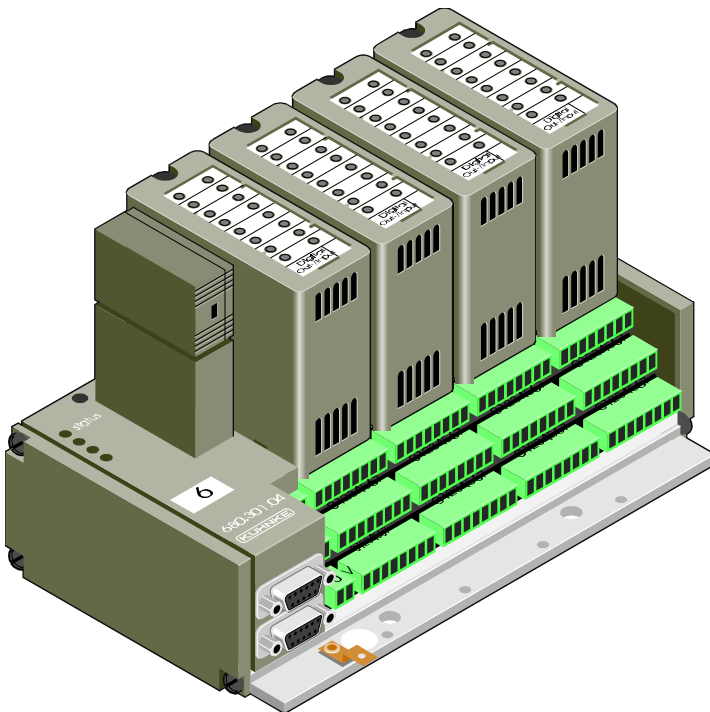


Kuhnke Electronics Instruction Manual

Profi Control 680I and 680I-SL
PLC with integrated PROFIBUS port

E 308 GB

20 August 1998 / 39.912



This manual is primarily intended for the use of the designing engineer, the project planning engineer, and the developing engineer. It does not give any information about delivery possibilities. Data is only given to describe the product and must not be regarded as guaranteed properties in the legal sense. Any claims for damages against us – on whatever legal grounds – are excluded except in instances of deliberate intent or gross negligence on our part.

We reserve the rights for errors, omissions or modifications.

Reproduction even of extracts only with the editor's express and written prior consent.

Table of contents

1. Introduction	1-1
1.1. Moving on: From separate controllers to network systems	1-2
2. Safety and Reliability	2-1
2.1. Target group	2-1
2.2. Reliability	2-1
2.3. Notes	2-2
2.3.1. Danger	2-2
2.3.2. Dangers caused by high contact voltage	2-2
2.3.3 Important information / cross reference	2-2
2.4. Safety	2-3
2.4.1. To be observed during project planning and installation	2-3
2.4.2. To be observed during maintenance and servicing	2-4
2.5. Electromagnetic compatibility	2-5
2.5.1. Definition	2-5
2.5.2. Resistance to interference	2-5
2.5.3. Interference emission	2-6
2.5.4. General notes on installation	2-6
2.5.5. Protection against external electrical influences	2-7
2.5.6. Cable routing and wiring	2-7
2.5.7. Location of installation	2-8
2.5.8. Particular sources of interference	2-8
3. Hardware	3-1
3.1. External appearance	3-1
3.2. Dimensions and installation of basic device	3-2
3.2.1. Installation at the wall	3-3
3.2.2. Installation on carrier rail	3-4

Table of contents

3.3. Power supply	3-5
3.3.1. System power supply	3-6
3.3.1.1. Special feature of devices with 8 module slots	3-6
3.3.2. Power supply of inputs and outputs	3-7
3.3.2.1. Switching off the outputs separately	3-8
3.3.3. Signal line connection	3-9
3.3.4. Connection to earth	3-10
3.4. Communication ports	3-11
3.4.1. V.24 interface (RS 232)	3-12
3.4.2. PROFIBUS port (RS 485)	3-13
3.5. Coding switches	3-14
3.5.1. Coding switch of Profi Control 680I	3-15
3.5.2. Coding switch of Profi Control 680I-SL	3-16
3.6. User program memory	3-17
3.6.1. Function	3-18
3.6.2. RAM memory modules	3-19
3.6.2.1. RAM memory module (32K x 16)	3-20
3.6.2.2. RAM memory module (256K x 16)	3-20
3.6.3. EPROM memory modules	3-21
3.6.3.1. EPROM memory module (32K x 16)	3-22
3.6.3.2. EPROM memory module (128K x 16)	3-22
3.6.4. EPROM-RAM memory modules	3-23
3.6.4.1. EPROM-RAM memory module (32K x 16)	3-24
3.6.4.2. EPROM-RAM memory module (128K x 16)	3-25
3.6.5. Flash-RAM memory modules (128K x 16)	3-26
3.7. Status and failure indication	3-28
3.8. I/O modules	3-29
4. PLC functions	4-1
4.1. Method of operation	4-1
4.2. Local operands	4-2
4.2.1. Overview	4-2
4.2.2. Short description of local operands	4-3

4.3. Commands overview	4-5
4.3.1. Logical operations commands	4-6
4.3.2. Arithmetics commands	4-12
4.3.3. Comparison commands	4-13
4.3.4. Shift and rotation commands	4-14
4.3.5. Byte and flag manipulation	4-15
4.3.6. Module calls	4-15
4.3.7. Jump commands	4-16
4.3.8. Copy and BCD commands	4-16
4.3.9. Programmable pulses, timers and counters	4-17
4.3.10. Initialisation module commands	4-18
4.3.11. Special commands	4-18
4.3.12. Data module commands	4-19
4.4. Registers	4-21
4.5. Addressing	4-22
4.5.1. Address mnemonic	4-22
4.5.2. Offset addressing	4-22
4.5.3. Types of addressing, overview	4-24
5. Profi Control 680I in PROFIBUS networks	5-1
5.1. Basic information	5-1
5.2. External operands	5-3
5.2.1. Process image of a slave	5-4
5.2.1.1. Specification of operands	5-4
5.2.1.2. Addressing	5-5
5.2.2. Process image of an FMS master	5-6
5.2.2.1. Specification of operands	5-7
5.2.2.2. Addressing	5-8
5.2.3. Data consistency ensured by OS_CRIT	5-9
5.2.4. PROFIBUS messages	5-11
6. Profi Control 680I-SL in PROFIBUS networks	6-1
6.1. Basic information	6-1

Table of contents

6.2. Data exchange via PROFIBUS-DP	6-3
6.2.1. Station address	6-3
6.2.2. External operands	6-3
6.2.2.1. External operands (overview)	6-4
6.2.3. Interrupt-controlled data exchange	6-5
6.2.4. Initialising PROFIBUS via the user program	6-6
6.3. Communication parameters	6-8
6.3.1. Sending parameterisation data (Prm_Data)	6-8
6.3.1.1. General bus parameters	6-9
6.3.1.2. Device-specific bus parameters	6-10
6.3.1.3. PROFIBUS status in BI15.15	6-11
6.4. Configuration	6-12
6.5. Diagnostic information (Diag_Data)	6-13
6.5.1. Standard diagnostic data	6-13
6.5.2. Device-specific diagnostic data	6-15
6.6. Master - slave communication (Profi Control 680I-SL)	6-16
6.6.1. Initialisation	6-16
6.6.2. Data communication	6-17
6.6.3. Error message	6-18
6.6.3.1. Requesting diagnostic data via the user program	6-19
6.7. Device master data file KUHN6800.GSD	6-20

Appendix

A. Summary of data	A-1
A.1. Technical data	A-1
A.2. Part numbers	A-3

B. PROFIBUS hardware installation	B-1
B.1. PROFIBUS cable	B-1
B.1.1. Cable type A	B-1
B.1.1.1. Transfer rate and Cable length	B-1
B.1.1.2. T-lines	B-2
B.1.2. Cable type B	B-3
B.1.3. Shielding	B-4
B.2. Connecting the stations	B-5
B.2.1. Connectors	B-5
B.2.1.1. Bus branches	B-6
B.2.1.2. Bus terminator	B-7
C. References to literature	C-1
D. Error handling	D-1
D.1. Short circuit at output (error 1)	D-3
D.2. Voltage monitoring (supply, error 2)	D-4
D.2.1. Overvoltage	D-4
D.2.2. Undervoltage	D-5
D.3. Watchdog (program runtime exceeded, error 3)	D-7
D.4. PROFIBUS error (errors 4, 5)	D-8
D.5. Checksum in the user program (error 8)	D-9
D.6. Hierarchy error (error 9)	D-10

E. EPROMs and Flash-EPROMs E-1

E.1 Programming EPROMs E-1

E.1.1. Using KUBES 4.00 or higher E-2

E.1.1.1. Creating files with KUBES E-2

E.1.1.2. Programming EPROMs E-3

E.1.2. Using KUBES 4.00ß E-4

E.1.2.1. Creating files with KUBES E-4

E.1.2.2. Programming EPROMs E-5

E.2. Programming Flash-EPROMs E-6

E.2.1. Using KUBES to copy the program to Flash-EPROM E-6

E.2.2. Using "FLASH_UP" to program Flash-EPROMs E-7

E.2.2.1. Creating the FLASH_UP diskette E-7

E.2.2.2. Transferring the program to the controller E-9

Index Index-1

Sales & Service

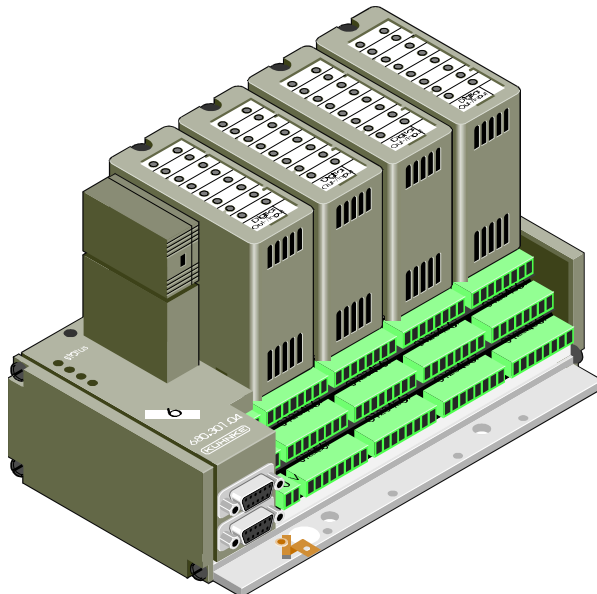
1. Introduction

KUAX 680I and Profi Control 680I-SL are high-performance controllers in a modular design. They are equipped with modules which either work directly on the PLC bus or, if they have their own processor, indirectly via a dual-port RAM.

There are connectors for three-core leads so that approximation switches etc. can be supplied via the same cable as the signal line. Additional terminal blocks are thus made redundant.

The room-saving construction allows you to install the device exactly where input signals are generated or where output signals are needed.

Their built-in PROFIBUS port turns them into separate controllers working on their own or into stations of a PROFIBUS network. In the latter case, Profi Control 680I acts as a master while Profi Control 680I-SL integrates into the network as a DP slave.



1.1. Moving on: From separate controllers to network systems

Programmable logic controllers (PLC) play a major role in industrial automation. Three main reasons can be given for that:

- they are universally applicable,
- programming is easy and comprehensible,
- they provide numerous utilities for testing and start-up.

As problem-oriented micro-computers, PLCs have taken over more and more elements of process computing systems in accordance with their permanently growing capacities. They have become universal instruments of automation in a wide range of action, designs and configurations. A strong tendency towards hierarchical process control systems has since become apparent. In these, tasks are separated. Each part-system executes tasks according to its optimal aptitude.

Generally speaking, PLCs perform on the process interfacing level whereas PCs are used for calculating and managing large amounts of data at the control level.

Task separation leads to decentralisation

Integrating further components such as sensors and actuators leads to field-level network systems.

High-capacity interfaces and pathways of transmission are of the greatest importance for communication between PLCs as well as between PLCs and other devices, controllers and PCs.

Benefits of decentralisation

- reduction of multicore cables,
 - material (cables, connectors..)
 - space (conduits, terminal blocks, switching cabinet)
 - installation (time, possibilities of mistakes)
- increased efficiency
- more transparency by the program structure being similar to the object structure
- failure treatment (if one part of the system fails other parts can continue to work)
- reduced start-up times
- pre-testing of individual stations
- devices supplied by different manufacturers can be combined

These benefits can take full effect only if a standardised solution for all part-systems is available. This solution is mainly characterised by its speed, reliability of data transfer, and its being designed as an open system. All of these features are provided by PROFIBUS.



For further information about PROFIBUS networks refer to chapters

"5. Profi Control 680I in PROFIBUS networks" and

"6. Profi Control 680I-SL in PROFIBUS networks".

2. Safety and Reliability

2.1. Target group

This instruction manual contains all information necessary for the use of the described product (control device, control terminal, software, etc.) according to instructions. It is written for the personnel of the construction, project planning, service and commissioning departments. For proper understanding and error-free application of technical descriptions, instructions for use and particularly of notes of danger and warning, extensive knowledge of automation technology is compulsory.

2.2. Reliability

Reliability of Kuhnke controllers is brought to the highest possible standards by extensive and cost-effective means in their design and manufacture.

These include:

- selecting high-quality components,
- quality arrangements with our sub-suppliers,
- measures for the prevention of static charge during the handling of MOS circuits,
- worst case dimensioning of all circuits,
- inspections during various stages of fabrication,
- computer aided tests of all assembly groups and their efficiency in the circuit,
- statistical assessment of the quality of fabrication and of all returned goods for immediate taking of corrective action.

Despite these measures, the occurrence of errors in electronic control units - even if most highly improbable - must be taken into consideration.

2.3. Notes

Please pay particular attention to the additional notes which we have marked by symbols in this instruction manual:

2.3.1. Danger



This symbol warns you of dangers which may cause death, (grievous) bodily harm or material damage if the described precautions are not taken.

2.3.2. Dangers caused by high contact voltage



This symbol warns you of dangers of death or (grievous) bodily harm which may be caused by high contact voltage if the described precautions are not taken.

2.3.3 Important information / cross reference



This symbol draws your attention to important additional information concerning the use of the described product. It may also indicate a cross reference to information to be found elsewhere.

2.4. Safety

Our product normally becomes part of larger systems or installations. The following notes are intended to help integrating the product into its environment without dangers for man or material/equipment.

2.4.1. To be observed during project planning and installation



- 24V DC power supply: Generate as electrically safely separated low voltage. Suitable devices are, for example, split transformers constructed in compliance with European standard EN 60742 (corresponds to VDE 0551)
- In case of power breakdowns or power fades: the program is to be structured in such a way as to create a defined state at restart that excludes dangerous states.
- Emergency switch-off installations must comply with EN 60204/IEC 204 (VDE 0113). They must be effective at any time.
- Safety and precautions regulations for qualified applications have to be observed.
- Please pay particular attention to the notes of warning which, at relevant places, will make you aware of possible sources of dangerous mistakes or failures.
- Relevant standards and VDE regulations are to be observed in every case.
- Control elements are to be installed in such a way as to exclude unintended operation.
- Control cables are to be laid in such a way as to exclude interference (inductive or capacitive) which could influence controller operation or its functionality.



To achieve a high degree of conceptual safety in planning and installing an electronic controller it is essential to follow the instructions given in the manual exactly because wrong handling could lead to rendering measures against dangerous failures ineffective or to creating additional dangers.

2.4.2. To be observed during maintenance and servicing

- Precaution regulation VBG 4.0 must be observed, and section 8 (Admissible deviations during working on parts) in particular, when measuring or checking a controller in a power-up condition.
- Repairs must only be made by specially trained Kuhnke staff (usually in the main factory in Malente). Warranty expires in every other case.
- Spare parts:
Only use parts approved of by Kuhnke. Only genuine Kuhnke modules must be used in modular controllers.
- Modules must only be connected to or disconnected from the controller with no voltage supplied. Otherwise they may be destroyed or (possibly not immediately recognisably!) suffer a negative impact on their functionality.
- Always deposit batteries and accumulators as hazardous waste.

2.5. Electromagnetic compatibility

2.5.1. Definition

Electromagnetic compatibility is the ability of a device to function satisfactorily in its electromagnetic environment without itself causing any electromagnetic interference that would be intolerable to other devices in this environment. Of all known phenomena of electromagnetic noise, only a certain range occurs at the location of a given device. This noise depends on the exact location. It is defined in the relevant product standards.

The international standard regulating construction and degree of noise resistance of programmable logic controllers is IEC 1131-2 which, in Europe, has been the basis for European standard EN 61131-2.

2.5.2. Resistance to interference

1. Electrostatic discharge, ESD
in acc. with EN 61000-4-2, 3rd degree of sharpness
2. Irradiation resistance of the device, HF
in acc. with EN 61000-4-3, 3rd degree of sharpness
3. Fast transient interference, burst
in acc. with EN 61000-4-4, 3rd degree of sharpness
4. Immunity to damped oscillations
in acc. with EN 61000-4-12 (1 MHz, 1 kV)

2.5.3. Interference emission

- Interfering emission of electromagnetic fields, HF
in acc with EN 55011, limiting value class A, group 1



If the controller is designed for use in residential areas, then high-frequency emissions must comply with limiting value class B as described in EN 55011.

Fitting the controller into an earthed metal cabinet and equipping the supply cables with filters are appropriate means for keeping the corresponding limiting values.

2.5.4. General notes on installation

As component parts of machines, facilities and systems, electronic control systems must comply with valid rules and regulations, depending on the relevant field of application.

General requirements concerning the electrical equipment of machines and aiming at the safety of these machines are contained in Part 1 of European standard EN 60204 (corresponds to VDE 0113).



For safe installation of our control system please observe the following notes:

2.5.5. Protection against external electrical influences

Connect the control system to the protective earth conductor to eliminate electromagnetic interference. Ensure practical wiring and laying of cables.

2.5.6. Cable routing and wiring

Separate laying of power supply circuits, never together with control current loops:

- DC voltage 60 V ... 400 V
- AC voltage 25 V ... 400 V

Joint laying of control current loops is allowed for:

- shielded data signals
- shielded analogue signals

- unshielded digital I/O lines
- unshielded DC voltages < 60 V
- unshielded AC voltages < 25 V

2.5.7. Location of installation

Make sure that there are no impediments due to temperatures, dirt, impact, vibration and electromagnetic interference.

Temperature

Consider heat sources such as general heating of rooms, sunlight, heat accumulation in assembly rooms or control cabinets.

Dirt

Use suitable casings to avoid possible negative influences due to humidity, corrosive gas, liquid or conducting dust.

Impact and vibration

Consider possible influences caused by motors, compressors, transfer routes, presses, ramming machines and vehicles.

Electromagnetic interference

Consider electromagnetic interference from various sources near the location of installation: motors, switching devices, switching thyristors, radio-controlled devices, welding equipment, arcing, switched-mode power supplies, converters / inverters.

