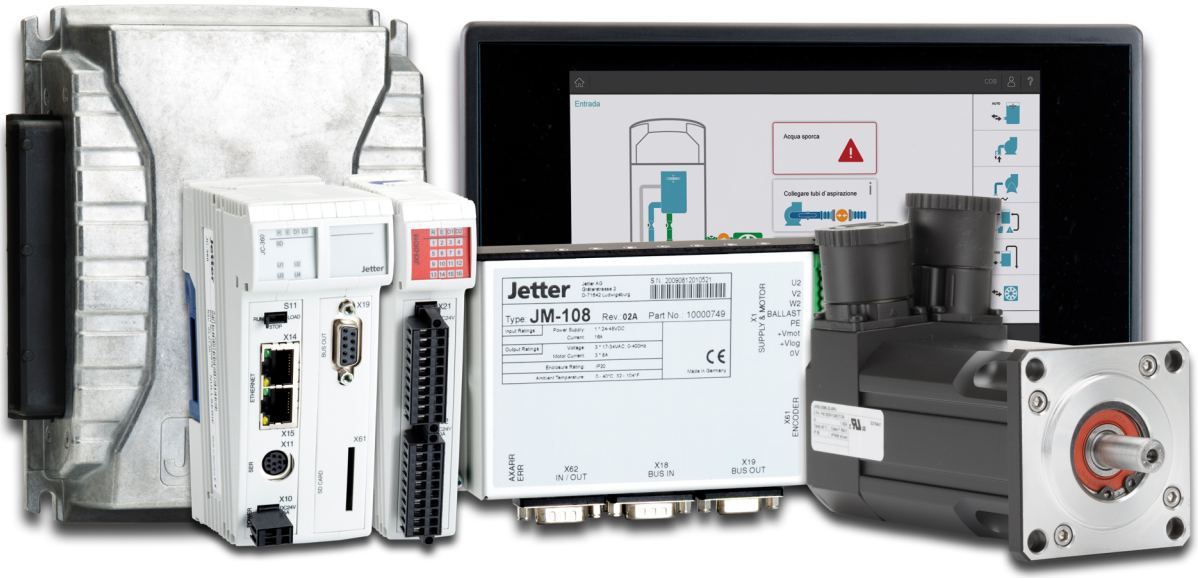
Jetter automation



Application-Oriented Manual

System registers

60882018

We automate your success.

Item # 60882018
Revision 1.01
April 2017 / Printed in Germany

Table of Contents

1 System commands

Introduction

In this chapter, the system command registers and the system commands will be explained in detail.

Contents

Description of system command registers

Registers - Overview

The following registers are described in this manual:

Register	Description
R 202960	System password register
R 202961	System command register
R 202962	System status register

R 202960

System password register

Enter system password 1112502132 (0x424F6F74) into this register. Then enter the required command value into the system command register. Now, the controller sets the value of this register to 0.

Register properties	
Value	1112502132 (0x424F6F74)

R 202961

System command register

Enter the system commands into this register. Then the controller executes the command. Then, it sets the value of this register to 0.

Commands	
102	**Restart the controller**
104	**Reset remanent parameters**
122	**Deactivate - Wait for communication**
123	**Activate - Wait for communication**
160	**Deactivate - Task switch on I/O access**
161	**Activate - Task switch on I/O access**
170	**Deactivate - Resume task time slot**

Commands	
171	**Activate - Resume task time slot**
310	**Load configuration files**
311	**Load module configuration**
312	**Load process data configuration for Ethernet system bus**
313	**Stop process data communication for Ethernet system bus**
330	**Disable JetIPScan client**
331	**Enable JetIPScan client**
410	**Disable JetSync blockage**
411	**Enable JetSync blockage for all ports**
412	**Enable JetSync blockage for port X15**

Register properties	
Access	System password register contains the correct password.

R 202962

System status register

The system status register lets you evaluate the system conditions.

