Network Interface Appendix

Anybus® CompactCom 30 CC-Link

Doc.Id. HMSI-27-319
Rev. 2.10

Connecting Devices™

HALMSTAD • CHICAGO • KARLSRUHE • TOKYO • BEIJING • MILANO • MULHOUSE • COVENTRY • PUNE • COPENHAGEN

HMS Industrial Networks
Mailing address: Box 4126, 300 04 Halmstad, Sweden
Visiting address: Stationsgatan 37, Halmstad, Sweden

E-mail: info@hms-networks.com
www.anybus.com

Important User Information

This document is intended to provide a good understanding of the functionality offered by CC-Link. The document only describes the features that are specific to the Anybus CompactCom CC-Link. For general information regarding the Anybus CompactCom, consult the Anybus CompactCom design guides.

The reader of this document is expected to be familiar with high level software design, and communication systems in general. The use of advanced CC-Link-specific functionality may require in-depth knowledge in CC-Link networking internals and/or information from the official CC-Link specifications. In such cases, the people responsible for the implementation of this product should either obtain the CC-Link specification to gain sufficient knowledge or limit their implementation in such a way that this is not necessary.

Liability

Every care has been taken in the preparation of this manual. Please inform HMS Industrial Networks AB of any inaccuracies or omissions. The data and illustrations found in this document are not binding. We, HMS Industrial Networks AB, reserve the right to modify our products in line with our policy of continuous product development. The information in this document is subject to change without notice and should not be considered as a commitment by HMS Industrial Networks AB. HMS Industrial Networks AB assumes no responsibility for any errors that may appear in this document.

There are many applications of this product. Those responsible for the use of this device must ensure that all the necessary steps have been taken to verify that the applications meet all performance and safety requirements including any applicable laws, regulations, codes, and standards.

HMS Industrial Networks AB will under no circumstances assume liability or responsibility for any problems that may arise as a result from the use of undocumented features, timing, or functional side effects found outside the documented scope of this product. The effects caused by any direct or indirect use of such aspects of the product are undefined, and may include e.g. compatibility issues and stability issues.

The examples and illustrations in this document are included solely for illustrative purposes. Because of the many variables and requirements associated with any particular implementation, HMS Industrial Networks AB cannot assume responsibility for actual use based on these examples and illustrations.

Intellectual Property Rights

HMS Industrial Networks AB has intellectual property rights relating to technology embodied in the product described in this document. These intellectual property rights may include patents and pending patent applications in the US and other countries.

Trademark Acknowledgements

Anybus ® is a registered trademark of HMS Industrial Networks AB. All other trademarks are the property of their respective holders.

Warning:	This is a class A product. in a domestic environment this product may cause radio interference in which case the user may be required to take adequate measures.
ESD Note:	This product contains ESD (Electrostatic Discharge) sensitive parts that may be damaged if ESD control procedures are not followed. Static control precautions are required when handling the product. Failure to observe this may cause damage to the product.

Anybus CompactCom CC-Link Network Interface Appendix
Rev 2.10
Copyright© HMS Industrial Networks AB
Apr 2015 Doc Id HMSI-27-319

Table of Contents

Preface **About This Document**

Related Documents 6
Document History 6
Conventions & Terminology 7
Support 7

Chapter 1 **About the Anybus-CompactCom CC-Link**

General 8
Features 8

Chapter 2 **Tutorial**

Introduction 9
Fieldbus Conformance Notes 9
Certification 9

Chapter 3 **Basic Operation**

General Information 10
Software Requirements 10
Communication Settings 10
Data Exchange 11
Bit Area vs. Word Area 11
Data Sizes 11

Chapter 4 **Process Data Mapping**

General 13
Standard Mapping 14
Advanced Mapping (Optional) 18

Chapter 5 **CC-Link System Area Implementation**

System Area Modes 19
System Area Layout 19
System Area Location 20
Error Code Location 21
Diagnostics 22

Chapter 6 Anybus Module Objects

General Information ... 23
Anybus Object (01h) ... 24
Diagnostic Object (02h) ... 25
Instance Attributes ... 25
Network Object (03h) ... 26
Network Configuration Object (04h) ... 28
Instance Attributes (Instance #2, 'Baud rate') ... 29
Multilingual Strings ... 29
Network CC-Link Object (08h) ... 30

Chapter 7 Host Application Objects

General Information ... 35
CC-Link Host Object (F7h) ... 36

Appendix A Categorization of Functionality

Basic ... 39
Extended ... 39
Advanced ... 39

Appendix B Implementation Details

SUP-Bit Definition ... 40
Anybus State Machine ... 40
Application Watchdog Timeout Handling ... 40
CC-Link Handshaking Implementation ... 41

Appendix C Additional Certification Information

Model Code ... 42
CC-Link Version 2 ... 42

Appendix D Exception Information

Appendix E Technical Specification

Front View ... 44
Protective Earth (PE) Requirements ... 44
Power Supply ... 45
Environmental Specification ... 45
EMC Compliance ... 45

Appendix F Timing & Performance

General Information 46

Process Data 47

Overview 47

Anybus Read Process Data Delay (Anybus Delay) 47

Anybus Write Process Data Delay (Anybus Delay) 47

Network System Read Process Data Delay (Network System Delay) 48

Network System Write Process Data Delay (Network System Delay) 48

P. About This Document

For more information, documentation etc., please visit the HMS website, 'www.anybus.com'.

P.1 Related Documents

Document	Author
Anybus-CompactCom Software Design Guide	HMS
Anybus-CompactCom Hardware Design Guide	HMS
Anybus-CompactCom Software Driver User Guide	HMS
CC-Link Conformance Test specification (Publication BAP-C0401-012-A)	CLPA
CC-Link Specification (Profile) (publication BAP-05028-E)	CLPA

P.2 Document History

Summary of Recent Changes (2.01 ... 2.10)

Change	Page(s)
Changed the default value of the Model Code	37, 42
Moved front view information to technical specification	44
Updated information in section Error Code Location	21
Added information to section on Data Sizes	11
Updated and added examples to Process Data Mapping	13

Revision List

Revision	Date	Author(s)	Chapter(s)	Description
1.00	2006-03-06	PeP	All	1st release
1.10	2006-05-11	PeP	7, 6	Added Network CC-Link Object + misc. minor changes
1.11	2007-03-05	PeP	1, 6	Misc. minor corrections
1.12	2008-05-30	PeP	E	Minor update
2.00	2010-04-14	KeL	2, 5, 6, All	Change of concept, minor corrections
2.01	2011-02-10	KeL	P, 6	Minor updates
2.10	2015-04-01	KeL	3, 4, 5, 7, E	Updatas and corrections

P.3 Conventions & Terminology

The following conventions are used throughout this manual:

- Numbered lists provide sequential steps
- Bulleted lists provide information, not procedural steps
- The terms 'Anybus' or 'module' refers to the Anybus-CompactCom module.
- The terms 'host' or 'host application' refers to the device that hosts the Anybus module.
- Hexadecimal values are written in the format NNNNh or 0xNNNN, where NNNN is the hexadecimal value.
- A byte always consists of 8 bits.

P.4 Support

For general contact information and where to find support, please refer to the contact and support pages at www.anybus.com.

1. About the Anybus-CompactCom CC-Link

1.1 General

The Anybus-CompactCom CC-Link communication module provides instant CC-Link slave functionality via the patented Anybus-CompactCom host interface. Any device that supports this standard can take advantage of the features offered by the module, allowing seamless network integration regardless of network type.

This product conforms to all aspects of the host interface for Active modules defined in the Anybus-CompactCom Hardware- and Software Design Guides, making it fully interchangeable with any other device following that specification. Generally, no additional network related software support is needed, however in order to take advantage of advanced network specific functionality, a certain degree of dedicated software support may be necessary.

