

The Siemens logo is displayed in a white rectangular box in the top right corner of the image. The background of the entire image is a blurred industrial factory setting with a man in a light blue shirt looking at a tablet. Overlaid on the scene are various digital and data-related icons, including a '24/7' circular arrow, a 'NEWS' box with a person icon, a 'Home' button, and a network diagram with nodes and binary code (0s and 1s).

**SIEMENS**

# Getting process values via CIP with SIMATIC S7

S7-1500/ S7-1200/ EtherNet/IP

<https://support.industry.siemens.com/cs/ww/en/view/109782317>

Siemens  
Industry  
Online  
Support



# Legal information

## Use of application examples

Application examples illustrate the solution of automation tasks through an interaction of several components in the form of text, graphics and/or software modules. The application examples are a free service by Siemens AG and/or a subsidiary of Siemens AG ("Siemens"). They are non-binding and make no claim to completeness or functionality regarding configuration and equipment. The application examples merely offer help with typical tasks; they do not constitute customer-specific solutions. You yourself are responsible for the proper and safe operation of the products in accordance with applicable regulations and must also check the function of the respective application example and customize it for your system.

Siemens grants you the non-exclusive, non-sublicensable and non-transferable right to have the application examples used by technically trained personnel. Any change to the application examples is your responsibility. Sharing the application examples with third parties or copying the application examples or excerpts thereof is permitted only in combination with your own products. The application examples are not required to undergo the customary tests and quality inspections of a chargeable product; they may have functional and performance defects as well as errors. It is your responsibility to use them in such a manner that any malfunctions that may occur do not result in property damage or injury to persons.

## Disclaimer of liability

Siemens shall not assume any liability, for any legal reason whatsoever, including, without limitation, liability for the usability, availability, completeness and freedom from defects of the application examples as well as for related information, configuration and performance data and any damage caused thereby. This shall not apply in cases of mandatory liability, for example under the German Product Liability Act, or in cases of intent, gross negligence, or culpable loss of life, bodily injury or damage to health, non-compliance with a guarantee, fraudulent non-disclosure of a defect, or culpable breach of material contractual obligations. Claims for damages arising from a breach of material contractual obligations shall however be limited to the foreseeable damage typical of the type of agreement, unless liability arises from intent or gross negligence or is based on loss of life, bodily injury or damage to health. The foregoing provisions do not imply any change in the burden of proof to your detriment. You shall indemnify Siemens against existing or future claims of third parties in this connection except where Siemens is mandatorily liable.

By using the application examples you acknowledge that Siemens cannot be held liable for any damage beyond the liability provisions described.

## Other information

Siemens reserves the right to make changes to the application examples at any time without notice. In case of discrepancies between the suggestions in the application examples and other Siemens publications such as catalogs, the content of the other documentation shall have precedence.

The Siemens terms of use (<https://support.industry.siemens.com>) shall also apply.

## Security information

Siemens provides products and solutions with Industrial Security functions that support the secure operation of plants, systems, machines and networks.

In order to protect plants, systems, machines and networks against cyber threats, it is necessary to implement – and continuously maintain – a holistic, state-of-the-art industrial security concept. Siemens' products and solutions constitute one element of such a concept.

Customers are responsible for preventing unauthorized access to their plants, systems, machines and networks. Such systems, machines and components should only be connected to an enterprise network or the Internet if and to the extent such a connection is necessary and only when appropriate security measures (e.g. firewalls and/or network segmentation) are in place.

For additional information on industrial security measures that may be implemented, please visit <https://www.siemens.com/industrialsecurity>.

Siemens' products and solutions undergo continuous development to make them more secure. Siemens strongly recommends that product updates are applied as soon as they are available and that the latest product versions are used. Use of product versions that are no longer supported, and failure to apply the latest updates may increase customer's exposure to cyber threats.

To stay informed about product updates, subscribe to the Siemens Industrial Security RSS Feed at: <https://www.siemens.com/industrialsecurity>.

# Table of contents

<b>Legal information</b> .....	<b>2</b>
<b>1 Preface</b> .....	<b>4</b>
<b>2 Introduction</b> .....	<b>6</b>
2.1 Description .....	6
2.2 Common Industrial Protocol .....	6
2.3 Programmable Controller Communication Commands .....	7
2.4 Message Router .....	7
2.5 Description .....	8
2.6 Function principle .....	8
2.7 Scope of delivery .....	9
2.8 What is new? .....	9
<b>3 Commissioning</b> .....	<b>10</b>
3.1 Preparation .....	10
3.2 Connecting the hardware components .....	12
<b>4 Configuration/Engineering</b> .....	<b>13</b>
4.1 Creating and managing projects .....	13
4.2 Creating the tag lists .....	16
4.2.1 Manual tag list creation .....	16
4.2.2 Tool based tag list creation .....	18
<b>5 Operating</b> .....	<b>23</b>
5.1 Start the application .....	23
5.2 Troubleshooting .....	26
5.2.1 Physical check .....	26
5.2.2 Network settings .....	27
5.2.3 SIMATIC Program .....	28
<b>6 LCCF_CipClient block</b> .....	<b>29</b>
6.1 Parameters .....	29
6.1.1 Block status messages .....	30
6.1.2 Technical data .....	32
6.2 What is next? .....	33
<b>7 Appendix</b> .....	<b>34</b>
7.1 CIP/ PCCC Communication path settings .....	34
7.1.1 Case 1 – Accessing data inside an Allen- Bradley PLC .....	35
7.1.2 Case 2 – Accessing the Allen- Bradley PLC via integrated Ethernet port .....	37
7.1.3 Case 3 – Accessing the Allen- Bradley PLC via the ControlNet interface .....	38
7.2 Service and support .....	39
7.3 Related literature .....	40
7.4 Change documentation .....	40

# 1 Preface

## Purpose

This document contains information about the “LCCF\_CipClient” function block for SIMATIC S7-1200 and S7-1500. The previously available blocks “PUT\_R” and “GET\_R” are being replaced with this single block, which is part of the LCCF (Library of Competitor Conversion Functions).

As with the beforementioned blocks the “LCCF\_CipClient” allows easy access to process values inside a supported Rockwell Automation control system. It is using the control systems native protocol preventing any changes inside the already running program.

For this matter, the block is ideally used in scenarios, where the control system program cannot be changed for the extension with a SIMATIC system.

## Core content

This document explains the use of the “LCCF\_CipClient” function block for SIMATIC to read or write process values inside the Rockwell Automation controller. The only necessary knowledge is, that the user can program the SIMATIC controller.

No changes inside the Rockwell Automation controller are necessary.

The following core issues are covered in this document:

- Purpose of the function block
- Parameterization
- Data Exchange with a CIP server

## Required basic knowledge

General knowledge in communications over Ethernet and programming and configuring the S7-1200 or S7-1500 with the TIA Portal is assumed and will not be part of this document. It is also assumed that the terms Server and Client and their meaning are familiar to the reader.

## Delimitation

The document does not describe:

- How to setup Ethernet networks
- How to assign IP addresses and the split into subnets
- How to configure the controllers in this example

Basic knowledge about the above topics is assumed.

## Validity

This document is valid for the following components

- TIA Portal
- SIMATIC S7 Controller

The block has been tested with the following Rockwell Automation systems

- ControlLogix/ CompactLogix
- SLC 551
- Micro850

The following hardware and software are used throughout this document

Table 1-1: used components

Name	Part number	Version
SIMATIC S7-1511-1PN	6ES7 511-1AK01-0AB0	V2.6 (or above)
SIMATIC S7-1215C	6ES7 215-1AG40-0XB0	V4.2 (or above)
SIMATIC S7-1505S	6ES7 672-5AC00-0YA0	V2.1 (or above)
TIA Portal STEP7 Prof.		V15.1 Update 4 (or above)
SCALANCE X208	6GK5 208-0BA10-2AA3	
CompactLogix	1769-L27ERM-QBFC1B	V33.13
Studio 5000	9324-RLD700NXENE	V33.13

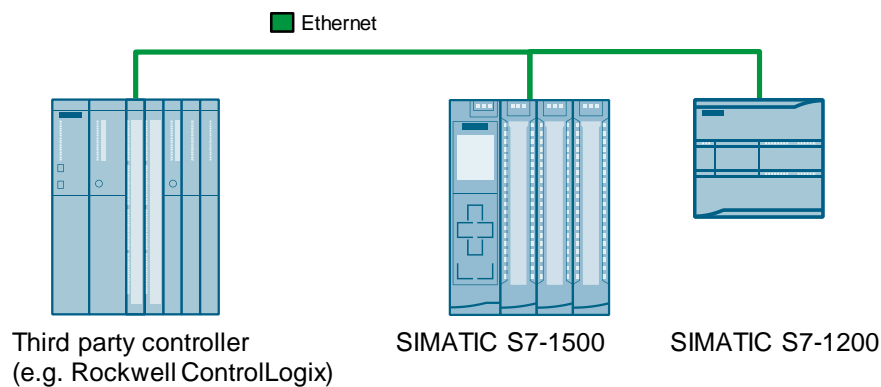
## 2 Introduction

### 2.1 Description

Often SIMATIC controller exchange data with other controller, peripheral systems, and upper-level control systems or with MES or SCADA systems. To do this in a meaningful manner the communication partners need to use the same protocol. The following application example demonstrates the use of CIP/ PCCC as the protocol to perform this data exchange.

A simplified setup is shown below

Figure 2-1: simplified setup



The physical connection between the controller uses Ethernet cables, such as PROFINET cable. A SCALANCE X208 switch has been put in between, but is not shown, as direct connections are possible as well.

As the “LCCF\_CipClient” block exists for both SIMATIC families, this application example is applicable to S7-1200 and S7-1500 systems. In the above schematic only one of the SIMATIC controller is necessary. As a CIP server may be able to serve multiple clients, multiple different SIMATIC controller may be connected to the CIP server. The number of connectable clients depends on the capabilities of the CIP server

### 2.2 Common Industrial Protocol

The Common Industrial Protocol (for short CIP) is defined by the Open DeviceNet Vendor Association (ODVA®). This protocol is an object-oriented mechanism to access process values as well as configuration and diagnostic data of a conformant device.

CIP defines several objects, which in turn provide services to access the attributes of that object. Each attribute has a certain meaning, which is either predefined or vendor specific.

The “LCCF\_CipClient” function block uses two different objects depending on the configured CIP server.

One object is the PCCC object. The LCCF\_CipClient implements a few of its services.

The other object is the Message Router (MR) object. The LCCF\_CipClient block implements also only a few defined services.

Each such service execution is realized as a “Request – Response” pair.