2.5.8. Particular sources of interference

Inductive actuators

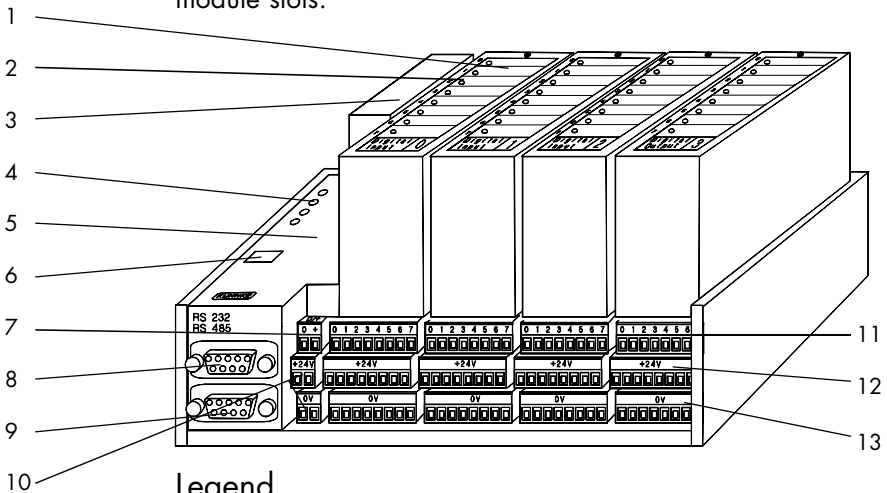
Switching off inductances (such as from relays, contactors, solenoids or switching magnets) produces overvoltages. It is necessary to reduce these extra voltages to a minimum. Reducing elements may be diodes, Z-diodes, varistors or RC elements. To provide suitably designed reducing elements, we recommend that you contact the manufacturer or supplier of the corresponding actuators for the relevant information.

3. Hardware

Dimensions, connectors, memory modules and slots are the same for both Profi Control 680I and Profi Control 680I-SL.

3.1. External appearance

The picture below represents a device equipped with 4 module slots:



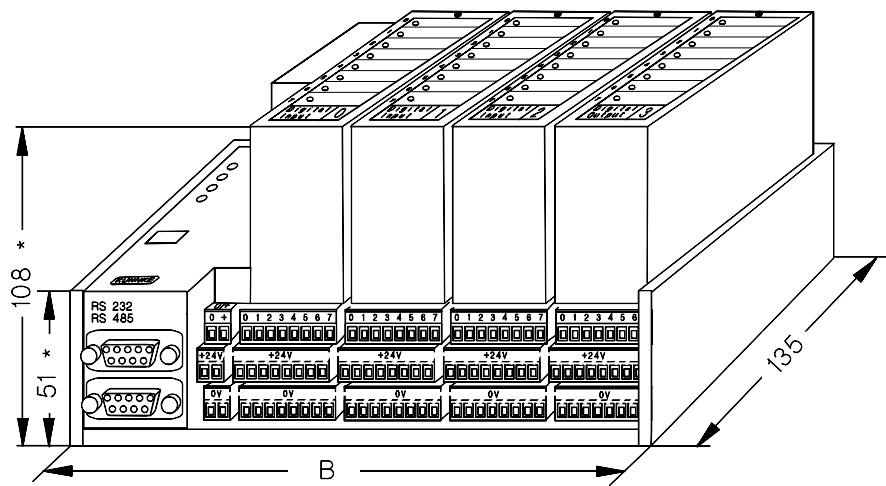
Legend

- 1 Modules with labels
- 2 Module status LED
- 3 User memory module
- 4 Status indicators (LEDs): run (green), stop (red), failure (red), com (yellow): *no function*
- 5 Cover of PROFIBUS coding switch
- 6 Label for station address
- 7 Module power supply connector
- 8 RS 232 (V.24 interface)
- 9 PROFIBUS port
- 10 Power supply to system and voltage supply terminals
- 11 X1: terminal block, signals
- 12 X2: terminal block, + 24V DC
- 13 X3: terminal block, 0 V

Basic device

3.2. Dimensions and installation of basic device

The basic device frame serves as a carrier for the various input/output modules (picture shows a device with 4 modules). All dimensions are in millimetres:



The width depends on the number of slots:

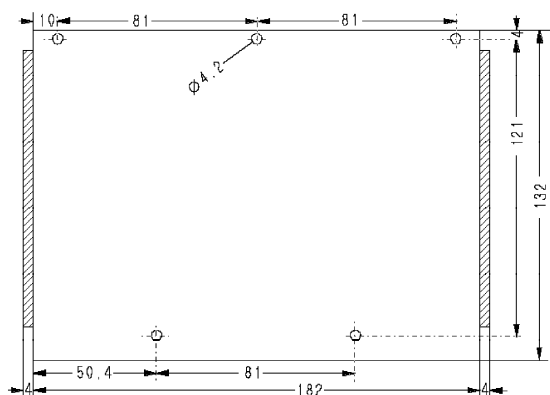
Slots	Width "B"
4	190 mm
8	332 mm

*The device is 7.5 mm higher if installed on a carrier rail.

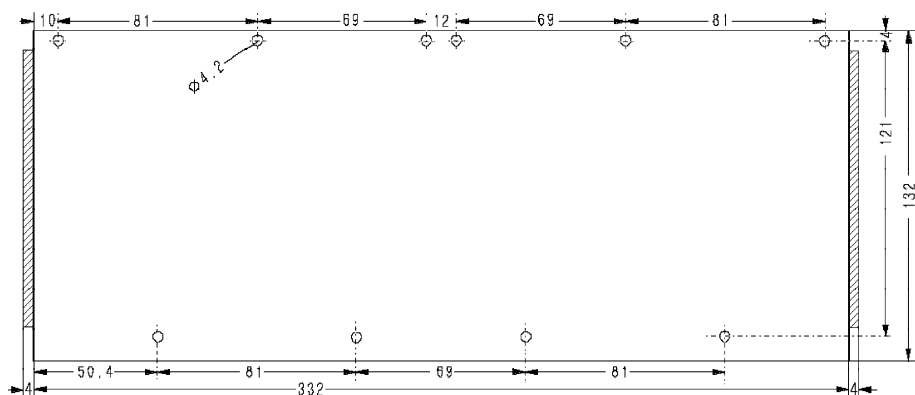
3.2.1. Installation at the wall

The device's baseplate contains 4.2 mm dia. drillholes for screwing the machine to the wall.

Basic device frame for 4 modules



Basic device frame for 8 modules



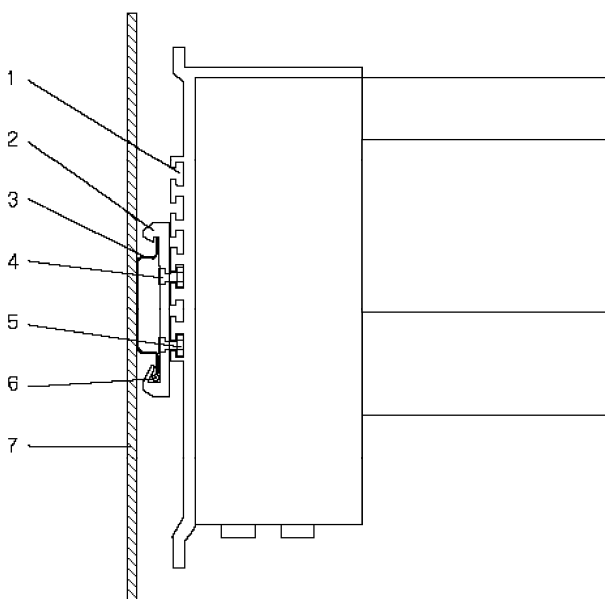
3.2.2. Installation on carrier rail

You can also install the device on carrier rails in compliance with DIN EN 50022 (35 x 7.5 mm).

This requires screwing 2 (or, in the case of devices with 8 slots, 3) quick mounts for carrier rail installation into the controller's baseplate. The quick mounts are to be ordered separately (see Appendix "B.1. Accessories").

There are 5 T-slots along the baseplate. Two of these you need to insert the nuts into which you screw the quick mounts (the picture shows a centered device attachment).

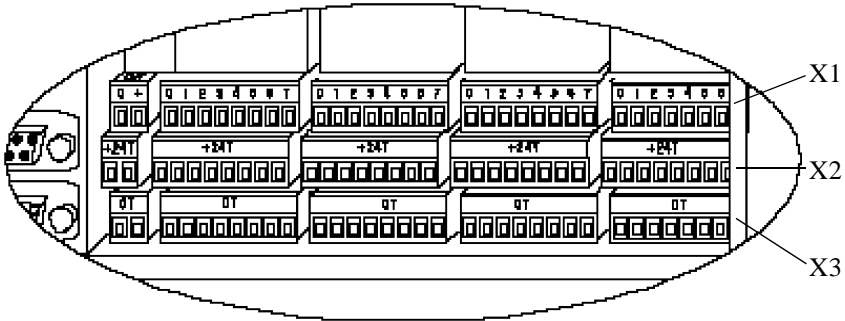
In the case of older controllers you have to remove one side panel before you can insert the nuts.



- 1 Slots for insertion of nuts
- 2 Quick mount for carrier rail installation
- 3 Carrier rail, 35 mm
- 4 Quick mount screw
- 5 Nut
- 6 Steel spring
- 7 Wall

3.3. Power supply

DC voltage is supplied to the device via clamp-screw terminals. Input/output modules and system are supplied separately.



Core diameter

You can connect flexible leads with 1.5 mm² dia. cores.

Screwdriver

Use a slot screwdriver with a max. blade width of 2.3 mm for connecting the leads.

To remove terminal blocks

The blocks fit very tightly to make sure that they don't come off when exposed to vibration. If you're finding it impossible to pull them off by hand, you can use a flat object such as a screwdriver with a wide blade.



Never pull at the leads to remove a terminal block because you might otherwise pull them out of the terminals or rip them off.

3.3.1. System power supply

Voltage:	24 V DC	+25 %	-20 %
Connection:	+24 V:	X2 blocks,	double terminal
	0V:	X3 blocks,	double terminal

The device is to be supplied with 24 V DC. A built-in power pack transforms this into the system voltage (5 V) which is then used to supply the CPU, the modules, the device bus and the PROFIBUS port. It supports loads of up to 1000 mA (for older devices see "A. Technical data").



When connecting: please always connect the 0V lead first and disconnect it last. Otherwise, equalisation current may flow through the interface if a V24 (RS 232) is connected, thus blowing the electronic fuse. It takes about one minute after that before the machine is ready to operate again (cooling down).

In older devices this may destroy the choke in the V.24 interface which would make all further communication impossible.

Supply terminals

The fed-in voltage is internally distributed among the supply terminals which then supply the external sensors and actuators (see figure "3.3.3. Signal line connection"). Like this, an external distributor terminal is no longer required.

3.3.1.1. Special feature of devices with 8 module slots

Devices with 8 module slots have a twin set of every double terminal for power supply.



The double terminals with the same function are not interconnected in the factory. You would therefore have to put in jumper wires.

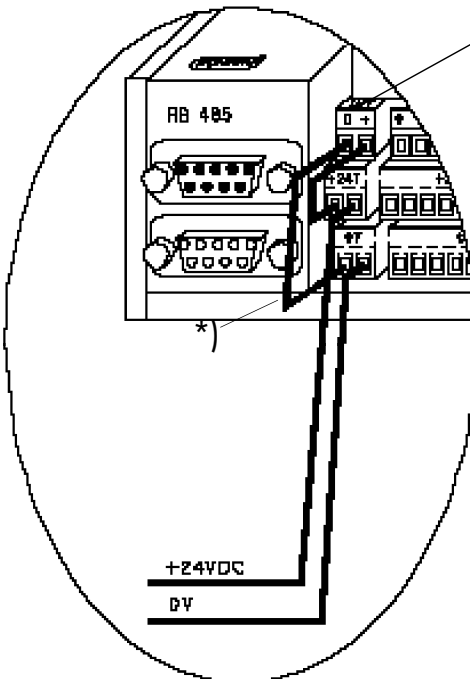
3.3.2. Power supply of inputs and outputs

Voltage: 24 V DC + 25 % - 20 %
 Connection: +24 V: X1 blocks, double terminal, connection "+"
 0V: X1 blocks, double terminal, connection "0"



A maximum current of 8 A may be lead via every terminal. For example, you must not load more than 16 outputs with 0.5 A at the same time.

The connections have been kept separate on purpose (upper double terminal) so that the outputs can be switched off separately (see next page).



Power supply to:
 - Outputs (+, 0V)
 - Inputs (0 V)

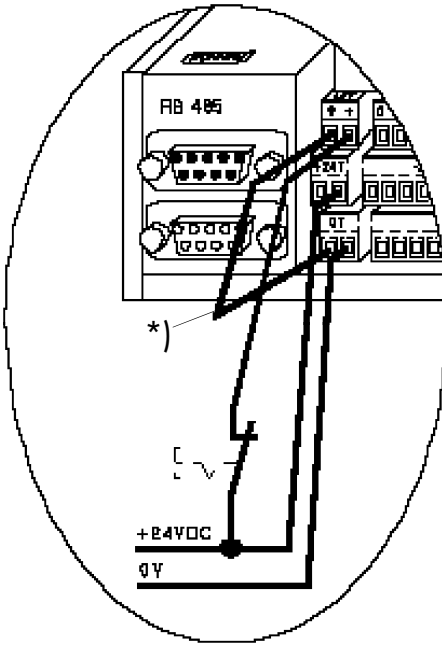
Normally, you only need two jumper wires to pick up the voltage from the double terminals of the X2 and X3 blocks (see figure "3.3. Connections for ..").

This voltage also supplies the status LEDs of the output modules. We therefore recommend that you apply the voltage in test mode also.



A connection between the 0V connectors (see "" in the picture above) is to be made in any case.*

3.3.2.1. Switching off the outputs separately



The outputs being connected separately allows you to switch them off without switching off the PLC as well.

The advantage is that the system is still supplied, input signals are read, and communication is continued.



Outputs must not be supplied in reverse while the power supply to them is switched off!

When you switch off the outputs make sure to also interrupt the supply of all auxiliary or other switching elements that may be connected in parallel with the outputs.

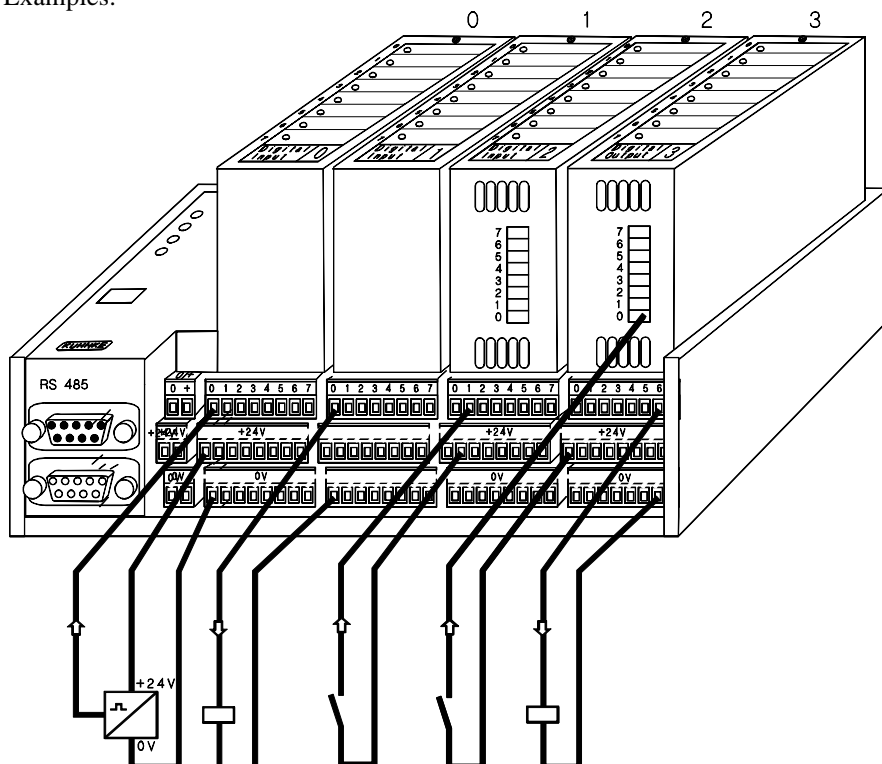
This only applies if the system power supply is still on. Outputs set by the program may be supplied via the protective diode of another output that is being supplied in reverse, thus by-passing the switch-off function for these outputs. Furthermore, a high load may destroy the protective diode of the feeding output.

*) A connection between the 0V terminals is to be made in every case to serve as potential equalisation. Failure to comply may cause destruction of components.

3.3.3. Signal line connection

Eight terminals for the signal lines are assigned to each slot for either input or output modules. They are located in the X1 blocks (blocks of 8) directly underneath the relevant slot. The terminals are numbered consecutively from 0 to 7 which corresponds to the channel number (for modules with 8 inputs or outputs).

Examples:



Legend

- 0 Input module, 8 inputs: sensor to input
- 1 Output module, 8 outputs: relay to output
- 2 Input module, 16 inputs: switch to input
(lower group)
- 3 Input/output module: switch to input
relay to output

3.3.4. Connection to earth

For reasons of operational safety, the 0V lines as well as the bus cable shield are connected to earth (frame) via capacitors. This allows high-frequency interference to bleed off.

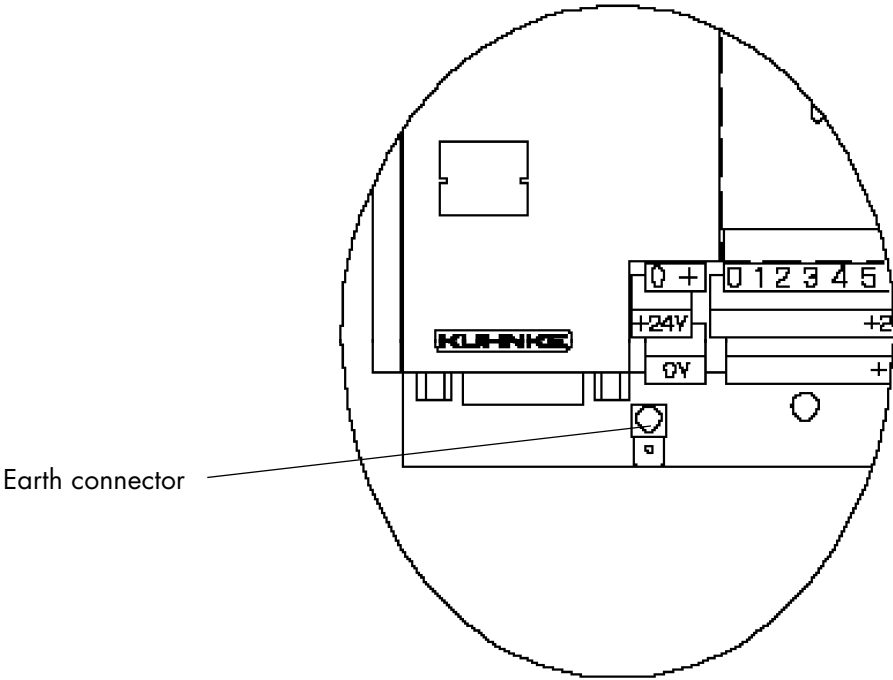


The basic device frame is to be connected to earth before this protective means can take effect.

Earth connects to the plain 6.3 x 0.8 mm plug located at the front left of the baseplate.

Earth wire

Minimum diameter: 1.5 mm²



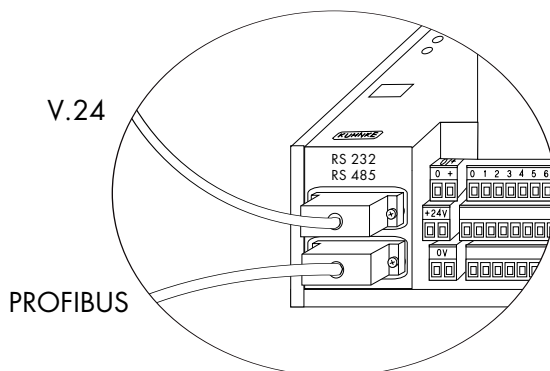
3.4. Communication ports

The device has two serial interfaces

- upper connector (RS 232):
female 9-pin D-Sub connector.

V.24 interface for programming or connection to an operating terminal (KDT 631, KDT 633...)

- lower connector (RS 485):
female 9-pin D-Sub connector
PROFIBUS interface



These two interfaces are described on the next couple of pages.

