Concept
User Manual

Volume 3

840 USE 503 00 eng Version 2.6 - SR1

33002204.00

Table of Contents

The chapters marked gray are not included in this volume.

About the Book

At a Glance

Document Scope

This user manual is intended to help you create a user program with Concept. It provides authoritative information on the individual program languages and on hardware configuration.

Validity Note

The documentation applies to Concept 2.6 for Microsoft Windows 98, Microsoft Windows 2000, Microsoft Windows XP and Microsoft Windows NT 4.x.

Note: Additional up-to-date tips can be found in the Concept README file.

Related Documents

Title of Documentation	Reference Number
Concept Installation Instructions	840 USE 502 00
Concept IEC Block Library	840 USE 504 00
Concept EFB User Manual	840 USE 505 00
Concept LL984 Block Library	840 USE 506 00

User Comments

We welcome your comments about this document. You can reach us by e-mail at TECHCOMM@modicon.com

IEC conformity

D

At a Glance

Overview

This Chapter contains the standards tables required by IEC 1131-1.

What's in this Chapter?

This chapter contains the following sections:

D.1 What is the IEC 1131-3 standard?

At a Glance

Overview

This section contains general information about IEC 1131-3 and the implemented IEC conformity test.

What's in this Section?

This section contains the following topics:

General information about IEC conformity

At a Glance

The IEC standard 1131-3 (compare chapter 1.4) specifies the syntax and semantics of a standardized series of programming languages for Programmable Logic Controls (PLC). These include the two text languages IL (Instruction List) and ST (Structured Text) and the two graphical languages LD (Ladder Diagram) and FBD (Function Block Diagram).

It also defines the elements of the sequential function chart (SFC) language for structuring the internal organization of PLC programs and Function Blocks. Configuration elements, used for installing PLC programs onto PLC systems, are also defined.

Note: Concept uses the English acronyms for the programming languages.

Furthermore, it defines methods to enable communication between the PLC and other automated system components.

Concept standard accordance

In accordance with the standard, the present version of the programming system Concept supports a subset of language elements, which are defined in the standard.

In this context, accordance with the standard means the following:
- The standard allows the individual implementing an IEC program system to select or deselect certain language properties or even complete languages from the selection tables, which represent an integrated part of the standard specifications. A system, which itself accords with the standard, may only implement the selected properties exactly as they are given in the standard.
- In addition, the standard enables the individual implementing to introduce defined language elements into an interactive programming environment. As the standard expressly emphasizes that the specification of such environments lies outside of its area of application, the person implementing has a certain degree of freedom to offer optimized forms of display and implementation mechanisms for the benefit of the user.
- Concept uses these degrees of freedom e.g. when introducing the term "Project" to implement the IEC language elements "Configuration", "Resource" and "Program" all together (Concept only supports one single cyclically running program within a single resource within the configuration). Apart from this, it uses them, for example, with implementation mechanisms made available for declaring variables and authorizing Function Blocks.

IEC standards tables

Information on which properties are supported and other implementation specific details can be found in the following statements on standard fulfilment and the associated standards tables.

IEC Conformity Test

Testing the Import/Export Interface

An interface for importing standard IEC programs and DFBs from ASCII files (menu **File** → **Import**) and exporting these programs into graphical languages in ASCII format (menu **File** → **Export**) is available in Concept. The conformity of this interface can be tested using files which can be obtained from IFAK (Institut für Automation und Kommunikation e.V. Magdeburg).

IEC conformity test scripts:
(c) 1994, IFAK Institut für Automation und Kommunikation e.V.
Magdeburg
Steinfeldstraße 3
D-39179 Barleben

Notes

The following points must be considered with regard to the conformity of the import interface:

- In Concept, IL operators are permitted as identifiers.
  R, S, LD, S1 and R1 are possible parameter names. Therefore, there will be no changes made to the standard functions/function blocks. Concept requires no change in the IEC table 54 with S to SET, R to RESET, S1 to SET1, R1 to RESET1.
- All IL operators not in conflict with functions are permitted as variable names in Concept (N, S, R, S1, R1, CLK, CU, CD, PV, IN, PT) – contrary to IEC table 54.
- Counter EFBs must be typified in Concept, e.g. CTU must become CTU_INT.
- Function block instances cannot be called up more than once; a restriction that is self-evident if IEC table 53, property 3 is required.
- An overflow of time span variables (e.g. t#100s) is not detected. The system calculates the time correctly, so that detection of an overflow is not necessary.
- IEC IL comments are only permitted as the last element in a line. Concept allows comments to be made everywhere.

D.2 IEC standards tables

At a Glance

Overview

This system fulfils the requirements of the IEC 1131-3 in the following properties of the language.

What's in this Section?

This section contains the following topics:

Common elements

IEC standards table

IEC standards table for common elements:

Table number	Property number	Property description
1	1	For required character set – see *Chapter 2.1.1* of 1131-3
1	2	Lower case characters
1	3a	Hash key (#)
1	4a	Dollar sign ($)
1	5a	Vertical line (|)
1	6a	Left and right square brackets "[ ]"
2	1	Upper case character and numbers
2	2	Upper and lower case characters, numbers, embedded underscore
2	3	Upper and lower case characters, numbers, leading and embedded underscore
3	1	Comments
4	1	Integer (whole number) literals
4	2	Real literals
4	3	Real literals with exponents
4	4	Base 2 literals
4	5	Base 8 literals
4	6	Base 16 literals
4	7	Boolean zero and one
4	8	Boolean FALSE and TRUE
7	1a	Time span without underscores: short prefix
7	1b	Time span without underscores: long prefix
7	2a	Time span with underscores short prefix
7	2b	Time span with underscores long prefix
10	1	BOOL: Boolean
10	3	INT: Integer
10	4	DINT: Double integer
10	7	UINT: Signed integer
10	8	UDINT: Signed double integer
10	10	REAL: Floating point number
10	12	TIME: Time span

Table number	Property number	Property description
10	17	BYTE: Bit sequence 8
10	18	WORD: Bit sequence 16
12	4	Data types for fields
12	5	Data types for structures
15	1	I: Input (*Note 1, p. 789*)
15	2	Q: Output (*Note 2, p. 789*)
15	4	X: Bit size (*Note 2, p. 789*, *Note 1, p. 789*)
15	5	no prefix: Bit size (*Note 2, p. 789*, *Note 1, p. 789*)
15	6	B: Byte size (*Note 2, p. 789*, *Note 1, p. 789*)
15	7	W: Word size (*Note 2, p. 789*, *Note 1, p. 789*)
15	8	D: Double word size (*Note 2, p. 789*, *Note 1, p. 789*)
17	2	Declaration of directly displayed buffered variables (*Note 5, p. 790*, *Note 9, p. 790*)
17	3	Declaration of storage locations with symbolic variables (*Note 5, p. 790*)
17	4	Assignment of storage locations with fields (*Note 5, p. 790*, *Note 11, p. 791*)
17	5	Automatic storage allocation for symbolic variables (*Note 5, p. 790*)
17	7	Declaration for buffered fields (*Note 5, p. 790*, *Note 11, p. 791*)
17	8	Declaration for structured variables (*Note 5, p. 790*)
18	2	Initialization of directly displayed buffered variables (*Note 5, p. 790*, *Note 9, p. 790*, *Note 10, p. 791*)
18	3	Assignment of storage locations and start values for fields(*Note 5, p. 790*)
18	4	Assignment of storage locations and start values for fields (*Note 5, p. 790*, *Note 11, p. 791*)
18	5	Initialization of symbolic variables (*Note 5, p. 790*)
18	7	Declaration and initialization of buffered variables (*Note 5, p. 790*, *Note 11, p. 791*)
18	8	Initialization of structured variables (*Note 5, p. 790*)
18	9	Initialization of constants
19	1	Negated input
19	2	Negated output
20	1	Use of "EN" and "ENO" - REQUIRED for LD (*Note 6, p. 790*)
20	2	Use of "EN" and "ENO" – OPTIONAL for FBD

Table number	Property number	Property description
20	3	FBD without "EN" and "ENO"
21	2	Standardized functions (*Note 3, p. 790*)
22	1	(*-TO-**) Type conversion functions (*Note 4, p. 790*
22	2	Truncation towards zero: TRUNC (*Note 3, p. 790*))
23	1	ABS: Absolute value
23	2	SQRT: Square root
23	3	LN: Natural logarithm
23	4	LOG: Base 10 logarithm
23	5	EXP: Exponential function
23	6	SIN: Sine, input in radians
23	7	COS: Cosine, input in radians
23	8	TAN: Tangent, input in radians
23	9	ASIN: Arc sine, principal value
23	10	ACOS: Arc cosine, principal value
23	11	ATAN: Arc tangent, principal value
24	12	ADD: Add
24	13	MUL: Multiply
24	14	SUB: Subtract
24	15	DIV: Divide
24	16	MOD: Modulo
24	17	EXPT: Exponentiation
24	18	MOVE: Assignment
25	1	SHL: move to the left
25	2	SHR: Move to the right
25	3	ROR: Rotate to the right
25	4	ROL: Rotate to the left
26	5	AND: LLogical And
26	6	OR: Logical Or
26	7	XOR Logical exclusive Or
26	8	NOT: Negation
27	1	SEL: Binary selection
27	2a	MAX: Extendable maximum
27	2b	MIN: Extendable minimum
27	3	LIMIT: Limit

Table number	Property number	Property description
27	4	MUX: Extendable multiplexer
28	5	GT: Falling sequence
28	6	GE: Monotonic sequence (decreasing)
28	7	EQ: Equality
28	8	LE: Monotonic sequence (increasing)
28	9	LT: Rising seqence
28	10	NE: Inequality
30	1	ADD: Adding TIME to TIME
30	4	SUB: Subtracting TIME from TIME
30	10	MUL: Multiplying TIME by ANY_NUM
30	11	DIV: Dividing TIME by ANY_NUM
33	1	RETAIN identifier for internal variables (*Note 5, p. 790*)
33	2	RETAIN identifier for output variables (*Note 5, p. 790*)
33	3	RETAIN identifier for internal Function Blocks (*Note 5, p. 790*)
34	1	Bistable Function Block (set priority)
34	2	Bistable Function Block (reset priority)
35	1	Detecting the rising edge
35	2	Detecting the falling edge
36	1	Up counter
36	2	Down counter
36	3	Up/Down counter
37	1	TP: Pulse (timer)
37	2a	TON: Switch-on delay
37	3a	TOF: Switch-off delay
39	1	RETAIN identifier for internal variables (*Note 5, p. 790*)
39	2	RETAIN identifier for output variables (*Note 5, p. 790*)
39	3	RETAIN identifier for internal Function Blocks (*Note 5, p. 790*)
39	14	Assignment of storage locations with fields (*Note 5, p. 790*)
39	18	Assignment of storage locations and start values for fields(*Note 5, p. 790*)
39	19	Use of directly displayed variables (*Note 2, p. 789*, *Note 1, p. 789*)
40	1	Step/Start step – graphical form with directional links
40	2	Step/Start step – text form without directional links (Note 8)
40	3a	Step marker – general form

Table number	Property number	Property description
40	4	Step time elapsed – general form
41	1	Transition condition in ST language within the graphic (*Note 8, p. 790*)
41	5	Transition condition in ST language – textual reference (*Note 9, p. 790*)
41	6	Transition condition in IL language – textual reference (*Note 9, p. 790*)
41	7	Use of the transition name
41	7b	Transition condition in FBD language
41	7c	Transition condition in IL language
41	7d	Transition condition in ST language
42	1	Each Boolean variable can be an action
43	1	Action block
43	2	Concatenated action blocks
43	3	Step body in text form (*Note 8, p. 790*)
44	1	Identifier
44	2	Action name
45	1	Not saved (no identifier)
45	2	N: not saved
45	3	R: Overriding reset
45	4	S: Set (saved)
45	5	L: Time limited
45	6	D: Delayed
45	7	P: Pulse
45	9	DS: Delayed and saved
46	1	Simple string
46	2a	Branching in string selection (priority from left to right)
46	3	Merging a string selection
46	4	Parallel strings - branch and merge
46	5a	String jump (priority from left to right)
46	6a	String loop (priority from left to right)

Note 1

Modicon TSX Quantum Präfix 3 is used in the prefix IB, ID position in all graphical languages.

Note 2

Modicon TSX Quantum Präfix 4 is used in the prefix QB, QD position in all graphical languages.

Note 3

The following functions are overloaded with reference to the data which is selected, multiplexed or assigned; the type statement refers to the selection parameters.

List of overloaded functions:
- SEL
- MUX
- MOVE

All other functions are standardized, e.g. REAL_TRUNC_INT.

Note 4

List of type conversion functions:
- BOOL_TO_BYTE, BOOL_TO_DINT, BOOL_TO_INT, BOOL_TO_REAL, BOOL_TO_TIME, BOOL_TO_UDINT, BOOL_TO_UINT, BOOL_TO_WORD,
- BYTE_TO_BOOL, BYTE_TO_DINT, BYTE_TO_INT, BYTE_TO_REAL, BYTE_TO_TIME, BYTE_TO_UDINT, BYTE_TO_UINT, BYTE_TO_WORD,
- DINT_TO_BOOL, DINT_TO_BYTE, DINT_TO_INT, DINT_TO_REAL, DINT_TO_TIME, DINT_TO_UDINT, DINT_TO_UINT, DINT_TO_WORD,
- INT_TO_BOOL, INT_TO_BYTE, INT_TO_DINT, INT_TO_REAL, INT_TO_TIME, INT_TO_UDINT, INT_TO_UINT, INT_TO_WORD,
- REAL_TO_BOOL, REAL_TO_BYTE, REAL_TO_DINT, REAL_TO_INT, REAL_TO_TIME, REAL_TO_UDINT, REAL_TO_UINT, REAL_TO_WORD,
- TIME_TO_BOOL, TIME_TO_BYTE, TIME_TO_DINT, TIME_TO_INT, TIME_TO_REAL, TIME_TO_UDINT, TIME_TO_UINT, TIME_TO_WORD,
- UDINT_TO_BOOL, UDINT_TO_BYTE, UDINT_TO_DINT, UDINT_TO_INT, UDINT_TO_REAL, UDINT_TO_TIME, UDINT_TO_UINT, UDINT_TO_WORD,
- UINT_TO_BOOL, UINT_TO_BYTE, UINT_TO_DINT, UINT_TO_INT, UINT_TO_REAL, UINT_TO_TIME, UINT_TO_UDINT, UINT_TO_WORD,
- WORD_TO_BOOL, WORD_TO_BYTE, WORD_TO_DINT, WORD_TO_INT, WORD_TO_REAL, WORD_TO_TIME, WORD_TO_UDINT, WORD_TO_UINT

The consequences of each conversion are described in the block library and the help texts, which are available for the library of IEC standard functions.

Note 5

The RETAIN identifier is implicitly required; no language elements displayed in non-buffered memory areas are supported.

Note 6

"EN" and "ENO" are offered as standard; they can, however, be hidden and any other input or output of data type BOOL can be used for links.

Note 7

Expressions are restricted to individual simple Boolean variables.

Note 8

Only available on import of IEC text form in graphical SFC representation.

Note 9

Only available in textual declaration in IL or ST sections.

Note 10 Initialization only possible for non Boolean outputs.

Note 11 Declaration of field variables only possible when using previously defined field data type names.

IL (AWL) language elements

IEC standards table

IEC standards table for IL (AWL) language elements:

Table number	Property number	Property description
52	1	LD operator: sets the current result to that of the operand
52	2	ST operator: saves the current result to the operand address
52	3	S operator: sets Boolean operands to "1" R operator: sets Boolean operands to "0"
52	4	AND operator
52	6	OR operator
52	7	XOR operator
52	8	ADD operator
52	9	SUB operator
52	10	MUL operator
52	11	DIV operator
52	12	GT operator: Comparison >
52	13	GE operator: Comparison >=
52	14	EQ operator: Comparison =
52	15	NE operator: Comparison <>
52	16	LE operator: Comparison <=
52	17	LT operator: Comparison <
52	18	JMP operator: Jump to tag (*Note 1, p. 791*)
52	19	CAL operator: Calls Function Block
52	21	Closing bracket ")": Editing deferred operations
53	1	CAL operator with list of input parameters
53	2	CAL operator with loading/saving of input parameters

Note 1 Jumps are only allowed within sections, not across section boundaries.

Note 2

The following keywords are not available:
- TYPE...END_TYP
- VAR_INPUT...END_VAR
- VAR_OUTPUT...END_VAR
- VAR_IN_OUT...END_VAR
- VAR_EXTERNAL...END_VAR
- FUNCTION...END_FUNCTION
- FUNCTION_BLOCK...END_FUNCTION_BLOCK
- PROGRAM...END_PROGRAM
- STEP...END_STEP
- TRANSITION...END_TRANSITION
- ACTION...END_ACTION
- SEGMENT_SCHEDULER
- RET
- &

ST language elements

IEC standards table

IEC standards table for ST language elements:

Table number	Property number	Property description
55	1	Placing in brackets: (Expression)
55	2	Function calls: Function name (list of arguments)
55	3	Exponentiation: **
55	4	Negation: -
55	5	Complement: NOT
55	6	Multiplication: *
55	7	Division: /
55	8	Modulo: MOD
55	9	Addition: +
55	10	Subtraction: -
55	11	Comparison: <, >, <=, >=
55	12	Equality: =
55	13	Inequality: <>
55	14	Boolean AND: &
55	15	Boolean AND: AND
55	16	Boolean exclusive OR: XOR
55	17	Boolean OR: OR
56	1	Assignment
56	2	Function Block calls and use of FB outputs
56	4	IF instruction
56	5	CASE instruction
56	6	FOR instruction
56	7	WHILE instruction
56	8	REPEAT instruction
56	9	EXIT instruction
56	10	Empty instruction

Note 1

The following keywords are not available:
- TYPE...END_TYP
- VAR_INPUT...END_VAR
- VAR_OUTPUT...END_VAR
- VAR_IN_OUT...END_VAR
- VAR_EXTERNAL...END_VAR
- FUNCTION...END_FUNCTION
- FUNCTION_BLOCK...END_FUNCTION_BLOCK
- PROGRAM...END_PROGRAM
- STEP...END_STEP
- TRANSITION...END_TRANSITION
- ACTION...END_ACTION
- SEGMENT_SCHEDULER
- RETURN

Common graphic elements

IEC standards table

IEC standards table for common graphic elements:

Table number	Property number	Property description
57	2	Horizontal lines: Graphic or semi-graphic
57	4	Vertical lines: Graphic or semi-graphic
57	6	Horizontal/vertical connection: Graphic or semi-graphic
57	8	Line intersection without connection: Graphic or semi-graphic
57	10	Connected and unconnected corners: Graphic or semi-graphic
57	12	Blocks with connecting lines: Graphic or semi-graphic

LD (KOP) language elements

IEC standards table

IEC standards table for LD (KOP) language elements:

Table number	Property number	Property description
59	1	Left power rail (with linked horizontal connection)
60	1	Horizontal connection
60	2	Vertical connection (with linked horizontal connections)
61	1	Closer
61	3	Opener
61	5	Contact for detection of positive transition
61	7	Contact for detection of negative transition
62	1	Coil (*Note 1, p. 795*)
62	2	Negative coil (*Note 1, p. 795*)-{}-
62	3	SET coil (*Note 1, p. 795*)
62	4	RESET coil (*Note 1, p. 795*)
62	8	Coil for detection of positive transition
62	9	Coil for detection of negative transition

Note 1

In start behavior of PLCs there is a distinction between cold starts and warm starts:
- **Cold start**
  Following a cold start (loading the program with **Online** → **Load**) all variables (irrespective of type) are set to "0" or, if available, their initial value.
- **Warm start**
  In a warm start (stopping and starting the program or **Online** → **Load changes**) different start behaviors are valid for located variables/direct addresses and unlocated variables:
  - **Located variables/direct addresses**
    In a warm start all 0x, 1x and 3x registers are set to "0" or, if available, their initial value.
    4x registers retain their current value (storage behavior).
  - **Unlocated variables**
    In a warm start all unlocated variables retain their current value (storing behavior).

This varying behavior in a warm start leads to peculiarities in the warm start behavior of set and reset functions.

- **Set and Reset in LD and IL**
  Warm start behavior is dependent on the variable type used (storage behavior in use of unlocated variables; non storage behavior in use of located variables/ direct addresses)
- **SR and RS Function Blocks in FBD, LD, IL and ST**
  These Function Blocks work with internal unlocated variables and therefore always have a storage behavior.

Implementation-dependent parameters

IEC standards table

IEC standards table for implementation-dependent parameters:

Parameters	Threshold values/behavior
Error-handling procedure	See *Error causes, p. 799* & EFB help
National characters used	All characters in the Windows ANSI character set are supported.
Maximum length of identifiers	Program name: 8 Formal parameter names: 8 DFB type names: 8 EFB type names: 17 Data type names: 24 all others: 32
Maximum comment length:	Limited only by Windows resources
Range of values for time span literals	0s to 49d_17h_2m_47.295s
Range of values for variables of type TIME	0s to 49d_17h_2m_47.295s
Accuracy of the seconds display with types TIME_OF_DAY and DATE_AND_TIME	not applicable
Maximum number of field indices	Practically no limit
Maximum field size	64 kB
Maximum number of structure elements	Only limited by Windows or PLC resources
Maximum structure size	64 kB
Maximum number of variables per declaration	Only limited by Windows or PLC resources
Maximum number of enumerated values	not applicable
Default maximum length of STRING variables	not applicable
Maximum authorized length of STRING variables	not applicable

Parameters	**Threshold values/behavior**
Maximum number of hierarchy tiers	1
Configured or physical illustration	Configured illustration, physical illustration through separate I/O projection
Parameters	Threshold values/behavior
Maximum number of indices	Practically no limit
Maximum range of index values	Range of data type INT
Maximum number of structure levels	Only limited by Windows or PLC resources
Initialization of system inputs	System zero; no user-definable start values
Maximum number of variables per declaration	Only limited by Windows or PLC resources
Information for the determination of execution times of program organization units	In preparation
Methods of function display (names or symbols)	Names
Maximum number of function specifications	not applicable
Maximum number of inputs for extendable functions	32
Type conversion accuracy	See EFB help
Accuracy of functions of a variable	INTEL floating point processor or emulator
Arithmetic function implementation	INTEL floating point processor or emulator
Maximum number of Function Block specifications	Only limited by Windows or PLC resources
Maximum number of Function Block authorizations	512 per section; number of sections per program organization unit is only limited by Windows or PLC resources
Pvmin, Pvmax of counters	Limited by rangess of the INT or DINT data types
Effect of a change in the value of a PT input during a time measurement operation	Directly affects the timer’s default time
Program size limits	Only limited by available PLC memory
Time behavior and porting effects of the execution control elements	The execution of SFC networks in different sections occurs sequentially, in the order given in these sections.

Parameters	Threshold values/behavior
Accuracy of elapsed step time	10 ms
Maximum number of steps per SFC	Limited by the available area for entering characters within the section; number of sections per program organization unit only limited by Windows or PLC resources; the upper limit for the total number of objects per SFC is 2000
Parameters	Threshold values/behavior
Maximum number of transitions per SFC and per step	Limited by the available area for entering characters within the section; number of sections per program organization unit only limited by Windows or PLC resources; the upper limit for the total number of objects per SFC is 2000
Action control mechanism	Functionally equivalent to the specification in the standard
Maximum number of actions per step	Only limited by Windows or PLC resources
Graphical display of the step situation	Green = active Red = inactive
Transition switch time	Of the magnitude of 10 ms
Maximum width of branches/connections	Limited by the available area for entering characters 32
Contents of the RESOURCE libraries	See EFB libraries & help
Maximum number of tasks	1
Task interval resolution	not applicable
Pre-justified and non pre-justified schedules	not applicable
Maximum length of expressions	Practically no limit
Partial evaluation of Boolean expressions	no partial evaluation
Maximum length of instructions	Practically no limit
Maximum number of CASE selections	Practically no limit
Value of the control variables on completion of FOR loops	undefined
Graphic/semi-graphic display	Graphic
Network topology restrictions	no restrictions

Parameters	Threshold values/behavior
Evaluation sequence of feedback loops	Within a network, the starting point of the FFB execution sequence is determined by the "single" available feedback variable
Means of specifying the network execution sequence	1: Execution sequence of program organization unit sections 2: The network execution sequence can be changed within sections; this is done by using a menu command to switch between the execution sequences of two selected FFB items

Error causes

IEC standards table

IEC standards table for error causes:

Error cause	Handling (see *Note 1, p. 800*)
Variable value exceeds the specified range	not applicable
Initialization list length and number of field elements do not agree	2) Error message during programming
Incorrect use of directly displayed or external variables in functions	not applicable
Type conversion error	4) Error message during execution
Numerical result exceeds the range for data type	4) Error message during execution
Division by zero	4) Error message during execution
Mixed input data types in a selection function	2) Error message during programming
Selector (K) outside MUX function range	4) Error message during execution
Invalid character position	not applicable
Result exceeds maximum sequence length	not applicable
Numerical result exceeds the range for data type	4) Error message during execution
Zero or more than one starting step in SFC network	3) Error message during analysis/loading/ connection
User program attempting to change step situation or step time	2) Error message during programming

Error cause	**Handling (see *Note 1, p. 800*)**
Simultaneously completed transitions without priority in a selection branch	not applicable
Side effects of evaluation of a transition condition	3) Error message during analysis/loading/ connection
Action control error	1) Error not reported
Unsafe or unreachable SFCs	3) Error message during analysis/loading/ connection
Data type conflict in VAR_ACCESS	not applicable
Tasks demanding too many processor resources	3) Error message during analysis/loading/ connect
Scan time overrun	4) Error message during execution
Error cause	Handling (see note 1)
Further task schedule conflicts	not applicable
Numerical result exceeds the range for data type	4) Error message during execution
Division by zero	4) Error message during execution
Invalid data type for operation	3) Error message during analysis/loading/ binding
Return from function without assigned value	not applicable
Occurrence arrives at no outcome	4) Error message during execution
The same identifier as connector tag and element name use	not applicable
Non-initialized feedback variable (initialized with system zero)	1) Error not reported

Note 1

Identification for the handling of error causes according to IEC 1131-3, chapter 1.5.1, d):
- 1) Error not reported
- 2) Error message during programming
- 3) Error message during analysis/loading/binding
- 4) Error message during execution

D.3 Expansions of IEC 1131-3

Expansions of IEC 1131-3

At a Glance

The Concept programming environment makes the construct of the so-called section available in all programming languages permitting the subdivision of a program organization unit. This construct provides the opportunity to mix several languages in the body of a POU (e.g. FBD sections, SFC sections), a property, which, if used for this purpose, represents an expansion of the IEC syntax. Sections do not generate their own name space; the name space for all language elements is the POU.

Sections appearing in the body of a POU written only in the FBD language are not to be viewed as an expansion, rather as a permitted means of specifying the execution sequences of several FBD networks furnished with tags, as specified in the corrigendum to 1131-3.

Purpose of sections

Sections serve various purposes

- Sections permit the functional division of an expansive POU body: The body of a POU can be divided into sensible functional parts. The section list represents a kind of functional table of contents for a large, otherwise unstructured POU body.
- Sections permit the graphical division of an expansive POU body: in accordance with an intentionally graphic form of representation, sub-structures of an expansive body can be established. Smaller or larger partial structures may be chosen.
- The division of an expansive POU body enables faster online changes: the section serves as the unit for online changes in Concept. If the POU body is changed in various places during the program runtime, all sections affected by the changes are taken into account if explicitly initiated reloading occurs.
- Sections permit the execution sequence to influence particular marked parts of the POU body: the section name serves as a marking for the part of the body contained in the section, and the execution sequence of the sections can be changed by ranking the sections (see also the last part of the "implementation-dependent parameters" table for information on the execution sequence of networks in the FBD language).
- Sections permit the parallel use of different languages in the same POU: this property is a considerable expansion of the syntax of the IEC 1131-3 standard, which only permits the use of a single IEC language for a POU body. Only the SFC language also provides the opportunity to formulate parts of the body in different languages, because transitions and actions can be expressed in any language, in as far as the corresponding properties are supported by the programming system.

D.4 Text language syntax

Text Language Syntax

Description

The programming system Concept supports the complete language syntax, as specified in appendix B of the IEC language standard 1131-3, with the following exceptions:

- Syntax productions in appendix B of 1131-3, belonging to properties, which according to the IEC standards tables in *IEC standards tables, p. 784* in this document are not supported by Concept, are not implemented.
- The use of some Concept supported properties is, according to the associated remarks in the IEC standards table, only possible in a restricted or modified form. The associated syntax productions are therefore only occasionally or somewhat differently implemented.
- Concept supports the NOT Operator for inverting Boolean battery content in IL.
- The implementation of some faulty syntax productions in appendix B of 1131-3, improved upon either in the corrigendum to 1131-3 or in the planned amendment to 1131-3, uses the suggestions in these documents for orientation.
  The improved productions are implemented in Concept as follows (chapter numbers refer to appendix B of 1131-3):
  - **B.1.3.3:**
    ```
    array_initialization ::= '[' array_initial_elements {','
    array_initial_elements} ']'

    initialized_structure ::= structure_type_name [':='
    structure_initialization]
    ```
  - **B.2.1:**
    ```
    il_operand_list ::= il_operand [',' [EOL] il_operand]

    il_fb_call ::= ('CAL' | 'CALC' | 'CALCN') fb_name '('
    il_operand_list ')'
    ```
  - **B.2.2:**
    ```
    il_operator ::= 'LD' | 'LDN' | 'ST' | 'STN' | 'S' | 'R'
    | ('AND' | 'ANDN' | 'OR' | 'ORN' | 'XOR' | 'XORN') ['(']
    | ('ADD' | 'SUB' | 'MUL' | 'DIV') ['(']
    | ('GT' | 'GE' | 'EQ' | 'NE' | 'LT' | 'LE') ['(']
    | 'JMP' | 'JMPC' | 'JMPCN' | ')' | function_name
    ```

Configuration examples

E

At a Glance

Overview

This section contains various configuration examples, given as step-by-step instructions.

What's in this Chapter?

This chapter contains the following sections:

E.1 Quantum Example - Remote Control with RIO

Introduction

Overview

This Chapter contains the step-by-step process for the configuration of remote control with RIO (**R**emote **I**/**O**).

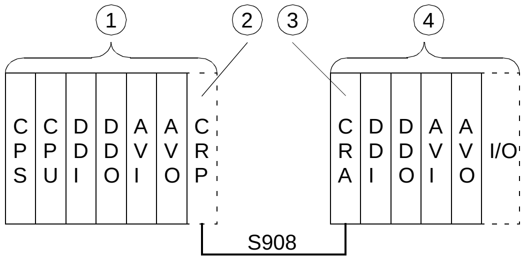
What's in this Section?

This section contains the following topics:

Editing local drop

Introduction

This section describes the configuration of the first (local) drop. The processing sequence begins first of all with the definition of all drops.
When editing the first (local) drop the modules must be set with their I/O references before the individual modules can be parameterized.
Quantum – remote controller with RIO



1 Local Quantum drop 1

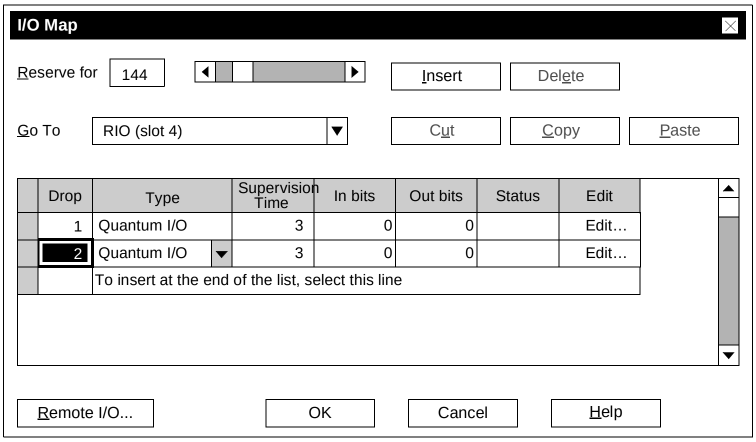
2 RIO master module

3 RIO slave module

4 RIO drop 2

Defining Drops

To define drops proceed as follows in the **PLC Configuration** window:

Step	Action
1	Select **PLC Selection**. **Response:** The **PLC selection** dialog is opened.
2	Select the **Quantum** PLC family and a **CPU x113 xx**. Using **OK** return to the **PLC Configuration** window.
3	Select **I/O Map**. **Response:** The **I/O Map** dialog is opened and the first drop is automatically entered in the table.
4	Select the **Head Setup** command button. **Response:** The **Head Setup** dialog is opened.
5	Enter a 7 in **RIO Slot** and quit the dialog using **OK**. **Response:** The CRP-93x-00 module is automatically inserted in the component list (in slot 7) of the selected drop. In the **Go To** list box, the **Local/RIO (Slot 7)** network link is displayed.
6	Select the last line in the table. Select the **Insert** command button. **Response:** The second drop is entered in the **Type** column. **Note:** The number of drops to be inserted is defined in the **Segment scheduler** dialog. The default predetermines a maximum number of 32. Dialog display 
7	Select the drop from the **Drop** column. Select the **Edit...** command button. **Response:** You reach the module map.

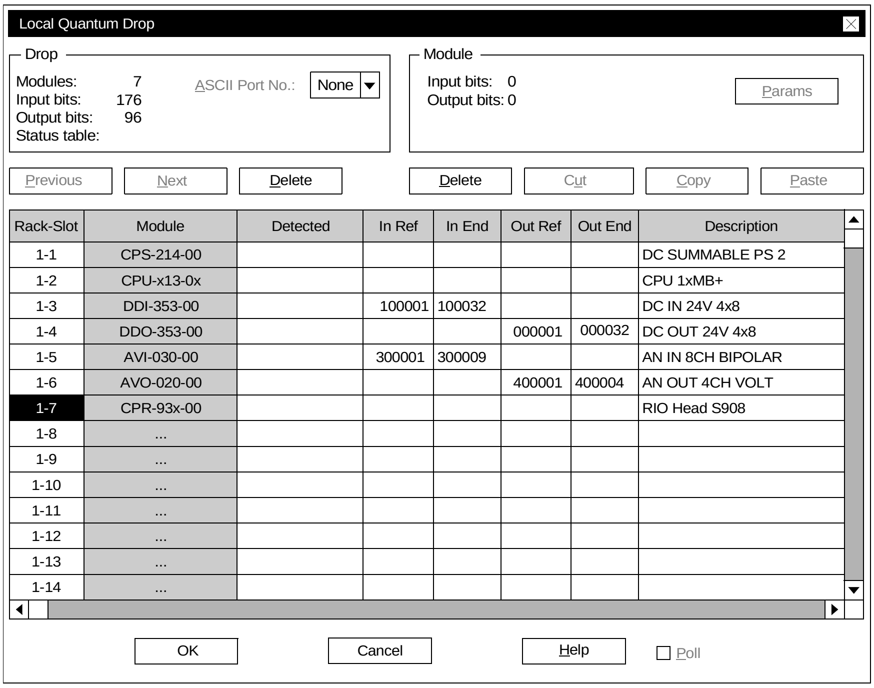
Mapping Modules and Specifying I/O References

To map the modules and specify the address ranges proceed as follows in the **Local Quantum Drop** dialog:

Step	Action
1	Select the **Module** → **...** column. **Response:** The **I/O Module Selection** dialog is opened.
2	From the **Category** column, select the **<all>** option. **Response:** All modules are listed in the **Modules** column.
3	In the **Modules** column, select the **CPS-214-00** module. Exit the dialog with **OK**. **Response:** The module is inserted in the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example (see **Local Quantum Drop** dialog representation).
5	In the **In Ref** and **Out Ref** columns, set the start references for the input and output modules. **Note:** Discrete Input References have the prefix 1 (e.g. 100001), Coil References have the prefix 0 (e.g. 000001), Input Register References have the prefix 3 (e.g. 300001) and Output Register References have the prefix 4 (e.g. 400001). **Response:** The end reference (column **In.End.** or **Out.End**) of the available address range is automatically entered.

Dialog display

Following module mapping and I/O reference specification the dialog looks like this:



Set module parameters

To set parameters for the individual modules, proceed as follows in the **Local Quantum Drop** dialog:

Step	Action
1	From the **Rack Slot** column select the **1-3** line. **Response:** The **1-3** text box has a dark background, i.e. the **DDI-353-00** module has been selected for editing. **Note:** Parameters are not set for the **CPS-214-00** and **CPU-x13-0x** modules.
2	Select the **Params** command button. **Response:** The **140-DDI-353-00** dialog is opened.
3	Select the **Discrete** option button. **Response:** You return to the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example. **Note:** The modules are sometimes set with different parameters. Help with this can be obtained from the corresponding help texts in the parameter dialog.

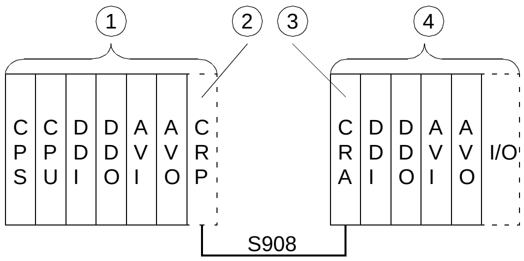
Editing Remote Drop

Editing of the I/O st. (Drop) defined second takes place in the dialog **RIO (Slot 7) - Quantum I/O-St. 2**.
This dialog can be reached in two ways:
- In the **I/O Map** dialog, using the **Edit....** command button, or
- in the **Local Quantum Drop** dialog, using the **Next** command button.

Editing Remote Drop

Introduction

This section describes the configuration of the second (remote) drop. The drop has already been defined in Editing the First (local) Drop (See *Editing local drop, p. 808*).
To edit the second (remote) drop, the modules must be specified with their I/O references before parameters for the individual modules can be set.
Quantum – remote controller with RIO



1 Local Quantum drop 1

2 RIO master module

3 RIO slave module

4 RIO drop 2

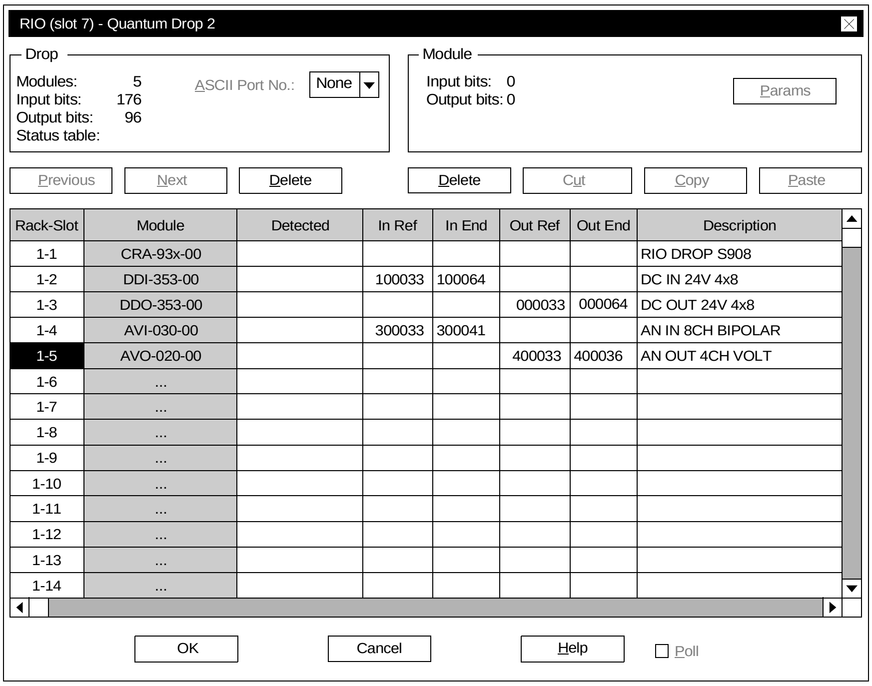
Mapping Modules and Specifying I/O References

To allocate the modules and specify the address ranges use the dialog **RIO (slot 7) - Quantum I/O-St. 2** and proceed as follows:

Step	Action
1	Select the **Module** → **...** column. **Response:** The **I/O Module Selection** dialog is opened.
2	From the **Category** column, select the **<all>** option. **Response:** All modules are listed in the **Modules** column.
3	In the **Modules** column select the **CRA-93x-00** module. Exit the dialog with **OK**. **Response:** The module is inserted in the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example (see dialog representation **RIO (Slot 7) Quantum Drop 2**).
5	In the **In Ref** and **Out Ref** columns, set the start references for the input and output modules. **Note:** Discrete Input References have the prefix 1 (e.g. 100001), Coil References have the prefix 0 (e.g. 000001), Input Register References have the prefix 3 (e.g. 300001) and Output Register References have the prefix 4 (e.g. 400001). **Response:** The end reference (column **In.End.** or **Out.End**) of the available address range is automatically entered.

Dialog display

Following module mapping and I/O reference specification the dialog looks like this:



Set module parameters

To set parameters for individual modules use the dialog **RIO (slot 7) - Quantum I/O-St. 2** and proceed as follows:

Step	Action
1	From the **Rack Slot** column select the **1-2** line. **Response:** The **1-2** text box has a dark background, i.e. the **DDI-353-00** module has been selected for editing.
2	Select the **Params** command button. **Response:** The **140-DDI-353-00** dialog is opened.
3	Select the **Discrete** option button. **Response:** You return to the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example. **Note:** The modules are sometimes set with different parameters. Help with this can be obtained from the corresponding help texts in the parameter dialog.

E.2 Quantum Example - Remote control with RIO (series 800)

Introduction

Overview

This Chapter contains the step-by-step process for the configuration of remote control with RIO (**R**emote **I**/**O**) and series 800 modules.

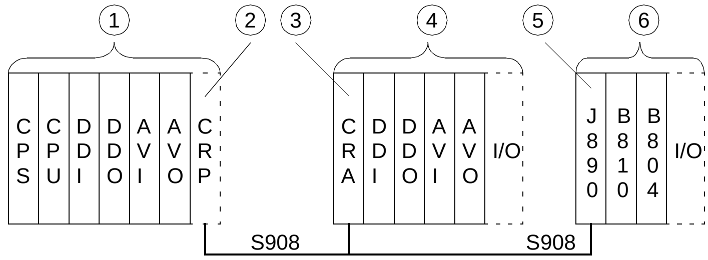
What's in this Section?

This section contains the following topics:

Editing Local Drop

Introduction

This section describes the configuration of the first (local) drop. The processing sequence begins first of all with the definition of all drops.
When editing the first (local) drop the modules must be set with their I/O references before parameters can be set forindividual modules.
Quantum – remote controller with RIO (Series 800)



1 Local Quantum drop 1

2 RIO master module

3 RIO slave module

4 RIO drop 2

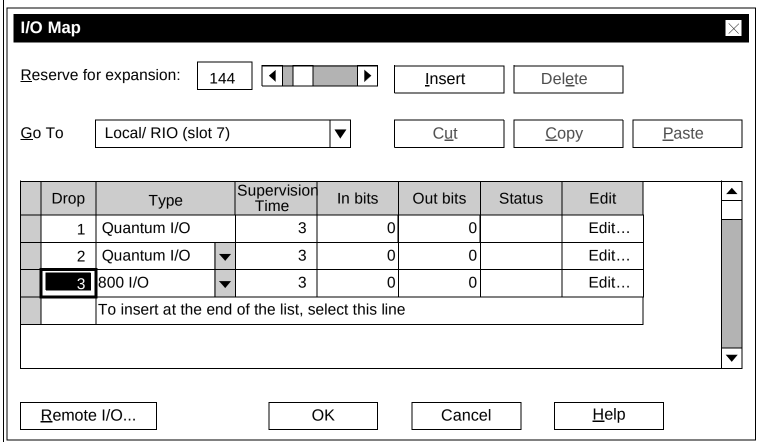
5 Adapter module

6 RIO drop 3 with series 800 modules

Defining Drops

To define drops proceed as follows in the **PLC Configuration** window:

Step	Action
1	Select **PLC Selection**. **Response:** The **PLC selection** dialog is opened.
2	Select the **Quantum** PLC family and a **CPU x113 xx**. Using **OK** return to the **PLC Configuration** window.
3	Select **I/O Map**. **Response:** The **I/O Map** dialog is opened and the first drop is automatically entered in the table.
4	Select the **Head Setup** command button. **Response:** The **Head Setup** dialog is opened.
5	Enter a 7 in **RIO Slot** and quit the dialog using **OK**. **Response:** The CRP-93x-00 module is automatically inserted in the I/O map (in slot 7) of the selected drop. In the **Go To** list box, the **Local/RIO (Slot 7)** network link is displayed.

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>6</td><td>Select the last free row in the table, and insert the second drop with the command button Insert.
Response: The second drop is entered in the Type column of the table.
Note: The number of drops to be inserted is defined in the Segment scheduler dialog. The default predetermines the maximum number of 32, so that settings are not necessary.</td></tr>
<tr><td>7</td><td>Select the last free row in the table again, and insert the third drop with the Insert command button.
Response: The second drop is entered in the Type column of the table.</td></tr>
<tr><td>8</td><td>Select the third drop and open the list box in the Type column.
Select the 800 I/O option.
Dialog display


</td></tr>
<tr><td>9</td><td>Select the first drop from the Drop column.
Select the Edit... command button.
Response: You reach the module map.</td></tr>
</table>

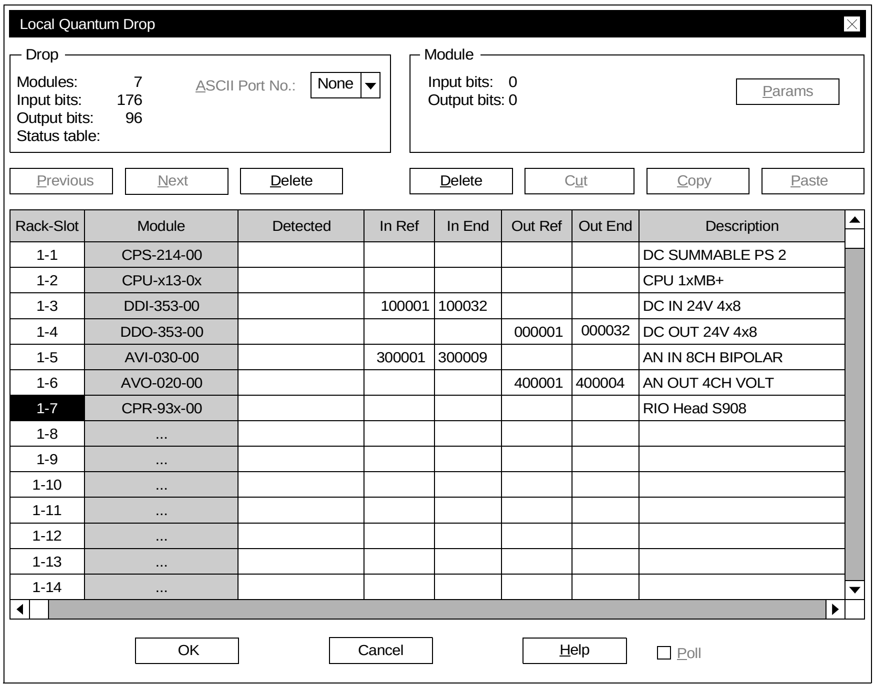
Mapping Modules and Specifying I/O References

To map the modules and specify the address ranges proceed as follows in the **Local Quantum Drop** dialog:

Step	Action
1	Select the **Module** → **...** column. **Response:** The **I/O Module Selection** dialog is opened.
2	From the **Category** column, select the **<all>** option. **Response:** All modules are listed in the **Modules** column.
3	In the **Modules** column, select the **CPS-214-00** module. Exit the dialog with **OK**. **Response:** The module is inserted in the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example (see **Local Quantum Drop** dialog representation).
5	In the **In Ref** and **Out Ref** columns, set the start references for the input and output modules. **Note:** Discrete Input References have the prefix 1 (e.g. 100001), Coil References have the prefix 0 (e.g. 000001), Input Register References have the prefix 3 (e.g. 300001) and Output Register References have the prefix 4 (e.g. 400001). **Response:** The end reference (column **In.End.** or **Out.End**) of the available address range is automatically entered.

Dialog display

Following module mapping and I/O reference specification the dialog looks like this:



Set module parameters

To set parameters for the individual modules, proceed as follows in the **Local Quantum Drop** dialog:

Step	Action
1	From the **Rack Slot** column select the **1-3** line. **Response:** The **1-3** text box has a dark background, i.e. the **DDI-353-00** module has been selected for editing. **Note:** Parameters are not set for the **CPS-214-00** and **CPU-x13-0x** modules.
2	Select the **Params** command button. **Response:** The **140-DDI-353-00** dialog is opened.
3	Select the **Discrete** option button. **Response:** You return to the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example. **Note:** The modules are sometimes set with different parameters. Help with this can be obtained from the corresponding help texts in the parameter dialog.