## 2.3 Programmable Controller Communication Commands

The Programmable Controller Communication Commands (for short PCCC, PC<sup>3</sup> or PC cube) are a standardized set of services defined in the PCCC object. These services allow the access to process values inside the automation system implementing this object.

The SIMATIC implementation provided in the “LCCF\_CipClient” function block implements two services of the PCCC object.

- SLC typed protected logical read with 3 address fields
- SLC typed protected logical write with 3 address fields

These two services are sufficient to access a wide range of automation system internal memories. There are Rockwell controller families supporting the PCCC object directly. These families are:

- SLC 5/0x programmed with RS Logix 500 therefore, known as SLC500
- MicroLogix 1x00 programmed with RS Logix 500

Both families of Rockwell Automation PLC have the same memory addressing schema. Variables are collected in so called files of common data type. This means, that variables of the same data type are stored in a file of that data type.

### NOTE

Another legacy Rockwell Automation system is the PLC-5. It also supports the PCCC object and has the same addressing schema as the SLC and MicroLogix. The PLC-5 uses different commands to provide access to the object's attributes. These services are:

- PLC-5 typed read
- PLC-5 typed write
- PLC-5 typed masked write

These services come with 3,4 or even 7 address fields.

This Rockwell Automation system is not in scope of the LCCF\_CipClient function block.

## 2.4 Message Router

The Message Router object defines services, which allow the access to process variables inside a different family of Rockwell controller.

The SIMATIC implementation provided in the “LCCF\_CipClient” function block implements two services of the Message Router object.

- CIP\_DataTableRead
- CIP\_DataTableWrite

These services are supported by the following Rockwell Automation systems:

- CompactLogix
- ControlLogix

and their respective Safety counterparts called GuardLogix.

## 2.5 Description

As with the “PUT\_R” and “GET\_R” blocks the “LCCF\_CipClient” block allows an easy access to Rockwell Automation control systems (a.k.a. Allen Bradley). It can read and write process values inside the controllers without any changes necessary inside the controllers.

The “LCCF\_CipClient” acts as a client requesting the services from the controller. It uses the controller native CIP (Common Industrial Protocol) encapsulated on Ethernet.

This document will show you how easy it is to achieve the process value access. What needs to be configured and what can be achieved.

## 2.6 Function principle

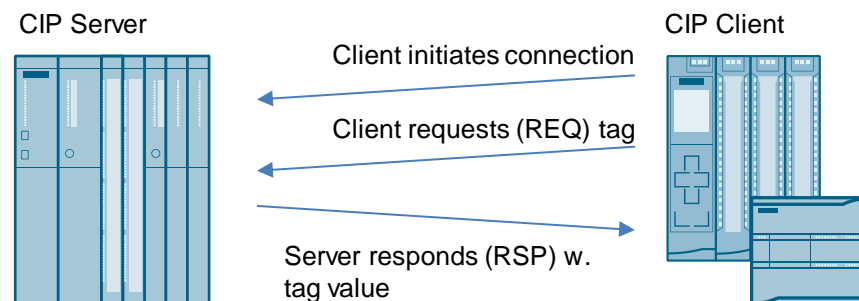
The application example demonstrates the necessary steps to set up the communication and to start the data exchange between a SIMATIC controller and a third-party system.

The communication with CIP/ PCCC uses the Client-Server principle. In this application the CIP server is being realized by the Rockwell ControlLogix controller. The CIP client is realized by either a SIMATIC S7-1500 or a SIMATIC S7-1200. Only one of the SIMATIC systems is necessary, but both or even multiple can be operated at the same time.

The CIP protocol is embedded into the payload of the EtherNet/IP packets sent to the Rockwell Automation system.

The CIP client establishes the connection and requests data from the CIP server to use in its own automation program. The LCCF\_CipClient block also allows writing to process values inside the Rockwell controller.

Figure 2-2: schematic functional principle



With the “LCCF\_CipClient” function block a user can establish a connection to a directly accessible Rockwell Automation controller system. Once enabled the function block will automatically contact the Rockwell Automation controller. It establishes a TCP/IP connection and negotiates a communication session.

Within the session context all parameterized process tags will be exchanged with the Rockwell Automation controller. Hereby the “LCCF\_CipClient” block executes both Reading and Writing requests within a single request.

For this the CIP client utilizes one TCP connection.



### Advantages

This application offers the following advantages:

- Direct connection between controllers of different vendors
- No gateways or protocol converters required
- Modern and easy to install network

## 2.7 Scope of delivery

The application example consists of a TIA Portal project containing two SIMATIC controller.

The program in the S7-1200 is identical to the program in the S7-1500 controller.

## 2.8 What is new?

As mentioned previously, the provided block replaces the previous blocks "PUT\_R" and "GET\_R". The "LCCF\_CipClient" block has several advantages:

- Auto allocation of Connection ID
- Single block for both reading and writing of tags
- Access to structure and array elements
- Reduction in communication load
- Configuration Tool with direct import into TIA Portal project


Unfortunately, the above advantages come with a disadvantage:

- access to memory areas, larger than 64 Bit (LWORD) is no longer possible

## 3 Commissioning

### 3.1 Preparation

As preparation for the application example to work the above-mentioned hardware components should be placed into a rack or on a solid table to prevent slip or fall.

 <b>WARNING</b>	<p><b>Risk of electric shock</b></p> <p>To operate this application example the connection of the above hardware to electrical power is required. Disregarding local regulations and common sense may cause an electric shock and because of that injury or death.</p> <p>Always follow the rules for working with electrical equipment.</p>
---	--

Further download the TIA Portal project from the SIOS entry or download the latest version of the LCCF (Library of Competitor Conversion Functions). Store the downloaded file onto your local computer for later use.

#### Hardware components

In the following description we will be using the below listed hardware. In your scenario you may exchange one or the other component with an equivalent.

Table 3-1: recommended hardware components

Component	No.	Order number	Note
SIMATIC S7-1500	1	6ES7 511-1AK01-0AB0	as of firmware V2.6 onwards
SIMATIC S7-1215C	1	6ES7 215-1AG40-0AA0	as of firmware V4.2 onwards
SIMATIC S7-150xS	1	6ES7 672-xAC00-0YA0	as of firmware V2.1 onwards
ControlLogix L72S	1	1756-5572S	only one of these systems or equivalent systems required
CompactLogix	1	1769-L27ERM-QBFC1B	
SLC500	1	1747-L551	

**NOTE** Even though the used ControlLogix is a GuardLogix Safety rated controller, the "LCCF\_CipClient" cannot access Safety Tags. Therefore, the GuardLogix is operated as regular ControlLogix.

#### Software components

In this document the below listed software components are being used.

Table 3-2: recommended software components

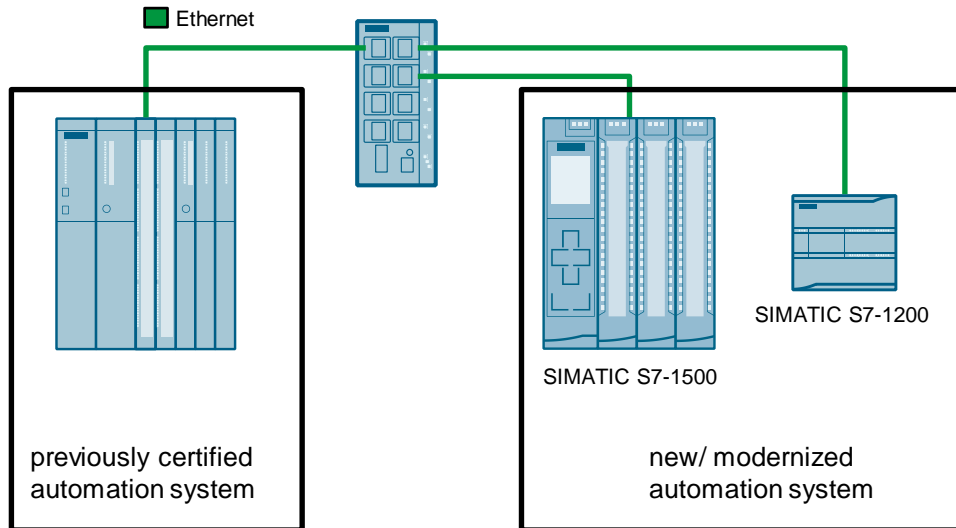
Component	No.	Order number	Note
TIA Portal V15.1 (incl. Openness)	1		used for programming the SIMATIC controller
Studio 5000 V33.13	1	9324-RLD700NXENE	used to define the tags inside the Rockwell ControlLogix or CompactLogix systems
RSLogix 500	1	9324-RL0300DEM	used to define data files inside the legacy SLC 500 or MicroLogix system

As an alternative to TIA Portal in V15.1 also V16, and V17 can be used. The sample projects and the libraries exist for all the above TIA Portal versions.

## 3.2 Connecting the hardware components

A possible general setup is shown below.

Figure 3-1: simplified network setup



### Setup

The communication is realized using Ethernet as the communication network. A direct connection using a cross over CAT5 cable or via a network switch with straight CAT5 cables can be used.

As already mentioned before, only one of the SIMATIC controller is necessary, but both can be operated simultaneously as well.

## 4 Configuration/Engineering

### 4.1 Creating and managing projects

Before we can parameterize the “LCCF\_CipClient” block and configure the tag list, we need to get a list of process values (tags) which we want to access.

This can be achieved either by getting a printout of the tag list, which is in question or browsing the controller to access.

The relevant information is slightly different for the currently supported Rockwell Automation systems.

In general, the different data types are used for certain things

Table 4-1: data type usage

to store a ...	... use this data type	
Bit	BOOL	
Bit array	BOOLEAN ARRAY (32 Bit)	
8 Bit Integer	SINT	
16 Bit Integer	INT	
32 Bit Integer	DINT	
32 Bit Float	REAL	
32 Bit Milliseconds	TIMER	Timer is a structure. Only individual elements can be addressed
32 Bit of range	COUNTER	Counter is a structure. Only individual elements can be addressed

Examples are shown in the following sub chapters on the following page.

## ControlLogix/ CompactLogix

In the current systems of ControlLogix or CompactLogix the most important information is the name of the process value.

Addressable process values are shown as examples in the below table.

