default value: 1
3xx9	Register Array: Value	R/W	Unit: - Default value: Firmware Version

Register array

Index	Name	R/W	Description
1	Firmware Version	R	Unit: - Default value: Firmware Version
2	Errors	R/W	Unit: - Default value: 0
4	Communication - Time Base	R/W	Unit: ms Default value: 255 ms
5	Communication - Multiplier	R/W	Unit: - Default value: 0

Index	Name	R/W	Description
10	Error History - Entry 0	R	Unit: - Default value: Error Code
11	Error History - Entry 1	R	Unit: - Default value: Error Code
12	Error History - Entry 2	R	Unit: - Default value: Error Code
13	Error History - Entry 3	R	Unit: - Default value: Error Code
14	Error History - Entry 4	R	Unit: - Default value: Error Code
15	Error History - Entry 5	R	Unit: - Default value: Error Code
16	Error History - Entry 6	R	Unit: - Default value: Error Code
17	Error History - Entry 7	R	Unit: - Default value: Error Code
18	Error History - Entry 8	R	Unit: - Default value: Error Code
19	Error History - Entry 9	R	Unit: - Default value: Error Code
20	Counter Configuration	R/W	Unit: - Default value: 0
28	Counter Value of Input 8	R/W	Unit: - Default value: 0
29	Polarity	R/W	Unit: - Default value: 0b 11111111
30	Edge / Status	R/W	Unit: - Default value: 0
31	Impulse Stretching - Input 1	R/W	Unit: - Default value: 0
32	Impulse Stretching - Input 2	R/W	Unit: - Default value: 0
33	Impulse Stretching - Input 3	R/W	Unit: - Default value: 0
34	Impulse Stretching - Input 4	R/W	Unit: - Default value: 0
35	Impulse Stretching - Input 5	R/W	Unit: - Default value: 0
36	Impulse Stretching - Input 6	R/W	Unit: - Default value: 0

Index	Name	R/W	Description
37	Impulse Stretching - Input 7	R/W	Unit: - Default value: 0
38	Impulse Stretching - Input 8	R/W	Unit: - Default value: 0
50	Outputs - Error Mode	R/W	Unit: - Default value: 0
51	Outputs - Error Condition	R/W	Unit: - Default value: 0

2.1.3 Register description

Register 3xx0: Status / Control	
Function	**Description**
Read	Actual module status
Write	Setting a new mode for the module, only for bits 8 through 11
Value range	Bit-coded, 24 bits
Value after reset	0b 00000000 00000001 00000000

Only bits 8 through 11 can be written into for setting the module mode. Any other bits are status bits which cannot be changed.

Status

Meaning of the individual bits:

Bit 0: **Manual Pulse Stretching Input 1**

Specifies, if manual pulse stretching has been activated for the respective input.

0 = Off

1 = On

Value after reset: 0

Bit 1: **Manual Pulse Stretching Input 2**

Bit 2: **Manual Pulse Stretching Input 3**

Bit 3: **Manual Pulse Stretching Input 4**

Bit 4: **Manual Pulse Stretching Input 5**

Meaning of the individual bits:

Bit 5:	**Manual Pulse Stretching Input 6**
Bit 6:	**Manual Pulse Stretching Input 7**
Bit 7:	**Manual Pulse Stretching Input 8**
Bits 8-11:	**Control Bits (see below)**
Bit 12:	**Reserved**
Bit 13:	**Busy** Specifies, whether the module has carried out the required action. After writing into certain registers, for instance, the bit must be queried for zero before carrying out another action (e.g. writing into another register) on the module. At activating an action, please check from the respective description, whether a busy bit must be considered. 0 = Ready 1 = Busy Value after reset: 0
Bit 14:	**Reserved**
Bit 15:	**Errors** Here it is specified whether there is an error (see error evaluation in the register array for the error analysis). 0 = No error 1 = Errors Value after reset: 0
Bit 16:	**Pulse Stretching Input 1** Here, it is specified, whether the pulse stretching (manual or automatic) of the respective input is active. 0 = Not active 1 = Active Value after reset: 0
Bit 17:	**Pulse Stretching Input 2**
Bit 18:	**Pulse Stretching Input 3**
Bit 19:	**Pulse Stretching Input 4**
Bit 20:	**Pulse Stretching Input 5**

Meaning of the individual bits:

Bit 21:	**Pulse Stretching Input 6**
Bit 22:	**Pulse Stretching Input 7**
Bit 23:	**Pulse Stretching Input 8**

Control

Meaning of the individual bits:

Bit 8: **Reaction to a Communication Error (this only refers to setup)**

Setting the reaction to a communication error:

The Module is to Remain Addressable
IOs and registers can further be addressed after clearing the communication error.
This only makes sense, if the communication problem can be solved without deactivating the logic voltages in controller and module. Neither must an error condition which could be defined by means of register array elements 50 and 51 be written to the outputs.

