W&T OPC Server for Network I/O Devices

Version 2.10

Subject to error and alteration

Since it is possible that we make mistakes, you must not use any of our statements without verification. Please inform us of any error or misunderstanding you come about, so we can identify and eliminate it as soon as possible.

Carry out your work on or with W&T products only to the extent as they are described here and after you have completely read and understood the manual or guide. We are not liable for unauthorized repairs or tampering. When in doubt, check first with us or with your dealer.

1 Introduction

1.1 About OPC Servers

OPC (OLE for Process Control) is a software interface for accessing industrial process data, based on Microsoft's OLE technology. Application programs like HMI or SCADA software that use this interface are called OPC clients. On the other side of the interface we have OPC Servers. Those are device drivers that represent some particular hardware in an abstract way, as a set of OPC variables.

The present OPC server implements the OPC Data Access 2.0 specification and is intended for use with devices of the W&T product families "Digital I/O Servers" and "Serial Com Servers". In terms of architecture, it is a monolithic application, which contains the OPC server itself as well as a user interface for configuration and diagnostics. This is in contrast to another common approach of having the OPC server run as a separate system service in the background.

1.2 Installation

Start the setup program from the installation disk (for example by clicking "Run..." in the Start menu, then typing "A:\Setup"). This will install and register the OPC server on your computer.

The OLE server name that OPC clients will have to supply in order to connect to the server is: "Wiesemann-Theis.DigitalEA". The OPC server will start automatically when such a connection is attempted. For configuration you can also start it manually: You will find an entry "W&T OPC Server" under "Programs" in the Start menu.

Note: Installation on Windows 95 will possibly fail, with error messages about missing functions in OLE32.dll. An updated version of this DLL that will fix the problem is contained in the "DCOM for Windows 95" package, which is available as a free download from Microsoft, at http://www.microsoft.com/com/dcom/dcom95/dcom1_3.asp.

1.3 Uninstallation

To remove the OPC Server, use the "Software" option on the Control Panel. Look for "W&T OPC-Server for Digital IO" in the list.

2 Device Configuration

You will come across the following two dialogs every time you add a new I/O Server or serial Com Server, or want to change some settings for an existing device.

2.1 I/O Servers

Host name or IP address: Must agree with the IP address (four numbers, separated by dots, e.g. 172.16.232.77) that was set on the device itself. In case the address has also been assigned a host name, using DNS or a similar name service, you may use this name instead of the IP address as well.

TCP port number: For devices from the old 50000 series this is usually (at least by default) 49153. For the new Web-IO devices (e.g. model 57630) please check with the settings in the device's web interface.

System password: Applies only to Web-IO devices. Enter the Configurator Password, if one has been assigned to the device. When in doubt about the correct password, wait and see what happens after closing the dialog. Look for the status messages "password required" and "incorrect password" in particular.

Unique device number: This number is automatically assigned when adding a server, and is intended to make sure that all devices end up with different names (Box1, Box2, Com3, Box4 and so on). Changing it by hand might make sense in order to let the I/O ports and the serial port on one physical device also share the same device number. The resulting device names could then be, for example, "Box1" and "Com1".

Update at least every ... seconds: I/O Servers report some events on their own initiative (changed input lines in particular), others require regular polling (output currents on devices with the measuring option, and input counters on a Web-IO, for example). This value sets the interval between two such queries, where your input will be rounded to a multiple of 0.1 seconds. Another purpose of regular polling is to quickly identify a disrupted TCP connection. A problem is assumed (and the connection is reset) when the device takes longer to reply than one of these intervals. Caution: Setting a very small value can therefore prevent any connections to be made at all.

Poll only (i.e. don't request active notifications): Doing without automatic notifications has only one advantage: The amount of network traffic no longer depends on external events. However, even under the worst possible conditions (bouncing or oscillating input signals), I/O servers generate no more than a hundred notifications per second. So even then, only many I/O servers on the same network could become a problem big enough to justify choosing this option.