Table 4-2: Logix5000 data

To access ...	... specify the tag name	Example
single integer tag named „parts“	tagname	“parts”
6 <sup>th</sup> element of an array of REALs named “setpoints”	tagname[index]	“setpoints[6]”
single integer [2,5,257] of a three-dimensional array named “profile”	tagname[index, index, index]	“profile[2,5,257]”
accumulated value of a timer named “dwell3”	tagname.ACC	“dwell3.ACC”
wear member inside the singular STRUCT_A structure “struct1”	tagname.structure element name	“struct1.wear”
5 <sup>th</sup> “hourlyCount” element of a single STRUCT_B structure “struct2”	tagname.structure element name[index]	“struct2.hourlyCount[5]”

### NOTE

Casing of the tag name to access in a ControlLogix or CompactLogix system is important. The following tags are not the same.

- “parts” <> “Parts”
- “parts” <> “PARTS”
- “parts” <> “pArts”

Make sure the spelling is correct, otherwise the tag cannot be accessed.

Misspelling can be avoided, when using the CIP service configuration tool.

## Micro800

Another current control system designed for micro automation is the Micro800 series controller. Tags are addressable in the same way as in the above mentioned CompactLogix and ControlLogix systems. They are referenced by their name.

As the Micro800 has a different addressing schema, the parameter “slot” needs to be set to 127 (fix). This value indicates the changed addressing schema.

### NOTE

As with the ControlLogix and CompactLogix systems the spelling of the tag's name is important.

## SLC – Small Logic Controller

The below shown addressing examples are applicable to the SLC family of controllers. They are valid for the MicroLogix family of controllers as well.

The tag name is identical with the addressed memory location extended by the prefix '@'.

Table 4-3 SLC/MicroLogix data

To access ...	... specify the tag name	Example
The 1 <sup>st</sup> single integer tag in N file 7	@filename filename:element number	"@N7:0"
The 3 <sup>rd</sup> bit of the binary file B3 in word 4	@filename filename:element number/bitnumber	"@B3:4/3"
the 9 <sup>th</sup> element of the REAL file F80	@filename filename:element number	"@F80:9"
the Accumulated value of the 3 <sup>rd</sup> counter in file C5	@filename filename:element number. <b>ACC</b>	"@C5:3.ACC"
the Preset time value of the 6 <sup>th</sup> timer in file T4	@filename filename:element number. <b>PRE</b>	"@T4:6.PRE"

### NOTE

The legacy Rockwell Automation system PLC-5 uses the same syntax as the SLC and MicroLogix with a different prefix. This allows the LCCF\_CipClient function block to distinguish between SLC/MicroLogix services and PLC-5 services to invoke.

## 4.2 Creating the tag lists

As mentioned, there are two possible ways to create a list of tags, which the “LCCF\_CipClient” block should access.

As prerequisite a TIA Portal project containing the PLC of your choice should already exist.

In this document this TIA Portal project is called “CIPClient” and uses TIA Portal V15.1.

### 4.2.1 Manual tag list creation

When you decide to create the list of tags manually, you need to take extra care for tags, which should be written. In this case it is important to know the data type code. The codes are listed below.

Table 4-4: Type codes for data types

Data type	type code	Note
BOOL	16#00C1	a Boolean value
SINT	16#00C2	8 bit signed integer
INT	16#00C3	16 bit signed integer
DINT	16#00C4	32 bit signed integer
REAL	16#00CA	32 bit single precision floating point number
DWORD	16#00D3	32 bit collection of bit

Each tag is stored in a variable of type “LCCF\_typeTagDef”. The tags should be stored in an array of any length. The type is provided as part of the “LCCF”.

A data block containing some tag definitions is shown below. Here an array of 10 tygs has been defined. For illustration purposes two of the defined tags are shown in detail.

- MyTags[2] is called “MovingBits”
- MyTags[3] is called “MoveLeft” and is of type BOOL with type code 16#00C1 (refer to the above table)

Figure 4-1: TIA Portal Datablock with tag definitions

The screenshot shows a table with columns: Name, Data type, Start value, Retain, Accessible f..., Writ..., Visible in..., Setpoint, Supervis..., and Comment. It lists definitions for MyTags[0], MyTags[1], MyTags[2], and MyTags[3].

Name	Data type	Start value	Retain	Accessible f...	Writ...	Visible in...	Setpoint	Supervis...	Comment
Static									
MyTags	Array[0..9] o...								
MyTags[0]	*typeTagDef								
MyTags[1]	*typeTagDef								
MyTags[2]	*typeTagDef								
MyTags[3]	*typeTagDef								

The used type is shown in detail below.



Figure 4-2: TIA Portal data type “LCCF\_typeTagDef”

	Name	Data type	Default value	Accessible f...	Writa...	Visible in ...	Setpoint	Comment
1	tagName	String[4...]	''	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	contains the tag name to access
2	tagType	Word	16#0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	data type code of the tag to be read/ written
3	quality	Char	'B'	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	'G' indicates 'Good'; 'B' or anything else means 'Bad'
4	writing	Bool	false	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	TRUE indicates writing; FALSE indicates reading
5	value	DWord	16#0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	contains the value of the tag stored in 32-Bit

As you can see, the type has several members which will be explained in the following table.

**NOTE**

The access direction in the below table is seen from the LCCF\_CipClient block’s viewpoint.

This means that the direction “read” indicates a reading access from the block. The block reads this value and acts accordingly.

The value “write” means that the block disregards any value in here and overwrites it with the value it deems fit.

An indication of “read/write” means that, depending on other field values, this fields value is either read or written.

Table 4-5: type “LCCF\_typeTagDef”

Member	data type	access direction	comment
tagName	STRING[40]	read	Contains the name of the process value to access. <ul style="list-style-type: none"> <li>Use the prefix '@' for access to SLC/ MicroLogix systems.</li> <li>Use the tag name to access ControlLogix or CompactLogix systems</li> </ul>
tagType	WORD	read/ write	contains the type-code for the process value accessed. This is relevant for writing access. Refer to the table “Type codes for data types”
quality	CHAR	write	contains a quality code for the value provided in the member ‘value’. ‘G’ means ‘GOOD’, the value is valid and can be used for further processing. ‘B’ means ‘BAD’, the value is not valid and shall not be used for further processing.
writing	BOOL	read	TRUE indicates a writing access, which will transfer the value provided in the member ‘value’ to the Rockwell Automation system. FALSE indicates a reading access, which will store the value from the Rockwell Automation system into the member ‘value’.

Member	data type	access direction	comment
value	DWORD	read/ write	In this member the value is stored, which is either retrieved from (writing = FALSE) the Rockwell Automation system or which is being written to (writing = TRUE) the Rockwell Automation system.

**NOTE**

When you keep the 'writing' member at its default 'FALSE' the first access is a 'Reading' access, which also populates the type-code into 'tagType'.


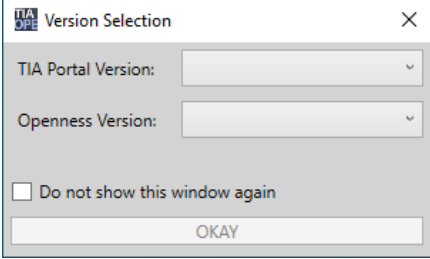
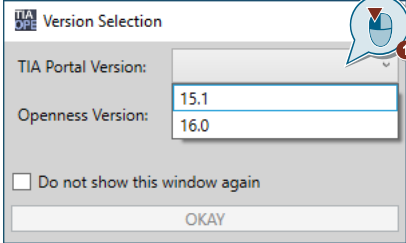
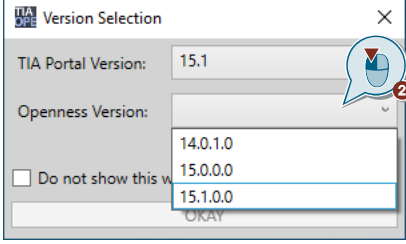
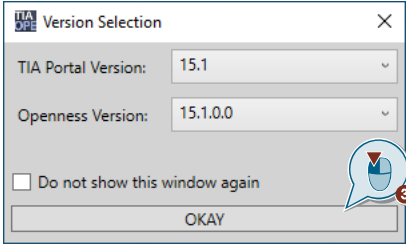
You may change the 'writing' member's value during operation at any time via your program. This will freeze the value on the Rockwell Automation side as the same value is being written if the 'writing' member remains 'TRUE'.

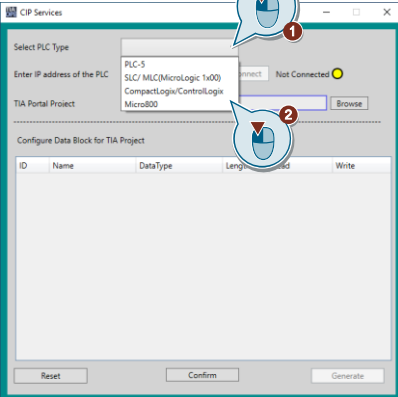
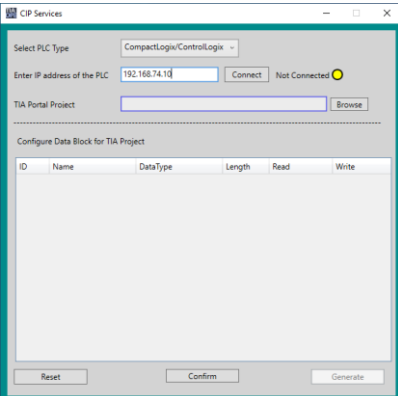
**4.2.2 Tool based tag list creation**