The Module is to go to Stop Condition
This means the module must first be re-initialized, before I/Os and registers can be addressed again. In this case, an error condition which could be defined by means of register array elements 50 and 51 may be written to the outputs.

0 = The module remains addressable **(for setup only)**

1 = The module goes to stop condition

Value after reset: 1

Bit 9: **Reserved**

Bit 10: **Reserved**

Bit 11: **Reserved**

Register 3xx1: Manual Pulse Stretching - Set

Function	Description
Read	Inputs for manual pulse stretching activated last
Write	Activate the inputs for manual pulse stretching. Wait for resetting of bit 13 "Busy" in register 3xx0 "Status / Control".
Value range	Bit-coded, 8 bits
Value after reset	0

Via this register, the inputs for manual pulse stretching can only be activated. For the inputs to be activated, 1 must be set. For the inputs that are not considered, 0 must be set. Bits 0 through 7 of register 3xx0 "Status / Control" specify, which inputs are activated, respectively deactivated, for manual pulse stretching.

The inputs can be deactivated, respectively reset, via register 3xx2 "Manual Pulse Stretching - Reset".

If pulse stretching that has already been enabled for certain inputs (see bits 16 - 23 in register "Status / Control") is to be reset and re-enabled at the same time, the input concerned is set again via register 3xx1. This input does **not** need to be reset first via register 3xx2 "Manual Pulse Stretching - Reset", in order to be enabled again via this register.

Important!

After writing into this register, resetting of the busy bit must be waited for, before the inputs of the module have been updated by the controller.

Meaning of the values:

0	:	The input is not considered
1	:	The input is enabled

Meaning of the individual bits:

Bit 0:	Input 1
Bit 1:	Input 2
Bit 2:	Input 3
Bit 3:	Input 4
Bit 4:	Input 5
Bit 5:	Input 6
Bit 6:	Input 7
Bit 7:	Input 8

Register 3xx2: Manual Pulse Stretching - Reset	
Function	**Description**
Read	Inputs for manual pulse stretching deactivated, respectively reset, last
Write	Deactivate, respectively reset, the inputs for manual pulse stretching. Wait for resetting of bit 13 "Busy" in register 3xx0 "Status / Control".
Value range	Bit-coded, 8 bits
Value after reset	0

Via this register, the inputs for manual pulse stretching can only be disabled, respectively reset. For the inputs to be disabled, respectively reset, 1 must be set. For the inputs that are not considered, 0 must be set. Bits 0 through 7 of register 3xx0 "Status / Control" specify, which inputs are activated, respectively deactivated, for manual pulse stretching.

The inputs can be activated, respectively reset, via register 3xx2 "Manual Pulse Stretching - Set".

Difference between disabling and resetting

- At deactivating, the enabled input, for which manual pulse stretching **has not been enabled** yet (bits 0 - 7 in register 3xx0 "Status / Control" have been reset), is disabled, in order to prevent manual pulse stretching from becoming disabled.

- At resetting, the enabled input, for which manual pulse stretching is **already active** (bits 0 through 7 have been set in register 3xx0 "Status / Control"), is disabled; further, active pulse stretching (bits 16 through 23 in register 3xx0 "Status / Control") is reset. Resetting the inputs with simultaneously enabling them is carried out via register 3xx1 "Manual Pulse Stretching - Setting".

Important!

After writing into this register, resetting of the busy bit must be waited for, before the inputs of the module have been updated by the PLC.

Meaning of the values:

0	:	The input is not considered
1	:	The input is disabled / reset

Meaning of the individual bits:

Bit 0:	Input 1
Bit 1:	Input 2
Bit 2:	Input 3
Bit 3:	Input 4
Bit 4:	Input 5
Bit 5:	Input 6
Bit 6:	Input 7
Bit 7:	Input 8

Register 3xx7: Counter Value of Input 8

Function	Description
Read	Current result of count
Write	New result of count
Value range	- 8,388,608 ... + 8,388,607
Value after reset	0

The counter value is incremented at each input edge that has been sensed. An individual counter value can be set by writing into the register. Further values will then be incremented starting from this value.

There is an overflow. This means that at reaching value 8,388,607, the register value at the next edge will be -8,388,608.

In the register array element 29 "Polarity", a decision can be made, whether the value is to be incremented at a rising or a falling edge. After this, the input for "edge" must be defined in the register array element 30 "Edge / Status"; this means that the bit for the input must be set to zero.

Register 3xx8: Register Array: Index	
Function	**Description**
Read	Actual index
Write	Set index
Value range	1 ... 51
Value after reset	1

Register 3xx9: Register Array: Value	
Function	**Description**
Read	Actual value of the array element
Write	Set value of the array element
Value range	Dependent on the array element
Value after reset	Version number (index 1)

2.1.4 Description of the register array

Besides registers, much additional information of the module can be accessed via register array. For this, the index of the array element is entered into register 8 "Register Array: Index". In register 9 "Register Array: Value", the value of the element can be read, respectively written, after this.