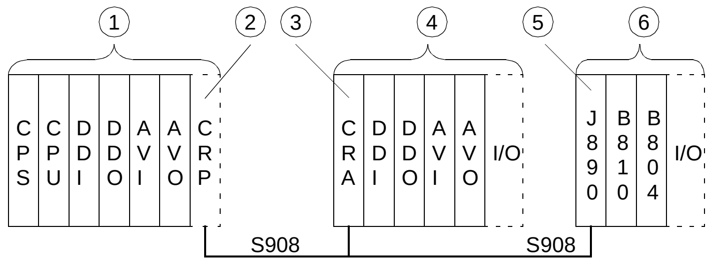
Editing Remote Drop

Editing of the I/O st. (Drop) defined second takes place in the dialog **RIO (Slot 7) - Quantum I/O-St. 2**.
This dialog can be reached in two ways:
- In the **I/O Map** dialog, using the **Edit...** command button, or
- in the **Local Quantum Drop** dialog, using the **Next** command button.

Editing Remote Drop

Introduction

This section describes the configuration of the second (remote) drop. The drop has already been defined in Editing the First (local) Drop (See *Editing Local Drop, p. 816*).
To edit the second (remote) drop, the modules must be specified with their I/O references before parameters for the individual modules can be set.
Quantum – remote controller with RIO (Series 800)



1 Local Quantum drop 1

2 RIO master module

3 RIO slave module

4 RIO drop 2

5 Adapter module

6 RIO drop 3 with series 800 modules

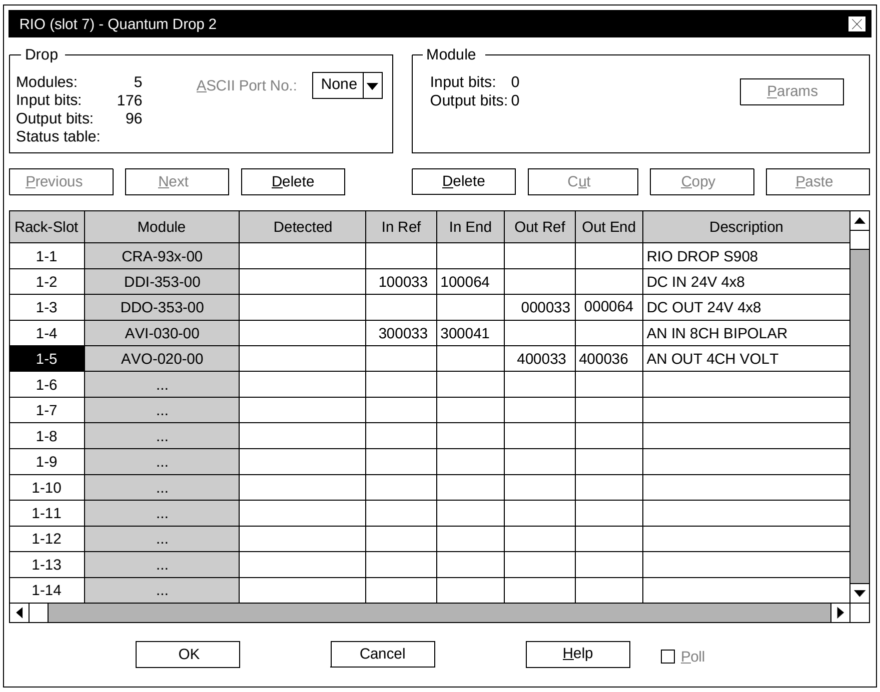
Mapping Modules and Specifying I/O References

To allocate the modules and specify the address ranges use the dialog **RIO (slot 7) - Quantum I/O-St. 2** and proceed as follows:

Step	Action
1	Select the **Module** → **...** column. **Response:** The **I/O Module Selection** dialog is opened.
2	From the **Category** column, select the **<all>** option. **Response:** All modules are listed in the **Modules** column.
3	In the **Modules** column select the **CRA-93x-00** module. Exit the dialog with **OK**. **Response:** The module is inserted in the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example (see dialog representation **RIO (Slot 7) Quantum Drop 2**).
5	In the **In Ref** and **Out Ref** columns, set the start references for the input and output modules. **Note:** Discrete Input References have the prefix 1 (e.g. 100001), Coil References have the prefix 0 (e.g. 000001), Input Register References have the prefix 3 (e.g. 300001) and Output Register References have the prefix 4 (e.g. 400001). **Response:** The end reference (column **In.End.** or **Out.End**) of the available address range is automatically entered.

Dialog display

Following module mapping and I/O reference specification the dialog looks like this:



Set module parameters

To set parameters for individual modules use the dialog **RIO (slot 7) - Quantum I/O-St. 2** and proceed as follows:

Step	Action
1	From the **Rack Slot** column select the **1-2** line. **Response:** The **1-2** text box has a dark background, i.e. the **DDI-353-00** module has been selected for editing.
2	Select the **Params** command button. **Response:** The **140-DDI-353-00** dialog is opened.
3	Select the **Discrete** option button. **Response:** You return to the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example. **Note:** The modules are sometimes set with different parameters. Help with this can be obtained from the corresponding help texts in the parameter dialog.

Editing Remote 800 Drops

The third defined drop is edited in the **RIO (Slot 7) - 800 Drop 3** dialog.
This dialog can be reached in two ways:
- In the **I/O Map** dialog, using the **Edit....** command button, or
- in the **RIO (Slot 7) - Quantum Drop 2** dialog using the **Next** command button.

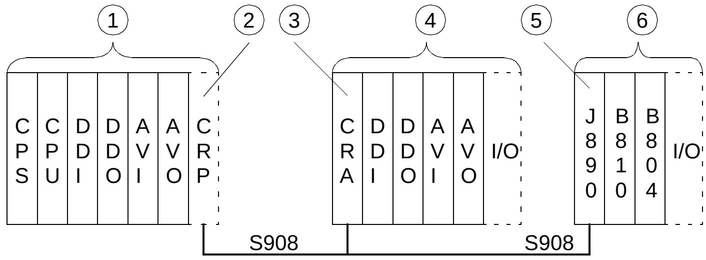
Editing Remote Drop

Introduction

This Section describes the configuration of the third (remote) drop. The drop has already been defined in Editing the First (local) Drop (See *Editing Local Drop, p. 816*).
To edit the third (remote) drop, the modules must be specified with their I/O references before the individual modules can be parameterized.

> **Note:** The J890 adapter module must be mounted in the rack of the third drop. However, this module is not visible either in the software or in the dialogs.

Quantum – remote controller with RIO (Series 800)



1 Local Quantum drop 1

2 RIO master module

3 RIO slave module

4 RIO drop 2

5 Adapter module

6 RIO drop 3 with series 800 modules

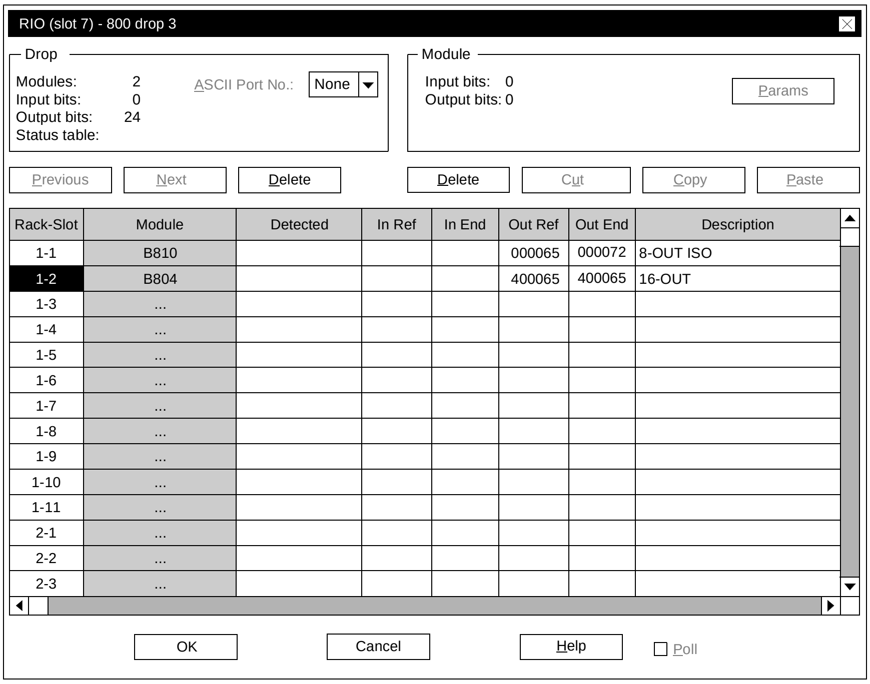
Mapping Modules and Specifying I/O References

To map the modules and specify the address ranges go to the **RIO (Slot 7) - 800 Drop 3** dialog and proceed as follows:

Step	Action
1	Select the **Module** → **...** column. **Response:** The **I/O Module Selection** dialog is opened.
2	From the **Category** column, select the **<all>** option. **Response:** All modules are listed in the **Modules** column.
3	In the **Modules** column select the **B810** module. Exit the dialog with **OK**. **Response:** The module is inserted in the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example (see dialog representation **RIO (slot 7) 800 drop 3**).
5	In the **Out Ref.** column, enter the start references for the output modules. **Note:** Discrete Input References have the prefix 1 (e.g. 100001), Coil References have the prefix 0 (e.g. 000001), Input Register References have the prefix 3 (e.g. 300001) and Output Register References have the prefix 4 (e.g. 400001). **Response:** The end reference of the available address range (**Out End** column) is entered automatically.

Dialog display

Following module mapping and I/O reference specification the dialog looks like this:



Set module parameters

To set parameters for individual modules go to the **RIO (slot 7) - 800 drop 3** dialog and proceed as follows:

Step	Action
1	From the **Rack Slot** column select the **1-1** line. **Response:** The **1-1** text box has a dark background, i.e. the **B810** module has been selected for editing.
2	Select the **Params** command button. **Response:** The **B810** dialog is opened.
3	Select the option button **Discrete** **Response:** You return to the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example. **Note:** The modules are sometimes set with different parameters. Help with this can be obtained from the corresponding help texts in the parameter dialog.

E.3 Quantum Example - Remote Control with DIO

Introduction

Overview

This Chapter contains the step-by-step process for the configuration of remote control with DIO (**D**istributed **I**/**O**).

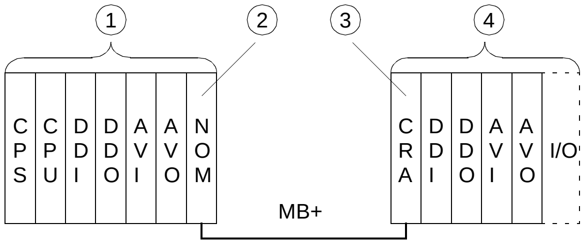
What's in this Section?

This section contains the following topics:

Editing Local Drop

Introduction

This section describes the configuration of the first (local) drop. The processing sequence begins first of all with the definition of the drop.
When editing the first (local) drop the modules must be set with their I/O references before parameters can be set forindividual modules.
Quantum – remote controller with DIO



1 Local Quantum drop 1

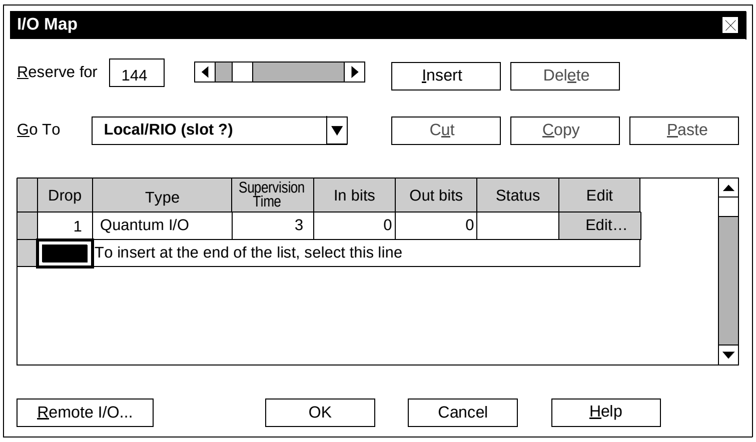
2 DIO master module

3 DIO slave module

4 DIO drop 2

Defining the Drop

To define the drop use **Configure** from the main menu and proceed as follows:

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Select PLC Selection.
Response: The PLC selection dialog is opened.</td></tr>
<tr><td>2</td><td>Select the Quantum PLC family and a CPU x113 xx. Using OK return to the PLC Configuration window.</td></tr>
<tr><td>3</td><td>Select I/O Map.
Response: The I/O Map dialog is opened and the first drop is automatically entered in the table.</td></tr>
<tr><td>4</td><td>Select the Head Setup command button.
Response: The Head Setup dialog is opened.</td></tr>
<tr><td>5</td><td>Enter a 7 in NOM Slot 1 and quit the dialog using OK.
Response: The NOM module NOM-2xx-00 is automatically inserted in the I/O map (in slot 7) of the selected drop. In the Go to list box, the network link Loca I/O RIO (Slot ?) is displayed.
Dialog display


</td></tr>
<tr><td>6</td><td>Select the drop from the Drop column.
Select the Edit... command button.
Response: You reach the module map.</td></tr>
</table>

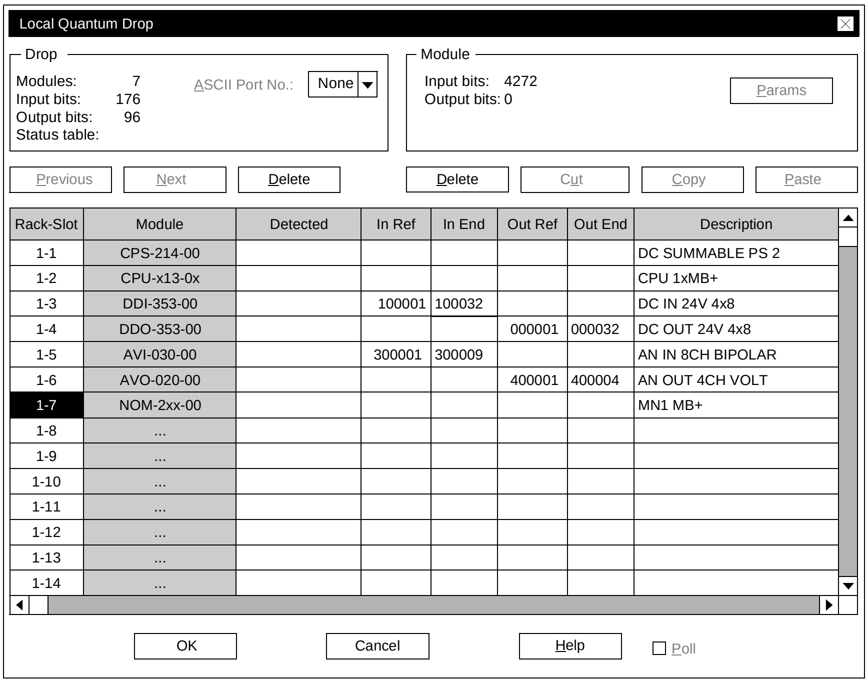
Mapping Modules and Specifying I/O References

To map the modules and specify the address ranges proceed as follows in the **Local Quantum Drop** dialog:

Step	Action
1	Select the **Module** → **...** column. **Response:** The **I/O Module Selection** dialog is opened.
2	From the **Category** column, select the **<all>** option. **Response:** All modules are listed in the **Modules** column.
3	In the **Modules** column, select the **CPS-214-00** module. Exit the dialog with **OK**. **Response:** The module is inserted in the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example (see **Local Quantum Drop** dialog representation).
5	In the **In Ref** and **Out Ref** columns, set the start references for the input and output modules. **Note:** Discrete Input References have the prefix 1 (e.g. 100001), Coil References have the prefix 0 (e.g. 000001), Input Register References have the prefix 3 (e.g. 300001) and Output Register References have the prefix 4 (e.g. 400001). **Response:** The end reference (column **In.End.** or **Out.End**) of the available address range is automatically entered.

Dialog display

Following module mapping and I/O reference specification the dialog looks like this:



Set module parameters

To set parameters for the individual modules, proceed as follows in the **Local Quantum Drop** dialog:

Step	Action
1	From the **Rack Slot** column select the **1-3** line. **Response:** The **1-3** text box has a dark background, i.e. the **DDI-353-00** module has been selected for editing. **Note:** Parameters are not set for the **CPS-214-00** and **CPU-x13-0x** modules.
2	Select the **Params** command button. **Response:** The **140-DDI-353-00** dialog is opened.
3	Select the **Discrete** option button. **Response:** You return to the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example. **Note:** The modules are sometimes set with different parameters. Help with this can be obtained from the corresponding help texts in the parameter dialog.

Editing Remote Drop

To edit the remote drop with DIO, you must return to the **I/O Map** dialog and define the drop.

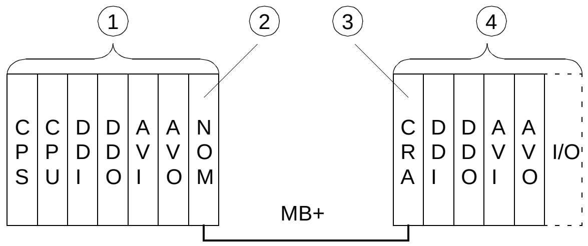
Editing Local Drop

Introduction

This section describes the configuration of the second (remote) drop. The processing sequence begins first of all with the definition of the drop.
To edit the second (remote) drop, the modules must be specified with their I/O references before parameters for the individual modules can be set.

Note: To link to the remote network, the coupling module CRA-21x-x0 must be entered during module mapping.

Quantum – remote controller with DIO



1 Local Quantum drop 1

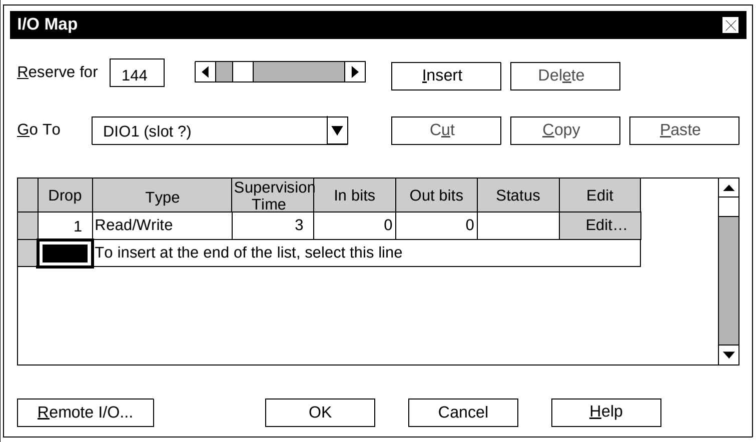
2 DIO master module

3 DIO slave module

4 DIO drop 2

Defining the Drop

To define the drop go to the **I/O map** dialog and proceed as follows:

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>From the Go to list box, select the DIO 1 (Slot 7) network link.
Response: The drop entered in the table is no longer displayed.</td></tr>
<tr><td>2</td><td>Select the Insert command button.
Response: In the Type column, the Read/Write type is entered.
Dialog display

</td></tr>
<tr><td>3</td><td>Select the drop from the Drop column.
Select the Edit... command button.
Response: You reach the module map.</td></tr>
</table>

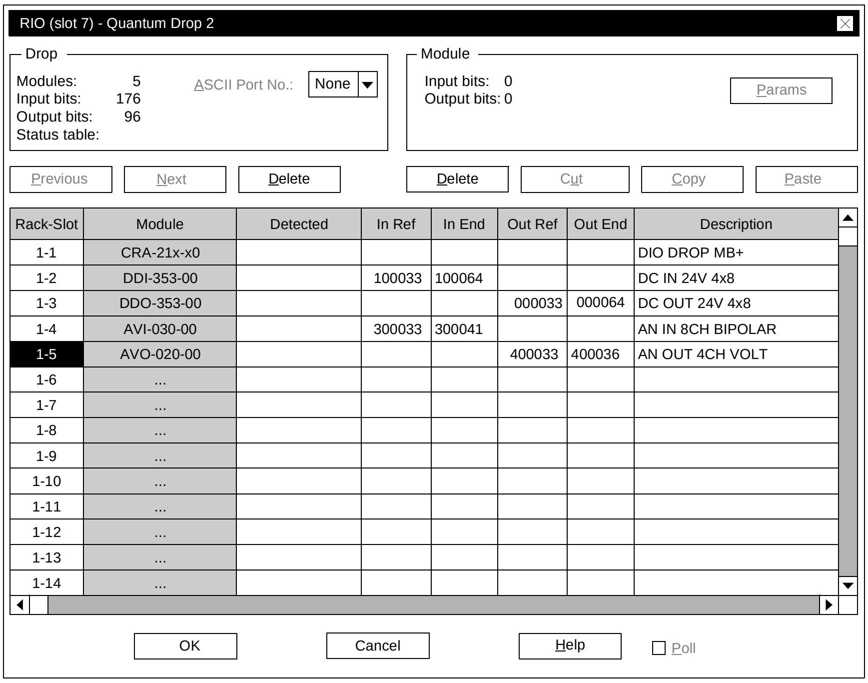
Mapping Modules and Specifying I/O References

To map the modules and specify the address ranges use the **DIO 1 (slot 7) - drop 1** dialog and proceed as follows:

Step	Action
1	Select the **Module** → **...** column. **Response:** The **I/O Module Selection** dialog is opened.
2	From the **Category** column, select the **<all>** option. **Response:** All modules are listed in the **Modules** column.
3	In the **Modules** column select the **CRA-21x-x0** module. Exit the dialog with **OK**. **Response:** The module is inserted in the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example (see dialog representation **DIO (slot 7) Quantum drop 1**).
5	In the **In Ref** and **Out Ref** columns, set the start references for the input and output modules. **Note:** Discrete Input References have the prefix 1 (e.g. 100001), Coil References have the prefix 0 (e.g. 000001), Input Register References have the prefix 3 (e.g. 300001) and Output Register References have the prefix 4 (e.g. 400001). **Response:** The end reference (column **In.End.** or **Out.End**) of the available address range is automatically entered.

Dialog display

Following module mapping and I/O reference specification the dialog looks like this:



Set module parameters

To set parameters for individual modules use the **DIO 1 (slot 7) - Drop 1** dialog and proceed as follows:

Step	Action
1	From the **Rack Slot** column select the **1-2** line. **Response:** The **1-2** text box has a dark background, i.e. the **DDI-353-00** module has been selected for editing.
2	Select the **Params** command button. **Response:** The **140-DDI-353-00** dialog is opened.
3	Select the **Discrete** option button. **Response:** You return to the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example. **Note:** The modules are sometimes set with different parameters. Help with this can be obtained from the corresponding help texts in the parameter dialog.

E.4 Quantum Example – INTERBUS Control

Introduction

Overview

This Chapter contains the step-by-step process for the configuration of INTERBUS control with the Quantum.

What's in this Section?

This section contains the following topics:

General Information

Introduction

INTERBUS control using Quantum occurs via module NOA-611-10. During this process the module collects the words of all remote bus nodes and creates a telegram with status information and I/O words. The telegram is then transferred to the CPU, so that the NOA behaves like an I/O module.

Note: Using branch interfaces in the remote bus, remote bus branches with further remote bus nodes (TIOs) can be constructed. However, the branch interfaces can only be inserted in the remote bus, not in the remote bus branch.

Parameterization

Command sequence parameterization (restart procedure) occurs in the CMD Tool, produced by the PHOENIX firm (see also " NOA 611 1 restart procedure" with an example for parameterizing the command sequence in CMD Tool).

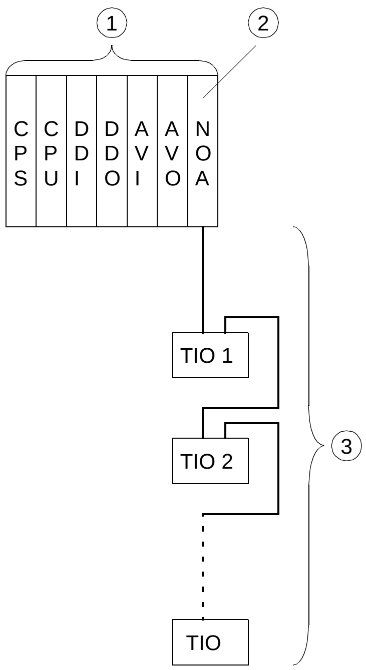
Editing Local Drop

Introduction

This section describes the configuration of the first (local) drop. The processing sequence begins first of all with the definition of the drop.
When editing the first (local) drop the modules must be set with their I/O references before parameters can be set for individual modules.

> **Note:** When the NOA-611-00 module is entered in the I/O map, the loadable ULEX is automatically installed.

Quantum - INTERBUS controller



1 Local Quantum Drop

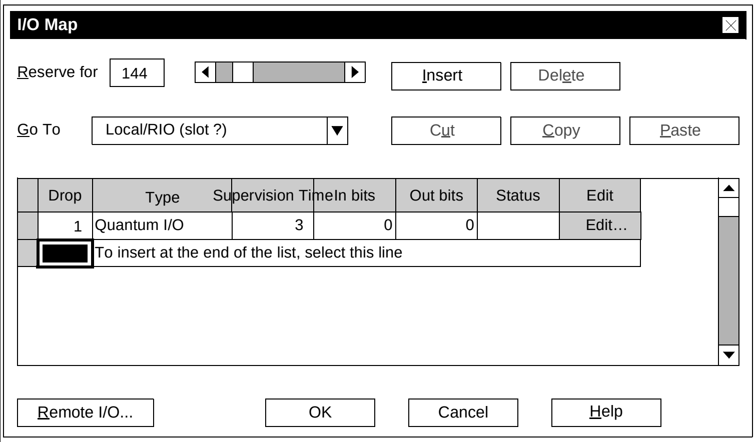
2 INTERBUS master module

3 Remote bus without branch interface

> **Note:** The configuration of remote bus nodes does not take place in Concept and is therefore not apparent in the I/O map. To edit the remote bus nodes, you must use the CMD tool produced by the PHOENIX firm (**C**onfiguration **M**onitoring and **D**iagnostic Software).

Defining Drops

To define drops proceed as follows in the **PLC Configuration** window:

Step	Action
1	Select **PLC Selection**. **Response:** The **PLC selection** dialog is opened.
2	Select the **Quantum** PLC family and a **CPU x113 xx**. Use **OK** return to the **PLC Configuration** window.
3	Select **I/O Map**. **Response:** The **I/O Map** dialog is opened and the first drop is automatically entered in the table. Dialog display 
4	Select the drop from the **Drop** column. Select the **Edit...** command button. **Response:** You reach the module map.

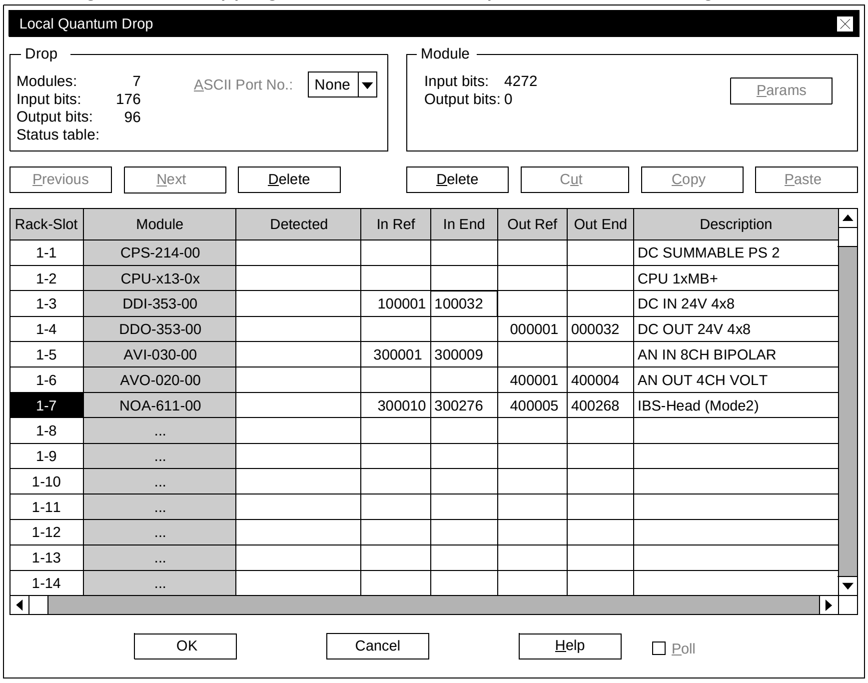
Mapping Modules and Specifying I/O References

To map the modules and specify the address ranges proceed as follows in the **Local Quantum Drop** dialog:

Step	Action
1	Select the **Module** → **...** column. **Response:** The **I/O Module Selection** dialog is opened.
2	From the **Category** column, select the **<all>** option. **Response:** All modules are listed in the **Modules** column.
3	In the **Modules** column, select the **CPS-214-00** module. Exit the dialog with **OK**. **Response:** The module is inserted in the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example (see **Local Quantum Drop** dialog representation).
5	In the **In Ref** and **Out Ref** columns, set the start references for the input and output modules. **Note:** Discrete Input References have the prefix 1 (e.g. 100001), Coil References have the prefix 0 (e.g. 000001), Input Register References have the prefix 3 (e.g. 300001) and Output Register References have the prefix 4 (e.g. 400001). **Response:** The end reference (column **In.End.** or **Out.End**) of the available address range is automatically entered.

Dialog display

Following module mapping and I/O reference specification the dialog looks like this:



Set module parameters

To set parameters for the individual modules, proceed as follows in the **Local Quantum Drop** dialog:

Step	Action
1	From the **Rack Slot** column select the **1-3** line. **Response:** The **1-3** text box has a dark background, i.e. the **DDI-353-00** module has been selected for editing. **Note:** Parameters are not set for the **CPS-214-00** and **CPU-x13-0x** modules.
2	Select the **Params** command button. **Response:** The **140-DDI-353-00** dialog is opened.
3	Select the **Discrete** option button. **Response:** You return to the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example. **Note:** The modules are sometimes set with different parameters. Help with this can be obtained from the corresponding help texts in the parameter dialog.

E.5 Quantum Example - SY/MAX Controller

Introduction

Overview

This Chapter contains the step-by-step process for the configuration of a SY/MAX controller.

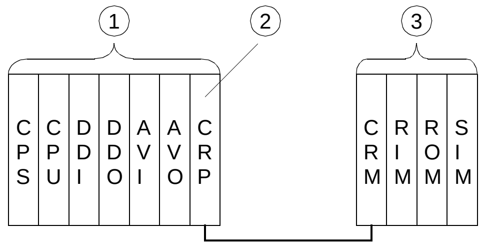
What's in this Section?

This section contains the following topics:

Editing Local Drop

Introduction

This section describes the configuration of the first (local) drop. The processing sequence begins first of all with the definition of all drops.
When editing the first (local) drop the modules must be set with their I/O references before parameters can be set forindividual modules.
Quantum – SY/MAX controller



1 Local Quantum drop 1

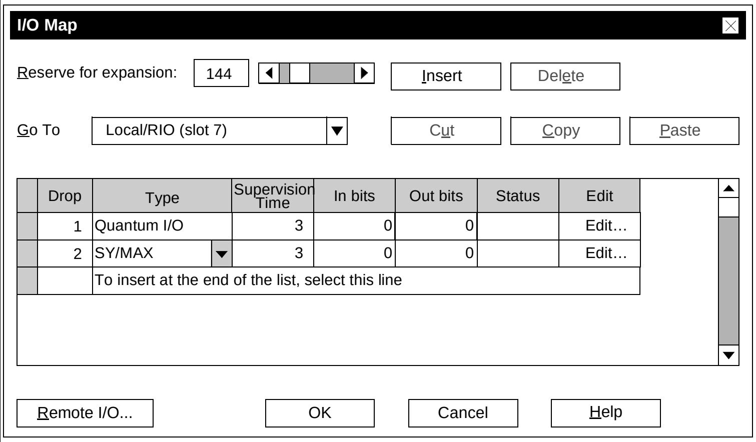
2 RIO master module

3 SY/MAX drop 2

Defining Drops

To define drops proceed as follows in the **PLC Configuration** window:

Step	Action
1	Select **PLC Selection**. **Response:** The **PLC selection** dialog is opened.
2	Select the **Quantum** PLC family and a **CPU x113 xx**. Using **OK** return to the **PLC Configuration** window.
3	Select **I/O Map**. **Response:** The **I/O Map** dialog is opened and the first drop is automatically entered in the table.
4	Select the **Head Setup** command button. **Response:** The **Head Setup** dialog is opened.
5	Enter a 7 in **RIO Slot** and quit the dialog using **OK**. **Response:** The CRP-93x-00 module is automatically inserted in the I/O map (in slot 7) of the selected drop. In the **Go To** list box, the **Local/RIO (Slot 7)** network link is displayed.
6	Select the last line in the table. Select the **Insert** command button. **Response:** The second drop is entered in the **Type** column. **Note:** The number of drops to be inserted is defined in the **segment scheduler** dialog. The default defines a maximum number of 32.

Step	Action
7	Select the second drop and in the **Type** column, open the list box. Select the **SY/MAX**option. Dialog display 
8	Select the first drop from the **Drop** column. Select the **Edit...** command button. **Response:** You reach the module map.

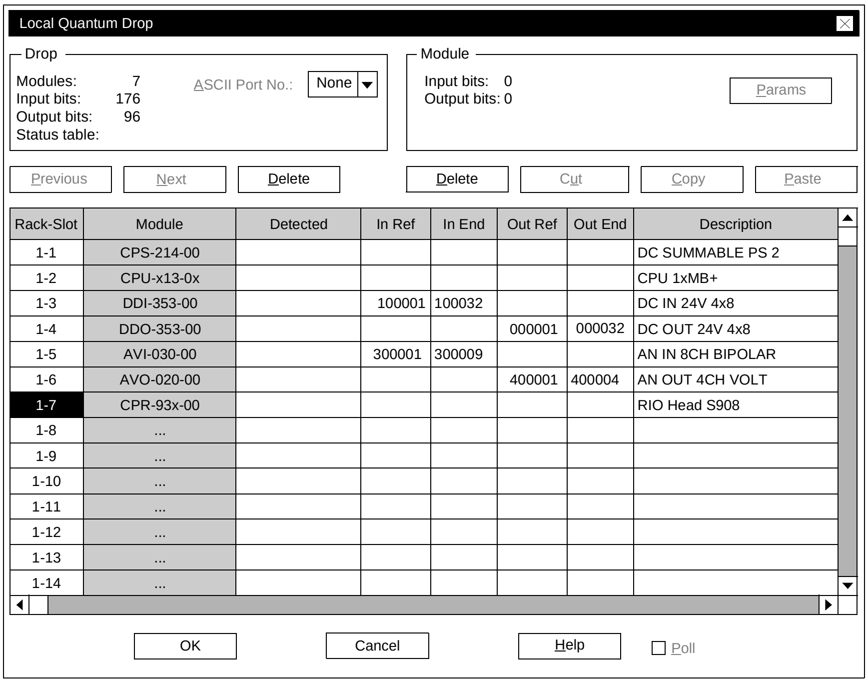
Mapping Modules and Specifying I/O References

To map the modules and specify the address ranges proceed as follows in the **Local Quantum Drop** dialog:

Step	Action
1	Select the **Module** → **...** column. **Response:** The **I/O Module Selection** dialog is opened.
2	From the **Category** column, select the **<all>** option. **Response:** All modules are listed in the **Modules** column.
3	In the **Modules** column, select the **CPS-214-00** module. Exit the dialog with **OK**. **Response:** The module is inserted in the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example (see **Local Quantum Drop** dialog representation).
5	In the **In Ref** and **Out Ref** columns, set the start references for the input and output modules. **Note:** Discrete Input References have the prefix 1 (e.g. 100001), Coil References have the prefix 0 (e.g. 000001), Input Register References have the prefix 3 (e.g. 300001) and Output Register References have the prefix 4 (e.g. 400001). **Response:** The end reference (column **In.End.** or **Out.End**) of the available address range is automatically entered.

Dialog display

Following module mapping and I/O reference specification the dialog looks like this:



Rack-Slot	Module	Detected	In Ref	In End	Out Ref	Out End	Description
1-1	CPS-214-00						DC SUMMABLE PS 2
1-2	CPU-x13-0x						CPU 1xMB+
1-3	DDI-353-00		100001	100032			DC IN 24V 4x8
1-4	DDO-353-00				000001	000032	DC OUT 24V 4x8
1-5	AVI-030-00		300001	300009			AN IN 8CH BIPOLAR
1-6	AVO-020-00				400001	400004	AN OUT 4CH VOLT
1-7	CPR-93x-00						RIO Head S908
1-8	...						
1-9	...						
1-10	...						
1-11	...						
1-12	...						
1-13	...						
1-14	...						

Set module parameters

To set parameters for the individual modules, proceed as follows in the **Local Quantum Drop** dialog:

Step	Action
1	From the **Rack Slot** column select the **1-3** line. **Response:** The **1-3** text box has a dark background, i.e. the **DDI-353-00** module has been selected for editing. **Note:** Parameters are not set for the **CPS-214-00** and **CPU-x13-0x** modules.
2	Select the **Params** command button. **Response:** The **140-DDI-353-00** dialog is opened.
3	Select the **Discrete** option button. **Response:** You return to the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example. **Note:** The modules are sometimes set with different parameters. Help with this can be obtained from the corresponding help texts in the parameter dialog.

Editing Remote Drop

Editing the drop defined second takes place in the dialog **RIO (slot 7) – SY/MAX I/O-St. 2**.
This dialog can be reached in two ways:
- In the **I/O Map** dialog, using the **Edit....** command button, or
- in the **Local Quantum Drop** dialog, using the **Next** command button.

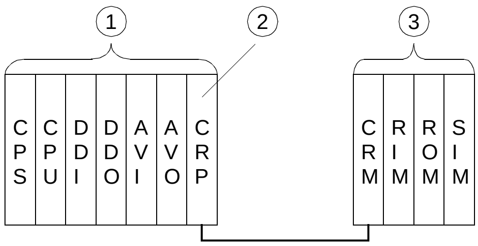
Editing Remote Drop

Introduction

This section describes the configuration of the second (remote) drop. The drop has already been defined in Editing the First (local) Drop (See *Editing Local Drop, p. 842*).
To edit the second (remote) drop, the modules must be specified with their I/O references before parameters for the individual modules can be set.

> **Note:** To link to the remote network, the coupling module CRM-931-RG must be entered during module mapping.

Quantum – SY/MAX controller



1 Local Quantum drop 1

2 RIO master module

3 SY/MAX drop 2

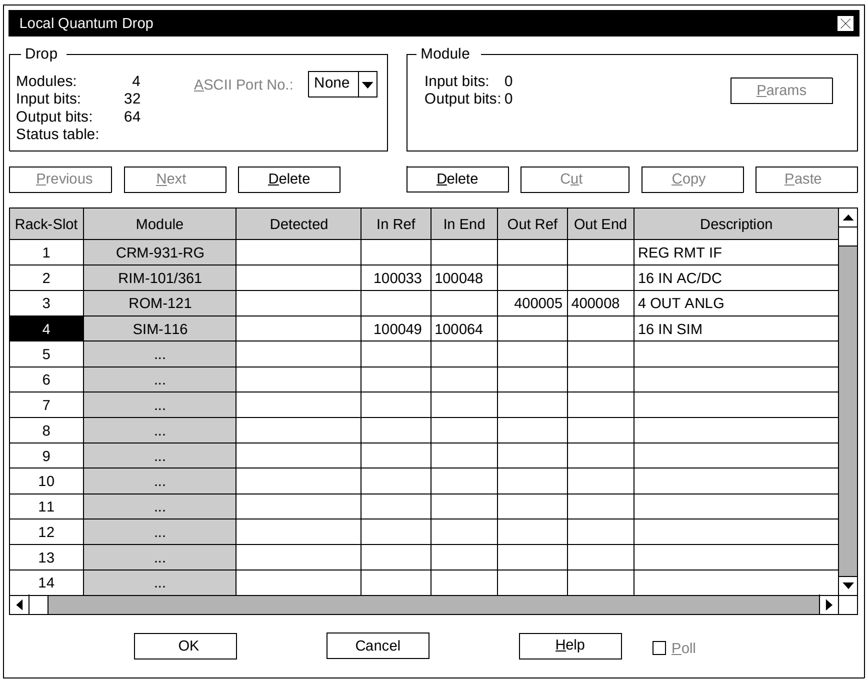
Mapping Modules and Specifying I/O References

To map the modules and specify the address ranges proceed as follows in the **RIO (slot 7) – SY/MAX I/O-St. 2** dialog:

Step	Action
1	Select the **Module** → **...** column. **Response:** The **I/O Module Selection** dialog is opened.
2	From the **Category** column, select the **<all>** option. **Response:** All modules are listed in the **Modules** column.
3	In the **Modules** column select the **CRM-931-RG** module. Exit the dialog with **OK**. **Response:** The module is inserted in the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example (see dialog representation **RIO (Slot 7) SY/MAX drop 2**).
5	In the **In Ref** and **Out Ref** columns, set the start references for the input and output modules. **Note:** Discrete Input References have the prefix 1 (e.g. 100001), Coil References have the prefix 0 (e.g. 000001), Input Register References have the prefix 3 (e.g. 300001) and Output Register References have the prefix 4 (e.g. 400001). **Response:** The end reference (column **In.End.** or **Out.End**) of the available address range is automatically entered.

Dialog display

Following module mapping and I/O reference specification the dialog looks like this:



Set module parameters

To parameter the individual modules use the dialog **RIO (slot 7) – SY/MAX I/O-St. 2** and proceed as follows:

Step	Action
1	In the **Slot** column, select line **2**. **Response:** The **2** text box has a dark background, i.e. the **RIM-101/361** module has been selected for editing.
2	Select the **Params** command button. **Response:** The **8030-RIM-101/361** dialog is opened.
3	Select the **Discrete** option button. **Response:** You return to the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example. **Note:** The modules are sometimes set with different parameters. Help with this can be obtained from the corresponding help texts in the parameter dialog.

E.6 Quantum Example - Profibus DP Controller

Introduction

Overview

This Chapter contains the step-by-step process for the configuration of a Profibus DP controller with the Quantum.

What's in this Section?

This section contains the following topics:

General Information

Introduction

Configuring Profibus DP is done using the SyCon (System Configurator) software produced by Hilscher GmbH. It is initially stored there as a file (*.CNF). This generated file is loaded into Concept and is visible in the I/O map of the configurator. Before the Profibus DP nodes (max. 32) can be imported, a bus controller (CRP 811 00) must be mapped in the drop (Quantum I/O). Depending on the CPU selection in the **Select Extensions** dialog box, a maximum of two to six bus controllers can be inserted.

Profibus DP Export Settings in SyCon

Introduction

SyCon is used to configure Profibus DP. The procedure for this is to be found in the user manual provided by the manufacturer. The settings for the export of the *.CNF file are explained in the following step-by-step instructions.

Preconditions

For CRP-811-00 diagnostics the serial interface of the host computer and the diagnostic interface of the bus controller must be linked with a V24 cable.
To display this diagnostic data, terminal emulation software must be started (e.g. PROCOMM using the settings: 19.2 kBd, 8 data bits, 1 stop bit, no parity).

Defining the Destination Directory

Firstly, specify the destination directory in which all files are to be saved:

Step	Action
1	Select in the main menu **Settings** → **Search Path...**. **Response:** The **Search Path** dialog is opened and is pre-set with the SyCon directory path as the project directory (e.g. C:\HILSCHER GMBH\SYCON\FIELDBUS\PROFIBUS).
2	Enter the path of the Concept directory (e.g. C:\CONCEPT\PROFIBUS) in the **Project Directory** text box. **Note:** You can also accept the default. **Response:** Execution of the **Save** and **Export** menu commands (in the **File** main menu) saves all files in the entered Concept directory.

Generating an Export File

To generate an export file (*.CNF) proceed as follows:

Step	Action
1	Select in the main menu **File** → **Save** → ***.PB**. **Response:** The configuration is stored as a database file *.PB in the specified directory.
2	Select in the main menu **File** → **Export** → **ASCII**. **Response:** The configuration is stored as an ASCII file *.CNF in the specified directory.
3	Exit SyCon and start Concept.

Note about Saving

The configuration must always be saved as a database file *.PB first, only then can an ASCII file be generated from the saved *.PB file. Every change must therefore also be saved as a *.PB file first, before an ASCII file can be generated for export. The files *.PB and *.CNF should always be saved in the same project directory.

Profibus DP Configuration in Concept

After the Profibus DP nodes have been configured in SyCon, the Profibus DP configuration is imported into the Concept I/O map.
An example of configuration and import is described in the chapter "Editing a Local Drop (See *Editing Local Drop, p. 852*)".

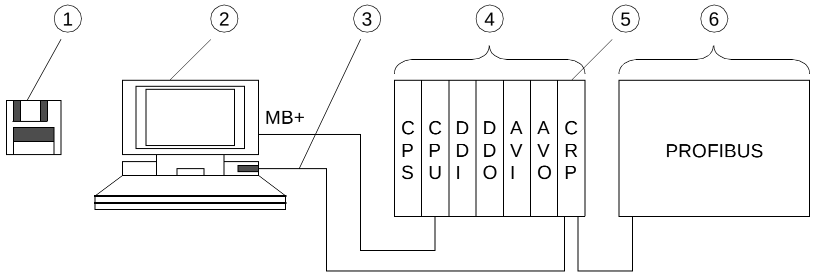
Editing Local Drop

Introduction

This section describes the configuration of the first (local) drop.
For Profibus DP configuration the CRP-811-00 coupling module must be registered in the I/O map. The configuration defined in SyCon is loaded into Concept as the generated *.CNF file is imported into the parameter dialog of the CRP-811-00 coupling module.

> **Note:** For an error free transfer of the Profibus DP configuration, it should be ensured that sufficient memory is available. To optimize storage occupancy open the dialog **PLC Memory Partition** (**PLC Configuration** → **PLC Memory Partition**).

When editing the first (local) drop the modules must be set with their I/O references before the individual modules can be parameterized.
Quantum – Profibus DP controller



1 Device data base for CRP-811-00 (load onto SyCon)

2 Host computer for Concept and SyCon

3 V24 cable

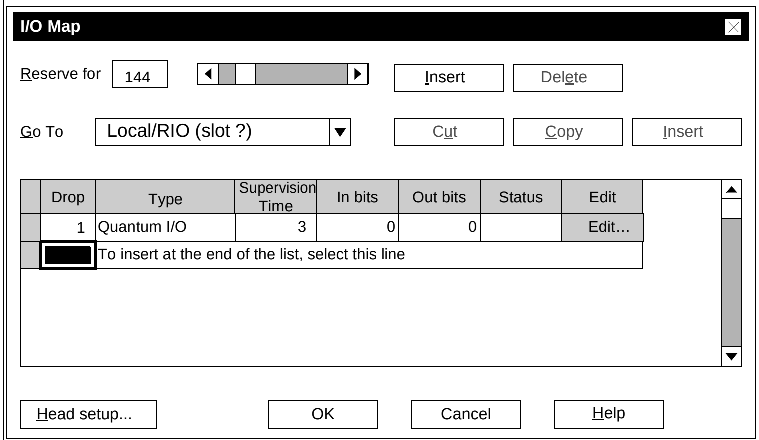
4 Local Quantum drop 1

5 RIO master module

6 Profibus DP configuration (External modules)

Defining Drops

To define drops proceed as follows in the **PLC Configuration** window:

Step	Action
1	Select **PLC Selection**. **Response:** The **PLC Selection** dialog is opened.
2	Select the **Quantum** PLC family and a **CPU x113 xx**. Using **OK** return to the **PLC Configuration** window.
3	Select **Config. Extensions** → **Select Extensions** list. Response: The **Select Extensions** dialog is opened.
4	In the **Profibus DP** list box select the **1** option. **Response:** The coupling module then appears in the **I/O Module Selection** dialog and can be used in the I/O map.
5	Select **I/O Map**. **Response:** The **I/O Map** dialog is opened and the first drop is automatically entered in the table. Dialog Representation 
6	Select the drop from the **Drop** column. Select the **Edit...** command button. **Response:** You reach the module map.

