



Product:	WagoIPCLib
Product version:	V1.0
Document ID:	UM-WagoIPCLib
Doc revision:	A
Written/Apr.:	RE / SL
Date:	10/03/2008

Industrial Control Design AS



WagoIPCLib V1.0

User Manual

The content of this document is confidential information not to be published without the consent of Industrial Control Design AS.

Industrial Control Design AS, www.icd.no, support@icd.no, forum.icd.no

Contents

1. INTRODUCTION.....	3
1.1. About.....	3
<hr/>	
2. CONFIGURATION.....	4
2.1. fs.....	4
2.2. IOConfig/Node.....	4
2.2.1. Description.....	4
2.2.2. Example XML.....	4
2.3. Inputs.....	4
2.3.1. Description.....	4
2.3.2. Example XML.....	4
2.3.3. Elements.....	5
2.4. Outputs.....	5
2.4.1. Description.....	5
2.4.2. Example XML.....	5
2.4.3. Elements.....	5
2.5. AnalogChannel Scaling.....	6
2.6. Alarms.....	6
2.7. Signals.....	6
2.8. Parameters.....	7
2.9. Autoconfiguration.....	7
2.10. Routing.....	7
2.11. Modify the project .xml files.....	7
<hr/>	
3. APPENDIX.....	9
3.1. Example WAGOIPCLIO XML-file.....	9

1. Introduction

1.1. About

This document describes how to set up and use the WAGOIPCIO with the CDP system. The WAGOIPCIO component has the following features:

- Works under On-Time RTOS32.
- Interrupt-driven PCI communication
- Converts CDP Signals to physical signals
- Converts Physical signals into CDP Signals
- Possibility to Auto-Detect module configuration and update the XML file to reflect the new configuration
- Has Online/Offline State for checking communication-problems.

2. Configuration

Configuration is done by modifying the component xml file inside the Application\Components\ folder. It should not be necessary to modify the model xml file. An example of WagoIPCIO.xml file (component xml) is found in 3.1.

2.1. fs

The `<fs>` element specifies the frequency of which to run the reading and writing of the Input/Output values. `<fs>100</fs>` means 100 Hz, so a read and write is done every 10 milliseconds. If the runningTooFast alarm is constantly set, then you could have specified a too high fs. Typically, the maximum fs for the Wago IPC is 200 Hz.

2.2. IOConfig/Node

2.2.1. Description

IOConfig is an XML element that wraps the Input / Output configuration. A minimal configuration is shown below.

2.2.2. Example XML

```
<IOConfig>
  <Node Name="WagoIPCIO">
    <Inputs>
    </Inputs>
    <Outputs>
    </Outputs>
  </Node>
</IOConfig>
```

2.3. Inputs

2.3.1. Description

Contains the ChannelGroups and Channels / signals to receive input from the physical Wago Input Modules.

2.3.2. Example XML

```
<Inputs>
  <ChannelGroup Type="Analog" NumberOf="2" ModuleNr="0">
    <Channel Nr="0" Type="short" Name="750-454 2AI 0.0"></Channel>
    <Channel Nr="1" Type="short" Name="750-454 2AI 0.1"></Channel>
  </ChannelGroup>
</Inputs>
```

2.3.3. Elements

Element	Description
ChannelGroup	An enclosing element for a group of channels/signals.
Channel	A Signal / Channel that correspond to one value received in.

--

ChannelGroup	Description
Type	Can be 'Analog' or 'Digital'
NumberOf	The Number of channels in this ChannelGroup
ModuleNr	The physical module number, used only for documentation.
SizeInBytes	Only for Digital ChannelGroups: Specifies the size in bytes of the ChannelGroup. For the Wago IPC this is always set to 2.

Channel Attribute	Description
Number	The number in a sequence, must be last number+1. On Digital channels this also signifies the bit position, starting at 0.
Type	The c++ data type, can be bool, char, byte, unsigned char, short, unsigned short, int, unsigned int, long, unsigned long, float or double. Typical values are bool and short.
Name	The signal name for this channel. Feel free to use more understandable names than what is used in the example, like 'Pressure', 'Oil Level' and so on.

2.4. Outputs

2.4.1. Description

Contains the ChannelGroups and Channels/signals to send as output from the Wago Modules.

2.4.2. Example XML

```

<Outputs>
  <ChannelGroup Type="Analog" NumberOf="2" ModuleNr="0">
    <Channel Nr="0" Type="short" Name="750-454 2AI 0.0"></Channel>
    <Channel Nr="1" Type="short" Name="750-454 2AI 0.1"></Channel>
  </ChannelGroup>
</Outputs>

```

2.4.3. Elements

Element	Description
ChannelGroup	An enclosing element for a group of channels/signals.
Channel	A Signal / Channel that correspond to one value received in.

ChannelGroup	Description
Type	Can be 'Analog' or 'Digital'
NumberOf	The Number of channels in this ChannelGroup
ModuleNr	The physical module number
SizeInBytes	Only for Digital ChannelGroups: Specifies the size in bytes of the ChannelGroup. For the Wago IPC this is always set to 2.