Program File

```
...
REGISTER_LOAD (3008, 2)              // Index = 2 -> Information
                                     // on Errors
IF REG 3009 > 0 THEN                 // Evaluating the error
                                     // information via register
                                        9
     Troubleshooting                 //
     . . . .
THEN
```

Register 3xx8 = 1 Register 3xx9: Firmware Version	
Function	**Description**
Read	Actual firmware version
Write	Illegal
Value range	1 ... 99,999
Value after reset	Actual firmware version

Interpreting the value: 290 = Version 2.90

Important!

When submitting technical support queries the number of the firmware version must be quoted.

Register 3xx8 = 2 Register 3xx9: Errors	
Function	**Description**
Read	Actual error indication
Write	Resetting the errors
Value range	Bit-coded, 8 bits
Value after reset	0

If errors have occurred, the array element is reset to zero by writing any value to it. At the same time, bit 15 of register "Status / Control" is reset.

Meaning of the individual bits:

Bit 0:	**Reserved**
Bit 1:	**Output Circuit Error** Possible reasons for transmitting an error bit: • Overcurrent of at least one output (I > 0.5 A per output) • Undervoltage of the output supply • Overtemperature of the output circuit
Bit 2:	**Communications Error** The system bus communication had been interrupted for at least the duration of the set timeout. The time base is set via array element 4 "Communication - Time Base" and 5 "Communication - Multiplier". ATTENTION: If one of the array elements for setting the timeout equals zero, the error will not be generated.
Bit 3:	**Reserved**
Bit 4:	**Reserved**
Bit 5:	**Reserved**
Bit 6:	**Reserved**
Bit 7:	**Reserved**

Register 3xx8 = 4
Register 3xx9: Communication - Time Base

Function	Description
Read	Actual time base
Write	New time base
Value range	0 ... 255 ms
Value after reset	255 ms

Together with array element 5, this array element sets the timeout for a communication error. The timeout is calculated by the following formula:

Timeout = Time Base (Array Element 4) * Multiplier (Array Element 5)

Important!

The timeout must not be smaller than 200 ms.

If one of the array elements 4, respectively 5, has got value zero, a communication error will not be generated.

Register 3xx8 = 5 Register 3xx9: Communication - Multiplier	
Function	**Description**
Read	Actual multiplier
Write	New multiplier
Value range	0 ... 255
Value after reset	0

Together with array element 4, this array element sets the timeout for a communication error. The timeout is calculated by the following formula:

Timeout = Time Base (Array Element 4) * Multiplier (Array Element 5)

Important!

The timeout must not be smaller than 200 ms.

If one of the array elements 4, respectively 5, has got value zero, a communications error will not be generated.

Register 3xx8 = 10 ... 19 Register 3xx9: Error History - Entry 0 ... 9	
Function	**Description**
Read	Entries of the latest 10 errors
Write	Illegal

Value range	Bit-coded, 8 bits
Value after reset	Error entry (0 = an error has not been entered yet)

In the error history, the 10 errors that occurred last are saved in remanent mode; this means the entries are kept even if the module is deactivated. If, during the entire module life, 10 errors have not occurred yet, entries of value 0 will be made in the history; this means an error has not been recorded yet.

Meaning of the individual bits:

Bit 0: **Reserved**

Bit 1: **Output Circuit Error**

Possible reasons for transmitting an error bit:
- Overcurrent of at least one output (I > 0.5 A per output)
- Undervoltage of the output supply
- Overtemperature of the output circuit

Bit 2: **Communications Error**

The system bus communication had been interrupted for at least the duration of the set timeout. The time base is set via array element 4 "Communication - Time Base" and 5 "Communication - Multiplier".

ATTENTION:

If one of the array elements for setting the timeout equals zero, the error will not be displayed.

Bit 3: **Reserved**

Bit 4: **Reserved**

Bit 5: **Reserved**

Bit 6: **Reserved**

Bit 7: **Reserved**

Register 3xx8 = 20
Register 3xx9: Counter Configuration

Function	Description
Read	Actual counter configuration
Write	New counter configuration
Value range	Bit-coded, 8 bits
Value after reset	0

Here, input 8 is set either to function as counter or not:

Bit 8 = 0:	Input 8 is a conventional input
Bit 8 = 1:	Input 8 is a counter

Register 3xx8 = 28
Register 3xx9: Counter Value of Input 8

Function	Description
Read	Current result of count
Write	New result of count
Value range	- 8,388,608 ... + 8,388,607
Value after reset	0

See description of register 3xx7 "Result of Count - Input 8".

Register 3xx8 = 29
Register 3xx9: Polarity

Function	Description
Read	Actual polarity
Write	New polarity
Value range	Bit-coded, 8 bits
Value after reset	0b 11111111

The polarity is always set together with array element 30 "Edge / Status". Via these array elements, the conditions for the pulse-stretching event or for the counter event of an input are specified.