Meaning of the individual bits		
Bit 0	**Task switch on I/O access**	
	0 =	No task switching in the application program on I/O access.
	1 =	Task switching is carried out in the application program on I/O access.
Bit 1	**Wait for communication**	
	0 =	The controller waits for communication requests for a short time.
	1 =	The controller does not wait for communication requests.
Bit 2	**JetIPScan client**	
	0 =	JetIPScan client is not active
	1 =	JetIPScan client is active
Bit 8	**JetSync blockage**	
	0 =	JetSync blockage is not active
	1 =	JetSync blockage is active

Register properties	
Access	Read

Description of system commands

System command 102 **Restart the controller**

Effect:

The controller is restarting. The effect is the same as when you switch the power supply off and on again.

Purpose:

Use this command, for example, if you have made changes to system registers or system files which become active only when the controller is rebooted.

System command 104 **Reset remanent parameters**

Effect:

The controller will reset remanent parameters to their default values (factory settings).

Register number	Description	Factory settings
100002023	JX3 system bus: I/O dummy modules	65535
100002034	JX3 system bus: Number of retries	1
200002023	JX2 system bus: I/O dummy modules	-1
200002024	JX2 system bus: Slave dummy modules	255
200002029	JX2 system bus: Baud rate	7
200002032	JX2 system bus: Switch-on delay	60
200002077	JX2 system bus: Special functions	0

Application:

Use this command, if you want to undo changes to remanent parameters.

System command 122 **Deactivate - Wait for communication**

Effect:

Not before there are definite requests, the controller will communicate with external communication partners.

Advantage:

The controller executes the application program faster.

Disadvantage:

On average, external communication partners have to wait longer for a response from the controller.

System command 123 — **Activate - Wait for communication**

Effect:

The controller cyclically checks for communication requests from external partners for 1 to 2 ms.

Advantage:

External communication partners get a faster reply from the controller.

Disadvantage:

Application program processing takes slightly longer.

System command 160 — **Deactivate - Task switch on I/O access**

Effect:

While the controller is accessing modules on the JX2 or JX3 system bus, other tasks of the application program are not processed.

Advantage:

The controller executes I/O accesses as fast as possible.

Disadvantage:

As certain I/O accesses are significantly slower than access to internal variables, response time of other tasks may increase.

System command 161 — **Activate - Task switch on I/O access**

Effect:

While the controller is accessing modules on the JX2 or JX3 system bus, it processes the other tasks of the application program.

Advantage:

The execution time of certain I/O accesses which may be relatively long does not affect the response time of other tasks.

Disadvantage:

The run time of the other tasks influences the execution time of several I/O accesses.

System command 170 — **Deactivate - Resume task time slot**

Effect:

When a normal application task has been interrupted by a cyclic task or the Ethernet system bus publisher, processing the following application task is resumed. The remaining time of the time slot of the interrupted task lapses for one cycle.

Advantage:

The total cycle time for processing all tasks is not influenced so much by the cyclic events.

Disadvantage:

This way, the interrupted task is assigned less runtime.

System command 171 **Activate - Resume task time slot**

Effect:

When a normal application task has been interrupted by a cyclic task or the Ethernet system bus publisher, processing the interrupted application task is resumed. This way, the interrupted task is processed for the remaining time of its time slot.

Advantage:

The interrupted task is assigned its total runtime.

Disadvantage:

The total cycle time for processing all tasks is influenced by the cyclic events to a greater extend.

System command 310 **Load configuration files**

Effect:

The controller loads the module configuration file (ModConfig.da) and the configuration files for process data communication on the Jetter Ethernet system bus (Publisher.pub, Subscriber.sub) from the file system. This corresponds to a combination of commands 311 and 312.

Purpose:

Once the transfer of these files into the controller's file system is completed, system command 310 enables their contents.

System command 311 **Load module configuration**

Effect:

The controller loads the module configuration file (ModConfig.da) from the file system.

Purpose:

Once the transfer of this file into the controller's file system is completed, system command 311 enables its contents.

System command 312 **Load process data configuration for Ethernet system bus**

Effect:

The controller loads the configuration files for process data communication on the Jetter Ethernet system bus (Publisher.pub, Subscriber.sub) from the file system.

Purpose:

Once the transfer of these files into the controller's file system is completed, system command 312 enables their contents.

System command 313 — **Stop process data communication for Ethernet system bus**

Effect:

Process data communication on the Jetter Ethernet system bus stops.