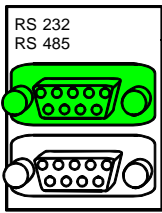
Further interfaces

There are communication modules available that are equipped with a variety of interfaces.



For a description of these modules refer to instruction manual E 326 GB "Modules..."

3.4.1. V.24 interface (RS 232)



The V.24 interface is located at the upper of the two female 9-pin D-Sub connectors.

Pin assignment

Pin	Connects with
1	unused
2	TxD
3	RxD
4	unused
5	Gnd
6	unused
7	CTS
8	RTS
9	unused



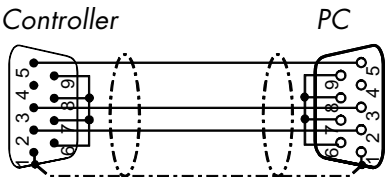
The cable shield is to be connected to the plug casing.

Use of the V.24 interface

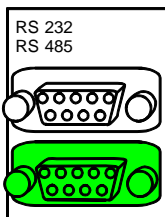
The V.24 interface is mainly used for programming. Another major application is data communication, e.g. with dialogue terminals, PCs or other devices.

Programming cable

The standard programming cable connects the controller and the PC. Use part number 657.151.03 to order it. The cable is 2.5 m long and connects as illustrated below:



3.4.2. PROFIBUS port (RS 485)



The PROFIBUS port is located at the lower of the two female 9-pin D-Sub connectors.

Pin assignment

Pin	Connects with
1	cable shield (connected to the frame)
2	unused
3	RxD/TxD-P (data +)
4	unused
5	DGnd
6	VP (+5 V supply)
7	unused
8	RxD/TxD-N (data -)
9	unused

Supply of bus termination

An external bus terminator can be supplied with 5 V via pins 5 (DGnd) and 6 (VP).

Transfer speed (baud rate)

This PROFIBUS connector supports baud rates of 9.6 kbit/s, 19.2 kbit/s and 500 kbit/s.

- Profi Control 680I: use dip switches 7 and 8 to set the baud rate (see chapter "5. Profi Control 680I in PROFIBUS networks").
- Profi Control 680I-SL automatically sets itself to the same baud rate as the master.



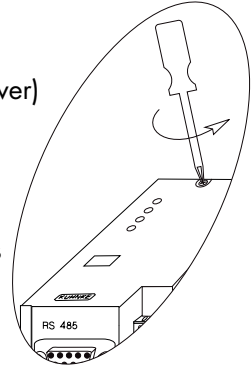
Refer to appendix "B. PROFIBUS installation" for information about cable specifications, cable connection, bus terminations, etc.

3.5. Coding switches

The coding switch is located underneath the side cover carrying the status LEDs.

To set the coding switches

- Switch off power supply
- Remove screws (size 1 slot screwdriver)
- Carefully pull the cover towards the front (across the D-Sub connectors) and lift it
- Use a suitable tool (e.g. thin screwdriver) to set the switches (see tables "3.5.1..." and "3.5.2...")
- Put cover back on



The settings of the coding switches are only verified when you switch on the power supply. Changes during operation will therefore have no effect.

3.5.1. Coding switch of Profi Control 680I

Use the coding switch of Profi Control 680I to set the combimaster's station address and data transfer rate:

Dip switch								Station address
1	2	3	4	5	6	7	8	
off	off	off	off	off	off			0
on	off	off	off	off	off			1
off	on	off	off	off	off			2
on	on	off	off	off	off			3
etc., through to:								
off	on	on	on	on	on			62
on	on	on	on	on	on			63
1 2 4 8 16 32 ←								Significance of switch 1...6 (bin.)
						7	8	Baud rate
						off	off	500 kbit/s
						on	off	19.2 kbit/s
						off	on	9.6 kbit/s
						on	on	illegal setting

To save space, not all station address settings have been listed in the table.

Switches 1...6 are binary coded. You will therefore find it easy to tell the missing station addresses by the significances.



The coding switch has 10 pins. Pins 9 and 10 have no function.

3.5.2. Coding switch of Profi Control 680I-SL

Use the coding switch of Profi Control 680I-SL to set the DP slave's station address:

Dip switch							PROFIBUS station address
1	2	3	4	5	6	7	
off	off	off	off	off	off	off	0
on	off	off	off	off	off	off	1
off	on	off	off	off	off	off	2
on	on	off	off	off	off	off	3
etc., through to:							
on	off	on	on	on	on	on	125
off	on	on	on	on	on	on	126, allows setting via KUBES module "INIT_SL"
on	on	on	on	on	on	on	127, illegal setting
<i>1 2 4 8 16 32 64 ← Significance of switch 1...7 (bin.)</i>							

To save space, not all station address settings have been listed in the table.

Switches 1...6 are binary coded. You will therefore find it easy to tell the missing station addresses by the significances.

To set the address via the user program

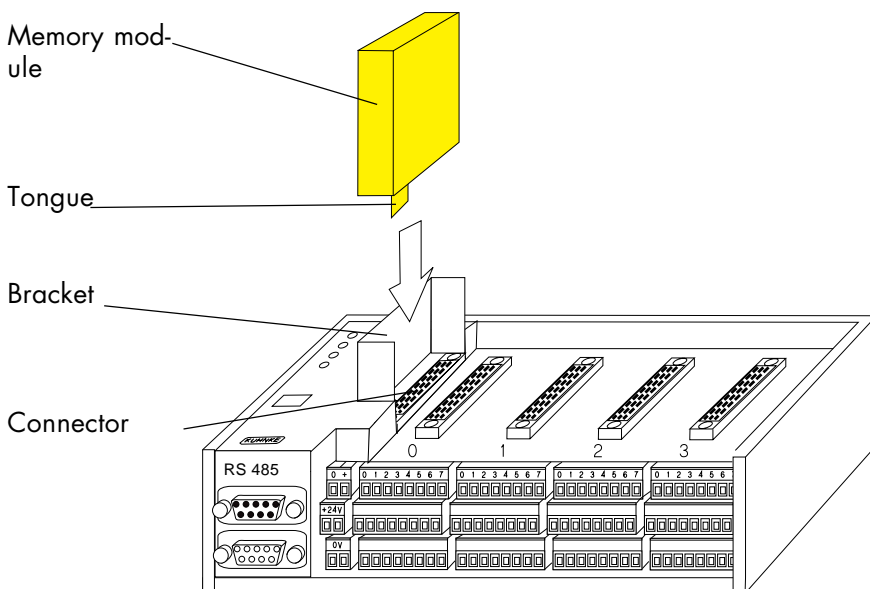
The station address can also be set via the user program. If you wish to use this option, first set the coding switch to "126" and then set the actual station address via KUBES module "INIT_SL".



The coding switch has 10 pins. Pins 8 to 10 have no function.

3.6. User program memory

Plug the memory module from above into the appropriate slot of the controller:



Legend

- Connector
The slot connector is of the same type as the ones for the input, output, and function modules.
- Tongue
The module casing has a tongue on its left side to avoid the memory module being plugged into one of these slots by mistake.
- Bracket
The purpose of the bracket is to hold the memory module firmly in the controller.



Make sure to switch off the power supply to the controller before you plug in or unplug memory modules. Failure to comply may lead to destruction of components inside the module.

User program memory

3.6.1. Function

Program memory

The main purpose of the user memory is to contain the user program which you write using the KUBES programming software (see instruction manual "E 327 GB, KUBES and MOBES").

Data memory

You can reserve some of the available memory space as data memory. The reserved space is, of course, no longer available as program memory.

As from monitor version 4.17 you can also copy data modules to the user memory.

Memory management

The user program memory occupies a maximum of 8 banks, bank 0 to bank 7. The actual configuration depends on the memory module you are using (see chapter 3.6.2. to 3.6.4.).

The size of the data memory can be set via the KUBES programming software.

Prerequisite:

- Online mode
- Program stop (Reset or Stop)

Start:

- "PLC" menu from the main menu bar
- Option "Set memory size"



- Reduce the "Bank" setting by the amount you wish to reserve for data (see E 327 GB "KUBES")

3.6.2. RAM memory modules

RAM memory modules are mainly applied for the development, startup and testing of programs. You can write into them, modify or delete them as you please.

Design

They are entirely designed as RAM modules equipped with SMD components.

Accu

The RAM memory module has an accumulator to protect all data when the module is not supplied with power:

Capacity: 80 mAh

Buffer time: ≥ 6 weeks (at 0...40 °C)

Recharge time: ≤ 72 h



Due to varying lengths of warehouse times, the accu's charge level is not defined at the time of delivery.

User program memory

3.6.2.1. RAM memory module (32K x 16)

Capacity

Total: 32K x 16 bit

Available: 52 Kbyte or ~ 10K of instructions

Configuration: bank 0, 52 Kbyte

Part number: 680.426.01

Application:

Program development for EPROM module "32K x 16"

3.6.2.2. RAM memory module (256K x 16)

Capacity

Total: 256K x 16 bit

Available: 128 Kbyte or ~ 40K of instructions

128 Kbyte data memory

Configuration

Bank 0 52 Kbyte

Bank 1...7 64 Kbyte each

Part number: 680.426.03

Application:

Program development for EPROM module "128K x 16" or
the EPROM-RAM module resp. (see pages below).

3.6.3. EPROM memory modules

Completed and tested programs are copied to EPROMs which provide the best possible protection against loss of data by undervoltage, noise pulses, etc.

EPROMs

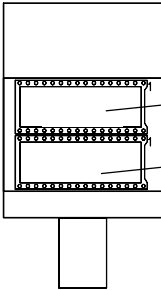
Every EPROM module is to be equipped with 2 EPROMs placed in parallel, parallel meaning that one EPROM is to store the lower 8 bit (low byte) and the other EPROM the upper 8 bit (high byte).



Refer to chapter "E. Programming EPROMs" to learn how to use KUBES to create EPROM files and to copy them to the EPROMs.

To plug EPROMs into the EPROM module

EPROM memory modules are open at the front. Plugging and unplugging EPROMs is therefore easy.



Socket for the "low byte" program EPROM

Socket for the "high byte" program EPROM



Make sure to stick to the above order of EPROMs because failure to do so will stop the program from starting in the controller.

User program memory

3.6.3.1. EPROM memory module (32K x 16)

Sockets:	2 Dip sockets, 28-pin
Capacity	
Total:	32K x 16 bit
Available:	52 Kbyte or ~ 10K of instructions
Configuration:	bank 0, 52 Kbyte
Access time:	≤ 100 ns
Part numbers	
Module:	680.427.01
EPROM 32K x 8, 2 pcs:	657.491.20

3.6.3.2. EPROM memory module (128K x 16)

Sockets:	2 Dip sockets, 32-pin
Capacity	
Total:	128K x 16 bit
Available:	128 Kbyte or ~ 40K of instructions
Configuration	
Bank 0	52 Kbyte
Bank 1...3	64 Kbyte each
Access time:	≤ 100 ns
Part numbers	
Module:	680.427.02
EPROM 128K x 8, 2 pcs:	657.491.21

3.6.4. EPROM-RAM memory modules

The EPROM-RAM memory modules are permanently equipped with RAM components (see chapter "3.6.2. RAM memory modules"). They provide 2 additional EPROM sockets (see chapter "3.6.3. EPROM memory modules"). Use them preferably as program and data memory. Both types of memory (EPROM and RAM) can be combined. Memory allocation is done via KUBES.



The on-board RAM is not suitable for program development.

EPROMs

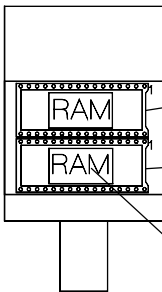
The EPROM module is to be equipped with 2 EPROMs placed in parallel, parallel meaning that one EPROM is to store the lower 8 bit (low byte) and the other EPROM the upper 8 bit (high byte).



Refer to chapter "E. Programming EPROMs" to learn how to use KUBES to create EPROM files and to copy them to the EPROMs.

To plug EPROMs into the EPROM module

EPROM memory modules are open at the front. Plugging and unplugging EPROMs is therefore easy.



Socket for the "low byte" program EPROM

Socket for the "high byte" program EPROM

Built-in RAM memory (SMD components) are located inside the EPROM sockets



Make sure to stick to the above order of EPROMs because failure to do so will stop the program from starting in the controller.

User program memory

Accu

The RAM memory module has an accumulator to protect all data when the module is not supplied with power:

Capacity: 80 mAh

Buffer time: ≥ 6 weeks (at 0...40 °C)

Recharge time: ≤ 72 h



Due to varying lengths of warehouse times, the accu's charge level is not defined at the time of delivery.

3.6.4.1. EPROM-RAM memory module (32K x 16)

EPROM

Sockets: 2 Dip sockets, 28-pin

Capacity

Total: 32K x 16 bit

Available: 52 Kbyte
or ~ 10K of instructions

Configuration:

Bank 0 52 Kbyte

Access time: ≤ 100 ns

RAM

Capacity

Total: 32K x 16 bit

Available: 64 Kbyte

Configuration:

Bank 4 64 Kbyte

Part numbers

Module: 680.428.01

EPROM 32K x 8, 2 pcs: 657.491.20

3.6.4.2. EPROM-RAM memory module (128K x 16)

EPROM

Sockets:	2 Dip sockets, 32-pin
Capacity	
Total:	128K x 16 bit
Available:	244 Kbyte or ~ 40K of instructions
Configuration	
Bank 0	52 Kbyte
Bank 1...3	64 Kbyte each
Access time:	≤ 100 ns

RAM

Capacity	
Total:	128K x 16 bit
Available:	256 Kbyte
Configuration:	
Bank 4...7	64 Kbyte each

Part numbers

Module:	680.428.02
EPROM 128K x 8, 2 pcs:	657.491.21

3.6.4. Flash-RAM memory modules (128K x 16)

Flash-RAM memory modules are equipped with Flash-EPROMs and RAM components. They can be used for program development and permanent operation because although Flash-EPROM can be reprogrammed, all data is secured against loss like in a normal EPROM. The integrated RAM can store data that may change during runtime.

Design

These modules are completely equipped with Flash-EPROM and RAM (SMD components).

Flash-EPROMs

When using KUBES to transfer the program to the controller, the program code is stored directly in the Flash-EPROM.



However, KUBES will copy the module table to a specially reserved RAM range. The table therefore remains independent of the accu's charge status.

Once you have completed the program you will have to transfer the module table to the Flash-EPROM as well to ensure that the entire program becomes independent of the accu's charge status.



Refer to chapter "E.2 Programming Flash-EPROMs" to learn how to transfer the program to Flash-EPROMs.

RAM/accu

The RAM memory module has an accumulator to protect all data when the module is not supplied with power:

Capacity:	80 mAh
Buffer time:	≥ 6 weeks (at 0...40 °C)
Recharge time:	≤ 72 h



Due to varying lengths of warehouse times, the accu's charge level is not defined at the time of delivery.

Technical data

Flash-EPROM

Capacity

Total:

128K x 16 bit

Available:

224 Kbyte

or ~ 37K of instructions

Configuration

Bank 0

32 Kbyte

Bank 1...3

64 Kbyte each

Access time:

 ≤ 100 ns

RAM

Capacity

Total:

128K x 16 bit

Available:

224 Kbyte

Configuration:

Bank 4...6

64 Kbyte each

Bank 7

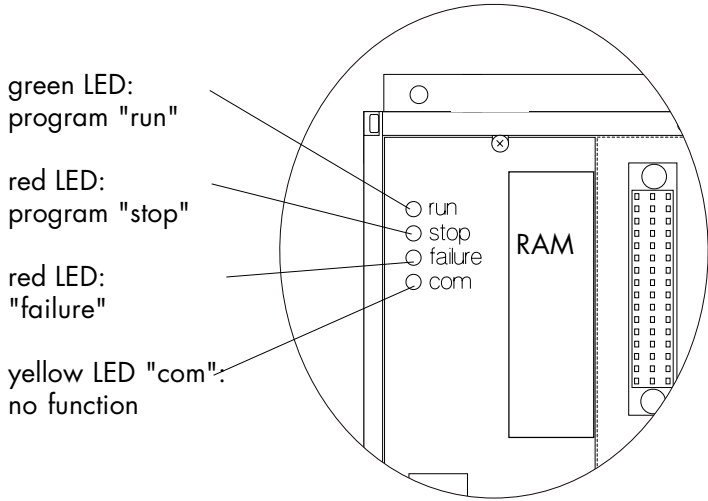
32 Kbyte

Part number:

680.428.03

3.7. Status and failure indication

There are four light emitting diodes (LEDs) that indicate the controller's current status. They are located on the left side:



LED "run"

in Profi Control 680I and 680I-SL:

- Permanently on: controller in operation

Profi Control 680I only:

- Flashing, 2 Hz (slow): ALI's trying to establish a connection to the registered PROFIBUS stations.
- Flashing 5 Hz (fast): ALI has found a parameterisation error.

Corrective action: Check network via VEBES, transfer project to the controller again and restart

LED "failure"

Failures and errors are indicated as the LED flashing at varying rates.



Refer to chapter "D. Error handling" to learn the significance and context of these messages.

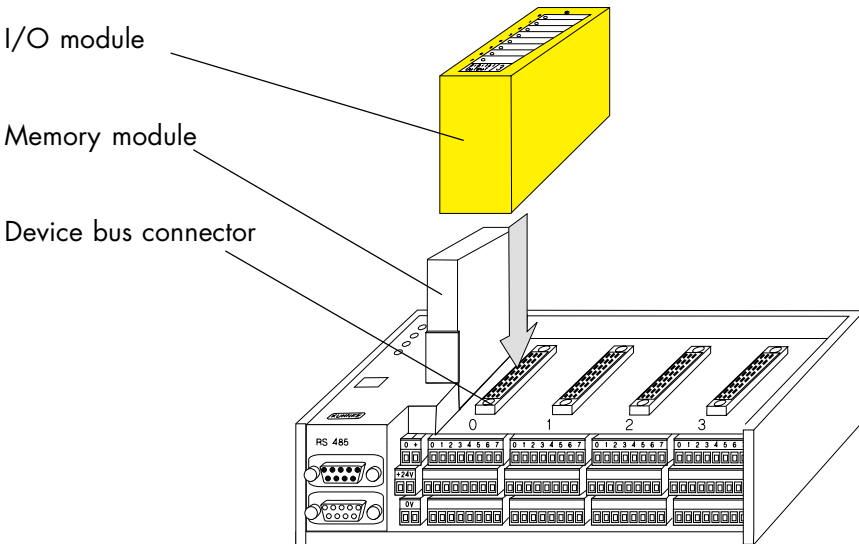
3.8. I/O modules

Modules plugged into the appropriate slots maintain the connection to the peripherals (inputs and outputs, counter inputs, further serial interfaces, etc.).

The quantity of modules depends on the variant you are using:

- devices with 4 slots (0...3): 4 modules
- devices with 8 slots (0...7): 8 modules

The modules are plugged in from above and connect to Profi Control 680I via the device bus connector:



Instruction manual E 326 GB "Modules of the 680 system" describes all I/O modules.



Never plug or unplug modules if the power is on.

4. PLC functions

This chapter is a treatment of the standard PLC functions of both Profi Control 680I and Profi Control 680I-SL. It does not go into any PROFIBUS details because they will be described in chapters 5 and 6.

4.1. Method of operation

The microprocessor responsible for the user program receives its program from two different types of program memory:

The monitor program memory

contains the monitor program (monitor) which represents an operating system containing all of the controller's system features. The monitor is delivered with the CPU.

The user program memory

contains the programs to control the machine or plant. These programs are created in the KUBES environment on a PC. KUBES has (online) access to the controller via one or several of the following interfaces:

- V.24 interface (RS 232)

Connects with one of the PC's COM ports via the programming cable (657.151.03)

- PROFIBUS port (RS 485)

Combimaster Profi Control 680I can also be programmed via the PROFIBUS port. This requires you to have a PC that is equipped with a PROFIBUS card (PC Control 645-500).