Meaning of the values:

0	:	Array element 30 = edge: Event at a falling edge of the input signal Array element 30 = status: Event at input signal = 0 V
1	:	Array element 30 = edge: Event at rising edge of the input signal Array element 30 = status: Event at input signal = 24 V

Meaning of the individual bits:

Bit 0:	Input 1
Bit 1:	Input 2

Meaning of the individual bits:

Bit 2:	Input 3
Bit 3:	Input 4
Bit 4:	Input 5
Bit 5:	Input 6
Bit 6:	Input 7
Bit 7:	Input 8

Register 3xx8 = 30
Register 3xx9: Edge / Status

Function	Description
Read	Present value
Write	New value
Value range	Bit-coded, 8 bits
Value after reset	0

The value is always set together with array element 29 "Polarity". Via these array elements, the conditions for the pulse-stretching event or for the counter event of an input are specified.

Meaning of the values:

0	:	Event at an edge The edge is evaluated. Via array element 29, a decision is made between rising and falling edge.
1	:	Event at the status (level) Only the voltage level (0 V or 24 V) without evaluation of the edge is considered. Via array element 29, a decision is made between a 0 V and a 24 V level.

Meaning of the individual bits:

Bit 0:	Input 1
Bit 1:	Input 2
Bit 2:	Input 3
Bit 3:	Input 4
Bit 4:	Input 5

Meaning of the individual bits:

Bit 5:	Input 6
Bit 6:	Input 7
Bit 7:	Input 8

Register 3xx8 = 31 ... 38
Register 3xx9: Pulse Stretching Time
Input 1 ... 8

Function	Description
Read	Actual pulse stretching time
Write	New pulse stretching time
Value range	0 ... 255 ms
Value after reset	0

Via the array elements, the pulse stretching times for automatic pulse stretching are specified.

Note!

At entering 0 ms, automatic pulse stretching is deactivated.

Register 3xx8 = 50
Register 3xx9: Outputs - Error Mode

Function	Description
Read	Actual error mode
Write	New error mode
Value range	Bit-coded, 8 bits
Value after reset	0

Via the error mode, a decision is made for each output individually how to behave in case of a communication error:

0	:	Keep the actual status
1	:	Set the error condition

Meaning of the individual bits:

Bit 0:	Output 1
Bit 1:	Output 2
Bit 2:	Output 3
Bit 3:	Output 4
Bit 4:	Output 5
Bit 5:	Output 6
Bit 6:	Output 7
Bit 7:	Output 8

Register 3xx8 = 51
Register 3xx9: Outputs - Error Condition

Function	Description
Read	Actual error condition
Write	New error condition
Value range	Bit-coded, 8 bits
Value after reset	0

Via the error condition a definition is made for each output individually, how it is to be set in a communication error condition, in case the error mode for the output has been set to 1 "Set the Error Condition":

0	:	Reset output
1	:	Set output

Meaning of the individual bits:

Bit 0:	Output 1
Bit 1:	Output 2
Bit 2:	Output 3
Bit 3:	Output 4
Bit 4:	Output 5
Bit 5:	Output 6
Bit 6:	Output 7
Bit 7:	Output 8

2.2 Pulse Stretching

2.2.1 General function

At pulse stretching, the time of the logic input status, which is read out of the module by the controller, is expanded. This means that even if the input signal is not active any more, the logic input status of the module still indicates the input signal via the input/output number in the application program. This way, even very short input pulses can definitely be recognized in the application program.

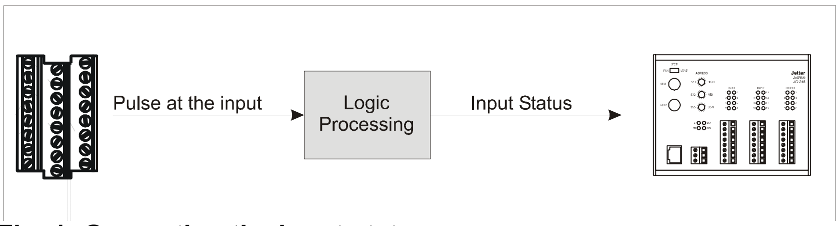


Fig. 1: Generating the input status

Via array elements 29 and 30 the conditions of pulse stretching are set for each input individually. This means that the input signal can be extended at, for example, either a certain rising edge, or a falling edge.