Purpose:

Transfer the configuration files for process data communication on the Jetter Ethernet system bus into the controller's file system. Then, stop process data communication by issuing system command 313. Finally, enable the contents of the new files.

System command 330 — **Disable JetIPScan client**

Effect:

This command lets you disable the JetIPScan client. The server, however, remains enabled.

Purpose:

For testing purposes

System command 331 — **Enable JetIPScan client**

Effect:

This command lets you enable the JetIPScan client.

Purpose:

This command lets you enable the JetIPScan client which has been disabled for testing purposes.

System command 410 — **Disable JetSync blockage**

Effect:

- The JetSync blockage is disabled for all ports. Bit 8 in R 202962 is reset.
- The Jetter Ethernet system bus multicast frames are transmitted to all ports (X14, X15 and CPU).

Purpose:

The JetSync blockage enabled by system command 411 or 412 is disabled. Forwarding the Jetter Ethernet system bus multicast frames to all ports again corresponds to the on-state of the controller.

System command 411 — **Enable JetSync blockage for all ports**

Effect:

- The JetSync blockage is enabled for all ports (X14, X15, and CPU). Bit 8 in R 202962 is set.

- Jetter Ethernet system bus multicast frames which are received on a port are not forwarded to any of the other ports.
- All other Ethernet frames are forwarded as usual.

Purpose:

This command lets you prevent forwarding Jetter Ethernet system bus multicast frames to the CPU and the other ports. This way, networks are split and thus data traffic - e.g. from the machine network to higher-level networks - is reduced.

Address space

Splitting is carried out on Ethernet level via the multicast address range of the Jetter Ethernet system bus.
0x01 00 5E 40 00 00 ... 0x01 00 5E 40 00 FF

System command 412 **Enable JetSync blockage for port X15**

Effect:

- The JetSync blockage is enabled for port X15 only. Bit 8 in R 202962 is set.
- Jetter Ethernet system bus multicast frames of the CPU are forwarded to port X14 only.
- Jetter Ethernet system bus multicast frames of port X14 are forwarded to the CPU only.
- Jetter Ethernet system bus multicast frames of port X15 are forwarded to the CPU and to port X14.
- All other Ethernet frames are forwarded as usual.

Purpose:

This command lets you prevent forwarding Jetter Ethernet system bus multicast frames to port X15. This way, networks are split and thus data traffic - e.g. from the machine network to higher-level networks - is reduced.

Address space

Splitting is carried out on Ethernet level via the multicast address range of the Jetter Ethernet system bus.
0x01 00 5E 40 00 00 ... 0x01 00 5E 40 00 FF

2 Startup delay time

Introduction

The device provides a register to which a delay time can be written.

Application

The boot process of the device is delayed by the entered delay time.

Contents

Setting the startup delay

Introduction

If other devices connected to the bus have got a longer startup time, the boot process must be delayed.

Set delay time

To set the delay time, proceed as follows:

Step	Action
1	Switch on the device.
2	Enter the password. For this, write value 1112502132 (0x424f6f74) to R 202970.
3	Enter the desired delay time in steps of 100 ms into register 202971.

Result: The next boot process will be delayed by the set startup delay time before initializing the JX2 and JX3 system bus.

R 202970

Password register

Enter 1112502132 (0x424F6F74) into this register. Then enter the desired value into the startup delay time register. Now, the controller sets the value of this register to 0.

Register properties	
Value	1112502132 (0x424F6F74)

R 202971

Startup delay time

Write into this register the delay time in multiples of 100 milliseconds.

Register properties	
Values	0 (OFF) ... 3,000 (300 seconds)
Value after reset	As described above (remanent)

Procedure

- The controller only executes start delay, when switch S11 is in *RUN* position.
- Start delay is terminated by leaving the *RUN* position.

Display

- LED **D1** flashing slowly during the first half of the start delay time (approx. 1 Hz).
- LED **D1** flashing fast during the second half of the start delay time (approx. 4 Hz).

3 Realtime clock

Introduction

The device is equipped with a component which maintains time and date settings for a certain time even when it is not energized.

