

XDX9100 Driver Manual

Johnson Controls Metasys DX9100 Protocol Driver



CPKSoft Engineering

Process Monitoring and Industrial Automation Software

Copyright 1990-2008, CPKSoft Engineering. All rights reserved.

Index

1.	Introduction	3
2.	Driver details	4
2.1.	Driver overview	4
2.2.	Supported devices.....	4
3.	Command list	5
3.1.	Get Device Type.....	5
3.2.	Read Single Item	6
3.3.	Read Extended Single Item	6
3.4.	Read Block of Consecutive Items	7
3.5.	Read Block of Consecutive Extended Items	8
3.6.	Read Block of Consecutive Bits.....	8
3.7.	Read Block of Consecutive Extended Bits	9
3.8.	Write Single Item	10
3.9.	Write Extended Single Item	10
3.10.	Write Block of Consecutive Items.....	11
3.11.	Write Block of Consecutive Extended Items.....	12
3.12.	Write Block of Consecutive Bits	13
3.13.	Write Block of Consecutive Extended Bits	13
4.	Appendices	15
4.1.	Error messages	15
4.2.	Keywords list.....	16

1. Introduction

CPKSoft Engineering assumes no responsibility for any errors that may appear in this document. If you have any suggestions for improvements or amendments or have found errors in this publication, please notify us.

This driver is included with all unlimited licenses of TAS-HMITalk. It is not sold separately. It requires the TAS-HMITalk ActiveX to work, therefore it cannot be used as a stand-alone driver.

If you use this driver in your applications, you need to include the xdx9100.tlk in the set of files that you distribute. This file must be located in the same folder where the hmitalk.ocx file is registered in order to be found by the activex when the applications are executed.

The source-code for the xdx9100.tlk driver is available in plain-C language for additional USD 499 if you own a license of TAS-HMITalk 8.04 or higher.

Refer to the following link to visit the xdx9100 driver page at CPKSoft Engineering website: <http://www.cpksoft.com/tabid/55/ProductID/32/PageIndex/1/Default.aspx>.

Visit this link if you want to see a complete list of drivers that are currently available for TAS-HMITak: <http://www.cpksoft.com/Drivers/tabid/55/Default.aspx>.

Also, refer to this link if you are interested in purchasing a license of the most recent version of TAS-HMITalk: <http://www.cpksoft.com/Products/tabid/54/Default.aspx>.

We welcome your comments about this document. You can reach us by e-mail at [contact @ cpksoft.com](mailto:contact@cpksoft.com).

2. Driver details

2.1. Driver overview

The XDX9100 driver allows you to connect to devices that conform to the Metasys DX9100 protocol specification.

The serial communication link between a supervisory system and up to 256 System 91 units, is an asynchronous half-duplex line split in RS-485 channels for 30 System 91 units each. The supervisory system initiates all data transactions on a point to point basis.

Typical port settings are 9600, N, 8, 1. The beginning of a request message should be delayed of at least $30000/BR$ msec (BR=baudrate bit/sec) after the reception of the last character of the previous answer message. You can use either T1Delay or T3Delay to implement this delay.

For a list of the Items that can be read, refer to your device's manual.

2.2. Supported devices

This driver can communicate with these devices, but is not necessarily limited to this list:

JOHNSON CONTROLS DR9100 Room Controllers
JOHNSON CONTROLS DC9100 Plant Controllers
JOHNSON CONTROLS IM9100 Modem Interfaces
JOHNSON CONTROLS DO9100 Digital Optimizers
JOHNSON CONTROLS DX9100 Digital Controllers
JOHNSON CONTROLS TC9100 Room Controllers
JOHNSON CONTROLS XT9100 Extension Modules
JOHNSON CONTROLS UI9100 User Interfaces
JOHNSON CONTROLS RC9100 Refrigeration Controllers

3. Command list

3.1. Get Device Type

Description of this command:

Executes a request for Item #0 to get the type of System 91 device connected to the serial link.

Possible device types are:

- 1 = DR9100 Room Controller Revision 1
- 2 = DC9100 Plant Controller Revision 1
- 3 = IM9100 Modem Interface Revision 1
- 4 = DO9100 Digital Optimizer Revision 1
- 5 = DX9100 Digital Controller Revision 1
- 6 = TC9100 Room Controller Revision 1
- 8 = XT9100 Extension Module Revision 1
- 9 = UI9100 User Interface Revision 1
- 10 = RC9100 Refrigeration Controller Revision 1
- 17 = DR9100 Room Controller Revision 2
- 18 = DC9100 Plant Controller Revision 2
- 21 = DX9100 Digital Controller Revision 2

Type of data handled by this command:

Analog Input

Number of points accepted by this command:

1

Meaning of the DriverP0 parameter:

Address of device (0-255)

Meaning of the DriverP1 parameter:

128 (80h)

Meaning of the DriverP2 parameter:

0

Meaning of the DriverP3 parameter:

0

Values that are returned:

Value in PointValue (0) = Device type

3.2. Read Single Item

Description of this command:

Executes a request for one single item in the range 0 to FFFh divided into four pages of up to 4095 items each.