This does not apply to DP slave Profi Control 680I-SL! Please refer to instruction manual E 405 GB "PC Control 645-500", chapter "5. PC Control 645-500 used as programming card".

In the next chapters you will find all the information you need for writing a user program to run in Profi Control 680I.



Refer to instruction manual E 327 GB "KUBES" to learn how to write programs using KUBES.

Operands

4.2. Local operands

Local operands can be addressed by the user program as inputs, outputs, or internal memory cells directly in the controller.



For details on external operands refer to chapters 5 and 6.

4.2.1. Overview

Group	Input	Function	Type	Qty.	Input range		Comment
					from	to	
I	I_	inputs	bit	128	I00.00	I15.07	inc. process image
O	O_	outputs		128	O00.00	O07.07	inc. process image
M	M_	markers		256	M00.00	M15.15	
SM	SM_			256	SM00.00	SM15.15	
LM	LM_			256	LM00.00	LM15.15	
FM	FM_			256	FM00.00	FM15.15	
SO	SO_		256	SO00.00	SO15.15		
R	R_	remanent markers	byte	256	R00.00	R15.15	buffered by accu on the device
SR	SR_			256	SR00.00	SR15.15	
BM	BM_	byte markers		256	BM00.00	BM15.15	
SBM	SBM_			256	SBM00.00	SBM15.15	
BI	BI_			256	BI00.00	BI15.15	in Profi Control 680t: byte markers in Profi Control 680t-SL: external inputs
BO	BO_			256	BO00.00	BO15.15	in Profi Control 680t: not used Profi Control 680t-SL: external outputs
DB0	DB0_	data proc. ranges for data modules	byte	256	DB000.00	DB015.15	These operands are available for monitor versions 4.17 and higher. They can be used like normal byte markers.
DB1	DB1_			256	DB100.00	DB115.15	
DB2	DB2_			256	DB200.00	DB215.15	
DB3	DB3_			256	DB300.00	DB315.15	
DB4	DB4_			256	DB400.00	DB415.15	
DB5	DB5_			256	DB500.00	DB515.15	
DB6	DB6_			256	DB600.00	DB615.15	
DB7	DB7_			256	DB700.00	DB715.15	
BR	BR_	markers	word	256	BR00.00	BR15.15	Buffered by accu on the device (to test accu function: write bit map to a remanent byte marker once and test at start of program)
SBR	SBR_			256	SBR00.00	SBR15.15	
ABM	ABM_			256	ABM00.00	ABM15.15	
BC	BC_			256	BC00.00	BC15.15	
SBC	SBC_			256	SBC00.00	SBC15.15	
BD	BD_			256	BD00.00	BD15.15	
SBD	SBD_			256	SBD00.00	SBD15.15	
LBM	LBM_			256	LBM00.00	LBM15.15	
FBM	FBM_			256	FBM00.00	FBM15.15	
C	C_	counters	word	32	C00.00	C01.15	
PT	PT_	timers		128	PT00.00	PT07.15	
PC	PC_	clock	byte	4	PC00.00	PC00.03	
PP	PP_	pulse	bit	128	PP00.00	PP07.15	
PL	PL00.00 PL00.01	logical 0 logical 1		1 1	PL00.00 PL00.01		
AI	AI_	inputs	word	32	AI0.0	AI7.3	
AO	AO_	outputs		32	AO0.0	AO7.3	
ERR	ERR00.00	error	byte	1	"System error messages (see appendix ""D. Error handling"")"		

4.2.2. Short description of local operands

All addresses which can be addressed for signal processing or data storing in the user program are called operands. They are "operated" with.

Digital inputs and outputs

They plug into the rack as modules if and when needed. You decide on the configuration yourself by selecting the modules you need. These inputs and outputs are updated between program cycles via the process image.

-Inputs

They "read" the signals of switches, sensors, initiators, etc. and report their signal status to the CPU via the control bus.

-Outputs

They output control signals (on, off, etc.) to relays, contactors, solenoids, etc. Controlled by the user program, the CPU transmits the signals to the output modules via the device bus. At the same time, the signals are also transmitted to RAM memory cells, which are addressed under the same address on the CPU.

Bit and byte markers

There are 1536 bit markers in 6 groups and 3072 byte markers in 11 groups available on the CPU (see chapter "4.2.1. Overview) for storing (marking) current data. Of these, 512 bit markers and 2304 byte markers are remanent if there is an accu on the CPU.

Timers

The controller has a standard set of 128 software timers (PT00.00-PT07.15).

The times range from 10ms - 65535s. These timers can be programmed with raising or falling delay, or as clock pulse or pulse generators respectively. If desired they can be remanent, i.e. buffered by the accu.

Counters

32 counters with a resolution of 16 bit (0-65535) can be programmed as up or down counters. They too can be remanent if desired.

Analogue inputs and outputs

They plug into the rack as modules if and when needed. You define the configuration yourself by selecting the relevant modules. They are updated between program cycles as process image.

-Inputs

They "read" the analogue values of temperatures, liquid levels, speeds etc. Analogue-to-digital conversion is done by the processor. The digital value can be processed further in the program.

-Outputs

They output analogue control signals to drives etc. Controlled by the user program, the signals are transmitted to the device bus by the CPU. The digital-to-analogue converter is located on the module itself. The analogue signal is picked up at the relevant terminals.

System error marker "ERR00.00"

The monitor program writes all system errors found are written into byte operand (8 bit) "ERR00.00". They can be read and evaluated by the user program (see appendix "D. Error handling").

4.3. Commands overview

The following overview contains information about all available commands including the possible types of addressing, the necessary memory capacity and the processing time.



Please take special care of only linking operands of the same size (bit, byte or word). Mixed operations must be avoided as they may lead to wrong results.

Programming external operands in Profi Control 680I (not Profi Control 680I-SL)

The execution times listed in the tables below also apply to external operands (see chapters 5 and 6) if there is a PROFIBUS connection.



When the network or a partner are switched on it may take a few seconds until the interconnections are established. During this time, the signal states of external inputs are not current values. The same applies to the addressing of external outputs.



For further information about programming, use of the commands, and example programs see our "Programming manual", E 417 GB.

Commands

4.3.1. Logical operations commands

Logic commands control logical operations between operands including the assignment of results.

They can be executed with bit, byte and word operands.

Load commands

Comm	Operand (example)	Byte	Tim [μs]	Function	C (*)	Z (*)
L	I00.00	4	0.45	load 1bit address	--	yes
	BM00.00	4	0.45	load 8bit address	--	yes
	I00	4	0.3	load 8bit constant	--	yes
	I00.00[10]	8	0.75	load 1bit address with constant offset	--	yes
	I00.00[BM01.00]	14	3	load 1bit address with variable offset	--	yes
	BM00.00[10]	8	0.75	load 8bit address with constant offset	--	yes
	BM00.00[BM01.00]	14	3	load 8bit slave address with variable offset	--	yes
LN	I00.00	6	0.6	load negation 1bit address	--	yes
	BM00.00	6	0.6	load negation 8bit address	--	yes
	I00.00[10]	10	0.9	load negation 1bit address with constant offset	--	yes
	I00.00[BM01.00]	16	3.2	load negation 1bit address with variable offset	--	yes
	BM00.00[10]	10	0.9	load negation 8bit address with constant offset	--	yes
	BM00.00[BM01.00]	16	3.2	load neg. 8bit slave addr. with variable offset	--	yes
LD	BM00.00	4	0.45	load 16bit address (even address)	--	yes
	BM00.01	10	3	load 16bit address (odd address)	--	yes
	I0000	4	0.3	load 16bit constant	--	yes
	BM00.00[10]	16	3	load 16bit address with constant offset	--	yes
	BM00.00[BM01.00]	22	5.3	load 16bit slave address with variable offset	--	yes

*) Influence on (C)arry and (Z)ero bit:

- no change
- yes defined flag alteration
- ++ undefined flag alteration

Commands

AND commands

Comm	Operand (example)	Byte	Tim [μs]	Function	C *)	Z *)
A	I00.00	4	0.45	AND 1bit address	--	yes
	BM00.00	4	0.45	AND 8bit address	--	yes
	I00	4	0.3	AND 8bit constant	--	yes
	I00.00[10]	10	0.9	AND 1bit address with constant offset	--	yes
	I00.00[BM01.00]	16	3.2	AND 1bit address with variable offset	--	yes
	BM00.00[10]	10	0.9	AND 8bit address with constant offset	--	yes
	BM00.00[BM01.00]	16	3.2	AND 8bit slave address with variable offset	--	yes
AN	I00.00	8	0.75	AND negation 1bit address	--	yes
	BM00.00	8	0.75	AND negation 8bit address	--	yes
	I00.00[10]	12	1.1	AND neg. 1bit address with constant offset	--	yes
	I00.00[BM01.00]	18	3.3	AND neg. 1bit address with variable offset	--	yes
	BM00.00[10]	12	1.1	AND neg. 8bit address with constant offset	--	yes
	BM00.00[BM01.00]	18	3.3	AND neg. 8bit slave addr. with variable offset	--	yes
	BM00.00[SLA01.00]	18	3.3	AND neg. 8bit slave addr. with variable offset	--	yes
AD	BM00.00	6	0.6	AND 16bit address (even address)	--	yes
	BM00.01	10	2.6	AND 16bit address (odd address)	--	yes
	I0000	4	0.3	AND 16bit constant	--	yes

*) influence on (C)arry and (Z)ero bit: -- no change
 yes defined flag alteration
 ++ undefined flag alteration

OR commands

Comm	Operand (example)	Byte	Tim [μs]	Function	C *)	Z *)
O	I00.00	4	0.45	OR 1bit address	--	yes
	BM00.00	4	0.45	OR 8bit address	--	yes
	I00	4	0.3	OR 8bit constant	--	yes
	I00.00[10]	10	0.9	OR 1bit address with constant offset	--	yes
	I00.00[BM01.00]	16	3.2	OR 1bit address with variable offset	--	yes
	BM00.00[10]	10	0.9	OR 8bit address with constant offset	--	yes
	BM00.00[BM01.00]	16	3.2	OR 8bit slave address with variable offset	--	yes
ON	I00.00	8	0.75	OR negation 1bit address	--	yes
	BM00.00	8	0.75	OR negation 8bit address	--	yes
	I00.00[10]	12	1.1	OR negation 1bit address with constant offset	--	yes
	I00.00[BM01.00]	18	3.3	OR negation 1bit address with variable offset	--	yes
	BM00.00[10]	12	1.1	OR negation 8bit address with constant offset	--	yes
	BM00.00[BM01.00]	18	3.3	OR neg. 8bit slave addr. with variable offset	--	yes
OD	BM00.00	6	0.6	OR 16bit address (even address)	--	yes
	BM00.01	10	0.3	OR 16bit address (odd address)	--	yes
	I0000	4	2.6	OR 16bit constant	--	yes

*) Influence on (C)arry and (Z)ero bit:

- no change
- yes defined flag alteration
- ++ undefined flag alteration

Commands

Exclusive OR commands

Comm	Operand (example)	Byte	Tim [μs]	Function	C *)	Z *)
XO	I00.00	4	0.45	Exclusive-OR 1bit address	--	yes
	BM00.00	4	0.45	Exclusive-OR 8bit address	--	yes
	I00	4	0.3	Exclusive-OR 8bit constant	--	yes
	I00.00[10]	10	0.9	Exclusive-OR 1bit addr. with constant offset	--	yes
	I00.00[BM01.00]	16	3.2	Exclusive-OR 1bit addr. with variable offset	--	yes
	BM00.00[10]	10	0.9	Exclusive-OR 8bit addr. with constant offset	--	yes
	BM00.00[BM01.00]	16	3.2	Excl.-OR 8bit slave addr. with variable offset	--	yes
XON	I00.00	8	0.75	Exclusive-OR negation 1bit address	--	yes
	BM00.00	8	0.75	Exclusive-OR negation 8bit address	--	yes
	I00.00[10]	12	1.1	Excl.-OR neg. 1bit addr. with constant offset	--	yes
	I00.00[BM01.00]	18	3.3	Excl.-OR neg. 1bit addr. with variable offset	--	yes
	BM00.00[10]	12	1.1	Excl.-OR neg. 8bit addr. with constant offset	--	yes
	BM00.00[BM01.00]	18	3.3	Excl.-OR neg. 8bit slave addr. with var. offset	--	yes

*) Influence on (C)arry and (Z)ero bit: -- no change
yes defined flag alteration
++ undefined flag alteration

Assignments and set commands

Comm	Operand (example)	Byte	Tim [μs]	Function	C (*)	Z (*)
=	I00.00	4	0.45	Assignment 1bit address	--	--
	BM00.00	4	0.45	Assignment 8bit address	--	--
	O00.00[10]	8	0.75	Assignment 1bit address with constant offset	--	--
	O00.00[BM01.00]	14	3.0	Assignment 1bit address with variable offset	--	--
	BM00.00[10]	8	0.75	Assignment 8bit address with constant offset	--	--
	BM00.00[BM01.00]	14	3.0	Assignment 8bit slave addr. with var. offset	--	--
=N	I00.00	16	1.4	Assignment negation 1bit address	--	yes
	BM00.00	16	1.4	Assignment negation 8bit address	--	yes
	O00.00[10]	20	1.8	Assignment neg. 1bit addr. with const. offset	--	yes
	O00.00[BM01.00]	26	4.9	Assignment neg. 1bit addr. with var. offset	--	yes
	BM00.00[10]	12	1.1	Assignment neg. 8bit addr. with const. offset	--	yes
	BM00.00[BM01.00]	18	3.3	Assignment neg. 8bit sl. addr. with var. offset	--	yes
=D	BM00.00	4	0.45	Assignment 16bit address (even address)	--	--
	BM00.01	18	3.6	Assignment 16bit address (odd address)	--	--
	BM00.00[10]	16	3.0	Assignment 16bit addr. with constant offset	--	--
	BM00.00[BM01.00]	22	5.3	Assignment 16bit slave addr. with var. offset	--	--
S	O00.00	12	1.2	Conditional set 1bit address	--	yes
R	O00.00	10	0.9	Conditional reset 1bit address	--	yes
=1	O00.00	8	0.45	Unconditional set 1bit address	--	yes
=0	O00.00	8	0.45	Unconditional reset 1bit address	--	yes

*) Influence on (C)arry and (Z)ero bit: -- no change
 yes defined flag alteration
 ++ undefined flag alteration

Commands

4.3.2. Arithmetics commands

Comm	Operand (example)	Byte	Time [μs]	Function	C (*)	Z (*)
ADD	BM00.00 100	4	0.45	Addition 8bit address	yes	yes
		4	0.3	Addition 8bit constant	yes	yes
ADDD	BM00.00 BM00.01 10000	4	0.45	Addition 16bit address (even address)	yes	yes
		10	3	Addition 16bit address (odd address)	yes	yes
		4	0.3	Addition 16bit constant	yes	yes
SUB	BM00.00 100	4	0.45	Subtraction 8bit address	yes	yes
		4	0.3	Subtraction 8bit constant	yes	yes
SUBD	BM00.00 BM00.01 10000	4	0.45	Subtraction 16bit address (even address)	yes	yes
		10	2.1	Subtraction 16bit address (odd address)	yes	yes
		4	0.3	Subtraction 16bit constant	yes	yes
MUL	BM00.00 100	8	3	Multiplication 8bit address	yes	yes
		10	3	Multiplication 8bit constant	yes	yes
MULD	BM00.00 BM00.01 10000	8	3	Multiplication 16bit address (even address)	yes	yes
		12	3.5	Multiplication 16bit address (odd address)	yes	yes
		8	3	Multiplication 16bit constant	yes	yes
DIV	BM00.00 100	8	4	Division 8bit address	yes	yes
		8	3.8	Division 8bit constant	yes	yes
DIVD	BM00.00 BM00.01 10000	8	4	Division 16bit address (even address)	yes	yes
		12	3.5	Division 16bit address (odd address)	yes	yes
		8	3.8	Division 16bit constant	yes	yes

~) Only approximative indication of time, as the time is dependent on the operand because of iterative processing

*) Influence on (C)arry and (Z)ero bit: -- no change
 yes defined flag alteration
 ++ undefined flag alteration

4.3.3. Comparison commands

Comm	Operand (example)	Byte	Tim [μs]	Function	C (*)	Z (*)
CMP	BM00.00 100	4	0.45	Compare 8bit address	yes	yes
		4	0.3	Compare 8bit constant	yes	yes
CMPD	BM00.00 BM00.01 10000	6	0.6	Compare 16bit address (even)	yes	yes
		10	2.6	Compare 16bit address (odd)	yes	yes
		4	0.45	Compare 16bit constant	yes	yes
CMP=	BM00.00 100	16	1.3	Compare if equal 8bit address	yes	yes
		16	1.1	Compare if equal 8bit constant	yes	yes
CMPD	BM00.00 BM00.01 10000	18	1.4	Compare if equal 16bit address (even)	yes	yes
		22	3.3	Compare if equal 16bit address (odd)	yes	yes
		16	1.3	Compare if equal 16bit constant	yes	yes
CMP<	BM00.00 100	16	1.3	Compare if unequal 8bit address	yes	yes
		16	1.1	Compare if unequal 8bit constant	yes	yes
CMPD	BM00.00 BM00.01 10000	18	1.4	Compare if unequal 16bit address (even)	yes	yes
		22	3.3	Compare if unequal 16bit address (odd)	yes	yes
		16	1.3	Compare if unequal 16bit constant	yes	yes
CMP<	BM00.00 100	16	1.3	Compare if < or = 8bit address	yes	yes
		16	1.1	Compare if < or = 8bit constant	yes	yes
CMPD	BM00.00 BM00.01 10000	18	1.4	Compare if < or = 16bit address (even)	yes	yes
		22	3.3	Compare if < or = 16bit address (odd)	yes	yes
		16	1.3	Compare if < or = 16bit constant	yes	yes
CMP>	BM00.00 100	16	1.3	Compare if > or = 8bit address	yes	yes
		16	1.1	Compare if > or = 8bit constant	yes	yes
CMPD	BM00.00 BM00.01 10000	18	1.4	Compare if > or = 16bit address (even)	yes	yes
		22	3.3	Compare if > or = 16bit address (odd)	yes	yes
		16	1.3	Compare if > or = 16bit constant	yes	yes

*) Influence on (C)arry and (Z)ero bit:
 -- no change
 yes defined flag alteration
 ++ undefined flag alteration

Commands

4.3.4. Shift and rotation commands

Comm	Operand (example)	Byte	Tim [μs]	Function	C *)	Z *)
LSL	Accu	2	0.3	Log. shift left in accu, 8bit	yes	yes
LSR	Accu	6	0.45	Log. shift right in accu, 8bit	yes	yes
LSLD	Accu	2	0.15	Log. shift left in accu, 16bit	yes	yes
LSRD	Accu	2	0.15	Log. shift right in accu, 16bit	yes	yes
LSLM	BM00.00	10	1.1	Log. shift left in 8bit address	yes	yes
LSRM	BM00.00	10	1.1	Log. shift right in 8bit address	yes	yes
LSLDM	BM00.00	10	1.1	Log. shift left in 16bit address	yes	yes
	BM00.01	14	5.4	Log. shift left in 16bit address (odd)		
LSRDM	BM00.00	10	1.1	Log. shift right in 16bit address	yes	yes
	BM00.01	14	5.4	Log. shift right in 16bit address (odd)		
ROL	Accu	2	0.15	Roll left in accu, 8bit	yes	yes
ROR	Accu	10	0.75	Roll right in accu, 8bit	yes	yes
ROLD	Accu	2	0.15	Roll left in accu, 16bit	yes	yes
RORD	Accu	20	2.1	Roll right in accu, 16bit	yes	yes
ROLM	BM00.00	10	1.1	Roll left in 8bit address	yes	yes
RORM	BM00.00	14	1.4	Roll right in 8bit address	yes	yes
ROLDM	BM00.00	10	1.1	Roll left in 16bit address	yes	yes
	BM00.01	14	4.5	Roll left in 16bit address (odd)		
RORDM	BM00.00	26	2.1	Roll right in 16bit address	yes	yes
	BM00.01	34	5.1	Roll right in 16bit address (odd)		