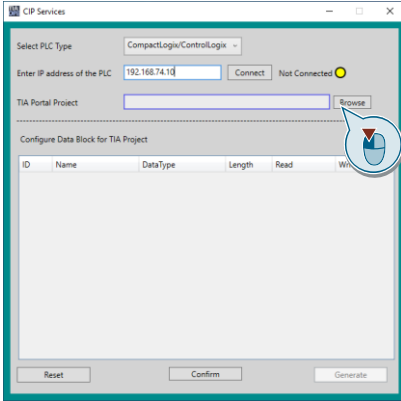
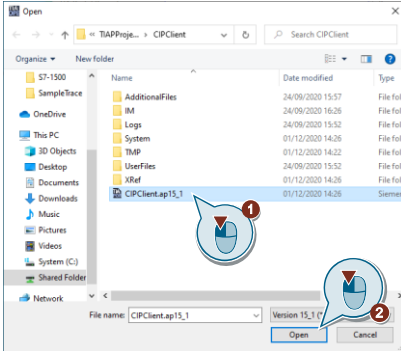
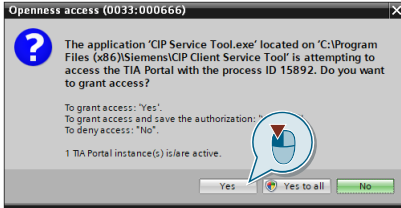
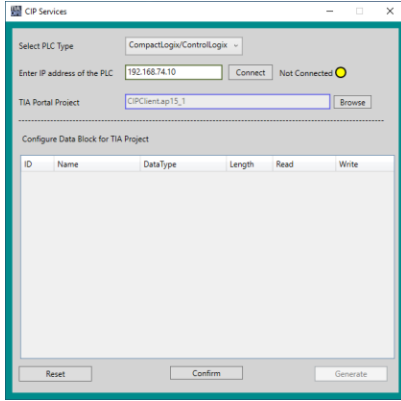
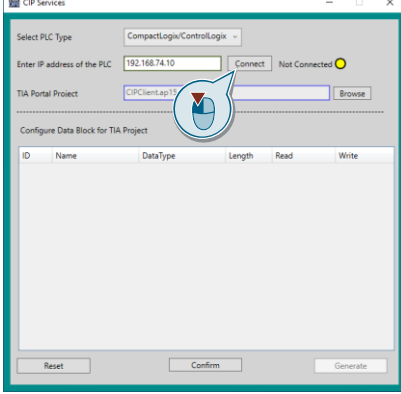
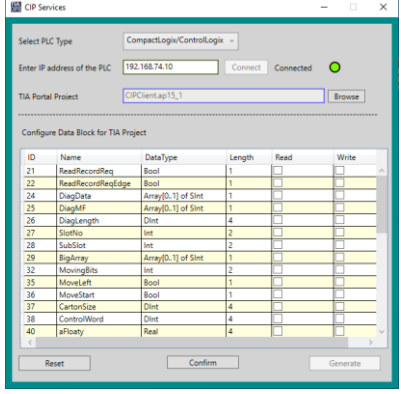
For your convenience we created a tool, which can create a SIMATIC data block with selected tag definitions in it. It also has the capability to import the data block directly into a TIA Portal project. Its use is explained on the following pages.

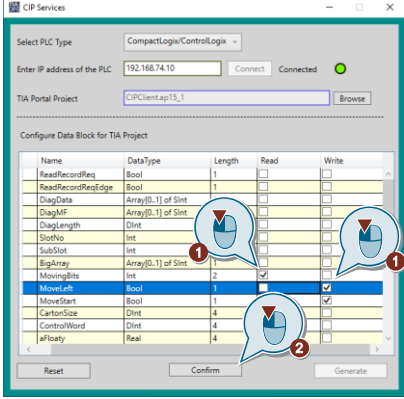
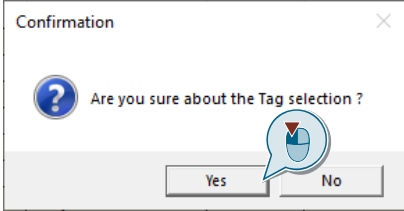
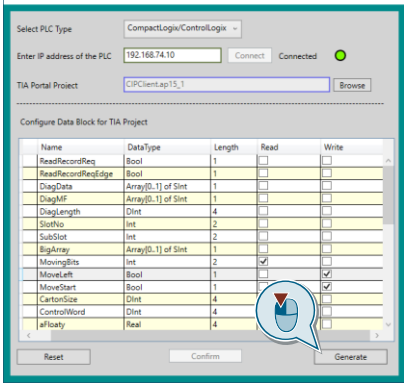
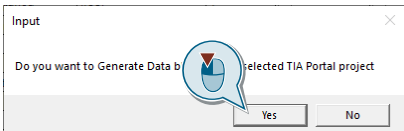
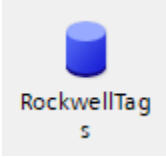
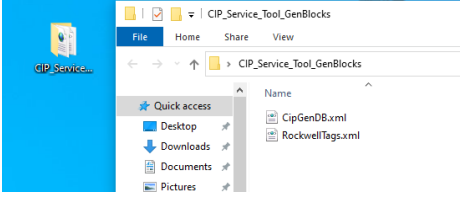
The tool uses TIA Portal Openness for the import feature and can browse into Rockwell Automation controllers, which are directly connected to the computer running the "CIP Service Tool"

Table 4-6 TagList

	What to do	Result
1.	<p>start the CIP Service Tool</p> 	<p>A dialog box pops up allowing the selection of the TIA Portal version to be used.</p> 
2.	<p>Select the TIA Portal version</p>  <p>and the Openness version you want to use.</p>  <p>finally confirm your selection with OKAY</p> 	<p>here we select TIA Portal V15.1, but TIA Portal V16 works the same way.</p>

	What to do	Result
3.	<p>As next we select the Rockwell Automation controller family from the drop down list on top.</p> 	<p>here we select the ControlLogix/ CompactLogix controller family.</p>
4.	<p>Enter the IPv4 address the Rockwell Automation controller is accessible on.</p> 	<p>you need to adjust the here selected IPv4 address to yours. Make sure your engineering system computer can access the controller.</p>

	What to do	Result																																																																																										
<p>5.</p>	<p>Browse for the TIA Portal project you want to perform the import.</p>  <p>Select the project from the File Open dialog</p>  <p>You may have to allow access to TIA Portal through the firewall</p> 	<p>you are now ready to generate a data block containing the tag definitions.</p> 																																																																																										
<p>6.</p>	<p>Click on “Connect” to establish a connection to the Rockwell Automation system and browse the available tags.</p> 	<p>The list of tags in the center is now populated and allows the selection of tags to read and/ or to write to.</p>  <table border="1" data-bbox="922 1688 1289 1935"> <thead> <tr> <th>ID</th> <th>Name</th> <th>DataType</th> <th>Length</th> <th>Read</th> <th>Write</th> </tr> </thead> <tbody> <tr><td>21</td><td>ReadResponseFlags</td><td>Bool</td><td>1</td><td><input type="checkbox"/></td><td><input type="checkbox"/></td></tr> <tr><td>22</td><td>ReadResponseEdge</td><td>Bool</td><td>1</td><td><input type="checkbox"/></td><td><input type="checkbox"/></td></tr> <tr><td>24</td><td>DiagData</td><td>Array(0..1) of SInt</td><td>1</td><td><input type="checkbox"/></td><td><input type="checkbox"/></td></tr> <tr><td>25</td><td>DiagMP</td><td>Array(0..1) of SInt</td><td>1</td><td><input type="checkbox"/></td><td><input type="checkbox"/></td></tr> <tr><td>26</td><td>DiagLength</td><td>DInt</td><td>4</td><td><input type="checkbox"/></td><td><input type="checkbox"/></td></tr> <tr><td>27</td><td>StartUp</td><td>Int</td><td>2</td><td><input type="checkbox"/></td><td><input type="checkbox"/></td></tr> <tr><td>28</td><td>SubSlot</td><td>Int</td><td>2</td><td><input type="checkbox"/></td><td><input type="checkbox"/></td></tr> <tr><td>29</td><td>BigArray</td><td>Array(0..1) of SInt</td><td>1</td><td><input type="checkbox"/></td><td><input type="checkbox"/></td></tr> <tr><td>32</td><td>MovingBits</td><td>Int</td><td>2</td><td><input type="checkbox"/></td><td><input type="checkbox"/></td></tr> <tr><td>35</td><td>MoveStart</td><td>Bool</td><td>1</td><td><input type="checkbox"/></td><td><input type="checkbox"/></td></tr> <tr><td>36</td><td>MoveStart</td><td>Bool</td><td>1</td><td><input type="checkbox"/></td><td><input type="checkbox"/></td></tr> <tr><td>37</td><td>CartonSize</td><td>DInt</td><td>4</td><td><input type="checkbox"/></td><td><input type="checkbox"/></td></tr> <tr><td>38</td><td>ControlWord</td><td>DInt</td><td>4</td><td><input type="checkbox"/></td><td><input type="checkbox"/></td></tr> <tr><td>40</td><td>afPasty</td><td>Real</td><td>4</td><td><input type="checkbox"/></td><td><input type="checkbox"/></td></tr> </tbody> </table>	ID	Name	DataType	Length	Read	Write	21	ReadResponseFlags	Bool	1	<input type="checkbox"/>	<input type="checkbox"/>	22	ReadResponseEdge	Bool	1	<input type="checkbox"/>	<input type="checkbox"/>	24	DiagData	Array(0..1) of SInt	1	<input type="checkbox"/>	<input type="checkbox"/>	25	DiagMP	Array(0..1) of SInt	1	<input type="checkbox"/>	<input type="checkbox"/>	26	DiagLength	DInt	4	<input type="checkbox"/>	<input type="checkbox"/>	27	StartUp	Int	2	<input type="checkbox"/>	<input type="checkbox"/>	28	SubSlot	Int	2	<input type="checkbox"/>	<input type="checkbox"/>	29	BigArray	Array(0..1) of SInt	1	<input type="checkbox"/>	<input type="checkbox"/>	32	MovingBits	Int	2	<input type="checkbox"/>	<input type="checkbox"/>	35	MoveStart	Bool	1	<input type="checkbox"/>	<input type="checkbox"/>	36	MoveStart	Bool	1	<input type="checkbox"/>	<input type="checkbox"/>	37	CartonSize	DInt	4	<input type="checkbox"/>	<input type="checkbox"/>	38	ControlWord	DInt	4	<input type="checkbox"/>	<input type="checkbox"/>	40	afPasty	Real	4	<input type="checkbox"/>	<input type="checkbox"/>
ID	Name	DataType	Length	Read	Write																																																																																							
21	ReadResponseFlags	Bool	1	<input type="checkbox"/>	<input type="checkbox"/>																																																																																							
22	ReadResponseEdge	Bool	1	<input type="checkbox"/>	<input type="checkbox"/>																																																																																							
24	DiagData	Array(0..1) of SInt	1	<input type="checkbox"/>	<input type="checkbox"/>																																																																																							
25	DiagMP	Array(0..1) of SInt	1	<input type="checkbox"/>	<input type="checkbox"/>																																																																																							
26	DiagLength	DInt	4	<input type="checkbox"/>	<input type="checkbox"/>																																																																																							
27	StartUp	Int	2	<input type="checkbox"/>	<input type="checkbox"/>																																																																																							
28	SubSlot	Int	2	<input type="checkbox"/>	<input type="checkbox"/>																																																																																							
29	BigArray	Array(0..1) of SInt	1	<input type="checkbox"/>	<input type="checkbox"/>																																																																																							
32	MovingBits	Int	2	<input type="checkbox"/>	<input type="checkbox"/>																																																																																							
35	MoveStart	Bool	1	<input type="checkbox"/>	<input type="checkbox"/>																																																																																							
36	MoveStart	Bool	1	<input type="checkbox"/>	<input type="checkbox"/>																																																																																							
37	CartonSize	DInt	4	<input type="checkbox"/>	<input type="checkbox"/>																																																																																							
38	ControlWord	DInt	4	<input type="checkbox"/>	<input type="checkbox"/>																																																																																							
40	afPasty	Real	4	<input type="checkbox"/>	<input type="checkbox"/>																																																																																							

	What to do	Result
<p>7.</p>	<p>Make the selection and “Confirm” your selection.</p>  <p>Acknowledge your selection</p> 	<p>The generation of the Datablock will now become available.</p>
<p>8.</p>	<p>Next click on “Generate”</p>  <p>and confirm the operation</p> 	<p>You will now have a new or modified data block named “RockwellTags” containing the tag definitions.</p>  <p>An XML file is generated containing the selections made. It can be imported using TIA Portal Openness later.</p> 

© Siemens AG 2022. All rights reserved  
© Siemens AG 2022. All rights reserved

**NOTICE** Make sure the “CIP Service Tool” is closed and has disconnected from the project you used for the import, before opening it with TIA Portal. Otherwise, you are unable to open it.