Type of data handled by this command:

Analog Input

Number of points accepted by this command:

1

Meaning of the DriverP0 parameter:

System 91 unit address (0-255)

Meaning of the DriverP1 parameter:

128 (80h)

Meaning of the DriverP2 parameter:

Page # (0-3)

Meaning of the DriverP3 parameter:

Item # (0h-FFFh, 0-4095)

Meaning of the DriverP6 parameter:

Format of item being requested. Must be one of these characters: B = 1 byte data N = 2 bytes data, floating point number W = 2 bytes data, unsigned 16 bits hexadecimal number L = 4 bytes data, unsigned 32 bits hexadecimal number

Values that are returned:

Value in PointValue (0) = Requested item value

3.3. Read Extended Single Item

Description of this command:

Executes a request for one extended single item in the range of 0 to FFFFh.

Type of data handled by this command:

Analog Input

Number of points accepted by this command:

1

Meaning of the DriverP0 parameter:

System 91 unit address (0-255)

Meaning of the DriverP1 parameter:

132 (84h)

Meaning of the DriverP2 parameter:

Address of begin of module (0-FFFh, 0-4095).

Meaning of the DriverP3 parameter:

Relative item index (RI) from begin of module.

Meaning of the DriverP6 parameter:

Format of item being requested. Must be one of these characters: B = 1 byte data N = 2 bytes data, floating point number W = 2 bytes data, unsigned 16 bits hexadecimal number L = 4 bytes data, unsigned 32 bits hexadecimal number

Values that are returned:

Value in PointValue (0) = Requested extended item value

3.4. Read Block of Consecutive Items

Description of this command:

Reads a block of consecutive items.

Type of data handled by this command:

Analog Input

Number of points accepted by this command:

1-69 (depend on the item formats requested)

Meaning of the DriverP0 parameter:

System 91 unit address (0-255)

Meaning of the DriverP1 parameter:

168 (A8h)

Meaning of the DriverP2 parameter:

0

Meaning of the DriverP3 parameter:

First item # (0h-FFh, 0-255)

Meaning of the DriverP6 parameter:

String of characters with the format of each item being requested. Must be one of these characters: B = 1 byte data N = 2 bytes data, floating point number W = 2 bytes data, unsigned 16 bits hexadecimal number L = 4 bytes data, unsigned 32 bits hexadecimal number

Example: If requesting 3 unsigned 16 bits and one floating point number, use P6=WWWN

Values that are returned:

Value in PointValue (0) = First requested item value

Value in PointValue (1) = Second requested item value

...

Value in PointValue (DriverNumPoints-1) = Last requested item value

3.5. Read Block of Consecutive Extended Items

Description of this command:

Reads a block of consecutive extended items.

Type of data handled by this command:

Analog Input

Number of points accepted by this command:

1-69 (depend on the item formats requested)

Meaning of the DriverP0 parameter:

System 91 unit address (0-255)

Meaning of the DriverP1 parameter:

172 (ACh)

Meaning of the DriverP2 parameter:

0

Meaning of the DriverP3 parameter:

First item # (0h-FFFh, 0-4095)

Meaning of the DriverP6 parameter:

String of characters with the format of each item being requested. Must be one of these characters: B = 1 byte data N = 2 bytes data, floating point number W = 2 bytes data, unsigned 16 bits hexadecimal number L = 4 bytes data, unsigned 32 bits hexadecimal number

Example: If requesting 3 unsigned 16 bits and one floating point number, use P6=WWWN

Values that are returned:

Value in PointValue (0) = First requested item value

Value in PointValue (1) = Second requested item value

...

Value in PointValue (DriverNumPoints-1) = Last requested item value

3.6. Read Block of Consecutive Bits

Description of this command:

Reads a block of consecutive bits from a given starting item, returning the status of the 8 bits for each byte read.

Type of data handled by this command:

Digital Input

Number of points accepted by this command:

1-552 (depend on the item formats involved)

Meaning of the DriverP0 parameter:

System 91 unit address (0-255)

Meaning of the DriverP1 parameter:

168 (A8h)

Meaning of the DriverP2 parameter:

Number of items to be read (1-69). According to each Item format in the block being read, there must be an enough number of bytes to provide as many bits as is requested in DriverNumPoints.