Device type: "Standard" means I/O server models 50210/50310. "With measuring/monitoring option" applies to models 50211/50311, which allow reading their output currents and internal temperature. The "detect" button may help to choose the right type, but it may also fail, for example if the device is currently in use by another TCP client.
Please note that the third device type, "Web-IO", needs to be configured appropriately, before the OPC server can access it.

2.2 Serial Com Servers

Options **"Host name or IP address"** and **"Unique device number"** have the same meaning as with I/O Servers, see above.

TCP port number: Is usually 8000, or 8100/8200/8300 for the serial ports B/C/D on a 4-port server. Each serial port actually occupies two TCP connections: One data connection on the port number as it was specified, and one control connection on a port number which is greater by 1094, that means 9094, 9194, 9294 or 9394.

System password: Is only supported by the more recent Highspeed Com Servers, where it is known by that very same name. This has nothing to do with the “telnet password” that you may still remember from earlier Com Server models.

Status update every ... seconds: Com Servers report incoming serial data on their own initiative, all other events (including changed handshake lines) require regular polling.

Handshake lines mapped for I/O: RS232 lines DSR/CTS/DTR/RTS can be referenced as OPC variables (E.0/E.1/A.0/A.1) in this mode and are no longer available for hardware handshake. This was originally intended for serial Com Servers equipped with the special interface module 18804 and has no useful application otherwise.

RS232 parameters: An initialization string like "19200,n,8,1,x", which sets the line parameters at the time the OPC server is started. Meaning of the individual parameters is: baud rate, parity bit (n = none, e = even, o = odd), number of data bits (5 - 8), number of stop bits (1 or 2), handshake protocol (x = software, XOn/XOff, h = hardware, RTS/CTS).
When no initialization string is given, the OPC server uses the serial settings as it finds them in the Com Server.

Incoming text complete after...: To represent the incoming serial data stream as an OPC string variable, it must first be split into packets. Either pauses or the occurrence of a series of special characters can be defined to end a packet.
When special characters are defined, pauses should be disabled (by setting a value of 0 seconds). Combination of both methods is possible, but not recommended.
Special characters are always entered as decimal numbers, even if they should be printable characters. For example, the correct input to have packets end at the occurrence of three plus signs ("+++") would be "43,43,43".

Filter from incoming data: Filtering special characters takes place after the serial data have already be split into packets, so filtering characters that are also packet delimiters is perfectly legal. Null bytes will be removed in any case, even if "0" is not expressly mentioned here, because they must not appear within text strings.

Append when sending: One reasonable use could be to append CR (13), LF (10) or both (13,10) to indicate end of line.

2.3 Setting up Web-IO Devices

Before accessing a Web-IO device by OPC, one of its binary channels must be allocated for that purpose: In the web interface, menu item "Config >> Device >> Basic Settings >> Binary 1 (or 2, respectively) >> TCP Server" should have "Application Mode: OPC Device" selected. On the same page, please also verify the setting of "Local Port", which must agree with the TCP port number supplied to the OPC server. (Recommended settings are 49153 for channel 1 and 49154 for channel 2.)

As another peculiarity, Web-IO devices allow various connections at the same time, including HTTP and the two binary channels just mentioned. To avoid conflicts in controlling the output lines, they may be read from any path, but only one path is allowed to write them. Which one that is, can be configured for each output individually, on page "Config >> Device >> Output Mode". You should do so, and assign some or all of the outputs to the binary channel that will be used with the OPC server.

2.4 Program Options

The "Options" dialog determines some details about the OPC server's behavior

Hide window: Recommended, after you are done configuring and testing everything else. To display a hidden window again, invoke the OPC server once more from the Start menu. Please not that this does not create a second instance of the OPC server, but really just brings up the one running in the background. Therefore don't close the window once you are done with your additional tests and/or changes, but rather just minimize it, which should make it disappear again.