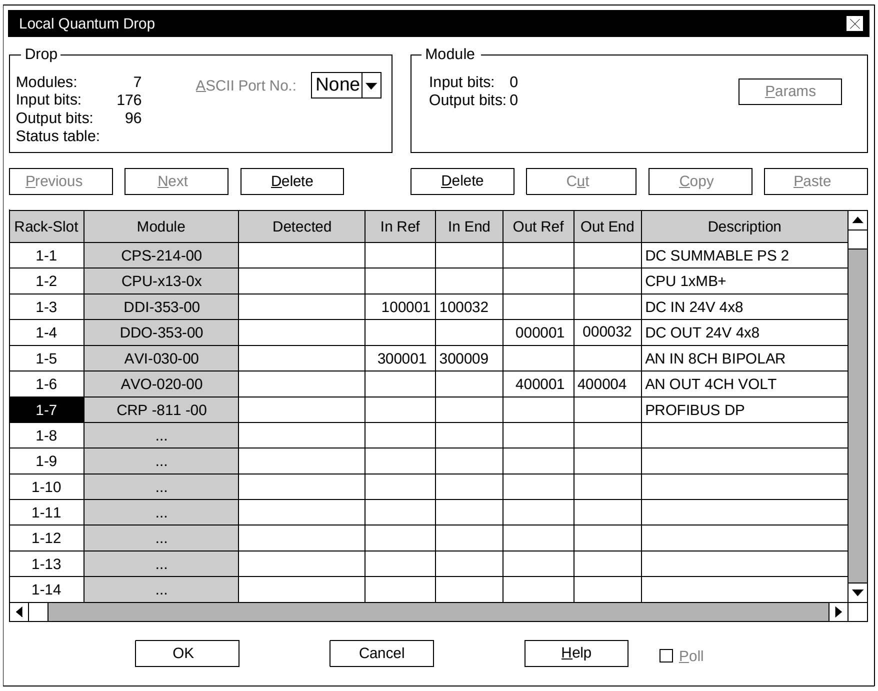
Mapping Modules and Specifying I/O References

To map the modules and specify the address ranges proceed as follows in the **Local Quantum Drop** dialog:

Step	Action
1	Select the **Module** → **...** column. **Response:** The **I/O Module Selection** dialog is opened.
2	From the **Category** column, select the **<all>** option. **Response:** All modules are listed in the **Modules** column.
3	In the **Modules** column, select the **CPS-214-00** module. Exit the dialog with **OK**. **Response:** The module is inserted in the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example (see **Local Quantum Drop** dialog representation).
5	In the **In Ref** and **Out Ref** columns, set the start references for the input and output modules. **Note:** Discrete Input References have the prefix 1 (e.g. 100001), Coil References have the prefix 0 (e.g. 000001), Input Register References have the prefix 3 (e.g. 300001) and Output Register References have the prefix 4 (e.g. 400001). **Response:** The end reference of the available address range (**In End** or **Out End** column) is entered automatically.

Dialog Representation

Following module mapping and I/O reference specification, the dialog looks like this:



Parameterization of Modules

To parameterize the individual modules, proceed as follows in the **Local Quantum Drop** dialog:

Step	Action
1	From the **Rack Slot** column, select line **1-3**. **Response:** The **1-3** text box has a dark background, i.e. the **DDI-353-00** module has been selected for editing. **Note:** The **CPS-214-00** and **CPU-x13-0x** modules are not parameterized.
2	Select the **Params** command button. **Response:** The **140-DDI-353-00** dialog is opened.
3	Select the **Discrete** option button. **Response:** You return to the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example. **Note:** The modules are sometimes parameterized differently. Help with this can be obtained from the corresponding help texts in the parameter dialog.

Importing a Profibus DP Configuration

Importing configured Profibus DP nodes occurs in the parameter dialog of the CRP-811-00 coupling module. This dialog opens when you select the CRP-811-00 row from the I/O map and press the **Params** command button.

Importing Profibus DP Configuration

Introduction

This section describes the import of the Profibus DP configuration. After that, further parameter settings for the master take place and the I/O map can be established.

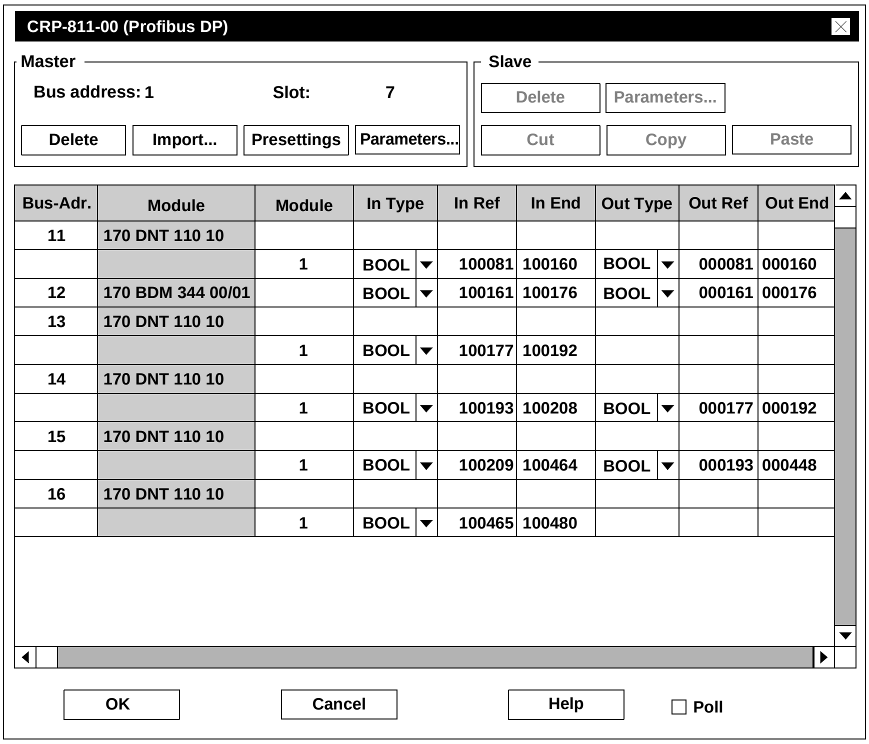
Downloading a Profibus DP Configuration to Concept

To import, proceed as follows:

Step	Action
1	Select the **Import...** command button. **Response:** The **Select Import File** standard window is opened.
2	Enter the path of the previously generated *.CNF file and exit the dialog with **OK**. **Response:** The transfer of the *.CNF file is displayed in the **Import Status** dialog.
3	Close the dialog after the transfer (100%). **Response:** The imported configuration is displayed in the **CRP-811-00 (Profibus DP)** dialog.

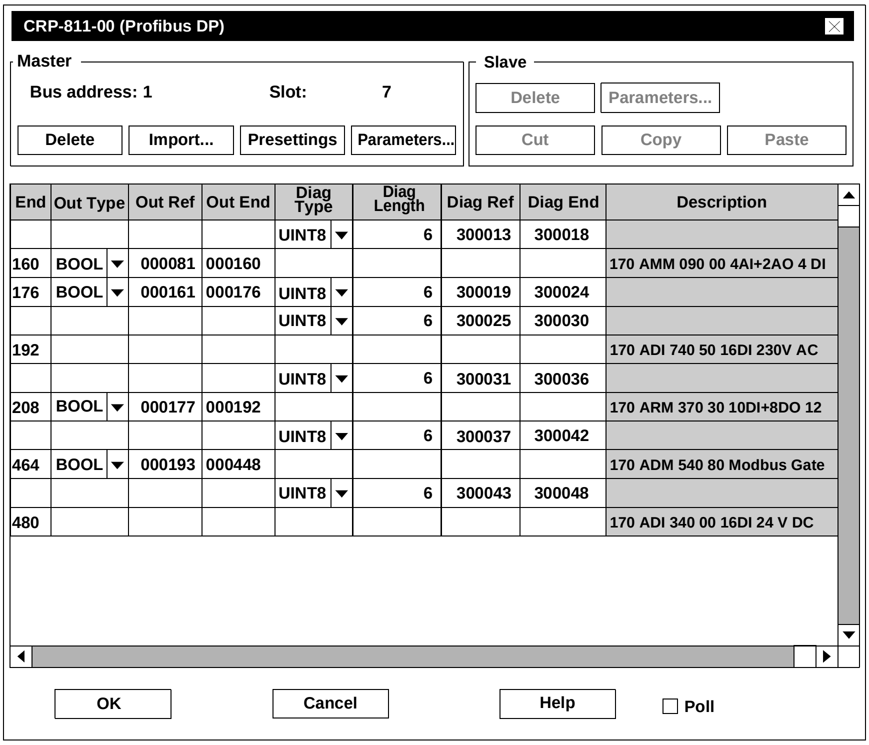
Dialog Representation

Following the import of the configuration, the dialog looks like this (view scrolled all the way to the left):



Bus-Adr.	Module	Module	In Type	In Ref	In End	Out Type	Out Ref	Out End
11	170 DNT 110 10							
		1	BOOL	100081	100160	BOOL	000081	000160
12	170 BDM 344 00/01		BOOL	100161	100176	BOOL	000161	000176
13	170 DNT 110 10							
		1	BOOL	100177	100192			
14	170 DNT 110 10							
		1	BOOL	100193	100208	BOOL	000177	000192
15	170 DNT 110 10							
		1	BOOL	100209	100464	BOOL	000193	000448
16	170 DNT 110 10							
		1	BOOL	100465	100480			

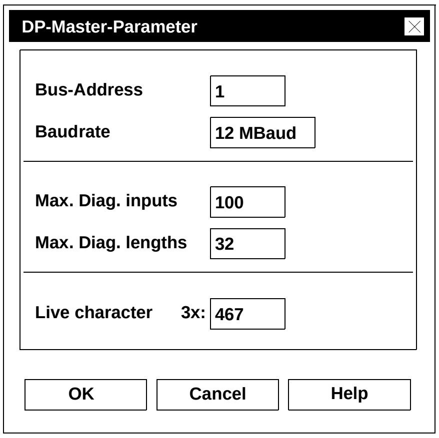
Following the import of the configuration, the dialog looks like this (view scrolled all the way to the right):



Note: In the **Slave** range, the **Parameter...** command button is used for displaying slave parameters. The slave modules are, however, parameterized in SyCon (see SyCon software user manual).

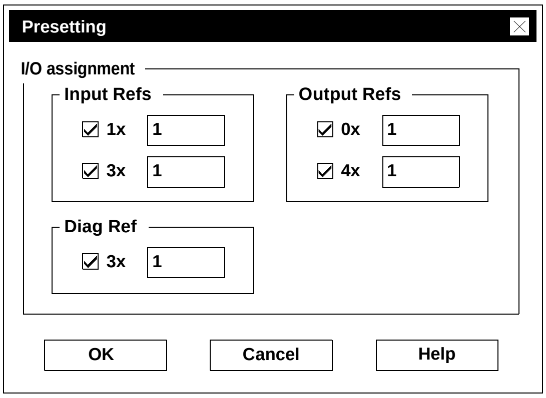
Parameterizing the Master

To parameterize the master, proceed as follows:

Step	Action
1	In the **Master** range, select the **Parameters...** command button. **Response:** The **DP Master Parameters** dialog is opened. Dialog Representation 
2	Accept the defaults, as shown in the figure above, or redefine them.
3	Close the dialog using **OK**. **Response:** You return to the **CRP-811-00 (Profibus DP)** dialog.

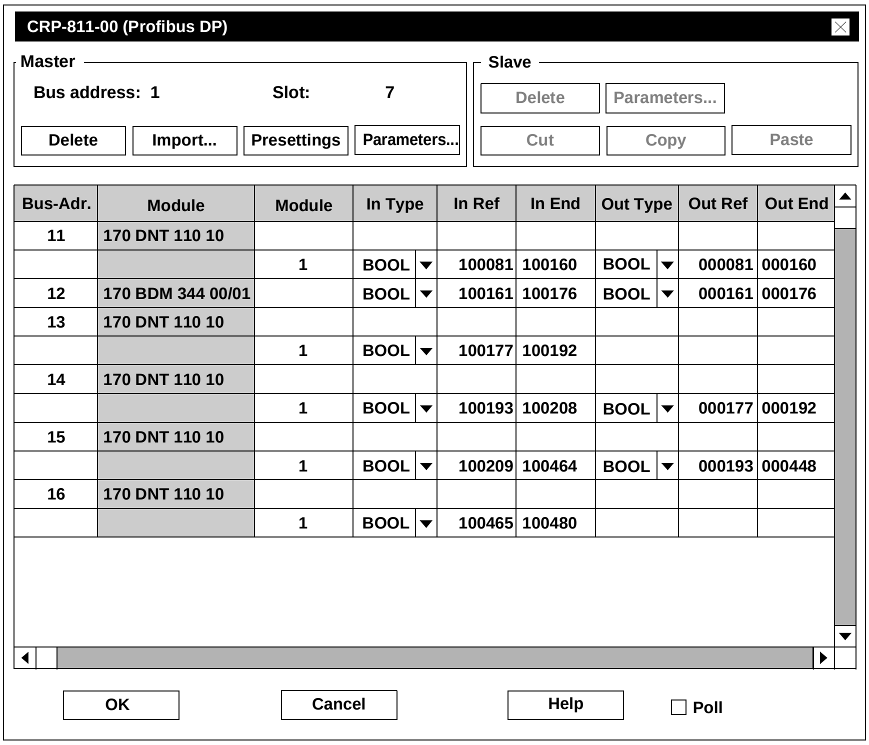
Setting I/O References

To set the I/O references proceed as follows:

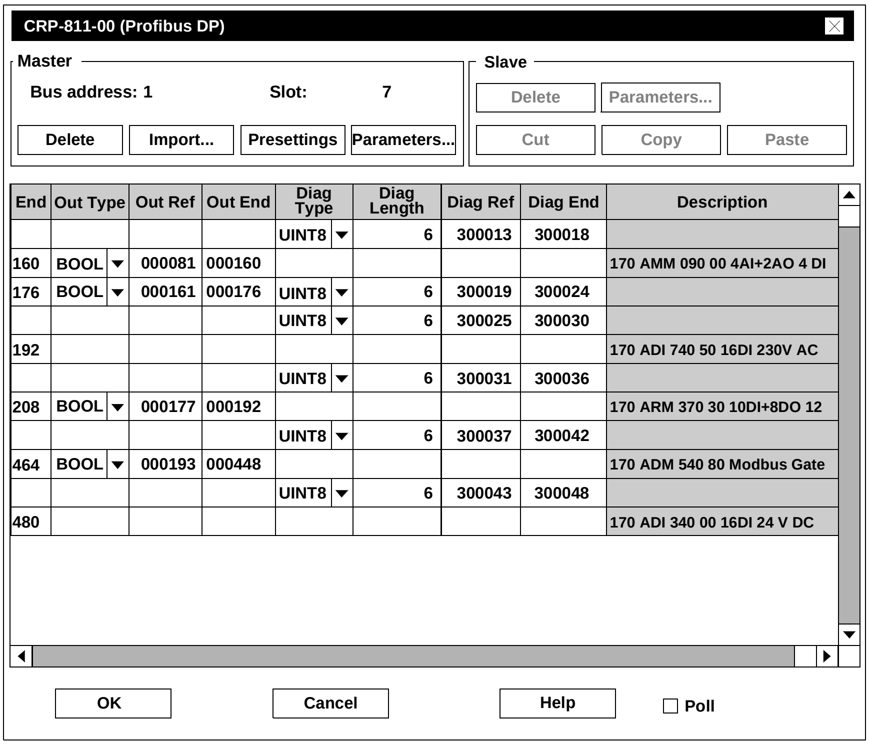
Step	Action
1	Select the command button **Preset. ...**. The **Preset** dialog is opened. Dialog Representation 
2	Accept the defaults, as shown in the figure above, or redefine them.
3	Close the dialog using **OK**. **Response:** You return to the **CRP-811-00 (Profibus DP)** dialog, in which the defined reference ranges have automatically been entered.

Dialog Representation

After the I/O references have been set the dialog looks like this (view scrolled all the way to the left):



After the I/O references have been set the dialog looks like this (view scrolled all the way to the right):



E.7 Quantum-Example - Peer Cop

At a glance

Introduction

In this chapter the configuration of Peer Cop is described step by step.

What's in this Section?

This section contains the following topics:

Generals to Peer Cop

Introduction

Peer Cop is a data exchange service provided by the Modbus Plus network. As an overview, imagine that every Modbus Plus network segment (max. 64 nodes) has a global memory, i.e. a certain number of global variables can be read by every node connected to the same segment. The total amount of global variables depends on the number. of connected (and active) nodes, every node can provide up to 32 words (16 bit) to the global memory. Only the 32 words provided by a node can be written by the same node, all other nodes have read only access to these variables. So by definition, there is a maximum of 64 * 32 words of global memory available to a Modbus Plus network segment. Nodes connected to different (through bridges or gateways) segment cannot share global memory.
When a PLC provides 32 words of global memory it does so by assigning holding registers for broadcast, and when the PLC wants to read global variables provided by another Peer Cop node, assigning holding registers to receive them. These registers are called Global Input (from other nodes) and Global Output (what this node provides) get updated cyclically (in case of a PLC after every scan).

To pass Routing Paths

Actually every Modbus Plus node has its own communication processor (the so called Peer processor), in addition to the processor that controls the node specific work (in case of a PLC: solving user logic).

This leads to some routing paths the global data has to pass to traverse from one node to the other:
- From the data provider (e.g. user logic) to the local (most times embedded peer processor).
- From the local peer processor to the other peer processors (this takes the token cycle time of the Modbus Plus network segment, that depends directly on the number of connected nodes).
- From the peer processor of the data receiver to the data receiver itself, (that is usually the user logic in the receiver PLC).

The actual update time depends on the speed of the Modbus Plus network segment and (that's the big time consumer) the scan times of the data provider and the data receiver.

Send directly

But the sharing of global memory is just the first part of the full Peer Cop service. Since the gobal memory architecture requires a setup (or configuration) for both communication partners, there is another subservice to communicate directly with rather than Configure nodes. This service is somewhat like a master to slave communication, where the master knows what data to send and the slave expects data in a fixed layout and uses this data in a fixed manner (like Terminal I/O). The limit of data that can be sent from the master to the slave is also 32 words. This mode is not global data, since it is sent from one node directly and explicitly to one other node. The sender specifies this as specific output and the receiver as specific input (this specification is hardwired on nonintelligent modules like Terminal I/O). The specific output and input words are also assigned to holding registers when a PLC makes use of this Peer Cop service.
Since both, global and specific data transfer, depend on scan time of the PLC's which provide and use this data in their logic, there is no big performance difference with the transfer from one holding register to the other registers.

Configuration of Peer Cop

Define Peer Cop functionality

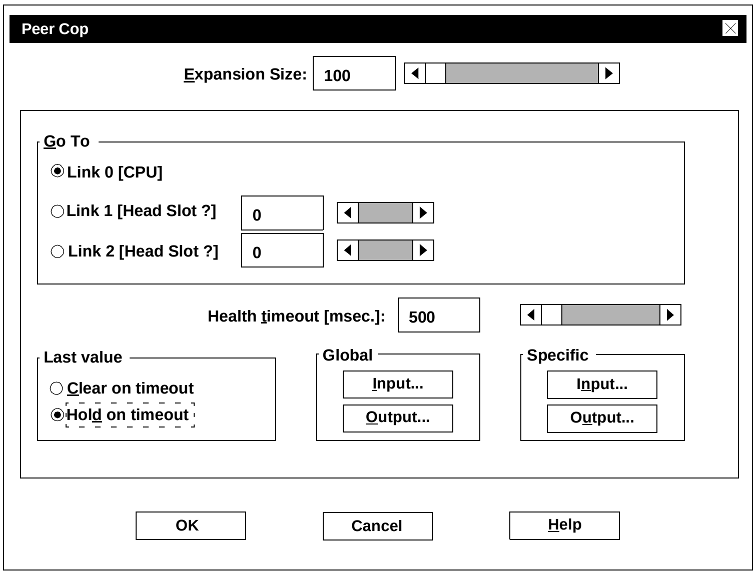
Before configure a Peer Cop you must activate the check box **Peer Cop** in the dialog box **Select Extensions**.

Note: Since every PLC can be connected to up to 3 different Modbus Plus network segments, you can setup Peer Cop for every connection separately (remember Peer Cop is reduced to one segment, it doesn't work through bridges).

Peer Cop settings

To configure a Peer Cop, proceed with the following steps:

Step	Action
1	In the window **PLC Configuration** with the menu command **Config. Extensions** → **Peer Cop** open the dialog box **Peer Cop**.
2	Assume the default value 100 in the text field **Expansion Size:**. **Note:** This text field is just a space of memory (in words) that gets reserved for future changes (in offline mode) that shall not cause the necessity for a complete download (this is especially importent for direct application setup at a plant).
3	Select the option button **Link 0 (CPU)** in the area **Go To**.
4	Assume the default value 500 in the text field **Health timeout (msec.):**. **Note:** The Health timeout value has the same meaning as it has in the I/O map for local and remote I/O.

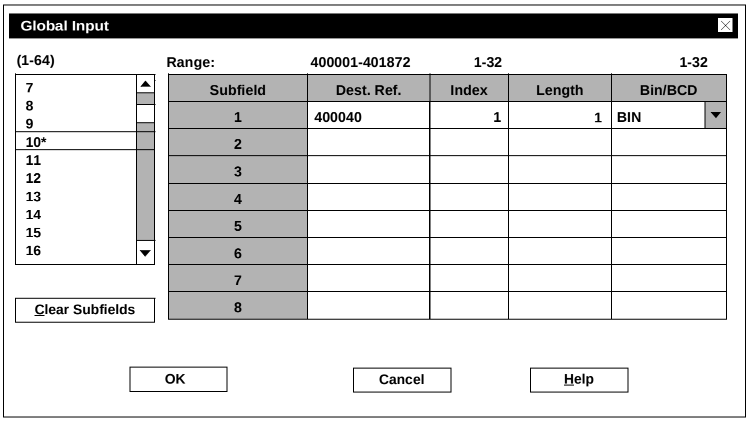
Step	Action
5	Select the option button **Hold on timeout** in the area **Last value**. Representation of the dialog: 

Global data transfer

Global Input

For Global Input proceed as follows:

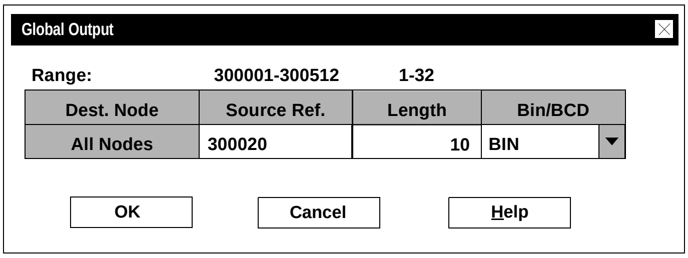
Step	Action
1	For global data transfer open the dialog box **Gobal Input** by clicking the command button **Input...** in the area **Global**.
2	Select node **10** in the list box of the left side of the dialog box.
3	Enter the Destination register, the index, the length and the Bin/BCD Code in the text field of the dialog box, as shown in the figure. Representation of the dialog box:



Step	Action
	Result: The holding register 400040 gets the first word of global output data of node 10, therefore this is global input data for this PLC. If the length value is higher, lets say 2, register 400041 would get the second word of global output data of node 10. The index value declares with what word the assignment shall start, in this case with the first word. The BIN/BCD column gives you the choice of getting the global data formatted either into the usual binary format or into binary coded decimals. The index value may not be higher than 32, since every node can provide a maximum of 32 word only for global output data. The lenght value may also not be higher than 32 for the same reason.
4	Close the dialog box **Global Input** with the command button **OK**.

Global Output

For Global Output proceed as follows:

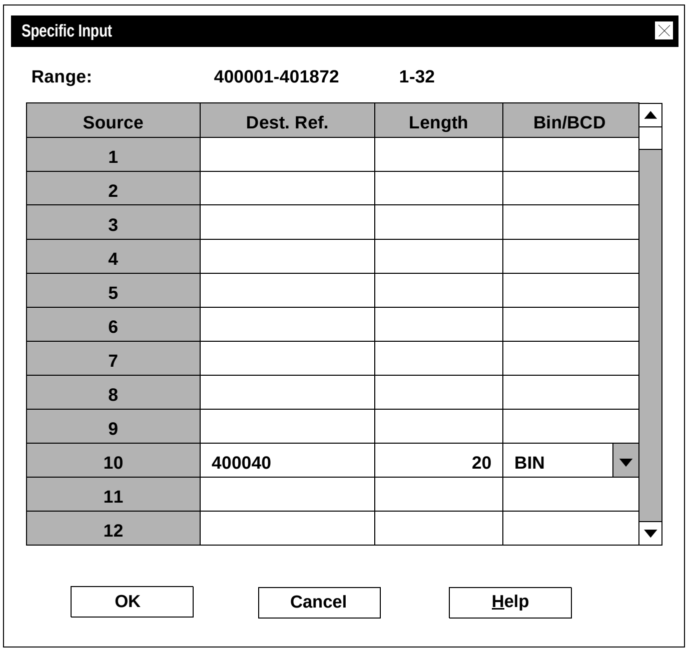
Step	Action
1	Open the dialog box **Gobal Output** by clicking the command button **Output...** in the area **Global**.
2	Enter the Source register, the length and the Bin/BCD Code in the text field of the dialog box, as shown in the figure. Representation of the dialog box: 
3	Close the dialog box **Global Output** with the command button **OK**.

Specific data transfer

Specific Input

For Specific Input proceed as follows:

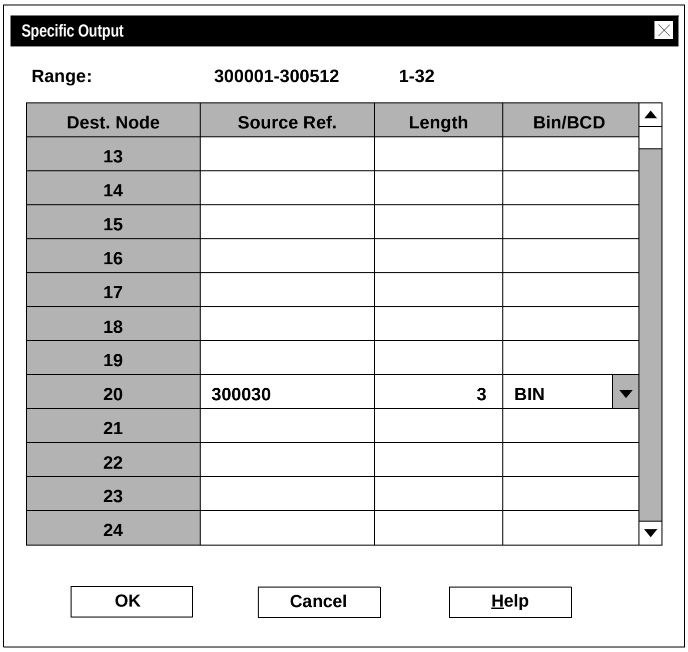
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>For specific data transfer open the dialog box Specific Input by clicking the command button Input... in the area Specific.</td></tr>
<tr><td>2</td><td>Enter the Destination register, the length and the Bin/BCD Code in the text field of the dialog box, as shown in the figure.
Representation of the dialog box:



Result: If node 10 has declared some specific output, which gets delivered with every token cycle on the Modbus Plus network segment (which is usually faster than the updating by the controller’s user logic), that gets sent to holding register 400040. And if it is more than one word, it gets stored in the following holding register, up to 400019 in this example. The formatting can also be either binary or binary coded decimals.</td></tr>
<tr><td>3</td><td>Close the dialog box with the command button OK.</td></tr>
</table>

Specific Output

For Specific Output proceed as follows:

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Open the dialog box Specific Output by clicking the command button Output... in the area Specific.</td></tr>
<tr><td>2</td><td>Enter the Destination Reference register, the length and the Bin/BCD Code option in the text field of the dialog box, as shown in the figure.
Representation of the dialog box:


Result: The values or registers 300030 to 300032 will be sent to node 20 (Target Source) in binary format.</td></tr>
<tr><td>3</td><td>Close the dialog box with the command button OK.</td></tr>
</table>

E.8 Compact Example

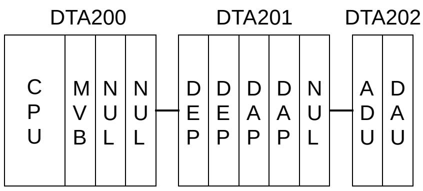
Editing Local Drop

Introduction

This section describes the configuration of the first (local) drop.
When editing the first (local) drop the modules must be set with their I/O references before the individual modules can be parameterized.

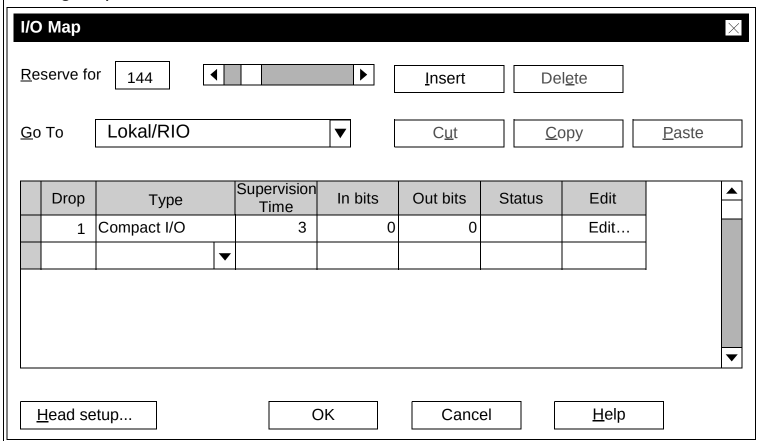
Note: The communication module MVB258A is parameterized in the TCN tool (Train Communication Network). A parameterization file (binary file) is generated and imported into the Concept parameter dialog.

Compact controller



Defining Drops

To define drops proceed as follows in the **PLC Configuration** window:

Step	Action
1	Select **PLC Selection**. **Response:** The **PLC Selection** dialog is opened.
2	Select the **Compact** PLC family and a **PC-E984-258**. Using **OK** return to the **PLC Configuration** window.
3	Select **I/O Map**. **Response:** The **I/O Map** dialog is opened and the first drop is automatically entered in the table. Dialog Representation 
4	Select the drop from the **Drop** column. Select the **Edit...** command button. **Response:** You reach the module map.

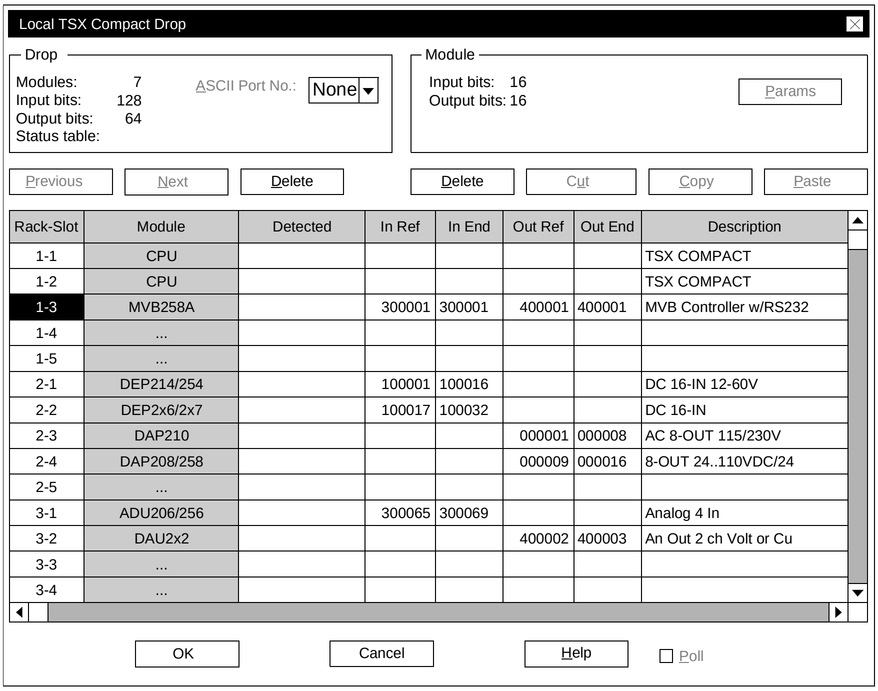
Mapping Modules and Specifying I/O References

To map the modules and specify the address ranges proceed as follows in the **Local TSX Compact Drop** dialog:

Step	Action
1	Select the **Module** → **...** column. **Response:** The **I/O Module Selection** dialog is opened.
2	From the **Category** column, select the **<all>** option. **Response:** All modules are listed in the **Modules** column.
3	In the **Modules** column, select the **MVB258A** module. Exit the dialog with **OK**. **Response:** The module is inserted in the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example (see Dialog Representation **Local TSX Compact Drop**).
5	In the **In Ref** and **Out Ref** columns, set the start references for the input and output modules. **Note:** Discrete Input References have the prefix 1 (e.g. 100001), Coil References have the prefix 0 (e.g. 000001), Input Register References have the prefix 3 (e.g. 300001) and Output Register References have the prefix 4 (e.g. 400001). **Response:** The end reference of the available address range (**In End** or **Out End** column) is entered automatically.

Dialog Representation

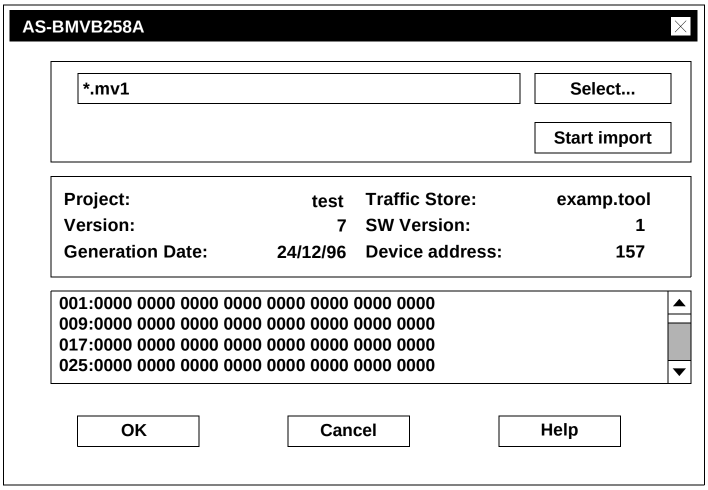
Following module mapping and I/O reference specification, the dialog looks like this:



Rack-Slot	Module	Detected	In Ref	In End	Out Ref	Out End	Description
1-1	CPU						TSX COMPACT
1-2	CPU						TSX COMPACT
1-3	MVB258A		300001	300001	400001	400001	MVB Controller w/RS232
1-4	...						
1-5	...						
2-1	DEP214/254		100001	100016			DC 16-IN 12-60V
2-2	DEP2x6/2x7		100017	100032			DC 16-IN
2-3	DAP210				000001	000008	AC 8-OUT 115/230V
2-4	DAP208/258				000009	000016	8-OUT 24..110VDC/24
2-5	...						
3-1	ADU206/256		300065	300069			Analog 4 In
3-2	DAU2x2				400002	400003	An Out 2 ch Volt or Cu
3-3	...						
3-4	...						

Parameter-ization of Modules

To parameterize the individual modules proceed as follows in the **Local TSX Compact Drop** dialog:

Step	Action
1	From the **Rack Slot** column, select line **1-3**. **Response:** The **1-3** text box has a dark background, i.e. the **MVB258A** module has been selected for editing. **Note:** The **CPU** module is not parameterized.
2	Select the **Params** command button. **Response:** The **AS-BMVB258A** dialog is opened.
3	Select the **Select** command button. **Response:** The **Select MVB Import File** dialog is opened.
4	Set the path of the parameterization file generated in the TCN tool, and exit the dialog using **OK**. **Response:** The selected parameterization file is displayed in the text box in the **AS-BMVB258A** dialog.
5	Select the **Do Import** command button. **Response:** The project data of the parameterization file is transferred to Concept and displayed in the lower list box. Dialog Representation 
6	Exit the dialog using **OK**.
7	Repeat steps 1 to 2 for all the modules in the example. **Note:** The modules are sometimes parameterized differently. Help with this can be obtained from the corresponding help texts in the parameter dialog.

E.9 Atrium Example – INTERBUS Controller

Introduction

Overview

This Chapter contains the step-by-step process for the configuration of an INTERBUS controller with Atrium (PC based).

What's in this Section?

This section contains the following topics:

General Information

Introduction

The configuration of the INTERBUS is done using the PHÖNIX software CMD. It is initially stored as a file (*.SVC). This generated file is loaded into Concept and is visible in the I/O map of the configurator.
Before the INTERBUS nodes are imported, set up the first drop (Atrium I/O) with the CPU board (180-CCO-121-01, 180-CCO-241-01 and 180 CCO 241 11) and the INTERBUS master (CRP-660-00). A maximum of two INTERBUS masters may be inserted. The diagnosis of the field bus can take place in the CRP-660-00 register in Concept.

INTERBUS Export Settings in CMD

Introduction

The CMD tool (Configuration Monitoring and Diagnostic tool) is used to configure the INTERBUS. For information about this, refer to the corresponding chapter in the Phoenix user manual

Preconditions

The serial interface of the host computer and the diagnostic interface of a PC104 board must be linked with a V24 cable.

Implementing Export Settings

Before you import the configuration into Concept, carry out the following settings in the CMD tool:

Step	Action
1	Select **Configuration** → **Controller Board** → **Data Type...**.
2	Select **IBS PC104 SC-T** and confirm the selection using **OK**.
3	Select **File** → **Operating State...**. **Response:** The **Operating State** dialog is opened.
4	Activate the **Configuration (Online)** option button and exit the dialog using **OK**.
5	Under **Configuration** → **Controller Board** → **Control** select the command **Activate Configuration Frame**. Confirm with **Yes**. **Response:** A configuration frame is generated.
6	Select **Configuration** → **Configuration Frame** → **Read Again**. **Response:** The configuration is read into the frame.
7	Under **Configuration** → **Parameterization Memory** → **Write ASCII File** select the command **INTERBUS Data (*.SVC)...**. **Response:** The INTERBUS data is stored in a file.
8	Enter the directory and the file name in the open dialog and confirm the entry using **OK**.
9	Select **File** → **Save As...**. **Response:** The INTERBUS project is saved.

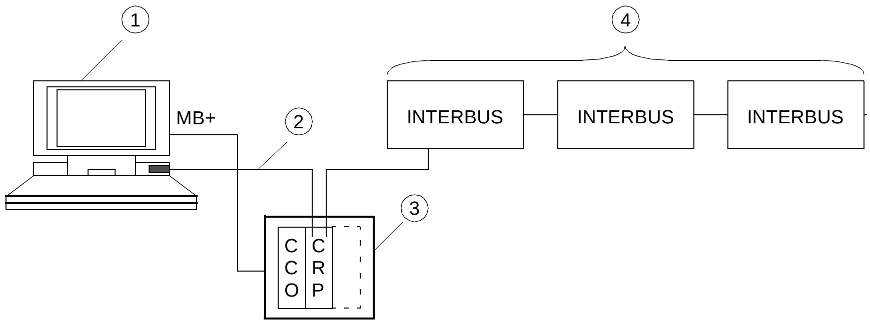
Editing Local Drop

Introduction

This section describes the configuration of the first (local) drop. The processing sequence begins first of all with the definition of all drops.

Note: For an error free transfer of the INTERBUS configuration, it should be ensured that sufficient memory is available. To optimise the storage allocation open the **PLC Memory Partition** dialog (**PLC Configuration** → **PLC Memory Partition**).

When editing the first (local) drop the modules must be set with their I/O references before parameters can be set forindividual modules.
Atrium –INTERBUS Controller



1 Host computer for Concept and CMD

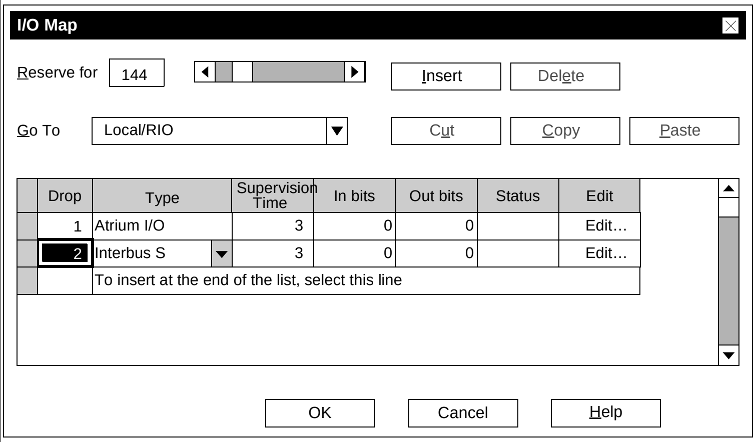
2 V24 cable

3 PC104 board on a standard AT board

4 INTERBUS configuration

Defining Drops

To define drops proceed as follows in the **PLC Configuration** window:

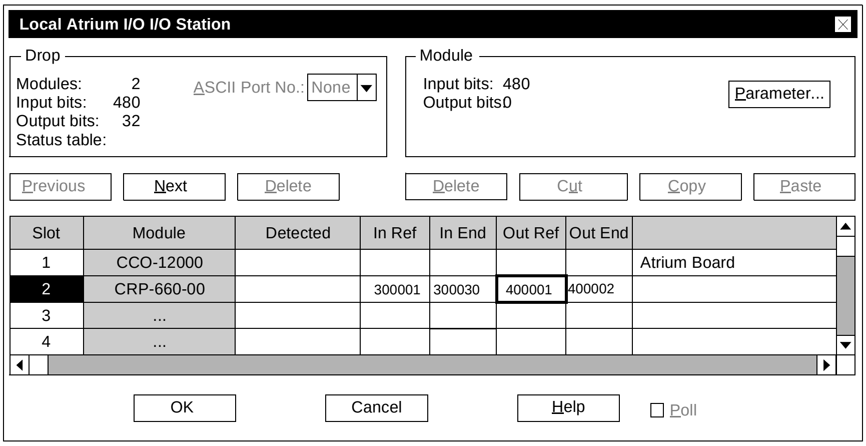
Step	Action
1	Select **PLC Selection**. **Response:** The **PLC selection** dialog is opened.
2	Select the **Atrium** PLC family and a **180-CCO-121-01**. Using **OK** return to the **PLC Configuration** window.
3	Select **I/O Map**. **Response:** The **I/O Map** dialog is opened and the first drop is automatically entered in the table.
4	Enter a 7 in **RIO Slot** and quit the dialog using **OK**. **Response:** The CRP-93x-00 module is automatically inserted in the I/O map (in slot 7) of the selected drop. In the **Go To** list box, the **Local/RIO (Slot 7)** network link is displayed.
5	Select the last line in the table. Select the **Insert** command button. **Response:** The second drop is entered in the **Type** column. **Note:** The number of INTERBUS masters to be inserted is limited to two drops. Dialog display 
6	Select the drop from the **Drop** column. Select the **Edit...** command button. **Response:** You reach the module map.

Setting I/O References

In the **Local Atrium Drop** dialog the INTERBUS master CRP-660-00 is automatically entered in the I/O map.
For the specification of the I/O references, enter the start references in the **In Ref.** and **Out Ref** columns. After the start reference has been entered, the end reference of the available address range of the module is automatically displayed.

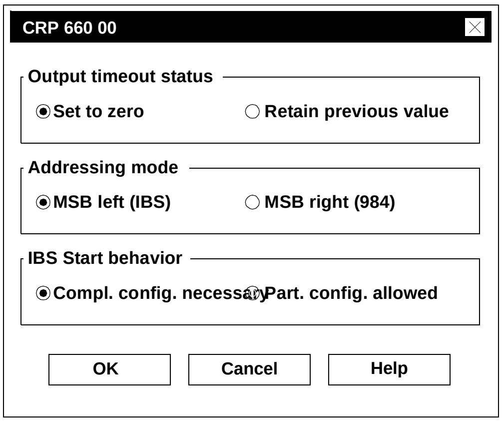
Note: Discrete input references have the prefix 1 (e.g. 100001), coil references have the prefix 0 (e.g. 000001), input register references have the prefix 3 (e.g. 300001) and output register references have the prefix 4 (e.g. 400001).

Dialog display



Set Module Parameters

To set parameters for the INTERBUS master proceed in the following way in the **Local Atrium Drop** dialog:

Step	Action
1	In the **Slot** column, select line **2**. **Response:** The **2** text box has a dark background, i.e. the **CRP-660-00** module has been selected for editing. **Note:** Parameters are not set for **CCO-12000** module.
2	Select the **Params** command button. **Response:** The **CRP-660-00** dialog is opened.
3	Press the option buttons as shown in the following figure and exit the dialog using **OK**. **Note:** Help with setting parameters is obtained via the dialog’s help text. Dialog display 

Editing Remote Drop

To edit the remote drop open the **INTERBUS Drop. 2** dialog. This dialog is reached via the **I/O map** dialog by pressing the **Edit...** command button in the second drop. (INTERBUS).

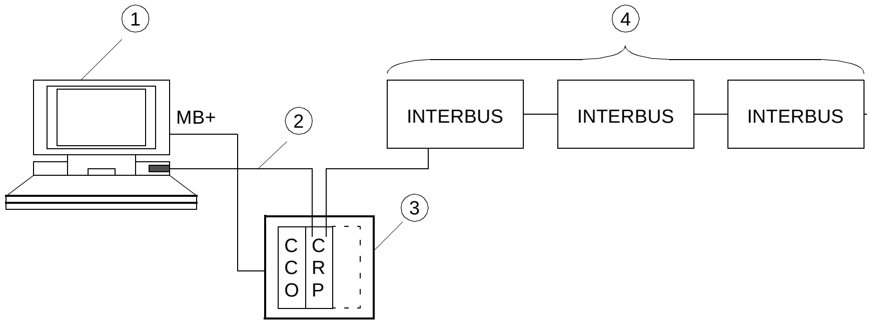
Editing Remote Drop (Importing INTERBUS Configuration)

Introduction

This section describes the import of the INTERBUS configuration. The assignment of I/O references takes place in the import dialog before the transfer of the configuration file is performed.

Note: The module parameters are set in the CMD tool (see CMD tool user manual), because the imported modules are not recognized in Concept.

Atrium –INTERBUS Controller



1 Host computer for Concept and CMD

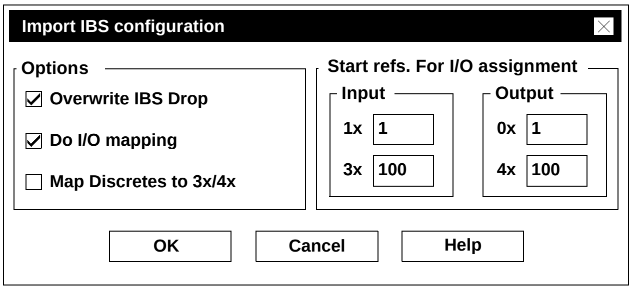
2 V24 cable

3 PC104 board on a standard AT board

4 INTERBUS configuration

Setting I/O References

To specify the address ranges proceed as follows in the **INTERBUS Drop 2** dialog:

Step	Action
1	Select the **Import...** command button. **Response:** The **Import IBS Configuration** dialog is opened.
2	Check the **Overwrite IBS Drop** box. **Response:** The **Do I/O Mapping** check box becomes available.
3	Check the **Do I/O Mapping** check box. **Response:** The **Digital Module in 3x/4x Range** check box and the **Input 3x** and **Output 4x** text boxes become available.
4	Uncheck the **Digital Module in 3x/4x Range** check box. **Response:** The **Input 1x** and **Output 0x** text boxes become available.
5	Enter the value 100 in the **Input 3x** and **Output 4x** text boxes. **Response:** The 3x and 4x address ranges of the imported module begin with the start references 300100 and 400100. **Note:** The 1x and 0x address ranges retain the default value 1, i.e. these address ranges begin with the start references 100001 and 000001. Dialog display 
6	Exit the dialog with **OK**. **Response:** The **Select Import File** dialog is opened.
7	Enter the path of the configuration file *.SVC. Select **OK**. **Response:** The Dialog **Import Status** dialog is opened, the file transfer starts, and the import status is displayed.
8	Following the transfer (100%), close the dialog. **Response:** The imported INTERBUS configuration is displayed in the **INTERBUS Drop 2** dialog in the I/O map.

Dialog display

Following the import of the INTERBUS configuration, the dialog looks like this:

Interbus 8 I/O St. 2

Drop
Modules: 12
Input bits: 144
Output bits: 206
Status table:
ASCII Port No.: None

Module
Input bits: 0
Output bits:16
Parameter...