Use by the customer

The customer uses the realtime clock for the following function:

- File date and time when creating a log file with timestamp

Restrictions

When using the real-time clock, the following restrictions apply:

- When the device is de-energized the power reserve is limited.
- The realtime clock has no automatic daylight savings time function.

Further information on programming

For further information on programming the realtime clock, please turn to the application-oriented manual *System Registers* in the download area of our **homepage https://www.jetter.de/downloads/application-notes/application-oriented-manual.html**.

Contents

Technical specifications

Technical data - Realtime clock

For fthe technical data, please refer to the user manual of the device.

Behavior when the power reserve has elapsed

If the device has been separated from the battery for a longer period of time and the RTC power reserve has elapsed, it takes the following actions when re-booting:

Step	Description
1	During the boot process the device detects that the power reserve has elapsed (R 367010). Register 367011 shows the battery voltage in millivolts. This function can be used to trigger a warning message in good time.
2	The device sets date and time to their default values: Date: Saturday, January 01, 2000 Time: 0:00 a.m.

Programming

Programming using STX

To program date and time it is advisable to use the functions provided by JetSym STX:

- DateTimeActual()
- DateTimeDecode()
- DateTimeEncode()
- DateTimeIsValid()
- DateTimeSet()

For more information on these functions refer to the JetSym online help.

If you make use of the above functions, the minimum time interval is one second. If you need a time interval of one second, programming must be made by using the registers described below.

Programming using registers

Depending on the respective application, access to the real-time clock via registers might be required. For this, there are two register sets:

- Register set 1 is for directly accessing individual real-time clock values.
- Changes to values in register set 1 are immediately transferred to the real-time clock.
- Register set 2 operates within a buffer. In the buffer, all real-time clock values are consistently read out and written.
- Not before the trigger register is written to, the value changes made in or out of register set 2 are transferred.

Register overview

The following registers have been assigned to the real-time clock:

Register set 1: Direct access

Register	Description
R 102910	Milliseconds
R 102911	Seconds
R 102912	Minutes
R 102913	Hours
R 102914	Weekday (0 = Sunday)
R 102915	Day
R 102916	Month
R 102917	Year

Register set 2: Buffer access

Register	Description
R 102920	Milliseconds
R 102921	Seconds
R 102922	Minutes
R 102923	Hours
R 102924	Weekday (0 = Sunday)
R 102925	Day
R 102926	Month
R 102927	Year
R 102928	Read/write trigger

R 102910

Milliseconds

This register contains the millisecond of the actual time.

Register properties	
Values	0 ... 999
Value after reset	0

R 102911

Seconds

This register contains the seconds of the actual time.

Register properties		
Values	0 ... 59	
Value after reset	**If ...**	**... then ...**
	the power reserve has not elapsed,	actual time
	the power reserve has elapsed,	0

R 102912

Minutes

This register contains the minutes of the actual time.

Register properties		
Values	0 ... 59	
Value after reset	**If ...**	**... then ...**
	the power reserve has not elapsed,	actual time
	the power reserve has elapsed,	0

R 102913

Hours

This register contains the hours of the actual time.

Register properties		
Values	0 ... 23	
Value after reset	**If ...**	**... then ...**
	the power reserve has not elapsed,	actual time
	the power reserve has elapsed,	0

R 102914

Weekday

This register contains the weekday of the actual date.

Register properties		
Values	0 ... 6 (0 = Sunday)	
Value following a reset	**If ...**	**... then ...**
	the power reserve has not elapsed,	actual time
	the power reserve has elapsed,	0

R 102915

Day

This register contains the day of the actual date.

Register properties		
Values	1 ... 31	
Value after reset	**If ...**	**... then ...**
	the power reserve has not elapsed,	actual time
	the power reserve has elapsed,	1

R 102916

Month

This register contains the month of the actual date.

Register properties		
Values	1 ... 12	
Value after reset	**If ...**	**... then ...**
	the power reserve has not elapsed,	actual time
	the power reserve has elapsed,	1

R 102917

Year

This register contains the year of the actual date.

Register properties		
Values	0 ... 99	
Value after reset	**If ...**	**... then ...**
	the power reserve has not elapsed,	actual time
	the power reserve has elapsed,	0

R 102920

Milliseconds

This register contains the milliseconds stored in the buffer.