*) Influence on (C)arry and (Z)ero bit:
 -- no change
 yes defined flag alteration
 ++ undefined flag alteration

4.3.5. Byte and flag manipulation

Comm	Operand (example)	Byte	Tim [μs]	Function	C (*)	Z (*)
INC	BM00.00	4	0.45	Increment 8bit address	--	yes
DEC	BM00.00	4	0.45	Decrement 8bit address	--	yes
INCD	BM00.00	10	1.1	Increment 16bit address	--	yes
	BM00.01	14	5.4	Increment 16bit address (odd)	--	yes
DECD	BM00.00	10	1.1	Decrement 16bit address	--	yes
	BM00.01	14	5.4	Decrement 16bit address (odd)	--	yes
CLR	BM00.00	4	0.45	Clear 8bit address	--	--
NOP		2	0.15	Do-nothing operation	--	--
SEC		2	0.15	Set Carry bit = 1	yes	--
CLC		2	0.15	Clear Carry bit = 0	yes	--

*) Influence on (C)arry and (Z)ero bit:
 -- no change
 yes defined flag alteration
 ++ undefined flag alteration

4.3.6. Module calls

Comm	Operand (example)	Byte	Tim [μs]	Function	C (*)	Z (*)
JPP	Program module	14	10	Jump to program module	--	--
JPCP	Program module	18	10.5	Conditional jump if yes to program module	--	--
JPF	Function module	18	17	Jump to function module	--	--
JPCF	Function module	26	17	Conditional jump if yes to function module	--	--
JPK	KUBES module	18	17	Jump to KUBES module	--	--
JPCK	KUBES module	26	17	Conditional jump if yes to KUBES module	--	--
JPINIT	Init module	14	10	Jump to Init module	--	--

*) Influence on (C)arry and (Z)ero bit:
 -- no change
 yes defined flag alteration
 ++ undefined flag alteration

Commands

4.3.7. Jump commands

Comm	Operand (example)	Byte	Tim [μs]	Function	C (*)	Z (*)
JP	Jump mark	4	0.3	Unconditional jump	--	--
JPC	Jump mark	8	0.6	Conditional jump if yes (logical 1)	--	--
JPCN	Jump mark	8	0.6	Conditional jump if no (logical 0)	--	--
JP=	Jump mark	4	0.3	Jump if equal	--	--
JP<>	Jump mark	4	0.3	Jump if unequal	--	--
JP<	Jump mark	4	0.3	Jump if smaller	--	--
JP>	Jump mark	4	0.3	Jump if greater	--	--
JP<=	Jump mark	4	0.3	Jump if smaller or equal	--	--
JP>=	Jump mark	4	0.3	Jump if greater or equal	--	--
JPCS	Jump mark	4	0.45	Jump if Carry bit = 1	--	--
JPC	Jump mark	4	0.45	Jump if Carry bit = 0	--	--
JPZS	Jump mark	4	0.45	Jump if Zero bit = 1	--	--
JPZC	Jump mark	4	0.45	Jump if Zero bit = 0	--	--
JP+	Jump mark	4	0.45	Jump if positive (two's complement)	--	--
JP-	Jump mark	4	0.45	Jump if negative (two's complement)	--	--

#Times: the higher value is valid if there is a jump,
the smaller value is valid if there is no jump

*) Influence on (C)arry and (Z)ero bit: -- no change
 yes defined flag alteration
 ++ undefined flag alteration

4.3.8. Copy and BCD commands

Comma	Operand (example)	Byte	Tim [μs]	Function	C (*)	Z (*)
C1T8	I00.00	16	12	Copy 1bit addresses to 8bit accu	--	--
C8T1	O00.00	16	10.6	Copy 8bit accu to 1bit addresses	--	--
C1T16	I00.00	16	21.5	Copy 1bit addresses to 16bit accu	--	--
C16T1	O00.00	16	19.2	Copy 16bit accu to 1bit addresses	--	--
BINBCD3	Accu	4	5.2	Binary-BCD conversion (3 decades)	++	++
BCDBIN3	Accu	4	5.1	BCD-binary conversion (3 decades)	++	++

*) Influence on (C)arry and (Z)ero bit: -- no change
 yes defined flag alteration
 ++ undefined flag alteration

4.3.9. Programmable pulses, timers and counters

Co	Operand (example)	Byte	Tim [μs]	Function	C (*)	Z (*)
=	PP00.00	16	5	Programmable pulse at positive slope	++	++
=N	PP00.00	16	5	Programmable pulse at negative slope	++	++
L A O	PP00.00	4 4 4	0.45 0.45 0.45	Logical operation with pulse signal	--	++
IN AN ON	PP00.00	6 8 8	0.6 0.75 0.75	Logical operation with pulse signal, negated	--	++
=	PT00.00:100*10ms:E:R	14	14	Progr. time with constant time value (log.1=On)	++	++
=	PT00.00:BM00.00*10ms:E:R	20	17.5	Progr. time with variable time value (log.1=On)**)	++	++
=N	PT00.00:100*10ms:E:R	18	15	Progr. time with constant time value (log.0=On)	++	++
=N	PT00.00:BM00.00*10ms:E:R	24	18.5	Progr. time with variable time value (log.0=On)**)	++	++
=TH	PT00.00	14	11.5	Time halt (actual value is kept)	++	++
L A O	PT00.00	4 4 4	0.45 0.45 0.45	Logical operation with time output	--	++
IN AN ON	PT00.00	6 8 8	0.6 0.75 0.75	Logical operation with time output, negated	--	++
=	C00.00:100:V:R	14	9.6	Set progr. counter with const. cnt. value (log.1=On)	++	++
=	C00.00:BM00.00:V:R	20	10	Set progr. counter with var. cnt. value (log.1=On)**)	++	++
=N	C00.00:100:V:R	18	10.5	Set progr. counter with const. cnt. value (log.0=On)	++	++
=N	C00.00:BM00.00:V:R	24	11	Set progr. counter with var. cnt. value (log.0=On)**)	++	++
=C	C00.00	14	10	Transfer of the clock pulse (count)	++	++
L A O	C00.00	4 4 4	0.45 0.45 0.45	Logical operation with counter output	--	++
IN AN ON	C00.00	6 8 8	0.6 0.75 0.75	Logical operation with counter output, negated	--	++

:R this entry at the last position renders counters and timers to become remanent (buffered by the accu on the CPU) when set.

**) No external operand (PROFIBUS) can be applied for the variable timer or counter value.

*) Influence on (C)arry and (Z)ero bit:

- no change
- yes defined flag alteration
- ++ undefined flag alteration

Commands

4.3.10. Special commands

Comm	Operand	Byte	Time [μs]	Function	C (*)	Z (*)
O_OFF	-			Switches the power pack of all outputs on.		
O_ON	-			Switches the power pack of all outputs off.		
RESET	-			Resets all non-remanent outputs, markers, timers and counters and stops the program run.		
WAIT	n			wait loop. Latency = n (1..6) * 10 ms]		

*) Influence on (C)arry-and (Z)ero-Bit: -- no change
 yes defined flag alteration
 ++ undefined flag alteration

4.3.11. Initialisation module commands

Initialisation modules are a special variety of modules. None of the commands described previously in this chapter can be used here and vice versa: the commands listed in the table below can only be used in conjunction with the initialisation modules.

Operand	Data type	Value	Function
O00.00	BIT	1 1,0,1,1,... [16],1	Write logical value into 1bit address Write logical values into 1bit and subsequent addresses Write log. value into 1bit and the 15 following addresses
BM00.00	BYTE	75 \$4B %01001011 1,18,0,1,25... [8],128 "KUHNKE"	Write decimal value into 8bit address Write hexadecimal value into 8bit address Write binary value into 8bit address Write values into 8bit and subsequent addresses Write value into 8bit and the following 7 addresses Write text into 8bit and subsequent addresses
BM00.00	WORD	19285 \$4B55 %0100101101010101 1,18,0,1,25... [8],13283	Write decimal value into 16bit address Write hexadecimal value into 16bit address Write binary value into 16bit address Write values into 16bit and subsequent addresses Write value into 16bit and the following 7 addresses
BM00.00	TEXT	"KUHNKE" "KUHNKE", " MALENTE" ...	Write text as from the specified address Write texts as from the specified address

4.3.12. Data module commands

Cmnd	Operand	Byte	Time [μs]	Function	C (*)	Z (*)
LoadDB	x,<name>	12	150	Load data module <name> into DBx00.00...15.15	++	++
	byte1,<name>			Load data module <name> into DBx00.00...15.15 (x = value 0...7 in byte1)		
	x,byte2			Load data module number y (y = value 1...255 in byte2) into DBx00.00...15.15 (x = 0...7)		
	byte1,byte2			Load data module number y (y = value 1...255 in byte2) into DBx00.00...15.15 (x = val. 0...7 in byte1)		
StoreDB	x,<name>	12	150	Store DBx00.00...15.15 (x = 0...7) in data module <name>	++	++
	byte1,<name>			Store DBx00.00...15.15 (x = value 0...7 in byte1) in data module <name>		
	x,byte2			Store DBx00.00...15.15 (x = 0...7) in data module y (y = value 1...255 in byte2)		
	byte1,byte2			Store DBx00.00...15.15 (x = value 0...7 in byte1) in data module number y (y = value 1...255 in byte2)		

*) Influence on (C)arry and
(Z)ero bit:

-- no alteration
yes defined flag alteration
++ undefined flag alteration

Data modules are available as from monitor version 4.17.

Commands

4.4. Registers

As a matter of size, there are three types of operands in the controller:

- 1bit operands
- 8bit operands (bytes)
- 16bit operands (words)

The accumulator in the CPU can be used as 1bit, 8bit or 16bit register.



Please do not confuse: the term "accu(mulator)" in the software part stands for a general-purpose register in the processor. In the hardware part it stands for a rechargeable battery.

- 1bit operands are used for internal byte operations. Only bit 7 of the 8bit accu is evaluated, however.
- For 16bit operands, a 16bit accu is used which uses the above-mentioned 8bit accu as lowbyte. Word processing is set off by the commands that have a "D" as their last sign.



In order to prevent mistakes, we recommend that you do not use different types of operands within operations that belong together.

4.5. Addressing

There are two different ways of assigning the value of the operand:

- Absolute value (voltage or current values, or constant)
- Contents of an operand (bit, byte, word)

4.5.1. Address mnemonic

The operand addresses are indicated as mnemonics, e.g. BM00.00, O00.00, PT00.00. The actual address management of the processor remains invisible.

Command line

```
L      BM00.00
```

therefore stands for loading the contents of a memory location with the mnemonic name "BM00.00".

4.5.2. Offset addressing

You can add an offset to the absolute addresses of the local operands. The address is then made up by adding absolute address and offset.

Command line

```
L      BM00.00 [BM00.01]
```

means that the value in BM00.01 (offset) is added to the address of BM00.00. The resulting new address then responds to the load command.



The value of the offset should be selected in a way that excludes exceeding the relevant operand range (max. 256 addresses).

Reason: Exceeding the operand range leads to reading (with read commands L, A, O...) from or writing (with assignment commands =, =N) into an operand from another range. This may lead to unintended machine functions or to program destruction.

Examples:

L	I00.00[5]	is the same as	L	I00.05
=	O01.00[6]	is the same as	=	O01.06
=	BM01.00[17]	is the same as	=	BM02.01

External operands used as offset

External operands (PROFIBUS inputs, see chapter 5) cannot be used for offset addressing.

Example:

L ~~003a00.00~~[3]

However, external byte operands can be used as variable offset.

Example:

L BR01.00[BIO3a00.]



These limitations apply to Profi Control 680I only. The external operands of Profi Control 680I-SL (BI... and BO...) can be used with offset without any restrictions.

4.5.4. Types of addressing, overview

The load command is taken as a example to give an overview of the various types of addressing.

Load contents of an operand		
L	I00.00	1bit address
L	BM00.00	8bit address
LD	BM00.00	16bit address
Load constant value		
8bit constant (0...255):		
L	100	decimal
L	\$64	hexadecimal
L	%01100100	binary
L	'A'	ASCII
16bit constant (0...65535):		
LD	10000	decimal
LD	\$3FEA	hexadecimal
LD	%0010011100010000	binary
LD	4.5V	Voltage (-10...+10V)
LD	5mA	Current(-20...+20mA)
Load contents of an offset-addressed operand		
with constant offset (0...255):		
L	I00.00[10]	1bit address
L	BM00.00[10]	8bit address
LD	BM00.00[10]	16bit address
with variable offset in the local operand (0...255):		
L	I00.00[BM01.00]	1bit address
L	BM00.00[BM01.00]	8bit address
LD	BM00.00[BM01.00]	16bit address
with variable offset in the external operand (0...255)		
L	I00.00[BIO3a00.]	1bit address
L	BM00.00[BIO3a00.]	8bit address
LD	BM00.00[BM01.00]	16bit address

5. Profi Control 680I in PROFIBUS networks

Profi Control 680I is a combimaster. As a combimaster it can use three different bus protocols for communication with its partner stations. In multiple-protocol networks, it is even capable of using several protocols in parallel, as it were, depending on the type of station it is to exchange data with.

5.1. Basic information

What is PROFIBUS?

PROFIBUS is a field bus. Its name results from an abbreviation of the term "Process Field Bus". It was developed for networking controllers (e.g. Profi Control 680I, PC Control 645, etc.) and for providing a connection to the field level, i.e. to sensors and actuators, via decentralised input/output units (Profi I/O 680E, Profi I/O 680S...).

It further acts as the interface to the coordinating level where one or several central computers may be implemented to control the overall process.

Bus protocols

PROFIBUS-FMS, PROFIBUS-DP and SINEC L2-DP.

Open communication

The principle of open communication is meant to guarantee interconnections between devices supplied by different manufacturers. This field bus has been standardised by Euro-norm EN 50170.

Topology

PROFIBUS is designed as a line. A tree structure can also be established, however. The number of stations on any one line is limited to 32. Where that is not sufficient, a second line can be opened which is connected to the first line by a bidirectional line amplifier (repeater). Repeaters also count as physical stations so that the number of "real" stations on one line is reduced to 31 (30, with 2 repeaters).

The number of bus stations can be increased to up to 122 by applying up to 3 repeaters.

Station address

Every logical PROFIBUS station is assigned its own station address under which it can be addressed by the other stations. The address range is 0 to max. 126. Profi Control 680I can use addresses 0...63 for communication.

Network configuration

VEBES, the network configuration software, is the right tool to use for setting up a PROFIBUS network, if a Kuhnke controller is the master.

Use VEBES to set the bus parameters, define stations, and specify the interconnections between stations.

Communication

By default, PROFIBUS communication is based on a process image stored in the master. The process image contains the status information of all external operands. It is automatically and acyclically updated, i.e. independent of the user program.



see chapter "5.2. External operands"

In FMS networks, there is an extra event-controlled type of communication, called block transfer, which can be used by the master, or rather the master's user program, to read or overwrite station data.



For a detailed description of block transfer communication refer to instruction manual E 365 GB, "PROFIBUS".

5.2. External operands

External operands are read as inputs or actuated as outputs via PROFIBUS. There are different sources or destinations resp.:

- decentralised input/output devices (e.g. KUAX 680S)
- other programmable logic controllers (e.g. Profi Control 680I or KUAX 657P)
- valve islands
- positioning units
- etc.

VEBES creates the list of external operands

VEBES (see instruction manual E 315 GB, "VEBES") is used to configure the network. This process includes creating projects for all Kuhnke controllers in the network which are then edited (i.e. the program are written) in the KUBES programming environment (see instruction manual E 327 GB, "KUBES").

Part of network configuration via VEBES is to define which stations are to communicate with each other and which objects (external operands) are to be used.

VEBES adds these external operands to the list of operands of every project and distributes them across the process image memory. This allows you to use them just like local operands in your program. Mnemonics (via the Symbol Table) are also supported.

PROFIBUS updates the process image

By task changes in the controller, PROFIBUS automatically transfers data to and from the process image as often as possible. Task changes are timer-controlled and interrupt the processing of the current user program module (exception: timer modules and initialisation modules are not interrupted).

Data consistency

see chapter "5.2.3. Data consistency ensured by OS_CRIT".



5.2.1. Process image of a slave

Profi I/O 680S is a typical PROFIBUS slave.

It is a decentralised input/output unit. The size of the process image to be reserved by the master depends on the configuration.

Modules are plugged in from left to right. Every module occupies one byte of process image space.

5.2.1.1. Specification of operands

I	03	a	02	.5	
				Channel number (input or output) 0...7 (bit)	
			Group address 00...max. 15 (byte)		
				PROFIBUS line address	
					Station address 00...99
Type (I=input, BI=byte input, O=output, BO=byte output)					

Type

Represents the inputs and outputs that the slave is equipped with:

- inputs are read via PROFIBUS
- outputs are actuated via PROFIBUS

Station address

Address of the PROFIBUS station to be communicated with. Profi Control 680I can communicate with any station that has been assigned an address between 00 and 99.

PROFIBUS line address

The line address of the built-in PROFIBUS port is always "a".

As from monitor version 6.10 an extra DP master module is supported which controls network "b". Further requirements are: VEBES version 3.10 or higher, and KUBES version 5.20 or higher.

Group address

Every 8 channels combine into a group, representing one byte. In the case of Profi I/O 680S, the groups of modules plugged into the device are numbered through from left to right.

Channel number

Only for bit addressing. The channel number is the same as the module's channel number, i.e. the input or the output.

5.2.1.2. Addressing

Bit addressing

(addressing one channel)

The operand is addressed with "I" (input) or "O" (output). The channel number selects the relevant bit.

Examples:

```
L  I03a02.5    ;read external input
=  O01.02      ; and write into local output
```

```
L  I00.03      ;read local input
=  O02a01.7    ; and write into ext. output
```

Byte addressing

(addressing all channels of a group)

Addresses 1 byte (8 bit) by one command. Inputs are specified as "BI", outputs as "BO". The channel number is not required.

Examples:

```
L  BI03a02.    ;read external byte input
=  BM01.00     ; and write into byte marker
```

```
L  BM00.02     ;read byte marker
=  BO03a04.    ; and write into ext. byte output
```

Word addressing (addressing all channels of 2 groups)

Addresses 1 word (2 byte or 16 bit) by one command. Inputs are specified as "BI", outputs as "BO". The channel number is not required.

Example:

```
LD BI03a02.      ;read 2 external byte inputs
=D BM01.00        ; and write into 2 byte markers

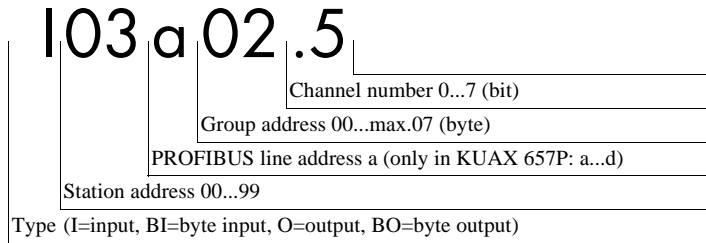
LD BM00.02        ;read 2 byte markers
=D BO03a04.        ; and write into 2 ext. byte outputs
```

5.2.2. Process image of an FMS master

The process image of a communicating station with a controller profile can be configured and adapted to the current connection. The following limitations apply:

- Any one process image connection can carry no more than 56 byte of user data
- The maximum amount of data carried by a communication link (binary and by byte, i.e. inputs and outputs) is 100 byte
- The total process image for all process image connections of a single Profi Control 680I has 496 byte

5.2.2.1. Specification of operands



Type

The type indicates the relation between operand and master:

- inputs are read via PROFIBUS
- outputs are actuated via PROFIBUS

Station address

Address of the PROFIBUS station to be communicated with. Profi Control 680I can communicate with any station that has been assigned an address between 00 and 99.