1.2 Features

- CC-Link Remote Device
- Up to 128 I/O (bit) points and 16 words (16 bit) of data (CC-Link v.1)
- Up to 896 I/O (bit) points and 128 words (16 bit) of data (CC-Link v.2)[1]
- Automatic CC-Link System Area handshaking (optional)
- Possibility to customize Vendor Code, Model Code and Version via application interface
- Baud rate and Station Number configuration via switches or application interface
- Galvanically isolated bus electronics

1. The host interface still limits the total amount of data to 256 bytes in each direction.

2. Tutorial

2.1 Introduction

This chapter is a complement to the Anybus CompactCom Implementation Tutorial. The ABCC tutorial describes and explains a simple example of an implementation with Anybus CompactCom. This chapter includes network specific settings that are needed for a host application to be up and running and possible to certify for use on CC-Link networks.

2.2 Fieldbus Conformance Notes

- The Anybus-CompactCom CC-Link has been tested standalone and found to comply with the CC-Link Conformance Test specification, publication BAP-C0401-012-A. The end product will however need to be re-certified in order to comply with CC-Link certification policies.
- The application alone is responsible for maintaining compatibility with the profiles defined in the CC-Link Specification (Profile), publication BAP-05028-E. This documentation is available free of charge to all registered CLPA members. To sign up as a member, please contact the CLPA (www.cc-link.org).
- In order to pass the conformance test the application will have to show an error status when an erroneous baud rate or station number according to CC-Link is used. The Anybus CompactCom CC-Link will not allow the setting of the Setup Complete attribute in the Anybus Object in such a case.
- If the host application handles the System area (no automatic handshake), it also must take full responsibility for all parts of the conformance test related to the system area.

2.3 Certification

The following steps are necessary to perform to obtain a certification:

1. **Change Vendor Code:**

   Replace the HMS Vendor ID with a unique Vendor Code[1]. This is done by implementing the CC-Link object (F7h), instance1, attribute 1 and returning the Vendor Code when receiving a Get_Attribute request.

2. **Increment SW Version:**

   Increment the SW version in the CC-Link object (F7h) if you want to keep track of the versions. This number should be incremented at each change in functionality, leading to a new recertification. Implement the CC-Link object (F7h), instance 1, attribute 2 and return the software version when receiving a Get_Attribute request.

These two steps are the smallest possible amount of actions that you need to perform to obtain a certification. Depending on how you want to use the module, several more steps will have to be taken. For more information see "Additional Certification Information" on page 42.

1. Membership in the CLPA organization is necessary to obtain a vendor code. The vendor code consists of digits 5 - 8 in the CLPA ID number, issued when you join.

3. Basic Operation

3.1 General Information

3.1.1 Software Requirements

No additional network support code needs to be written in order to support the Anybus-CompactCom CC-Link, however due to the nature of the CC-Link networking system certain restrictions must be taken into account:

- No acyclic data exchange
- ADIs must be mapped as Process Data in order to be represented on the network
- ADI Names, types and similar attributes cannot be accessed from the network.
- No support for network reset requests
- Up to 5 diagnostic instances (see 6-25 “Diagnostic Object (02h)”) can be created by the host application during normal operation. An additional 6th instance may be created in event of a major fault.

For in depth information regarding the Anybus-CompactCom software interface, consult the general Anybus-CompactCom Software Design Guide.

3.1.2 Communication Settings

Network related communication which can be accessed by the end user are grouped in the Network Configuration Object (04h).

In the case of CC-Link, this includes:

- **Station Number**

  On CC-Link, each device on the network must be assigned a unique Station Number. The highest possible Station Number depends on the number of occupied stations.

- **Baud rate**

  The module supports all common CC-Link baud rates up to 10Mbps.

3.2 Data Exchange

3.2.1 Bit Area vs. Word Area

On CC-Link, data is divided into two categories as follows:

- **Bit Area**

  Data is accessed on a bit-by-bit basis. Data is commonly referred to as RX #nn (Slave->Master) and RY #nn (Master->Slave) where 'nn' represents an addressable point (i.e. a single bit) in the Bit Area.

- **Word Area**

  Data is accessed as 16-bit words. Data is commonly referred to as RWr #nn (Slave->Master) and RWw #nn (Master->Slave) where 'nn' represents an addressable point (i.e. a word) in the Word Area.

3.2.2 Data Sizes

Please note that in default mode, the system area occupies 16 bits of the bit area, thus limiting the possible amount of process data. For more information see "CC-Link System Area Implementation" on page 19 and the examples presented in "Process Data Mapping" on page 13.

CC-Link Version 1

By default, the module automatically calculates the required number of occupied stations based on the mapped Process Data. The following data sizes are possible.d

Occupied Stations	Bit Points	Word Points	Total (Bit + Word, in bytes)
1	32 bits	4 words	12
2	64 bits	8 words	24
3	96 bits	12 words	36
4	128 bits	16 words	48

CC-Link Version 2

By implementing the 'Network Settings'-attribute (#4) in the CC-Link Object (F7h) it is possible to customize the implementation for CC-Link version 2 and use larger data sizes through extension cycles. In such case, the following sizes are possible:

Occupied Stations	1 Extension Cycle		2 Extension Cycles		4 Extension Cycles		8 Extension Cycles	
	Points	Total	Points	Total	Points	Total	Points	Total
1	32 bits	12 bytes	32 bits	20 bytes	64 bits	40 bytes	128 bits	80 bytes
	4 words		8 words		16 words		32 words	
2	64 bits	24 bytes	96 bits	44 bytes	192 bits	88 bytes	384 bits	176 bytes
	8 words		16 words		32 words		64 words	
3	96 bits	36 bytes	160 bits	68 bytes	320 bits	136 bytes	640 bits	272 bytes
	12 words		24 words		48 words		96 words	
4	128 bits	48 bytes	224 bits	92 bytes	448 bits	184 bytes	896 bits	368 bytes
	16 words		32 words		64 words		128 words	

Note 1: Although as much as 368 bytes may be exchanged physically on the bus, the host interface still limits the total amount of data to 256 bytes in each direction.

Note 2: On CC-Link, certain parts of the Bit and Word data may be reserved and should not be used for data exchange. For more information, see "CC-Link System Area Implementation" on page 19.

Note 3: For conformance test of CC-Link version 2, it is necessary to enable the CC-Link conformance test mode, see "Network CC-Link Object (08h)" on page 30, command CCL_Conf_Test_Mode on page 34.

4. Process Data Mapping

4.1 General

The module features two different mapping schemes for the Process Data.

- **Standard Mapping**

  Process Data is mapped using the standard mapping commands (Map_ADI_Write_Area & Map_ADI_Read_Area).

  See also...

  - "Standard Mapping" on page 14

- **Advanced Mapping (Optional)**

  To be able to fully exploit the functionality offered on CC-Link, an additional network-specific mapping method has been implemented. This is realized through the use of two network-specific commands, Map_ADI_Specified_Write_Area and Map_ADI_Specified_Read_Area.

  See also...

  - "Advanced Mapping (Optional)" on page 18

See also...