Release I/O devices: In this context, "release" means to terminate all TCP connections to the devices, so that other clients can access them again. The effect is similar to the one achieved by disabling an individual device manually. The OPC variables of such a device still exist, but provide no more valid values.

Watchdog (VT_R4, R/W) is a global OPC variable, i.e. one not assigned to any particular I/O device. It contains a time value (in seconds) which is continuously decreased in value, if the respective option has been selected. Once it reaches zero, all I/O devices are released. The state of the OPC server then changes to OPC_STATUS_SUSPENDED, and reading individual variables yields nothing but OPC_QUALITY_NOT_CONNECTED. The client can prevent that by writing a non-zero value in regular intervals, for example a value of 15 every 10 seconds.

The other release mechanism, which only considers if there are still OPC clients connected, is obviously more elegant, since it requires no additional effort on behalf of the client. However it relies on all clients signing on and off correctly, which is not necessarily guaranteed. Conceivable problems might be programming errors or unexpected crashes in the client software, or a severed network connection to a remote client. The watchdog technique is not vulnerable to such problems.

Quit, if already hidden: As it says, automatic program termination strictly requires that the program window is not visible at that moment. Likewise, the program never terminates itself in a situation, where closing the window by hand would require an additional confirmation: That means neither if OPC clients are still connected, nor while there are unsaved changes to the device configuration.

3 Direct Access by Control Panels

For each server on the list, you can open a control panel, which offers essentially the same level of access as the OPC interface. In particular, the individual controls are labeled with the same names as the corresponding OPC variables.

This feature is useful mainly to become familiar with the behavior of the devices, and for example to see how the identification of serial data packets is affected by the various options. But don't expect too much from it, because you cannot open more than one control panel at the same time.

Abbreviations

OLE data types:

VT_BOOL: binary value
VT_I2, VT_I4: integer number
VT_R4: floating-point number
VT_BSTR: string of characters

Access:

R/W: read and write
R: read only
W: write only

3.1 OPC Variables for I/O Servers

For each I/O Server that is added to the configuration, the OPC server creates the following variables, where multiple I/O Servers are distinguished by the first part of the variable name: Box1, Box2, etc.

Box1.E.0 — Box1.E.11 (VT_BOOL, R): state of the inputs.

Box1.A.0 — Box1.A.11 (VT_BOOL, R/W): state of the outputs.

For I/O Servers with measuring and monitoring, there is also:

Box1.I.0 — Box1.I.11 (VT_R4, R): output currents (ampere).

Box1.T (VT_R4, R): internal temperature (Centigrade).

Box1.U (VT_R4, R): supply voltage (volt).

Box1.F (VT_BOOL, R/W): overload, outputs have been disabled. Reset the variable to enable them again.

These are for Web-IO only:

Box1.N.0 — Box1.N.11 (VT_I4, R/W): Event counters for inputs E.0 — E.11. Both rising and falling edges will be counted, i.e. a single pulse on the input increases the counter value by 2. In order to reset a counter, write a value of 0 to it.

Some I/O Server models have an additional serial port. In case you want to access that port by OPC as well, and can't seem to find the required variables in this list: You're right, they are not here. From the OPC server's point of view, such a serial port is a Com Server of its own. To reach your goal, you can add two individual devices, sharing the same IP address, where one of them is an I/O Server, and the other's type is "serial".

3.2 OPC Variables for Serial Com Servers

The device names for serial Com Servers are Com1, Com2, etc., and the individual variables are:

Com1.TxD (VT_BSTR, W): RS232 transmit data, assign a value to have it sent over the serial port.

Com1.RxD (VT_BSTR, R): RS232 received data (the most recent packet of text received from the serial port).

Com1.N (VT_I4, R/W): packet counter, is incremented every time a new packet of text arrives.

When handshake lines are mapped for I/O, there is also:

Com1.E.0 (VT_BOOL, R): digital input (DSR, or pin 4 on the 18804 I/O module).