Register properties	
Values	0 ... 999
Value after reset	0
Takes effect	After read/write access to register 102928

R 102921

Seconds

This register contains the seconds stored in the buffer.

Register properties	
Values	0 ... 59
Value after reset	0
Takes effect	After read/write access to register 102928

R 102922

Minutes

This register contains the minutes stored in the buffer.

Register properties	
Values	0 ... 59
Value after reset	0
Takes effect	After read/write access to register 102928

R 102923

Hours

This register contains the hours stored in the buffer.

Register properties	
Values	0 ... 23
Value after reset	0
Takes effect	After read/write access to register 102928

R 102924 **Weekday**

This register contains the weekday stored in the buffer.

Register properties	
Values	0 ... 6 (0 = Sunday)
Value following a reset	0
Takes effect	After read/write access to register 102928

R 102925 **Day**

This register contains the day stored in the buffer.

Register properties	
Values	0 ... 31
Value after reset	0
Takes effect	After read/write access to register 102928

R 102926 **Month**

This register contains the month stored in the buffer.

Register properties	
Values	0 ... 12
Value after reset	0
Takes effect	After read/write access to register 102928

R 102927 **Year**

This register contains the year stored in the buffer.

Register properties	
Values	0 ... 99
Value after reset	0
Takes effect	After read/write access to register 102928

R 102928 **Read/write trigger**

This register allows transferring values between buffer register and real-time clock.

Register properties	
Read	The actual date and time are transferred from real-time clock to buffer registers 102920 through 102927. The reading is undefined.
Write	The values contained in buffer registers 102920 ... 102927 are transferred to the real-time clock. The value written is ignored.

Sample program for real-time clock

Task

Read the actual time and date of the device and have the values displayed.

Solution

An application program task reads out the real-time clock at regular intervals. Then it outputs the readings properly formatted as trace message.

When you activate the trace mode in JetSym, JetSym displays these readings.

Software versions

The sample program has been tested for compliance with the following software versions:

- JetSym version 5.2
- Controller JC-350, OS version 1.24

For other sample programs, refer to JetSym online help.

JetSym STX program

```
Type
    // Structure of the RTC buffer
    TimeAndDate:   Struct
                    Second:      Int;
                    Minute:      Int;
                    Hour:        Int;
                    DayOfWeek:   Int;
                    Day:         Int;
                    Month:       Int;
                    Year:        Int;
                    Trigger:     Int;
                    End_Struct;
End_Type;

Var
    RTCregs:    TimeAndDate At %VL 102921;
End_Var;

Task ShowTimeAndDate Autorun
    Var
        Dummy:    Int;
    End_Var;

    Loop
        // Wait for one second
        Delay(T#1s);
        // Copy current time and current date
        // to RTC buffer
        Dummy := RTCregs.Trigger;
```

```
            // Displaying day of the week
            Case RTCregs.DayOfWeek Of
                0:  Trace('Sunday');
                    Break;
                1:  Trace('Monday');
                    Break;
                2:  Trace('Tuesday');
                    Break;
                3:  Trace('Wednesday');
                    Break;
                4:  Trace('Thursday');
                    Break;
                5:  Trace('Friday');
                    Break;
                6:  Trace('Saturday');
                    Break;
            End_Case;
            // Displaying date
            Trace(StrFormat(' , %2d.%02d.%4d , ',
                            RTCregs.Day,
                            RTCregs.Month,
                            RTCregs.Year + 2000));
            // Zeit anzeigen (plus cr/lf)
            Trace(StrFormat('%2d:%02d:%02d$n',
                            RTCregs.Hour,
                            RTCregs.Minute,
                            RTCregs.Second));
        End_Loop;
    End_Task;
```

4 System runtime

Introduction

The device provides several registers which are incremented by the operating system at regular intervals.

Application

These registers can be used to easily carry out time measurements in the application program.

Contents

Description of the runtime registers

Register overview

The device is equipped with the following runtime registers:

Register	Description
R 201000	Application time base in milliseconds
R 201001	Application time base in seconds
R 201002	Application time base in R 201003 * 10 ms
R 201003	Application time base units for R 201002
R 201004	System time base in milliseconds
R 201005	System time base in microseconds

R 201000

Application time base in milliseconds

Every millisecond this register is incremented by one.