With the “CIP Service Tool” you can easily create the necessary data block, which contains the process values accessible to you.

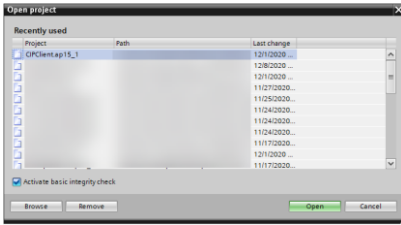
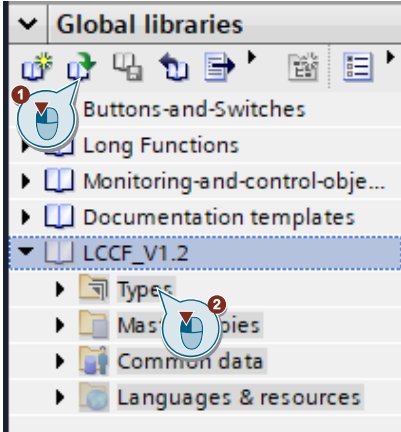
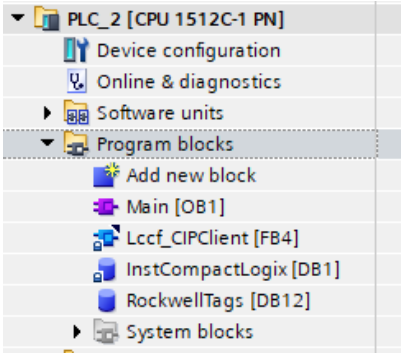
# 5 Operating

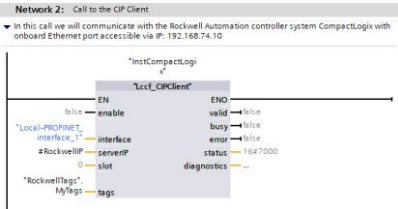
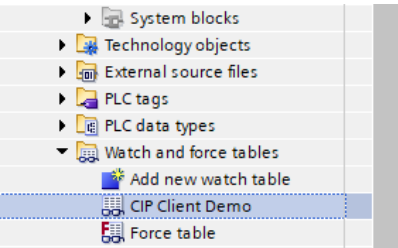
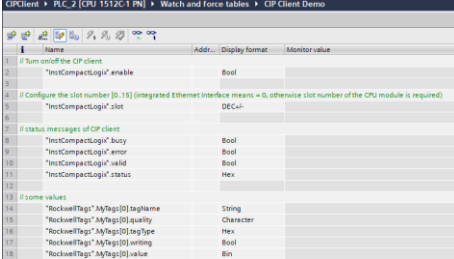
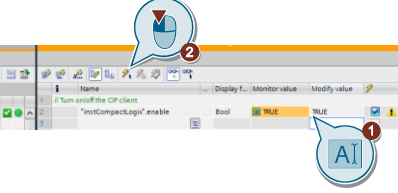
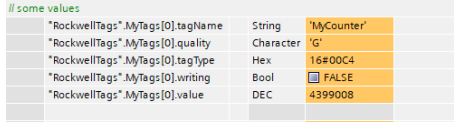
## 5.1 Start the application

Once the data block with the Rockwell Automation process values is generated this document will show you how the “LCCF\_CipClient” block needs to be parameterized to function properly.

Follow the steps listed below to parameterize the “LCCF\_CPIClient” block.

Table 5-1

	What to do	Result
1.	<p>open the TIA Portal project of your choice. Preferably this is the same as you imported the data block into.</p> 	<p>The project contains already a PLC with the imported data block “RockwellTags” in.</p>
2.	<p>open the “LCCF” and</p>  <p>import the “LCCF_CipClient” block from the Types</p> 	<p>You now have the “LCCF_CipClient” block in your project.</p>



	What to do	Result																																																																												
3.	<p>Open the block, where you want to place the call to the “LCCF_CipClient” block.</p>  <p>It is recommended to place it either in the free running program cycle (OB1) or into a cyclic interrupt (OB30) but not in both.</p>	<p>You will be asked for an instance DB to be created. This document uses “InstCompactLogix” as the instance DB.</p>																																																																												
4.	<p>Compile and Download to your SIMATIC controller.</p>																																																																													
5.	<p>open a new or existing watch table</p> 																																																																													
6.	<p>Populate at least the tags “InstCompactLogix”.enable “InstCompactLogix”.slot</p>	 <table border="1"> <thead> <tr> <th>Name</th> <th>Addr.</th> <th>Display format</th> <th>Monitor value</th> </tr> </thead> <tbody> <tr><td>1   If turn on/off the CIP client</td><td></td><td></td><td></td></tr> <tr><td>2   "InstCompactLogix".enable</td><td></td><td>Bool</td><td></td></tr> <tr><td>3</td><td></td><td></td><td></td></tr> <tr><td>4   If configure the slot number (0..15) (integrated ethernet interface means = 0, otherwise slot number of the CPU module is required)</td><td></td><td></td><td></td></tr> <tr><td>5   "InstCompactLogix".slot</td><td></td><td>DEC</td><td></td></tr> <tr><td>6</td><td></td><td></td><td></td></tr> <tr><td>7   status messages of CIP client</td><td></td><td></td><td></td></tr> <tr><td>8   "InstCompactLogix".busy</td><td></td><td>Bool</td><td></td></tr> <tr><td>9   "InstCompactLogix".error</td><td></td><td>Bool</td><td></td></tr> <tr><td>10   "InstCompactLogix".valid</td><td></td><td>Bool</td><td></td></tr> <tr><td>11   "InstCompactLogix".status</td><td></td><td>Hex</td><td></td></tr> <tr><td>12</td><td></td><td></td><td></td></tr> <tr><td>13   if some values</td><td></td><td></td><td></td></tr> <tr><td>14   "RockwellTags".MyTags[0].tagName</td><td></td><td>String</td><td></td></tr> <tr><td>15   "RockwellTags".MyTags[0].quality</td><td></td><td>Character</td><td></td></tr> <tr><td>16   "RockwellTags".MyTags[0].tagType</td><td></td><td>Hex</td><td></td></tr> <tr><td>17   "RockwellTags".MyTags[0].writing</td><td></td><td>Bool</td><td></td></tr> <tr><td>18   "RockwellTags".MyTags[0].value</td><td></td><td>Bin</td><td></td></tr> </tbody> </table>	Name	Addr.	Display format	Monitor value	1   If turn on/off the CIP client				2   "InstCompactLogix".enable		Bool		3				4   If configure the slot number (0..15) (integrated ethernet interface means = 0, otherwise slot number of the CPU module is required)				5   "InstCompactLogix".slot		DEC		6				7   status messages of CIP client				8   "InstCompactLogix".busy		Bool		9   "InstCompactLogix".error		Bool		10   "InstCompactLogix".valid		Bool		11   "InstCompactLogix".status		Hex		12				13   if some values				14   "RockwellTags".MyTags[0].tagName		String		15   "RockwellTags".MyTags[0].quality		Character		16   "RockwellTags".MyTags[0].tagType		Hex		17   "RockwellTags".MyTags[0].writing		Bool		18   "RockwellTags".MyTags[0].value		Bin	
Name	Addr.	Display format	Monitor value																																																																											
1   If turn on/off the CIP client																																																																														
2   "InstCompactLogix".enable		Bool																																																																												
3																																																																														
4   If configure the slot number (0..15) (integrated ethernet interface means = 0, otherwise slot number of the CPU module is required)																																																																														
5   "InstCompactLogix".slot		DEC																																																																												
6																																																																														
7   status messages of CIP client																																																																														
8   "InstCompactLogix".busy		Bool																																																																												
9   "InstCompactLogix".error		Bool																																																																												
10   "InstCompactLogix".valid		Bool																																																																												
11   "InstCompactLogix".status		Hex																																																																												
12																																																																														
13   if some values																																																																														
14   "RockwellTags".MyTags[0].tagName		String																																																																												
15   "RockwellTags".MyTags[0].quality		Character																																																																												
16   "RockwellTags".MyTags[0].tagType		Hex																																																																												
17   "RockwellTags".MyTags[0].writing		Bool																																																																												
18   "RockwellTags".MyTags[0].value		Bin																																																																												
7.	<p>Enable the CIP client block by modifying the value of “InstCompactLogix”.enable = TRUE</p> 	<p>You will see the read values from the Rockwell Controller</p>  <table border="1"> <thead> <tr> <th>if some values</th> <th></th> <th></th> <th></th> </tr> </thead> <tbody> <tr><td>"RockwellTags".MyTags[0].tagName</td><td>String</td><td></td><td>"MyCounter"</td></tr> <tr><td>"RockwellTags".MyTags[0].quality</td><td>Character</td><td></td><td>"G"</td></tr> <tr><td>"RockwellTags".MyTags[0].tagType</td><td>Hex</td><td></td><td>16#00C4</td></tr> <tr><td>"RockwellTags".MyTags[0].writing</td><td>Bool</td><td></td><td>FALSE</td></tr> <tr><td>"RockwellTags".MyTags[0].value</td><td>DEC</td><td></td><td>4399008</td></tr> </tbody> </table>	if some values				"RockwellTags".MyTags[0].tagName	String		"MyCounter"	"RockwellTags".MyTags[0].quality	Character		"G"	"RockwellTags".MyTags[0].tagType	Hex		16#00C4	"RockwellTags".MyTags[0].writing	Bool		FALSE	"RockwellTags".MyTags[0].value	DEC		4399008																																																				
if some values																																																																														
"RockwellTags".MyTags[0].tagName	String		"MyCounter"																																																																											
"RockwellTags".MyTags[0].quality	Character		"G"																																																																											
"RockwellTags".MyTags[0].tagType	Hex		16#00C4																																																																											
"RockwellTags".MyTags[0].writing	Bool		FALSE																																																																											
"RockwellTags".MyTags[0].value	DEC		4399008																																																																											