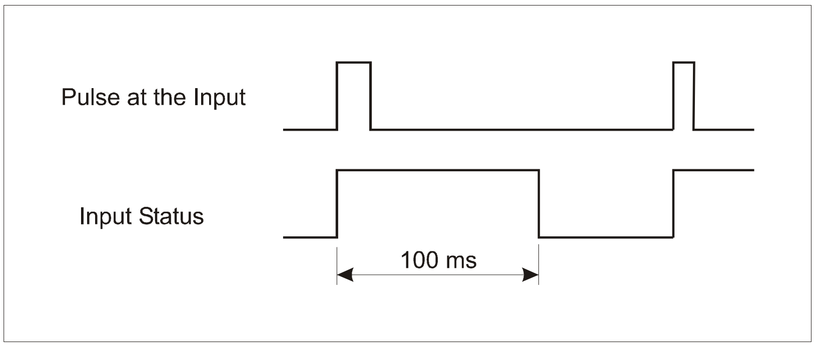


Fig. 2: The effects of pulse stretching of 100 ms at a rising edge

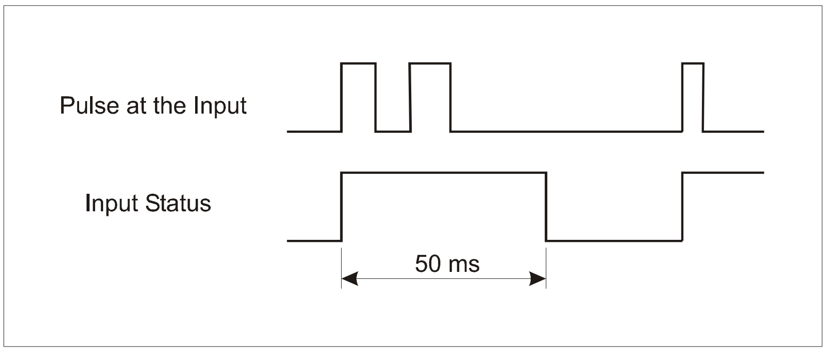


Fig. 3: The effects of pulse stretching at two short pulses

While pulse stretching is active, further edge modifications of the input signal are being ignored. For this reason, pulse stretching can, for example, be used for debouncing an input signal.

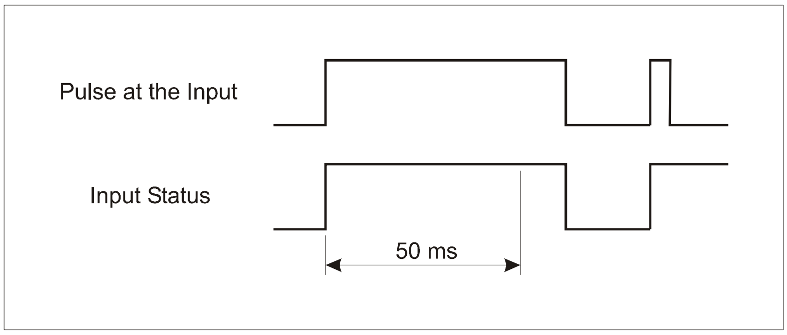


Fig. 4: The effects of pulse stretching at one long pulse

If the input pulse has been activated longer than pulse stretching is active, this pulse stretching will be of no effect. In this case, the length of the input pulse in input condition is determined by the length of the actual pulse.

When pulse stretching has been disabled, the actual input status will always be displayed again.

The module provides two ways of pulse stretching:

- Manual pulse stretching
- Automatic pulse stretching

Manual pulse stretching

At manual pulse stretching, the input pulse is being extended, until the application program resets pulse stretching.

Automatic pulse stretching

At automatic pulse stretching, the input pulse is being extended, until the set pulse stretching time has expired.

2.2.2 Manual pulse stretching

The following registers, respectively array elements, are available:

- Register 3xx0 "Status / Control"
- Register 3xx1 "Manual Pulse Stretching - Set"
- Register 3xx2 "Manual Pulse Stretching - Reset"
- Array element 29 "Polarity"
- Array element 30 "Edge / Status"

Via array elements 29 and 30 the conditions of pulse stretching are set for each input individually.
Then, the inputs are enabled separately via register 3xx1 for manual pulse stretching. Enabling is displayed in register 3xx0 by the respective bits. At each writing into register 3xx1, resetting of the busy bit in register 3xx0 must be waited for, before the inputs of the module have been updated by the controller.

If the pulse stretching conditions for an input have been met, the module will "freeze" the status of this input, until pulse stretching has been reset in the application program.

Whether pulse stretching is enabled can be read via register 3xx0, bits 16 through 23, for each input individually.

Pulse stretching can be reset in two ways:

- Writing into register 3xx1 a second time, if the input is to be re-enabled for manual pulse stretching immediately.
- Writing into register 3xx2, if the input is not to be used for manual pulse stretching any more at that instant.

Important!

After writing into register 3xx1 respectively 3xx2, resetting of the busy bit in register 3xx0 must be waited for, before the inputs of the module have been updated by the controller.

If manual pulse stretching is applied, automatic pulse stretching must be deactivated.

Example

At the first module following the controller, the input signal of input 1 is to be manually stretched at rising edge.