Com1.E.1 (VT_BOOL, R): digital input (CTS, or I/O pin 7).

Com1.A.0 (VT_BOOL, R/W): digital output (DTR, or I/O pin 6).

Com1.A.1 (VT_BOOL, R/W): digital output (RTS, or I/O pin 8).

The following variables grant access to the same serial line parameters that can also be set by the initialization string in the device configuration dialog:

Com1.Line.Baud (VT_I4, R/W): baud rate, in bits per second.

Com1.Line.Bits (VT_I2, R/W): number of data bits per character (5, 6, 7 or 8).

Com1.Line.Stop (VT_I2, R/W): number of stop bits (1 or 2).

Com1.Line.Parity (VT_I2, R/W): parity bit: 0 = none, 1 = odd, 2 = even parity

Com1.Line.Flow (VT_BOOL, R/W): use a handshake protocol for flow control?

Com1.Line.Hard (VT_BOOL, R/W): use hardware or software handshake? (This variable is not present on serial ports with handshake lines mapped for I/O.)

4 Access from Visual Basic or VBA

Other than the basic Custom Interface, this OPC server only offers an Automation Interface according to OPC version 2.0. In case you are familiar with the old 1.0 Automation Interface: There are severe differences between the two versions. Besides that, at least Visual Basic version 5.0 or Office 97 is required to use the new interface. For a complete specification of the interface, see the OPC Foundation home page, http://www.opcfoundation.org. For a first impression, however, the following examples (Excel macros) may be helpful as well.

Note: Visual Basic can use the OPC interface only after the entry "OPC Automation 2.0" in the list of references has been activated. (In Visual Basic 6.0 this list is found in the "Project" menu, for Excel 97 see the "Extras" menu in the Visual Basic editor.)

4.1 Example: Looking for variables

This example uses the namespace browser to find all the variables on the OPC server, and places their names in column 1 of the current Excel worksheet.

```
Option Base 1

Sub OpcGetNames()
    ' Retrieve the available variable names and place them in column 1.
    Dim TheOpcServer As OPCServer
    Dim MyBrowser As OPCBrowser
    Set TheOpcServer = New OPCServer
    TheOpcServer.Connect ("Wiesemann-Theis.DigitalEA")
    Set MyBrowser = TheOpcServer.CreateBrowser

    Dim i As Integer
    ' First clear the contents of column 1.
    Columns("A").ClearContents
    MyBrowser.ShowLeafs (True)
    For i = 1 To MyBrowser.Count
        Cells(i, 1) = MyBrowser.Item(i)
    Next i

    Set MyBrowser = Nothing
    TheOpcServer.Disconnect
    Set TheOpcServer = Nothing
End Sub
```

4.2 Example: Reading values

This Example takes the variable names from column 1 of the current Excel worksheet, and for each of them reads the properties "value", "unit" and "description", and places them in the adjacent columns.
There is a special significance to the property "signal quality", which essentially says if the OPC server can offer a valid value for a particular variable. One possible reason for such problems could be that the OPC server has only just been started (maybe automatically in response to the macro invocation, so this is not an unlikely scenario), and has not yet established the TCP connections to its I/O devices.

```
Option Base 1

Sub OpcUpdate()
    ' Query all variables found in column 1 for description
    ' and current contents.
    Dim TheOpcServer As OPCServer
    Set TheOpcServer = New OPCServer
    TheOpcServer.Connect ("Wiesemann-Theis.DigitalEA")

    Dim PropertyIDs(5) As Long
    Dim Data() As Variant
    Dim Errors() As Long
    Dim i, j As Integer
    PropertyIDs(1) = 3      ' OPC_PROP_QUALITY
    PropertyIDs(2) = 2      ' OPC_PROP_VALUE
    PropertyIDs(3) = 100    ' OPC_PROP_UNIT
    PropertyIDs(4) = 101    ' OPC_PROP_DESC
    PropertyIDs(5) = 4      ' OPC_PROP_TIME
    Columns("B").ClearContents
    Columns("E:F").ClearContents
    i = 1
    While Cells(i, 1) <> ""
        TheOpcServer.GetItemProperties Cells(i, 1), 5, PropertyIDs, Data,
Errors
        For j = 2 To 5
            Cells(i, j) = Data(j)
        Next j
        If Data(1) = 20 Then        ' OPC_QUALITY_LAST_KNOWN
            Cells(i, 6) = "STALE"
        ElseIf Data(1) <> 192 Then  ' OPC_QUALITY_GOOD
            Cells(i, 6) = "ERROR"
            Range(Cells(i, 2), Cells(i, 3)).ClearContents
        End If
    n = n + 1
    Wend

    TheOpcServer.Disconnect
    Set TheOpcServer = Nothing
End Sub
```