Previous | Next | Delete | Delete | Cut | Copy | Paste

No.	Module	Detected	In Ref	In End	Out Ref	Out End	Description
1	BK-012-00						
2	DIO-003-16		100001	100016	000001	000016	
3	DIO-011-16		100017	100032	000017	000032	
4	DI-130-16		100033	100048			
5	AD-065-64				400100	400103	
6	DO-129-16				000033	000048	
7	BK-052-00						
8	AIO-067-64		300100	300103	400104	400107	
9	PCP-203-00						
10	DI-002-16		100049	100064			
11	DO-001-16				000049	000064	
12	DIO-003-16		100065	100080	000065	000080	
13	...						
14	...						

OK | Cancel | Help | Poll

E.10 Momentum Example - Remote I/O Bus

Introduction

Overview

This Chapter contains the step-by-step process for the configuration of a remote I/O bus (Momentum).

What's in this Section?

This section contains the following topics:

General Information

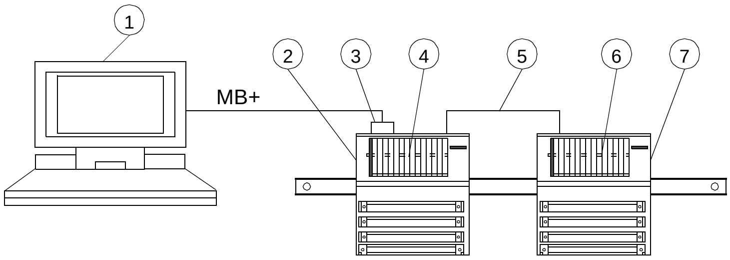
Introduction

TSX Momentum is a modular system. Bus adapters (e.g. 170 INT 110 00) and CPU adapters (e.g. 171-CCC-760-10-IEC) work in conjunction with an I/O unit as independent modules. In order to function properly, each I/O unit must be equipped with an adapter.

Editing local drop

Introduction

This section describes the configuration of the first (local) drop. The processing sequence begins first of all with the definition of all drops.
When editing the first (local) drop the modules must be set with their I/O references before parameters can be set forindividual modules.
Momentum – remote controller with I/O bus



1 Host Computer

2 I/O unit e.g. 170-AAI-030-00

3 Interface adapter

4 CPU adapter e.g. 171-CCC-760-10-IEC

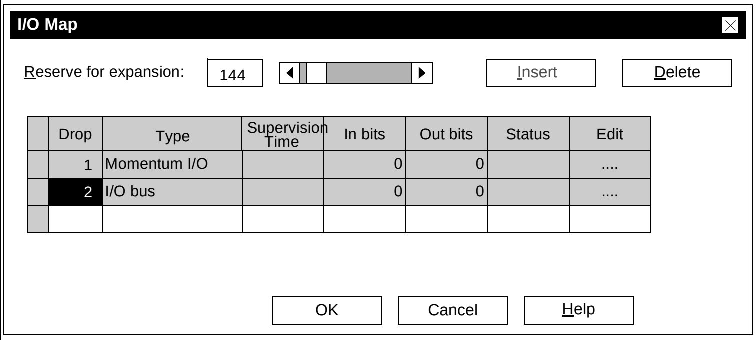
5 I/O bus interface e.g. 172-PNN-210-22

6 Bus adapter e.g. 170-INT-110-00

7 I/O unit e.g. 170-AMM-090-00

Defining Drops

To define drops proceed as follows in the **PLC Configuration** window:

Step	Action
1	Select **PLC Selection**. **Response:** The **PLC selection** dialog is opened.
2	Select the PLC family **Momentum** and CPU **171-CCC-760-10-IEC**. Use **OK** return to the **PLC Configuration** window.
3	Select **I/O Map**. **Response:** The **I/O Map** dialog is opened and the first drop is automatically entered in the table.
4	Select the last line in the table. Select the **Insert** command button. **Response:** The second drop is entered in the **Type** column. **Note:** Only one I/O bus can be configured. Dialog display 
5	Select the drop from the **Drop** column. Select the **Edit...** command button. **Response:** You reach the module map.

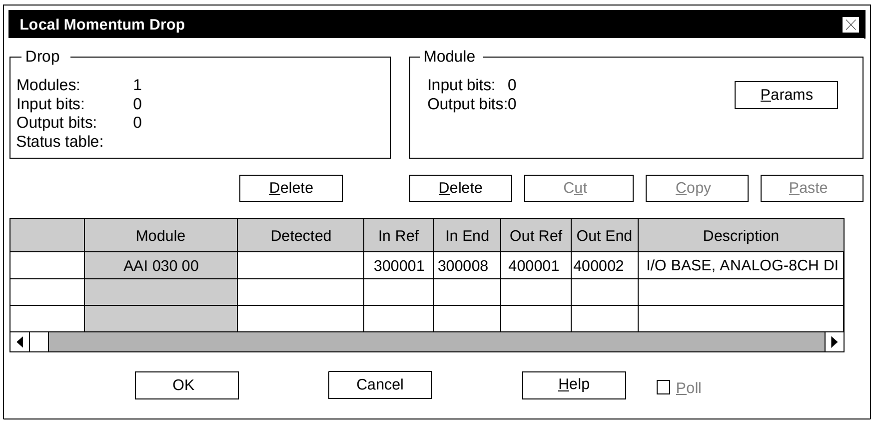
Mapping Modules and Specifying I/O References

To map the modules and specify the address ranges proceed as follows in the **Local Quantum Drop** dialog:

Step	Action
1	Select the **Module** → **...** column. **Response:** The **I/O Module Selection** dialog is opened.
2	From the **Category** column, select the **<all>** option. **Response:** All modules are listed in the **Modules** column.
3	Select from the column **Modules**, the module**AAI-030-00**. Exit the dialog with **OK**. **Response:** The module is inserted in the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example (see **Local Momentum drop**).
5	In the **In Ref** and **Out Ref** columns, set the start references for the input and output modules. **Note:** Discrete Input References have the prefix 1 (e.g. 100001), Coil References have the prefix 0 (e.g. 000001), Input Register References have the prefix 3 (e.g. 300001) and Output Register References have the prefix 4 (e.g. 400001). **Response:** The end reference (column **In.End.** or **Out.End**) of the available address range is automatically entered.

Dialog display

Following module mapping and I/O reference specification the dialog looks like this:

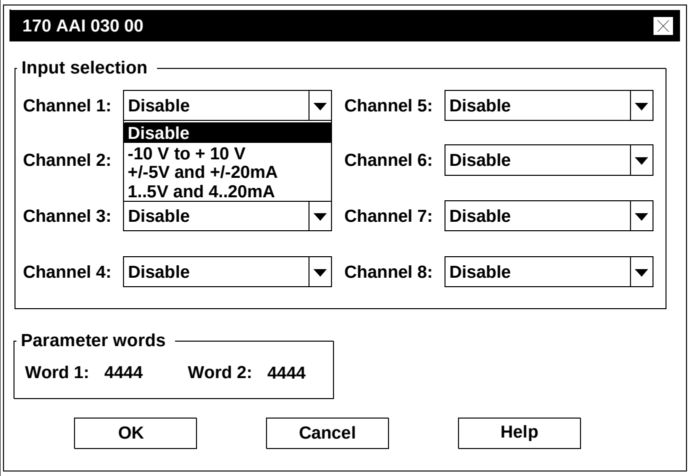


Note: With this addressing the 8 measurements of the AAI-030-00 are to be found in the words 300001-300008. The parameters are in the words 400001 and 400002.

Set Module Parameters

To set parameters for the module proceed as follows in the **Local Momentum Drop** dialog:

Step	Action
1	Select the **Params** command button. **Response:** The **170-DDI-353-00** dialog is opened.
2	Select the signal conditions for the input and output channels from the list boxes and exit the dialog using **OK**. **Note:** Help with this can be obtained from the corresponding help text in the parameter dialog. **Response:** The parameter settings are automatically allocated to the addresses 400001 and 400002. Dialog display



Step	Action
3	Exit the dialog using **OK**. **Response:** You return automatically to the **I/O Map** dialog.

Editing Remote Drops (I/O bus)

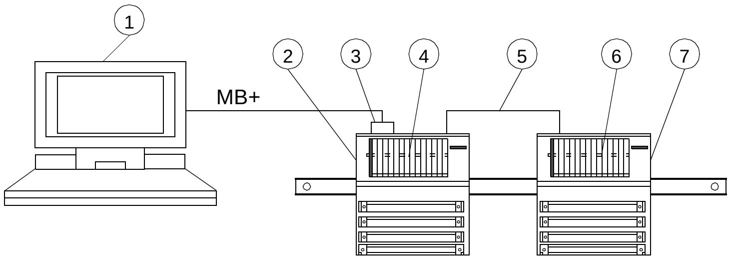
To edit the remote drop open the **RIO I/O Bus Drop** dialog. This dialog is reached via the **I/O Map** dialog by pressing the **Edit...** command button in the second drop (I/O bus).

Example 10 – Editing Remote Drops (I/O Bus)

Introduction

This section describes the configuration of the Momentum I/O bus. The drop has already been defined in Editing the First (local) Drop (See *Editing local drop, p. 888*). When editing the I/O bus the modules must be specified with their I/O references before the individuals modules can be parameterized.

Momentum – remote controller with I/O bus



1 Host Computer
2 I/O unit e.g. 170-AAI-030-00
3 Interface adapter e.g. 172-PNN-210-22
4 CPU adapter e.g. 171-CCC-760-10-984
5 I/O bus interface
6 Bus adapter e.g. 170-INT-110-00
7 I/O unit e.g. 170-AMM-090-00

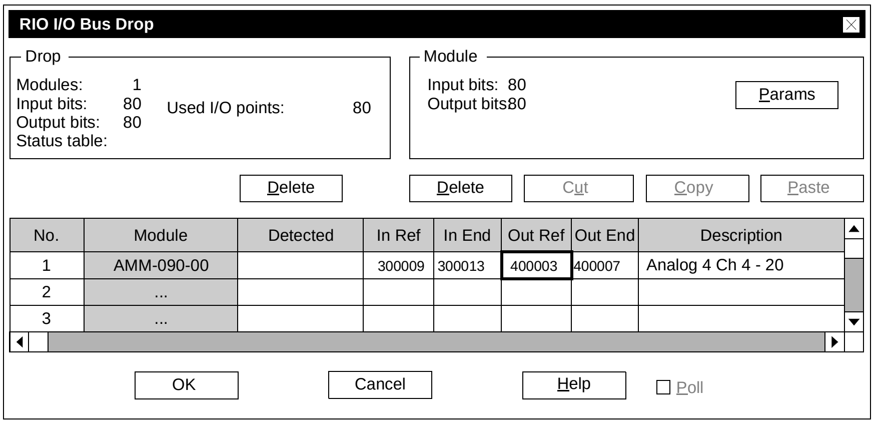
Mapping Modules and Specifying I/O References

To map the modules and specify the address ranges proceed as follows in the **Local TSX Compact Drop** dialog:

Step	Action
1	Select the **Module** → **...** column. **Response:** The **I/O Module Selection** dialog is opened.
2	From the **Category** column, select the **<all>** option. **Response:** All modules are listed in the **Modules** column.
3	Select from the column **Modules**, the module **AMM-090-00**. Exit the dialog with **OK**. **Response:** The module is inserted in the I/O map.
4	Repeat steps 1 to 3 for all the modules in the example (see **Local Quantum Drop** dialog representation).
5	In the **In Ref** and **Out Ref** columns, set the start references for the input and output modules. **Note:** Discrete Input References have the prefix 1 (e.g. 100001), Coil References have the prefix 0 (e.g. 000001), Input Register References have the prefix 3 (e.g. 300001) and Output Register References have the prefix 4 (e.g. 400001). **Response:** The end reference (column **In.End.** or **Out.End**) of the available address range is automatically entered.

Dialog display

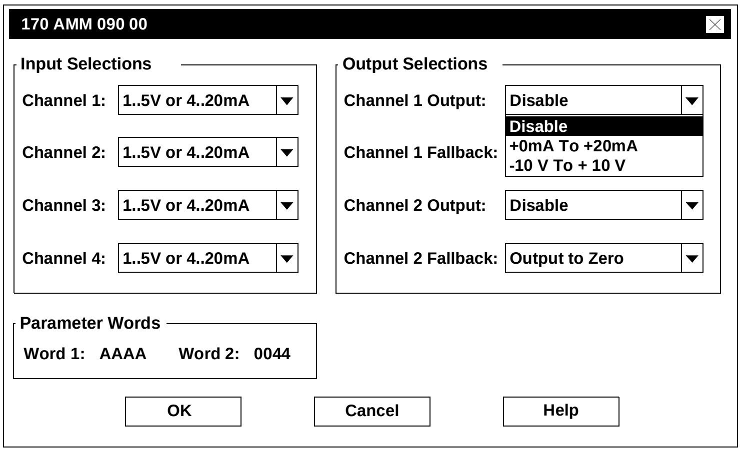
Following module mapping and I/O reference specification the dialog looks like this:



Note: With this addressing, the 4 measurements of the AMM-090-00 are to be found in the words 300009-300013. The parameters are in the words 400009-400013.

Set Module Parameters

To set parameters for the module proceed as follows in the **RIO I/O bus drop** dialog:

Step	Action
1	Select column **No.** line **1**. **Response:** The **1** text box has a dark background, i.e. the **AMM-090-00** module has been selected for editing.
2	Select the **Params** command button. **Response:** The **170-AMM-090-00** dialog is opened.
3	Select the signal states for the input and output channels from the list boxes and exit the dialog using **OK**. **Note:** Help with this can be obtained from the help text in the parameter dialog. **Response:** The parameter settings are automatically allocated to the addresses 400009-400013. Dialog display 

E.11 Momentum Example - Ethernet Bus System

Introduction

Overview

This chapter contains step-by-step instructions for the configuration of an Ethernet bus system with Momentum.

What's in this Section?

This section contains the following topics:

Configure Ethernet

Preconditions

To configure an Ethernet bus system, the following preconditions must be fulfilled:
- PCI network cards in the host computer
- Installation of the network card driver
- Setting Ethernet interface parameters
- Addressing the M1 Ethernet CPU

Installing the PCI network card

For a link to an Ethernet bus system an Ethernet interface located on a PCI network card must be available in the host computer. This card can be upgraded in PCs, as long as a PCI slot is available. Information about this can be found in the computer manufacturer's user manual.

Network configuration

Network configurations for different operating systems are given in section *Network Configuration in Different Operating Systems, p. 897*.

Installing Drivers

Following the installation of the PCI network card the drivers, which come with the network card, must be installed.
To proceed further, the IP address of the network card is required (it may be necessary to contact network administrator).

Addressing the M1 Ethernet CPU

The M1 Ethernet CPU does not have an IP address when supplied, and must therefore be determined in the **Ethernet / I/O Scanner** dialog. The address for the gateway and Subnet Mask is also determined in this dialog.
The IP address can be assigned via the system administrator or the BOOTP server.

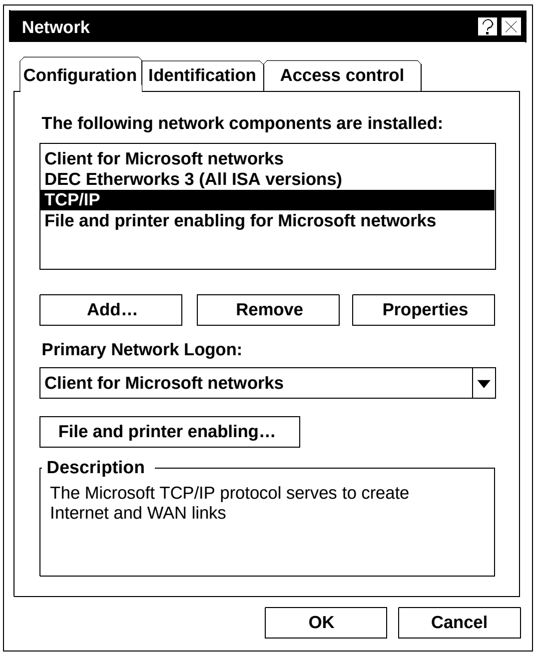
> **Note:** It is important to ensure that the IP address has not already been assigned to another device. Double addressing causes an unforeseeable function in the network.

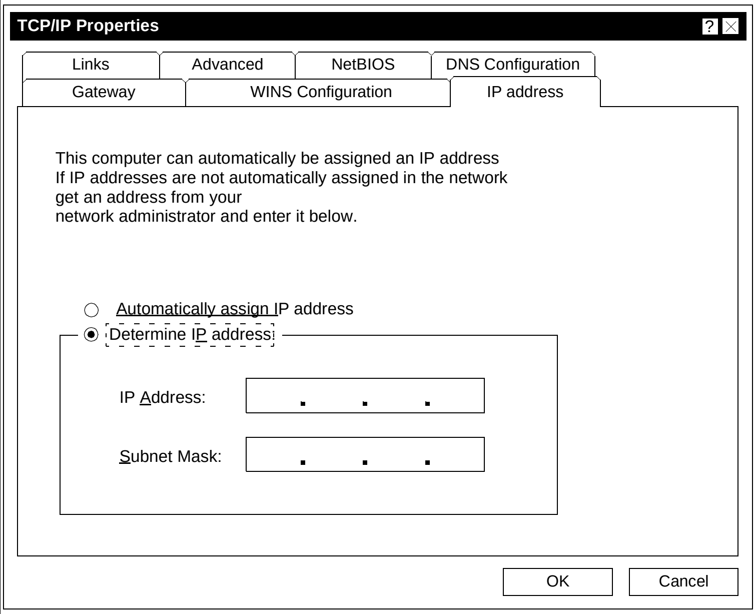
After addressing, saving to Flash is recommended (**Online Control Panel** → **Flash Program...**), so that the settings are not lost in case of a power outage.

Network Configuration in Different Operating Systems

Network configuration in Win 98

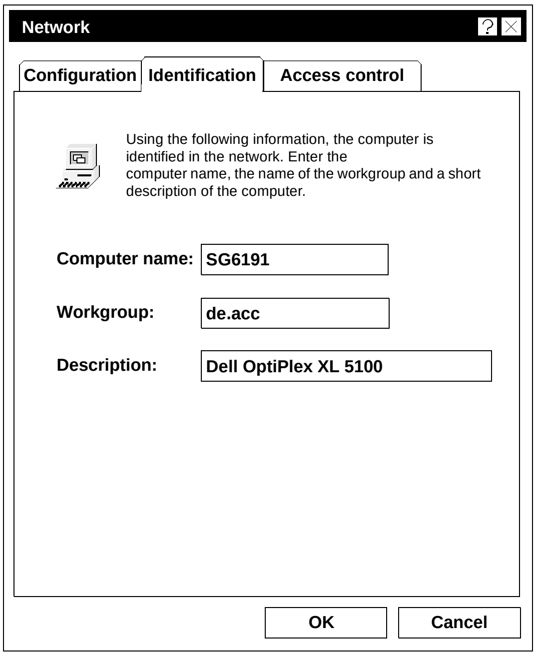
Declare this IP address in the operating system as follows:

Step	Action
1	Select **Start** → **Settings** → **Control Panel** → **Network**. **Response:** The **Network** dialog box is opened. Dialog display 
2	Select the register **Configuration**. Select the network connection **TCP/IP**.

Step	Action
3	Select the **Properties** command button. **Response:** The **TCP/IP Properties** dialog is opened. Dialog display 
4	Select the register **IP Address** and make the following settings. **Response:** The programming device is then registered for network operation with the IP address.

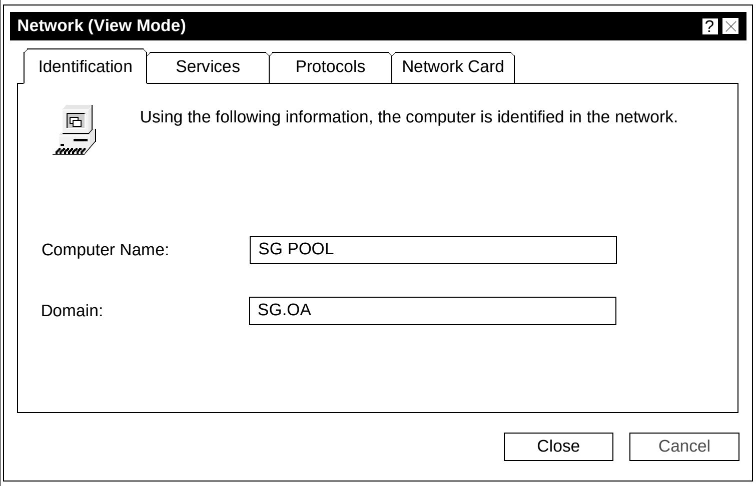
Computer Identification in Win 98/NT

The information is used to identify the computer in the network:

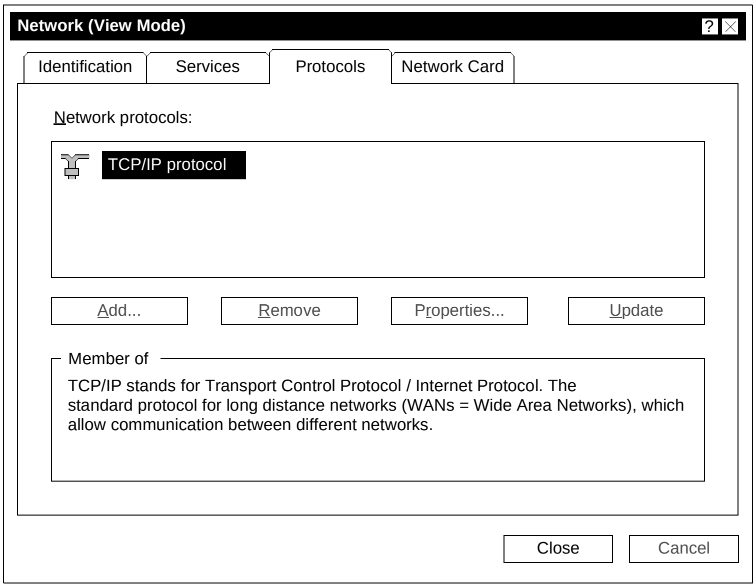
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Select Start → Settings → Control Panel → Network.
Response: The Network dialog box is opened.</td></tr>
<tr><td>2</td><td>Select the register Identification.
Enter the computer name, the name of the workgroup and a short description of the computer.
Dialog display

</td></tr>
<tr><td>3</td><td>Exit the dialog using OK.</td></tr>
</table>

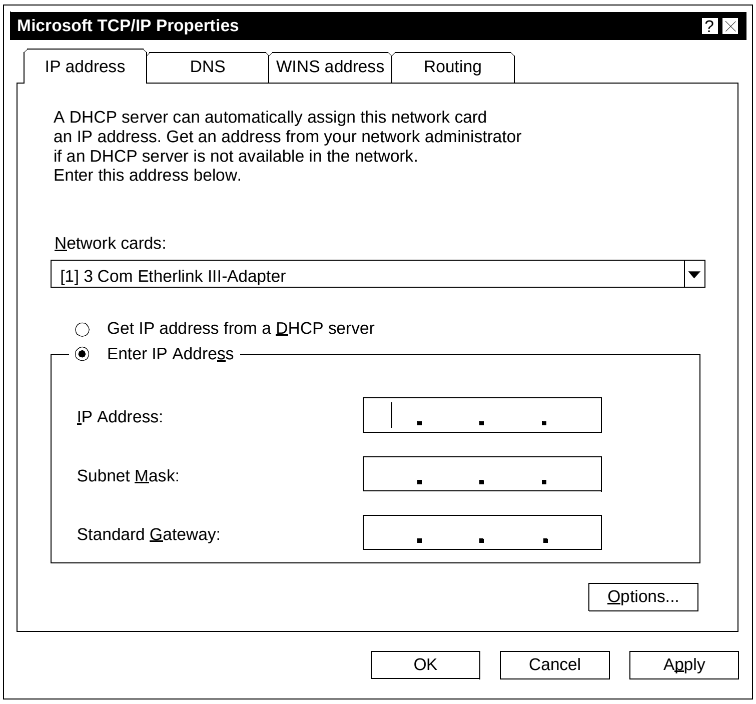
Network configuration in Win NT

Declare this IP address in the operating system as follows:

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Select Start → Settings → Control Panel → Network.
Response: The Network dialog box is opened.
Dialog display


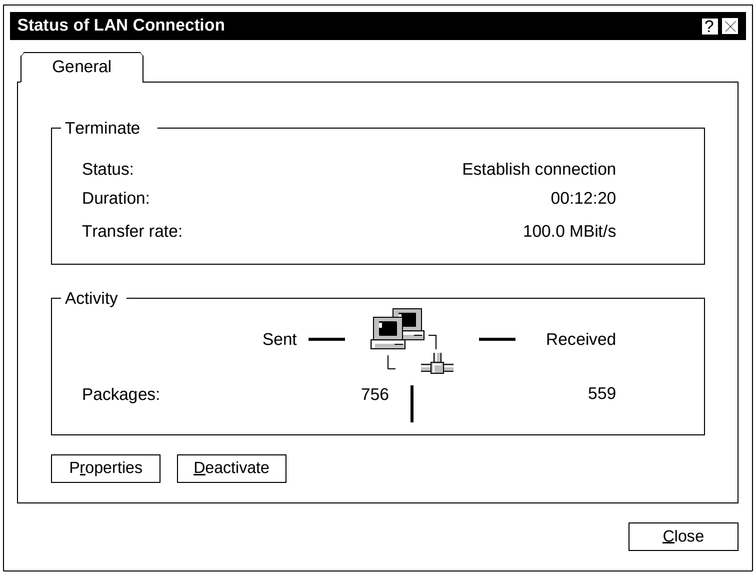
</td></tr>
</table>

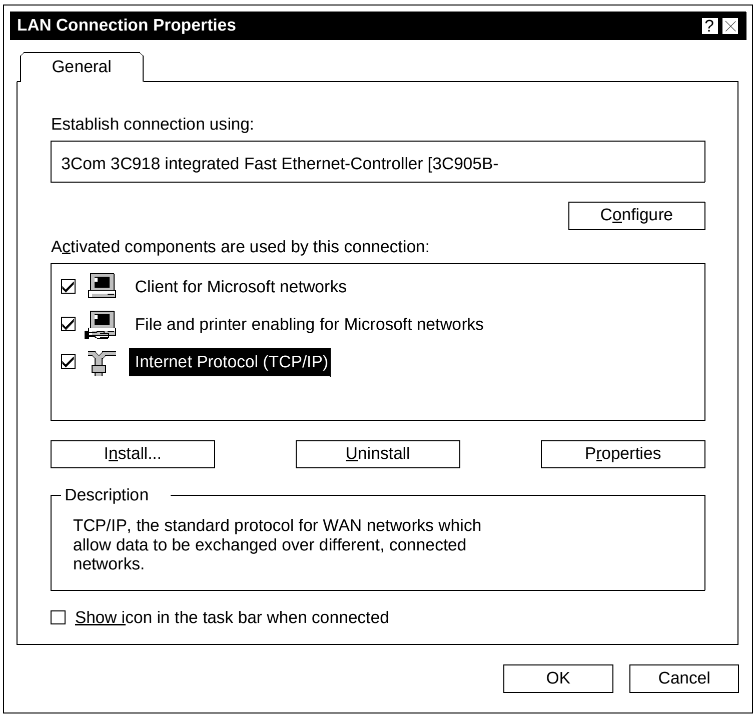
Step	Action
2	Select the register **Protocols**. Dialog display  Select the network connection **TCP/IP Protocol**.

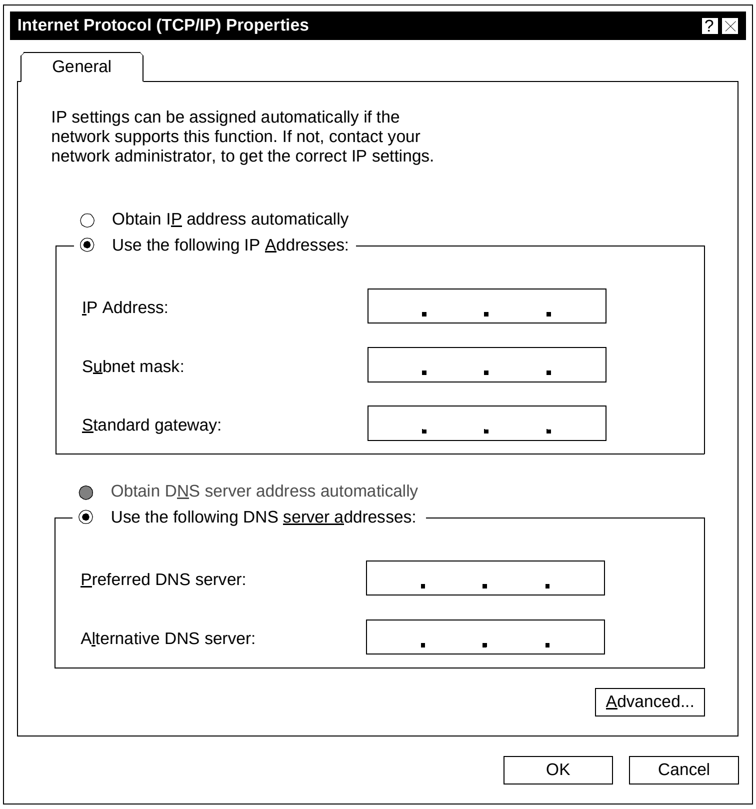
Step	Action
3	Select the **Properties** command button. **Response:** The **Microsoft TCP/IP Properties** dialog box is opened. Dialog display 
4	Select the register **IP Address** and make the following settings. **Response:** The programming device is then registered for network operation with the IP address.

Network configuration in Win 2000

Declare this IP address in the operating system as follows:

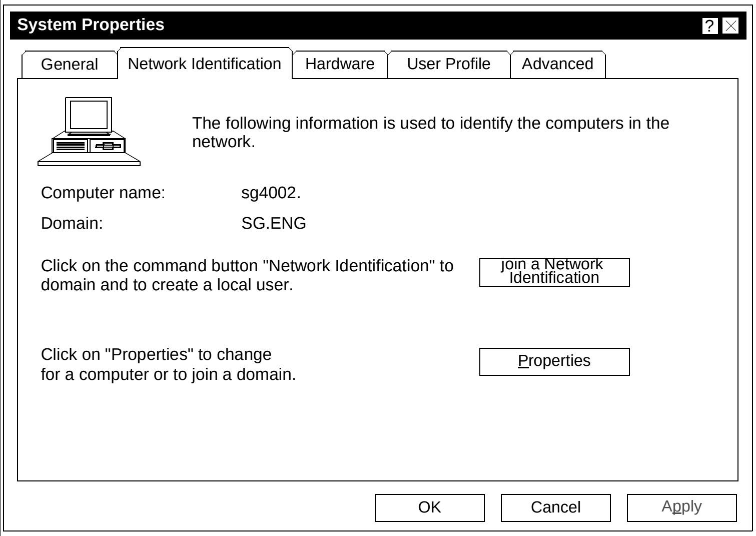
<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Select Start → Settings → Network and Dial-Up Connections.
Response: The Network and Dial-Up Connections window is opened.</td></tr>
<tr><td>2</td><td>Select the LAN Connection icon.
Response: The LAN Connection status dialog box is opened.
Dialog display

</td></tr>
</table>

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>3</td><td>Select the Properties command button.
Response: The LAN Connection Properties dialog box is opened.
Dialog display

</td></tr>
<tr><td>4</td><td>Select the network connection Internet Protocol (TCP/IP).</td></tr>
</table>

Step	Action
5	Select the **Properties** command button. **Response:** The **Internet Protocol (TCP/IP) Properties** dialog box is opened. Dialog display 
6	Make the settings there. **Response:** The programming device is then registered for network operation with the IP address.

Computer Identification in Win 2000

The information is used to identify the computer in the network:

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Select Start → Settings → Control Panel → System.
Response: The System Properties window is opened.</td></tr>
<tr><td>2</td><td>Select the register Network Identification.
Dialog display


</td></tr>
<tr><td>3</td><td>Select the Network ID command button.
Response: The assistant for creating a user on the network is started.
Or select the Properties command button.
Response: The Identification Changes dialog box is opened.</td></tr>
<tr><td>4</td><td>Exit the dialog using OK.</td></tr>
</table>

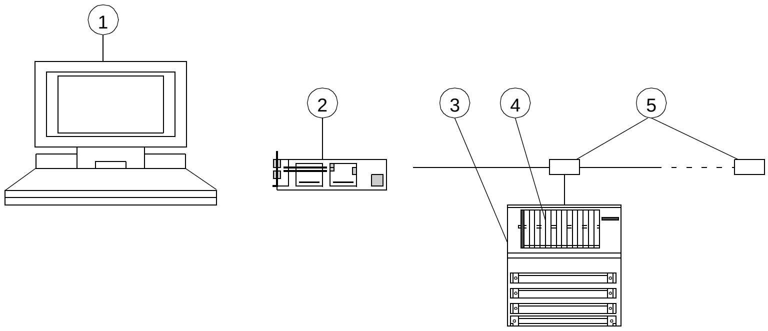
Editing local drop

Introduction

This section describes the configuration of the local I/O station (Drop). The processing sequence begins first of all with the definition of the drop.
When editing the local I/O station (Drop) the I/O unit must be specified with its I/O references before parametering of the individual assemblies can take place.

Note: Only particular CPUs can be used for the Ethernet bus configuration.
The following CPUs are available:
- 171 CCC 980 30
- 171 CCC 960 30
- 171 CCC 980 20
- 171 CCC 960 20

Momentum - Ethernet Bus System



1 Host Computer

2 Ethernet network card

3 I/O unit e.g. 170-AMM-090-00

4 CPU adapter e.g. 171-CCC-960-20-IEC

5 Hub or Switch

Defining Drops

To define drops proceed as follows in the **PLC Configuration** window:

Step	Action
1	Select **PLC Selection**. **Response:** The **PLC selection** dialog is opened.
2	Select the PLC family **Momentum** and CPU **171-CCC-960-20-IEC**. Use **OK** return to the **PLC Configuration** window.
3	Select **I/O Map**. **Response:** The **I/O Map** dialog is opened and the first drop is automatically entered in the table.
4	Select the drop from the **Drop** column. Select the **Edit...** command button. **Response:** You reach the module map.

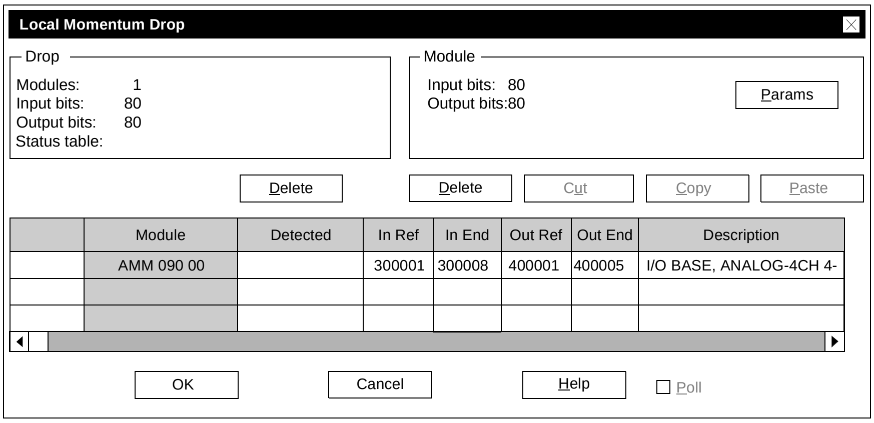
Mapping Modules and Specifying I/O References

To map the modules and specify the address ranges proceed as follows in the **Local Quantum Drop** dialog:

Step	Action
1	Select the **Module** → **...** column. **Response:** The **I/O Module Selection** dialog is opened.
2	From the **Category** column, select the **<all>** option. **Response:** All modules are listed in the **Modules** column.
3	Select from the column **Modules**, the module**AMM-090-00**. Exit the dialog with **OK**. **Response:** The module is inserted in the I/O map.
4	In the **In Ref** and **Out Ref** columns, set the start references for the input and output modules. **Note:** Discrete Input References have the prefix 1 (e.g. 100001), Coil References have the prefix 0 (e.g. 000001), Input Register References have the prefix 3 (e.g. 300001) and Output Register References have the prefix 4 (e.g. 400001). **Response:** The end reference (column **In.End.** or **Out.End**) of the available address range is automatically entered.

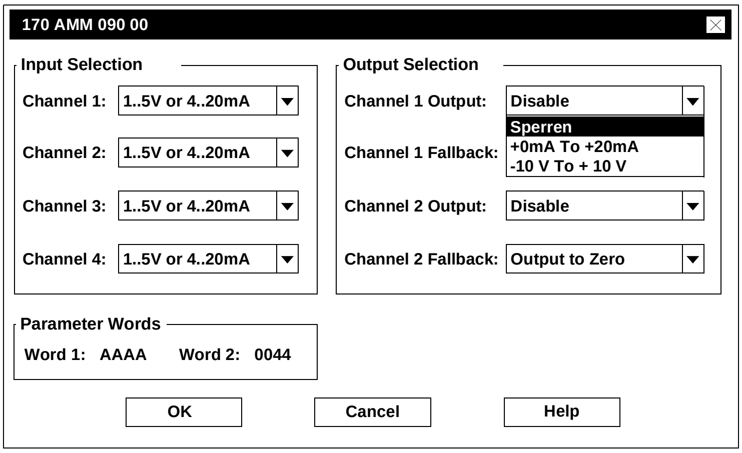
Dialog display

Following module mapping and I/O reference specification the dialog looks like this:



Set module parameters

To set parameters for the individual modules, proceed as follows in the **Local Momentum Drop** dialog:

Step	Action
1	Select the **Params** command button. **Response:** The **170-AMM-090-00** dialog is opened.
2	Select the signal states for the input and output channels from the list boxes and exit the dialog using **OK**. **Note:** Help with this can be obtained from the corresponding help text in the parameter dialog. Dialog display 

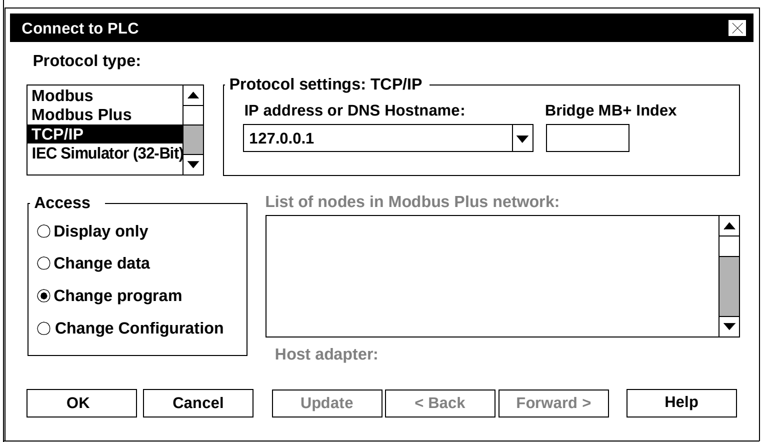
Create online connection

Introduction

This chapter describes how a link is created between the programming device and the Ethernet bus system.

Creating a link

For the link between the programming device and the Ethernet bus system use the Concept main menu **Online** and proceed as follows.

Step	Action
1	Select menu command**Connect...**. **Response:** The **Connect to PLC** dialog box opens.
2	From the list **Protocol type** select the link **TCP/IP**. **Response:** The zone **Protocol settings** alters for the TCP/IP settings.
3	In the text box IP address or DNS hostname enter the IP address of the Ethernet network card (PCI card). **Note:** Make sure that the address in Concept matches the address in Network settings of the operating system (See *Network configuration in Win 98, p. 897*). **Response:** An online link exists between the programming device and the Ethernet bus system, and all bus nodes are displayed in the list. Dialog display 
4	Exit the dialog using **OK**.

Convert Projects/DFBs/Macros

F

Converting projects/DFBs

At a Glance

The four main steps for converting projects/DFBs are as follows:

Step	Action
1	Exporting projects/DFBs/macros within the earlier version of Concept, see *Exporting project/DFB/macro (earlier version of Concept), p. 913*.
2	For information on installing the new version of Concept, see *Installing new versions of Concepts, p. 914*.
3	For information on importing projects/DFBs/macros, see *Importing project/DFB/macro, p. 914*.
4	For information on editing projects/DFBs/macros, see *Editing the project/DFB/macro, p. 914*.

Converting EFBs

CAUTION

Risk of losing data

If user-defined EFBs are being used in the project (EFBs which have been created manually), the current version of the EFB toolkit must be used to convert them (**File** → **Concept library…**). The Concept converter is not able to convert user-defined EFBs.

Failure to follow this precaution can result in injury or equipment damage.

Exporting project/DFB/ macro (earlier version of Concept)

The procedure for exporting projects/DFBs/macros is as follows:

	CAUTION
⚠	**Risk of losing data** The following steps must be performed in the **EARLIER** version of Concept. The new version of Concept may only be installed once all existing projects have been exported. **Failure to follow this precaution can result in injury or equipment damage.**

Step	Action
1	Start the Concept converter.
2	From **File** → **Export...** open the menu to select the export range.
3	Select the required export range: ● **Project with used DFBs:** All project information including the DFBs and data structures used within the project (derived data types) will be exported. ● **Project with all DFBs + macros:** All project information including all the DFBs and data structures (derived data types) will be exported. ● **Project without DFBs:** All project information including all data structures (derived data types), but excluding DFBs and macros will be exported. ● **Single DFB with used DFBs/single macro:** Only the selected DFB/macro will be exported. **Reaction:** The select export data dialog box will be opened.
4	Different file extensions must be selected depending on the element to be exported: ● **Exporting projects:** From the Format list select the extension .prj. ● **Exporting DFBs:** From the Format list select the extension .dfb. ● **Exporting macros:** From the Format list select the extension .mac.
5	Select the project / DFB / macro and confirm with **OK**. **Reaction:** The project/DFBs/macros/data structures (derived data types) will be contained in the current directory as an ASCII data file (.asc).
6	Quit the Concept converter with **File** → **Quit**.

Installing new versions of Concepts

	CAUTION
	Risk of losing data Only install the NEW version of Concept if you have performed the previous steps. **Failure to follow this precaution can result in injury or equipment damage.**

Follow the procedure described in the "Installation" chapter of the installation instructions.

Importing project/DFB/ macro

The procedure for importing projects/DFBs/macros is as follows:

Step	Action
1	Start the Concept converter.
2	From **File** → **Import...** open the select import projects/DFBs/macros dialog box.
3	Select the project/DFB/macro (data file format .asc) and confirm with **OK**. **Reaction:** The project/DFBs/macros/data structures will be contained in the current directory as Concept data files.
4	Quit the Concept converter with **File** → **Quit**.

Editing the project/DFB/ macro

Start the Concept/Concept DFB and edit the project/DFBs/macros/data structures in the usual way.

Concept ModConnect

G

At a Glance

Introduction

This chapter describes how to integrate third party modules into the Concept I/O map and how to remove it.

What's in this Chapter?

This chapter contains the following sections:

G.1 Introduction

Introduction

Overview

Information on hardware and I/O modules is stored in the Concept System Information Database (SysInfDb). This database is maintained and updated by Schneider and included with every Concept release.Nevertheless, Concept is able to support new I/O modules without having to wait for a new release. That's where the ModConnect Tool comes in - it takes a textual module description (MDC) and adds this information into the SysInfDb. This means that supplier of a new I/O module, who wants this module to be available in Concept, must also deliver an MDC file which describes the characteristics of this module.

Once installed, the I/O modules have the same functionality as existing Schneider Automation modules. This includes the ability to set module parameters and to display an online help.

For the installation of new modules, the third party module manufacturer has to supply a disk which contains a specific MDC file and the help information.

> **Note:** The MDC file is dependent on the version of Concept so if you upgrade your Concept version, make sure you get also an upgraded version of your previously used MDC files. You will have to reinstall them.

G.2 Integration of Third Party Modules

At a Glance

Introduction

This chapter describes the procedures which have to be used in Concept ModConnect in order to integrate third party modules into Concept or to remove it.

What's in this Section?

This section contains the following topics:

Integrating new Modules

Precondition

The specific MDC file for the new module has to be available.

Integrating new Modules

For integrating new modules, proceed the following steps:

Step	Action
1	For starting the application select **ModConnect Tool** in the Concept programm group. **Reaction:** Concept ModConnect displays its main window. If any Modules have been installed, a lis of installed modules is shown.
2	Copy the MDC file and the help file supplied with module to the Concept installation path.
3	Select **File** → **Open Installation File..**. **Reaction:** A dialog for selection the specific MDC file is opened.
4	Set the correct path to the MDC file and select it (e.g. SAMPLE.MDC). Confirm with **OK**. **Reaction:** The path including the name of the MDC file is now displayed in the **Select Module** dialog along with the defined modules.
5	Select the module you want to add and click **Add Module** or in the case of multiple entries click on the **Add All** button. You may additionally click the **Browse** button to return to the **Open file** dialog where you can select another .MDC for evaluation.
6	Click on the **Close** button to return to the main window. **Reaction:** The main window will now be displayed with the module information appearing in the **Imported Modules in Concept Database** window. By clicking on the added module (to select it) the module details are shown. With **Help** → **Help on Module** the help of the selected module can be displayed.
7	Select **File** → **Save Changes** to save the changes data base.
8	Select **File** → **Exit** for terminating Concept ModConnect. **Reaction:** The installed modules are now avaiable in the Concept I/O map (See *Use of third party module in Concept, p. 921*).

Upgrate of Concept

Note: The MDC-File is dependent on the version of Concept so if you upgrade your Concept version, make sure you get also an upgraded version of your previously used MDC files. You will have to reinstall them.

Removing Modules

Removing Modules

For removing modules, proceed the following steps:

Step	Action
1	For starting the application select **ModConnect Tool** in the Concept programm group. **Reaction:** Concept ModConnect displays its main window with a lis of the installed modules.
2	Select the module you want to remove and select **File** → **Remove selected Module**. **Reaction:** The **Confirm IOModule Removal** dialog is displayed.
3	Selecting **OK**, causes the removal of the module from Concept. **Reaction:** The module is no longer listed in the main window of Concept ModConnect or in the **I/O Module Selection** list box of Concept. **Note:** When removing modules. If the module has been used in existing Concept projects, the integrity of these projects will be compromised.
4	Select **File** → **Save Changes** to save the changes data base.
5	Select **File** → **Exit** for terminating Concept ModConnect. **Reaction:** The installed modules are now avaiable in the Concept I/O map (See *Use of third party module in Concept, p. 921*).

G.3 Use of third party module in Concept

Use of Third Party Modules in Concept

Precondition

The modules have to be installed according to the procedure *Integrating new Modules, p. 919*.

Insert module to I/O Map

To insert a module to the I/O map, proceed the following steps:

Step	Action
1	Start Concept.
2	Open the configurator with **Project** → **Configurator**.
3	Open the I/O map with **I/O map...** → **Edit...**.
4	Open the **I/O Module Selection** dialog by clicking on **...** at the **Module** column. **Reaction:** The third party modules appear in the **Other** column.
5	Select the module by clicking. **Reaction:** A short description appear at the top of the dialog. You may press the **Help on Module** button to display the module’s help file supplied by the vendor.
6	Click on **OK** (or doubleclick on the module) to insert the module the the I/O map. **Reaction:** The **I/O Module Selection** dialog is cloes and the selected module is inserted in the I/O map.
7	For entering the module’s parameters (if available), select the **Rack-Slot** column of the module and click on the **Params** button. **Reaction:** The parameter screen for the selected dialog is opened.
8	Set the parameters for the module and confirm with **OK**.
9	Enter the input and output references for the module.
10	Confirm the I/O map with **OK** and save the project with **File** → **Save project**.

Convertion of Modsoft Programs

H

At a Glance

Introduction

This information provides you with the necessary process required to change previously generated Modsoft derived Ladder Logic programs into the Concept environment.

What's in this Chapter?

This chapter contains the following topics:

Introduction

Overview

For the convertion of an existing Modsoft program to a valid Concept 984 Ladder Logic project the Mosdsoft Converter is used. The Modsoft Converter provides current Modsoft users with a migration path to the 984 Ladder Logic for Windows environment. The Modsoft Converter requires no previous knowledge of the Concept programming environment. The term project is synonymous with a Modsoft program.

Starting the Modsoft Converter

Windows 98, Windows 2000 or Windows NT allows you to run the program from the **Start** menu, by selecting **Modsoft Converter** in the Concept programm group.

The Modsoft .ENV File

For the convertion the Modsoft .ENV file is needed. The .ENV file contains all the file information pertaining to the Modsoft program.