PROFIBUS line address

The line address of the built-in PROFIBUS port is always "a".

As from monitor version 6.10 an extra DP master module is supported which controls network "b". Further requirements are: VEBES version 3.10 or higher, and KUBES version 5.20 or higher.

Group address

Every group has a size of one byte.

Channel number

Selects a bit of the specified group (see below, section "Bit addressing").

5.2.2.2. Addressing

Bit operands in the process image can be addressed by bit, by byte, or by word. Operands declared as byte connections can be addressed by byte and by word only.

Bit addressing

The operand is addressed with "I" (input) or "O" (output). The channel number selects the relevant bit.

Examples:

```
L  I03a02.5    ;read external input
=  O01.02      ; and write into local output
```

```
L  I00.03      ;read local input
=  O02a01.7    ; and write into ext. output
```

Byte addressing

Addresses 1 byte (8 bit) by one command. Inputs are specified as "BI", outputs as "BO". The channel number is not required.

Examples:

```
L  BI03a02.    ;read external byte input
=  BM01.00     ; and write into byte marker
```

```
L  BM00.02     ;read byte marker
=  BO03a04.    ; and write into ext. byte output
```

Word addressing

Addresses 1 word (2 byte or 16 bit) by one command. Inputs are specified as "BI", outputs as "BO". The channel number is not required.

Example:

```
LD BI03a02.    ;read 2 external byte inputs
=D BM01.00     ; and write into 2 byte markers
```

```
LD BM00.02     ;read 2 byte markers
=D BO03a04.    ; and write into 2 ext. byte outputs
```

5.2.3. Data consistency ensured by OS_CRIT

The operating system of Kuhnke controllers such as Profi Control 680I is designed as a multi-tasking system. Tasks such as processing the user program or PROFIBUS communication are acyclic processes. Task changes are timer-controlled, i.e. independent of the processing status of the preceding task.

This frequently leads to the process image being updated by the PROFIBUS task between two user program commands. A possible consequence is that the external operands read by the next user program command are more up to date than the ones read before.

This may cause problems if the different and differing data sets are functionally related. Misinterpretations cannot be excluded. Just think of an analogue 24 bit value prepared by a positioning slave to be read by 2 subsequent commands and then there is a task switchover between these two commands of all possibilities. The analogue value may be entirely misinterpreted.

KUBES module OS_CRIT has been written exactly for this situation. It influences the operating system by preventing task changes within limited sections of the user program. OS_CRIT is delivered with KUBES and called up using a constant value as parameter.



see next page for an example

Example for OS_CRIT function

Prevent change of task (constant = 1)

```
JPK      OS_CRIT  ,  _____  
          1        - | _____ | -  
LD        BI01a00. ; low word  
=D        BM00.00  
LD        BI01a02. ; high word  
=D        BM00.02
```

Allow change of task (constant = 0)

```
JPK      OS_CRIT  ,  _____  
          0        - | _____ | -
```

All PROFIBUS data between the two module calls is consistent, i.e., in the example above, the "low word" and "high word" values have been updated by PROFIBUS at the same time.

OS_CRIT directly influences the operating system. Please make sure to comply with the following terms:



1. Always call up OS_CRIT in pairs (first using constant 1 then constant 0).
2. Keep the program section between the two module calls as small as possible.
3. Task changes are always allowed at the end of a module.

5.2.4. PROFIBUS messages

Profi Control 680I can receive (and partly send) three types of PROFIBUS messages. Special operand ranges are reserved for this purpose during network configuration by VEBES. These are

Type of message:	Operand range:
- Error messages	PFxx.yy
- Status messages	PSaxx.yy
- Event notifications	PEaxx.yy



If a program links external operands (e.g. inputs of Profi I/O 680S), take particular care that these operands contain valid data. For example, the value received no longer corresponds to the actual status of the input if the bus connection has been interrupted. To avoid any faulty switching operations we strongly recommend that you carefully evaluate the above PROFIBUS messages via your user program.



For a detailed description of all error messages refer to instruction manual E 365 GB, "PROFIBUS".

6. Profi Control 680I-SL in PROFIBUS networks

Profi Control 680I-SL is a controller that can be integrated into PROFIBUS networks as a DP slave.

6.1. Basic information

What is PROFIBUS?

PROFIBUS is a field bus. Its name results from an abbreviation of the term "Process Field Bus". It was developed for networking controllers (e.g. Profi Control 680I, PC Control 645, etc.) and for providing a connection to the field level, i.e. to sensors and actuators, via decentralised input/output units (Profi I/O 680E, Profi I/O 680S...). It further acts as the interface to the coordinating level where one or several central computers may be implemented to control the overall process.

Bus protocols

PROFIBUS-FMS, PROFIBUS-DP and SINEC L2-DP. The controller described in this chapter supports the PROFIBUS-DP protocol only.

Open communication

The principle of open communication is meant to guarantee interconnections between devices supplied by different manufacturers. This field bus has been standardised by Euro-norm EN 50170.

Topology

PROFIBUS is designed as a line. A tree structure can also be established, however. The number of stations on any one line is limited to 32. Where that is not sufficient, a second line can be opened which is connected to the first line by a bidirectional line amplifier (repeater). Repeaters also count as physical stations so that the number of "real" stations on one line is reduced to 31 (30, with 2 repeaters). The number of bus stations can be increased to up to 122 by applying up to 3 repeaters.

Station address

Every PROFIBUS station is assigned its own station address under which it can be addressed by the other stations. The address range is 0 to max. 126. Profi Control 680I-SL can use addresses 0...125 for communication.

Network configuration

Profi Control 680I-SL is designed for communication with DP masters (class 1). VEBES, the network configuration software, is the right tool to use for setting up a PROFIBUS network, if this master is a Kuhnke controller (e.g. Profi Control 680I, see chapter 5). Use VEBES to set the bus parameters, define stations, and specify the interconnections between stations. If the DP master is supplied by another manufacture, you must use the relevant network configurator.

The configurator entry requires the GSD file (see chapter "6.7. Device master data file KUHN6800.GSD") which contains all necessary device information.

Communication

By default, PROFIBUS communication is based on a process image stored in the master. The process image contains the status information of all external operands. The user defines the size of the process image.

see chapter "6.2.2. External operands"



Master

Actual signal processing is taken care of by a higher-priority controller, the master (class 1). This could be a combimaster Profi Control 680I (version 5.0 or higher), for example. The master connects with Profi Control 680I-SL via the PROFIBUS cable.

The terms master (without any further classification) or DP master used in this text always refer to a master (class 1).



6.2. Data exchange via PROFIBUS-DP

This chapter describes how to integrate Profi Control 680I-SL into PROFIBUS networks and how data is exchanged between the slave and the DP master.

6.2.1. Station address

Profi Control 680I-SL's station address can be set via a coding switch (see chapter "3.5.2. Coding switch of Profi Control 680I-SL"). Station addresses 0...125 are supported.

However, if you set the coding switch to 126, you enable the user program to define the station address. In this case, the address entered as the first parameter into KUBES module "INIT_SL" described below (see chapter 6.2.3. ...) applies.

6.2.2. External operands

Data is exchanged via an interrupt-controlled process image. Via PROFIBUS, the DP master writes the external operands as inputs (as seen by the DP slave) into the process image and read from the process image as outputs.

Process image

Taking the KUBES module described further down in this chapter, the user defines the size of the process image for external operands.

– External inputs

contain the data that the DP master sends to Profi Control 680I-SL. They are stored in operand range BI00.00...max. BI14.15 (max. 240 byte).

– External outputs

contain the data to be sent to the DP master. They are stored in operand range BO00.00...max. BO14.15 (max. 240 byte)

6.2.2.1. External operands (overview)

Process image of the PROFIBUS inputs

BI00.	PROFIBUS inputs (max. 240 byte of dat sent by the DP master)															
BI01.																
.																
.																
BI13.																
BI14.																
BI15.	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
						User_Prm_Data (see chapter "6.3.1.2. Device-specific bus parameters")										
PROFIBUS status (see chapter "6.3.1.3. PROFIBUS status")																

Process image of the PROFIBUS outputs

BO00.	PROFIBUS outputs (max. 240 byte of data sent to the DP master)															
BO01.																
.																
.																
.																
BO13.																
BO14.																
BO15.	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15
Ext_Diag_Data (see chapter "6.5.2. Device-specific diagnostic information", octet 9)																

6.2.3. Interrupt-controlled data exchange

The process image is updated automatically, not depending on the use program cycle. Interrupts stop the user program every time new data has been received from the DP master. Output data is sent back to the master as a response frame. Every interrupt calls up interrupt module 19. The following steps are worked through automatically and in that order:

- Set-off condition:
 - new data received from the DP master
- Output data sent back to the DP master
- Input data written into BI00.00...max. BI14.15
- Interrupt module 19 is called up (required to ensure data consistency, see below "Consistency")

Reducing the interrupt frequency

To relieve the user program, you can set the fourth module parameter to any value between 1...255 ms to reduce the frequency of interrupts. Inputting 0 means that an interrupt is set off every time a frame from the master has been received.



Reducing the interrupt frequency still means that the user program is interrupted only when data frames have been received from the DP Master (during communication).

Consistency

The PROFIBUS connection maintains data consistency in steps of 8 byte. The user program can fall back on this consistency by copying the input and output data contained in interrupt module 19.



You don't need interrupt module 19 if no data consistency is required.

6.2.4. Initialising PROFIBUS via the user program

A KUBES module in the user program is responsible for initialising the PROFIBUS functions.
The module is called INIT_SL and has 5 parameters each set by a byte operand:

```
JPX      INIT_SL  ,  _____
                STATION -|         | - STATUS,
                LEN_IN  -|         | - ,
                LEN_OUT -|         | - ,
                CLOCK   -|         | -
```

- STATION
PROFIBUS station address (0...125).
- LEN_IN
Length, in byte, of the process image containing the input data (0...240) in operand range BI00.00...BI14.15
- LEN_OUT
Length, in byte, of the process image containing the output data (0...240) in operand range BO00.00...BO14.15



Specify the lengths of operand ranges LEN_IN and LEN_OUT in steps of 8 (8, 16, 24...), and input the relevant value into the second and third module parameter. The KUBES module rounds up all values between two complete multiples of 8 to the next higher correct multiple.

- CLOCK
Interrupt frequency (0...255 [ms])
- STATUS
Into this output parameter, the KUBES module writes the initialisation status:

Value	Description	Action
0	Initialising	Continue to cyclically call up the KUBES module
1	Station address > 125	Set right and restart
2	Data length >240 byte	
255	Initialisation complete	Stop calling up the KUBES module

Example program for initialisation

```

===== KUBES =====

                                Organisation module IL

Project : 680I_SL                Network :
Module : ORG                    No.: 1    created : Feb 06 1998 11:37
User :                          modified : Feb 06 1998 14:52

=====
1: ; Initialise PROFIBUS
2: ; *****
3:          L      PBI_OK          M00.00 ; (initialisation ok)
4:          JPC     INI_END          ; ok?-> jump if so
5:          L      1                ; station address 1
6:          =       STATION          BM00.00 ; (station number)
7:          L      16               ; 16 byte input data
8:          =       LEN_IN           BM00.01 ; (length input op.)
9:          L      16               ; 16 byte output data
10:         =       LEN_OUT          BM00.02 ; (length output op.)
11:         L      10               ; interrupt every 10 ms
12:         =       CLOCK            BM00.03 ; (interrupt time [ms])
13:
14: ; Call up initialisation module
15:         JPK     INIT_SL , _____
15:         STATION -| _____|- STATUS,
15:         LEN_IN  -| _____|- ,
15:         LEN_OUT -| _____|- ,
15:         CLOCK   -| _____|-
16: ; Read status
17:         L      STATUS            BM00.04 ; (PROFIBUS init. status)
18:         CMP     0                 ; initialising?
19:         JP=     INI_END
20:         CMP     255                ; init. complete?
21:         JP<>    INI_FAIL
22:         =1      PBI_OK            M00.00 ; (initialisation ok)
23:         JP      INI_END
24: ; Error check
25: INI_FAIL NOP
26:
27: ; Evaluation of error message
28:
29: ; Initialisation complete
30: INI_END NOP
31:

```

6.3. Communication parameters

6.3.1. Sending parameterisation data (Prm_Data)

The passive stations (slaves) are parameterised during the startup phase of the DP system. The master is also entitled to parameterise them during normal operation.

The first 7 bytes (octets 1...7) contain the general bus parameters. They are obligatory for all DP slaves and are almost completely supported by all standard configurators.

The eighth and all following bytes can contain device-specific parameters. These need to be taken special notice of.



continued on next page

6.3.1.1. General bus parameters

Octet 1: Station_status

Refer to EN 50 170, "Sending parameterisation data", for details

Octet 2: WD_Fact_1

Range of values: 1...255

Octet 3: WD_Fact_2

Range of values: 1...255

Octets 2 and 3 define the response monitoring timeout (watchdog time T_{WD}) using the following equation:

$$T_{WD} [s] = 10 \text{ ms} * WD_FACT_1 * WD_FACT_2$$

This allows you to set times between 10 ms and 650 s, independent of the baud rate.

The machine outputs error no. 4 (see chapter "D. Error handling") when there has been a watchdog alert.



Bit 3 (WD_ON) in octet 1 enables (=1) or disables (=0) the watchdog. This setting can be made separately for every device.

Octet 4: Min. Station Delay Responder (min T_{SDR})

Default: 11

Specifies the minimum time the DP slave has to wait before it is allowed to send its response frames back to the DP master. Inputting "0" means that the previous value is maintained.

Octets 5..6: Ident_Number (unsigned16)

Profi Control 680I-SL: 6800 (hex)

DP slave's identification number

Octet 7: Group_Ident

Refer to EN 50 170, "Sending parameterisation data", for details

6.3.1.2. Device-specific bus parameters

Octets 8...15: User_Prm_Data

Apart from the general bus parameters, the DP master can send extra sets of device-specific parameters (octets 8...max. 244) to every DP slave station. There are 8 byte (octets) available for the DP slave Profi Control 680I-SL. The user inputs the relevant values into the PROFIBUS configurator. Profi Control 680I-SL receives the data via operands BI15.05...15.12:

Byte	Octet	Operand	Description
0	8	BI15.05	Available for application specific parameters sent by the DP-Master to Profi Control 680I-SL. The user program in the latter can read and process them.
1	9	BI15.06	
2	10	BI15.07	
3	11	BI15.08	
4	12	BI15.09	
5	13	BI15.10	
6	14	BI15.11	
7	15	BI15.12	



The parameterisation data contained in BI15.05...12 applies if operand BI15.15, "PROFIBUS status", contains status values 4, 7, or 255 (see table on next page).

6.3.1.3. PROFIBUS status in BI15.15

Operand BI15.15 informs the user program running in Profi Control 680I-SL of the PROFIBUS status:

Value	Status	Description
0	Wait Init	The PROFIBUS connection is waiting for initialisation by KUBES module INIT_SL. It has not yet been PROFIBUS enabled.
1	Wait Prm	The PROFIBUS connection is waiting for a valid parameterisation frame from the DP master. It is now enabled as a bus station.
4	Communication Error	A set watchdog control has been set off.
5	Bus Error	Physical PROFIBUS failure in Profi Control 680I-SL.
7	Cfg. Error	Configuration error. The settings made via KUBES module INIT_SL are not the same as the configuration data sent by the DP master.
8	Prm. Error	The parameterisation data sent by the DP master is in excess of 8 byte..
255	Run	Operation ok. Data is being exchanged via PROFIBUS.

6.4. Configuration

In the case of Profi Control 680I-SL, the data to be sent via PROFIBUS is configured by means of KUBES module INIT_SL (see chapter "6.2.4. Initialising PROFIBUS via the user program"). This applies to the amount of input and output data.

At bus system startup, the master verifies whether the "modules" set via the configurator (e.g. VEBES) are the same as the number of initialised modules.

The input and output data (as seen by the master) can be set via the configurator:

Modules

8 byte of input data

64 byte of input data

8 byte of output data

64 byte of output data

Maximum:

240 byte of input data

240 byte of output data

The input and output data to be set via the configurator is seen with the eyes of the master. Seen by the DP slave (Profi Control 680I-SL), its function is exactly reversed:

<i>Operands</i>	<i>seen by the master</i>	<i>seen by the DP slave</i>
BI...	output data	input data
BO...	input data	output data

The set amount of input and output data must be the same as the setting in INIT_SL.

If not, the system outputs error no. 7 (see chapter "D. Error handling").

6.5. Diagnostic information (Diag_Data)

The status of a DP slave is told the master by writing the diagnostic data into the master's specified byte operands (in the case of Kuhnke masters, this operand range is set via the VEBES configurator software). The diagnostic data of Profi Control 680I-SL is contained in six byte (octets 1...6). Byte 7...11 (octets 7...11) contain the device-specific diagnostic information.

6.5.1. Standard diagnostic data

A set bit (=1) indicates that the relevant event has occurred.

Octet 1: Station_Status_1

Bit	set by	Description
0	Master	Diag.Station_Non_Existent (does not apply to Kuhnke masters) DP slave cannot be addressed via the bus
1	Slave	Diag.Station_Not_Ready DP slave not ready for data exchange
2	Slave	Diag.Cfg_Fault The configuration data last received by the slave is not the same as the one found by the DP slave
3	Slave	Diag.Ext_Diag Entry in the device-specific diagnostic data (octet 7...)
4	Slave	Diag.Not_Supported Function requested not supported by the DP slave
5	Master	Diag.Invalid_Slave_Response DP slave sent an invalid response
6	Slave	Diag.Param_Fault The last parameterisation frame was wrong
7	Master	Diag.Master_Lock DP slave already parameterised by another master

Octet 2: Station_Status_2

Bit	set by	Description
0	Slave	Diag.Prm_Req DP slave needs to be reparameterised
1	Slave	Diag.Stat_Diag (statistical diagnosis) DP master requested to get diagnostic data
2	Slave	Diag.Cfg_Fault The configuration data last received by the slave is not the same as the one found by the DP slave
3	Slave	Diag.Ext_Diag Entry in the device-specific diagnostic data (octet 7...)
4	Slave	Diag.Not_Supported Function requested is not supported by the DP slave
5	Master	Diag.Invalid_Slave_Response The DP slave sent an invalid response
6	Slave	Diag.Param_Fault The last parameterisation frame was wrong
7	Master	Diag.Master_Lock DP slave already parameterised by another master

Octet 3: Station_Status_3

Bit	set by	Description
0		reserved
1		reserved
2		reserved
3		reserved
4		reserved
5		reserved
6		reserved
7	Slave	Diag.Ext_Diag_Overflow There is more diagnostic data than specified in the device-specific diagnostic data

Octet 4: Diag.Master_Add

Address of the DP master that parameterised the DP slave.

Octets 5...6: Ident_Number

Supplier ID for DP slave identification

Profi Control 680I-SL: 6800 (hex)

6.5.2. Device-specific diagnostic data

Octets 7...11 (max. 244)

are reserved for device-specific diagnostic data. Only the DP slave has access to this range.

(Profi Control 680I-SL does not support diagnostic data with relation to IDs and channels).