**NOTE**

When you want to write a value to the CIP server (Rockwell controller), then toggle the writing to 'TRUE'



For this you can either type the word TRUE or a 1 into the marked column

20	"RockwellTags".MyTags[1].tagName	String	'MoveLeft'		
21	"RockwellTags".MyTags[1].quality	Character	'G'		
22	"RockwellTags".MyTags[1].writing	Bool	<input checked="" type="checkbox"/> TRUE	TRUE	
23	"RockwellTags".MyTags[1].value	Hex	16#0000_0000		



followed by modifying the value

20	"RockwellTags".MyTags[1].tagName	String	'MoveLeft'		
21	"RockwellTags".MyTags[1].quality	Character	'G'		
22	"RockwellTags".MyTags[1].writing	Bool	<input checked="" type="checkbox"/> TRUE	TRUE	
23	"RockwellTags".MyTags[1].value	Hex	16#0000_00FF	16#0000_00FF	

to finally remove the 'write' command ('writing' = FALSE)

20	"RockwellTags".MyTags[1].tagName	String	'MoveLeft'		
21	"RockwellTags".MyTags[1].quality	Character	'G'		
22	"RockwellTags".MyTags[1].writing	Bool	<input type="checkbox"/> FALSE	FALSE	
23	"RockwellTags".MyTags[1].value	Hex	16#0000_00FF		

As a test you may overwrite the value to see it updating.

20	"RockwellTags".MyTags[1].tagName	String	'MoveLeft'		
21	"RockwellTags".MyTags[1].quality	Character	'G'		
22	"RockwellTags".MyTags[1].writing	Bool	<input type="checkbox"/> FALSE	FALSE	
23	"RockwellTags".MyTags[1].value	Hex	16#0000_00FF	16#0000_0000	

## 5.2 Troubleshooting

In case the result is not as expected the cause could be found on both sides of the communication path.

Before you try to change any of the program or configuration, check the physical installation first.

### 5.2.1 Physical check

1. Is the SIMATIC powered up?
2. Is the SCALANCE switch powered up if you have used a network switch?
3. Is the SIMATIC Comfort Panel powered up?
4. Are the network cable properly inserted into the LAN sockets?  
This can be determined by evaluating the port LEDs of the devices. At least the Link LED should be illuminated.

Table 5-2: Physical checks

observation	possible cause	remedy
SIMATIC is not reachable from TIA Portal	SIMATIC is not powered up.	<ul style="list-style-type: none"> <li>• Check power supply and wiring with the installation manual.</li> <li>• Correct wiring</li> <li>• Power the Power Supply up</li> </ul>
	SIMATIC does not have network connection	<ul style="list-style-type: none"> <li>• Check network cable to be inserted properly into the network socket (P1.X1 or P1.X2)</li> <li>• Check and correct network settings of your PC</li> </ul>
SIMATIC Comfort Panel is not reachable from TIA Portal	Comfort Panel is not powered up	<ul style="list-style-type: none"> <li>• Check and correct the power supply to Comfort Panel</li> </ul>
	Comfort Panel does not have network connection	<ul style="list-style-type: none"> <li>• Check and correct the power supply to the network switch</li> <li>• Check network cable to be inserted properly into the network socket.</li> <li>• Check and correct network settings of your PC.</li> </ul>
SIMATIC cannot communicate with SIMATIC Comfort Panel	Network switch is not powered up	<ul style="list-style-type: none"> <li>• Check and correct power supply to the network switch.</li> </ul>

If you checked everything and there is no communication at all, then perform the checks recommended in the next chapter.

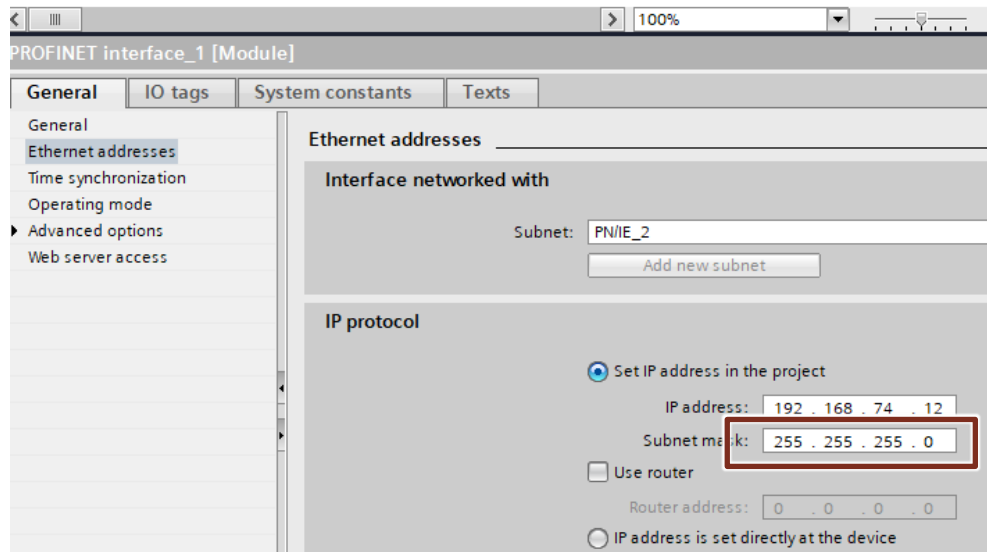
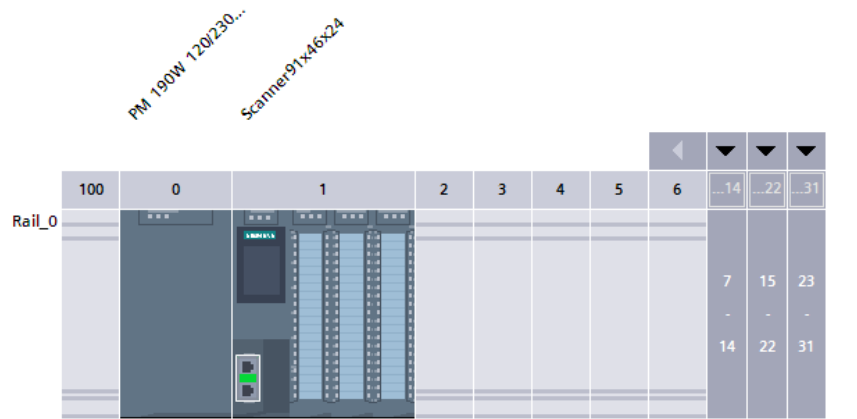
### 5.2.2 Network settings

Communication issues can be caused also by a misconfiguration of one of the communication partners along the whole path.

Check the Ethernet settings for the communication partners to have

1. The same subnet mask (here: 255.224.0.0)

Figure 5-1: Ethernet settings



2. Different IP addresses of the same subnet  
 For example this application example uses  
 192.168.74.12 for the S7-1500  
 192.168.74.10 for the Rockwell CompactLogix

### 5.2.3 SIMATIC Program

Answering the following questions may give you a hint on what needs to be corrected.

Table 5-3: CIP client checks

Observation	Cause	Remedy
status information does not change their values, when enable is set to true	The block is not executed	place an unconditional call to the block in either <ul style="list-style-type: none"> <li>• cyclic program</li> <li>• cyclic interrupt program</li> </ul>
error is true, the moment enable is set to true	Parameterization error	check the status code and correct the parameterization
valid becomes false after a certain time	Connection problems	check the status code and follow the specific recommendations further down in the document.

The CIP client block reports certain error codes to inform the user about issues in the execution. This document describes the status codes the CIP client block reports in the chapter "Block Parameter".

## 6 LCCF\_CipClient block

### 6.1 Parameters

The CIP client block has been designed to require a minimum of parameters to make its use as easy as possible. Still a minimum external configuration is necessary, which is explained in the following chapter.

A call to the LCCF\_CipClient block requires an instance DB to store operation relevant data internally as shown in the below figure.

Table 6-1: Block call to CIP Client

LAD	SCL	
	<pre> 7 // We are setting the IP address of the CIP server 8 // Here this should be the CompactLogix at IP: 192.168.74.10 9 // 10 "InstCompactLogix".serverIP.ADDR[1] := 192; 11 "InstCompactLogix".serverIP.ADDR[2] := 168; 12 "InstCompactLogix".serverIP.ADDR[3] := 74; 13 "InstCompactLogix".serverIP.ADDR[4] := 10; 14 15 // Now we issue the read/ write commands for the configured tags 16 // 17 "InstCompactLogix"(&lt;table&gt; 18   &lt;tbody&gt; <tr> <td> <p>The instance DB is generated automatically by the TIA Portal, when the call to the block is placed. In here it is named “InstCompactLogix”</p> </td> </tr> </pre>	<p>The instance DB is generated automatically by the TIA Portal, when the call to the block is placed. In here it is named “InstCompactLogix”</p>
<p>The instance DB is generated automatically by the TIA Portal, when the call to the block is placed. In here it is named “InstCompactLogix”</p>		

Table 6-2: Parameter of the LCCF\_CipClient block

Name	Direction	Data Type	Description
enable	Input	BOOL	Rising edge enables the functionality of the block. Any previously reported fault will be cleared, and conditions re-evaluated. Falling edge shuts the block down and stops any communications.
interface	Input	HW_ANY	Hardware Identifier of the interface to use for the communication. This typically uses a system defined constant. It is possible to use any “Open User Communication” supporting interface. This includes Industrial Ethernet CMs and CPs
serverIP	Input	IP_V4	Identifies the CIP Server to communicate with. The server is identified by its IP v4 address.
slot	Input	USINT	Identifies the slot, where the CIP server is located in. Valid numbers are in the range from 0 to 15. When addressing a Micro800 system, the correct value is 127 Default value is 0

Name	Direction	Data Type	Description
updateTime <sup>1</sup>	Input	Time	Defines the desired update time. The default setting is 10ms.
tags	InOut	Array[*] of "LCCF_typeTagDef"	List of tags to read from or to write to as array with variable length. Each array element is of type "LCCF_typeTagDef". Refer to chapter 5.2 <a href="#">Creating the tag lists</a> for details about the tag definitions
valid	Output	BOOL	TRUE indicates that the values in the mapping variables are valid. FALSE some or all values are invalid and should NOT be used for process control.
busy	Output	BOOL	TRUE indicates the CIP Client block is actively processing requests. FALSE indicates the block is not processing requests.
error	Output	BOOL	TRUE indicates that an error occurred during the operation of the block. Depending on the type of the error indicated by status (see below) cycling of the enable flag may clear the error. FALSE indicates no error.
status	Output	WORD	Status information about the operational state of this block. For details see CIP Client block status messages
diagnostic <sup>2</sup>	Output	LCCF_typeDiagnostic	A structure containing additional information in case of an error, which are relevant for debugging the CIP server block. The content is of value for the developer.