Meaning of the DriverP3 parameter:

First item # (0h-FFh, 0-255)

Values that are returned:

Value in PointValue (0) = First bit of first byte read

Value in PointValue (1) = Second bit of first byte read

...

Value in PointValue (DriverNumPoints-1) = Last bit of last byte read

3.7. Read Block of Consecutive Extended Bits

Description of this command:

Reads a block of consecutive bits from a given starting extended item, returning the status of the 8 bits for each byte read.

Type of data handled by this command:

Digital Input

Number of points accepted by this command:

1-552 (depend on the item formats involved)

Meaning of the DriverP0 parameter:

System 91 unit address (0-255)

Meaning of the DriverP1 parameter:

172 (ACh)

Meaning of the DriverP2 parameter:

Number of items to be read (1-69). According to each Item format in the block being read, there must be an enough number of bytes to provide as many bits as is requested in DriverNumPoints.

Meaning of the DriverP3 parameter:

First item # (0h-FFFh, 0-4095)

Values that are returned:

Value in PointValue (0) = First bit of first byte read

Value in PointValue (1) = Second bit of first byte read

...

Value in PointValue (DriverNumPoints-1) = Last bit of last byte read

3.8. Write Single Item

Description of this command:

Executes a request to write one single item in the range 0 to FFFh divided into four pages of up to 4095 items each.

Type of data handled by this command:

Analog Output

Number of points accepted by this command:

1

Meaning of the DriverP0 parameter:

System 91 unit address (0-255)

Meaning of the DriverP1 parameter:

192 (C0h)

Meaning of the DriverP2 parameter:

Page # (0-3)

Meaning of the DriverP3 parameter:

Item # (0h-FFFh, 0-4095)

Meaning of the DriverP6 parameter:

Format of item being written. Must be one of these characters: B = 1 byte data W = 2 bytes data, unsigned 16 bits hexadecimal number L = 4 bytes data, unsigned 32 bits hexadecimal number

Values that are sent:

Value in PointValue (0) = New item value

3.9. Write Extended Single Item

Description of this command:

Executes a request to write one extended single item in the range of 0 to FFFh.

Type of data handled by this command:

Analog Output

Number of points accepted by this command:

1

Meaning of the DriverP0 parameter:

System 91 unit address (0-255)

Meaning of the DriverP1 parameter:

196 (C4h)

Meaning of the DriverP2 parameter:

Address of begin of module (0-FFFh, 0-4095).

Meaning of the DriverP3 parameter:

Relative item index (RI) from begin of module.

Meaning of the DriverP6 parameter:

Format of item being written. Must be one of these characters: B = 1 byte data W = 2 bytes data, unsigned 16 bits hexadecimal number L = 4 bytes data, unsigned 32 bits hexadecimal number

Values that are sent:

Value in PointValue (0) = New extended item value

3.10. Write Block of Consecutive Items

Description of this command:

Writes a block of consecutive items.

Type of data handled by this command:

Analog Output

Number of points accepted by this command:

1-69 (depend on the item formats written)

Meaning of the DriverP0 parameter:

System 91 unit address (0-255)

Meaning of the DriverP1 parameter:

232 (E8h)

Meaning of the DriverP2 parameter:

0

Meaning of the DriverP3 parameter:

First item # (0h-FFFh, 0-255)

Meaning of the DriverP6 parameter:

String of characters with the format of each item being written. Must be one of these characters: B = 1 byte data W = 2 bytes data, unsigned 16 bits hexadecimal number L = 4 bytes data, unsigned 32 bits hexadecimal number

Example: If writing 3 unsigned 16 bits and one byte point number, use P6=WWWB

Values that are sent:

Value in PointValue (0) = New first item value

Value in PointValue (1) = New second item value

...

Value in PointValue (DriverNumPoints-1) = New last item value

3.11. Write Block of Consecutive Extended Items

Description of this command:

Writes a block of consecutive extended items.

Type of data handled by this command:

Analog Output

Number of points accepted by this command:

1-69 (depend on the item formats written)

Meaning of the DriverP0 parameter:

System 91 unit address (0-255)

Meaning of the DriverP1 parameter:

236 (ECh)

Meaning of the DriverP2 parameter:

0

Meaning of the DriverP3 parameter:

First item # (0h-FFFh, 0-4095)

Meaning of the DriverP6 parameter:

String of characters with the format of each item being written. Must be one of these characters: B = 1 byte data W = 2 bytes data, unsigned 16 bits hexadecimal number L = 4 bytes data, unsigned 32 bits hexadecimal number

Example: If writing 3 unsigned 16 bits and one byte point number, use P6=WWWB

Values that are sent:

Value in PointValue (0) = New first item value

Value in PointValue (1) = New second item value

...