The Modsoft .ENV file contains the following files:
- **.CFG** Configuration file
- **.PRG** Ladder Logic file
- **.PCM** Network comments
- **.PCT** Network comments
- **.ASC** ASCII file
- **.USL** User Loadables
- **.RFD** Reference presets set by the user in the Modsoft Reference Data Editor
- **.REF** Reference contents contained in the PLC, from an upload
- **.RSF** Reference symbols

The convert process requires the .CFG file to be present in the .ENV file. If it does not exist, an error dialog is displayed indicating that the .ENV file does not reference a .CFG file. All other files are optional.

By forcing you to enter the Modsoft *.ENV filename, some of the validation is avoided that would otherwise be required if you were allowed to enter a *.PRG and *.CFG name separately, i.e. Loadables (DX, User and EXE), state ram and builtin functions.

Incompatibilities

Due to differences in "address calculations in the configuration table" between Modsoft 2.6 and Concept 2.2 or later, the same Modsoft program loaded in a PLC and converted using the Modsoft Converter will cause a configuration miscompare in certain page zero locations. This will not affect the validity of the converted program.

Invalid PLC Types

If the Modsoft configuration file PLC type, is not legal for Concept, you are warned that the convert process will not continue. You then have to return to Modsoft and change the PLC type to one that is valid.

Handling of SY/MAX

SY/MAX programs converted to Modsoft file format will migrate to the Quantum PLC type. The Modsoft Convert utility can then bring the SY/MAX program into Concept.

Modsoft Version

The Convert utility handles Modsoft file format supported in revision 2.2 or greater.

Handling of SFC and Macros

Modsoft does allow the user to save a Ladder Logic program that consists of undefined elements, and Concept needs to resolve those elements. The Modsoft Ladder Logic program is converted without performing any validity checks against the Configuration. When the Modsoft *.prg file contains either SFC or Macros the convert process is aborted and an Error dialog is displayed informing you to return to Modsoft and use **Segment Status** → **Commands** → **Convert to File**. This process expands the Macros and translates the SFC elements.

Handling of I/O Map

Modsoft sets a default I/O map size of 512. Concept does not, but calculates the size as required. Uploading a Controller that has been downloaded with Modsoft will cause a miscompare. You are allowed to continue.

Handling of References

Modsoft can have two types of reference data or none at all. There exists online reference data information (RAM) if you have uploaded from the PLC. There are also references defined using the offline Reference Data Editor. When both types of data exist in the .env file, the convert utility first imports the online references then overlays the offline reference data.

How to Convert a Modsoft Program

Precondition

For converting a Modsoft program the Modsoft .ENV file (See *The Modsoft .ENV File, p. 924*) is necessary. The .ENV file contains all the file information pertaining to the Modsoft program. Once selected the conversion takes place and you are prompted to a **Save as** dialog.

How to Convert a Mosdsoft Programm

For converting a Modsoft programm, proceed the following steps:

Step	Action
1	Open the Modsoft Converter.
2	Select **File** → **Convert...**.
3	Select the drive and the directory, where to find the Modsoft .ENV file. (The file will be found in the Modsoft program directory, e.g. C:\Modsoft\Programs.)
4	Pick the file from the list.
5	Start the convertion with **Convert**. **Reaction:** The convertion is started. • A convert progress dialog is displayed after the validity checks on the *.ENV file are performed. The first line of the dialog indicates the section currently being converted and the second line indicates progress as it pertains to the whole convert process. • f any errors, such as **Out of memory**, **Out of disk space** or **File access errors**, occur during the convert process, an error dialog is displayed. • An operation completed error free results in the automatic display of the **Save as Concept project** dialog. The default name of the project, displayed in the Save project dialog, is the *.ENV filename prompt.
6	You can then change the project name and the directory in which Concept project will be saved. **Reaction:** If the project name selected already exists a confirmation dialog is displayed. **Note:** Saving the Modsoft converted program as a Concept project does not have to be done at this time, you can still save using the **File** → **Save project as** menu item.

Exceptions

Description

0x and 1x references in a Modsoft program are converted to a Located Variable with data type BOOL in Concept. This data type is compatible with the use of these references.

However, 3x and 4x are converted to integer.

Note: This straight conversion precludes both Modsoft bit defination and floating point types.

Example

If you have the following defined in **Modsoft**:

REF	BIT	SYMBOL	DESCRIPTOR
000001		located_0x_boolean	located 0x boolean descriptor
100001		located_1x_boolean	located 1x boolean descriptor
300001	/16	bit_16_of_3000001	16th bit of 300001 descriptor
400100		incoming_integer	incoming integer descriptor
400200		outgoing_interger	outgoing flt32 descriptor
400300	/ 1	bit_1_of_400300	bit 1 of 400300 descriptor

A conversion of the above to **Concept** using the Convert program yields:

Variable Name	Data Type	Address	Comment
located_0x_boolean	BOOL	000001	located 0x boolean descriptor
located_1x_boolean	BOOL	100001	located 1x boolean descriptor
bit_16_of_3000001	INT	300001	16th bit of 300001 descriptor
incoming_integer	INT	400100	incoming integer descriptor
outgoing_interger	INT	400200	outgoing flt32 descriptor
bit_1_of_400300	INT	400300	bit 1 of 400300 descriptor

Modsoft and 984 References

I

At a Glance

Introduction

This chapter contains the Modsoft and 984 References.

What's in this Chapter?

This chapter contains the following topics:

Modsoft Keys with Concept Equivalents

Keys

Note: When possible, the **Ctrl** key is used in place of the Modsoft **Alt** key.

Table of keys:

Funtion	**Modsoft 2.x Key**	**Concept Key**
Normally open contact	**'** or **"**	same
Coil	**(** or **[**	same
Normally closed contacts	**/** or ****	same
Horizontal short	**=**	same
Vertical short	**|**	same
Negative transitional contact	**Alt**+**N**	**N**
Positive transitional contact	**Alt**+**P**	**P**
Inserting a function block by name	**Alt**+**F**	**Ctrl**+**F**
Copy element(s)	**Alt**+**F3**	**Ctrl**+**C**
Delete element(s)	**Alt**+**F4** or **Del**	**Ctrl**+**X** or **Del**
Paste	**Alt**+**F5**	**Ctrl**+**V**
Offset references	**Alt**+**F6**	**Ctrl**+**H**
Search	**Alt**+**F7**	**F3**
Search next	**Alt**+**F8**	**F6** When online in direct mode, Concept uses a nonmodal dialog with accelerators for search previous and search next.
Network comments	**Alt**+**C**	**Ctrl**+**M**
Goto network	**Alt**+**G**	**Ctrl**+**G**
Insert network	**Alt**+**I**	**Ctrl**+**I**
Append network	**Alt**+**A**	**Ctrl**+**A**
Trace	**Alt**+**T**	**Ctrl**+**T**
Retrace	**Alt**+**B**	**Ctrl**+**B** or **Ctrl**+**T**
Dx zoom	**Alt**+**Z**	**Ctrl**+**D**
Goto node (1,1) of active network	**Home**	same
Goto node (7,11) of active network	**End**	same
Goto first network in current segment	**Ctrl**+**Home**	same

Funtion	Modsoft 2.x Key	Concept Key
Goto last network in current segment	**Ctrl**+**End**	same
Insert equation	**Ins**	**Ctrl**+**Q**
Append	-	**Ctrl**+**A**
Append equation	-	**Ctrl**+**U**
Delete current network	-	**Ctrl**+**K**
Copy to the clipboard	-	**Ctrl**+**C**
Undo	-	**Ctrl**+**Z**
Closing an mdi child window	-	**Ctrl**+**F4**
Switching to the next open mdi child window	-	**Ctrl**+**F6**

Status Line Values

These Concept keys change the status line display value of the currently selected reference:

A ASCII
H Hexidecimal
D Decimal (signed)
U Decimal (unsigned)
R Real
L Long (32 bit)
S Short (16 bit)

Modsoft Function Compatibility

Not Supported Features

The following Modsoft functions are **not** supported in Concept:
- Macros/macro programming
- SFC (use IEC SFC instead)
- Search of comments

User Interface Difference

Concept is an MS-Windows based application. Modsoft is a DOS based application. Concept uses MS-Windows user interface standards and practices. Functions of Concept with 984 Ladder editor are based on the pre-existing functions of Concept.

There are no exact similarities of specific user actions required to perform Concept tasks as compared to Modsoft tasks.

Constant Sweep

Concept has no off line selection to set the constant sweep mode. This mode is available from the **Online Control Panel**.

Once constant sweep has been set in the controller, you can upload the controller and save the project. The constant sweep settings will be retained in the project. If this project is downloaded, the constant sweep settings will be set.

Note: Any changes to the controller configuration cause the constant sweep settings to be reset, i.e, constant sweep is disabled whenever the controller configuration changes. Follow the steps above to reenable constant sweep.

How to Start the Constant Sweep

To set constant sweep before starting the controller, follow these steps:

Step	Action
1	Create your configuration and program logic, offline.
2	Download your program to the controller. When the dialog appears asking `Do you want to start the controller?`"click on the **No** button.
3	From the **Online** menu, choose **Online Control Panel**.
4	Set the constant sweep mode and sweep time.
5	Start the controller.

Presettings when using Modbus Plus for startup

J

Introduction

Overview

This chapter provides a brief description of the presettings when using Modbus Plus for first startup.

What's in this Chapter?

This chapter contains the following topics:

Installing the SA85/PCI85 with Windows 98/2000/XP

Introduction

A Modbus Plus connection can be made using the SA85 or PCI85 adapters.
The difference between the adapters is in the bus used:
- SA85 for ISA Bus
- PCI85 for PCI Bus

While the Modbus Node Address and Memory Based Address for the SA85 is set directly on the card with the DIP switches, the address for the PCI85 is made during the configuration in Windows.

SA85 Hardware settings

Carry out the following steps to configure the Hardware settings for the SA85:

Step	Action
1	Enter the Modbus node location (Modbus Plus Port Location) and the memory based address in SA85 (see documentation "IBM Host Based Devices").
2	Install the SA85 as described in the "IBM Host Based Devices" documentation.

PCI85 Installation

Install the PCI85 (416 NHM 300 30 or 416 NHM 300 32) as described in the "Modbus Plus PCI-85 Interface Adapter" 890 USE 162 00 documentation.

Driver installation

Install the Virtual MBX driver and then the MBX or Remote MBX driver.
See also:

- *Virtual MBX Driver for 16 bit application capability with Windows 98/2000/NT, p. 940*
- *MBX Driver for connection between ModConnect Host interface adapters and 32 bit applications with Windows 98/2000/NT, p. 941*
- *Remote MBX - Driver for Remote Operation, p. 942*

Configuration

Carry out the following steps to configure the adapter after installing the driver:

Step	Action
1	Open the Control Panel (**Start** → **Settings** → **Control Panel**).
2	**Windows XP:** Select the **Printer and other Hardware** icon.
3	**Windows XP:** Select the **System** icon.
4	Select the **Hardware** icon. **Result:** The hardware wizard is called.
5	Select the **Next** command button.
6	**Windows 98:** Select the option **Yes (Recommended)**. **Windows 2000/XP:** Select the option **Add/Troubleshoot a device**. Select the **Next** command button. **Result:** Hardware detection is started.
7	**Only for Windows 98:** Select the **Next** command button. **Result:** The hardware detection status is displayed.
8	**Only for Windows 98:** Select the **Next** command button. **Result:** All hardware types are displayed in a list.
9	Select the hardware type **MBX Devices for Modicon Networks**, and press the **Next** command button. **Result:** The database with driver information is created.
10	Select the **SA85-000** adapter or **PCI85-000** and press the **Next** command button. **Result:** A memory range is automatically defined.
11	Select the **Next** command button. **Result:** The automatically assigned device number and request mode (20 ms) is displayed.
12	Select the **Next** command button. **Result:** The software for the new hardware components is installed.
13	Select the **Next** command button. **Result:** You are asked to shutdown the computer.
14	Press the **No** command button. **Result:** The adapter is configured with the default settings.

Win 98: Edit configuration

Carry out the following steps to edit the configuration using Windows 98 after the first configuration:

Step	Action
1	Open the Control Panel (**Start** → **Settings** → **Control Panel**).
2	Select the **System** icon. **Result:** The **System Properties** window is opened.
3	Select the **Device Manager** tab.
4	Select the **SA85-000** adapter or **PCI85-000** and press the **Properties** command button. **Result:** The **SA85-000/PCI85-000 Adapter Properties** window is opened.
5	Select the **Device Settings** tab.
6	Make the changes as required. (See also the Help file LMBX9X on the driver CD.)
7	Select the **Resources** tab to change the memory area.
8	Use the **OK** command button to exit the window. **Result:** The changes are accepted by the system.

Win 2000/XP: Edit configuration

Carry out the following steps to edit the configuration using Windows 2000/XP after the first configuration:

Step	Action
1	Open the Control Panel (**Start** → **Settings** → **Control Panel**).
2	**Windows XP:** Select the **Printer and other Hardware** icon.
3	Select the **System** icon. **Result:** The **System Properties** window is opened.
4	Select the **Hardware** tab.
5	Select the **Device Manager...** command button. **Result:** The **Device Manager** window is opened.
6	Select the **Network adapter** → **SA85-000** or **PCI85-000**.
7	Select the **Properties** command button. **Result:** The **SA85-000PCI85-000 Adapter Properties** window is opened.
8	Select the **Device Settings** tab.
9	Make the changes as required. (See also the Help file LMBX9X on the driver CD.)
10	Select the **Resources** tab to change the memory area.
11	Use the **OK** command button to exit the window. **Result:** The changes are accepted by the system.

Peer Cop functions

Several parameter settings must be made to enable Peer Cop communication via the adapter. The Peer Cop function is disabled by default, and should only be enabled if your applications require Peer Cop communication.
To enable and set parameters for Peer Cop communication, start with the first steps as with "Edit Configuration". In the **SA85-000/PCI85-000 Adapter Properties** window, select the **Peer Cop** tab and make your settings as desired.

Installing the SA85/PCI85 in Windows NT

Introduction

A Modbus Plus connection can be made using the SA85 or PCI85 adapters.
The difference between the adapters is in the bus used:
- SA85 for ISA Bus
- PCI85 for PCI Bus

While the Modbus Node Address and Memory Based Address for the SA85 is set directly on the card with the DIP switches, the address for the PCI85 is made during the configuration in Windows.

SA85 Hardware Settings

Carry out the following steps to set the SA85 hardware settings:

Step	Action
1	Set the Modbus node address (Modbus Plus Port Address) and the memory based address on the SA85 (see documentation "IBM Host Based Devices").
2	Install the SA85 as described in the "IBM Host Based Devices" documentation.

PCI85 Installation

Install the PCI85 (416 NHM 300 30 or 416 NHM 300 32) as described in the "Modbus Plus PCI-85 Interface Adapter" 890 USE 162 00 documentation.

Installing drivers

Install the Virtual MBX driver and then the MBX or Remote MBX driver.
Also see:
- *Virtual MBX Driver for 16 bit application capability with Windows 98/2000/NT, p. 940*
- *MBX Driver for connection between ModConnect Host interface adapters and 32 bit applications with Windows 98/2000/NT, p. 941*
- *Remote MBX - Driver for Remote Operation, p. 942*

Configuration

Carry out the following steps to configure the adapter after installing the driver:

Step	Action
1	In the start menu, open the folder **WinConX/MBXDriver** (**Start** → **Program** → **WinConX**).
2	Double-click on the **MBX Driver Configuration** icon. **Result:** The dialog box **MBX Driver configuration** is opened.
3	In the **Device Configuration** register, click on the command button **New**. **Result:** A list box will appear in the **Device type** column.
4	Select the option **SA85** or **PCI85** from the list. **Result:** The dialog box **SA85 configuration** is opened.
5	Make the following settings. (also see Help file LMBX9X on the driver CD.) **Note:** With the PCI85 you enter the Modbus Node address in the **Node** list box.
6	Exit the dialog box by clicking **Close**. **Result:** The settings are accepted by the system.

Edit configuration

Carry out the following steps to edit the configuration after the first configuration:

Step	Action
1	In the start menu, open the folder **WinConX/MBXDriver** (**Start** → **Program** → **WinConX**).
2	Double-click on the **MBX Driver configuration** icon. **Result:** The dialog box **MBX Driver configuration** is opened.
3	Select SA85 from the **Device configuration** register.
4	Click on the command button **Edit**. **Result:** The **SA85 configuration** dialog box is opened.
5	Make the following changes. (also see Help file LMBX9X on the driver CD.)
6	Exit the dialog box by clicking **Close**. **Result:** The settings are accepted by the system.

Peer Cop functionality

Several parameter settings must be made to enable Peer Cop communication via the adapter. The Peer Cop function is deactivated as standard, and should only be enabled if your application requires Peer Cop communication.
To enable and set parameters for Peer Cop communication, start with the first steps as with "Edit SA85 Configuration". In the dialog box **SA85 configuration**, select the **Peer Cop** register and make your settings.

Installing the Modbus Plus Driver in Windows 98/2000/NT

Introduction

In order to use the Modbus Plus communication, you must first install the CyberLogic MBX driver for Windows 98/2000/NT version >=4.20 (+ Service Release 1 for Windows 2000)
The following drivers are available on the CD "MBX Driver Suite v4.20":

Driver	Operating system
MBX Driver *MBX Driver for connection between ModConnect Host interface adapters and 32 bit applications with Windows 98/2000/NT, p. 941*	Windows 98/2000/NT
Virtual MBX Driver *Virtual MBX Driver for 16 bit application capability with Windows 98/2000/NT, p. 940*	Windows 98/2000/NT
Remote MBX Driver *Remote MBX - Driver for Remote Operation, p. 942*	Windows 98/2000/NT
Ethernet MBX Driver *Ethernet MBX - Driver for Modbus Plus Function via TCP/IP, p. 943*	Windows NT

Installation

Carry out the following steps to install the Modbus Plus driver:

Step	Action
1	Start Windows.
2	Insert the CD "MBX Driver Suite ver. 4.20".
3	Select the **Start** → **Execute** command.
4	Enter the CD drive and **:\SETUP** in the command line.
5	Confirm with **OK**.
6	Follow the onscreen instructions. **Response:** After installation the WinConX program with all installed drivers is created in the Start Menu.

Configuration

Configuration occurs automatically after installing the driver. To make changes to the configuration, open the dialog to be edited from the **WinConX** → **xxx MBX Driver** Start Menu, by double clicking on the **xxx MBX Configuration Editor** symbol.

Virtual MBX Driver for 16 bit application capability with Windows 98/2000/NT

Introduction

Installing the Virtual MBX driver guarantees the run capability of all 16 bit DOS or Windows 3.x NETLIB/NetBIOS compatible applications in their original binary form in Windows 98/2000/NT.

Note: A detailed description of this driver can be found in the VMBX9X or VMBXNT Help file on the "MBX Driver Suite ver4.20" CD.

Preconditions

In order for the Virtual MBX driver to function correctly, additional drivers must be installed.
The following additional drivers can be installed to enhance the Virtual MBX driver's run capability:

Driver	Operating system	Application
MBX	Windows 98/ 2000/NT	Driver for Modbus Plus Host interface adapter
Remote MBX	Windows 98/ 2000/NT	Driver for accessing remote nodes on the Modbus Plus and Ethernet network
Ethernet MBX	Windows NT	Driver for Modbus Plus Emulation via TCP/IP

Installation

The virtual MBX driver software for Windows 98/2000 and Windows NT is included along with other drivers, on the CD "MBX Driver Suite ver4.20".
Installation is done by Autorun when the CD is inserted or can be started manually (CD drive:\SETUP.EXE). Select the driver to be installed from the main menu. You will then be taken through the installation step by step. The driver is then configured.

Configuration

Note: To guarantee a connection to Concept (= 16-Bit-Application), in the **Virtual MBX Driver Configuration** → **16-bit Windows Applications** dialog, check the **Support 16-bit Windows Applications** checkbox.

MBX Driver for connection between ModConnect Host interface adapters and 32 bit applications with Windows 98/2000/NT

Introduction

The installation of the MBX driver guarantees the connection between the MODConnect Host interface adapter and 32 bit applications with Windows 98/2000/NT. This driver also supports the program interfaces MBXAPI and NETLIB. This means that practically all Modbus Plus compatible software programs can be operated via Modbus, Modbus Plus and Ethernet networks, without having to make changes. This also includes 32 bit Windows 98/2000/NT applications and 16 bit old DOS/Windows applications.

Note: A detailed description of the driver is included on the CD "MBX Driver Suite ver4.20" in the Help file LMBX9X or LMBXNT.

Hardware support

The MBX driver operates either in Interrupt or Polled mode.
It supports the following ModConnect Host interface adapter:
- ISA
- EISA
- MCA
- PC card (PCMCIA)

Remote connection

The MBX driver includes the MBX Remote Server. This enables remote nodes to access local MBX devices (including the Host interface adapter) via any Windows 98/2000/NT compatible network. Also see *Remote MBX - Driver for Remote Operation, p. 942*.

Installation

The MBX driver software for Windows 98/2000/NT is included along with other drivers, on the CD "MBX Driver Suite ver4.20".
The installation is carried out by Autorun when the CD is inserted or can be manually started(CD drive:\SETUP.EXE). Select the driver to be installed from the main menu. You will then be taken through the installation step by step. The driver is then configured.

Remote MBX - Driver for Remote Operation

Introduction

The installation of the remote MBX driver allows remote connection of applications operated on remote station client nodes. Remote station access of the Modbus Plus network takes place using a standard LAN (Local Area Network).
This driver also unites applications that support the program interfaces MBXAPI and NETLIB.

Note: A detailed description of the driver is included on the CD "MBX Driver Suite ver4.20" in the Help file RMBX9X or RMBXNT.

Preconditions

This connection is only made if your programming device is a node on the MBX Remote Server. Also install one of these drivers because the MBX and Ethernet MBX drivers include the MBX Remote Server.

Installation

The remote MBX driver software for Windows 98/2000/NT is included along with other drivers, on the CD "MBX Driver Suite ver4.20".
The installation is carried out by Autorun when the CD is inserted or can be manually started(CD drive:\SETUP.EXE). Select the driver to be installed from the main menu. You will then be taken through the installation step by step. The driver is then configured.

Configuration

The configuration of the remote MBX driver is presently the same as the configuration of the other MBX drivers. The remote MBX driver is operated as a remote client node, which does not require a physical host interface adapter. Therefore the driver configuration also includes the creation of logical devices (MBX Remote Client), which refer to the physical devices found on the server node.

Ethernet MBX - Driver for Modbus Plus Function via TCP/IP

Introduction

The installation can only be carried out in Windows NT.
When the Ethernet MBX driver is installed, Modbus Plus function is emulated via TCP/IP. This driver also supports the program interfaces MBXAPI and NETLIB. This means that practically all Modbus Plus compatible software programs immediately have access to TCP/IP based communication without having to make changes. This also includes 32 bit Windows 98/2000/NT applications and 16 bit old DOS/Windows applications.

Note: A detailed description of the driver is included on the CD "MBX Driver Suite ver4.20" in the Help file EMBXNT.

Winsock API

When using Winsock API, the Ethernet MBX driver can solve certain critical problems created by the Winsock interface.
For example: TCP Port 502 can only receive one process with incoming messages. If several applications attempt to receive unexpected messages, a conflict occurs. The Ethernet MBX driver eliminated this problem by acting as global dispatcher for these messages. When using the slave path, Concept in Modbus Plus determines that several (up to 256) applications refer to these unexpected messages and execute them simultaneously.

Advantage of using the driver

The most important advantages when using the driver via the Winsock API are:
- Changes are no longer needed for existing NETLIP/NetBIOS/MBXAPI compatible applications. End user and developer software investments are completely secured.
- Consistent management and dispatching of unexpected messages, which prevents overlaps between various products on the same system.
- Complete functionality of TCP/IP communication, while protecting existing NETLIP/NetBIOS/MBXAPI standards.
  For example: Ethernet TCP/IP communication requires a identifier address in the form of an IP address, and a message contains an identifier index byte. The Ethernet MBX driver protects this functionality.
- Working with TCP/IP communication is an advantage for software developers not experienced with the complicated Winsock API.
- A single program model for software developers handles communication in Modbus, Modbus Plus and Ethernet TCP/IP networks.
- Increased compatibility with various products. Winsock API is more oriented towards developer executed, TCP/IP strategies in a slightly different manner and can create compatibility problems in various products.
- Compatible with all MBX products. How e.g. the Virtual MBX driver for use of old 16-bit DOS/Windows applications and the MBX driver which dispatches messages between Ethernet, Modbus, Modbus Plus and remote MBX nodes.

Remote connection

The Ethernet MBX driver includes the MBX Remote Server. This enables remote nodes to access local MBX devices (including Ethernet MBX devices) via any Windows compatible network. The remote client can be a Windows 98/2000/NT node with the remote MBX driver installed. Also see *Remote MBX - Driver for Remote Operation, p. 942*.

Installation

The Ethernet MBX driver software for Windows NT is included along with other drivers, on the CD "MBX Driver Suite ver4.20".
The installation is carried out by Autorun when the CD is inserted or can be manually started(CD drive:\SETUP.EXE). Select the driver to be installed from the main menu. You will then be taken through the installation step by step. The driver is then configured.

Establishing the hardware connection

Introduction

Note: Please refer to the "Modbus Protocol Reference Guide" for a detailed description of the hardware setup.

Procedure

To establish the hardware connection, do the following:

Step	Action
1	Set a unique Modbus node address for the CPU using the rotary switch on the back of the module.
2	Note the Modbus node address set on the CPU’s sliding cover.
3	Connect the CPU to the SA85 interface with a Modbus Plus cable. **Result:** The flash interval on the CPU "Modbus +" display changes from 3 flashes per second with a pause to 6 without a pause.

Presettings when using Modbus for startup

K

Introduction

Overview

The chapter provides a brief description of the presettings when using Modbus for startup.

What's in this Chapter?

This chapter contains the following topics:

Interface Settings in Windows 98/2000/XP

Win 98: Interface settings

Carry out the following steps to configure the interface in Windows 95/98/2000:

Step	Action
1	Select the **My Computer** icon. **Result:** All available objects are displayed.
2	Select the **Control Panel** icon. **Result:** All available objects are displayed.
3	Select the **System** icon. **Result:** The **System Properties** dialog box is opened.
4	Select the **Device Manager** tab.
5	Select **Ports (COM and LPT)**. **Result:** The branches**Communications Port (COMx)** and **Printer Port (LPTx)** are displayed.
6	Select **Communications Port (COMx)**. **Result:** The **Communications Port (COMx) Properties** dialog box is opened.
7	Select the **Port Settings** tab.
8	Select the **Advanced...** command button. **Result:**The **Advanced Settings** dialog box is opened.
9	Check the **Use FIFO buffers** check box. **Note:** Using the FIFO(First In First Out) buffer requires a serial port with 16550 compatible UART (Universal Asynchronous Receiver Transmitter).
10	Use the slider to modify the receive and send buffer by setting both buffers to the maximum size.
11	Close all dialog boxes using the **OK** command button.

Win 2000/XP: Interface settings

Carry out the following steps to configure the interface in Windows 2000/XP:

Step	Action
1	Select the **My Computer** icon. **Result:** All available objects are displayed.
2	Select the **Control Panel** icon. **Result:** All available objects are displayed.
3	**Only with Win XP:** Select the **Printer and other Hardware** icon.
4	Select the **System** icon. **Result:**The **System Properties** dialog box is opened.
5	Select the **Hardware** tab.
6	Select the **Device Manager...** command button. **Result:** The **Device Manager** window is opened.
7	Select **Ports (COM and LPT)**. **Result:** The branches**ECP Printer Port (LPT1)** and **Communications Port (COMx)** are displayed.
8	Select **Communications Port (COMx)**. **Result:** The **Communications Port (COMx) Properties** dialog box is opened.
9	Select the **Port Settings** tab.
10	Select the **Advanced...** command button. **Result:** The **Advanced settings for COMx** dialog box is opened.
11	Check the **Use FIFO buffers** check box. **Note:** Using the FIFO(First In First Out) buffer requires a serial port with 16550 compatible UART (Universal Asynchronous Receiver Transmitter).
12	Use the slider to modify the receive and send buffer by setting both buffers to the maximum size.
13	Close all dialog boxes using the **OK** command button.

Interface Settings in Windows NT

Interface setting

Carry out the following steps to set the interface in Windows NT:

Step	Action
1	Double-click on the **My Computer** icon. **Response:** All available objects are displayed.
2	Double-click on the **Control Panel** icon. **Response:** All available objects are displayed.
3	Double-click on the **Connections** icon. **Response:** The **Connections** dialog box is opened.
4	Select the connection to be set in the list box and click on the command button **Settings...**. **Response:** The **COMx Settings** dialog box is opened.
5	Click on the command button **Extended...**. **Response:** The **Advanced Settings for COMx** dialog box is opened.
6	Activate the check box **FIFO activated**. **Note:** Using the FIFO(First In First Out) buffer requires a serial port with 16550 compatible UART (Universal Asynchronous Receiver Transmitter).
7	Close all dialogs with **OK**.

Setting up the hardware connection

Introduction

Note: Please refer to the "Modbus Protocol Reference Guide" for a detailed description of the hardware setup.

Procedure

To establish the hardware connection, do the following:

Step	Action
1	Set a unique Modbus node address for the CPU using the rotary switch on the back of the module.
2	Note the Modbus node address set on the CPU’s sliding cover.
3	Connect the Modbus interface CPU to the PC serial COM interface with a Modbus cable.

Transfer problems

Introduction

Communication errors can occur when loading the EXEC file. Communication, made via the COM interface with Windows, depends on several factors. These factors include the programming device clock speed, the communication software and the other programs (or applications) that are used in the system.

Check list for transfer problems

Refer to the following check list if transfer problems occur:

Step	Action
1	Check that no other applications are running in the background. Another application running in the background can mean that the active communication application in the foreground cannot receive information fast enough.
2	Check that the programming device is running at the highest possible clock speed. Some programming devices can prolong the lifetime of the buffer battery with lower speeds. Look in the documentation for you computer.
3	Use a serial connector with a 16550A Universal Asynchronous Receiver Transmitter (UART). Windows uses the buffering capability of these connections so that Windows data transfer applications can reach higher speeds even on slower computers.

RTU transfer problems

If sporadic errors occur during data transfer, transfer cannot be carried out successfully with RTU mode. If this is the case, select ASCII mode. (See Quantum/ Compact/Momentum/Atrium first startup.)

Startup when using Modbus with the EXECLoader

L

Introduction

Overview

This chapter describes loading executive data (EXEC) onto the PLC with the EXECLoader program.

What's in this Chapter?

This chapter contains the following topics:

Quantum first startup with EXECLoader

Introduction

This section describes the first startup of Quantum when used with Modbus.
The first startup is subdivided into 5 main sections:

- *Start EXECLoader, p. 954*
- *Define Modbus interface, p. 954*
- *Protocol settings, p. 955*
- *Select EXEC file, p. 956*
- *Load EXEC file, p. 956*

Start EXECLoader

The procedure for launching EXECLoaders is as follows:

Step	Action
1	Open the Concept start menu. **Response:** All installed Concept programs are displayed as symbols.
2	Click on the symbol with the identifier **EXECLoader**. **Response:** The EXECLoader program is started.
3	Click on the command **Next**, as soon as you have read the information on the page. **Response:** The dialog **Communication Protocol** is opened.

Define Modbus interface

Carry out the following steps to set the Modbus interface:

Step	Action
1	Select the Option button **Modbus RTU (RS232)** for the RTU transfer mode. Select the Option button **Modbus ASCII (RS232)** for the ASCII transfer mode. **Note:** Data transfer can only take place if you have configured the same transfer mode (RTU or ASCII) on the CPU (using a button on the front of the module).
2	Click on the command button **Next**. **Response:** The dialog **Modbus Target** → **RTU/ASCII mode** is opened.
3	Use the command button **COM Port Settings...** to open the dialog **COM Properties**.
4	Use the list field **Connect using:** to select the programming cable interface on the PC (default setting is **COM1**).
5	Use the list field **Bits per second:** to select the Baudrate (default is **9600**).
6	Use the list field **Parity:** to select the parity (default is **EVEN**).
7	Use the list field **Stop Bits** to select the Stop bits (default is **1**).
8	Click on the command button **OK**. **Response:** The dialog is closed and you return to the dialog **Modbus Target** → **RTU/ASCII Mode**.

Protocol settings Carry out the following steps to set the Modbus protocol:

Step	Action
1	Click on the command button **Scan**. **Response:** The nodes on the Modbus network are read and displayed graphically in the left window. A green point in the graphic indicates that the CPU is in RUN mode. To stop the CPU continue as described in step 3.
2	Double-click on the read network node in the graphical display. **Response:**The Modbus address of the node is automatically entered in the textfeld **Modbus Address**.
3	Click the right mouse button in the left window. **Response:** A context menu with individual PLC commands is opened.
4	If the "Run" display is lit on the CPU, stop the program using the command **Stop PLC**. **Response:** A message window appears where you can click **OK** to confirm stopping the CPU.
5	Click the command button **OK**, to confirm stopping the CPU. **Response:** You return to the dialog **Modbus Target** → **RTU mode**, and the green point disappears from the graph.
6	Activate the check box **Bridge**, if the connection to the Modbus node should be made via a Modbus Plus network using a Modbus bridge.
7	Press the appropriate Option button for your system (**PLC**, **Direct Device**, **Local Head**, **Remote I/O Drop**).
8	Click on the command button **Next**. **Response:** The **Operation** dialog is opened.

Select EXEC file

Carry out the following steps to select the EXEC file:

Step	Action
1	Press the Option button **Transfer EXEC to Device**.
2	Click on the command button **Browse...**. **Response:** The Concept directory is opened in a standard window.
3	Double-click on the DAT directory. **Response:** All available*.BIN files are displayed.
4	Click on the *.BIN file that corresponds to your CPU and the desired programming language. See the table *Loading Firmware for Quantum PLC Types, p. 1024*. **Response:** The selected *.BIN file is displayed in **File name:** text field.
5	Click on the command button **Open**. **Response:** You return to the dialog **Operation**, and the path to the selected *.BIN file is displayed in the **Filename** text field.
6	Click on the command button **Next**. **Response:** The dialog **File and Device Info** is opened. Information is provided here about the selected *.BIN file and also about the PLC.
7	Click on the command button **Next**. **Response:** The **Summary** dialog is opened. This gives you an overview of the settings made for you to check.

Load EXEC file

Carry out the following steps to load the EXEC file in the CPU flash RAM:

Step	Action
1	Click on the command button **Transfer**. **Response:** A message box appears warning you that all data available on the PLC will be lost, and the configuration and program must be reloaded on the PLC.
2	Click on the command button **Yes**, to continue the transfer. **Response:** The **Progress** dialog is opened. This gives information about the progress of the transfer in a progress bar and text.
3	Click **Close** once the transfer is complete. **Response:** The dialog is closed, and you return to the dialog **Summary**.
4	Click on the command button **Close**, to close the EXECLoader.

CPU display during transfer

During transfer the CPU display is as follows:

LED	Response
Ready	lit
Run	slow flashing
Modbus	lit (with some interruptions)
Modbus +	3x flashes with interruptions

CPU display after transfer

After transfer the CPU display is as follows:

LED	Response
Ready	lit
Run	not lit
Modbus	not lit
Modbus +	3x flashes with interruptions

Note: The three flash sequence Modbus + display idicates that no communication is present on the bus. This is displayed on Modbus by a non-lit Modbus display. The Modbus display is lit again once connection is made with Concept.

Creating the software connection

Carry out the steps given in chapter *Creating a Project, p. 47*.

Compact first startup with EXECLoader

Introduction

This section describes the first startup of Compact when used with Modbus.
The first startup is subdivided into 5 main sections:

- *Start EXECLoader, p. 958*
- *Define Modbus interface, p. 958*
- *Protocol settings, p. 959*
- *Select EXEC file, p. 960*
- *Load EXEC file, p. 960*

Start EXECLoader

The procedure for launching EXECLoaders is as follows:

Step	Action
1	Open the Concept start menu. **Response:** All installed Concept programs are displayed as symbols.
2	Click on the symbol with the identifier **EXECLoader**. **Response:** The EXECLoader program is started.
3	Click on the command **Next**, as soon as you have read the information on the page. **Response:** The dialog **Communication Protocol** is opened.

Define Modbus interface

Carry out the following steps to set the Modbus interface:

Step	Action
1	Select the Option button **Modbus RTU (RS232)** for the RTU transfer mode. Select the Option button **Modbus ASCII (RS232)** for the ASCII transfer mode. **Note:** Data transfer can only take place if you have configured the same transfer mode (RTU or ASCII) on the CPU (using a button on the front of the module).
2	Click on the command button **Next**. **Response:** The dialog **Modbus Target** → **RTU/ASCII mode** is opened.
3	Use the command button **COM Port Settings...** to open the dialog **COM Properties**.
4	Use the list field **Connect using:** to select the programming cable interface on the PC (default setting is **COM1**).
5	Use the list field **Bits per second:** to select the Baudrate (default is **9600**).
6	Use the list field **Parity:** to select the parity (default is **EVEN**).
7	Use the list field **Stop Bits** to select the Stop bits (default is **1**).
8	Click on the command button **OK**. **Response:** The dialog is closed and you return to the dialog **Modbus Target** → **RTU/ASCII Mode**.

Protocol settings Carry out the following steps to set the Modbus protocol:

Step	Action
1	Click on the command button **Scan**. **Response:** The nodes on the Modbus network are read and displayed graphically in the left window. A green point in the graphic indicates that the CPU is in RUN mode. To stop the CPU continue as described in step 3.
2	Double-click on the read network node in the graphical display. **Response:**The Modbus address of the node is automatically entered in the textfeld **Modbus Address**.
3	Click the right mouse button in the left window. **Response:** A context menu with individual PLC commands is opened.
4	If the "Run" display is lit on the CPU, stop the program using the command **Stop PLC**. **Response:** A message window appears where you can click **OK** to confirm stopping the CPU.
5	Click the command button **OK**, to confirm stopping the CPU. **Response:** You return to the dialog **Modbus Target** → **RTU mode**, and the green point disappears from the graph.
6	Activate the check box **Bridge**, if the connection to the Modbus node should be made via a Modbus Plus network using a Modbus bridge.
7	Press the appropriate Option button for your system (**PLC**, **Direct Device**, **Local Head**, **Remote I/O Drop**).
8	Click on the command button **Next**. **Response:** The **Operation** dialog is opened.

Select EXEC file

Carry out the following steps to select the EXEC file:

Step	Action
1	Press the Option button **Transfer EXEC to Device**.
2	Click on the command button **Browse...**. **Response:** The Concept directory is opened in a standard window.
3	Double-click on the DAT directory. **Response:** All available*.BIN files are displayed.
4	Click on the *.BIN file CTSX201D. **Response:** The selected *.BIN file is displayed in **File name:** text field.
5	Click on the command button **Open**. **Response:** You return to the dialog **Operation**, and the path to the selected *.BIN file is displayed in the **Filename** text field.
6	Click on the command button **Next**. **Response:** The dialog **File and Device Info** is opened. Information is provided here about the selected *.BIN file and also about the PLC.
7	Click on the command button **Next**. **Response:** The **Summary** dialog is opened. This gives you an overview of the settings made for you to check.

Load EXEC file

Carry out the following steps to load the EXEC file in the CPU flash RAM:

Step	Action
1	Click on the command button **Transfer**. **Response:** A message box appears warning you that all data available on the PLC will be lost, and the configuration and program must be reloaded on the PLC.
2	Click on the command button **Yes**, to continue the transfer. **Response:** The **Progress** dialog is opened. This gives information about the progress of the transfer in a progress bar and text.
3	Click **Close** once the transfer is complete. **Response:** The dialog is closed, and you return to the dialog **Summary**.
4	Click on the command button **Close**, to close the EXECLoader.

CPU display during transfer

During transfer the CPU display is as follows:

LED	Response
Ready	lit
Run	slow flashing
Modbus	lit (with some interruptions)
Modbus +	3x flashes with interruptions

CPU display after transfer

After transfer the CPU display is as follows:

LED	Response
Ready	lit
Run	not lit
Modbus	not lit
Modbus +	3x flashes with interruptions

Note: The three flash sequence Modbus + display idicates that no communication is present on the bus. This is displayed on Modbus by a non-lit Modbus display. The Modbus display is lit again once connection is made with Concept.

Creating the software connection

Carry out the steps given in chapter *Creating a Project, p. 47*.

Momentum first startup for IEC with EXECLoader

Introduction

This section describes the first startup of Momentum for IEC when used with Modbus.
The first startup is subdivided into 5 main sections:

Start EXECLoader

The procedure for launching EXECLoaders is as follows:

Step	Action
1	Open the Concept start menu. **Response:** All installed Concept programs are displayed as symbols.
2	Click on the symbol with the identifier **EXECLoader**. **Response:** The EXECLoader program is started.
3	Click on the command **Next**, as soon as you have read the information on the page. **Response:** The dialog **Communication Protocol** is opened.

Define Modbus interface

Carry out the following steps to set the Modbus interface:

Step	Action
1	Select the Option button **Modbus RTU (RS232)** for the RTU transfer mode. Select the Option button **Modbus ASCII (RS232)** for the ASCII transfer mode. **Note:** Data transfer can only take place if you have configured the same transfer mode (RTU or ASCII) on the CPU (using a button on the front of the module).
2	Click on the command button **Next**. **Response:** The dialog **Modbus Target** → **RTU/ASCII mode** is opened.
3	Use the command button **COM Port Settings...** to open the dialog **COM Properties**.
4	Use the list field **Connect using:** to select the programming cable interface on the PC (default setting is **COM1**).
5	Use the list field **Bits per second:** to select the Baudrate (default is **9600**).
6	Use the list field **Parity:** to select the parity (default is **EVEN**).
7	Use the list field **Stop Bits** to select the Stop bits (default is **1**).
8	Click on the command button **OK**. **Response:** The dialog is closed and you return to the dialog **Modbus Target** → **RTU/ASCII Mode**.

Protocol settings

Carry out the following steps to set the Modbus protocol:

Step	Action
1	Click on the command button **Scan**. **Response:** The nodes on the Modbus network are read and displayed graphically in the left window. A green point in the graphic indicates that the CPU is in RUN mode. To stop the CPU continue as described in step 3.
2	Double-click on the read network node in the graphical display. **Response:**The Modbus address of the node is automatically entered in the textfeld **Modbus Address**.
3	Click the right mouse button in the left window. **Response:** A context menu with individual PLC commands is opened.
4	If the "Run" display is lit on the CPU, stop the program using the command **Stop PLC**. **Response:** A message window appears where you can click **OK** to confirm stopping the CPU.
5	Click the command button **OK**, to confirm stopping the CPU. **Response:** You return to the dialog **Modbus Target** → **RTU mode**, and the green point disappears from the graph.
6	Activate the check box **Bridge**, if the connection to the Modbus node should be made via a Modbus Plus network using a Modbus bridge.
7	Press the appropriate Option button for your system (**PLC**, **Direct Device**, **Local Head**, **Remote I/O Drop**).
8	Click on the command button **Next**. **Response:** The **Operation** dialog is opened.

Select EXEC file

Carry out the following steps to select the EXEC file:

Step	Action
1	Press the Option button **Transfer EXEC to Device**.
2	Click on the command button **Browse...**. **Response:** The Concept directory is opened in a standard window.
3	Double-click on the DAT directory. **Response:** All available*.BIN files are displayed.
4	Click on the *.BIN file that corresponds to your CPU and the desired programming language. See the table *Loading Firmware for Momentum PLC Types, p. 1025*. **Response:** The selected *.BIN file is displayed in **File name:** text field.
5	Click on the command button **Open**. **Response:** You return to the dialog **Operation**, and the path to the selected *.BIN file is displayed in the **Filename** text field.
6	Click on the command button **Next**. **Response:** The dialog **File and Device Info** is opened. Information is provided here about the selected *.BIN file and also about the PLC.
7	Click on the command button **Next**. **Response:** The **Summary** dialog is opened. This gives you an overview of the settings made for you to check.

Load EXEC file

Carry out the following steps to load the EXEC file in the CPU flash RAM:

Step	Action
1	Click on the command button **Transfer**. **Response:** A message box appears warning you that all data available on the PLC will be lost, and the configuration and program must be reloaded on the PLC.
2	Click on the command button **Yes**, to continue the transfer. **Response:** The **Progress** dialog is opened. This gives information about the progress of the transfer in a progress bar and text.
3	Click **Close** once the transfer is complete. **Response:** The dialog is closed, and you return to the dialog **Summary**.
4	Click on the command button **Close**, to close the EXECLoader.

CPU display during transfer

During transfer the CPU display is as follows:

LED	Response
Run	slow flashing
COM ACT	lit (with some interruptions)
Modbus +	3x flashes with interruptions

CPU display after transfer

After transfer the CPU display is as follows:

LED	Response
Run	not lit
COM ACT	not lit
Modbus +	3x flashes with interruptions

Note: The three flash sequence Modbus + display idicates that no communication is present on the bus. This is displayed on Modbus by a non-lit COM ACT display. The COM ACT display is lit again once connection is made with Concept.

Creating the software connection

Carry out the steps given in chapter *Creating a Project, p. 47*.

Momentum first startup for LL984 with EXECLoader

Introduction

This section describes the first startup of Momentum for LL984 when used with Modbus.

> **Note:** Loading the EXEC file for LL984 is not necessary with a new computer, since it is preloaded in the the CPUs Flash RAM. Loading the EXEC file for LL984 is only necessary if you have already loaded the EXEC file for IEC, and now wish to change.
> You should always check to see if a new EXEC version has been released in the meantime. This information and the current EXEC file can be found on our website at www.schneiderautomation.com. You can see the currently loaded version of the EXEC file in Concept using the **Online** → **Online control panel...** menu command.

The first startup is subdivided into 5 main sections:

Start EXECLoader

The procedure for launching EXECLoaders is as follows:

Step	Action
1	Open the Concept start menu. **Response:** All installed Concept programs are displayed as symbols.
2	Click on the symbol with the identifier **EXECLoader**. **Response:** The EXECLoader program is started.
3	Click on the command **Next**, as soon as you have read the information on the page. **Response:** The dialog **Communication Protocol** is opened.

Define Modbus interface

Carry out the following steps to set the Modbus interface:

Step	Action
1	Select the Option button **Modbus RTU (RS232)** for the RTU transfer mode. Select the Option button **Modbus ASCII (RS232)** for the ASCII transfer mode. **Note:** Data transfer can only take place if you have configured the same transfer mode (RTU or ASCII) on the CPU (using a button on the front of the module).
2	Click on the command button **Next**. **Response:** The dialog **Modbus Target** → **RTU/ASCII mode** is opened.
3	Use the command button **COM Port Settings...** to open the dialog **COM Properties**.
4	Use the list field **Connect using:** to select the programming cable interface on the PC (default setting is **COM1**).
5	Use the list field **Bits per second:** to select the Baudrate (default is **9600**).
6	Use the list field **Parity:** to select the parity (default is **EVEN**).
7	Use the list field **Stop Bits** to select the Stop bits (default is **1**).
8	Click on the command button **OK**. **Response:** The dialog is closed and you return to the dialog **Modbus Target** → **RTU/ASCII Mode**.

Protocol settings Carry out the following steps to set the Modbus protocol:

Step	Action
1	Click on the command button **Scan**. **Response:** The nodes on the Modbus network are read and displayed graphically in the left window. A green point in the graphic indicates that the CPU is in RUN mode. To stop the CPU continue as described in step 3.
2	Double-click on the read network node in the graphical display. **Response:**The Modbus address of the node is automatically entered in the textfeld **Modbus Address**.
3	Click the right mouse button in the left window. **Response:** A context menu with individual PLC commands is opened.
4	If the "Run" display is lit on the CPU, stop the program using the command **Stop PLC**. **Response:** A message window appears where you can click **OK** to confirm stopping the CPU.
5	Click the command button **OK**, to confirm stopping the CPU. **Response:** You return to the dialog **Modbus Target** → **RTU mode**, and the green point disappears from the graph.
6	Activate the check box **Bridge**, if the connection to the Modbus node should be made via a Modbus Plus network using a Modbus bridge.
7	Press the appropriate Option button for your system (**PLC**, **Direct Device**, **Local Head**, **Remote I/O Drop**).
8	Click on the command button **Next**. **Response:** The **Operation** dialog is opened.