- "CC-Link System Area Implementation" on page 19
- "Network Object (03h)" on page 26
- "Command Details: Map_ADI_Specified_Write_Area" on page 31
- "Command Details: Map_ADI_Specified_Read_Area" on page 32

4.2 Standard Mapping

This scheme is used when the host application uses the standard mapping commands:

- Map_ADI_Write_Area maps data to the RX (bit) and RWr (word) areas.
- Map_ADI_Read_Area maps data from the RY (bit) and RWw (word) areas.
- ADIs are mapped to consecutive locations in the respective areas in the same order as the mapping commands are issued.
- ADIs of BOOL-type are mapped to the first unused bit of the corresponding Bit Area.
- Non-BOOL ADIs are mapped to consecutive words beginning at the least significant byte of the first unused word of the corresponding Word Area. Additional rules apply as follows:
  - ADIs are always treated as single entities, even if they consist of multiple elements
  - ADIs of odd total size are padded to even length (i.e. the most significant byte of the last mapped word will be empty).
- The number of stations will always be the same in both directions.

The system area will always be located at the very end of the Bit Area. The error code will occupy the last word of the word area for slave to master data. See Chapter 5 for more information.

Example 1 (One station)

4 ADIs are mapped:

ADI #	Mapping Command	Data Type	No. of Elements
1	Map_ADI_Write_Area	BOOL	16
2	Map_ADI_Write_Area	UINT8	6
3	Map_ADI_Read_Area	BOOL	16
4	Map_ADI_Read_Area	UINT8	8

These ADIs will, together with the system area and the error code, occupy one station (12 bytes) in each direction.

Slave to Master Data (Including Write Process Data):

Resulting Bit Area Contents:

Point	Contents							
RX #7... 0	ADI1[7]	ADI1[6]	ADI1[5]	ADI1[4]	ADI1[3]	ADI1[2]	ADI1[1]	ADI1[0]
RX #15... 8	ADI1[15]	ADI1[14]	ADI1[13]	ADI1[12]	ADI1[11]	ADI1[10]	ADI1[9]	ADI1[8]
RX #23... 16	System Area							
RX #31...24								

Resulting Word Area Contents:

Point	Contents (LSB)	Contents (MSB)
RWr #0	ADI2[0]	ADI2[1]
RWr #1	ADI2[2]	ADI2[3]
RWr #2	ADI2[4]	ADI2[5]
RWr #3	Error Code[LSB]	Error Code[MSB]

Master to Slave Data (Including Read Process Data):

Resulting Bit Area Contents

Point	Contents							
RY #7... 0	ADI3[7]	ADI3[6]	ADI3[5]	ADI3[4]	ADI3[3]	ADI3[2]	ADI3[1]	ADI3[0]

Point	Contents							
RY #15... 8	ADI3[15]	ADI3[14]	ADI3[13]	ADI3[12]	ADI3[11]	ADI3[10]	ADI3[9]	ADI3[8]
RY #23... 16	System Area							
RY #31...24								

Resulting Word Area Contents:

Point	Contents (LSB)	Contents (MSB)
RWw #0	ADI4[0]	ADI4[1]
RWw #1	ADI4[2]	ADI4[3]
RWw #2	ADI4[4]	ADI4[5]
RWw #3	ADI4[6]	ADI4[7]

Example 2 (Two Stations)

6 ADIs are mapped:

ADI #	Mapping Command	Data Type	No. of Elements
1	Map_ADI_Write_Area	BOOL	9
2	Map_ADI_Write_Area	BOOL	8
3	Map_ADI_Write_Area	UINT8	2
4	Map_ADI_Write_Area	UINT8	2
5	Map_ADI_Read_Area	BOOL	3
6	Map_ADI_Read_Area	BOOL	13
7	Map_ADI_Read_Area	UINT8	4
8	Map_ADI_Read_Area	UINT32	1

These ADIs will occupy more than one station in each direction as the number of bit points (including system Area) for the slave to master data is larger than 32 . As you can see the contents will "spill over" because of an extra BOOL that is mapped as bit area point data. Please note, that if the mapped contents exceed the max amount for one station in only one direction, one station will be added in both directions.

Slave to Master Data (Including Write Process Data):

Resulting Bit Area Contents:

Point	Contents							
RX #7... 0	ADI1[7]	ADI1[6]	ADI1[5]	ADI1[4]	ADI1[3]	ADI1[2]	ADI1[1]	ADI1[0]
RX #15... 8	ADI2[6]	ADI2[5]	ADI2[4]	ADI2[3]	ADI2[2]	ADI2[1]	ADI2[0]	ADI1[8]
RX #23... 16								ADI2[7]
RX #31... 24								
RX #39... 32								
RX #47... 40								
RX #55... 48	System Area							
RX #63... 56								

Resulting Word Area Contents:

Point	Contents (LSB)	Contents (MSB)
RWr #0	ADI3[0]	ADI3[1]
RWr #1	ADI4[0]	ADI4[1]
RWr #2		
RWr #3		
RWr #4		
RWr #5		

Point	Contents (LSB)	Contents (MSB)
RWr #6		
RWr #7	Error Code[LSB]	Error Code[MSB]

Master to Slave Data (Including Read Process Data):

Resulting Bit Area Contents

Point	Contents							
RY #7... 0	ADI6[4]	ADI6[3]	ADI6[2]	ADI6[1]	ADI6[0]	ADI5[2]	ADI5[1]	ADI5[0]
RY #15... 8	ADI6[12]	ADI6[11]	ADI6[10]	ADI6[9]	ADI6[8]	ADI6[7]	ADI6[6]	ADI6[5]
RY #23... 16								
RY #31... 24								
RY #39... 32								
RY #47... 40								
RY #55... 48	System Area							
RY #63... 56								

Resulting Word Area Contents:

Point	Contents (LSB)	Contents (MSB)
RWw #0	ADI7[0]	ADI7[1]
RWw #1	ADI7[2]	ADI7[3]
RWw #2	ADI8[Least significant word]	
RWw #3	ADI8[Most significant word]	
RWw #4		
RWw #5		
RWw #6		
RWw #7		

Example 3 (Two Stations)

4 ADIs are mapped:

ADI #	Mapping Command	Data Type	No. of Elements
1	Map_ADI_Write_Area	BOOL	16
2	Map_ADI_Write_Area	UINT8	7
3	Map_ADI_Read_Area	BOOL	16
4	Map_ADI_Read_Area	UINT8	9

These ADIs will occupy more than one station in each direction as the amount of word data mapped is more than 4 (8 bytes). As you can see the contents will "spill over" because of an extra UINT8 in each direction, mapped as word area point data. As mention earlier, even if the amount of data only spills over in one direction, an extra station will be added in both directions.

Slave to Master Data (Including Write Process Data):

Resulting Bit Area Contents:

Point	Contents							
RX #7... 0	ADI1[7]	ADI1[6]	ADI1[5]	ADI1[4]	ADI1[3]	ADI1[2]	ADI1[1]	ADI1[0]
RX #15... 8	ADI1[15]	ADI1[14]	ADI1[13]	ADI1[12]	ADI1[11]	ADI1[10]	ADI1[9]	ADI1[8]
RX #23... 16								
RX #31... 24								
RX #39... 32								
RX #47... 40								

Point	Contents
RX #55... 48	System Area
RX #63... 56	

Resulting Word Area Contents:

Point	Contents (LSB)	Contents (MSB)
RWr #0	ADI2[0]	ADI2[1]
RWr #1	ADI2[2]	ADI2[3]
RWr #2	ADI2[4]	ADI2[5]
RWr #3	ADI2[6]	
RWr #4		
RWr #5		
RWr #6		
RWr #7	Error Code[LSB]	Error Code[MSB]

Master to Slave Data (Including Read Process Data):

Resulting Bit Area Contents

Point	Contents							
RY #7... 0	ADI3[7]	ADI3[6]	ADI3[5]	ADI3[4]	ADI3[3]	ADI3[2]	ADI3[1]	ADI3[0]
RY #15... 8	ADI3[15]	ADI3[14]	ADI3[13]	ADI3[12]	ADI3[11]	ADI3[10]	ADI3[9]	ADI3[8]
RY #23... 16								
RY #31... 24								
RY #39... 32								
RY #47... 40								
RY #55... 48	System Area							
RY #63... 56								