Channel Attribute	Description
Number	The number in a sequence, must be last number+1. On Digital channels this also signifies the bit position
Type	The c++ data type, can be bool, char, byte, unsigned char, short, unsigned short, int, unsigned int, long, unsigned long, float or double. Typical values are bool and short.
Name	The signal name for this channel.

2.5. AnalogChannel Scaling

Analog Channels can do multipoint scaling each time they are read or written. Please see the document 'AnalogChannel with Multipoint scaling.pdf' in the Doc folder where you installed CDP for more information about this. On a Windows install, there is also a shortcut placed on 'Start Menu'-'>CDP'-'>Doc'-'>IOServers'.

2.6. Alarms

The following alarms can trigger from this IOserver:

Alarm Name	Description
Transmission Error	An error is causing the transmission of signals to fail
RunningTooFast	The <fs> is set too high, the I/O can not keep up.
ModuleError	There is an error on one or more of the IO Nodes that causes the process image update to fail

In addition, you can set up alarms to trigger directly on a Channel mask.

A Channel Alarm is set up in XML like this:

```
<Channel Nr="0" Type="short" Name="750-454 2AI 0.0" ErrorMask="0x0003"
AlarmMessageCommand="Stop" AlarmMessageDestination="..ControlCode" AlarmText="Cable break on
Valve feedback" Description="A Cable break was detected on the first input module in the Winch
IO Cabinet (cable from winch speed valve feedback)"></Channel>
```

The Alarm will get the name of the channel, and " Alarm" is appended to that name. For this reason, make sure the Channel Name is less than 25 characters to avoid the problem with ShortName>31 characters.

Channel Alarm Attribute	Description
ErrorMask	Required. Specifies a Mask to AND (&) with the channel. If the result is non-zero, the alarm is set (see Timeout below)
AlarmMessageCommand	MessageCommand to send when alarm is set.
AlarmMessageDestination	MessageDestination for the AlarmMessageCommand.
AlarmText	Text to set in the Alarm, visible in a visualizer
AlarmDescription	Description to set in the alarm, visible in a visualizer
Timeout	Time in seconds that the error condition must be active, before the alarm is set. Default value 0.

2.7. Signals

The following signals are in the IOserver:

Signal Name	Description
Send-Receive Roundtrip time	The time used for outputting and inputting data, including signal conversion.

Signal Name	Description
ReadTimeStamp	The globalTime when last read happened
WriteTimeStamp	The globalTime when last write happened
OutputDisabled	Only used in conjunction with CDP Redundancy: Is set to 1 if parameter 'RD output disable control' is set to 1 and the RDmanager is not in the Active state, else it is 0. This signals tells if the outputs are written out to the physical modules(if value=0) or not (value=1).

2.8. Parameters

Parameter Name	Description
doReadFirst	Set to 1 to perform reads before writes.
RD output disable control	Only used in conjunction with CDP Redundancy: Set to 1 to make RDManager disable output if RDManager is not in Active State.
AutoConfig (see below)	Set to 1 to Automatically adjust the .xml file to reflect the actual layout on a change in configuration, or when the controller boots up. Please note that all the xml for physical modules is regenerated when AutoConfig is set to 1, causing you to loose existing settings.

2.9. Autoconfiguration

AutoConfiguration can be enabled by setting parameter AutoConfig to 1. When adding a new Module to the Wago IPC, or when removing one, the XML is then updated to reflect the change. This is the preferred method of setting up the WagoIPCIO the first time. After configuring, you can set AutoConfig="0" and then modify the names of the channels to suit your needs.

2.10. Routing

To get the IOserver to output sensible signals, you will have to route the signals from another component. In CDP version 2.3.1.0 and earlier you can **not** set routing directly on an IOserver such as this. You will have to 'push' the routing from a signal on the same controller as the IOserver is located. 'Push' routing only works on components running inside the same controller. For instance, a 'Sinus' component could have routing on its 'Output' signal set to 'WAGOIPCIO.Analog Output 1', this will effectively 'write' the sinus out to the Module containing that signal.

2.11. Modify the project .xml files

Add the following to your project's Application.xml file:

Inside the <Components> element, add an instance of a WagoIPCIO component, for instance:

```
<Component Name="WAGOIPCIO" src="Components/WAGOIPCIO.xml"></Component>
```

This will make the component a so-called 'top-level' component, quickly accessible from anywhere in the network. In a CDPBrowser, the component will then be visible on the same level as other Application components.

If you want to have the component 'hidden' inside your application, add the following inside the <Subcomponents> element:

```
<Subcomponent Name="WAGOIPCIO" Model="WAGOIPCIO"  
src="Components/WAGOIPCIO.xml"></Subcomponent>
```

This will tell CDP to initialize a component named “WAGOIPCIO” from a component file located at “Components/ WAGOIPCIO.xml”, and place it below the Application component.

Make sure that your Models\ folder contains a WAGOIPCIO.xml model file, or the component will not be initialized correctly.