### 6.1.1 Block status messages

The LCCF\_CipClient block reports a status information to the user, which follows a standardized pattern.

The status code is split into the error flag and a status information value.

Table 6-3: Error and status message format

15	14	12	11	8	7	0
Error	Info/ Warning	Class Code			Specific Status Codes	
16#7 = Information	16#8 = Error	0 = Information				
		2 = Parameter related				
		4 = Internal Cause				
		6 = External Cause				

<sup>1</sup> The parameter "updateTime" may be hidden in the block call

<sup>2</sup> The parameter „diagnostics“ may be hidden in the block call.

The CIP client reports specific status codes. They are listed and explained in the following table.

Table 6-4: LCCF\_CipClient block status messages

Valid	Busy	Error	Status Code (in hex)	Cause	Remedy
TRUE	TRUE	FALSE	16#0000	Success/ OK	--
FALSE	FALSE	FALSE	16#7000	No Call/ Idle	Block is called with enable = FALSE. Create rising edge on enable to start execution
FALSE	TRUE	FALSE	16#7001	Initial call	Block starts initialization and performs parameter check
TRUE	TRUE	FALSE	16#7002	Follow Up call	Block continues initialization
TRUE	TRUE	FALSE	16#7601	Warning: Server TimeOut	Nothing to do. Server will reset connection and restart automatically.
FALSE	FALSE	TRUE	16#8201	Invalid Interface specified	Review the Hardware Identifier specified at the Interface parameter. This should be a system managed constant pointing to an interface.
FALSE	FALSE	TRUE	16#8202	Invalid tag list boundaries	The assigned tag list array is invalid, e.g. lower limit is greater than upper limit. Review the tag list assigned variable.
FALSE	FALSE	TRUE	16#8203	Invalid tag definition found	At least one of the tags marked for writing contains an unsupported data type code (tagType).
FALSE	FALSE	TRUE	16#8204	Invalid slot number	The defined slot number has an invalid value. It must be either 0 to 15 or 127 (when addressing a Micro800)
FALSE	FALSE	TRUE	16#8401	Error: Cannot set up server connection	The CIP client block cannot set up a TCP connection. This indicates either an improperly selected interface or the lack of communication resources
FALSE	FALSE	TRUE	16#8402	Error: Cannot disconnect the server connection	The CIP client block fails to reset the server connection. It is recommended to reset the SIMATIC controller.
FALSE	FALSE	TRUE	16#8601	Error: failure during receive of data	The CIP client failed to complete a receive system call. This may indicate a connection break. Reset the server by cycling enable is recommended.
FALSE	FALSE	TRUE	16#8602	Error: failure	The CIP client failed to

Valid	Busy	Error	Status Code (in hex)	Cause	Remedy
				during send of data	complete a send system call. This may indicate a broken connection. Reset the server by cycling the enable in recommended.
FALSE	FALSE	TRUE	16#8603	Error: Unknown ServiceCode received	An unknown service code has been received. This could have been the case, when the communication has been corrupted during transmission. Please inform the developer about this and take notes, when this happened.
FALSE	FALSE	TRUE	16#8600	Error: Undefined state	The CIP server block requested an undefined internal state. This needs to be reported to the developer alongside with the information stored in the diagnostics parameter.

### 6.1.2 Technical data

For better planning of the automation program the user must be aware, that operating the LCCF\_CipClient block has certain impacts on the PLC load.

As all the protocol handling is done in a user program, the cycle time will be extended by the time the selected CPU model needs to place the requests and process the responses. As one could imagine the more often data are requested, the more often these requests will be answered.

Setting the update rates to the lowest acceptable value will reduce the load on the automation program.

Measurements for a SIMATIC S7-1512C show an average load below 1ms.

Table 6-5: Execution times for CIP Client

System	min. load	average load	max. load
S7-1512C	0.1ms	1.0ms	1.5ms
S7-1215C	0.6ms	1.2ms	1.8ms

The measured load was in both cases reading 3 values and writing 1 value.

Besides program execution time memory considerations should also be made, when selecting a CPU model.

For the S7-1200 controller family the technical data are listed in the following table

Table 6-6: Memory Consumption S7-1200

Block	Load Memory	Work Memory
LCCF_CipClient	271.599 Bytes	14.889 Bytes
Instance DB	11.346 Bytes	1.932 Bytes



The execution time of the block is CPU model dependent and was measured on a SIMATIC S7-1215C to be less than 1ms in average.

In the S7-1500 controller family the two required blocks require the memory listed in the following table.

Table 6-7: Memory Consumption S7-1500

Block	Load Memory	Work Memory
LCCF_CipClient	277.508 Bytes	15.200 Bytes
Instance DB	11.398 Bytes	2.020 Bytes

<b>NOTICE</b>	Each defined tag requires data memory additional to what has been listed already. Defining many tags may cause the memory requirements exceed the available memory in your system.	
	<b>Load Memory</b>	<b>Work Memory</b>
	42 Bytes	48 Bytes
The provided information is on a per tag definition base. This means, that this value must be multiplied by the number of tags to get the size of the DB storing the tag definitions.		

The CIP client block puts requests to a single server. It is possible to use multiple instances of the block to request values from different servers.

The number of parallel interrogated servers depends on the number of available OUC connection resources in the selected CPU model. For details refer to the technical data of the specific CPU model.

## 6.2 What is next?

Future extensions will enhance communication capabilities to support further Rockwell Automation control systems such as:

- MicroLogix
- PLC-5

Also, some improvements are planned for later versions in regard of supported data types:

- Complete Arrays
- Structured tags
- Larger number of tags

Further enhancements are planned for the configuration tool as well, which are:

- Delta Creation of tag definition
- Support for multiple target controller

Support for multiple SIMATIC controller per TIA Portal project

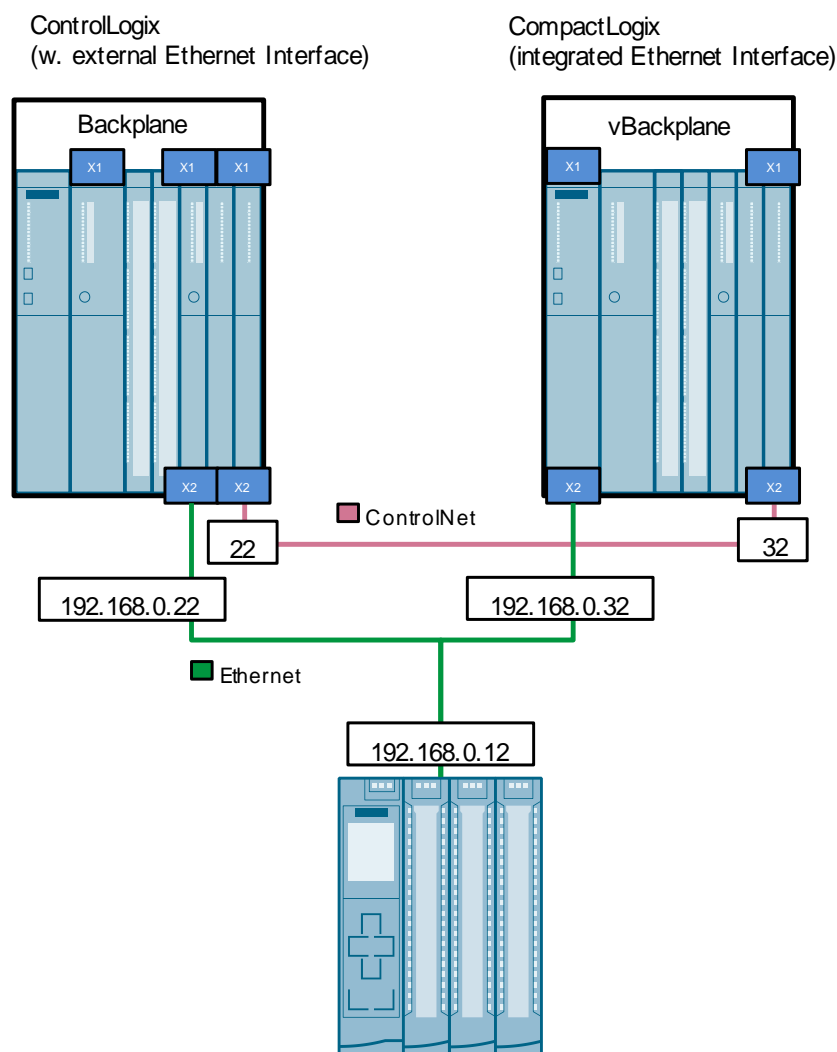
## 7 Appendix

### 7.1 CIP/ PCCC Communication path settings

The CIP uses a method of encapsulating the messages adding communication path segments to the command segment. This leads to a longer and longer growing message as the number of routing stations grows.

As an example, the following simple layout is used to explain that fact. Here the S7-1500 is acting as a CIP Client instead the WinCC Adv. RT used throughout the application example. The CIP server is represented by a ControlLogix or CompactLogix system.

Figure 7-1: Schematic communication path setup



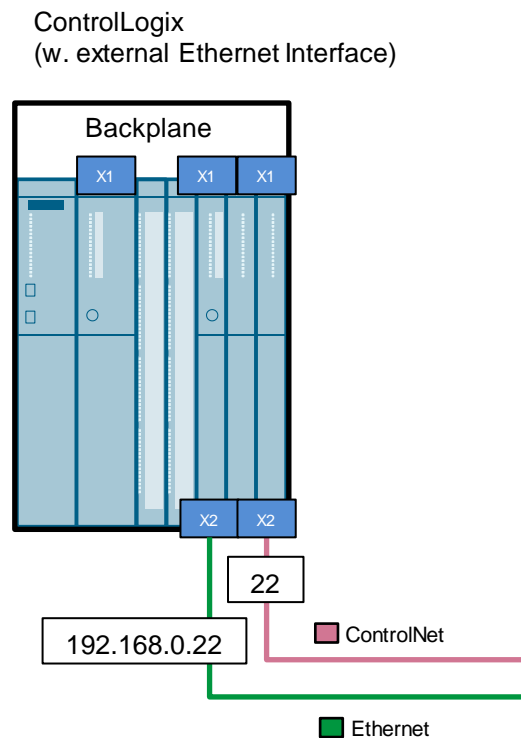
Each CIP message contains the routing information to the target. Every time a bridge accepts such a CIP message it strips out its own address information and forwards the modified CIP message to the next addressee along the way. When there is no further routing information available the destination target has been reached and the CIP message will be interpreted. Once interpreted the response will be sent the same way back the request came. This means each bridge keeps a reference to the request package.