Resulting Word Area Contents:

Point	Contents (LSB)	Contents (MSB)
RWw #0	ADI4[0]	ADI4[1]
RWw #1	ADI4[2]	ADI4[3]
RWw #2	ADI4[4]	ADI4[5]
RWw #3	ADI4[6]	ADI4[7]
RWw #4	ADI2[8]	
RWw #5		
RWw #6		
RWw #7		

4.3 Advanced Mapping (Optional)

This scheme is used when the host application uses the network-specific mapping commands:

- Map_ADI_Specified_Write_Area maps data to the RX (bit) and RWr (word) areas.
- Map_ADI_Specified_Read_Area maps data from the RY (bit) and RWw (word) areas.
- When mapping to the Word Area, each ADI will occupy at least one 16-bit word. The bits are mapped from the least significant bit towards the most significant bit.
- ADIs of BOOL-type can be mapped to both the Bit and Word Area
  - Arrays of BOOL are mapped to consecutive bits
- Non-BOOL ADIs can be mapped to both the Bit and Word Areas under the following conditions:
  - ADIs are always treated as single entities, even if they consist of multiple elements
  - When mapping to the Bit Area, each ADI must start on an even 8-bit boundary.

Example

In this example, Map_ADI_Specified_Read_Area is issued 5 times, labelled A...E:

Command	Data Type	No. of Elements	Target Area	Target Offset
A	UINT16	1	0 (Bit Area)	8
B	BOOL	9	1 (Word Area)	2
C	BOOL	2	0 (Bit Area)	5
D	SINT16	1	1 (Word Area)	8
E	SINT8	3	1 (Word Area)	3

Resulting Bit Area Contents:

Point	Contents							
RY #7... 0	-	C[1]	C[0]	-	-	-	-	-
RY #15... 8	A:LSB							
RY #23... 16	A:MSB							

Resulting Word Area Contents:

Point	Contents (LSB)								Contents (LSB)	
RWw #0	-								-	
RWw #1	-								-	
RWw #2	B[7]	B[6]	B[5]	B[4]	B[3]	B[2]	B[1]	B[0]	-	B[8]
RWw #3	E[0]								E[1]	
RWw #4	E[2]								-	
RWw #5	-								-	
RWw #6	-								-	
RWw #7	-								-	
RWw #8	D								D	

5. CC-Link System Area Implementation

5.1 System Area Modes

An essential part of the CC-Link communication is the CC-Link System Area. This area holds various status- and diagnostic flags, and can either be handled automatically by the Anybus module (default) or by the host application.

- **System Area handled by Anybus (Default, basic)**

  All flags in the System Area are handled automatically by the module, unless this functionality has been explicitly disabled in the 'System Area Handler'-attribute (#5) in the CC-Link Object (F7).

- **System Area handled by Host Application (Advanced)**

  If the 'System Area Handler'-attribute (#5) has been set to -1 (disabled), the host application alone is responsible for handling the CC -Link status flags in accordance with one of the profiles defined in the CC-Link specification. To achieve this, the host application must map one or several ADIs to the corresponding location(s) in the CC-Link memory map.

5.2 System Area Layout

Note: This section is only relevant when the System Area is handled automatically by the module.

Slave -> Master		Master -> Slave	
Bit Offset	**Contents**	**Bit Offset**	**Contents**
0	(reserved)	0	(reserved)
1		1	
2		2	
3		3	
4		4	
5		5	
6		6	
7		7	
8	Initial Data Processing Request	8	Initial Data Processing Complete
9	Initial Data Setting Complete	9	Initial Data Setting Request
10	Error Status	10	Error Reset Request
11	Remote READY	11	(reserved)
12	(reserved)	12	
13		13	
14		14	
15		15	

The various flags listed in the table above are handled as described in B-41 "CC-Link Handshaking Implementation".

See also...

- "System Area Location" on page 20
- "Diagnostics" on page 22
- "CC-Link Handshaking Implementation" on page 41

5.3 System Area Location

Note: This section is only relevant when the System Area is handled automatically by the module.

The default location of the System Area is at the very end of the Bit Area as follows:

Point	Contents	Point	Contents
RX #0	User area (holds Write Process Data)	RY #0	User area (holds Read Process Data)
RX #1		RY #1	
...		...	
RX #Q-18		RY #Q-18	
RX #Q-17		RY #Q-17	
RX #Q-16	(Reserved for CC-Link System Area)	RY #Q-16	(Reserved for CC-Link System Area)
RX #Q-15		RY #Q-15	
...		...	
RX #Q-2		RY #Q-2	
RX #Q-1		RY #Q-1	

(The table above illustrates how data is represented as seen from the CC-Link master. 'Q' represents the number of addressable points in the Bit Area of the ABCC.)

It is possible to change the location of the System Area by implementing the 'System Area Handler'-attribute (#5) in the CC-Link Object (F7h). It is also possible to disable it altogether by setting this attribute to -1. In such case, the host application is responsible for handling the CC-Link communication in consistency with one of the profiles defined in the CC-Link Specification (profile).

See also...

- "Standard Mapping" on page 11
- "CC-Link Handshaking Implementation" on page 41

5.4 Error Code Location

Note: This section is only relevant when the System Area is handled automatically by the module.

If the 'Model Code'-attribute (#3) in the CC-Link Object (F7h) equals 7Fh ('Generic device', default), the last word of the Word Area is reserved for diagnostic functionality as follows:

Point	Contents	Point	Contents
RWr #0	User area (holds Write Process Data)	RWw #0	User area (holds Read process Data)
RWr #1		RWw #1	
...		...	
RWr #Z-2		RWw #Z-2	
RWr #Z-1	(Error Code)	RWw #Z-1	

(The table above illustrates how data is represented as seen from the CC-Link master. 'Z' represents the number of addressable points in the Word Area of the ABCC.)

Please note that the Error Code word **only** exists if the 'Model Code'-attribute (#3) in the CC-Link Object (F7h) is set to 7Fh (default, not set by the CC-Link Host Object) in conjunction with an enabled Automatic System Area Handler. If not, this word will be used for normal data exchange. Furthermore, disabling the System Area (i.e. by setting the 'System Area Handler'-attribute (#5) in the CC-Link Object (F7h) to -1) also disables the Error Code word.

See also...

- "Standard Mapping" on page 11
- "System Area Location" on page 20

5.5 Diagnostics

Note: This section is only relevant when the System Area is handled automatically by the module.

As mentioned previously, the module supports up to 5 diagnostic entries during normal conditions, plus an additional 6th entry in case of a major unrecoverable event.

Diagnostics are represented through the 'Error Status'-flag, the 'Remote READY'-flag, and the 'Error Code'-word.

- **'Error Status'-flag**

  This flag reflects the state of the Diagnostic Object as follow:

  1: Diagnostic events exists[1], see 'Error Code'-word below
  0: No diagnostic events exists - *or* - 'Error Reset Request'-flag high

  See also...
  - "System Area Modes" on page 19
  - "CC-Link Handshaking Implementation" on page 41

- **'Error Code'-word**

  This word holds additional diagnostic information as follows:

  High byte: Severity Code (diagnostic instance attribute #1)
  Low byte: Event Code (diagnostic instance attribute#2)

  The module updates this word on the rising edge of 'Error Status'. With each update, the module cycles through the available events, which means that the master can gather information from several diagnostic events by toggling the 'Error Reset Request'-flag accordingly.