Octet 7: Ext_Diag_Data: Header byte

Bits 0...5: length of block in byte, inc. header byte 2...63

Bits 6...7: permanently set to 0

Octet 8: Ext_Diag_Data

Information is entered into this byte as error numbers. In the case of Kuhnke masters (except for DP masters up to 12 Mbit/s, e.g. 645-12M) the information is written into PROFIBUS-EVENT operand PEa... .

The error numbers correspond to the messages described in chapter "D. Error handling".

Octet 9: Ext_Diag_Data

BO15.15, can be written into by the user program

Octet 10: Ext_Diag_Data

Software version, place before the separator (decimal point)

Octet 10: Ext_Diag_Data

Software version, place after the separator (decimal point)

6.6. Master - slave communication (Profi Control 680I-SL)

6.6.1. Initialisation

After startup, the master starts to initialise the slaves it is connected to:

Sent by	Frame	Received by	
		Device	SAP
Master	Sends new parameters	Slave	61
Slave	Acknowledges by "NR"	Master	-
Master	Requests configuration data	Slave	62
Slave	Sends configuration data	Master	-
Master	Requests diagnostic data	Slave	60
Slave	Sends diagnostic data ¹⁾ - 6 byte: std. diagnostic data only - 11 byte: plus device-spec. data	Master	-



¹⁾ The DP slave responds by sending 6 byte (of standard diagnostic data) if there is no failure condition. If there is one, the DP slave will also send the device-specific diagnostic data (11 byte). In this case, the communication link will not be established.

6.6.2. Data communication

The master starts data communication as soon as initialisation is complete. The data communication process is always the same:

Sent by	Frame	Received by	
		Device	SAP
Master	Sends the output data in a high-priority frame (exception: combimasters can also send low-priority frames)	Slave	-
Slave	Sends the input data: - in a low-priority frame if there is no error - in a high-priority frame if there is an error	Master	-

6.6.3. Error message

Message sent by the slave

Usually, the slave would respond by sending a low-priority frame. However, if it has found an error or failure condition, the slave will send the master a high-priority response frame and continue sending a high-priority frame as long as the master does not react to it but continues data communication as before.

Response by the master

The correct answer of the master to a high-priority frame is to request the diagnostic data.:

Sent by	Frame	Received by	
		Device	SAP
Master	Requests diagnostic data	Slave	60
Slave	Sends diagnostic data (11 byte)	Master	62



see chapter "6.5. Diagnostic data (Diag_Data)"

After successful diagnosis, the master resumes data communication unless the user program contains different instructions.

6.6.3.1. Requesting diagnostic data via the user program

Apart from automatically generated diagnostic data requests, the user program can also request this data. Operand BO15.15 is available for this procedure (see chapter "6.5.2. Device-specific diagnostic data", octet 9: Ext_Diag_Data).

The bits in BO15.15 are assigned as follows

Bit	7	6	5	4	3	2	1	0
		Diagnostic data						
	Request for diagnostic data							

Requesting diagnostic data

The mechanism described below ensures that the user program realises when the diagnostic data has been sent to the DP master:

- The user program writes the diagnostic data (value 0...127) into bits 0...6 and sets bit 7.
- When the diagnostic data has been sent to the DP master, the bus connection automatically resets bit 7.
- Afterwards, only the diagnostic data is contained in bits 0...6.

6.7. Device master data file KUHN6800.GSD

This file contains a record of DP slave master data for Profi Control 680I-SL. You need the file if you wish to network the device by means of a PROFIBUS configurator.

There are various ways of obtaining the file from Kuhnke:

- together with VEBES version 3.10 or higher
- by mail on diskette
- via modem from our mailbox
(telephone +4523/402-310)



If none of the possibilities above applies for any reason, you can also use a text editor to write the file yourself. Just type in the printout on the next couple of pages. Comments (in lines starting with ";") are optional.



see next page for a printout of the file KUHN6800.GSD

Printout of the file KUHN6800.GSD

```
;*****
; (c) 1997      Kuhnke GmbH
;               Luetjenburger Str. 101
;               D-23714 Malente
;               Telephone: ++49 4523/402-0
;               DeviceMasterDataFile for:
; PROFIBUS-DP slave 680I-SL      Part no.: 680.421.08
;*****
#Profibus_DP
GSD_Revision      = 1
;
Vendor_Name       = "Kuhnke"
Model_Name        = "Profi Control 680I-SL"
Revision          = "3.10"
Ident_Number      = 0x6800
;
Protocol_Ident    = 0
Revision_Number   = 1
Station_Type      = 0
;
FMS_supp          = 0
;
Hardware_Release  = "Version 2"
Software_Release  = "V1.00"
;
9.6_supp          = 1
19.2_supp         = 1
500_supp          = 1
;
MaxTsdr_9.6       = 60
MaxTsdr_19.2      = 60
MaxTsdr_500       = 100
;
Redundancy        = 0
;
Repeater_Ctrl_Sig = 0
24V_Pins          = 0
```

DP slave Profi Control 680I-SL in PROFIBUS networks

```
;  
Implementation_Type      = "TMG itec"  
Bitmap_Device           = "NOR6800"  
Bitmap_Diag             = "DIA6800"  
Bitmap_SF               = "SF_6800"  
Freeze_Mode_supp       = 0  
Sync_Mode_supp         = 0  
Auto_Baud_supp         = 1  
Set_Slave_Add_supp     = 0  
Min_Slave_Interval     = 20  
;  
Modular_Station         = 1  
Max_Modules             = 60  
Max_Input_Len           = 240  
Max_Output_Len          = 240  
Max_Data_Len            = 480  
Module_Offset           = 0  
Max_User_Prm_Data_Len  = 8  
;  
Fail_Safe               = 0  
Slave_Family            = 10  
Max_Diag_Data_Len      = 11  
;  
Unit_Diag_Area          = 0-7  
Value(0)                = "No error"  
;  
Value(2)                = "Supply: undervoltage"  
Value(3)                = "Internal watchdog"  
Value(4)                = "Communication error"  
;  
Value(5)                = "Bus error"  
Value(6)                = "Wrong baud rate"  
Value(7)                = "Configuration error"  
;  
Value(8)                = "Wrong parameter"  
Value(11)               = "Internal communication error"  
Unit_Diag_Area_End  
;  
Unit_Diag_Area          = 8-15  
Value(0)                = "User diagnosis"
```

```

Unit_Diag_Area_End
;
Unit_Diag_Area      = 16-23
Value(0)            = "Software version I"
Unit_Diag_Area_End
;
Unit_Diag_Area      = 24-31
Value(0)            = "Software version II"
Unit_Diag_Area_End
Ext_User_Prm_Data_Const(0) = 0x00,0x00,0x00,0x00,0x00,\
                                0x00,0x00,0x00
Module = „ 8 byte of input data „ 0x97
1
EndModule
Module = „ 64 byte of input data „
0x97,0x97,0x97,0x97,0x97,0x97,\
0x97,0x97
2
EndModule
Module = „ 8 byte of output data „ 0xA7
3
EndModule
Module = „ 64 byte of output data „
0xA7,0xA7,0xA7,0xA7,0xA7,0xA7,\
0xA7,0xA7
4
EndModule

```


A. Summary of data

A.1. Technical data

Processors C166 for the user program
8051 for PROFIBUS

Basic device with 4 slots

No. of module slots 4
Max. number of digital I/Os 64 (*up to vers. 4.01: 32*)
Dimensions (B*T*H) 190*135*108 mm

Basic device with 8 slots

No. of module slots 8
Max. number of digital I/Os 128 (*up to vers. 4.01: 64*)
Dimensions (B*T*H) 332*135*108 mm

Program memory memory modules

RAM memory modules

- Size 32K, 128K or 256K * 16
- Data secured by accu, 80 mAh, min. buffering time 6 weeks (at 0...40 °C), max. recharge time 72 h

EPROM memory modules

- Size 32K or 128K x 16
- Access time max. 100 ns

EPROM-RAM memory modules

- EPROM size 32K or 128K x 16
- EPROM access time max. 100 ns
- RAM size 32K or 128K x 16
- RAM data secured by accu, 80 mAh, min. buffering time 6 weeks (at 0...40 °C), max. recharge time 72 h

Flash-RAM memory module

- Flash-EPROM size 128K x 16
- RAM size 128K x 16
- RAM data secured by accu, 80 mAh, min. buffering time 6 weeks (at 0...40 °C), max. recharge time 72 h

Appendix

Programming

Programming device	IBM-PC (or compatible)
Operating systems	Windows 3.1 or Windows 95
Programming software	KUBES (user program) VEBES (PROFIBUS network)
Type of programming	IL, C
Documentation	IL, FUP, LD, SymT, CRL

Line interfacing

Power supply and signals	clamp-screw terminals
Interface(s)	D-Sub connector, 9-pin

Power supply 24 V DC +25% -20%

Power consumption of basic device ... max. 160 mA (without modules)

Profi Control 680I networking capacity:

Function	combimaster
Supported protocols	- PROFIBUS-FMS - PROFIBUS-DP - SINEC L2-DP
Own station address	64 (0...63)
Communication station addresses ..	100 (0...99)
Baud rate / bus length	- 500 Kbit/s - 19.2 Kbit/s - 9.6 Kbit/s

Profi Control 680I-SL networking capacity:

Function	DP slave
Supported protocols	PROFIBUS-DP
Own station address	126 (0...125)
Communication station addresses ..	100 (0...125)
Baud rate	- 500 Kbit/s - 19.2 Kbit/s - 9.6 Kbit/s

Admissible ambient conditions

Storage temperature	-25...+70 °C
Ambient temperature during operation	0...55 °C
Relative humidity	50...95 %

A.2. Part numbers

Combimaster Profi Control 680I

with 4 slots	680.423.04
with 8 slots	680.423.08

DP slave Profi Control 680I-SL

with 4 slots	680.421.04
with 8 slots	680.421.08

User program memory

RAM memory module, 32K x 16	680.426.01
RAM memory module, 128K x 16	680.426.02
EPROM memory module, 32K x 16	680.427.01
EPROM memory module, 128K x 16	680.427.02
EPROM-RAM memory module, 2 x 32K x 16	680.428.01
EPROM-RAM memory module, 2 x 128K x 16	680.428.02
Flash-RAM memory module, 2 x 128K x 16	680.428.03

Clamp-screw terminals

2-pin, type Phönix MC1.5/2-ST-3.81 (3 pcs)	680.180.01
8-pin, type Phönix MC1.5/8-ST-3.81 (12 pcs)	680.180.02
ditto, preassembled with blue leads (3 m)	680.180.08

PROFIBUS connector (D-Sub, 9-pin)

Connector kit (1xmale, 1xfemale) for the bus cable	680.180.03
Bus branches (T-line) with screw-type connectors	
- horizontal cable outlet	690.180.05
- vertical cable outlet	690.180.06

PROFIBUS bus terminators (D-Sub, 9-pin)

Passive, line B (connector kit 1xmale, 1xfemale)	680.180.07
Passive, line A (connector kit 1xmale, 1xfemale)	680.180.13
Passive bus terminator for cable type A with screw-type connectors	
- horizontal cable outlet	690.180.07
- vertical cable outlet	690.180.08
Active, line B, supply 230 V AC	680.180.10
Active, line B, supply 24 V DC	680.180.15
Active, line A, supply 230 V AC	680.180.12
Active, line A, supply 24 V DC	680.180.14

Appendix

Other accessories

Simulator box for digital inputs (4 x 8-pin	680.155.50
Quick mounts for carrier rail installation (2 pcs)	680.180.05

Digital input modules

24 V DC, 8 inputs	680.451.01
24 V DC, 8 inputs, 1 ms	680.451.04
24 V DC, 8 inputs, with realtime clock	680.451.02
24 V DC, 16 inputs	680.451.03
24 V DC, 16 inputs, interrupts supported	680.451.06
24 V DC, 16 inputs, 1 ms	680.451.07

Digital output modules

24 V DC, 0.5 A, 8 outputs	680.452.01
4 pneumatic outputs 3/2-way	680.453.01

Digital input/output modules

24 V DC, 8 inputs, 8 outputs	680.450.01
------------------------------------	------------

Analogue input modules

0...10 V, 10 bit, 4 channels	680.441.01
0(4)...20 mA, 10 bit, 4 channels	680.441.02
PT100, 0...300 °C, 10 bit, 4 channels	680.441.04
Thermo-couple Ni-Cr-Ni (type K)	
0...1200 °C, 10 bit, 4 channels	680.441.07
Potentiometer, 10 bit, 4 channels	680.441.05

Analogue output modules

0...10 V, 8 bit, 4 channels	680.442.01
0(4)...20 mA, 8 bit, 4 channels	680.442.02

Analogue input/output modules

2 I 0...10 V, 2 O 0...± 10 V	680.441.03
2 I 0...20 mA, 2 O 0...±10 V	680.441.06
2 I 0...10 V, 2 O 0...20 mA	680.441.08
2 I 0...20 mA, 2 O 0...20 mA	680.441.09

Counter modules

1 multi-function counter, 24 bit	680.454.01
2 multi-function counters, 24 bit, 24 V	680.454.02

2 multi-function counters, 24 bit, RS 422	680.454.08
2 event counters, 16 bit, 24 V	680.454.03
SSI module, 2 encoder connectors, 24 bit	680.454.04

Communication modules

V.24 module	680.440.01
TTY module	680.440.02
RS 485 module (up to max. 19600 Baud)	680.440.03
PROFIBUS-DP slave module	680.440.05
PROFIBUS-DP master module	680.440.06

Stepper motor modules

without processor, 1 channel	680.444.01
without processor, 2 channels	680.444.02
with processor, 2 channels	680.444.03

Positioning modules

with encode input and analogue output, 1 channel	680.454.06
Servo counter module, 2 channels	680.454.05

User software

Communication programs SE_680I, KUSI-680, RS485	680.505.01
Regulator program (C task)	680.506.01
Positioning program 680 (C task)	680.506.02

Instruction manuals

Modules	E 326 GB
KUBES programming software	E 327 GB
VEBES network configurator	E 315 GB
Programming manual	E 417 GB
KUBES modules	E 386 Gb
PROFIBUS manual	E 365 GB

Appendix

B. PROFIBUS hardware installation

B.1. PROFIBUS cable

The PROFIBUS standard (EN 50 170, PROFIBUS) specifies two cables for the PROFIBUS-FMS and PROFIBUS-DP protocols.

Both cable types are discussed below. Due to its benefits, we recommend that you use cable type A.

B.1.1. Cable type A

Cable type A is particularly well suited for high transfer speeds (> 500 kbaud).

Technical data

- Characteristic impedance: 135...165 Ω
(measuring frequency 3 to 20 MHz)
- Cable capacity: < 30 pF per metre
- Core diameter: > 0.34 mm², compl. with AWG 22
- Cable type: twisted pair,
1 x 2, 2 x 2 or 1 x 4-core
- Loop resistance: < 110 Ω per km
- Signal attenuation: max. 9 dB across the entire length
of the cable section
- Shielding: copper gauze sheath or
gauze sheath and foil sheath

B.1.1.1. Transfer rate and Cable length

Baud rate [kbit/s]	9.6	19.2	93.75	187.5	500	1500	3000	6000	12000
Cable length [m]	1200	1200	1200	1000	400	200	100	100	100

Both Profi Control 680I and 680I-SL support baud rates 9.6/19.2 and 500.



See next page for information about T-lines

B.1.1.2. T-lines

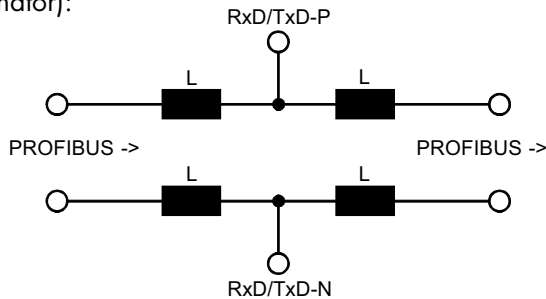
Up to 500 kbit/s
max. 0.3 m per T-line

1500 kbit/s

The overall T-line capacity is to be smaller than 0.2 nF. This results in a total length of 6.6 m for all T-lines together.

3000 ... 12000 Mbit/s

Although T-lines are allowed they should be avoided. The overall T-line capacity is to be smaller than 0.05 nF. This results in a total length of 1.6 m for all T-lines together. Integrate series inductances into the connector to reduce inter-cable reflexions. One practical solution would be to install the inductances in the connector (bus branch or bus terminator):



Use the following series inductance for the connecting capacity of a typical bus station (connector, lead length to the RS-485 driver, RS-485 driver, components, etc.):

$L = 100 \text{ nH}$



The capacity of the connected station is taken into account for calculating the inductance. Pulling off this kind of connector may therefore cause maladjustments that might disturb bus operation.

B.1.2. Cable type B

Cable type B must be used only for baud rates up to 500 kbit/s and for small distance requirements.

Technical data

- Characteristic impedance: 100...130 Ohm
(measuring frequency > 100 kHz)
- Cable capacity: typ. < 60 pF per metre
- Core diameter: > 0.22 mm², compl. with AWG 24
- Cable type: twisted pair
1 x 2, 2 x 2 or 1 x 4-core
- Signal attenuation: max. 9 dB across the entire length
of the cable section
- Shielding: copper gauze sheath or
gauze sheath and foil sheath

Transfer rate and cable length

Baud rate [kbit/s]	9.6	19.2	93.75	187.5	500
Cable length [m]	1200	1200	1200	600	200

T-lines must be max. 0.3 m long.

B.1.3. Shielding

EN 50 170 leaves it up to the operator whether he wishes to use shielded or unshielded cables. Unshielded cables are allowed for non-interference environments.

For the following reasons we recommend that you always use shielded cables:

- a) Non-interference rooms may exist inside of shielding switching cabinets if at all. However, as soon as there are relays inside the cabinet, a non-interference condition can no longer be guaranteed.
- b) The use of unshielded cables would require the taking of additional protective measures against overvoltages at the bus signal inputs.

This recommendation also applies to leads that may connect any external power supplies with the PROFIBUS devices. Double-shielded leads are the best choice for very non-EMC environments. To ensure optimal protection, both the outer (gauze) sheath and the inner (foil) sheath are to be connected to protective earth using an earthing clip at both ends of the cable. If you are using a shielded bus cable we recommend that you connect the shield to protective earth using low inductances at both ends of the cable. This ensures best EMC results possible.



An exception are separate potentials, e.g. in refineries. In these cases it is usually enough to connect one cable end to earth.

...in conjunction with D-Sub connectors

Preferably, the connection between cable shield and protective earth is made via a metal device casing and the screw-on cap of the D-Sub connectors.

...in conjunction with other plug-type connectors

Connect the cable shield to functional or protective earth directly at the device. In many cases, this connector is located directly at the device casing.

B.2. Connecting the stations



When connecting the stations make sure not to twist the data lines.

Tip: Make a basic decision as to what colour core you want to use for the data lines.

We recommend

red or white (lighter colour) for RXD/TXD-P
green or brown (darker colour) for RXD/TXD-N

Connect a bus terminator to terminating stations.

B.2.1. Connectors

There is a variety of connectors available. A difference is made between bus branches that mark connecting points between two stations and bus terminators that connect with terminating stations.