4.3 Example: Writing values

Here we take a variable name from column 1 in the current worksheet, a value from column 2, and assign the value to the name.
Unlike reading values, writing cannot be done by using the "item properties" of the OPC server, but only by directly accessing the appropriate OPCItem object. By the way, reading could also be done this way, and it even would be more efficient than the method shown in example 2. (And speaking of efficiency: In a real VB application, as opposed to just an Excel macro, one would try to keep the OPCItem object around as long as possible, rather than creating and deleting it again for each and every variable access.)

```
Option Base 1

Sub OpcWrite()
    ’ Assign the variable (found in column 1) on the current row a
    ’ value (from column 2).
    Dim TheOpcServer As OPCServer
    Dim MyGroup As OPCGroup
    Dim MyItem As OPCItem
    Set TheOpcServer = New OPCServer
    TheOpcServer.Connect ("Wiesemann-Theis.DigitalEA")
    Set MyGroup = TheOpcServer.OPCGroups.Add("group")

    Set MyItem = MyGroup.OPCItems.AddItem(Cells(ActiveCell.Row, 1), 1234)
    MyItem.Write (Cells(ActiveCell.Row, 2))

    TheOpcServer.OPCGroups.Remove (MyGroup.ServerHandle)
    TheOpcServer.Disconnect
    Set TheOpcServer = Nothing
End Sub
```

5 One I/O Server, several Clients

The OPC server gains exclusive access to the I/O Servers and serial Com Servers that have been assigned to it, at the moment it successfully establishes a TCP connection to them. That is why a configuration with OPC servers on several computers accessing the same I/O Server makes no sense or at least would not work reliably.

To achieve a similar effect, use a single OPC server instead, with clients from several computers accessing it by DCOM (distributed COM). Please note that you have to grant explicit access rights before using DCOM. This is done using the program "DcomCnfg.exe", which is a standard component in Windows NT.

In order to use DCOM on Windows 95, you first have to install the update packages DCOM95.EXE and DCM95CFG.EXE. To see if this has already been done, try invoking DcomCnfg.exe from the command prompt. And if that succeeds, you can go right on, open the "Default Security" page in that program and set the "Enable remote connection" checkmark. Finally, be aware that Windows 95 does not allow an OLE server to be started in response to an external DCOM request. So on a Win95 computer, the OPC server would need to be started manually before OPC clients on other computers can connect to it.

More detailed information on DCOM is available in the Microsoft Knowledge Base. Some of the relevant articles are:

Q176799, INFO: Using DCOM Config (DCOMCNFG.EXE) on Windows NT
Q165101, HOWTO: Use Win95 as a DCOM Server
Q182248, HOWTO: Use DCOM Config (DCOMCNFG.EXE) with Windows 95
Q158508, FAQ: COM Security Frequently Asked Questions
Q174024, FAQ: DCOM95 Frequently Asked Questions

One of many approaches for locating these articles is the search engine on Microsoft’s support web site, http://support.microsoft.com. Make sure that category "English Knowledge Base" is selected, and enter an article number such as "Q165101".