  Events tagged with 'Major' severity are prioritized by the module, which means that if present, only those particular events will be reported to the network. If no 'Major' events are present, the module will report the 'Minor' events (when applicable).

  Note: This word only exists if the 'Model Code'-attribute (#3) in the CC-Link Object (F7h) is set to 7Fh (default). If not, this word will be used for normal data exchange.

  See also...
  - "System Area Modes" on page 19
  - "CC-Link Handshaking Implementation" on page 41

- **'Remote READY'-flag**

  1: Normal operation
  0: Diagnostic event with 'Major' severity exists[2] - *or* - 'Initial Data Setting Request'-flag high

  See also...
  - "System Area Modes" on page 19
  - "CC-Link Handshaking Implementation" on page 41

1. The flag stays high until the master has acknowledged the event through the 'Error Reset Request'-flag.
2. Normal behaviour is resumed when the event has been resolved (i.e. when the host application removes the corresponding diagnostic instance) - and - the master has acknowledged the event through the 'Error Reset Request'-flag (see also Error Status and Error Code above)

6. Anybus Module Objects

6.1 General Information

This chapter specifies the Anybus Module Object implementation and how they correspond to the functionality in the Anybus-CompactCom CC-Link.

The following Anybus Module Objects are implemented:

- "Anybus Object (01h)" on page 24
- "Diagnostic Object (02h)" on page 25
- "Network Object (03h)" on page 26
- "Network Configuration Object (04h)" on page 28
- "Network CC-Link Object (08h)" on page 30

6.2 Anybus Object (01h)

Category

Basic

Object Description

This object groups common Anybus information, and is described thoroughly in the general Anybus-CompactCom Software Design Guide.

Supported Commands

Object:	Get_Attribute
Instance:	Get_Attribute Set_Attribute Get_Enum_String

Object Attributes (Instance #0)

(Consult the general Anybus-CompactCom Software Design Guide for further information.)

Instance Attributes (Instance #1)

Basic

#	Name	Access	Type	Value
1	Module type	Get	UINT16	0401h (Standard Anybus-CompactCom)
2... 11	-	-	-	Consult the general Anybus-CompactCom Software Design Guide for further information.
12	LED colors	Get	struct of: UINT8(LED1A) UINT8(LED1B) UINT8(LED2A) UINT8(LED2B)	Value:Color: 01h Green 02h Red 01h Green 02h Red
13... 15	-	-	-	Consult the general Anybus-CompactCom Software Design Guide for further information.

6.3 Diagnostic Object (02h)

Category

Basic

Object Description

This object provides a standardised way of handling host application events & diagnostics, and is thoroughly described in the general Anybus-CompactCom Software Design Guide.

In the case of CC-Link, diagnostics can be represented through the CC-Link System Area flags, and through a dedicated register in the Word Area. If the module handles the System Areas you have to create/remove one Diagnostics instance to pass the certification. If the host application handles the System Area (advanced) the Diagnostic Object does not have to be implemented.

See also...

- "System Area Modes" on page 19
- "System Area Layout" on page 19
- "Diagnostics" on page 22

Supported Commands

Object:	Get_Attribute Create Delete
Instance:	Get_Attribute

Object Attributes (Instance #0)

#	Name	Access	Data Type	Value
1	Name	Get	Array of CHAR	'Diagnostic'
2	Revision	Get	UINT8	01h
3	Number of instances	Get	UINT16	See general Anybus-CompactCom Software Design Guide
4	Highest instance no.	Get	UINT16	
11	Max no. of instances	Get	UINT16	5+1

6.3.1 Instance Attributes

#	Name	Access	Type	Value
1	Severity	Get	UINT8	See general Anybus-CompactCom Software Design Guide
2	Event Code	Get	UINT8	

6.4 Network Object (03h)

Category

Basic

Object Description

For more information regarding this object, consult the general Anybus-CompactCom Software Design Guide.

See also...

- "Network CC-Link Object (08h)" on page 30

Supported Commands

Object:	Get_Attribute
Instance:	Get_Attribute Set_Attribute Get_Enum_String Map_ADI_Write_Area Map_ADI_Read_Area

Object Attributes (Instance #0)

#	Name	Access	Data Type	Value
1	Name	Get	Array of CHAR	"Network"
2	Revision	Get	UINT8	02h
3	Number of instances	Get	UINT16	0001h
4	Highest instance no.	Get	UINT16	0001h

Instance Attributes (Instance #1)

Basic

#	Name	Access	Type	Value
1	Network type	Get	UINT16	0090h
2	Network type string	Get	Array of CHAR	'CC-Link'
3	Data format	Get	ENUM	00h (LSB first)
4	Parameter data support	Get	BOOL	False
5	Write process data size	Get	UINT16	Current write process data size (in bytes) Updated on each successful Map_ADI_Write_Area[a] and Map_Specified_ADI_Write_Area[b]
6	Read process data size	Get	UINT16	Current read process data size (in bytes) Updated on each successful Map_ADI_Read_Area[a] and Map_Specified_ADI_Read_Area[c]
7	Exception Information	Get	UINT8	See "Exception Information" on page 43

a. Consult the general Anybus-CompactCom Software Design Guide for further information.
b. See "Command Details: Map_ADI_Specified_Write_Area" on page 31
c. See "Command Details: Map_ADI_Specified_Read_Area" on page 32

6.5 Network Configuration Object (04h)

Category

Basic

Object Description

This object holds network specific configuration parameters that may be set by the end user. A reset command (factory default) issued towards this object will result in all instances being set to their default values. The Station Number is set to "Not set" and the Baud Rate to 2 (2.5 Mbps).

See also...

- "Communication Settings" on page 10
- "CC-Link Host Object (F7h)" on page 36

Note: Instances #1 and #2 have to be implemented if the end-product is going to be re-certified according to CC-Link certification policies.

Supported Commands

Object:	Get_Attribute Reset
Instance:	Get_Attribute Set_Attribute Get_Enum_String

Object Attributes (Instance #0)

#	Name	Access	Data Type	Value
1	Name	Get	Array of CHAR	'Network configuration'
2	Revision	Get	UINT8	01h
3	Number of instances	Get	UINT16	0002h
4	Highest instance no.	Get	UINT16	0002h

Instance Attributes (Instance #1, ‘Station number’)

This instance holds the actual CC-Link Station Number.

Basic

#	Name	Access	Type	Description
1	Name[a]	Get	Array of CHAR	‘Station number’
2	Data type	Get	UINT8	04h (= UINT8)
3	Number of elements	Get	UINT8	01h (one element)
4	Descriptor	Get	UINT8	07h (get/set/shared access)
5	Value	Get/Set	UINT8	Value:Setting: 0: Not set (invalid to use) 1... 64: Station Number (see note) >64: (not valid) **Notes:** The sum of the Station Number and the Number of Occupied Stations may not exceed 65.

a. Multilingual, see 6-29 “Multilingual Strings”.

6.5.1 Instance Attributes (Instance #2, ‘Baud rate’)

This instance holds the actual CC-Link data rate.

Basic

#	Name	Access	Type	Description
1	Name[a]	Get	Array of CHAR	‘Baud rate’
2	Data type	Get	UINT8	08h (= ENUM)
3	Number of elements	Get	UINT8	01h (one element)
4	Descriptor	Get	UINT8	07h (get/set/shared access)
5	Value	Get/Set	UINT8	Value:Speed/String: 0: ‘156 kbps’ 1: ‘625 kbps’ 2: ‘2.5 Mbps’ 3: ‘5 Mbps’ 4: ‘10 Mbps’ other: (not valid)

a. Multilingual, see “Multilingual Strings” on page 29.