Select EXEC file

Carry out the following steps to select the EXEC file:

Step	Action
1	Press the Option button **Transfer EXEC to Device**.
2	Click on the command button **Browse...**. **Response:** The Concept directory is opened in a standard window.
3	Double-click on the DAT directory. **Response:** All available*.BIN files are displayed.
4	Click on the *.BIN file that corresponds to your CPU and the desired programming language. See the table *Loading Firmware for Momentum PLC Types, p. 1025*. **Response:** The selected *.BIN file is displayed in **File name:** text field.
5	Click on the command button **Open**. **Response:** You return to the dialog **Operation**, and the path to the selected *.BIN file is displayed in the **Filename** text field.
6	Click on the command button **Next**. **Response:** The dialog **File and Device Info** is opened. Information is provided here about the selected *.BIN file and also about the PLC.
7	Click on the command button **Next**. **Response:** The **Summary** dialog is opened. This gives you an overview of the settings made for you to check.

Load EXEC file

Carry out the following steps to load the EXEC file in the CPU flash RAM:

Step	Action
1	Click on the command button **Transfer**. **Response:** A message box appears warning you that all data available on the PLC will be lost, and the configuration and program must be reloaded on the PLC.
2	Click on the command button **Yes**, to continue the transfer. **Response:** The **Progress** dialog is opened. This gives information about the progress of the transfer in a progress bar and text.
3	Click **Close** once the transfer is complete. **Response:** The dialog is closed, and you return to the dialog **Summary**.
4	Click on the command button **Close**, to close the EXECLoader.

CPU display during transfer

During transfer the CPU display is as follows:

LED	Response
Run	slow flashing
COM ACT	lit (with some interruptions)
Modbus +	3x flashes with interruptions

CPU display after transfer

After transfer the CPU display is as follows:

LED	Response
Run	not lit
COM ACT	not lit
Modbus +	3x flashes with interruptions

Note: The three flash sequence Modbus + display idicates that no communication is present on the bus. This is displayed on Modbus by a non-lit COM ACT display. The COM ACT display is lit again once connection is made with Concept.

Creating the software connection

Carry out the steps given in chapter *Creating a Project, p. 47*.

Startup when using Modbus with DOS Loader

M

Introduction

Overview

This chapter describes loading executive data (EXEC) onto the PLC with the DOS Loader program.

What's in this Chapter?

This chapter contains the following topics:

Quantum first startup with DOS Loader

Introduction

This section describes the first startup of Quantum when used with Modbus.
The first startup is subdivided into 5 main sections:
- *Start DOS Loader, p. 974*
- *Define Modbus interface, p. 974*
- *Protocol settings, p. 975*
- *Select EXEC file, p. 975*
- *Load EXEC file, p. 975*

Start DOS Loader

The procedure for launching DOS Loaders is as follows:

Step	Action
1	If the CPU display "Run" is lit, stop the program with Concept (in main menu **Online**). **Response:** The "Run" display is no longer lit; the "Ready" is now lit.
2	Open the directory DAT (CONCEPT\DAT).
3	Double-click on the file LOADER.EXE. **Response:** The installation program for the Executive file (EXEC) is started.
4	Select the option **Communication Parameters**. **Response:** The dialog box **Communication setup** is opened.
5	Select the **Modbus**option. **Response:** The **Modbus communication setup** window is opened.

Define Modbus interface

Carry out the following steps to set the Modbus interface:

Step	Action
1	Select the programming cable interface on the PC (default setting is **COM1**).
2	Select the Baudrate (default is **9600**).
3	Select the parity (default is **EVEN**).
4	Select the Option button **RTU - 8 Bits** for the RTU transfer mode. Select the option **ASCII -7 Bits** for the ASCII transfer mode.
5	Select the Stop bits (default is **1**). **Note:** Data transfer can only take place if you have configured the same transfer mode (ASCII or RTU) on the CPU (using a button on the front of the module).

Protocol settings

Carry out the following steps to set the Modbus protocol:

Step	Action
1	Enter in **Enter PLC Address:** the node address set on the CPU (using a rotary switch on the back of the module).
2	Select the **TARGET PATH 0** option.
3	Select the **ACCEPT CHANGES** option. **Response:** You return to the main menu.

Select EXEC file

Carry out the following steps to select the EXEC file:

Step	Action
1	Select the option **File Selection**. **Response:** The **File Selection** window is opened.
2	Click on the *.BIN file that corresponds to your CPU and the desired programming language. See the table *Loading Firmware for Quantum PLC Types, p. 1024*.
3	Confirm your selection with RETURN. **Response:** You return to the main menu.

Load EXEC file

Carry out the following steps to load the EXEC file in the CPU flash RAM:

Step	Action
1	Select the option **Load File To PLC**. **Response:** The **Loading Process** window is opened, and the warning "The PROGRAM contents of the device being loaded could be lost after loading a new Executive. The CONTENTS stored in the Micro H H P will always be lost after loading a new Executive. Would you like to continue (Y/N) ? N" is displayed.
2	Acknowledge the warning with **Y**. **Response:** The message "Node failed to enter normal mode" appears.
3	The DOWNLOAD PROGRESS window appears which shows the transfer rate. **Response:** After the transfer is complete, the message "Download Operation Successful" appears.
4	Confirm the message with RETURN. **Response:** You return to the main menu.
5	Select the **Exit Program** option. **Response:** The DOS Loader is exited.

CPU display during transfer

During transfer the CPU display is as follows:

LED	Response
Ready	lit
Run	slow flashing
Modbus	lit (with some interruptions)
Modbus +	3x flashes with interruptions

CPU display after transfer

After transfer the CPU display is as follows:

LED	Response
Ready	lit
Run	not lit
Modbus	not lit
Modbus +	3x flashes with interruptions

Note: The three flash sequence Modbus + display idicates that no communication is present on the bus. This is displayed on Modbus by a non-lit Modbus display. The Modbus display is lit again once connection is made with Concept.

Creating the software connection

Carry out the steps given in chapter *Creating a Project, p. 47*.

Compact first startup with DOS Loader

Introduction

This section describes the first startup of Compact when used with Modbus.
The first startup is subdivided into 5 main sections:
- *Start DOS Loader, p. 977*
- *Define Modbus interface, p. 977*
- *Protocol settings, p. 978*
- *Select EXEC file, p. 978*
- *Load EXEC file, p. 978*

Start DOS Loader

The procedure for launching DOS Loaders is as follows:

Step	Action
1	If the CPU display "Run" is lit, stop the program with Concept (in main menu **Online**). **Response:** The "Run" display is no longer lit; the "Ready" is now lit.
2	Open the directory DAT (CONCEPT\DAT).
3	Double-click on the file LOADER.EXE. **Response:** The installation program for the Executive file (EXEC) is started.
4	Select the option **Communication Parameters**. **Response:** The dialog box **Communication setup** is opened.
5	Select the **Modbus**option. **Response:** The **Modbus communication setup** window is opened.

Define Modbus interface

Carry out the following steps to set the Modbus interface:

Step	Action
1	Select the programming cable interface on the PC (default setting is **COM1**).
2	Select the Baudrate (default is **9600**).
3	Select the parity (default is **EVEN**).
4	Select the Option button **RTU - 8 Bits** for the RTU transfer mode. Select the option **ASCII -7 Bits** for the ASCII transfer mode.
5	Select the Stop bits (default is **1**). **Note:** Data transfer can only take place if you have configured the same transfer mode (ASCII or RTU) on the CPU (using a button on the front of the module).

Protocol settings

Carry out the following steps to set the Modbus protocol:

Step	Action
1	Enter in **Enter PLC Address:** the node address set on the CPU (using a rotary switch on the back of the module).
2	Select the **TARGET PATH 0** option.
3	Select the **ACCEPT CHANGES** option. **Response:** You return to the main menu.

Select EXEC file

Carry out the following steps to select the EXEC file:

Step	Action
1	Select the option **File Selection**. **Response:** The **File Selection** window is opened.
2	Select the *.BIN file CTSX201D.
3	Confirm your selection with RETURN. **Response:** You return to the main menu.

Load EXEC file

Carry out the following steps to load the EXEC file in the CPU flash RAM:

Step	Action
1	Select the option **Load File To PLC**. **Response:** The **Loading Process** window is opened, and the warning "The PROGRAM contents of the device being loaded could be lost after loading a new Executive. The CONTENTS stored in the Micro H H P will always be lost after loading a new Executive. Would you like to continue (Y/N) ? N" is displayed.
2	Acknowledge the warning with **Y**. **Response:** The message "Node failed to enter normal mode" appears.
3	The DOWNLOAD PROGRESS window appears which shows the transfer rate. **Response:** After the transfer is complete, the message "Download Operation Successful" appears.
4	Confirm the message with RETURN. **Response:** You return to the main menu.
5	Select the **Exit Program** option. **Response:** The DOS Loader is exited.

CPU display during transfer

During transfer the CPU display is as follows:

LED	Response
Ready	lit
Run	slow flashing
Modbus	not lit
Modbus +	fast flashing

CPU display after transfer

After transfer the CPU display is as follows:

LED	Response
Ready	lit
Run	not lit
Modbus	not lit
Modbus +	fast flashing

Note: The three flash sequence Modbus + display idicates that no communication is present on the bus. This is displayed on Modbus by a non-lit Modbus display. The Modbus display is lit again once connection is made with Concept.

Creating the software connection

Carry out the steps given in chapter *Creating a Project, p. 47*.

Momentum first startup for IEC with DOS Loader

Introduction

This section describes the first startup of Momentum for IEC when used with Modbus.
The first startup is subdivided into 5 main sections:
- *Start DOS Loader, p. 980*
- *Define Modbus interface, p. 980*
- *Protocol settings, p. 981*
- *Select EXEC file, p. 981*
- *Load EXEC file, p. 981*

Start DOS Loader

The procedure for launching DOS Loaders is as follows:

Step	Action
1	If the CPU display "Run" is lit, stop the program with Concept (in main menu **Online**). **Response:** The "Run" display is no longer lit; the "Ready" is now lit.
2	Open the directory DAT (CONCEPT\DAT).
3	Double-click on the file LOADER.EXE. **Response:** The installation program for the Executive file (EXEC) is started.
4	Select the option **Communication Parameters**. **Response:** The dialog box **Communication setup** is opened.
5	Select the **Modbus** option. **Response:** The **Modbus communication setup** window is opened.

Define Modbus interface

Carry out the following steps to set the Modbus interface:

Step	Action
1	Select the programming cable interface on the PC (default setting is **COM1**).
2	Select the Baudrate (default is **9600**).
3	Select the parity (default is **EVEN**).
4	Select the Option button **RTU - 8 Bits** for the RTU transfer mode. Select the option **ASCII -7 Bits** for the ASCII transfer mode.
5	Select the Stop bits (default is **1**). **Note:** Data transfer can only take place if you have configured the same transfer mode (ASCII or RTU) on the CPU (using a button on the front of the module).

Protocol settings

Carry out the following steps to set the Modbus protocol:

Step	Action
1	Enter in **Enter PLC Address:** the node address set on the CPU (using a rotary switch on the back of the module).
2	Select the **TARGET PATH 0** option.
3	Select the **ACCEPT CHANGES** option. **Response:** You return to the main menu.

Select EXEC file

Carry out the following steps to select the EXEC file:

Step	Action
1	Select the option **File Selection**. **Response:** The **File Selection** window is opened.
2	Click on the *.BIN file that corresponds to your CPU and the desired programming language. See the table *Loading Firmware for Momentum PLC Types, p. 1025*.
3	Confirm your selection with RETURN. **Response:** You return to the main menu.

Load EXEC file

Carry out the following steps to load the EXEC file in the CPU flash RAM:

Step	Action
1	Select the option **Load File To PLC**. **Response:** The **Loading Process** window is opened, and the warning "The PROGRAM contents of the device being loaded could be lost after loading a new Executive. The CONTENTS stored in the Micro H H P will always be lost after loading a new Executive. Would you like to continue (Y/N) ? N" is displayed.
2	Acknowledge the warning with **Y**. **Response:** The message "Node failed to enter normal mode" appears.
3	The DOWNLOAD PROGRESS window appears which shows the transfer rate. **Response:** After the transfer is complete, the message "Download Operation Successful" appears.
4	Confirm the message with RETURN. **Response:** You return to the main menu.
5	Select the **Exit Program** option. **Response:** The DOS Loader is exited.

CPU display during transfer

During transfer the CPU display is as follows:

LED	Response
Run	slow flashing
COM ACT	lit (with some interruptions)
Modbus +	3x flashes with interruptions

CPU display after transfer

After transfer the CPU display is as follows:

LED	Response
Run	not lit
COM ACT	not lit
Modbus +	3x flashes with interruptions

Note: The three flash sequence Modbus + display idicates that no communication is present on the bus. This is displayed on Modbus by a non-lit COM ACT display. The COM ACT display is lit again once connection is made with Concept.

Creating the software connection

Carry out the steps given in chapter *Creating a Project, p. 47*.

Momentum first startup for LL984 with DOS Loader

Introduction

This section describes the first startup of Momentum for LL984 when used with Modbus.

Note: Loading the EXEC file for LL984 is not necessary with a new computer, since it is preloaded in the the CPUs Flash RAM. Loading the EXEC file for LL984 is only necessary if you have already loaded the EXEC file for IEC, and now wish to change.
You should always check to see if a new EXEC version has been released in the meantime. This information and the current EXEC file can be found on our website at www.schneiderautomation.com. You can see the currently loaded version of the EXEC file in Concept using the **Online** → **Online control panel...** menu command.

The first startup is subdivided into 5 main sections:

- *Start DOS Loader, p. 983*
- *Define Modbus interface, p. 984*
- *Protocol settings, p. 984*
- *Select EXEC file, p. 984*
- *Load EXEC file, p. 985*

Start DOS Loader

The procedure for launching DOS Loaders is as follows:

Step	Action
1	If the CPU display "Run" is lit, stop the program with Concept (in main menu **Online**). **Response:** The "Run" display is no longer lit; the "Ready" is now lit.
2	Open the directory DAT (CONCEPT\DAT).
3	Double-click on the file LOADER.EXE. **Response:** The installation program for the Executive file (EXEC) is started.
4	Select the option **Communication Parameters**. **Response:** The dialog box **Communication setup** is opened.
5	Select the **Modbus**option. **Response:** The **Modbus communication setup** window is opened.

Define Modbus interface

Carry out the following steps to set the Modbus interface:

Step	Action
1	Select the programming cable interface on the PC (default setting is **COM1**).
2	Select the Baudrate (default is **9600**).
3	Select the parity (default is **EVEN**).
4	Select the Option button **RTU - 8 Bits** for the RTU transfer mode. Select the option **ASCII -7 Bits** for the ASCII transfer mode.
5	Select the Stop bits (default is **1**). **Note:** Data transfer can only take place if you have configured the same transfer mode (ASCII or RTU) on the CPU (using a button on the front of the module).

Protocol settings

Carry out the following steps to set the Modbus protocol:

Step	Action
1	Enter in **Enter PLC Address:** the node address set on the CPU (using a rotary switch on the back of the module).
2	Select the **TARGET PATH 0** option.
3	Select the **ACCEPT CHANGES** option. **Response:** You return to the main menu.

Select EXEC file

Carry out the following steps to select the EXEC file:

Step	Action
1	Select the option **File Selection**. **Response:** The **File Selection** window is opened.
2	Click on the *.BIN file that corresponds to your CPU and the desired programming language. See the table *Loading Firmware for Momentum PLC Types, p. 1025*.
3	Confirm your selection with RETURN. **Response:** You return to the main menu.

Load EXEC file

Carry out the following steps to load the EXEC file in the CPU flash RAM:

Step	Action
1	Select the option **Load File To PLC**. **Response:** The **Loading Process** window is opened, and the warning "The PROGRAM contents of the device being loaded could be lost after loading a new Executive. The CONTENTS stored in the Micro H H P will always be lost after loading a new Executive. Would you like to continue (Y/N) ? N" is displayed.
2	Acknowledge the warning with **Y**. **Response:** The message "Node failed to enter normal mode" appears.
3	The DOWNLOAD PROGRESS window appears which shows the transfer rate. **Response:** After the transfer is complete, the message "Download Operation Successful" appears.
4	Confirm the message with RETURN. **Response:** You return to the main menu.
5	Select the **Exit Program** option. **Response:** The DOS Loader is exited.

CPU display during transfer

During transfer the CPU display is as follows:

LED	Response
Run	slow flashing
COM ACT	lit (with some interruptions)
Modbus +	3x flashes with interruptions

CPU display after transfer

After transfer the CPU display is as follows:

LED	Response
Run	not lit
COM ACT	not lit
Modbus +	3x flashes with interruptions

Note: The three flash sequence Modbus + display idicates that no communication is present on the bus. This is displayed on Modbus by a non-lit COM ACT display. The COM ACT display is lit again once connection is made with Concept.

Creating the software connection

Carry out the steps given in chapter *Creating a Project, p. 47*.

Startup when using Modbus Plus with the EXECLoader

N

Introduction

Overview

This chapter describes loading executive data (EXEC) onto the PLC with the EXECLoader.

What's in this Chapter?

This chapter contains the following topics:

Quantum first startup with EXECLoader

Introduction

This section describes the first startup of Quantum when used with Modbus Plus. The first startup is subdivided into 5 main sections:

- *Start EXECLoader, p. 988*
- *Define SA85 adapter, p. 988*
- *Protocol settings, p. 989*
- *Select EXEC file, p. 990*
- *Load EXEC file, p. 990*

Start EXECLoader

The procedure for launching EXECLoaders is as follows:

Step	Action
1	Open the Concept start menu. **Response:** All installed Concept programs are displayed as symbols.
2	Click on the symbol with the identifier **EXECLoader**. **Response:** The EXECLoader program is started.
3	Click on the command **Next**, as soon as you have read the information on the page. **Response:** The dialog **Communication Protocol** is opened.

Define SA85 adapter

Carry out the following steps to define the LAN address set in the CONFIG.SYS file:

Step	Action
1	Select the option button **Modbus Plus**.
2	Click on the command button **Next**. **Response:** The **Modbus Plus Target** dialog is opened.
3	Select from the list **Devices Online:** the adapter address you set when installing the SA85 in the CONFIG.SYS file (Parameter /n).

Protocol settings Carry out the following steps to set the Modbus Plus protocol settings:

Step	Action
1	Click on the command button **Scan**. **Response:** The nodes on the Modbus Plus network are read and displayed graphically in the left window. A green point in the graphic indicates that the CPU is in RUN mode. To stop the CPU continue as described in step 3.
2	Double-click on the read network node in the graphical display. **Response:** The Modbus address of the node is automatically entered in the textfeld In the text field **Modbus Plus Address**.
3	Click the right mouse button in the left window. **Response:** A context menu with individual PLC commands is opened.
4	If the "Run" display is lit on the CPU, stop the program using the command **Stop PLC**. **Response:** A message window appears where you can click **OK** to confirm stopping the CPU.
5	Click the command button **OK**, to confirm stopping the CPU. **Response:** You return to the **Modbus Plus Target** dialog, and the green point disappears from the graph.
6	Activate the check box **Bridge**, if the connection to the Modbus network node should be made via a Modbus network using a Modbus bridge.
7	Press the appropriate Option button for your system (**PLC**, **Local Head**, **Remote I/O Drop**).
8	Click on the command button **Next**. **Response:** The **Operation** dialog is opened.

Select EXEC file

Carry out the following steps to select the EXEC file:

Step	Action
1	Press the Option button **Transfer EXEC to Device**.
2	Click on the command button **Browse...**. **Response:** The Concept directory is opened in a standard window.
3	Double-click on the DAT directory. **Response:** All available*.BIN files are displayed.
4	Click on the *.BIN file that corresponds to your CPU and the desired programming language. See the table *Loading Firmware for Quantum PLC Types, p. 1024*. **Response:** The selected *.BIN file is displayed in **File name:** text field.
5	Click on the command button **Open**. **Response:** You return to the dialog **Operation**, and the path to the selected *.BIN file is displayed in the **Filename** text field.
6	Click on the command button **Next**. **Response:** The dialog **File and Device Info** is opened. Information is provided here about the selected *.BIN file and also about the PLC.
7	Click on the command button **Next**. **Response:** The **Summary** dialog is opened. This gives you an overview of the settings made for you to check.

Load EXEC file

Carry out the following steps to load the EXEC file in the CPU flash RAM:

Step	Action
1	Click on the command button **Transfer**. **Response:** A message box appears warning you that all data available on the PLC will be lost, and the configuration and program must be reloaded on the PLC.
2	Click on the command button **Yes**, to continue the transfer. **Response:** The **Progress** dialog is opened. This gives information about the progress of the transfer in a progress bar and text.
3	Click **Close** once the transfer is complete. **Response:** The dialog is closed, and you return to the dialog **Summary**.
4	Click on the command button **Close**, to close the EXECLoader.

CPU display during transfer

During transfer the CPU display is as follows:

LED	Response
Ready	lit
Run	slow flashing
Modbus	not lit
Modbus +	fast flashing

CPU display after transfer

After transfer the CPU display is as follows:

LED	Response
Ready	lit
Run	not lit
Modbus	not lit
Modbus +	fast flashing

Creating the software connection

Carry out the steps given in chapter *Creating a Project, p. 47*.

Note: If you recieve an error message, close Concept and start the BDRESET.EXE file (in the \Concept directory) to reset the SA85. Then start again from the first step.

Compact first startup with EXECLoader

Introduction

This section describes the first startup of Compact when used with Modbus Plus. The first startup is subdivided into 5 main sections:

- *Start EXECLoader, p. 992*
- *Define SA85 adapter, p. 992*
- *Protocol settings, p. 993*
- *Select EXEC file, p. 994*
- *Load EXEC file, p. 994*

Start EXECLoader

The procedure for launching EXECLoaders is as follows:

Step	Action
1	Open the Concept start menu. **Response:** All installed Concept programs are displayed as symbols.
2	Click on the symbol with the identifier **EXECLoader**. **Response:** The EXECLoader program is started.
3	Click on the command **Next**, as soon as you have read the information on the page. **Response:** The dialog **Communication Protocol** is opened.

Define SA85 adapter

Carry out the following steps to define the LAN address set in the CONFIG.SYS file:

Step	Action
1	Select the option button **Modbus Plus**.
2	Click on the command button **Next**. **Response:** The **Modbus Plus Target** dialog is opened.
3	Select from the list **Devices Online:** the adapter address you set when installing the SA85 in the CONFIG.SYS file (Parameter /n).

Protocol settings Carry out the following steps to set the Modbus Plus protocol settings:

Step	Action
1	Click on the command button **Scan**. **Response:** The nodes on the Modbus Plus network are read and displayed graphically in the left window. A green point in the graphic indicates that the CPU is in RUN mode. To stop the CPU continue as described in step 3.
2	Double-click on the read network node in the graphical display. **Response:** The Modbus address of the node is automatically entered in the textfeld In the text field **Modbus Plus Address**.
3	Click the right mouse button in the left window. **Response:** A context menu with individual PLC commands is opened.
4	If the "Run" display is lit on the CPU, stop the program using the command **Stop PLC**. **Response:** A message window appears where you can click **OK** to confirm stopping the CPU.
5	Click the command button **OK**, to confirm stopping the CPU. **Response:** You return to the **Modbus Plus Target** dialog, and the green point disappears from the graph.
6	Activate the check box **Bridge**, if the connection to the Modbus network node should be made via a Modbus network using a Modbus bridge.
7	Press the appropriate Option button for your system (**PLC**, **Local Head**, **Remote I/O Drop**).
8	Click on the command button **Next**. **Response:** The **Operation** dialog is opened.

Select EXEC file

Carry out the following steps to select the EXEC file:

Step	Action
1	Press the Option button **Transfer EXEC to Device**.
2	Click on the command button **Browse...**. **Response:** The Concept directory is opened in a standard window.
3	Double-click on the DAT directory. **Response:** All available*.BIN files are displayed.
4	Click on the *.BIN file CTSX201D. **Response:** The selected *.BIN file is displayed in **File name:** text field.
5	Click on the command button **Open**. **Response:** You return to the dialog **Operation**, and the path to the selected *.BIN file is displayed in the **Filename** text field.
6	Click on the command button **Next**. **Response:** The dialog **File and Device Info** is opened. Information is provided here about the selected *.BIN file and also about the PLC.
7	Click on the command button **Next**. **Response:** The **Summary** dialog is opened. This gives you an overview of the settings made for you to check.

Load EXEC file

Carry out the following steps to load the EXEC file in the CPU flash RAM:

Step	Action
1	Click on the command button **Transfer**. **Response:** A message box appears warning you that all data available on the PLC will be lost, and the configuration and program must be reloaded on the PLC.
2	Click on the command button **Yes**, to continue the transfer. **Response:** The **Progress** dialog is opened. This gives information about the progress of the transfer in a progress bar and text.
3	Click **Close** once the transfer is complete. **Response:** The dialog is closed, and you return to the dialog **Summary**.
4	Click on the command button **Close**, to close the EXECLoader.

CPU display during transfer

During transfer the CPU display is as follows:

LED	Response
Ready	lit
Run	slow flashing
Modbus	not lit
Modbus +	fast flashing

CPU display after transfer

After transfer the CPU display is as follows:

LED	Response
Ready	lit
Run	not lit
Modbus	not lit
Modbus +	fast flashing

Creating the software connection

Carry out the steps given in chapter *Creating a Project, p. 47*.

Note: If you recieve an error message, close Concept and start the BDRESET.EXE file (in the \Concept directory) to reset the SA85. Then start again from the first step.

Atrium first startup with EXECLoader

Introduction

This section describes the first startup of Atrium when used with Modbus Plus. The hardware requirements for loading EXEC files can be seen in the "Modicon TSX Atrium" manual.
The first startup is subdivided into 5 main sections:

- *Start EXECLoader, p. 996*
- *Define SA85 adapter, p. 996*
- *Protocol settings, p. 997*
- *Select EXEC file, p. 998*
- *Load EXEC file, p. 998*

Start EXECLoader

The procedure for launching EXECLoaders is as follows:

Step	Action
1	Open the Concept start menu. **Response:** All installed Concept programs are displayed as symbols.
2	Click on the symbol with the identifier **EXECLoader**. **Response:** The EXECLoader program is started.
3	Click on the command **Next**, as soon as you have read the information on the page. **Response:** The dialog **Communication Protocol** is opened.

Define SA85 adapter

Carry out the following steps to define the LAN address set in the CONFIG.SYS file:

Step	Action
1	Select the option button **Modbus Plus**.
2	Click on the command button **Next**. **Response:** The **Modbus Plus Target** dialog is opened.
3	Select from the list **Devices Online:** the adapter address you set when installing the SA85 in the CONFIG.SYS file (Parameter /n).

Protocol settings Carry out the following steps to set the Modbus Plus protocol settings:

Step	Action
1	Click on the command button **Scan**. **Response:** The nodes on the Modbus Plus network are read and displayed graphically in the left window. A green point in the graphic indicates that the CPU is in RUN mode. To stop the CPU continue as described in step 3.
2	Double-click on the read network node in the graphical display. **Response:** The Modbus address of the node is automatically entered in the textfeld In the text field **Modbus Plus Address**.
3	Click the right mouse button in the left window. **Response:** A context menu with individual PLC commands is opened.
4	If the "Run" display is lit on the CPU, stop the program using the command **Stop PLC**. **Response:** A message window appears where you can click **OK** to confirm stopping the CPU.
5	Click the command button **OK**, to confirm stopping the CPU. **Response:** You return to the **Modbus Plus Target** dialog, and the green point disappears from the graph.
6	Activate the check box **Bridge**, if the connection to the Modbus network node should be made via a Modbus network using a Modbus bridge.
7	Press the appropriate Option button for your system (**PLC**, **Local Head**, **Remote I/O Drop**).
8	Click on the command button **Next**. **Response:** The **Operation** dialog is opened.

Select EXEC file

Carry out the following steps to select the EXEC file:

Step	Action
1	Press the Option button **Transfer EXEC to Device**.
2	Click on the command button **Browse...**. **Response:** The Concept directory is opened in a standard window.
3	Double-click on the DAT directory. **Response:** All available*.BIN files are displayed.
4	Click on the *.BIN file that corresponds to your CPU and the desired programming language. See the table *Loading Firmware for Atrium PLC Types, p. 1026*. **Response:** The selected *.BIN file is displayed in **File name:** text field.
5	Click on the command button **Open**. **Response:** You return to the dialog **Operation**, and the path to the selected *.BIN file is displayed in the **Filename** text field.
6	Click on the command button **Next**. **Response:** The dialog **File and Device Info** is opened. Information is provided here about the selected *.BIN file and also about the PLC.
7	Click on the command button **Next**. **Response:** The **Summary** dialog is opened. This gives you an overview of the settings made for you to check.

Load EXEC file

Carry out the following steps to load the EXEC file in the CPU flash RAM:

Step	Action
1	Click on the command button **Transfer**. **Response:** A message box appears warning you that all data available on the PLC will be lost, and the configuration and program must be reloaded on the PLC.
2	Click on the command button **Yes**, to continue the transfer. **Response:** The **Progress** dialog is opened. This gives information about the progress of the transfer in a progress bar and text.
3	Click **Close** once the transfer is complete. **Response:** The dialog is closed, and you return to the dialog **Summary**.
4	Click on the command button **Close**, to close the EXECLoader.

Creating the software connection

Carry out the steps given in chapter *Creating a Project, p. 47*.

Note: If you recieve an error message, close Concept and start the BDRESET.EXE file (in the \Concept directory) to reset the SA85. Then start again from the first step.

Momentum first startup for IEC with EXECLoader

Introduction

This section describes the first startup of Momentum for IEC when used with Modbus Plus.
The first startup is subdivided into 5 main sections:

- *Start EXECLoader, p. 999*
- *Define SA85 adapter, p. 999*
- *Protocol settings, p. 1000*
- *Select EXEC file, p. 1001*
- *Load EXEC file, p. 1001*

Start EXECLoader

The procedure for launching EXECLoaders is as follows:

Step	Action
1	Open the Concept start menu. **Response:** All installed Concept programs are displayed as symbols.
2	Click on the symbol with the identifier **EXECLoader**. **Response:** The EXECLoader program is started.
3	Click on the command **Next**, as soon as you have read the information on the page. **Response:** The dialog **Communication Protocol** is opened.

Define SA85 adapter

Carry out the following steps to define the LAN address set in the CONFIG.SYS file:

Step	Action
1	Select the option button **Modbus Plus**.
2	Click on the command button **Next**. **Response:** The **Modbus Plus Target** dialog is opened.
3	Select from the list **Devices Online:** the adapter address you set when installing the SA85 in the CONFIG.SYS file (Parameter /n).

Protocol settings

Carry out the following steps to set the Modbus Plus protocol settings:

Step	Action
1	Click on the command button **Scan**. **Response:** The nodes on the Modbus Plus network are read and displayed graphically in the left window. A green point in the graphic indicates that the CPU is in RUN mode. To stop the CPU continue as described in step 3.
2	Double-click on the read network node in the graphical display. **Response:** The Modbus address of the node is automatically entered in the textfeld In the text field **Modbus Plus Address**.
3	Click the right mouse button in the left window. **Response:** A context menu with individual PLC commands is opened.
4	If the "Run" display is lit on the CPU, stop the program using the command **Stop PLC**. **Response:** A message window appears where you can click **OK** to confirm stopping the CPU.
5	Click the command button **OK**, to confirm stopping the CPU. **Response:** You return to the **Modbus Plus Target** dialog, and the green point disappears from the graph.
6	Activate the check box **Bridge**, if the connection to the Modbus network node should be made via a Modbus network using a Modbus bridge.
7	Press the appropriate Option button for your system (**PLC**, **Local Head**, **Remote I/O Drop**).
8	Click on the command button **Next**. **Response:** The **Operation** dialog is opened.

Select EXEC file

Carry out the following steps to select the EXEC file:

Step	Action
1	Press the Option button **Transfer EXEC to Device**.
2	Click on the command button **Browse...**. **Response:** The Concept directory is opened in a standard window.
3	Double-click on the DAT directory. **Response:** All available*.BIN files are displayed.
4	Click on the *.BIN file that corresponds to your CPU and the desired programming language. See the table *Loading Firmware for Momentum PLC Types, p. 1025*. **Response:** The selected *.BIN file is displayed in **File name:** text field.
5	Click on the command button **Open**. **Response:** You return to the dialog **Operation**, and the path to the selected *.BIN file is displayed in the **Filename** text field.
6	Click on the command button **Next**. **Response:** The dialog **File and Device Info** is opened. Information is provided here about the selected *.BIN file and also about the PLC.
7	Click on the command button **Next**. **Response:** The **Summary** dialog is opened. This gives you an overview of the settings made for you to check.

Load EXEC file

Carry out the following steps to load the EXEC file in the CPU flash RAM:

Step	Action
1	Click on the command button **Transfer**. **Response:** A message box appears warning you that all data available on the PLC will be lost, and the configuration and program must be reloaded on the PLC.
2	Click on the command button **Yes**, to continue the transfer. **Response:** The **Progress** dialog is opened. This gives information about the progress of the transfer in a progress bar and text.
3	Click **Close** once the transfer is complete. **Response:** The dialog is closed, and you return to the dialog **Summary**.
4	Click on the command button **Close**, to close the EXECLoader.

CPU display during transfer

During transfer the CPU display is as follows:

LED	Response
Run	slow flashing
COM ACT	not lit
Modbus +	fast flashing

CPU display after transfer

After transfer the CPU display is as follows:

LED	Response
Run	not lit
COM ACT	not lit
Modbus +	fast flashing

Creating the software connection

Carry out the steps given in chapter *Creating a Project, p. 47*.

Note: If you recieve an error message, close Concept and start the BDRESET.EXE file (in the \Concept directory) to reset the SA85. Then start again from the first step.

Momentum first startup for LL984 with EXECLoader

Introduction

This section describes the first startup of Momentum for LL984 when used with Modbus Plus.

Note: Loading the EXEC file for LL984 is not necessary with a new computer, since it is preloaded in the the CPUs Flash RAM. Loading the EXEC file for LL984 is only necessary if you have already loaded the EXEC file for IEC, and now wish to change.
You should always check to see if a new EXEC version has been released in the meantime. This information and the current EXEC file can be found on our website at www.schneiderautomation.com. You can see the currently loaded version of the EXEC file in Concept using the **Online** → **Online control panel...** menu command.

The first startup is subdivided into 5 main sections:

- *Start EXECLoader, p. 1003*
- *Define SA85 adapter, p. 1003*
- *Protocol settings, p. 1004*
- *Select EXEC file, p. 1005*
- *Load EXEC file, p. 1005*

Start EXECLoader

The procedure for launching EXECLoaders is as follows:

Step	Action
1	Open the Concept start menu. **Response:** All installed Concept programs are displayed as symbols.
2	Click on the symbol with the identifier **EXECLoader**. **Response:** The EXECLoader program is started.
3	Click on the command **Next**, as soon as you have read the information on the page. **Response:** The dialog **Communication Protocol** is opened.

Define SA85 adapter

Carry out the following steps to define the LAN address set in the CONFIG.SYS file:

Step	Action
1	Select the option button **Modbus Plus**.
2	Click on the command button **Next**. **Response:** The **Modbus Plus Target** dialog is opened.
3	Select from the list **Devices Online:** the adapter address you set when installing the SA85 in the CONFIG.SYS file (Parameter /n).

Protocol settings Carry out the following steps to set the Modbus Plus protocol settings:

Step	Action
1	Click on the command button **Scan**. **Response:** The nodes on the Modbus Plus network are read and displayed graphically in the left window. A green point in the graphic indicates that the CPU is in RUN mode. To stop the CPU continue as described in step 3.
2	Double-click on the read network node in the graphical display. **Response:** The Modbus address of the node is automatically entered in the textfeld In the text field **Modbus Plus Address**.
3	Click the right mouse button in the left window. **Response:** A context menu with individual PLC commands is opened.
4	If the "Run" display is lit on the CPU, stop the program using the command **Stop PLC**. **Response:** A message window appears where you can click **OK** to confirm stopping the CPU.
5	Click the command button **OK**, to confirm stopping the CPU. **Response:** You return to the **Modbus Plus Target** dialog, and the green point disappears from the graph.
6	Activate the check box **Bridge**, if the connection to the Modbus network node should be made via a Modbus network using a Modbus bridge.
7	Press the appropriate Option button for your system (**PLC**, **Local Head**, **Remote I/O Drop**).
8	Click on the command button **Next**. **Response:** The **Operation** dialog is opened.

Select EXEC file

Carry out the following steps to select the EXEC file:

Step	Action
1	Press the Option button **Transfer EXEC to Device**.
2	Click on the command button **Browse...**. **Response:** The Concept directory is opened in a standard window.
3	Double-click on the DAT directory. **Response:** All available*.BIN files are displayed.
4	Click on the *.BIN file that corresponds to your CPU and the desired programming language. See the table *Loading Firmware for Momentum PLC Types, p. 1025*. **Response:** The selected *.BIN file is displayed in **File name:** text field.
5	Click on the command button **Open**. **Response:** You return to the dialog **Operation**, and the path to the selected *.BIN file is displayed in the **Filename** text field.
6	Click on the command button **Next**. **Response:** The dialog **File and Device Info** is opened. Information is provided here about the selected *.BIN file and also about the PLC.
7	Click on the command button **Next**. **Response:** The **Summary** dialog is opened. This gives you an overview of the settings made for you to check.

Load EXEC file

Carry out the following steps to load the EXEC file in the CPU flash RAM:

Step	Action
1	Click on the command button **Transfer**. **Response:** A message box appears warning you that all data available on the PLC will be lost, and the configuration and program must be reloaded on the PLC.
2	Click on the command button **Yes**, to continue the transfer. **Response:** The **Progress** dialog is opened. This gives information about the progress of the transfer in a progress bar and text.
3	Click **Close** once the transfer is complete. **Response:** The dialog is closed, and you return to the dialog **Summary**.
4	Click on the command button **Close**, to close the EXECLoader.

CPU display during transfer

During transfer the CPU display is as follows:

LED	Response
Run	slow flashing
COM ACT	not lit
Modbus +	fast flashing

CPU display after transfer

After transfer the CPU display is as follows:

LED	Response
Run	not lit
COM ACT	not lit
Modbus +	fast flashing

Creating the software connection

Carry out the steps given in chapter *Creating a Project, p. 47*.

> **Note:** If you recieve an error message, close Concept and start the BDRESET.EXE file (in the \Concept directory) to reset the SA85. Then start again from the first step.

Startup when using Modbus Plus with DOS Loader

O

Introduction

Overview

This chapter describes loading executive data (EXEC) onto the PLC with the DOS Loader program.

What's in this Chapter?

This chapter contains the following topics:

Quantum first startup with DOS Loader

Introduction

This section describes the first startup of Quantum when used with Modbus Plus. The first startup is subdivided into 5 main sections:

- *Start DOS Loader, p. 1008*
- *Define SA85 adapter, p. 1008*
- *Protocol settings, p. 1009*
- *Select EXEC file, p. 1009*
- *Load EXEC file, p. 1010*

Start DOS Loader

The procedure for launching DOS Loaders is as follows:

Step	Action
1	If the CPU display "Run" is lit, stop the program with Concept (in main menu **Online**). **Response:** The "Run" display is no longer lit; the "Ready" is now lit.
2	Open the directory DAT (CONCEPT\DAT).
3	Double-click on the file LOADER.EXE. **Response:** The installation program for the Executive file (EXEC) is started.
4	Select the option **Communication Parameters**. **Response:** The dialog box **Communication setup** is opened.
5	Select the **Modbus Plus**option. **Response:** The **Modbus Plus communication setup** window is opened.

Define SA85 adapter

Carry out the following steps to define the LAN address set in the CONFIG.SYS file:

Step	Action
1	Select the adapter address you set when installing the SA85 in the CONFIG.SYS file (Parameter /n).
2	Confirm your selection with RETURN.

Protocol settings

Carry out the following steps to set the Modbus Plus protocol settings:

Step	Action
1	Enter in **Enter First Routing Path:** the node address set on the CPU (using a rotary switch on the back of the module).
2	Confirm the entry with RETURN. **Response:** The option **Enter second Routing Path:** appears.
3	Acknowledge the option with RETURN. **Response:** The window for selecting the TARGET PATH appears.
4	Select the **TARGET PATH 1**option.
5	Enter in **Enter Software Interrupt ->** the Interrupt (5c or 5d) selected in the CONFIG.SYS file.
6	Select the **ACCEPT CHANGES**option. **Response:** You return to the main menu.

Select EXEC file

Carry out the following steps to select the EXEC file:

Step	Action
1	Select the option **File Selection**. **Response:** The **File Selection** window is opened.
2	Click on the *.BIN file that corresponds to your CPU and the desired programming language. See the table *Loading Firmware for Quantum PLC Types, p. 1024*.
3	Confirm your selection with RETURN. **Response:** You return to the main menu.

Load EXEC file

Carry out the following steps to load the EXEC file in the CPU flash RAM:

Step	Action
1	Select the option **Load File To PLC**. **Response:** The **Loading Process** window is opened, and the warning "The PROGRAM contents of the device being loaded could be lost after loading a new Executive. The CONTENTS stored in the Micro H H P will always be lost after loading a new Executive. Would you like to continue (Y/N) ? N" is displayed.
2	Acknowledge the warning with **Y**. **Response:** The message "Node failed to enter normal mode" appears.
3	The DOWNLOAD PROGRESS window appears which shows the transfer rate. **Response:** After the transfer is complete, the message "Download Operation Successful" appears.
4	Confirm the message with RETURN. **Response:** You return to the main menu.
5	Select the **Exit Program**option. **Response:** The DOS Loader is exited.

CPU display during transfer

During transfer the CPU display is as follows:

LED	Response
Ready	lit
Run	slow flashing
Modbus	not lit
Modbus +	fast flashing

CPU display after transfer

After transfer the CPU display is as follows:

LED	Response
Ready	lit
Run	not lit
Modbus	not lit
Modbus +	fast flashing

Creating the software connection

Carry out the steps given in chapter *Creating a Project, p. 47*.

Note: If you recieve an error message, close Concept and start the BDRESET.EXE file (in the \Concept directory) to reset the SA85. Then start again from the first step.

Compact first startup with DOS Loader

Introduction

This section describes the first startup of Compact when used with Modbus Plus. The first startup is subdivided into 5 main sections:

- *Start DOS Loader, p. 1011*
- *Define SA85 adapter, p. 1011*
- *Protocol settings, p. 1012*
- *Select EXEC file, p. 1012*
- *Load EXEC file, p. 1013*

Start DOS Loader

The procedure for launching DOS Loaders is as follows:

Step	Action
1	If the CPU display "Run" is lit, stop the program with Concept (in main menu **Online**). **Response:** The "Run" display is no longer lit; the "Ready" is now lit.
2	Open the directory DAT (CONCEPT\DAT).
3	Double-click on the file LOADER.EXE. **Response:** The installation program for the Executive file (EXEC) is started.
4	Select the option **Communication Parameters**. **Response:** The dialog box **Communication setup** is opened.
5	Select the **Modbus Plus**option. **Response:** The **Modbus Plus communication setup** window is opened.

Define SA85 adapter

Carry out the following steps to define the LAN address set in the CONFIG.SYS file:

Step	Action
1	Select the adapter address you set when installing the SA85 in the CONFIG.SYS file (Parameter /n).
2	Confirm your selection with RETURN.

Protocol settings

Carry out the following steps to set the Modbus Plus protocol settings:

Step	Action
1	Enter in **Enter First Routing Path:** the node address set on the CPU (using a rotary switch on the back of the module).
2	Confirm the entry with RETURN. **Response:** The option **Enter second Routing Path:** appears.
3	Acknowledge the option with RETURN. **Response:** The window for selecting the TARGET PATH appears.
4	Select the **TARGET PATH 1**option.
5	Enter in **Enter Software Interrupt ->** the Interrupt (5c or 5d) selected in the CONFIG.SYS file.
6	Select the **ACCEPT CHANGES**option. **Response:** You return to the main menu.

Select EXEC file

Carry out the following steps to select the EXEC file:

Step	Action
1	Select the option **File Selection**. **Response:** The **File Selection** window is opened.
2	Select the *.BIN file CTSX201D.
3	Confirm your selection with RETURN. **Response:** You return to the main menu.

Load EXEC file

Carry out the following steps to load the EXEC file in the CPU flash RAM:

Step	Action
1	Select the option **Load File To PLC**. **Response:** The **Loading Process** window is opened, and the warning "The PROGRAM contents of the device being loaded could be lost after loading a new Executive. The CONTENTS stored in the Micro H H P will always be lost after loading a new Executive. Would you like to continue (Y/N) ? N" is displayed.
2	Acknowledge the warning with **Y**. **Response:** The message "Node failed to enter normal mode" appears.
3	The DOWNLOAD PROGRESS window appears which shows the transfer rate. **Response:** After the transfer is complete, the message "Download Operation Successful" appears.
4	Confirm the message with RETURN. **Response:** You return to the main menu.
5	Select the **Exit Program**option. **Response:** The DOS Loader is exited.

Creating the software connection

Carry out the steps given in chapter *Creating a Project, p. 47*.

Note: If you recieve an error message, close Concept and start the BDRESET.EXE file (in the \Concept directory) to reset the SA85. Then start again from the first step.

Atrium first startup with DOS Loader

Introduction

This section describes the first startup of Atrium when used with Modbus Plus. The hardware requirements for loading EXEC files can be seen in the "Modicon TSX Atrium" manual.
The first startup is subdivided into 5 main sections:

Start DOS Loader

The procedure for launching DOS Loaders is as follows:

Step	Action
1	If the CPU display "Run" is lit, stop the program with Concept (in main menu **Online**). **Response:** The "Run" display is no longer lit; the "Ready" is now lit.
2	Open the directory DAT (CONCEPT\DAT).
3	Double-click on the file LOADER.EXE. **Response:** The installation program for the Executive file (EXEC) is started.
4	Select the option **Communication Parameters**. **Response:** The dialog box **Communication setup** is opened.
5	Select the **Modbus Plus**option. **Response:** The **Modbus Plus communication setup** window is opened.

Define SA85 adapter

Carry out the following steps to define the LAN address set in the CONFIG.SYS file:

Step	Action
1	Select the adapter address you set when installing the SA85 in the CONFIG.SYS file (Parameter /n).
2	Confirm your selection with RETURN.

Protocol settings

Carry out the following steps to set the Modbus Plus protocol settings:

Step	Action
1	Enter in **Enter First Routing Path:** the node address set on the CPU (using a rotary switch on the back of the module).
2	Confirm the entry with RETURN. **Response:** The option **Enter second Routing Path:** appears.
3	Acknowledge the option with RETURN. **Response:** The window for selecting the TARGET PATH appears.
4	Select the **TARGET PATH 1** option.
5	Enter in **Enter Software Interrupt ->** the Interrupt (5c or 5d) selected in the CONFIG.SYS file.
6	Select the **ACCEPT CHANGES** option. **Response:** You return to the main menu.

Select EXEC file

Carry out the following steps to select the EXEC file:

Step	Action
1	Select the option **File Selection**. **Response:** The **File Selection** window is opened.
2	Click on the *.BIN file that corresponds to your CPU and the desired programming language. See the table *Loading Firmware for Atrium PLC Types, p. 1026*.
3	Confirm your selection with RETURN. **Response:** You return to the main menu.

Load EXEC file

Carry out the following steps to load the EXEC file in the CPU flash RAM:

Step	Action
1	Select the option **Load File To PLC**. **Response:** The **Loading Process** window is opened, and the warning "The PROGRAM contents of the device being loaded could be lost after loading a new Executive. The CONTENTS stored in the Micro H H P will always be lost after loading a new Executive. Would you like to continue (Y/N) ? N" is displayed.
2	Acknowledge the warning with **Y**. **Response:** The message "Node failed to enter normal mode" appears.
3	The DOWNLOAD PROGRESS window appears which shows the transfer rate. **Response:** After the transfer is complete, the message "Download Operation Successful" appears.
4	Confirm the message with RETURN. **Response:** You return to the main menu.
5	Select the **Exit Program** option. **Response:** The DOS Loader is exited.

Creating the software connection

Carry out the steps given in chapter *Creating a Project, p. 47*.

Note: If you recieve an error message, close Concept and start the BDRESET.EXE file (in the \Concept directory) to reset the SA85. Then start again from the first step.

Momentum first startup for IEC with DOS Loader

Introduction

This section describes the first startup of Momentum for IEC when used with Modbus Plus.
The first startup is subdivided into 5 main sections:

Start DOS Loader

The procedure for launching DOS Loaders is as follows:

Step	Action
1	If the CPU display "Run" is lit, stop the program with Concept (in main menu **Online**). **Response:** The "Run" display is no longer lit; the "Ready" is now lit.
2	Open the directory DAT (CONCEPT\DAT).
3	Double-click on the file LOADER.EXE. **Response:** The installation program for the Executive file (EXEC) is started.
4	Select the option **Communication Parameters**. **Response:** The dialog box **Communication setup** is opened.
5	Select the **Modbus Plus** option. **Response:** The **Modbus Plus communication setup** window is opened.