The WAGOIPCIO Model file can be found in \$(CDPBase)\Templates\Project_template\Application\Models\
and an example WAGOIPCIO component file can be found in \$(CDPBase)\Templates\XML Templates\IO Servers.

3. Appendix

3.1. Example WAGOIPCI0 XML-file

```

<?xml version="1.0" encoding="ISO-8859-1" ?>

<!-- WagoIPCI0 component model. -->
<Component Name="WagoIPCI0" Type="WagoIPCI0">
  <!-- The frequency that data-access should be performed at -->
  <fs>150</fs>
  <Activate>1</Activate>
  <Description>
  <![CDATA[ Wago IPC IO driver for On-Time RTOS 32 ]]>
  </Description>
  <IOConfig>
    <Node Name="WagoIPCI0">
      <!-- No need for 'Packet' here, only one is supported -->
      <Inputs>
        <ChannelGroup Type="Analog" NumberOf="2" ModuleNr="0">
          <Channel Nr="0" Type="short" Name="Analog Input 1">
            <Scaling>
              <Point HardwareValue="0" EngineeringValue="0.000000"></Point>
              <Point HardwareValue="32767" EngineeringValue="1.000000"></Point>
            </Scaling>
          </Channel>
          <Channel Nr="1" Type="short" Name="Analog Input 2">
            <Scaling>
              <Point HardwareValue="0" EngineeringValue="0.000000"></Point>
              <Point HardwareValue="32767" EngineeringValue="1.000000"></Point>
            </Scaling>
          </Channel>
        </ChannelGroup>
        <ChannelGroup Type="Digital" NumberOf="4" ModuleNr="1" SizeInBytes="2">
          <Channel Nr="0" Type="bool" Name="4DI 1.0"></Channel>
          <Channel Nr="1" Type="bool" Name="4DI 1.1"></Channel>
          <Channel Nr="2" Type="bool" Name="4DI 1.2"></Channel>
          <Channel Nr="3" Type="bool" Name="4DI 1.3"></Channel>
        </ChannelGroup>
      </Inputs>
      <Outputs>
        <ChannelGroup Type="Analog" NumberOf="2" ModuleNr="2">
          <Channel Nr="0" Type="short" Name="Analog Output 1">
            <Scaling>
              <Point HardwareValue="0" EngineeringValue="0.000000"></Point>
              <Point HardwareValue="32767" EngineeringValue="1.000000"></Point>
            </Scaling>
          </Channel>
          <Channel Nr="1" Type="short" Name="Analog Output 2">
            <Scaling>
              <Point HardwareValue="0" EngineeringValue="0.000000"></Point>
              <Point HardwareValue="32767" EngineeringValue="1.000000"></Point>
            </Scaling>
          </Channel>
        </ChannelGroup>
        <ChannelGroup Type="Digital" NumberOf="2" ModuleNr="3" SizeInBytes="2">
          <Channel Nr="0" Type="bool" Name="2DO 3.0"></Channel>
          <Channel Nr="1" Type="bool" Name="2DO 3.1"></Channel>
        </ChannelGroup>
      </Outputs>
    </Node>
  </IOConfig>
  <Alarms>
    <Alarm Name="Transmission Error" Text="ModbusTCPIOserver transmission-error alarm"
    Level="Warning" Enabled="1"></Alarm>
  </Alarms>

```

```

<Alarm Name="RunningTooFast" Group="" Level="Warning" Trig="1" Enabled="1"
EnabledState="Online" Signal="" Inverted="0" SignalOutSet="" Text="The I/O frequency is set
too high, the hardware can not keep up." Description="The I/O frequency is set too high, the
hardware can not keep up."></Alarm>
<Alarm Name="ModuleError" Group="" Level="Error" Trig="0" Enabled="1" EnabledState=""
Signal="" Inverted="0" SignalOutSet="" Text="There is an error on one or more of the IO Nodes
that causes the process image update to fail." Description="There is an error on one or more
of the IO Nodes that causes the process image update to fail."></Alarm>
</Alarms>
<Signals>
<Signal Name="Send-Receive Roundtrip time" Unit="s" Type="double" Description="The time
needed to perform one send and receive."></Signal>
<Signal Name="ReadTimeStamp" Input="0" Type="double" Unit="s" Description="Timestamp for
when data was read"></Signal>
<Signal Name="WriteTimeStamp" Input="0" Type="double" Unit="s" Description="Timestamp for
when data was written"></Signal>
</Signals>
<Parameters>
<Parma Name="doReadFirst" Unit="0 / 1" Value="1" DefaultValue="0" PreviousValue="0"
TimeLastChanged="Fri Oct 03 11:21:50 2008" Description="Set to 1 to have I/O do Read, then
Write. If not, then it is Write, then Read."></Parma>
<Parma Name="AutoConfig" Value="0" DefaultValue="0" PreviousValue="1" TimeLastChanged="Fri
Oct 03 10:44:25 2008" Description="Set to 1 to enable AutoConfiguration and updating of XML."
Unit=""></Parma>
</Parameters>
</Component>

```