6.5.2 Multilingual Strings

The instance names in this object are multilingual and are translated based on the current language settings as follows:

Instance	English	German	Spanish	Italian	French
1	Station number	Geräteadresse	Direcc nodo	Indirizzo	Adresse
2	Baud rate	Datenrate	Veloc transf	Velocità dati	Vitesse

6.6 Network CC-Link Object (08h)

Category

Basic, extended

Object Description

-

Supported Commands

Object:	Get_Attribute
Instance:	Get_Attribute Map_ADI_Specified_Write_Area (see "Command Details: Map_ADI_Specified_Write_Area" on page 31) Map_ADI_Specified_Read_Area (see "Command Details: Map_ADI_Specified_Read_Area" on page 32) CCL_Conf_Test_Mode (see "Command Details: CCL_Conf_Test_Mode" on page 34)

Object Attributes (Instance #0)

#	Name	Access	Data Type	Value
1	Name	Get	Array of CHAR	“Network CC-Link”
2	Revision	Get	UINT8	01h
3	Number of instances	Get	UINT16	0001h
4	Highest instance no.	Get	UINT16	0001h

Instance Attributes (Instance #1)

Basic

#	Name	Access	Type	Value
1	Network Settings[a]	Get	Struct of: UINT8 UINT8 UINT8	Current network settings: - CC-Link Version; 01h = v1.10, 02h = v2.0 - Number of Occupied Stations - Number of extension cycles
2	System Area Handler[a]	Get	SINT16	System Area location (or -1 in case the system area is handled by the host application)
3	Error Code Position[a]	Get	SINT8	Location of the ‘Error Code’-word in the Anybus memory map (or -1 in case the system area is handled by the host application, or if module code is unequal to 7Fh (‘Generic device’))
4	Last Mapping Info	Get	Struct of:	-
			UINT8 (Area)	Target Area for the latest ADI mapping Value:Area: 00h Write Bit Area (RX #nn) 01h Write Word Area (RWr #nn) 02h Read Bit Area (RY #nn) 03h Read Word Area (RWw #nn)
			UINT16 (Starting Point)	Point number (nn) in the CC-Link memory map where the latest mapping was placed
			UINT16 (Points)	No. of points mapped by the latest mapping command
5	CC-Link V.2 Conformance test mode	Get	BOOL	TRUE if the special CC-Link version 2 conformance test mode is active.

a. These attributes are calculated automatically by the module unless the host application has specified other values in the CC-Link Object. The attribute values are valid from the first transition to WAIT_PROCESS, and can be used to establish the location of the CC-Link System Area / Error Code during development etc.

Command Details: Map_ADI_Specified_Write_Area

Category

Extended

Details

Command Code: 10h

Valid for: Object Instance

Description

This command is functionally equivalent to Map_ADI_Write_Area, with the exception of two extra parameters with CC-Link specific meaning. These extra parameters specify where the data shall be located in the CC-Link address map. Mixed calls to Map_ADI_Specified_Write_Area and Map_ADI_Write_Area is not permitted.

- **Command details:**

Field	Contents
CmdExt[0]	(See specification for Map_ADI_Write_Area)
CmdExt[1]	
Msg_Data[0]	
Msg_Data[1]	
Msg_Data[2]	
Msg_Data[3]	
Msg_Data[4]	CC-Link Area; 00h = Bit Area, 01h = Word Area
Msg_Data[5]	Offset within the area (low byte)
Msg_Data[6]	Offset within the area (high byte)

- **Response details (Success):**

Field	Contents
Msg_Data[0]	Offset of the mapped ADI from the start of the Write Process Data.

- **Response details (Error):**

Error	Contents
Object Specific Error	Object specific error, see Msg_Data[1] for details: Value:Error: 01h Invalid ADI data type 02h Invalid number of elements 03h Invalid total size 04h Invalid order number 05h Invalid mapping command sequence 06h Invalid CC-Link Area 07h Invalid offset 08h Overlap with other data on bus

See also...

- "Network Object (03h)" on page 26
- Map_ADI_Write (consult the Anybus-CompactCom Software Design Guide)

Command Details: Map_ADI_Specified_Read_Area

Category

Extended

Details

Command Code: 11h

Valid for: Object Instance

Description

This command is functionally equivalent to Map_ADI_Read_Area, with the exception of two extra parameters with CC-Link specific meaning. These extra parameters specify where the data shall be located in the CC-Link address map. Mixed calls to Map_ADI_Specified_Read_Area and Map_ADI_Read_Area is not permitted.

- **Command details:**

Field	Contents
CmdExt[0]	(See specification for Map_ADI_Read_Area)
CmdExt[1]	
Msg_Data[0]	
Msg_Data[1]	
Msg_Data[2]	
Msg_Data[3]	
Msg_Data[4]	CC-Link Area; 00h = Bit Area, 01h = Word Area
Msg_Data[5]	Offset within the area (low byte)
Msg_Data[6]	Offset within the area (high byte)

- **Response details (Success):**

Field	Contents
Msg_Data[0]	Offset of the mapped ADI from the start of the Read Process Data.

- **Response details (Error):**

Error	Contents
Object Specific Error	Object specific error, see Msg_Data[1] for details: Value:Error: 01h Invalid ADI data type 02h Invalid number of elements 03h Invalid total size 04h Invalid order number 05h Invalid mapping command sequence 06h Invalid CC-Link Area 07h Invalid offset 08h Overlap with other data on bus

See also...

- "Network Object (03h)" on page 26
- Map_ADI_Read (consult the Anybus-CompactCom Software Design Guide)

Command Details: CCL_Conf_Test_Mode

Category

Extended

Details

Command Code: 12h

Valid for: Object Instance

Description

This command enables/disables the special CC-Link version 2 conformance test mode. For conformance test of CC-link version 2, it is necessary that a special version of the slave can loop all accepted data from RY/RWw to RX/RWx. This command can activate the functionality temporarily or permanently, or deactivate the functionality permanently. The command may only be issued during SETUP.

- **Command details:**

Field	Contents
CmdExt[0]	Conformance test mode Value: Description: 00h Deactivate the conformance test mode permanently 01h Activate the conformance test mode temporarily. The conformance test mode will not be activated after a reset/power cycle 02h Activate the conformance test mode permanently.
CmdExt[1]	(reserved)

- **Response details (Success):**

Field	Contents
Data[0 .. n]	(not used)

- **Response details (Error):**

Error	Contents
Standard error codes	According to ABCC Functional Specification

7. Host Application Objects

7.1 General Information

This chapter specifies the host application object implementation in the module. The objects listed here may optionally be implemented within the host application firmware to expand the CC-Link implementation.

Standard Objects:

- Application Object (see Anybus-CompactCom Software Design Guide)
- Application Data Object (see Anybus-CompactCom Software Design Guide)

Network Specific Objects:

- "CC-Link Host Object (F7h)" on page 36

7.2 CC-Link Host Object (F7h)

Category

Basic, extended, advanced

Object Description

This object implements CC-Link specific features in the host application.

The implementation of this object is optional; the host application can support none, some, or all of the attributes specified below. The module will attempt to retrieve the values of these attributes during start-up; if an attribute is not implemented in the host application, simply respond with an error message (06h, "Invalid CmdExt[0]"). In such case, the module will use its default value.

If the module attempts to retrieve a value of an attribute not listed below, respond with an error message (06h, "Invalid CmdExt[0]").

Supported Commands

Object:	Get Attribute Initial_Data_Setting_Notification (see "Command Details: Initial_Data_Setting_Notification" on page 38)
Instance:	Get Attribute

Object Attributes (Instance #0)

#	Name	Access	Data Type	Value
1	Name	Get	Array of CHAR	'CC-Link'
2	Revision	Get	UINT8	01h
3	Number of instances	Get	UINT16	0001h
4	Highest instance no.	Get	UINT16	0001h

Instance Attributes (Instance #1)

Basic

#	Name	Access	Type	Default Value	Comment
1	Vendor Code	Get	UINT16	0212h ('HMS')	Assigned by the CLPA
2	SW Version	Get	UINT8	(Anybus firmware release no.)	Valid settings range from 1 to 3Fh.