Register properties	
Values	-2,147,483,648 ... 2,147,483,647 (overflowing)

R 201001

Application time base in seconds

Every second this register is incremented by one.

Register properties	
Values	-2,147,483,648 ... 2,147,483,647 (overflowing)

R 201002

Application time base in application time base units

Every [R 201003] * 10 ms this register value is incremented by one. Using the reset value 10 in register 201003, this register is incremented every 100 ms.

Register properties	
Values	-2,147,483,648 ... 2,147,483,647 (overflowing)

R 201003

Application time base units for R 201002

This register contains the multiplier for runtime register R 201002.

Register properties	
Values	1 ... 2,147,483,647 (* 10 ms)
Value after reset	10 (--> 100 ms)
Enabling conditions	After at least 10 ms

R 201004

System time base in milliseconds

Every millisecond this register value is incremented by one.

Register properties	
Values	-2,147,483,648 ... 2,147,483,647 (overflowing)
Type of access	Read

R 201005

System time base in microseconds

Every microsecond this register value is incremented by one.

Register properties	
Values	-2,147,483,648 ... 2,147,483,647 (overflowing)
Type of access	Read

Sample program - Runtime registers

Task

Measure how much time it takes to store variable values to a file.

Solution

Before storing the values, set register 201000 to 0.

Once the values have been stored, you can see from this register how much time it took to store the values [in milliseconds].

Software versions

The sample program has been tested for compliance with the following software versions:

- JetSym version 5.2
- Controller JC-350, OS version 1.24

For other sample programs, refer to JetSym online help.

JetSym STX program

```
Var
    DataArray:    Array[2000] Of Int;
    File1:        File;
    WriteTime:    Int;
    WriteIt:      Bool;
    MilliSec:     Int At %VL 201000;
End_Var;

Task WriteToFile Autorun
    Loop
        // Resetting the start flag
        WriteIt := False;
        // Wait for user to set start flag
        When WriteIt Continue;
        // Opening the file in write mode
        // If there is no file available, a new file
        // is created
        If FileOpen(File1, 'Test.dat', fWrite) Then
            // Set the application time base register to null
            MilliSec := 0;

            // Write the data range into the file
            FileWrite(File1, DataArray, SizeOf(DataArray));
            // Register the run time
            WriteTime := MilliSec;
            FileClose(File1);
            // Display the run time
            Trace(StrFormat('Time : %d [ms]$n', WriteTime));
        Else
            // Display the error message
            Trace('Unable to open file!$n');
        End_If;
    End_Loop;
End_Task;
```

5 Monitoring interface activities

Introduction

Several servers for variables have been integrated into the controller to make variables used within the controller accessible from outside. These servers support several protocols on different interfaces. The servers do not require any programming in the application program, but process requests from external clients on their own.

This chapter explains one possibility for detecting from within the application program whether communication with the servers takes place through these interfaces.

Monitored interface activities

The following interface activities can be monitored:

- pcomX server via serial interface
- JetIP server via Ethernet interface
- STX debug server via Ethernet interface

Purpose

The monitoring function for interface activities can be used, amongst others, for the following scenarios:

- Plants requiring process visualization to ensure safe operation. They can be transferred into a save state if communication fails.
- When the service technician connects an HMI, the application program automatically displays additional status information.