Define SA85 adapter

Carry out the following steps to define the LAN address set in the CONFIG.SYS file:

Step	Action
1	Select the adapter address you set when installing the SA85 in the CONFIG.SYS file (Parameter /n).
2	Confirm your selection with RETURN.

Protocol settings

Carry out the following steps to set the Modbus Plus protocol settings:

Step	Action
1	Enter in **Enter First Routing Path:** the node address set on the CPU (using a rotary switch on the back of the module).
2	Confirm the entry with RETURN. **Response:** The option **Enter second Routing Path:** appears.
3	Acknowledge the option with RETURN. **Response:** The window for selecting the TARGET PATH appears.
4	Select the **TARGET PATH 1** option.
5	Enter in **Enter Software Interrupt ->** the Interrupt (5c or 5d) selected in the CONFIG.SYS file.
6	Select the **ACCEPT CHANGES** option. **Response:** You return to the main menu.

Select EXEC file

Carry out the following steps to select the EXEC file:

Step	Action
1	Select the option **File Selection**. **Response:** The **File Selection** window is opened.
2	Click on the *.BIN file that corresponds to your CPU and the desired programming language. See the table *Loading Firmware for Momentum PLC Types, p. 1025*.
3	Confirm your selection with RETURN. **Response:** You return to the main menu.

Load EXEC file

Carry out the following steps to load the EXEC file in the CPU flash RAM:

Step	Action
1	Select the option **Load File To PLC**. **Response:** The **Loading Process** window is opened, and the warning "The PROGRAM contents of the device being loaded could be lost after loading a new Executive. The CONTENTS stored in the Micro H H P will always be lost after loading a new Executive. Would you like to continue (Y/N) ? N" is displayed.
2	Acknowledge the warning with **Y**. **Response:** The message "Node failed to enter normal mode" appears.
3	The DOWNLOAD PROGRESS window appears which shows the transfer rate. **Response:** After the transfer is complete, the message "Download Operation Successful" appears.
4	Confirm the message with RETURN. **Response:** You return to the main menu.
5	Select the **Exit Program** option. **Response:** The DOS Loader is exited.

CPU display during transfer

During transfer the CPU display is as follows:

LED	Response
Run	slow flashing
COM ACT	not lit
Modbus +	fast flashing

CPU display after transfer

After transfer the CPU display is as follows:

LED	Response
Run	not lit
COM ACT	not lit
Modbus +	fast flashing

Creating the software connection

Carry out the steps given in chapter *Creating a Project, p. 47*.

Note: If you recieve an error message, close Concept and start the BDRESET.EXE file (in the \Concept directory) to reset the SA85. Then start again from the first step.

Momentum first startup for LL984 with DOS Loader

Introduction

This section describes the first startup of Momentum for LL984 when used with Modbus Plus.

> **Note:** Loading the EXEC file for LL984 is not necessary with a new computer, since it is preloaded in the the CPUs Flash RAM. Loading the EXEC file for LL984 is only necessary if you have already loaded the EXEC file for IEC, and now wish to change.
> You should always check to see if a new EXEC version has been released in the meantime. This information and the current EXEC file can be found on our website at www.schneiderautomation.com. You can see the currently loaded version of the EXEC file in Concept using the **Online** → **Online control panel...** menu command.

The first startup is subdivided into 5 main sections:
- *Start DOS Loader, p. 1020*
- *Define SA85 adapter, p. 1021*
- *Protocol settings, p. 1021*
- *Select EXEC file, p. 1021*
- *Load EXEC file, p. 1022*

Start DOS Loader

The procedure for launching DOS Loaders is as follows:

Step	Action
1	If the CPU display "Run" is lit, stop the program with Concept (in main menu **Online**). **Response:** The "Run" display is no longer lit; the "Ready" is now lit.
2	Open the directory DAT (CONCEPT\DAT).
3	Double-click on the file LOADER.EXE. **Response:** The installation program for the Executive file (EXEC) is started.
4	Select the option **Communication Parameters**. **Response:** The dialog box **Communication setup** is opened.
5	Select the **Modbus Plus** option. **Response:** The **Modbus Plus communication setup** window is opened.

Define SA85 adapter

Carry out the following steps to define the LAN address set in the CONFIG.SYS file:

Step	Action
1	Select the adapter address you set when installing the SA85 in the CONFIG.SYS file (Parameter /n).
2	Confirm your selection with RETURN.

Protocol settings

Carry out the following steps to set the Modbus Plus protocol settings:

Step	Action
1	Enter in **Enter First Routing Path:** the node address set on the CPU (using a rotary switch on the back of the module).
2	Confirm the entry with RETURN. **Response:** The option **Enter second Routing Path:** appears.
3	Acknowledge the option with RETURN. **Response:** The window for selecting the TARGET PATH appears.
4	Select the **TARGET PATH 1** option.
5	Enter in **Enter Software Interrupt ->** the Interrupt (5c or 5d) selected in the CONFIG.SYS file.
6	Select the **ACCEPT CHANGES** option. **Response:** You return to the main menu.

Select EXEC file

Carry out the following steps to select the EXEC file:

Step	Action
1	Select the option **File Selection**. **Response:** The **File Selection** window is opened.
2	Click on the *.BIN file that corresponds to your CPU and the desired programming language. See the table *Loading Firmware for Momentum PLC Types, p. 1025*.
3	Confirm your selection with RETURN. **Response:** You return to the main menu.

Load EXEC file

Carry out the following steps to load the EXEC file in the CPU flash RAM:

Step	Action
1	Select the option **Load File To PLC**. **Response:** The **Loading Process** window is opened, and the warning "The PROGRAM contents of the device being loaded could be lost after loading a new Executive. The CONTENTS stored in the Micro H H P will always be lost after loading a new Executive. Would you like to continue (Y/N) ? N" is displayed.
2	Acknowledge the warning with **Y**. **Response:** The message "Node failed to enter normal mode" appears.
3	The DOWNLOAD PROGRESS window appears which shows the transfer rate. **Response:** After the transfer is complete, the message "Download Operation Successful" appears.
4	Confirm the message with RETURN. **Response:** You return to the main menu.
5	Select the **Exit Program** option. **Response:** The DOS Loader is exited.

CPU display during transfer

During transfer the CPU display is as follows:

LED	Response
Run	slow flashing
COM ACT	not lit
Modbus +	fast flashing

CPU display after transfer

After transfer the CPU display is as follows:

LED	Response
Run	not lit
COM ACT	not lit
Modbus +	fast flashing

Creating the software connection

Carry out the steps given in chapter *Creating a Project, p. 47*.

Note: If you recieve an error message, close Concept and start the BDRESET.EXE file (in the \Concept directory) to reset the SA85. Then start again from the first step.

EXEC files

P

Loading Firmware

At a Glance

You obtain the PLC types of the different firmware by loading the EXEC files (*.BIN).

Loading Firmware for Quantum PLC Types

Assigning the EXEC files:

140 CPU	Q186Vxxx (IEC+LL984)	Q486Vxxx (IEC+LL984)	Q58VxxxD (IEC+LL984)	Q5RVxxxD (IEC+LL984)	QIECVxxx (IEC only) *	IEC Memory (kByte)
113 02	X (LL984 only)	-	-	-	-	
113 02S	-	-	-	-	X	max. 150
113 02X	X (LL984 only)	-	-	-	-	
113 03	X	-	-	-	-	max. 136
113 03S	-	-	-	-	X	max. 379
113 03X	X	-	-	-	-	max. 136
213 04	X	-	-	-	-	max. 305
213 04S	-	-	-	-	X	max. 610
213 04X	X	-	-	-	-	max. 305
424 0x	-	X	-	-	-	max. 465
424 0xX	-	X	-	-	-	max. 465
434 12	-	-	X	-	-	max. 890
534 14	-	-	X	-	-	max. 2550
434 12A (Redesigned CPU)	-	-	-	X	-	max. 890
534 14A (Redesigned CPU)	-	-	-	X	-	max. 2550

Note: * After the QIECVxxx.BIN EXEC file is loaded the EMUQ.EXE loadable must be loaded in Concept in the **Loadables** (**PLC Configuration** → **Loadables...**) dialog box.

Loading Firmware for Quantum LL984 Hot Standby Operation

The Quantum CPUs not ending in X or S can be used for the LL984-Hot Standby operation. A special EXEC file must be downloaded onto the CPU for this. The loadable for LL984 Hot Standby (CHS_208.DAT) is automatically installed by the system.

Loading Firmware for Quantum IEC Hot Standby Operation

The CPUs 140 CPU 434 12 and 140 CPU 534 14 can also be used for IEC Hot Standby. A special EXEC file must be downloaded onto the CPU for this. The loadables for IEC Hot Standby (IHSB196.EXE and CHS_208.DAT) are automatically installed by the system.

Loading Firmware for Quantum Equation Editor

The Quantum CPUs not ending in X or S can be used for the LL984- equation editor. A special EXEC file must be downloaded onto the CPU flash for this. This EXEC file is not part of the Concept delivery range but can be obtained via the internet at www.schneiderautomation.com.

Loading Firmware for Momentum PLC Types

Assigning the EXEC files for Momentum PLC type (CPU 171 CCC 7x0 x0):

171 CCC	M1Vxxx (LL984 only)	M1IECxxx (IEC only)	IEC Memory (kByte)
760 10-984	X	-	
760 10-IEC	-	X	256
780 10-984	X	-	
780 10-IEC	-	X	256

Assigning the EXEC files for Momentum PLC type (CPU 171 CCC 9x0 x0):

171 CCC	M1EVxxx (LL984 only)	M1EWIxxx (IEC only)	IEC Memory (kByte)
960 20-984	X	-	
960 30-984	X	-	
960 30-IEC	-	X	220
980 20-984	X	-	
980 30-984	X	-	
980 30-IEC	-	X	220

Assigning the EXEC files for Momentum PLC type (CPU 171 CCS 7x0 x0):

171 CCS	M1Vxxx (LL984 only)	M1IECxxx (IEC only)	IEC Memory (kByte)
700 10	X	-	
700/780 00	X	-	
760 00-984	X	-	
760 00-IEC	-	X	160

The stripped EXEC of the M1 supports up to a maximum of 44 I/O modules.

Loading Firmware for Compact PLC Types

The **CTSXxxxD.BIN** EXEC file must be downloaded onto the CPU flash for all Compact CPUs.

Loading Firmware for Atrium PLC Types

A special EXEC file must be downloaded onto the CPU flash for each Atrium CPU (see table below).

180 CCO	EXEC File
121 01	AI38Vxxx.BIN
241 01 241 11	AI58Vxxx.BIN AI5Vxxx.BIN
242 02	AI5Vxxx.BIN

INI Files

Q

Introduction

Overview

This chapter contains settings that can be made in several INI files.

What's in this Chapter?

This chapter contains the following sections:

Q.1 Settings in the CONCEPT.INI File

Introduction

Overview

This section describes the settings in the CONCEPT.INI file.

What's in this Section?

This section contains the following topics:

General information on the Concept INI file

Introduction

Software settings can be specified in the Concept INI file. Settings generated by the program are also stored in the INI file. The INI file initially contains defaults that can subsequently be changed.

Where is the CONCEPT.INI file situated?

After the installation of Concept, the CONCEPT.INI file can be found in the Windows directory.

Editing the INI File

Various settings are given (exception: path information) and divided into several keywords. The lines that begin with a semicolon (;) represent comments or explain the subsequent setting.
To edit the INI file, only change the lines without semicolons (;) or insert a new line after the comment, in which to specify the path. Then save the modified file.

Note: Changes in the INI file are only accepted after Concept/DFB Editor/ Converter is restarted.

INI Print Settings

Printing FBD Sections

Defining default page break values for FBD sections:

Setting	Description
DX_FBD_PORTRAIT=	Specify portrait width (default value at delivery = 75)
DY_FBD_PORTRAIT=	Specify portrait height (default value at delivery = 100)
DX_FBD_LANDSCAPE=	Specify landscape width (default value at delivery = 100)
DY_FBD_LANDSCAPE=	Specify landscape height (default value at delivery = 50)

Printing LD Sections

Defining default page break values for LD sections:

Setting	Description
DX_LD_PORTRAIT=	Specify portrait width (default value at delivery = 70)
DY_LD_PORTRAIT=	Specify portrait height (default value at delivery = 35)
DX_LD_LANDSCAPE=	Specify landscape width (default value at delivery = 105)
DY_LD_LANDSCAPE=	Specify landscape height (default value at delivery = 18)

Printing SFC Sections

Defining default page break values for SFC sections:

Setting	Description
DX_SFC_PORTRAIT=	Specify portrait width (default value at delivery = 11)
DY_SFC_PORTRAIT=	Specify portrait height (default value at delivery = 20)
DX_SFC_LANDSCAPE=	Specify landscape width (default value at delivery = 15)
DY_SFC_LANDSCAPE=	Specify landscape height (default value at delivery = 11)

INI Settings for Register Address Format, Variable Storage and Project Name Definition

Defining the register address format [Common]

Specifying the register address format (e.g. 4x reference):

Setting	Description
`AddrStyle=0`	0 = 400001 (default)
`AddrStyle=1`	1 = 4:00001 (separator)
`AddrStyle=2`	2 = 4:1 (compact)
`AddrStyle=3`	3 = QW00001 (IEC)

Defining variable storage [Common]

Store variables in file:

Setting	Description
`ExportVariables=1`	After a project has been downloaded and saved, all variables are stored in a file. This file is called *.VAR and is found in the **"Project directory"** → **VAR** → ***.VAR**. All variables and their attributes are shown in this file.
`ExportVariables=0`	Additional storage of variables in a file does not take place.

Determining the validity of digits in project names [Common]

Determining the validity of digits in project names:

Setting	Description
`ProjectPrefixDigit=1`	Project names that begin with a digit are allowed.
`ProjectPrefixDigit=0`	Project names that begin with a digit are not allowed.

Determining the validity of located variables in DFBs [Common]

Determining the validity of located variables in DFBs:

Setting	Description
`AllowLocatedVarsInDFB=1`	Located variables are allowed in DFBs. **Note:** This setting can also be made directly in Concept in **Options** → **Preferences** → **IEC Extensions** → **Allow Located Variable in DFBs** dialog box.

INI Settings for Path Entries and Global DFBs [Path] [Upload]

Defining the Path for Global DFBs and Help Files [Path]

Defining paths:

Setting	Description
`GlobalDFBPath=`	Specify path for global DFBs.
`HelpPath=`	Specify path for help files.
`MBPRoutingPathFile=`	Specify the path for the MBPPATH.INI file.

Defining the Storage of Global DFBs during Upload

Defining a new directory for global DFBs:

Setting	Description
`PreserveGlobalDFBs=1`	During the upload process, a GLB directory for the global DFBs is created in the project directory. By doing this, existing global DFBs in the Concept DFB directory will not be overwritten. **Advantage:** No impact on other projects, as the global DFBs in these projects are not overwritten. **Disadvantage:** Multiple copies of global DFBs. **Note:** Also read the sections entitled *How are Global DFBs Stored?, p. 1034* and *How are Global DFBs Read?, p. 1034*.
`PreserveGlobalDFBs=0`	During the upload process, global DFBs are downloaded into the Concept DFB directory. Different versions of duplicated DFBs are recognized and overwritten after being queried. **Advantage:** Only one copy of global DFBs for several projects. **Disadvantage:** Existing global DFBs whose versions differ from the uploaded DFBs are overwritten. This can cause other projects to be inconsistent in certain circumstances. **Note:** Also read the sections entitled *How are Global DFBs Stored?, p. 1034* and *How are Global DFBs Read?, p. 1034*.

How are Global DFBs Stored?

Storage of global DFBs depends on the settings in the INI file:

If a project...	then the global DFBs are...
is recreated, and no new DFB path has been defined in the INI file,	stored in the x:\CONCEPT\DFB directory.
is recreated, and a new DFB path has been defined in the INI file,	stored in the DFB directory defined in the path.
is uploaded, and the following settings exist in the INI file: - the [Path] option "GlobalDFBPath=x:\DFB", - the [Upload] option "PreserveGlobalDFBs=0",	stored in the DFB directory defined in the path (x:\DFB).
is uploaded, and the following settings exist in the INI file: - the [Path] option "GlobalDFBPath=x:\DFB", - the [Upload] option "PreserveGlobalDFBs=1",	stored in the project's GLB directory. **Note:** The GLB directory is always used first, as soon as the "PreserveGlobalDFBs=1" [Upload] option is specified.

How are Global DFBs Read?

When a project is opened, the system looks for DFBs in the following order:

Step	Description
1	The project directory is searched for an existing GLB directory.
2	The relevant settings are checked in the INI file. For example: [Path]: GlobalDFBPath=x:\DFB [Upload]: PreserveGlobalDFBs=0 In this example, the DFB directory of the path defined is searched for global DFBs.
3	The DFB directory in x:\CONCEPT\DFB is searched.

Only the global DFBs from one directory are used, i.e. if step 1 is unsuccessful, then step 2 follows, step 3 is only performed if neither of the first two are successful.

Representation of Internal Data in the INI File

Representation of Internal Data

The following keywords appear in the INI file and contain internal data according to specific Concept applications:
- [Debug]
- [Configurator]
- [Search]
- [Registration]
- [Register]

INI Settings for the LD Section

Defining the Contact Connection

Defining the contact connection to the power rail:

Setting	Description
`ExtendedAutoConnect=0`	Only the contacts from the first column in the LD editor are automatically connected to the power rail.
`ExtendedAutoConnect=1`	The contacts from the first and second columns in the LD editor are automatically connected to the power rail.

Defining the Number of Columns/Fields

Defining the number of columns/fields (only available when editing with keys):

Setting	Description
`AutowrapColumn=51`	The section contains 51 columns/fields by default. It is possible to set from 2 to 51 columns/fields. When the last column/field is reached, the following objects are automatically placed in the next lines. When this happens, a link with the previous lines is established, i.e. the objects are generated within a common rung. **Note:** Since with automatic line breaking, the objects that follow are placed in the second column/field, it is recommended that you set the contact connection to the power rail as `ExtendedAutoConnect=0`.

INI Settings for Online Processing [Colors]

Online Animation

Specify the representation of the line width and color:

Setting	Description
`AnimationSize=`	Specifying the line width of connections in FBD and LD and for objects in LD: The default setting is 1. It can be set from 1 to 10.
`ColorScheme=`	Specifying the color scheme for FBD, IL, ST, LD and SFC. It is possible to make a setting from 0 to 11. **Note:** An overview of the 12 different color schemes can be found in the online help (see "Colors" in the index).

INI Settings for Warning Messages and the Address Format

Multiple assignment [Warnings]

Reducing the number of warnings (referring to multiple assignment) in the message window:

Setting	Description
`Multiassignment=1`	Warning given if at least one variable X and a component X.C. was written.
`Multiassignment=0`	Warning given if one variable X was written at least twice as a whole.

Address format in LOG file [Logging]

Define address format in LOG file:

Setting	Description
`DD_MONTH_YYYY=1`	In Concept, the month is shown with 3 characters and in English. **Example:** 24-Dec-2002 14:46:24
`DD_MONTH_YYYY=0`	The format set in Windows is shown. The setting can be made in Windows with: **Control Panel** → **Regional Options** → **Date** → **Short date format:**

INI Security Settings

Concept Password Length [Securit]

Define the character length of the Concept password (see Concept Security):

Setting	Description
`MinPasswordLength=X`	The Concept password must have at least X characters. X = 6 to 12

Q.2 Settings in the Projectname.INI File

Introduction

Overview

This section describes the settings in the Projectname.INI file.

What's in this Section?

This section contains the following topics:

General Information for Projectname.INI File

Introduction

Project specific software settings can be made in the Projectname.INI file. The file is either created automatically by Concept (after configuration changes) or can be created by the user. Make sure that the file name always contains the respective project names, e.g. TESTPRJ.INI. The file can contain preset values which can then be changed.

Where is the Projectname.INI file situated?

The Projectname.INI file must be in the Concept project directory , e.g. C:\CONCEPT\TESTPRJ\TESTPRJ.INI

Editing the INI File

The different settings are separated in keywords, e.g. **[Configurator]**. To edit the INI file, enter the command line with the value or the path for the keyword (see *INI Settings for the Event Viewer [Online Events], p. 1039*). Then save the created or modified file.

Note: Changes to the INI file are accepted directly by Concept.

INI Settings for the Event Viewer [Online Events]

Event Viewer [Online Events]

Specifying a user defined error description:

Setting	Description

`Error code="Error description"`
Example:
`-2676="Error in process D"`
	The defined error description is assigned to the error code.
	Note: The error code is entered in the event viewer (in the main menu **Online**).

`Parameter value="Error description"`
Example:
`62860="Error in process B"`
	The defined error description is assigned to the parameter value.
	Note: For EFB ONLEVT, error code -2696 is always used. Therefore the value at the PARAM input is selected for the assignment of the error description.

Interrupt Processing

R

Introduction

Overview

This chapter describes Interrupt processing handling with Quantum and Concept IEC.

What's in this Chapter?

This chapter contains the following sections:

R.1 General information about interrupt sections

General Information about Interrupt Processing

At a Glance

Starting from Concept version 2.6 together with the Quantum modules 140-CPU-434 A or 140-CPU-534 14 A and the 140-HLI-340-00 if required, Interrupt processing functions are made available for the configuration of IEC conforming programs. Special Interrupt sections allows the creation of both Time interrupts (Timer event sections) and I/O interrupts (I/O event sections).
The following Interrupt processing is possible:

- **Timer event sections**
  - Timer event sections enable program sections to be processed in constant, programmable time intervals. The internal time interrupt is used for this.
    To determine the time interrupts, every Timer event section is assigned a constant time value for their execution (scan rate) in a range from 10 ms to 1023s in the **Section Properties for Timer Event Sections** (in the **File** main menu) dialog box. Optimal runtime behavior can be configured using the Interval, Time base and Phase parameters (moving the execution to another cycle with the same interval). The processing of a cyclic section is immediately stopped if a time interrupt occurs. After the Timer event section is processed, the program execution is continued from the point where it was stopped.
    Using the direct I/O blocks IMIO_IN and IMIO_OUT on these sections enables current inputs and outputs to be processed at predetermined intervals. Time critical problems can be easily resolved, i.e. the realization of control loops in closed loop control systems.
    The maximum permitted execution duration for a Timer event section is 20 ms!
- **I/O Event sections**
  - I/O event sections enable program parts to be processed according to a signal change on a specific Hardware interrupt input. The Interrupts required for this (spontaneous I/O) are generated by the 140-HLI-340-00 module.
    Every I/O event section is assigned a pin (input) on the 140-HLI-340-00 module via the **Section Properties for I/O Event Sections** (in the **File** main menu) dialog box. Depending on the parameters set in the Configurator, the signal change on pin (0->1, 1->0 or 0->1 and 1->0) triggers a hardware interrupt on the CPU logic processor. The processing of a cyclic section or Timer event section is immediately stopped if a hardware interrupt occurs. After the I/O event section is processed, the program execution is continued from the point where it was stopped.
    Using the direct I/O blocks IMIO_IN and IMIO_OUT on these sections enables spontaneous outputs (and further inputs) to be processed extremely quickly. A critical event can be reacted to immediately, i.e. independently from the cycle.
    Special EFBs for Interrupt sections allow among other things, the program dependent disable/enable of Interrupt sections.
    The maximum permitted execution duration for an I/O event section is 20 ms!

Note: The direct I/O blocks IMIO_IN and IMIO_OUT only work when the corresponding I/O module is installed on the local backplane or backplane expander!

Limitations

Interrupt sections cannot be used together with the following functions:

- Hot Standby
  - If an additional Hot Standby is configured in a project, an error message is returned!
- ULEX/ASUP with modules 140-NOA-611-00, 140-NOA-611-10 and 140-ESI-062-00
  - The 140-NOA-622-00 can be used together with Interrupt sections instead of the 140-NOA-611-x0.
- LL984 sections

R.2 Interrupt section: Timer event section

Introduction

Overview

This chapter contains a description for timer event sections.

What's in this Section?

This section contains the following topics:

Timer Event Sections

Introduction

Timer event sections are created in the same way as cyclic sections using the **File** → **New Section...** menu command. A maximum of 16 Timer event sections can be created. A Timer event section can only be selected when a CPU 140-CPU-434 A or 140 CPU 534 A is configured in the Configurator.
The CPU Hardware 140 CPU 434 12 A or 140 CPU 534 14 A is required to execute a program with Timer event sections!
A new **Timer Events** group is created automatically for Timer event sections where the Timer event sections are displayed. This group is placed in before the cyclic sections and after the **I/O Events** group in the project browser (see *Execution Order, p. 1051*).
Timer event sections are programmed principally as with cyclic sections (see *Step 3: Creating the User Program, p. 57*), only the selection of existing EFBs is limited.
Blocks (EFB) not available in Timer event sections:
- **F_TRIG**, **R_TRIG** (IEC library, group: Edge Detection)
- **TOF**, **TON**, **TP** (IEC library, group: Timer)
- **ERR2HMI**, **ERRMSG** (DIAGNO library, group: Diag View)
- **ACT_DIA**, **DYN_DIA**, **GRP_DIA**, **LOCK_DIA**, **PRE_DIA**, **REA_DIA** (DIAGNO library, group: Diagnostics)
- **XACT**, **XACT_DIA**, **XDYN_DIA**, **XGRP_DIA**, **XLOCK_DIA**, **XLOCK**, **XPRE_DIA**, **XREA_DIA**, (DIAGNO library, group: Extended)

Timer event section parameters are set in the **Section Properties for Timer Event Sections** dialog box using the **Interval**, **Time Base** and **Phase** parameters. This enables you to specify at what intervals (scan rate) the sections is processed.
Simultaneously created Timer event sections, for example with the same scan rate, are processed consecutively according to the execution order and priority in the same cycle. The runtimes of these Timer event sections are added in this cycle and make it longer. This can be avoided by using a time delayed execution of the section (phase) which allows a more constant total cycle time to be achieved.
After the program is started, the execution of the 1st Timer event section is delayed by 1 second!

Defining the Scan Rate

Description

Using the entries **Time Base** and **Interval**, it is possible to define nearly any scan rate for a timer event section.
Selectable time base:
- 10 ms
- 100 ms
- 1 s

Interval values:
- whole number multiples of the time base in the range from 1 to 1023

scan rate = interval * time base
- can be defined in range from 10 ms to 1023 s
- can be set in steps that correspond to the selected time base

Examples

Example 1:
Required scan rate = 0.310 s (310 ms)

Scan rate (ms)	Interval	Time base (ms)
310	31	10

For a scan rate of 0.31 s, define a value of 31 for the interval
Example 2:
Required scan rate = 0.3 s (300 ms)

Scan rate (ms)	Interval	Time base (ms)
300	30	10
300	3	100

For a scan rate of 0.3 s, a value of 30 or 3 can be defined for the interval depending on the selected time base.
Basically, any resulting setting can be selected. However, the possible setting should take the phase into consideration (see *Defining the Phase, p. 1049*).

Defining the Phase

Description

To prevent several timer event sections from being processed in the same cycle, they can be assigned different phase values.
Phase values:
- whole number multiples of the time base
- Range from 0 to interval -1, max. 1022

Examples

Example 3:

Scan rate (ms)	Interval	Time base (ms)	Phases	Max. number of time displaced timer event sections
300	30	10	0...29	30
300	3	100	0...2	3

Example 4:
Several timer event sections with 300 ms scan rate (see Example 3)

Interval	Time base (ms)	Phase
30	10	0 (defined for all sections)

Result: All sections are processed in the same cycle, i.e. the program cycle time increased by the sum of the runtimes for all sections to be executed every 300 ms!

Example 5:
3 to 16 timer event sections with 300 ms scan rate (see Example 3)
1. Section

Interval	Time base (ms)	Phase
30	10	0

2. Section

Interval	Time base (ms)	Phase
30	10	1

3. Section

Interval	Time base (ms)	Phase
30	10	2

... Section

Interval	Time base (ms)	Phase
30	10	...

16. Section

Interval	Time base (ms)	Phase
30	10	15

Result: After the program is started, the 1st execution takes place for the
- 2nd timer event section (phase 1) after 1s+1*10 ms = **1s+10 ms**
- 3rd timer event section (phase 2) after 1s+2*10 ms = **1s+20 ms**
- ...
- 16. timer event section (phase 15) after 1s+15*10 ms = **1s+150 ms**
- 1st timer event section (phase 0) after 1s+30*10 ms = **1s+300 ms**

The second execution after the program start takes place for the
- 2nd timer event section (phase 1) after 1s+300 ms+1*10 ms = **1s+310 ms**
- 3rd timer event section (phase 2) after 1s+300 ms+2*10 ms = **1s+320 ms**
- ...
- 16th timer event section (phase 15) after 1s+300 ms+15*10 ms = **1s+450 ms**
- 1st timer event section (phase 0) after 1s+300 ms+30*10 ms = **1s+600 ms**

Each further execution of a timer event section takes place after exactly 300 ms, i.e. the runtimes of the (max. 16) timer event sections are distributed over (max. 30 selectable) different program cycles.

Explanation of example 5

If a time base of 10 ms is selected in example 5 (phase 0...29), the maximum number of 16 timer event sections can be executed using time displacement. That means a time displacement between 10 and 300 ms per section can be selected in steps of 10 ms. Each of the maximum possible 16 timer event sections is executed in a different program cycle. Every 10...20 ms, a program cycle extended by the execution duration of a timer event section occurs.

If a time base of 100 ms is selected in example 5 (phase 0, 1 and 2), the execution of the max. 16 possible timer event sections is distributed over only a max. of 3 time displaced program cycles. In these 3 program cycles, several timer event sections must be executed one after the other. Every 100 ms, a program cycle extended by the sum of the execution times of several timer event sections occurs!
The user should use the time base and phase to guarantee evenly distributed timer event sections (see timing diagram).)

Execution Order

Description

When creating the first timer event section, a new **Timer Events** group is created automatically in which the new section appears. This group is placed in before the cyclic sections and after the **I/O Events** group in the project browser. The next timer event section to be created is automatically always placed at the end in the group **Timer Events**.

Priority

Timer event sections do not have priorities set between them, i.e. they cannot be interrupted by another timer event section.
If several timer event sections are triggered at the same time in a program cycle, they are executed consecutively according to the order in which they were created. However, an *I/O Event Sections, p. 1060* has a higher priority and therefore interrupts a timer event section. The interrupted timer event section is only returned to after the execution of the I/O section is completed.

Operating System

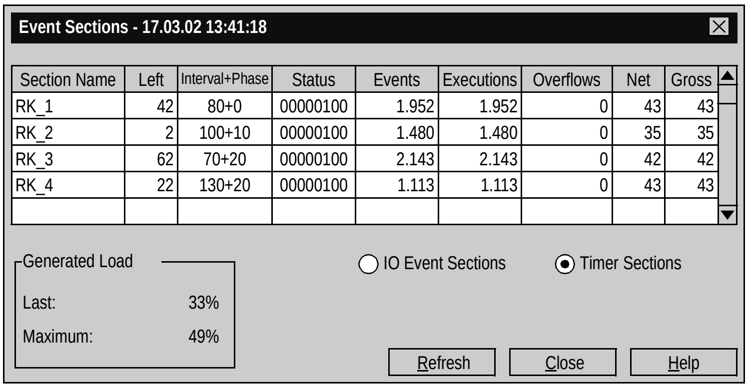
Setting parameters according to runtime aspects

Take the following into consideration when setting parameters for timer event sections:

1. The runtime for a timer event section can be a maximum of 20 ms (see also *Runtime Error, p. 1053*).
2. The scan rate (interval * time base) must be larger than the runtime for the timer event section.
3. Select the phase in a program cycle so that only one timer event section is executed whenever possible.
4. Take note of the distance between phases for time base 10 ms and a runtime for the timer event section > 10 ms! (Select a distance between phases >1 to prevent runtime overlaps.)
5. For optimal processor load:
   The execution of all timer event sections must be evenly distributed by selecting a suitable phase using the time for the scan rates.
6. Sufficient time must be remaining for the execution of the cyclic sections so that the cyclic I/O is handled in acceptable intervals!
7. Execution of inputs and outputs as direct I/O modules IMIO_IN and IMIO_OUT in the timer event section. For example if the uneven intervals for the cyclic I/O are not sufficient to create a control loop.
8. Create a timing diagram (see *Examples for Parameterization, p. 1054*):
   This makes it possible to determine the optimal phase, as well as the actual time intervals for the cyclic I/O.
9. Do not create all control loops as timer event sections:
   Control loops can also be programmed as cyclic sections using the necessary high-speed CPUs and the SAMPLETM module!

Runtime Error

Any errors that occur when processing the program, e.g. runtime is exceeded, overflow, etc. are shown in a table in the **Event Sections** dialog box (in the **Online** main menu).
The following table is based on *Example 4: Control loops with different scan rates, with phases, constant cycle time, p. 1058* in section "Examples for Setting Parameters":


Event Sections - 17.03.02 13:41:18

Section Name	Left	Interval+Phase	Status	Events	Executions	Overflows	Net	Gross
RK_1	42	80+0	00000100	1.952	1.952	0	43	43
RK_2	2	100+10	00000100	1.480	1.480	0	35	35
RK_3	62	70+20	00000100	2.143	2.143	0	42	42
RK_4	22	130+20	00000100	1.113	1.113	0	43	43

Runtime exceeded for a timer event section

If the runtime for a timer event section is >20 ms, the following process is carried out:
1. In the **Event Sections** table, status bit 2 is set (watchdog timer has expired)
2. The timer event section is disabled.
Carry out the following steps to detect when the runtime is exceeded in timer event sections:

Step	Action
1	Using the ISECT_STAT function block.
2	Activate the mode to display enable states in the project browser. Then the symbols for the disabled sections are marked red.
3	Call the status table in **Online** → **Event Sections**.

Carry out the following steps if an timer event section is disabled:

Step	Action
1	Reduce the runtime of the timer event section to <20 ms.
2	Enabling a timer event section Examples: • In the project browser, activate the **Switch enable state** command. • Programming: 0 -> sectname.disable **Caution:** If the runtime error still occurs, the timer event section is not processed even though the section symbol is marked green in the project browser!
3	After enabling the timer event section, the RESET function block parameter on the ISECT_STAT EFB must be set. Only then are current values shown in the status table (**Online** → **Event Sections**).

Examples for Parameterization

Introduction

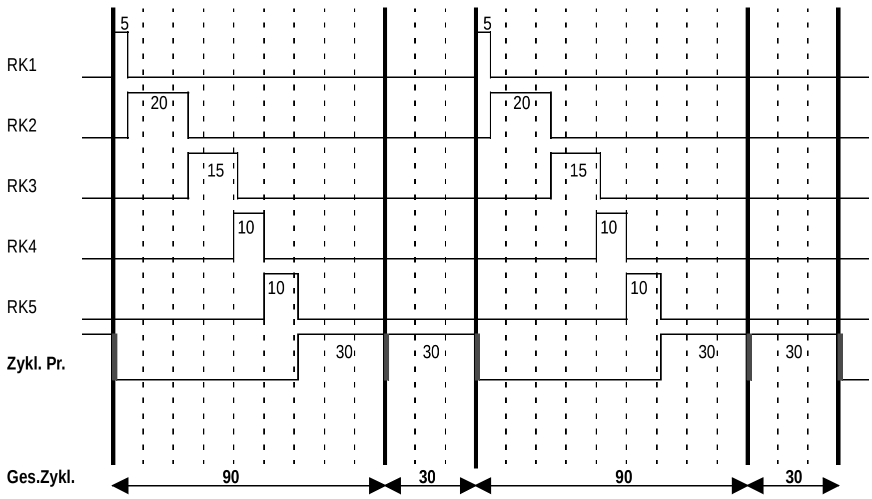
The examples shown here with the values given represent theoretical information and should mainly be used to clarify the effects of various phase values and distances between phases on the entire system. With (preset) values determined using timing diagrams and tests, the user can establish a predictive view and reach an optimal distribution of the timer event sections and prevent runtime overflows.

Example 1: Control loops with the same scan rates, all phases = 0

Preset values:

Preset values	Scan rate	Time base	Interval	Runtime	Phase
Cyclic Program				30	
RK 1	120 ms	10 ms	12	5	0
RK 2	120 ms	10 ms	12	<20	0
RK 3	120 ms	10 ms	12	15	0
RK 4	120 ms	10 ms	12	10	0
RK 5	120 ms	10 ms	12	10	0

Timing diagram (times in ms)



RK Control loop

Cyclic Program Cyclic Program

Total Cycle Time Total Cycle Time

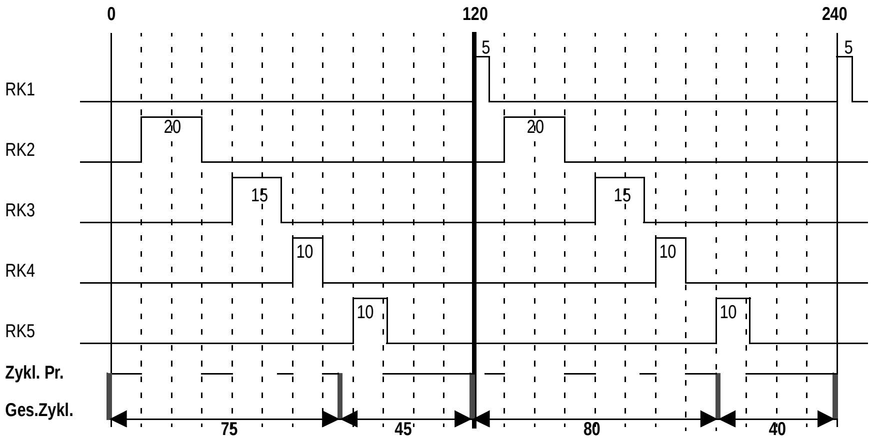
Note: The total cycle time switches between 90 and 30 ms.

Example 2: Control loops with the same scan rates, with phases, minimum distance between phases

Preset values:

Preset values	Scan rate	Time base	Interval	Runtime	Minimum distance between phases	Phase
Cyclic Program				30		
RK 1	120 ms	10 ms	12	5		0
RK 2	120 ms	10 ms	12	<20	5<10 ms->+1	1
RK 3	120 ms	10 ms	12	15	20<30 ms->+3	4
RK 4	120 ms	10 ms	12	10	15<20 ms->+2	6
RK 5	120 ms	10 ms	12	10	10<20 ms->+2	8

Timing diagram (times in ms)



RK Control loop

Cyclic Program Cyclic Program

Total Cycle Time Total Cycle Time

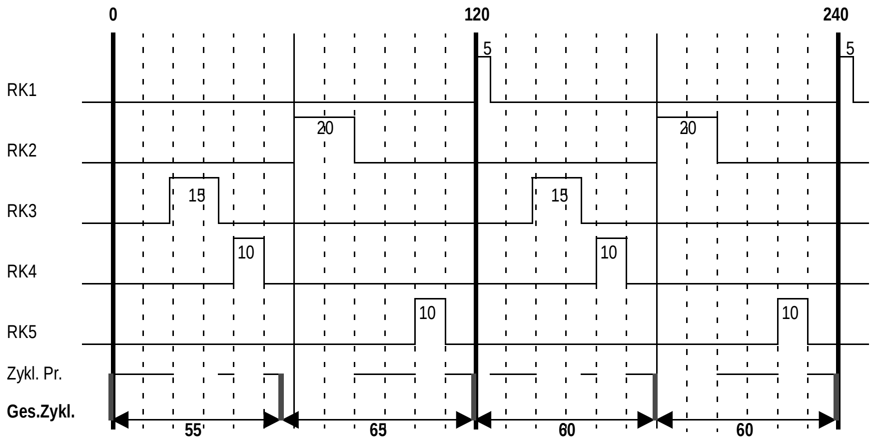
Note: The total cycle time (except for the first cycle) switches between 80 and 40 ms.

Example 3: Control loops with the same scan rates, with phases, constant cycle time

Preset values:

Preset values	Scan rate	Time base	Interval	Runtime	Phase
Cyclic Program				30	
RK 1	120 ms	10 ms	12	5	0
RK 2	120 ms	10 ms	12	<20	6
RK 3	120 ms	10 ms	12	15	2
RK 4	120 ms	10 ms	12	10	4
RK 5	120 ms	10 ms	12	10	10

Timing diagram (times in ms)



RK Control loop

Cyclic Program Cyclic Program

Total Cycle Time Total Cycle Time

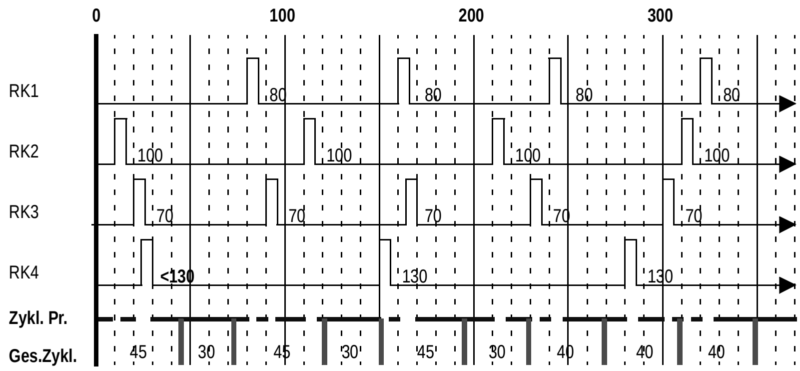
Note: The total cycle time (except for the first cycle) is always 60 ms.

Example 4: Control loops with different scan rates, with phases, constant cycle time

Preset values:

Preset values	Scan rate	Time base	Interval	Runtime	Phase
Cyclic Program				30	
RK 1	80 ms	10 ms	8	5	0
RK 2	100 ms	10 ms	10	5	1
RK 3	70 ms	10 ms	7	5	2
RK 4	130 ms	10 ms	13	5	2

Timing diagram (times in ms)



RK Control loop

Cyclic Program Cyclic Program

Total Cycle Time Total Cycle Time

> **Note:** The timing diagram beginning here shows a favorable distribution of the execution of all sections. The cyclic I/O is also handled in predictable intervals. However, overlapping of individual runtimes cannot be ruled out.

R.3 Interrupt section: I/O event section

Introduction

Overview

This chapter contains a description for I/O event sections.

What's in this Section?

This section contains the following topics:

I/O Event Sections

Introduction

An I/O event section is carried out depending on the hardware interrupts of a 140-HLI-340-00.
The 140-HLI-340-00 module is equipped with 16 inputs that can be configured as either fast inputs or interrupt inputs. Only interrupt inputs can trigger the execution of an I/O event section with the edge defined. Parameters must be set for the 140-HLI-340-00 module in the PLC Configuration accordingly.
I/O event sections are created in the same way as cyclic sections using the **File** → **New Section...** menu command. An I/O event section can only be selected when a CPU 140-CPU-434 oder 140-CPU-534 is configured in the Configurator. The CPU Hardware 140 CPU 434 12 A or 140 CPU 534 14 A is required to execute a program with I/O event sections!
A maximum of 64 I/O event sections can be created. The required hardware interrupts can be created by more than 4 HLI modules.
When creating the first I/O event section, a new **I/O Events** group is created automatically in which the new section appears. The **I/O Events** group appears first in the project browser, i.e. before the **Timer Events** groups and the cyclic sections. Every additional I/O event section to be created is automatically placed in the **I/O Events** group according to their priority (from top to bottom).
An execution order has no relevance for I/O event sections, since these sections can only be processed when a hardware interrupt occurs.
An I/O event section can only be interrupted by hardware interrupts with higher *Priority, p. 1061*.
I/O event sections are programmed principally as with cyclic sections (see *Step 3: Creating the User Program, p. 57*), only the selection of existing EFBs is limited.
Blocks (EFB) not available in Timer event sections:
- **F_TRIG**, **R_TRIG** (IEC library, group: Edge Detection)
- **TOF**, **TON**, **TP** (IEC library, group: Timer)
- **ERR2HMI**, **ERRMSG** (DIAGNO library, group: Diag View)
- **ACT_DIA**, **DYN_DIA**, **GRP_DIA**, **LOCK_DIA**, **PRE_DIA**, **REA_DIA** (DIAGNO library, group: Diagnostics)
- **XACT**, **XACT_DIA**, **XDYN_DIA**, **XGRP_DIA**, **XLOCK_DIA**, **XLOCK**, **XPRE_DIA**, **XREA_DIA**, (DIAGNO library, group: Extended)

I/O event section parameters are set in the **Section Properties for I/O Event Sections** dialog box using the **Slot** and **Input Pin** parameters. The **Slot** entry defines the slot on the local backplane when the 140-HLI-340-00 module is positioned for the triggered interrupt. The **Input Pin** defines pin number (1 to 16) for the 140-HLI-340-00 inputs that trigger the section processing.

Priority

Description

I/O event sections have priorities set between them. An active I/O event section can be interrupted by an I/O event section with higher priority. The interrupted section is continued after the section with higher priority has been processed.
If other interrupts with lower priority occur while processing an I/O event section, the active I/O event section is not interrupted. However, these interrupt signals are saved and the respective sections are processed according to their priority when the active I/O event section is complete. If an interrupt which is saved but not yet processed occurs again, the second interrupt is lost . The overflow counter is incremented (see the table in the **Event Sections** dialog box in the **Online** main menu).
The priority of an I/O event section is determined by the position of an input pin on the 140-HLI-340-00 module in the local backplane. Therefore:
The lower the slot address and the lower the pin number, the higher the priority. The slot and the input pin number of an I/O event section is assigned in the **Section Properties for I/O Event Sections** dialog box.
Example 1:

Priority	Slot	Input Pin
Higher	1	5
Lower	6	1

Example 2:

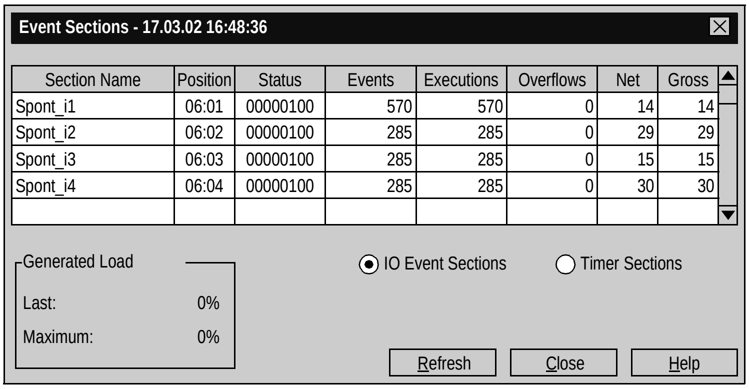
Priority	Slot	Input Pin
Higher	3	5
Lower	3	6

Note: I/O event sections can be interrupted over several priority levels (interrupt in interrupt), therefore the total cycle can be greatly increased.

Runtime Error

Description

Any errors that occur when processing the program, e.g. runtime is exceeded, overflow, etc. are shown in a table in the **Event Sections** dialog box (in the **Online** main menu).
Table in the dialog box **Event Sections**:



Section Name	Position	Status	Events	Executions	Overflows	Net	Gross
Spont_i1	06:01	00000100	570	570	0	14	14
Spont_i2	06:02	00000100	285	285	0	29	29
Spont_i3	06:03	00000100	285	285	0	15	15
Spont_i4	06:04	00000100	285	285	0	30	30

Runtime exceeded for an I/O event section

If the runtime for a timer event section is >20 ms, the following process is carried out:
1. In the **Event Sections** table, status bit 2 is set (watchdog timer has expired)
2. The I/O event section is disabled.
Carry out the following steps to detect when the runtime is exceeded in I/O event sections:

Step	Action
1	Using the ISECT_STAT function block.
2	Activate the mode to display enable states in the project browser. Then the symbols for the disabled sections are marked red.
3	Call the status table in **Online** → **Event Sections**.

Carry out the following steps if an I/O event section is disabled:

Step	Action
1	Reduce the runtime of the I/O event section to <20 ms.
2	Enable the I/O event section. Examples: • In the project browser, activate the **Switch enable state** command. • Programming: 0 -> sectname.disable **Caution:** If the runtime error still occurs, the I/O event section is not processed even though the section symbol is marked green in the project browser!
3	After enabling the I/O event section, the RESET function block parameter on the ISECT_STAT EFB must be set. Only then are current values shown in the status table (**Online** → **Event Sections**).