Program File

```
...
REGISTER_LOAD (3008, 30)              // Select array element 30 -
                                      // > edge / status
                                      //
BIT_CLEAR (3009, 0)                   // Select the edge
REGISTER_LOAD (3008, 29)              // Select array element 29 -
                                      // > polarity
BIT_SET (3009, 0)                     // Rising edge
REGISTER_LOAD (3001, 1)               // Activate the input for
                                      // manual stretching
                                      //
WHEN BIT_CLEAR (3000, 13) THEN        // Wait for busy bit to be
                                         cleared
WHEN IN 201 THEN                      // Wait for the first rising
                                      // edge to be recognized
REGISTER_LOAD (3001, 1)               // Reset and re-enable pulse
                                      // stretching immediately
                                      //
WHEN BIT_CLEAR (3000, 13) THEN        // Wait for busy bit to be
                                         cleared
...
```

2.2.3 Automatic pulse stretching

The following registers, respectively array elements, are available:

- Register 3xx0 "Status / Controller"
- Array element 29 "Polarity"
- Array element 30 "Edge / Status"
- Array elements 31 through 38 "Pulse Stretching Time"

Via array elements 29 and 30 the conditions of pulse stretching are set for each input individually.

Via array elements 31 - 38, the pulse stretching time (255 ms max.) is set for each input individually, respectively, pulse stretching is deactivated.

If the pulse stretching conditions for an input are active, the module will "freeze" the status of this input, until the pulse stretching time for this input has expired. After this, pulse stretching is reset by the module itself automatically. .

Whether pulse stretching is enabled can be read via register 3xx0, bits 16 through 23, for each input individually.

After each resetting and repeating of the condition, pulse stretching will be repeated, until it is disabled.
Pulse stretching is disabled, when 0 is defined as pulse stretching time.

Important!

If automatic pulse stretching is applied, manual pulse stretching must be disabled.

Example

At the first module following the controller, the input signal of input 1 is to be automatically stretched to 255 ms at rising edge.

Program file

```
...
REGISTER_LOAD (3008, 30)              // Select array element 30 -
                                      // > edge / status
                                      //
BIT_CLEAR (3009, 0)                   // Select the edge
REGISTER_LOAD (3008, 29)              // Select array element 29 -
                                      // > polarity
BIT_SET (3009, 0)                     // Rising edge
REGISTER_LOAD (3008, 31)              // Select array element 31 -
                                         > Stretching time
REGISTER_LOAD (3009, 255)             // Stretching time of 255 ms

LABEL lInput_1_Check                  // Beginning of the loop
WHEN IN 201 THEN                      // Wait for the first rising
                                         edge to be recognized
WHEN -IN 201 THEN                     // Wait, until pulse
                                      // stretching has been reset
GOTO lInput_1_Check                   // End of loop
...
```

2.3 Counting Function

Input 8 can also be used as a counter input. Whether the input is to be used as a conventional input or as a counter must be specified via register array element 20 "Counter Configuration".

Important!

If input 8 is to be used as a counter, this must be configured in register array element 20 "Counter Configuration" explicitely. There, bit 7 must be set. Yet, bit 7 cannot be used for activating, respectively deactivating counters. Deactivating, respectively re-activating, if it is needed, is carried out by manual impulse stretching.

If input 8 is used as a counter, updating the input status is dependent on the I/O monitoring interval of the controller, which can be set at will.

The condition, at which signal waveform the rising edge, for example, is to be counted, is set via register array elements 29 and 30.

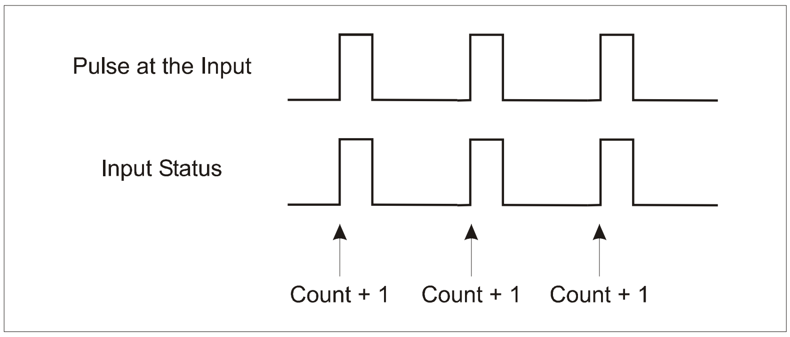


Fig. 5: Counting at rising edge

Note!

Pulse-stretching is also involved in the counting function.