Extended

#	Name	Access	Type	Default Value	Comment
3	Model Code	Get	UINT8	7Fh ('Generic device')[a]	For possible settings, consult the CC-Link Specification (Profile) for device type codes.[b]
4	Network Settings	Get	Array of:		
			UINT8	01h	CC-Link Version: 01h:v1.10 02h:v2.0
			UINT8	(calculated automatically)	Number of Occupied Stations: Valid settings: 01h, 02h, 03h, 04h
			UINT8	1	Number of extension cycles. Valid settings, CC-Link v1: 01h Valid settings, CC-Link v2: 01h, 02h, 04h, 08h
5	System Area Handler	Get	SINT16	(calculated automatically; last 16 bits in the bit area)	System Area handled by... -1[c]:Host application 0...880:Module. Value specifies the offset of the system area in the bit area. Must be located on an even 16-bit boundary.

a. Due to changes in the CLPA Model Code pool allocation, the default model code has been changed to 7Fh ('Generic device). The change applicesfrom software version 1.05 and onwards. Any application that implements the CC-Link Host Object with default model code will have to be updated.

b. Any other value than default will disable the Error Code word.

c. Advanced

Advanced

#	Name	Access	Type	Default Value	Comment
5	System Area Handler	Get	SINT16	(calculated automatically; last 16 bits in the bit area)	System Area handled by... -1:Host application 0...880[a]:Module. Value specifies the offset of the system area in the bit area. Must be located on an even 16-bit boundary.

a. Extended

Command Details: Initial_Data_Setting_Notification

Category

Advanced

Details

Command Code: 10h

Valid for: Object Instance

Description

Note: This section is only relevant when the System Area is handled automatically by the module.

This command will be issued when the master initiates the Initial Data Setting cycle, i.e. on the rising edge of the 'Initial Data Setting Request'-flag.

The host application may either accept or reject (i.e. by responding with 'Unsupported Object' or 'Unsupported Command') the command; in either case, the module will continue the Initial Data Setting Cycle by setting the 'Initial Data Setting Complete'-flag once the response has been received.

- **Command Details**
  (No data)
- **Response Details**
  (No data)

A. Categorization of Functionality

The objects, including attributes and services, of the Anybus CompactCom and the application are divided into three categories: basic, advanced and extended.

A.1 Basic

This category includes objects, attributes and services that are mandatory to implement or to use. They will be enough for starting up the Anybus CompactCom and sending/receiving data with the chosen network protocol. The basic functions of the industrial network are used.

Additional objects etc, that will make it possible to certify the product also belong to this category.

A.2 Extended

Use of the objects in this category extends the functionality of the application. Access is given to the more specific characteristics of the industrial network, not only the basic moving of data to and from the network. Extra value is given to the application.

A.3 Advanced

The objects, attributes and services that belong to this group offer specialized and/or seldom used functionality. Most of the available network functionality is enabled and accessible. Access to the specification of the industrial network is normally required.

B. Implementation Details

B.1 SUP-Bit Definition

The Supervised-bit (SUP) indicates that the module is exchanging data with the CC-Link master.

B.2 Anybus State Machine

The table below describes how the Anybus State Machine relates to the CC-Link network.

Anybus State	Implementation (System Area handled by Anybus module)	Implementation (System Area handled by Host Application)
WAIT_PROCESS	Automatic CC-Link handshaking in progress	Waiting for the first refresh message
ERROR	Possible causes: - Timeout error (ERR21) - 0 channel carrier detection status (ERR22) - insufficient number of data for the host (ERR30...32) When the problem disappears, the module returns to the previous state.	
PROCESS_ACTIVE	The module enters this state on the rising edge of 'Initial Data Processing Complete'	The module enters this state when the first refresh message has been received
IDLE	PLC/Master in STOP mode	
	When the PLC/Master returns to RUN mode, the module returns to the previous state.	
EXCEPTION	Possible causes: - Configuration error - other error that cannot be indicated to the application in any other way Examples: - Illegal combinations of attribute settings in the CC-Link Object (F7h) - Illegal combinations of mapping commands and attribute settings in the CC-Link Object (F7h)	

B.3 Application Watchdog Timeout Handling

Upon detection of an application watchdog timeout, the module will cease network participation and shift to state 'EXCEPTION'. No other network specific actions are performed.

B.4 CC-Link Handshaking Implementation

Note: This section is only relevant when the System Area is handled automatically by the module.

When the System Area is handled by the Anybus module, all CC-Link handshaking is performed automatically as described in the table below.

Flag	Set when...	Cleared when...
Initial Data Processing Request	State changes from NW_INIT to WAIT_PROCESS	The following sequence has finished: **1. State transition from WAIT_PROCESS to PROCESS_ACTIVE** **2. At least one telegram has been sent to the host application** **3. At last one telegram has been received from the host application** This purpose of this procedure is to ensure that the host application has detected that the module has shifted to the PROCESS_ACTIVE state.
Initial Data Setting Complete	The host application responds to Initial_-Data_Setting_Notification[a]	At negative flank of 'Initial Data Setting Request'
Error Status[b]	When there is at least one Diagnostic Instance[c], and 'Error Reset Request' is false	'Error Reset Request' is set.
Remote READY[d]	(initial setting) - At the rising edge of 'Initial Data Processing Complete'	Either 'Initial Data Setting Request' is set or at least one Diagnostic instance with major severity exist
	(runtime) - When 'Error Status' and 'Error Reset Request' is false	

a. See "Command Details: Initial_Data_Setting_Notification" on page 38
b. Additional functionality is handled through this flag, see "Diagnostics" on page 22
c. See "Diagnostics" on page 22
d. If both the Set and Clear conditions are true, the Clear functionality is given priority

Note: The initial value of the System Area is false, i.e. all flags are cleared during startup.

C. Additional Certification Information

The absolute basics for certification of the Anybus CompactCom CC-Link module is given in "Certification" on page 9. This section includes more information and recommendations when certifying a product.

C.1 Model Code

When the module is delivered, the Model Code (CC-Link Host Object (F7h), Instance 1, Attribute 3) is set to 7Fh ("Generic device"), which is a model code that HMS has had registered with CLPA. If the host application is similar to an existing CC-Link profile, this code should be changed to reflect that profile.

If the value of the Model Code is changed, the Error Code word will not be enabled, see "Error Code Location" on page 21.

Note: Due to changes in the CLPA Model Code pool allocation, the default model code has been changed to 7Fh ('Generic device). The change applies from software version 1.05 and onwards. Any application that implements the CC-Link Host Object with default model code will have to be updated.

C.2 CC-Link Version 2

By implementing the 'Network Settings'-attribute (#4) in the CC-Link Object (F7h) it is possible to customize the implementation for CC-Link version 2 and use larger data sizes through extension cycles.

For conformance test of CC-Link version 2, it is necessary to enable the CC-Link conformance test mode, see "Network CC-Link Object (08h)" on page 30, command CCL_Conf_Test_Mode on page 34.