R.4 Modules for interrupt sections

EFBs for Interrupt Sections

EFBs to disable and enable interrupt sections

The following function blocks are available:
- ISECT_OFF
  - The ISECT_OFF block can be used to disable a specific I/O event section or Timer event section, i.e. the interrupt has no effect on this Interrupt section. The name of the section to be disabled is defined by the SECT_CTRL data type variable entered at the input. This variable is automatically created for each section.
- ISECT_ON
  - The ISECT_ON block can be used to enable a specific I/O event section or Timer event section, i.e. the interrupt has effect on this Interrupt section again. The name of the section to be enabled is defined by the SECT_CTRL data type variable entered at the input. This variable is automatically created for each section.
- I_LOCK
  - The I_LOCK block is used to disable all I/O event sections or Timer event sections, i.e. the interrupts have no effect on Interrupt sections.
- I_UNLOCK
  - The I_UNLOCK block is used to enable all I/O event sections or Timer event sections, i.e. the interrupts have effect on the respective Interrupt sections.

Other EFBs for Interrupt sections

The following function blocks are available:
- ISECT_STAT
  - With the ISECT_STAT block, the status (see **Event Sections**dialog box) of a section can be read and evaluated by the program.
- I_MOVE
  - The I_MOVE block prevents an interruption of value assignments from an input to an output by an interrupt. This means the processing of an I_MOVE is not interrupted by an interrupt. This enables data consistency between an input and an output if the variable is used both in cyclic as well as in interrupt sections. The MOVE block has the same function, but value assignment is not interrupt protected (for further details see the block description).

Automatic Connection to the PLC

S

At a Glance

Overview

This chapter is a description of both methods of automatically connecting with a PLC.

What's in this Chapter?

This chapter contains the following topics:

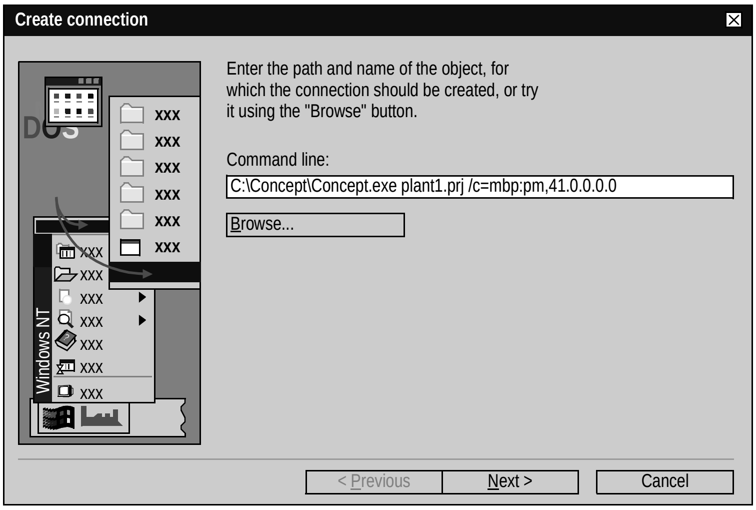
Automatic Connection with Command Line Parameters (Modbus, Modbus +, TCP/IP)

At a Glance

You can make a connection with any PLC automatically by using the command line of the Windows dialog **Create Shortcut** and entering the Modbus Plus Routing path.

> **Note:** If, in dialog **Common Preferences** option **Connect to controller at Startup** (**Options** → **Preferences** → **Common...**) is activated then the extended parameters for the connection to the PLC are priority.

The command line parameters for automatically connecting are added to the end of the command line parameters for the project symbol (see following image).
Dialog box: **Create connection**



Creating an Automatic Connection with Command Line Parameters

The procedure for connecting automatically using command line parameters is as follows:

Step	Action
1	Go to **Start** → **Settings** → **Taskbar...** and open the dialog **Taskbar Properties**.
2	In the register **Start Menu Programs**/**Expanded** (Win2000), select the **Add...** command button.
3	In the **Create Shortcut** dialog box, select the **Browse...** command button.
4	In the **Browse** dialog box, go to the Concept installation path and double-click on the file **CONCEPT.EXE**. **Reaction:** The **Browse** dialog box is closed and the file **CONCEPT.EXE** including the path is entered in the **command line:** E.g. **C:\CONCEPT\CONCEPT.EXE**.
5	Now, add the project name of the project in the command line, e.g. **C:\CONCEPT\CONCEPT.EXE PLANT1.PRJ**.
6	Now add the Modbus Plus-Routing path of the PLC to the command line, e.g. **C:\CONCEPT\CONCEPT.EXE PLANT1.PRJ /c=mbp : 41.0.0.0.0** and confirm your entries with the **Next >**button. **Note:** A definition of the various command line parameters can be found in section *Definition of Command Line Parameters, p. 1070*.
7	In the **Select program group** dialog box, select an existing program group for the symbol or create a new one using **New folder...** . Confirm the entry using the **Next >**command button.
8	In the **Select program designation** dialog box, select the project name and confirm using the **Finish** command button.
9	Close the **Taskbar Properties** dialog box with **OK**. **Reaction:** The properties dialog box is closed and the project symbol is available in the start menu of the folder you selected.
10	Open the folder with the project symbol in the Start menu. Select the project symbol and click the right mouse button. **Reaction:** A menu window is opened.
11	Select the **Properties** command button. **Reaction:** The **"Project Symbol Name" Properties** dialog box is opened.
12	Go to the **Connection** register and complete the command line **Working directory**/**Target** (Win2000) with the name of the project directory, e.g. **C:\CONCEPT\PROJECTS**. Confirm the entry using the **Apply** command button.
13	Then exit the dialog box by selecting **OK**.
14	Open the project by clicking on the project symbol. **Reaction:** Concept reads the defined Modbus Plus Routing path and automatically creates a connection to the PLC.

Definition of Command Line Parameters

The command line parameters contain the PLC address and the protocol type (Modbus, Modbus Plus, TCP/IP).

Command line parameters for Modbus:

Protocol type	Command line parameters	Meaning
Modbus	`/c=[x,]mb:p[, m]`	**x** - Serial connection (COM): 1 - 4, optional, default is 1 **p** - PLC address: 0 - 255 **m** - Modus for device communication: RTU/ASCII, optional, default is RTU **Note:** The settings (Baud Rate, Data Bits, Parity, Stop Bits) for the serial connection (COM) are made in the **Connect to PLC** dialog box.

Example:

```
c:\concept\concept.exe plant1.prj /c=2,mb:001,ASCII
```

Command line parameters for Modbus Plus:

Protocol type	Command line parameters	Meaning
Modbus Plus	`/c=[x,]mbp: n.n.n.n.n`	**x** - Modbus Plus connection: 0 -1, optional, default is 0 **n** - Routing path (PLC address): 0 - 64

Example:

```
c:\concept\concept.exe plant1.prj /c=mbp:41.0.0.0.0
```

Command line parameters for TCP/IP:

Protocol type	Command line parameters	Meaning
TCP/IP	`/c=[x,]mbt:m.m.m.m` `/c=[x,]mbt:HostName`	**x** - Bridge Modbus Plus Index: 0 -255, optional, default is 0 **m** - IP address: 0 - 255 **HostName** - e.g. for the PLCSIM32, the HostName = Localhost

Example:

```
c:\concept\concept.exe plant1.prj /c=mbt:139.158.107.9
c:\concept\concept.exe plant.prj /c=mbt:Localhost
```

Disadvantage

In a large Modbus Plus Network, a separate command line is required for each PLC. If an address changes at any time (e.g. Bridge address the command line parameters must be redone for every programming device that makes access.

Automatic Connection with the CCLaunch Tool (Modbus Plus)

At a Glance

You can use the CCLaunch tool to create a complete Routing path, which then creates a connection to the PLC in the corresponding Modbus Plus segment automatically.
The CCLaunch tool can also only be used to open the assigned project for making changes.
The CCLaunch tool is executed with the CCLAUNCH.EXE file in the Concept directory.

Selection condition

The CCLaunch tool can only be used if a topology file exists and the path is entered in the CCLaunch tool.

Creating an Automatic Connection

Create an ASCII file (topology file) for the automatic connection to the PLC and name it e.g. CCLEXAMP.TXT. Define all Routing paths and segment names of the entire project network in this *.TXT file. Then, copy the file to your server so that every programming device can access it. Compared with a command line parameter entry (See *Automatic Connection with Command Line Parameters (Modbus, Modbus +, TCP/IP), p. 1068*), you have the advantage of only having to change one file when an address (e.g. bridge address) changes in the routing path.
To activate the automatic connection, start the CCLaunch tool. Enter the path of the topology file, the path for the projects and the address of the Modbus Plus adapter one time only. The definitions will then remain until they are changed again by the user. Enter the start segment that must be defined for the programming device (PC). Define the target segment that must be defined for the PLC to which it will be connected as another setting. Then select the PLC that you have defined in the topology file.

Note: For creating the automatic connection, the check box **Autoconnect to PLC** (**Options** → **Preferences** → **Common...**) must be activated.

These entries allow CCLaunch to create a complete Routing path, which then creates a connection to the PLC automatically.

Opening Associated Project

To open the assigned project directly, define the same settings as for the automatic connection. Then activate the check box **Open Associated Project**.

Note: If you only want to open this project, deactivate the check box **Autoconnect to PLC**.

Creating the Topology File (*.TXT)

The topology file (*.TXT) only has to be created one time and it contains the description of the entire Modbus Plus network as well as an option description of the projects assigned to the PLC. These can then be stored centrally on the network/ server.
The topology file (*.TXT) contains the two keywords [Segment] and [Routing]. The definition of the individual segment begins with keyword [Segment]. The definition of each individual Routing path begins with the keyword [Routing].

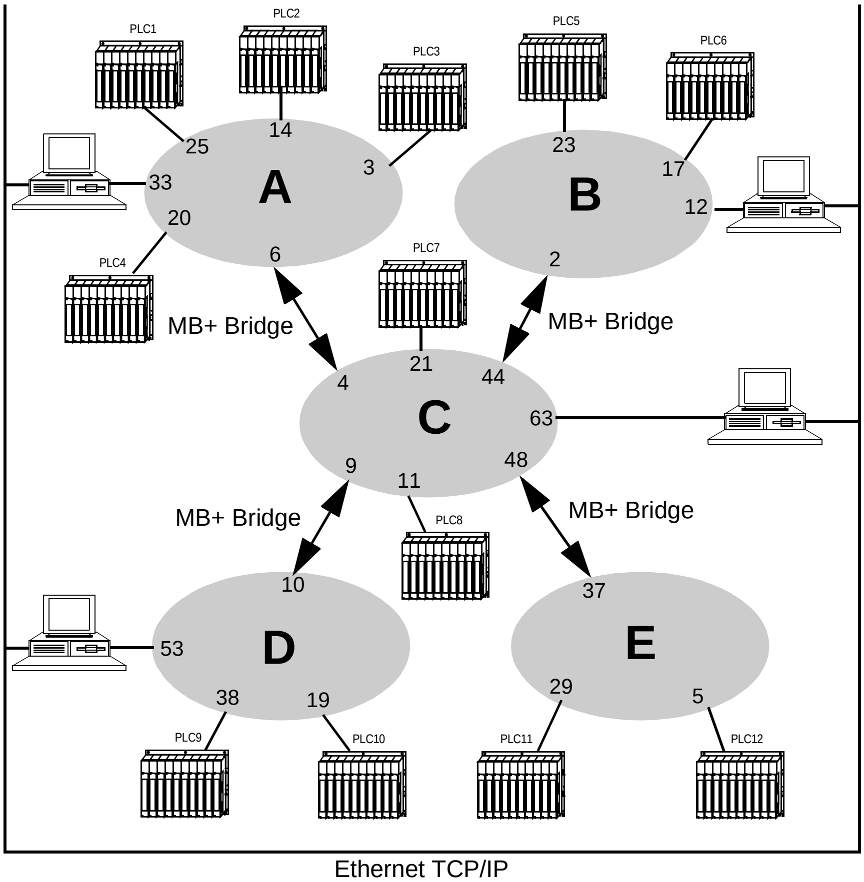
> **Note:** For the definition of the PLC, the PLC name must be unique throughout the entire Modbus Plus network.

Example:

```
[Segment]="Segment name"
"PLC Name"="MB+Address" : "Project name" (optional)
"PLC Name"="MB+Address" : "Project name" (optional)
[Routing]SegmentX="Routing path"
[Routing]SegmentY="Routing path"
```

Example of a Topology File (*.TXT)

Example of a Modbus Plus network with different segments:



A Segment A

B Segment B

C Segment C

D Segment D

E Segment E

Contents of the Topology File (*.TXT)

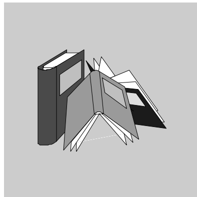
```
[Segment]=SegmentA
PLC1 = 25 : Project 1
PLC2 = 14 : Project 2
PLC3 = 3  : Project 3
PLC4 = 20 : Project 4
[Routing] SegmentB=6.44
[Routing] SegmentC=6
[Routing] SegmentD=6.9
[Routing] SegmentE=6.48
[Segment]=SegmentB
PLC5 = 23 : Project 5
PLC6 = 17 : Project 6
[Routing] SegmentA=2.4
[Routing] SegmentC=2
[Routing] SegmentD=2.9
[Routing] SegmentE=2.48
[Segment]=SegmentC
PLC7 = 21 : Project 7
PLC8 = 11 : Project 8
[Routing] SegmentA=4
[Routing] SegmentB=44
[Routing] SegmentD=9
[Routing] SegmentE=48
[Segment]=SegmentD
PLC9 = 38 : Project 9
PLC10 = 19: Project 10
[Routing] SegmentA=10.4
[Routing] SegmentB=10.44
[Routing] SegmentC=10
[Routing] SegmentE=10.48
[Segment]=SegmentE
PLC11 = 21: Project 11
PLC12 = 11: Project 12
[Routing] SegmentA=37.4
[Routing] SegmentB=37.44
[Routing] SegmentC=37
[Routing] SegmentD=37.9
```

Editing with the CCLaunch Tool

After creating the topology file (*.TXT), execute the following steps in the CCLaunch tool for the automatic connection:

Step	Action
1	Double click on the CCLAUNCH.EXE file in the Concept directory. **Reaction:** The CCLaunch tool is started.
2	Go to the **Settings** tab and define the path for the topology file (*.TXT) and the project path. **Note:** This is normally only defined once since this path should not have to be changed. This means that these settings only have to be made once and they remain saved until you change them for whatever reason. Example: **Topology file:** C:\CONCEPT\CONNECT\CCLEXAMP.TXT **Path for projects:** C:\CONCEPT\TESTPRJ\
3	Select the hardware address for the network connection in the **Modbus+ Port** field. **Note:** Whether this is port 0 or port 1 can be determined from the Windows system settings.
4	Select tab **Select PLC**, and enter the start segment, the target segment and the PLC that you want to connect with for the Routing path. Example: **Start Segment:** SegmentB **Dest. Segment:** SegmentE **PLC:** PLC8 In this example, the programming device is in segment B and should create a connection to the PLC with the name "PLC8" in segment E.
5	Go to the **Start Options** area and activate the check box **Autoconnect to PLC**.
6	Press the **Start Concept** button. **Reaction:** Concept reads the created Routing path and automatically creates a connection to the PLC.

Glossary



A

Active Window

The window, which is currently selected. Only one window can be active at any given time. When a window is active, the color of the title bar changes, so that it is distinguishable from the other windows. Unselected windows are inactive.

Actual Parameters

Current connected Input / Output Parameters.

Addresses

(Direct) addresses are memory ranges in the PLC. They are located in the State RAM and can be assigned Input/Output modules.
The display/entry of direct addresses is possible in the following formats:
- Standard Format (400001)
- Separator Format (4:00001)
- Compact format (4:1)
- IEC Format (QW1)

ANL_IN

ANL_IN stands for the "Analog Input" data type and is used when processing analog values. The 3x-References for the configured analog input module, which were specified in the I/O component list, are automatically assigned data types and should therefore only be occupied with Unlocated Variables.

ANL_OUT

ANL_OUT stands for the "Analog Output" data type and is used when processing analog values. The 4x-References for the configured analog input module, which were specified in the I/O component list, are automatically assigned data types and should therefore only be occupied with Unlocated Variables.

ANY

In the above version "ANY" covers the BOOL, BYTE, DINT, INT, REAL, UDINT, UINT, TIME and WORD elementary data types and related Derived Data Types.

ANY_BIT In the above version "ANY_BIT" covers the BOOL, BYTE and WORD data types.

ANY_ELEM In the above version "ANY_ELEM" covers the BOOL, BYTE, DINT, INT, REAL, UDINT, UINT, TIME and WORD data types.

ANY_INT In the above version "ANY_INT" covers the DINT, INT, UDINT and UINT data types.

ANY_NUM In the above version "ANY_NUM" covers the DINT, INT, REAL, UDINT and UINT data types.

ANY_REAL In the above version "ANY_REAL" covers the REAL data type.

Application Window The window containing the workspace, menu bar and the tool bar for the application program. The name of the application program appears in the title bar. An application window can contain several Document windows. In Concept the application window corresponds to a Project.

Argument Synonymous with Actual parameters.

ASCII-Mode The ASCII (American Standard Code for Information Interchange) mode is used to communicate with various host devices. ASCII works with 7 data bits.

Atrium The PC based Controller is located on a standard AT board, and can be operated within a host computer in an ISA bus slot. The module has a motherboard (requiring SA85 driver) with two slots for PC104 daughter-boards. In this way, one PC104 daughter-board is used as a CPU and the other as the INTERBUS controller.

B

Backup file (Concept-EFB) The backup file is a copy of the last Source coding file. The name of this backup file is "backup??.c" (this is assuming that you never have more than 100 copies of the source coding file). The first backup file has the name "backup00.c". If you have made alterations to the Definitions file, which do not cause any changes to the EFB interface, the generation of a backup file can be stopped by editing the source coding file (**Objects** → **Source**). If a backup file is created, the source file can be entered as the name.

Base 16 literals

Base 16 literals are used to input whole number values into the hexadecimalsystem. The base must be denoted using the prefix 16#. The values can not have any signs (+/-). Single underscores (_) between numbers are not significant.

Example
16#F_F or 16#FF (decimal 255)
16#E_0 or 16#E0 (decimal 224)

Base 2 literals

Base 2 literals are used to input whole number values into the dualsystem. The base must be denoted using the prefix 2#. The values can not have any signs (+/-). Single underscores (_) between numbers are not significant.

Example
2#1111_1111 or 2#11111111 (decimal 255)
2#1110_0000 or 2#11100000 (decimal 224)

Base 8 literals

Base 8 literals are used to input whole number values into the octosystem. The base must be denoted using the prefix 8#. The values can not have any signs (+/-). Single underscores (_) between numbers are not significant.

Example
8#3_77 or 8#377 (decimal 255)
8#34_0 or 8#340 (decimal 224)

Binary Connections

Connections between FFB outputs and inputs with the data type BOOL.

Bitsequence

A data element, which consists of one or more bits.

BOOL

BOOL stands for the data type "boolean". The length of the data element is 1 bit (occupies 1 byte in the memory). The value range for the variables of this data type is 0 (FALSE) and 1 (TRUE).

Bridge

A bridge is a device, which connects networks. It enables communication between nodes on two networks. Each network has its own token rotation sequence - the token is not transmitted via the bridge.

BYTE

BYTE stands for the data type "bit sequence 8". Entries are made as base 2 literal, base 8 literal or base 16 literal. The length of the data element is 8 bits. A numerical value range can not be assigned to this data type.

C

Clipboard The clipboard is a temporary memory for cut or copied objects. These objects can be entered in sections. The contents of the clipboard are overwritten with each new cut or copy.

Coil A coil is a LD element which transfers the status of the horizontal short on its left side, unchanged, to the horizontal short on its right side. In doing this, the status is saved in the relevant variable/direct address.

Compact format (4:1) The first digit (the Reference) is separated from the address that follows by a colon (:) where the leading zeros are not specified.

Constants Constants are Unlocated variables, which are allocated a value that cannot be modified by the logic program (write protected).

Contact A contact is a LD element, which transfers a status on the horizontal link to its right side. This status comes from the boolean AND link of the status of the horizontal link on the left side, with the status of the relevant variable/direct address. A contact does not change the value of the relevant variable/direct address.

D

Data transfer settings Settings which determine how information is transferred from your programming device to the PLC.

Data Types The overview shows the data type hierarchy, as used for inputs and outputs of functions and function blocks. Generic data types are denoted using the prefix "ANY".

- ANY_ELEM
  - ANY_NUM
    ANY_REAL (REAL)
    ANY_INT (DINT, INT, UDINT, UINT)
  - ANY_BIT (BOOL, BYTE, WORD)
  - TIME
- System Data types (IEC Extensions)
- Derived (from "ANY" data types)

DCP I/O drop A remote network with a super-ordinate PLC can be controlled using a Distributed Control Processor (D908). When using a D908 with remote PLC, the super-ordinate PLC considers the remote PLC as a remote I/O drop. The D908 and the remote PLC communicate via the system bus, whereby a high performance is achieved with minimum effect on the cycle time. The data exchange between the D908 and the super-ordinate PLC takes place via the remote I/O bus at 1.5Mb per second. A super-ordinate PLC can support up to 31 D908 processors (addresses 2-32).

DDE (Dynamic Data Exchange) The DDE interface enables a dynamic data exchange between two programs in Windows. The user can also use the DDE interface in the extended monitor to invoke their own display applications. With this interface, the user (i.e. the DDE client) can not only read data from the extended monitor (DDE server), but also write data to the PLC via the server. The user can therefore alter data directly in the PLC, while monitoring and analyzing results. When using this interface, the user can create their own "Graphic Tool", "Face Plate" or "Tuning Tool" and integrate into the system. The tools can be written in any language, i.e. Visual Basic, Visual C++, which supports DDE. The tools are invoked, when the user presses one of the buttons in the Extended Monitor dialog field. Concept Graphic Tool: Configuration signals can be displayed as a timing diagram using the DDE connection between Concept and Concept Graphic Tool.

Declaration Mechanism for specifying the definition of a language element. A declaration usually covers the connection of an identifier to a language element and the assignment of attributes such as data types and algorithms.

Definitions file (Concept-EFB) The definitions file contains general descriptive information on the selected EFB and its formal parameters.

Derived Data Type Derived data types are data types, which are derived from Elementary Data Types and/or other derived data types. The definition of derived data types is found in the Concept data type editor.
A distinction is made between global data types and local data types.

Derived Function Block (DFB) A derived function block represents the invocation of a derived function block type. Details of the graphic form of the invocation can be found in the "Functional block (instance)". In contrast to the invocation of EFB types, invocations of DFB types are denoted by double vertical lines on the left and right hand side of the rectangular block symbol.
The body of a derived function block type is designed using FBD language, LD language, ST language, IL language, however, this is only the case in the current version of the programming system. Furthermore, derived functions can not yet be defined in the current version.
A distinction is made between local and global DFBs.

Device Address The device address is used to uniquely denote a network device in the routing path. The address is set on the device directly, e.g. using the rotary switch on the back of the modules.

DFB Code The DFB code is the section's DFB code, which can be executed. The size of the DFB code is mainly dependant upon the number of blocks in the section.

DFB instance data The DFB instance data is internal data from the derived function block used in the program.

DINT DINT stands for the data type "double length whole number (double integer)". Entries are made as integer literal, base 2 literal, basis 8 literal or base 16 literal. The length of the data element is 32 bits. The value range for variables of this datatype reaches from -2 exp (31) to 2 exp (31) -1.

Direct Representation A method of displaying variables in the PLC program, from which the assignment to the logical memory can be directly - and indirectly to the physical memory - derived.

Document Window A window within an application window. Several document windows can be open at the same time in an application window. However, only one document window can ever be active. Document windows in Concept are, for example, sections, the message window, the reference data editor and the PLC configuration.

DP (PROFIBUS) DP = Remote Peripheral

Dummy An empty file, which consists of a text heading with general file information, such as author, date of creation, EFB designation etc. The user must complete this dummy file with further entries.

DX Zoom This property enables the user to connect to a programming object, to monitor and, if necessary change, its data value.

E

EFB code The EFB code is the section's EFB code, which can be executed. In addition the used EFBs count in DFBs.

Elementary functions/ function blocks (EFB) Identifier for Functions or Function blocks, whose type definitions are not formulated in one of the IEC languages, i.e. whose body for example can not be modified with the DFB editor (Concept-DFB). EFB types are programmed in "C" and are prepared in a pre-compiled form using libraries.

EN / ENO (Enable / Error signal)

If the value of EN is equal to "0" when the FFB is invoked, the algorithms that are defined by the FFB will not be executed and all outputs keep their previous values. The value of ENO is in this case automatically set to "0". If the value of EN is equal to "1", when the FFB is invoked, the algorithms which are defined by the FFD will be executed. After the error-free execution of these algorithms, the value of ENO is automatically set to "1". If an error occurs during the execution of these algorithms, ENO is automatically set to "0". The output behavior of the FFB is independent of whether the FFBs are invoked without EN/ENO or with EN=1. If the EN/ENO display is switched on, it is imperative that the EN input is switched on. Otherwise, the FFB is not executed. The configuration of EN and ENO is switched on or off in the Block Properties dialog box. The dialog box can be invoked with the **Objects** → **Properties...** menu command or by double-clicking on the FFB.

Error

If an error is recognized during the processing of a FFB or a step (e.g. unauthorized input values or a time error), an error message appears, which can be seen using the **Online** → **Online events...** menu command. For FFBs, the ENO output is now set to "0".

Evaluation

The process, through which a value is transmitted for a Function or for the output of a Function block during Program execution.

F

FFB (Functions/ Function blocks)

Collective term for EFB (elementary functions/function blocks) and DFB (Derived function blocks)

Field variables

A variable, which is allocated a defined derived data type with the key word ARRAY (field). A field is a collection of data elements with the same data type.

FIR Filter

(Finite Impulse Response Filter) a filter with finite impulse answer

Formal parameters

Input / Output parameters, which are used within the logic of a FFB and led out of the FFB as inputs/outputs.

Function (FUNC)

A program organization unit, which supplies an exact data element when processing. a function has no internal status information. Multiple invocations of the same function using the same input parameters always supply the same output values.

Details of the graphic form of the function invocation can be found in the "Functional block (instance)". In contrast to the invocation of the function blocks, function invocations only have a single unnamed output, whose name is the same as the function. In FBD each invocation is denoted by a unique number via the graphic block, this number is automatically generated and can not be altered.

Function block (Instance) (FB)

A function block is a program organization unit, which correspondingly calculates the functionality values that were defined in the function block type description, for the outputs and internal variable(s), if it is invoked as a certain instance. All internal variable and output values for a certain function block instance remain from one function block invocation to the next. Multiple invocations of the same function block instance with the same arguments (input parameter values) do not therefore necessarily supply the same output value(s).
Each function block instance is displayed graphically using a rectangular block symbol. The name of the function block type is stated in the top center of the rectangle. The name of the function block instance is also stated at the top, but outside of the rectangle. It is automatically generated when creating an instance, but, depending on the user’s requirements, it can be altered by the user. Inputs are displayed on the left side of the block and outputs are displayed on the right side. The names of the formal input/output parameters are shown inside the rectangle in the corresponding places.
The above description of the graphic display is especially applicable to the function invocation and to DFB invocations. Differences are outlined in the corresponding definitions.

Function Block Dialog (FBD)

One or more sections, which contain graphically displayed networks from Functions, Function blocks and Connections.

Function block type

A language element, consisting of: 1. the definition of a data structure, divided into input, output and internal variables; 2. a set of operations, which are performed with elements of the data structure, when a function block type instance is invoked. This set of operations can either be formulated in one of the IEC languages (DFB type) or in "C" (EFB type). A function block type can be instanced (invoked) several times.

Function Number

The function number is used to uniquely denote a function in a program or DFB. The function number can not be edited and is automatically assigned. The function number is always formed as follows: .n.m

n = section number (current number)
m = Number of the FFB object in the section (current number)

G

Generic Data Type A data type, which stands in place of several other data types.

Generic literals If the literal's data type is not relevant, simply specify the value for the literal. If this is the case, Concept automatically assigns the literal a suitable data type.

Global Data Global data are Unlocated variables.

Global derived data types Global derived data types are available in each Concept project and are occupied in the DFB directory directly under the Concept directory.

Global DFBs Global DFBs are available in each Concept project. The storage of the global DFBs is dependant upon the settings in the CONCEPT.INI file.

Global macros Global macros are available in each Concept project and are occupied in the DFB directory directly under the Concept directory.

Groups (EFBs) Some EFB libraries (e.g. the IEC library) are divided into groups. This facilitates EFB location especially in expansive libraries.

H

Host Computer Hardware and software, which support programming, configuring, testing, operating and error searching in the PLC application as well as in a remote system application, in order to enable source documentation and archiving. The programming device can also be possibly used for the display of the process.

I

I/O Map The I/O and expert modules from the various CPUs are configured in the I/O map.

Icon Graphical representation of different objects in Windows, e.g. drives, application programs and document windows.

IEC 61131-3 International standard: Programmable Logic Controls - Part 3: Programming languages.

IEC Format (QW1) There is an IEC type designation in initial position of the address, followed by the five-figure address.
- %0x12345 = %Q12345
- %1x12345 = %I12345
- %3x12345 = %IW12345
- %4x12345 = %QW12345

IEC name conventions (identifier) An identifier is a sequence of letters, numbers and underscores, which must begin with either a letter or underscore (i.e. the name of a function block type, an instance, a variable or a section). Letters of a national typeface (i.e.: ö,ü, é, õ) can be used, except in project and DFB names.
Underscores are significant in identifiers; e.g. "A_BCD" and "AB_CD" are interpreted as two separate identifiers. Several leading and multiple successive underscores are not allowed.
Identifiers should not contain any spaces. No differentiation is made between upper and lower case, e.g. "ABCD" and "abcd" are interpreted as the same identifier.
Identifiers should not be Keywords.

IEC Program Memory The IEC memory consists of the program code, EFB code, the section data and the DFB instance data.

IIR Filter (Infinite Impulse Response Filter) a filter with infinite impulse answer

Initial step The first step in a sequence. A step must be defined as an initial step for each sequence. The sequence is started with the initial step when first invoked.

Initial value The value, which is allocated to a variable when the program is started. The values are assigned in the form of literals.

Input bits (1x references) The 1/0 status of the input bits is controlled via the process data, which reaches from an input device to the CPU.

Note: The x, which follows the initial reference type number, represents a five-figure storage location in the user data memory, i.e. the reference 100201 signifies an output or marker bit at the address 201 in the State RAM.

Input parameter (Input) Upon invocation of a FFB, this transfers the corresponding argument.

Input words (3x references)

An input word contains information, which originates from an external source and is represented by a 16 bit number. A 3x register can also contain 16 sequential input bits, which were read into the register in binary or BCD (binary coded decimal) format. Note: The x, which follows the initial reference type number, represents a five-figure storage location in the user data memory, i.e. the reference 300201 signifies an input word at the address 201 in the State RAM.

Input/output marker bits (0x references)

An input/output marker bit can be used to control real output data using an output unit of the control system, or to define one or more discrete outputs in the state RAM. Note: The x, which follows the initial reference type number, represents a five-figure storage location in the user data memory, i.e. the reference 000201 signifies an output or marker bit at the address 201 in the State RAM.

Instance Name

An identifier, which belongs to a certain function block instance. The instance name is used to clearly denote a function block within a program organization unit. The instance name is automatically generated, but it can be edited. The instance name must be unique throughout the whole program organization unit, and is not case sensitive. If the name entered already exists, you will be warned and you will have to choose another name. The instance name must comply with the IEC name conventions otherwise an error message appears. The automatically generated instance name is always formed as follows: FBI_n_m

FBI = Function Block Instance
n = section number (current number)
m = Number of the FFB object in the section (current number)

Instancing

Generating an Instance.

Instruction (IL)

Instructions are the "commands" of the IL programming language. Each instruction begins on a new line and is performed by an operator with a modifier if necessary, and if required for the current operation, by one or more operands. If several operands are used, they are separated by commas. A character can come before the instruction, which is then followed by a colon. The commentary must, where available, be the last element of the line.

Instruction (LL984)

When programming electrical controls, the user should implement operation-coded instructions in the form of picture objects, which are divided into a recognizable contact form. The designed program objects are, on a user level, converted to computer usable OP codes during the download process. The OP codes are decoded in the CPU and processed by the firmware functions of the controller in a way that the required control is implemented.

Instruction (ST)

Instructions are the "commands" of the ST programming language. Instructions must be concluded by semicolons. Several instructions can be entered in one line (separated by semicolons).

Instruction list (IL)
IL is a text language according to IEC 1131, which is shown in operations, i.e. conditional or unconditional invocations of Functions blocks and Functions, conditional or unconditional jumps etc. through instructions.

INT
INT stands for the data type "whole number (integer)". Entries are made as integer literal, base 2 literal, basis 8 literal or base 16 literal. The length of the data element is 16 bits. The value range for variables of this datatype reaches from -2 exp (15) to 2 exp (15) -1.

Integer literals
Integer literals are used to input whole number values into the decimalsystem. The values can have a preceding sign (+/-). Single underscores (_) between numbers are not significant.

Example
-12, 0, 123_456, +986

INTERBUS (PCP)
The new INTERBUS (PCP) I/O drop type is entered into the Concept configurator, to allow use of the INTERBUS PCP channel and the INTERBUS process data pre-processing (PDV). This I/O drop type is assigned the INTERBUS switching module 180-CRP-660-01.
The 180-CRP-660-01 differs from the 180-CRP-660-00 only in the fact that it has a clearly larger I/O range in the control state RAM.

Invocation
The process, through which an operation is carried out.

J

Jump
Element of the SFC language. Jumps are used to skip zones in the sequence.

K

Keywords
Keywords are unique combinations of characters, which are used as special syntactical components, as defined in Appendix B of the IEC 1131-3. All keywords which are used in the IEC 1131-3 and therefore in Concept, are listed in Appendix C of the IEC 1131-3. These keywords may not be used for any other purpose, i.e. not as variable names, section names, instance names etc.

L

Ladder Diagram (LD)
Ladder Diagram is a graphic programming dialog according to IEC1131, which is optically oriented to the "rung" of a relay contact plan.

Ladder Logic 984 (LL)
The terms Ladder Logic and Ladder Diagram refer to the word Ladder being executed. In contrast to a circuit diagram, a ladder diagram is used by electrotechnicians to display an electrical circuit (using electrical symbols), which should show the course of events and not the existing wires, which connect the parts with each other. A usual user interface for controlling the actions of automation devices permits a Ladder Diagram interface, so that electrotechnicians do not have to learn new programming languages to be able to implement a control program.
The structure of the actual Ladder Diagram enables the connection of electric elements in such a way that generates a control output, which is dependant upon a logical power flow through used electrical objects, which displays the previously requested condition of a physical electrical device.
In simple form, the user interface is a video display processed by the PLC programming application, which sets up vertical and horizontal grid, in which programming objects are classified. The diagram contains the power grid on the left side, and when connected to activated objects, the power shifts from left to right.

Landscape
Landscape means that when looking at the printed text, the page is wider than it is high.

Language Element
Every basic element in one of the IEC programming languages, e.g. a step in SFC, a function block instance in FBD or the initial value of a variable.

Library
Collection of software objects, which are intended for re-use when programming new projects, or even building new libraries. Examples are the libraries of the Elementary function block types.
EFB libraries can be divided up into Groups.

Link
A control or data flow connection between graphical objects (e.g. steps in the SFC Editor, function blocks in the FBD Editor) within a section, represented graphically as a line.

Literals
Literals are used to provide FFB inputs, and transition conditions etc using direct values. These values can not be overwritten by the program logic (read only). A distinction is made between generic and standardized literals.
Literals are also used to allocate a constant, a value or a variable an initial value. Entries are made as base 2 literal, base 8 literal, basis 16 literal, integer literal, real literal or real literal with exponent.

Local derived data types Local derived data types are only available in a single Concept project and the local DFBs and are placed in the DFB directory under the project directory.

Local DFBs Local DFBs are only available in a single Concept project and are placed in the DFB directory under the project directory.

Local Link The local network is the network, which connects the local nodes with other nodes either directly or through bus repeaters.

Local macros Local macros are only available in a single Concept project and are placed in the DFB directory under the project directory.

Local network nodes The local node is the one, which is currently being configured.

Located variable A state RAM address (reference addresses 0x, 1x, 3x,4x) is allocated to located variables. The value of these variables is saved in the state RAM and can be modified online using the reference data editor. These variables can be addresses using their symbolic names or their reference addresses.

All inputs and outputs of the PLC are connected to the state RAM. The program can only access peripheral signals attached to the PLC via located variables. External access via Modbus or Modbus Plus interfaces of the PLC, e.g. from visualization systems, is also possible via located variables.

M

Macro Macros are created with the help of the Concept DFB software.
Macros are used to duplicate frequently used sections and networks (including their logic, variables and variable declaration).
A distinction is made between local and global macros.

Macros have the following properties:
- Macros can only be created in the FBD and LD programming languages.
- Macros only contain one section.
- Macros can contain a section of any complexity.
- In programming terms, there is no difference between an instanced macro, i.e. a macro inserted into a section and a conventionally created section.
- DFB invocation in a macro
- Declaring variables
- Using macro-specific data structures
- Automatic transfer of the variables declared in the macro.
- Initial value for variables
- Multiple instancing of a macro in the entire program with differing variables
- The name of the section, variable names and data structure names can contain up to 10 different exchange marks (@0 to @9).

MMI Man-Machine-Interface

Multi element variables Variables to which a Derived data type defined with STRUCT or ARRAY is allocated. A distinction is made here between field variables and structured variables.

N

Network A network is the collective switching of devices to a common data path, which then communicate with each other using a common protocol.

Network node A node is a device with an address (1...64) on the Modbus Plus network.

Node Node is a programming cell in a LL984 network. A cell/node consists of a 7x11 matrix, i.e. 7 rows of 11 elements.

O

Operand An operand is a literal, a variable, a function invocation or an expression.

Operator An operator is a symbol for an arithmetic or boolean operation, which is to be carried out.

Output parameter (outputs): A parameter, through which the result(s) of the evaluation of a FFB is/are returned.

Output/marker words (4x references) An output / marker word can be used to save numerical data (binary or decimal) in the state RAM, or to send data from the CPU to an output unit in the control system. Note: The x, which follows the initial reference type number, represents a five-figure storage location in the user data memory, i.e. the reference 400201 signifies a 16 bit output or marker word at the address 201 in the State RAM.

P

Peer CPU The Peer CPU processes the token execution and the data flow between the Modbus Plus network and the PLC user logic.

PLC Memory programmable controller

Portrait Portrait means that the sides are larger than the width when printed.

Print-out Expressions consist of operators and operands.

Program The uppermost program organization unit. A program is closed on a single PLC download.

Program organization unit A function, a function block, or a Program. This term can refer to either a type or an instance.

Program redundancy system (Hot Standby) A redundancy system consists of two identically configured PLC machines, which communicate with one another via redundancy processors. In the case of a breakdown of the primary PLC, the secondary PLC takes over the control check. Under normal conditions, the secondary PLC does not take over the control function, but checks the status information, in order to detect errors.

Project General description for the highest level of a software tree structure, which specifies the super-ordinate project name of a PLC application. After specifying the project name you can save your system configuration and your control program under this name. All data that is created whilst setting up the configuration and program, belongs to this super-ordinate project for this specific automation task.
General description for the complete set of programming and configuration information in the project database, which represents the source code that describes the automation of a system.

Project database The database in the host computer, which contains the configuration information for a project.

Prototype file (Concept-EFB) The prototype file contains all the prototypes of the assigned functions. In addition, if one exists, a type definition of the internal status structure is specified.

R

REAL REAL stands for the data type "floating point number". The entry can be real-literal or real-literal with an exponent. The length of the data element is 32 bits. The value range for variables of this data type extends from +/- 3.402823E+38.

> **Note:** Dependent on the mathematical processor type of the CPU, different ranges within this permissable value range cannot be represented. This applies to values that are approaching ZERO and for values that approach INFINITY. In these cases NAN (**N**ot **A N**umber) or INF (**INF**inite will be displayed in the animation mode instead of a number value.

Real literals Real literals are used to input floating point values into the decimal system. Real literals are denoted by a decimal point. The values can have a preceding sign (+/-). Single underscores (_) between numbers are not significant.

Example
-12.0, 0.0, +0.456, 3.14159_26

Real literals with exponents Real literals with exponents are used to input floating point values into the decimal system. Real literals with exponents are identifiable by a decimal point. The exponent indicates the power of ten, with which the existing number needs to be multiplied in order to obtain the value to be represented. The base can have a preceding negative sign (-). The exponent can have a preceding positive or negative sign (+/-). Single underscores (_) between numbers are not significant. (Only between numbers, not before or after the decimal point and not before or after "E", "E+" or "E-")

Example
-1.34E-12 or -1.34e-12
1.0E+6 or 1.0e+6
1.234E6 or 1.234e6

Reference

Every direct address is a reference that begins with an indicator, which specifies whether it is an input or an output and whether it is a bit or a word. References that begin with the code 6, represent registers in the extended memory of the state RAM.
0x range = Coils
1x range = Discrete inputs
3x range = Input registers
4x range = Output registers
6x range = Register in the extended memory

> **Note:** The x, which follows each initial reference type number, represents a five-figure storage location in the user data memory, i.e. the reference 400201 signifies a 16 bit output or marker word at the address 201 in the State RAM.

Register in the extended memory (6x-reference)

6x references are holding registers in the extended memory of the PLC. They can only be used with LL984 user programs and only with a CPU 213 04 or CPU 424 02.

Remote Network (DIO)

Remote programming in the Modbus Plus network enables maximum performance when transferring data and dispenses of the need for connections. Programming a remote network is simple. Setting up a network does not require any additional ladder logic to be created. All requirements for data transfer are fulfilled via corresponding entries in the Peer Cop Processor.

RIO (Remote I/O)

Remote I/O indicates a physical location of the I/O point controlling devices with regard to the CPU controlling them. Remote inp./outputs are connected to the controlling device via a twisted communication cable.

RTU-Mode

Remote Terminal Unit
The RTU mode is used for communication between the PLC and an IBM compatible personal computer. RTU works with 8 data bits.

Runtime error

Errors, which appear during program processing on the PLC, in SFC objects (e.g. Steps) or FFBs. These are, for example, value range overflows with figures or timing errors with steps.

S

SA85 module The SA85 module is a Modbus Plus adapter for IBM-AT or compatible computers.

Scan A scan consists of reading the inputs, processing the program logic and outputting the outputs.

Section A section can for example be used to describe the mode of functioning of a technological unit such as a motor.
A program or DFB consists of one or more sections. Sections can be programmed with the IEC programming languages FBD and SFC. Only one of the named programming languages may be used within a section at any one time.
Each section has its own document window in Concept. For reasons of clarity, it is however useful to divide a very large section into several small ones. The scroll bar is used for scrolling within a section.

Section Code Section Code is the executable code of a section. The size of the Section Code is mainly dependent upon the number of blocks in the section.

Section Data Section data is the local data in a section such as e.g. literals, connections between blocks, non-connected block inputs and outputs, internal status memory of EFBs.

Note: Data which appears in the DFBs of this section is not section data.

Separator Format (4:00001) The first digit (the reference) is separated from the five figure address that follows by a colon (:).

Sequence language (SFC) The SFC Language Elements enable a PLC program organization unit to be divided up into a number of Steps and Transitions, which are connected using directional Links. A number of actions belong to each step, and transition conditions are attached to each transition.

Serial Connections With serial connections (COM) the information is transferred bit by bit.

Source code file (Concept-EFB) The source code file is a normal C++ source file. After executing the **Library** → **Create files** menu command, this file contains an EFB-code frame, in which you have to enter a specific code for the EFB selected. To do this invoke the **Objects** → **Source** menu command.

Standard Format (400001)
The five figure address comes directly after the first digit (the reference).

Standardized literals
If you would like to manually determine a literal's data type, this may be done using the following construction: 'Data type name'#'value of the literal'.

Example
INT#15 (Data type: integer, value: 15),
BYTE#00001111 (Data type: byte, value: 00001111)
REAL#23.0 (Data type: real, value: 23.0)

To assign the data type REAL, the value may also be specified in the following manner: 23.0.
Entering a comma will automatically assign the data type REAL.

State RAM
The state RAM is the memory space for all variables, which are accessed via References (Direct representation) in the user program. For example, discrete inputs, coils, input registers, and output registers are situated in the state RAM.

Status Bits
For every device with global inputs or specific inp./outputs of Peer Cop data, there is a status bit. If a defined group of data has been successfully transferred within the timeout that has been set, the corresponding status bit is set to 1. If this is not the case, this bit is set to 0 and all the data belonging to this group is deleted (to 0).

Step
SFC-language element: Situation, in which the behavior of a program occurs, regarding its inputs and outputs of those operations which are defined by the actions belonging to the step.

Step name
The step name is used to uniquely denote a step in a program organization unit. The step name is generated automatically, but it can be edited. The step name must be unique within the entire program organization unit, otherwise an error message will appear.
The automatically generated step name is always formed as follows: S_n_m

S = step
n = section number (current number)
m = Number of the step in the section (current number)

Structured text (ST)
ST is a text language according to IEC 1131, in which operations, e.g. invocations of Function blocks and Functions, conditional execution of instructions, repetitions of instructions etc. are represented by instructions.

Structured variables
Variables to which a Derived data type defined with STRUCT (structure) is allocated. A structure is a collection of data elements with generally different data types (elementary data types and/or derived data types).

SY/MAX In Quantum control devices, Concept includes the providing of I/O-map SY/MAX-I/O modules for remote contolling by the Quantum PLC. The SY/MAX remote backplane has a remote I/O adapter in slot 1, which communicates via a Modicon S908 R I/O System. The SY/MAX-I/O modules are executed for you for labelling and inclusion in the I/O map of the Concept configuration.

T

Template file (Concept-EFB) The template file is an ASCII file with layout information for the Concept FBD Editor, and the parameters for code creation.

TIME TIME stands for the data type "time". The entry is time literal. The length of the data element is 32 bits. The value range for variables of this data type extends from 0 to 2exp(32)-1. The unit for the TIME data type is 1 ms.

Time literals Permissable units for times (TIME) are days (D), hours (H), minutes (M), seconds (S) and milliseconds (MS) or combinations of these. The time must be marked with the prefix t#, T#, time# or TIME#. The "overflow" of the unit with the highest value is permissible, e.g. the entry T#25H15M is allowed.

Example
t#14MS, T#14.7S, time#18M, TIME#19.9H, t#20.4D, T#25H15M, time#5D14H12M18S3.5MS

Token The network "token" controls the temporary possession of the transfer right via a single device. The token passes round the devices in a rotating (increasing) address sequence. All devices follow the token rotation and can receive all the possible data that is sent with it.

Total IEC memory The total IEC memory consists of the IEC program memory and the global data.

Traffic Cop The traffic cop is an IO map, which is generated from the user-IO map. The traffic cop is managed in the PLC and in addition to the user IO map, contains e.g. status information on the I/O stations and modules.

Transition The condition, in which the control of one or more predecessor steps passes to one or more successor steps along a directed link.

U

UDEFB

User-defined elementary functions/function blocks
Functions or function blocks, which were created in the C programming language, and which Concept provides in libraries.

UDINT

UDINT stands for the data type "unsigned double integer". Entries are made as integer literal, base 2 literal, basis 8 literal or base 16 literal. The length of the data element is 32 bits. The value range for variables of this data type extends from 0 to 2exp(32)-1.

UINT

UINT stands for the data type "unsigned integer". Entries are made as integer literal, base 2 literal, basis 8 literal or base 16 literal. The length of the data element is 16 bits. The value range for variables of this data type extends from 0 to (2exp 16)-1.

Unlocated variable

Unlocated variables are not allocated a state RAM address. They therefore do not occupy any state RAM addresses. The value of these variables is saved in the internal system and can be changed using the reference data editor. These variables are only addressed using their symbolic names.

Signals requiring no peripheral access, e.g. intermediate results, system tags etc., should be primarily declared as unlocated variables.

V

Variables

Variables are used to exchange data within a section, between several sections and between the program and the PLC.
Variables consist of at least one variable name and one data type.
If a variable is assigned a direct address (reference), it is called a located variable.
If the variable has no direct address assigned to it, it is called an unlocated variable.
If the variable is assigned with a derived data type, it is called a multi element variable.
There are also constants and literals.

W

Warning If a critical status is detected during the processing of a FFB or a step (e.g. critical input values or an exceeded time limit), a warning appears, which can be seen using the **Online** → **Event Viewer...** menu command. For FFBs, the ENO remains set to "1".

WORD WORD stands for the data type "bit sequence 16". Entries are made as base 2 literal, base 8 literal or base 16 literal. The length of the data element is 16 bits. A numerical value range can not be assigned to this data type.

Index

D

E

J

K

L

N

O

P