Contents

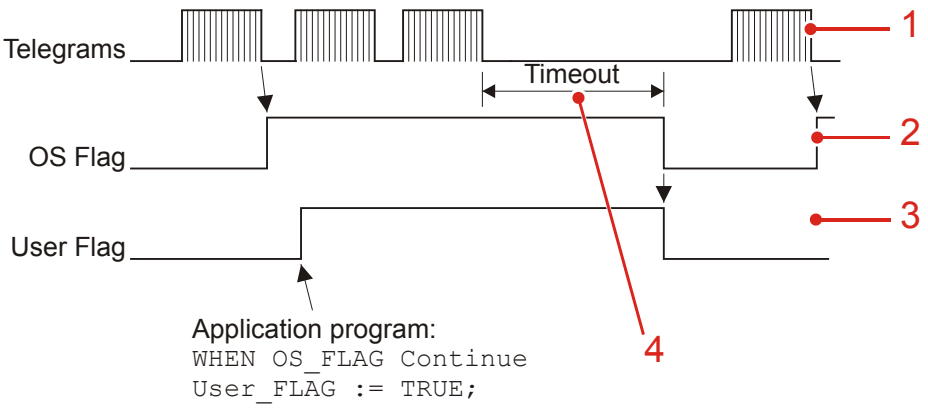
Operating principle

Introduction

The application program monitors the activity of a client communicating with a server of the device by means of two special flags and one special register per interface.

Overview

The illustration below shows the interdependence between interface activity and the two special flags, as well as the special register:



Number	Element	Description
1	Telegrams	The client places requests to the server.
2	OS flag	OS flag set by the device after receiving a request
3	User flag	You must set the user flag in the application program once the device has set the OS flag.This indicates that the connection has temporarily been disrupted even if the device resets the OS flag very quickly.
4	Timeout	Time of inactivity after which the OS resets both special flags. This time can be set in a special register.

Description

Interface activities are monitored as follows:

Step	Description
1	Enter the desired value into the timeout register of the application program. This way, the monitoring mode is activated as well.
2	When the controller receives the next telegram, the device sets the corresponding OS flag.
3	If the OS flag has been set, the application program also sets the respective user flag.

Step	Description
4	Each new telegram causes the timeout to restart.
5	If telegrams cease to arrive, both special flags are reset by the controller upon expiry of the timeout interval.
6	The application program detects that the device has reset the special flags and therefore takes appropriate action.
7	When further telegrams start arriving, the device sets the corresponding OS flag. The user flag, however, remains reset.

Programming

Registers/flags - Overview

For interface monitoring, the device provides the following registers and flags:

Timeout registers

Registers	Interface	Use
R 203000	JetIP (Ethernet)	■ Visualization ■ Controller networking
R 203005	STX debugging (Ethernet)	■ JetSym via Ethernet

Further features if control systems are applied

Register	Interface	Use
R 203001	pcomX (serial interface)	■ HMIs with alphanumeric display ■ JetSym via serial interface

Special flags

Flags	Interface	Use
F 2088	JetIP (Ethernet)	OS flag
F 2089		User flag
F 2098	STX debugging (Ethernet)	OS flag
F 2099		User flag

Further features if control systems are applied

Flags	Interface	Use
F 2090	pcomX (serial interface)	OS flag
F 2091		User flag

R 203000

Timeout in the case of JetIP (Ethernet)

This register contains the timeout for the JetIP server (Ethernet) in milliseconds.

Register properties	
Values	0 ... 2,147,483,647 [ms]
Value after reset	0 (monitoring disabled)

R 203001

Timeout in the case of pcomX (serial interface)

This register contains the timeout period for the pcomX server (serial interface) in milliseconds.

Register properties	
Values	0 ... 2,147,483,647 [ms]
Value after reset	0 (monitoring disabled)

R 203005

Timeout in the case of STX debugging (Ethernet)

This register contains the timeout for the STX debug server (Ethernet) in milliseconds.

Register properties	
Values	0 ... 2,147,483,647 [ms]
Value after reset	0 (monitoring disabled)

Enabling the monitoring function

To enable monitoring of interface activities, proceed as follows:

Step	Action
1	Enter the desired value into the timeout register of this interface.
2	Wait until the controller has set the OS flag of this interface.
3	Set the corresponding user flag.

Detecting a timeout

To detect a timeout, proceed as follows:

Step	Action	
1	Enable monitoring of interface activities (see above).	
2	Wait until the controller has reset the user flag of this interface. **Result**: A timeout has occurred.	
3	Check the corresponding OS flag.	
	If ...	**... then ...**
	... the OS flag is set,	... the connection was temporarily disrupted.
	... the OS flag is reset,	... the connection is still disrupted.

Jetter automation

Jetter AG
Graeterstrasse 2
71642 Ludwigsburg | Germany

Phone +49 7141 2550-0
Fax +49 7141 2550-425
info@jetter.de
www.jetter.de

We automate your success.