D. Exception Information

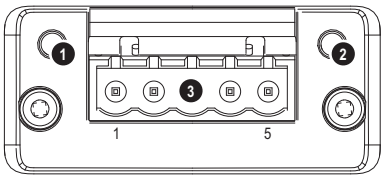
When the module has entered the EXCEPTION-state, further details about the problem can be read from the 'Exception Information'-attribute (#7) in the Network Object (03h)

Value	Meaning
00h	No information.
01h	Value out of range for SW Version attribute of CC-Link object.
02h	Value out of range for CC-Link version in the Network Settings attribute of the CC-Link object.
03h	Value out of range for Number of stations in the Network Settings attribute of the CC-Link object.
04h	Invalid value for Extension cycles in the Network Settings attribute of the CC-Link object.
05h	Invalid value for System area handler attribute of the CC-Link object.
06h	CC-Link version and extension cycles are incompatible.
07h	Data mapped at too high offset in CC-Link map for CC-Link version 1.
08h	Data mapped at offset were System area is automatically located.
09h	No room for automatically located System area in CC-Link version 1.
0Ah	Data mapped at too high offset in CC-Link map for specified network settings.
0Bh	System area location out of range specified in Network settings.
0Ch	System area located at too high offset in CC-Link map for CC-Link version 1.
0Dh	Data mapped at System area location.
0Eh	Data mapped at location where Anybus Error code should be located according to Network settings.
0Fh	Data mapped at location where Anybus Error code should be located in CC-Link version 1.
10h	The set Device address is too high for the used number of occupied stations.

E. Technical Specification

E.1 Front View

#	Item
1	Run LED
2	Error LED
3	CC-Link Interface



Run LED

State	Meaning
Off	- No network participation, timeout status (no power)
Green	- Participating, normal operation
Red	- Major fault (FATAL error)

Error LED

State	Meaning
Off	- No error detected (no power)
Red	- Major fault (Exception or FATAL event)
Red, flickering	- CRC error (temporary flickering)
Red, flashing	- Station Number or Baud rate has changed since startup (flashing)

CC-Link Interface

Pin	Signal	Comment
1	DA	Positive RS485 RxD/TxD
2	DB	Negative RS485 RxD/TxD
3	DG	Signal Ground
4	SLD	Cable Shield
5	FG	Protective Earth

E.2 Protective Earth (PE) Requirements

In order to ensure proper EMC behaviour, the module must be properly connected to protective earth via the PE pad / PE mechanism described in the general Anybus-CompactCom Hardware Design Guide.

HMS Industrial Networks does not guarantee proper EMC behaviour unless these PE requirements are fulfilled.

E.3 Power Supply

Supply Voltage

The module requires a regulated 3.3V power source as specified in the general Anybus-CompactCom Hardware Design Guide.

Power Consumption

The Anybus-CompactCom CC-Link is designed to fulfil the requirements of a Class B module. For more information about the power consumption classification used on the Anybus-CompactCom platform, consult the general Anybus-CompactCom Hardware Design Guide.

The current hardware design consumes up to 280mA[1].

Note: It is strongly advised to design the power supply in the host application based on the power consumption classifications described in the general Anybus-CompactCom Hardware Design Guide, and not on the exact power requirements of a single product.

E.4 Environmental Specification

Consult the Anybus-CompactCom Hardware Design Guide for further information.

E.5 EMC Compliance

Consult the Anybus-CompactCom Hardware Design Guide for further information.

1. Note that in line with HMS policy of continuous product development, we reserve the right to change the exact power requirements of this product without prior notification. Note however that in any case, the Anybus-CompactCom CC-Link will remain as a Class B module.

F. Timing & Performance

F.1 General Information

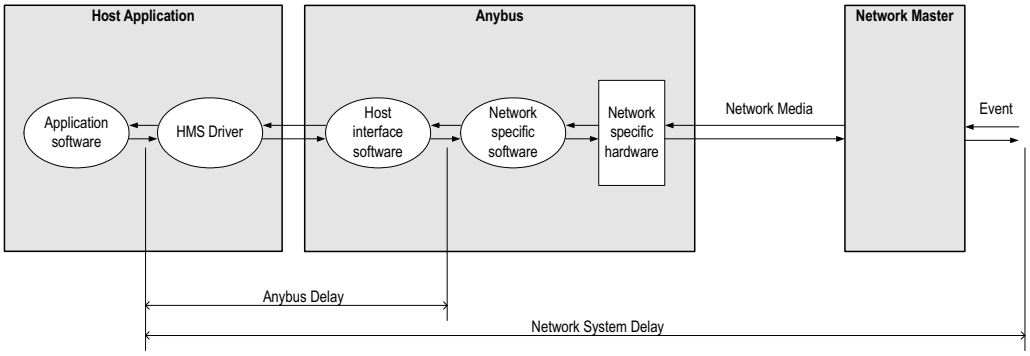
This chapter specifies timing and performance parameters that are verified and documented for the Anybus CompactCom CC-Link.

The following timing aspects are measured:

Category	Parameters	Page
Startup Delay	T1, T2	Please consult the Anybus CompactCom Software Design Guide, App. B.
NW_INIT Delay	T3	
Telegram Delay	T4	
Command Delay	T5	
Anybus Read Process Data Delay (Anybus Delay)	T6, T7, T8	
Anybus Write Process Data Delay (Anybus Delay)	T12, T13, T14	
Network System Read Process Data Delay (Network System Delay)	T9, T10, T11	48
Network System Write Process Data Delay (Network System Delay)	T15, T16, T17	48

F.2 Process Data

F.2.1 Overview



F.2.2 Anybus Read Process Data Delay (Anybus Delay)

The Read Process Data Delay (labelled 'Anybus delay' in the figure above) is defined as the time measured from just before new data is buffered and available to the Anybus host interface software, to when the data is available to the host application (just after the new data has been read from the driver).

Please consult the Anybus CompactCom Software Design Guide, Appendix B, for more information.

F.2.3 Anybus Write Process Data Delay (Anybus Delay)

The Write Process Data Delay (labelled 'Anybus delay' in the figure) is defined as the time measured from the point the data is available from the host application (just before the data is written from the host application to the driver), to the point where the new data has been forwarded to the network buffer by the Anybus host interface software.

Please consult the Anybus CompactCom Software Design Guide, Appendix B, for more information.

F.2.4 Network System Read Process Data Delay (Network System Delay)

The Network System Read Process Data Delay (labelled 'Network System Delay in the figure), is defined as the time measured from the point where an event is generated at the network master to when the corresponding data is available to the host application (just after the corresponding data has been read from the driver).

Parameter	Description	Avg.	Max.	Unit.
T9	Network System Read Process Data delay, 8 ADIs (single UINT8)	-	3.8	ms
T10	Network System Read Process Data delay, 16 ADIs (single UINT8)	-	6.7	ms
T11	Network System Read Process Data delay, 32 ADIs (single UINT8)	-	6.8	ms

Conditions:

Parameter	Conditions
Application CPU	-
Timer system call interval	1 ms
Driver call interval	0.2... 0.3 ms
No.of ADIs (single UINT8) mapped to Process Data in each direction.	8, 16 and 32
Communication	Parallel
Telegram types during measurement period	Process Data only
Bus load, no. of nodes, baud rate etc.	Normal

F.2.5 Network System Write Process Data Delay (Network System Delay)

The Network System Write Process Data Delay (labelled 'Network System Delay in the figure), is defined as the time measured from the time after the new data is available from the host application (just before the data is written to the driver) to when this data generates a corresponding event at the network master.

Parameter	Description	Avg.	Max.	Unit.
T15	Network System Write Process Data delay, 8 ADIs (single UINT8)	3.96	4.88	ms
T16	Network System Write Process Data delay, 16 ADIs (single UINT8)	6.12	7.72	ms
T17	Network System Write Process Data delay, 32 ADIs (single UINT8)	10.1	13.3	ms

Conditions: as in "Network System Read Process Data Delay (Network System Delay)" on page 48.