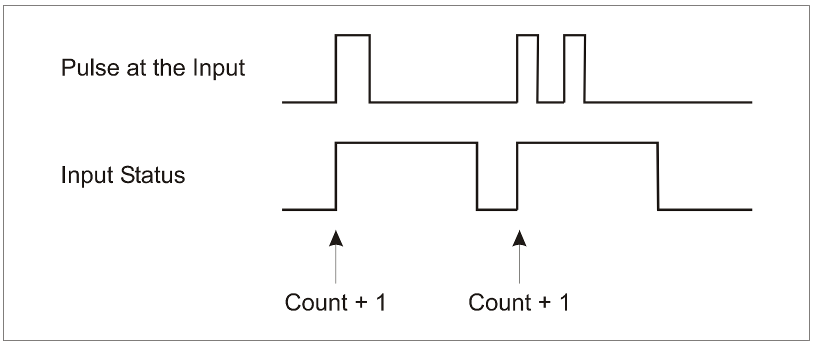


Fig. 6: Counting with pulse stretching

Via register3xx7 "Count of Input 8", the actual count can be read out. The count can also be manipulated via this register.

Example

Via input 8 of the first module following the controller, the pulses are to be counted. At a certain number of pulses, a stop signal is to be transmitted via output 1.

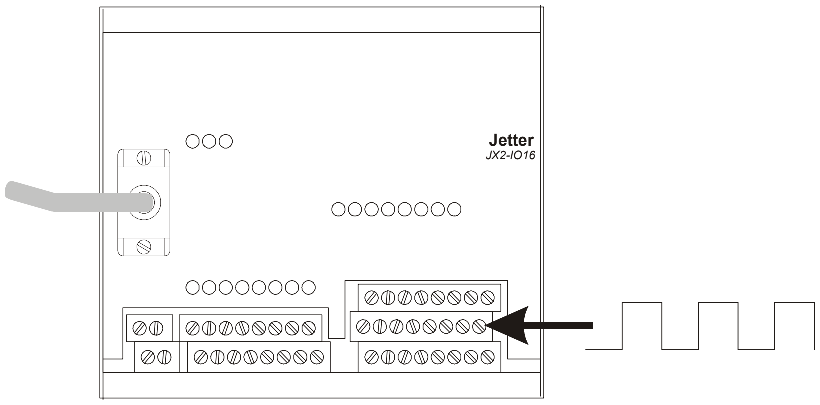


Fig. 7: View of the module regarding counting pulses

Program file

```
...
REGISTER_LOAD (3008, 20)            // Select array element 20 -
                                    // > counter configuration
                                    //
BIT_SET (3009, 7)                   // Configure input 8 as a
                                    // counter
REG_ZERO 3007                       // Setting the count to zero
WHEN REG 3007 > 94 THEN             // Have 95 or more pulses
                                    // been counted?
OUT 201                             // Setting output 1 on the
                                    // module
...
```

2.4 Diagnostics and Adminstration Registers

2.4.1 Error diagnosis

The module can have the following errors displayed:

- Communication error
- Output circuit error

An error is displayed as follows:

- The ERR LED is lit red.
- In register 3xx0 "Status / Controller", bit 15 "Error" is set.
- In array element "Error", the respective error bit is set.
- The content of the array element "Error" is entered into the remanent error history.

An error can be acknowledged as follows:

- Write any value into the array element "Error".
- Switch the module off and on again.
- Re-initialization of the system bus via the controller.

Communication error

A communication error occurs, if the module has not received monitoring telegrams any more for a time that can be set at will.

The set timeout for a communication error can be set via the two array elements 4 "Communication - Time Base" and element 5 "Communication - Multiplier".

Important!

The communication error is deactivated by default! If one of the array elements 4, respectively 5, has got value zero, a communication error will not be displayed.

Possible reasons of the communication error:

- The controller has failed
- Cable breakage or problems with contacts at system bus cables

Important!

When a communications error has occurred, the module is deactivated; this means it cannot be addressed any more, unless it is re-initialized.

Output circuit error

The error occurs, if the output circuit reports an error for at least 30 ms.

Possible reasons of the output circuit error:

- Overcurrent of at least one output (I > 0.5 A per output)
- Undervoltage of the output supply
- Overtemperature of the output circuit

2.4.2 Error behaviour of the digital outputs

In case of a communications error, each output can be set to an error condition to be defined at will.

The output can be preset to either being set or reset or to keep its present status.

The error condition is defined via array elements 50 "Outputs - Error Mode" and 51 "Outputs - Error Status".

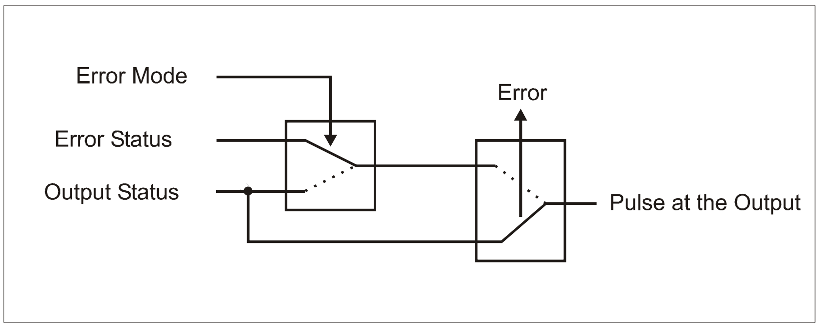


Fig. 8: Pattern of defiing output error conditions

Re-initialization after communications error

When a communications error has occurred, the module is deactivated; this means it cannot be addressed any more, unless it is re-initialized.