### 7.1.1 Case 1 – Accessing data inside an Allen- Bradley PLC

In this case a ControlLogix uses the ENET bridge in slot 4 (0 based counting) for communicating with the CIP client. The processor module itself is in slot 1. An additional ControlNet Bridge is in slot 6 and is not used in this scenario.

The S7-1500 sends a request to the ControlLogix on 192.168.0.22, slot 1.

Figure 7-2: Access Path ControlLogix with external Ethernet Interface



Therefore, the parameters in the WinCC Panel's driver configuration needs to be:

Table 7-1: WinCC EtherNet/IP driver settings

Parameter Name	Value
IPv4	192.168.0.22
Communication Path	1,1

This communication path is formed as follows:

“192.168.0.22, 1, 1”

The IPv4 address (192.168.0.22) is addressing the X2 interface of the ENET card. With the following part (1) in the communication path the interface X1 (more accurate the backplane) is being addressed. This means the CIP message is being forwarded to the backplane after it has removed the IPv4 address from the communication path. The backplane therefore sees a reduced communication path like this

“1, 1”

It recognized that the CIP message was not meant for it and therefore, it removes its address part (1) from the CIP message. The remaining CIP message is being forwarded to the processor module as indicated by the remaining address

information (1). The message has now a reduced communication path. Only the slot number is contained. The processor removes that information and starts interpreting the remaining message as there is no further routing information in the packet.

The following Wireshark packet analysis shows the additional segment.

Figure 7-3: Access Path encoding

```

▶ Internet Protocol Version 4, Src: 10.11.20.51, Dst: 10.11.20.56
▶ Transmission Control Protocol, Src Port: 49189, Dst Port: 44818, Seq: 115, Ack: 73, Len: 64
▶ Ethernet/IP (Industrial Protocol), Session: 0x00170001, Send RR Data
└─ Common Industrial Protocol
  ▶ Service: Unknown Service (0x52) (Request)
    Request Path Size: 2 (words)
    Request Path: Connection Manager, Instance: 0x01
  └─ CIP Connection Manager
    └─ Service: Unconnected Send (Request)
      0... .... = Request/Response: Request (0x0)
      .101 0010 = Service: Unconnected Send (0x52)
    └─ Command Specific Data
      ...0 .... = Priority: 0
      ... 0111 = Tick time: 7
      Time-out ticks: 233
      Actual Time Out: 29824ms
      Message Request Size: 10
    └─ Message Request
      └─ Common Industrial Protocol
        ▶ Service: Unknown Service (0x4c) (Request)
          Request Path Size: 3 (words)
          └─ Request Path: cTag
            ▶ Path Segment: 0x91 (ANSI Extended Symbol Segment)
          └─ CIP Class Generic
            └─ Command Specific Data
              Data: 0a00
              Route Path Size: 1 (words)
              Reserved: 0x00
              └─ Route Path: Port: 1, Address: 5
                └─ Path Segment: 0x01 (Port Segment)
                  000. .... = Path Segment Type: Port Segment (0)
                  ...0 .... = Extended Link Address: False
                  ... 0001 = Port: 1
                └─ Port Segment
                  Link Address: 5

```

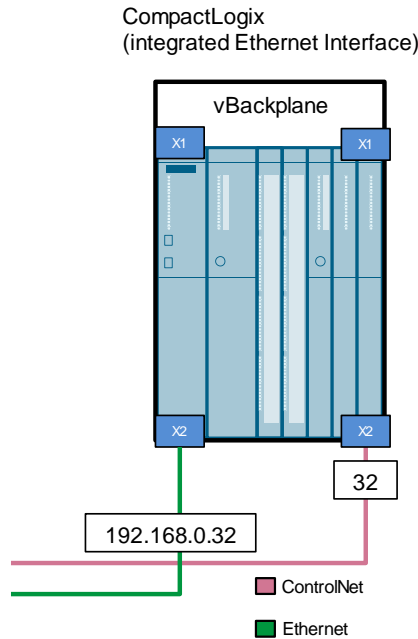
Contrary to the above-described situation, the communication path here shows that the processor module is in slot 5. Here the CIP message follows the same path as above, only difference is the slot number.

### 7.1.2 Case 2 – Accessing the Allen- Bradley PLC via integrated Ethernet port

In this scenario the CIP client wants to put some data into the CPU module using the integrated Ethernet port. In a CPU module with integrated Ethernet port the Ethernet port is in the same slot as the CPU module.

The CIP client (here the S7-1500) sends a request package to the CPU module at 192.168.0.32 addressing the processor module in slot 0.

Figure 7-4: Access Path CompactLogix with internal Ethernet Interface



© Siemens AG 2022. All rights reserved  
© Siemens AG 2022. All rights reserved

Following the previous schema, the communication path would look like this

“192.168.0.32, 1, 0”

Following the previous made explanations the CIP message would first travel through the processor module to the backplane to be bounced back to the processor module for interpretation. For such situations it is possible to use an abbreviated communication path only containing the IPv4 address.

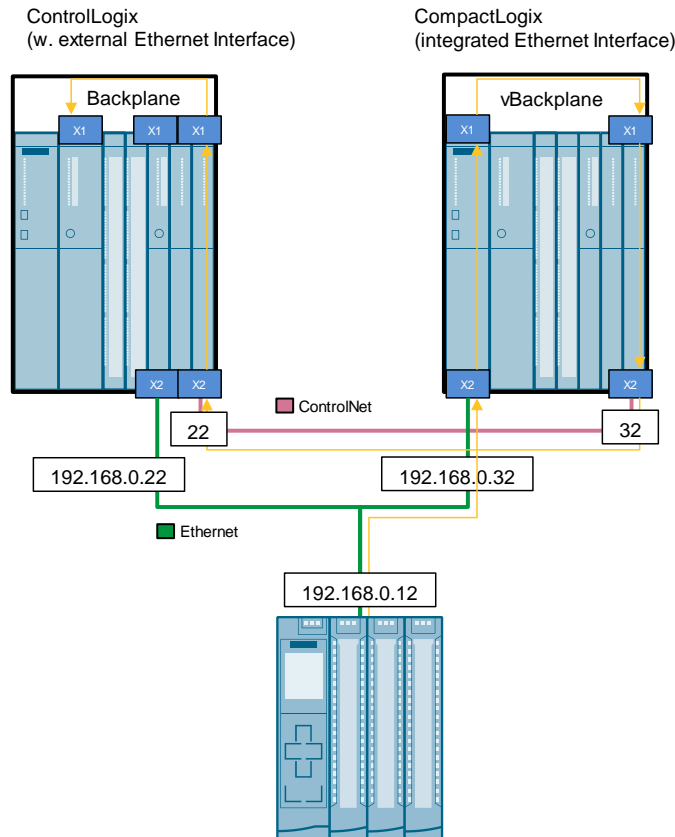
Table 7-2: WinCC EtherNet/IP driver settings

Parameter Name	Value
IPv4	192.168.0.32
Communication Path	1,0

### 7.1.3 Case 3 – Accessing the Allen- Bradley PLC via the ControlNet interface

In this scenario the CIP client (here S7-1500) wants to read data from the Allen-Bradley PLC. Now the communication path will be through the CompactLogix acting as bridge between EtherNet/IP and ControlNet.

Figure 7-5: Access Path with CompactLogix as bridge



The yellow line in the above picture is represented by the following communication path.

“192.168.0.32, 1, 6, 2, 22, 1, 1”

In the communication path the red marked sections always contain a network specific address. The blue marked part identifies the backplane or the interface to be used. Here it does not matter, whether this backplane is physically existing or virtual. The green part identifies the slot number of the next bridge module (next hop) or the destination module.

Table 7-3: WinCC EtherNet/IP driver settings

Parameter Name	Value
IPv4	192.168.0.32
Communication Path	1,6,2,22,1,1

**NOTICE** This scenario is not supported by the LCCF\_CipClient block. There is no possibility to define the communication path to such an extend at the LCCF\_CipClient block.

## 7.2 Service and support

### Industry Online Support

Do you have any questions or need assistance?

Siemens Industry Online Support offers round the clock access to our entire service and support know-how and portfolio.

The Industry Online Support is the central address for information about our products, solutions, and services.

Product information, manuals, downloads, FAQs, application examples and videos – all information is accessible with just a few mouse clicks:

[support.industry.siemens.com](https://support.industry.siemens.com)

### Technical Support

The Technical Support of Siemens Industry provides you fast and competent support regarding all technical queries with numerous tailor-made offers – ranging from basic support to individual support contracts. Please send queries to Technical Support via Web form:

[www.siemens.com/industry/supportrequest](https://www.siemens.com/industry/supportrequest)

### SITRAIN – Training for Industry

We support you with our globally available training courses for industry with practical experience, innovative learning methods and a concept that is tailored to the customer's specific needs.

For more information on our offered trainings and courses, as well as their locations and dates, refer to our web page:

[www.siemens.com/sitrain](https://www.siemens.com/sitrain)

### Service offer

Our range of services includes the following:

- Plant data services
- Spare parts services
- Repair services
- On-site and maintenance services
- Retrofitting and modernization services
- Service programs and contracts

You can find detailed information on our range of services in the service catalogue web page:

[support.industry.siemens.com/cs/sc](https://support.industry.siemens.com/cs/sc)

### Industry Online Support app

You will receive optimum support wherever you are with the "Siemens Industry Online Support" app. The app is available for iOS and Android:

[support.industry.siemens.com/cs/ww/en/sc/2067](https://support.industry.siemens.com/cs/ww/en/sc/2067)

## 7.3 Related literature

Table 7-4

	Topic
\1\	Siemens Industry Online Support <a href="https://support.industry.siemens.com">https://support.industry.siemens.com</a>
\2\	Download page of this entry <a href="https://support.industry.siemens.com/cs/ww/en/view/109782317">https://support.industry.siemens.com/cs/ww/en/view/109782317</a>

## 7.4 Change documentation

Table 7-5

Version	Date	Modifications
V1.0.0	01/2021	First version
V1.0.1	05/2022	Added Information about Micro800 communications