Value in PointValue (DriverNumPoints-1) = New last item value

3.12. Write Block of Consecutive Bits

Description of this command:

Writes a block of consecutive bits, starting from a given item. Every 8 bits, one byte will be output. Make sure that you write as many bits as required to complete each whole item size, according of the format of each item.

Type of data handled by this command:

Digital Output

Number of points accepted by this command:

1-552

Meaning of the DriverP0 parameter:

System 91 unit address (0-255)

Meaning of the DriverP1 parameter:

232 (E8h)

Meaning of the DriverP2 parameter:

Number of items that will be affected (1-69), according to the number of bits that will be written. This depends on the format of each item in the affected block. The number of bits is given by DriverNumPoints.

Meaning of the DriverP3 parameter:

First item # (0h-FFh, 0-255)

Values that are sent:

Value in PointValue (0) = first bit of first item affected

Value in PointValue (1) = second bit of first item affected

...

Value in PointValue (DriverNumPoints-1) = last bit of last item affected

3.13. Write Block of Consecutive Extended Bits

Description of this command:

Writes a block of consecutive extended bits, starting from a given item. Every 8 bits, one byte will be output. Make sure that you write as many bits as required to complete each whole item size, according of the format of each item.

Type of data handled by this command:

Digital Output

Number of points accepted by this command:

1-552

Meaning of the DriverP0 parameter:

System 91 unit address (0-255)

Meaning of the DriverP1 parameter:

236 (ECh)

Meaning of the DriverP2 parameter:

Number of items that will be affected (1-69), according to the number of bits that will be written. This depends on the format of each item in the affected block. The number of bits is given by DriverNumPoints.

Meaning of the DriverP3 parameter:

First item # (0h-FFFh, 0-4095)

Values that are sent:

Value in PointValue (0) = first bit of first item affected

Value in PointValue (1) = second bit of first item affected

...

Value in PointValue (DriverNumPoints-1) = last bit of last item affected

4. Appendices

4.1. Error messages

The following list shows all the possible error messages that can be returned by the protocol driver during a failed communication in the 'DriverStatus' property.

This list does not include some error messages that can be returned by the activex component while attempting to establish a connection.

- [1005] DRIVER (Internal): Invalid driver stage
- [1300] PROTOCOL (Timeout): No answer
- [1429] PROTOCOL (Format): Unknown response
- [1433] PROTOCOL (Format): Validation error in device response
- [2147] CONFIG (NumValues): Only one value can be read or written
- [2148] CONFIG (NumValues): Only one value can be requested
- [2149] CONFIG (NumValues): Only one value can be written
- [2189] CONFIG (NumValues): Too many values (max=16)
- [2245] CONFIG (NumValues): Too many values requested
- [2280] CONFIG (NumValues): Too many items to be written
- [2281] CONFIG (NumValues): Too many bits to be written
- [3014] CONFIG (P0): Invalid device address (0-255)
- [3508] CONFIG (P1): Invalid command
- [4062] CONFIG (P2): Invalid first rack module number (0-15)
- [4086] CONFIG (P2): Invalid rack module number (0-16)
- [4130] CONFIG (P2): Invalid page number (0-3)
- [4131] CONFIG (P2): Invalid module begin address (0000h-FFFFh)
- [4133] CONFIG (P2): Invalid number of items
- [4551] CONFIG (P3): Invalid number of samples (0-255)
- [4587] CONFIG (P3): Invalid item number
- [8351] CONFIG (Remote): Unknown error type
- [8368] CONFIG (Remote): Idle device
- [8369] CONFIG (Remote): Not existing command code
- [8370] CONFIG (Remote): Data not matching the item or function type
- [8371] CONFIG (Remote): Not existing item or function
- [8372] CONFIG (Remote): Temporarily impossible to access the item
- [8373] CONFIG (Remote): Not programmable item
- [8374] CONFIG (Remote): Table programmed with illegal items
- [8375] CONFIG (Remote): Trend programmed with illegal items
- [8376] CONFIG (Remote): Invalid functional module
- [8377] CONFIG (Remote): exceeding addressing range
- [8378] CONFIG (Remote): Password not enabled

4.2. Keywords list

The following list shows a set of words directly related to this driver.

"Controllers, CONTROLS, DC9100, DO9100, DR9100, DX9100, Extension, IM9100, Interfaces, JOHNSON, Metasys, Modem, Modules, Optimizers, Plant, RC9100, Refrigeration, Room, TC9100, UI9100, User, XT9100".