Important!

The controller stores the last output status of the module before the communications error having occurred. If the output status has changed due to setting an error condition on the module, the output status stored by the controller is not valid any more.

The module can be initialized in two ways:

- Initialisation 1:
  The module is re-initialized via the controller by means of a command, respectively a special function.
- Initialisation 2:
  The module is re-initialized by deactivating and re-activating the controller and the module itself.

The output status is influenced by the ways of initialization in different ways.

- Initialisation 1:
  The error condition of the outputs that has been set at a communications error, is kept at the module. The output status stored by the controller is updated at re-initialization.
- Initialisation 2:
  The output status of the module and of the controller is reset to zero (also refer to Fig. 9).

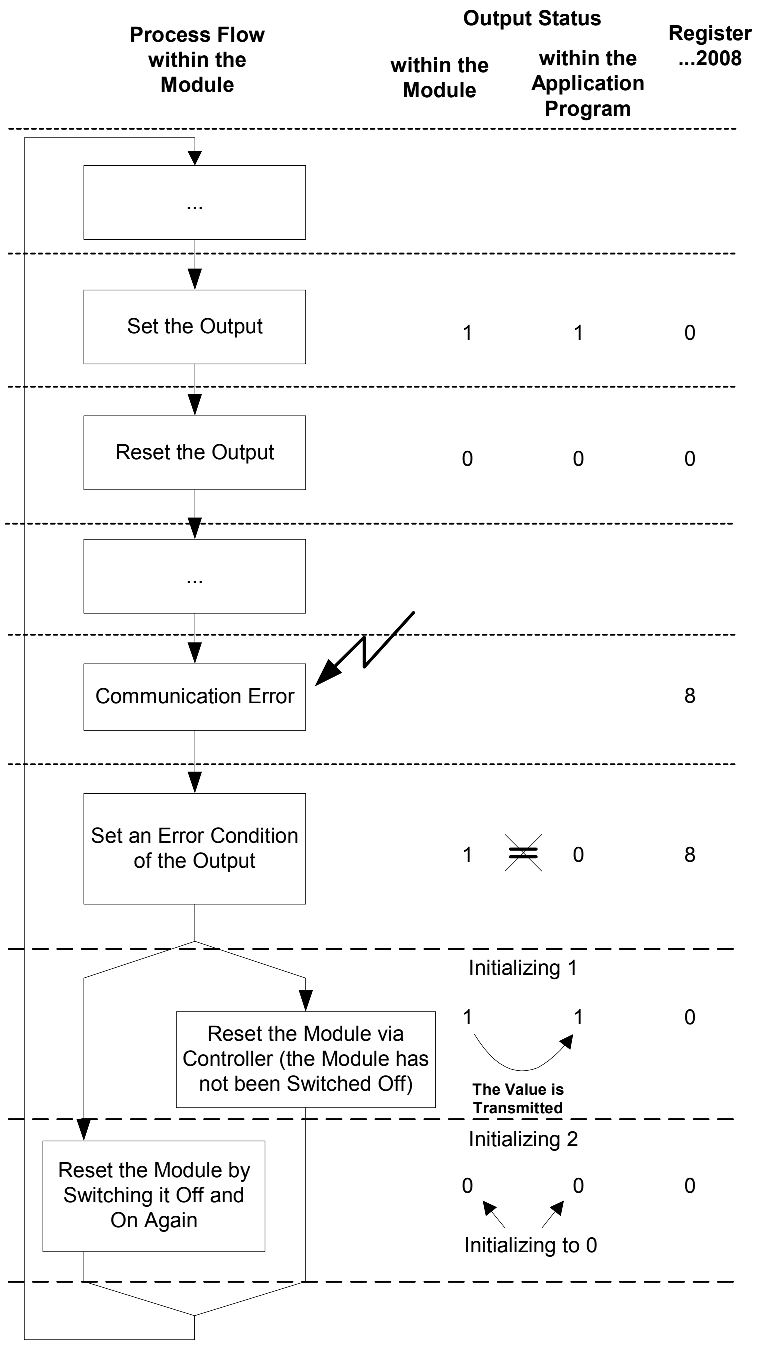

Process Flow within the Module
Output Status
within the Module
within the Application Program
Register ...2008
...
Set the Output
1
1
0
Reset the Output
0
0
0
...
Communication Error
8
Set an Error Condition of the Output
1
0
8
Initializing 1
Reset the Module via Controller (the Module has not been Switched Off)
1
1
0
The Value is Transmitted
Initializing 2
Reset the Module by Switching it Off and On Again
0
0
0
Initializing to 0

Fig. 9: Behaviour of the output status at re-initializing