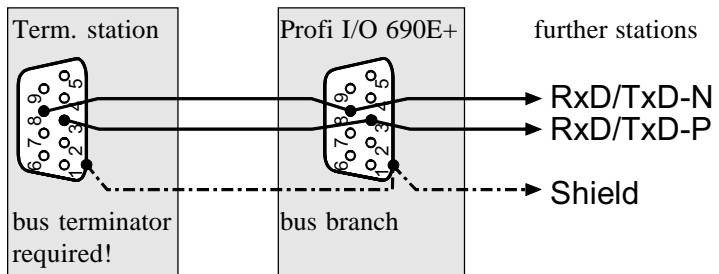
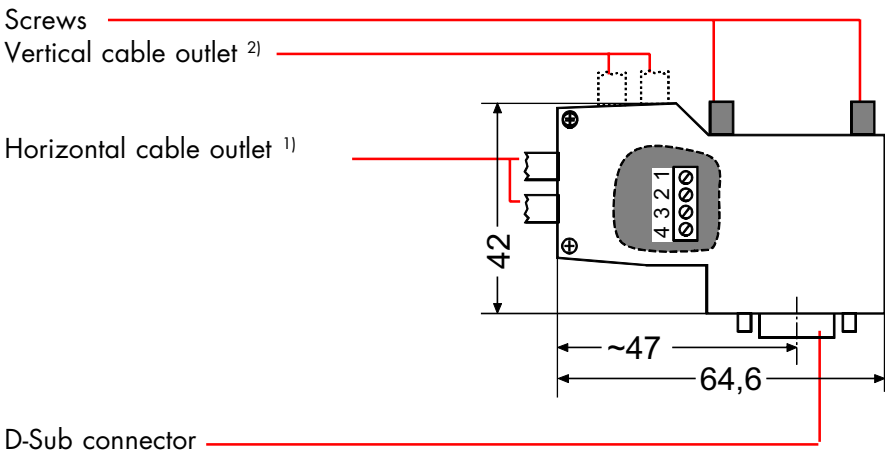


Fig.: Network consisting of at least three stations.

B.2.1.1. Bus branches

A bus branch connects one bus stations with 2 other stations. It is therefore also referred to as T-stub as it has two cable connectors and one D-Sub connector for the bus station. Series inductances for A type cables are integrated.

To make attaching the bus leads easier, the connector provides 4 clamp-screw terminals, two for the incoming lead and two for the outgoing lead (see figure).



There are two bus branch variants:

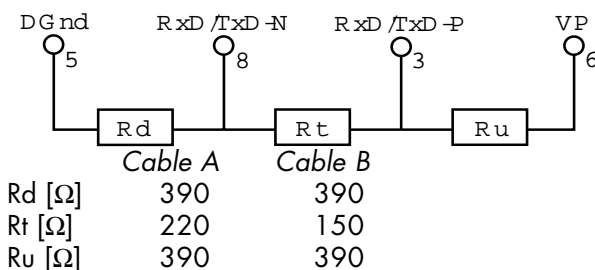
Name	Part number
¹⁾ with horizontal cable outlet	690.180.05
²⁾ with vertical cable outlet	690.180.06

There are also connectors which are delivered as kits. The lead has to be soldered to the connector.

Name	Part number
Connector kit (1xmale, 1xfemale)	680.180.03

B.2.1.2. Bus terminator

Both ends of the PROFIBUS cable are to be connected to a bus terminator. This ensures a defined idle potential on the line at time when there is no communication. A bus terminator consists of a resistor network which is connected as follows:



Passive bus terminator for type A cables

We supply a connector with built-in bus terminator and series inductances for type A cables. Due to the fact that bus terminators are connected to terminating bus stations, it only has one lead outlet and 2 terminals:

Terminal 1: RxD/TxD-N

Terminal 2: RxD/TxD-P

Otherwise the passive bus terminator is designed just like the T-stub describe in chapter "B.2.1.1. Bus branch".

Part names and numbers

Name	Part number
Passive bus terminator for type A cables	
–with horizontal lead outlet	690.180.07
–with vertical lead outlet	690.180.08

We also supply bus termination connectors which we deliver as kits. The lead has to be soldered to the connector.

Name	Part number
Passive bus terminator	
- Line B (kit: 1xmale, 1xfemale)	680.180.07
- Line A (kit: 1xmale, 1xfemale)	680.180.13

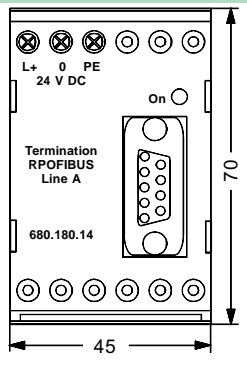
Active bus terminator



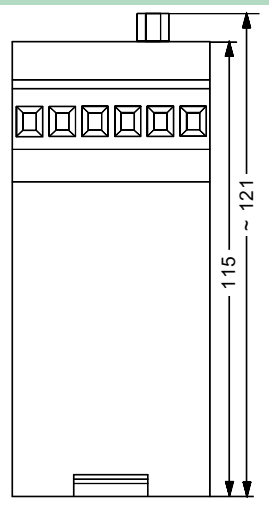
Bus terminators need to be supplied with power at all times. This is not always guaranteed in the case of the (passive) bus terminators described earlier as these terminators are supplied via the controller. When you switch the controller off, the bus terminator function is also disabled.

Thus, active bus terminators are to be used in systems that allow terminating stations to be switched off during operation. Active terminators have their own power supply which ensures that its function is available at all times.

Top view:



Side view:



Wiring diagram:

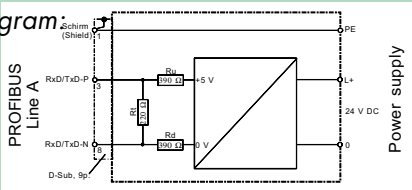


Fig: Active bus terminator, 24 V DC power supply

Active bus terminators have a snap-on device for carrier rail installation directly next to the terminating station. Connect a T-stub to the terminating station and, via a short lead, to the bus terminator.

Part names and numbers

<i>Name</i>	<i>Part number</i>
Active bus terminator, 230 V AC power supply	
– for type A cables	680.180.12
– for type B cables	680.180.10
Active bus terminator, 24 V DC power supply	
– for type A cables	680.180.14
– for type B cables	680.180.15

C. References to literature

Instruction manual E 326 GB, Modules

Kuhnke GmbH, Malente

Instruction manual E 334 GB, SE_680I

Communication program for the V.24 module

Kuhnke GmbH, Malente

Programming manual E 417GB

Programming of the KUAX systems

Kuhnke GmbH, Malente

Instruction manual E 315 GB, VEBES

PROFIBUS network configurator

Kuhnke GmbH, Malente

Instruction manual E 327 GB, KUBES

Kuhnke user software

User interface for programming, testing and documenting
user programs for Kuhnke controllers

Kuhnke GmbH, Malente

Instruction manual E 365 GB, PROFIBUS

Kuhnke GmbH, Malente

EN 50 170, PROFIBUS

Transmission technique, bus access and transfer protocol,
service interface to the application layer, management,
communication model, application services, protocol, syn-
tax, coding, interface to the data link layer, management

PROFIBUS, The fieldbus for industrial automation

Klaus Bender (Ed.)

Carl Hanser Verlag/Prentice Hall, München Wien

ISBN 13-012691-8 (hbk)

Appendix

D. Error handling

The controller monitors itself. Any errors occurring are reported and cause the controller to react according to their seriousness.

The errors are numbered through from 1 to max. 255.

They can be indicated in several ways:

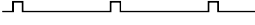



Errors overview

No.	Error Description	Indication				
		LED failure flashes n times	Error byte ERR.00.00	680I PB-Event	680I-SL Ext_Diag_Data octet 8	Interrupt module [no.]
1	Short circuit at output	1	1	3	1	18
2	Undervoltage	2	2	5	2	17
3	Watchdog	3	no	2	no	
4	No communication	no	4	no	4	
5	Bus error / wrong station address	no	5	no	5	
6	Illegal baud rate	no	6	no	6	
7	Wrong module configuration	7	7	7	7	
8	Checksum error in user program	8	no	1	no	
9	Hierarchy error	9	no	4	no	
10	No memory module	3	no	6	no	
11	Overtemperature	no	no	no	no	
12	Short circuit removed	no	0	no	0	
13	Voltage ok	no	0	no	0	

Legend for the errors overview

-LED "failure"

The red light emitting diode is located on the left side of the device. It flashes in a rhythm that indicates the error number:

No.	Flash rhythm
1	
2	
3	
4	

etc.

The counting impulses follow each other fast (250/250 ms). Then there is a break (1 s) whereupon the counting impulses are repeated.

-Error byte "ERR00.00"

The error number is written into an error byte (ERR00.00). It can be evaluated by the user program:

Example:

```
L          ERR00.00
C8T1      000.00 ;binary output to 8 outputs
```

-PROFIBUS (Event) - Profi Control 680I only

The error is transmitted onto the PROFIBUS as event notification together with its error number (cf. instruction manual E313, "PROFIBUS module..."). The messages are sorted internally according to a priority list. If there are several messages at the same time, then the message with the highest priority (lowest priority number) is transferred first.

-Ext_Diag_Data, octet 8 - Profi Control 680I-SL only

The error is transferred as diagnostic data by telling PROFIBUS the error number (see chapter "6.5. Diagnostic data")

-Interrupt module [no.]

The error causes an interrupt (IRQ). This causes the monitor to immediately calling the assigned interrupt module.



In the following paragraphs, individual types of errors are described and suggestions for handling them are made.

D.1. Short circuit at output (error 1)

Cause

- Short circuit
- Overload

Indication

- LED "failure" flashes
- KUBES displays the error in plain text
- Error byte "ERR00.00" is set to 1
- Event notification or diagnostic data to PROFIBUS

Reaction

- the relevant output is switched off thermally
- interrupt module (interrupt) no. 18 is activated. Use this module to define the desired reactions of the controller:
Example: O_OFF ;switch all outputs off
 =1 Mxx.xx ;set marker
the program run is continued, only the outputs are switched off externally (their internal status remains unaltered, however; the LEDs are switched off too, though)

Corrective action

- remove short circuit and then either
- switch the outputs back on again via the program (do not program this in the interrupt module as this is only activated once upon occurrence of the error).

```
Example: L      Mxx.xx    ;outputs switched off?
          JPCN   RETURN   ; jump if no
          L      Iyy.yy    ;input "SC removed"
          JPCN   RETURN   ; jump if no
          O_ON   Mxx.xx    ;switch outputs back on
          =0     Mxx.xx    ;reset marker
          RETURN ....     ;normal program
```

- all internally set outputs are switched back on, program operation is continued
- LED "failure" extinguishes
- error byte "ERR00.00" is reset
- or
- restart the controller:
via hardware: switch the supply off and back on again
via software: KUBES commands RESET and RUN

Error handling

D.2. Voltage monitoring (supply, error 2)

Supply voltage: 24 V DC \pm 20%

A built-in voltage monitoring element reacts to exceeding or falling below certain limits.

D.2.1. Overvoltage

Cause

– Supply voltage approx. > 35 V

Reaction

– components are destroyed

D.2.2. Undervoltage

1st Step

Cause

- Supply voltage approx. $< 19\text{ V}$

Reaction

- interrupt module no. 17 is activated
- program operation is not yet interrupted



Buffered operands (markers, timers and counters) can be reset unvoluntarily if the user program is processed further at this stage, the cause being that inputs might find 0 signals due to the undervoltage.

Indication

- LED "failure" flashes
- Error byte "ERR00.00" is set to 2

2nd Step

Cause 1st alternative

Supply voltage rises back to 24 V DC

Reaction

- LED "failure" extinguishes
- Error byte "ERR00.00" is reset
- The program is continued without interruption

Cause 2nd alternative

Supply voltage drops further to approx. $< 17.5\text{ V}$

Reaction

5 V system voltage is interrupted

- => - STOP: program run is stopped
- RESET: local outputs, error byte (ERR00.00) as well as unbuffered markers, timers and counters are reset
- all LEDs off



continued overleaf

Error handling

Corrective action

– as a precaution, in the user program, to back up buffered operands:

Processing of the user program should be interrupted until step 2 of the cause is reached or until a realistic waiting time has been exceeded.

program for interrupt module no. 17, example:

```
WAIT      5          ;wait 5 * 10 ms = 50 ms
L          ERR00.00  ;read error byte
CMP        2          ;still undervoltage?
JP<>>      RETURN    ; jump if not
RESET                                ; else RESET and stop prog.
RETURN     NOP        ;continue normal program
```



*The WAIT command causes a program loop whose length is specified as $n * 10$ ms. Should this time be longer than approx. 70 ms, a watchdog error is found which then switches the controller off. Waiting times should therefore be no longer than 50 ms.*

D.3. Watchdog (program runtime exceeded, error 3)

Cause

- runtime of a module > 50...70 ms
- or
- runtime of the overall program > 2 s

Indication

- LED "failure" flashes
- KUBES displays the error in plain text
- Event notification or diagnostic data to PROFIBUS

Reaction

- STOP: stops program operation
- RESET: resets local outputs and unbuffered markers, timers and counters
- external outputs (PROFIBUS) off

Corrective action

- change the program structure to achieve shorter runtimes
- restart controller:
 - via hardware: switch the supply off and back on again
 - via software: KUBES commands RESET and RUN

D.4. PROFIBUS error (errors 4, 5)

These error messages occur in network systems only.

Cause

There is a malfunction in the communication with the other stations, caused by:

- PROFIBUS cable interrupted,
- failure in other station,
- wrong station address set,
- baud rate set to invalid value ...

Reaction

- the values of the external inputs are no longer valid

Indication

- LED "failure" flashes
- Error byte "ERRO0.00" is set to 4 or 5
- Status information in operand B15.15 (see chapter "6.3.1.3. PROFIBUS status in B15.15")

Corrective action

- find and remove the cause of the error

D.5. Checksum in the user program (error 8)

A specific algorithm is applied to generate a checksum (CS) for the entire user program memory during program creation.

Cause

When you switch on the controller, the monitor re-calculates the checksum and compares it to the stored value. If the two results are not identical, the monitor outputs an error message.

Reaction

- Controller does not start

Indication

- LED "failure" flashes
- LED "Stop" is on
- Error byte "ERR00.00" is set to 8
- Diagnostic data (680I-SL) to PROFIBUS

Corrective action

- find and remove the cause of the error
- by using KUBES, transfer the project again into the control

D.6. Hierarchy error (error 9)

Program calls and other module calls must not exceed certain hierarchy limits.



During programming, the controller indicates a preliminary hierarchy error every time you transmit a program. At this stage, this is only a warning that there could be an error. Only at start-up does the controller verify whether there is really a hierarchy error. The message disappears if there is not.

Cause

When you start the controller either by switching it on or via KUBES' Start command, the monitor program verifies whether there is a hierarchy error (a module calls the module by which itself was called or nesting exceeds 5 levels)

Reaction

- the controller does not start

Indication

- LED "failure" flashes
- LED "Stop" is on
- Error byte "ERR00.00" is set to 9
- Diagnostic data (680I-SL) to PROFIBUS

Corrective action

- find and remove the cause of the error
- use KUBES to transmit the project again to the controller

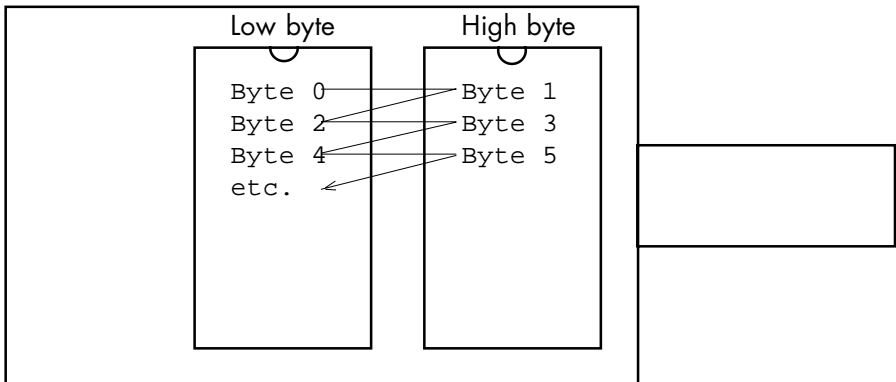
E. EPROMs and Flash-EPROMs

E.1 Programming EPROMs

The user program is written in the KUBES environment and tested and put into operation by means of RAM memory modules plugged into the controller. To secure your valuable data, the program is then stored in EPROMs which plug into EPROM memory modules.

Program distribution amongst the EPROMs

An EPROM (or EPROM-RAM) memory module has 2 sockets for EPROMs. The program is divided up such that the low byte of the program is stored in one EPROM and the high byte in the other:



This distribution must be taken into account when programming the EPROMs

The ways of achieving this distribution change with the different KUBES releases (see next page):

Programming EPROMs

E.1.1. Using KUBES 4.00 or higher

These KUBES releases create two files:

- Low byte
- High byte

E.1.1.1. Creating files with KUBES

Prerequisites

- There is a RAM module containing the user program plugged into the controller
- On the PC, KUBES is running with an opened project
- Online mode
- Reset mode
- The program has been adjusted
- The program has been in RUN mode (main status line displays "A" (adjusted), not "AH"!)

Procedure

- Open the "EPROM" menu
- Select item "Create HEX/BIN file"
- Choose the EPROM type from the dialog (size 32k, 128k, or 256k). The choice of EPROM depends on the size of the program or the available memory.
- Choose the file type from the dialog:
 - "Hex file" if your EPROM programming device can read files in the Intel-Hex format
 - "Binary file" if your EPROM programming device can read binary files
- File name and directory
 - KUBES uses the project name and directory. You can change both.
- "Create file"
 - KUBES creates two files, one containing the high byte, the other the low byte. The extensions are assigned automatically by KUBES:

Extensions of
EPROM file names:

Size	Prog. part	Hex file		Binary file	
32 Kx16	Low byte	<name>.	HLA	<name>.	BLA
	High byte		HHA		BHA
128 Kx16	Low byte	<name>.	HLC	<name>.	BLC
	High byte		HHC		BHC

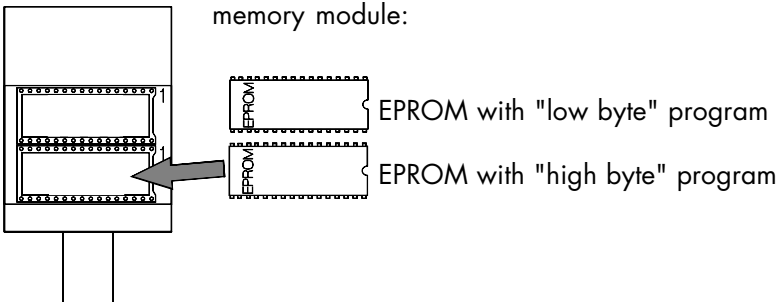
E.1.1.2. Programming EPROMs

Prerequisites

- The EPROM programming device must be capable of programming the required EPROM types (see chapter "3.6.3. EPROM memory modules").

Procedure

- Connect the EPROM programming device to the PC
- Use the software installed in the EPROM programming device to transfer the "low byte" Hex or Binary file you just created to the programming device
- Plug the first EPROM into the EPROM programming device, burn in the program, mark with "low byte" and remove
- Use the software installed in the EPROM programming device to transfer the "high byte" Hex or Binary file you just created to the programming device
- Plug the second EPROM into the EPROM programming device, burn in the program, mark with "low byte" and remove
- Plug the EPROMs into the EPROM (or EPROM-RAM) memory module:



E.1.2. Using KUBES 4.00ß

This older KUBES release creates a separate file for each memory bank that contains program code. High byte and low byte are not separated. Thus, the program distribution is to be done by the EPROM programming device. The process is as follows:

E.1.2.1. Creating files with KUBES

Prerequisites

- There is a RAM module with the user program plugged into the control
- The program has been adjusted
- The program has been in RUN mode (main status line displays "A" (adjusted) and not "AH"!)
- On the PC, KUBES is running with an opened project
- Online mode
- Reset mode

Procedure

- Open the "EPROM" menu
- Select command "Create HEX/BIN file"
- Select the file type from the dialog:
 - "Hex file" if your EPROM programming device can read files in the Intel-Hex format
 - "Binary file" if your EPROM programming device can read binary files
- Select memory banks
 - A separate file is created for every bank you select.
 - KUBES preselects all banks that contain program code.
- File name
 - KUBES uses the project name, adds ".Hxx" or ".Bxx" (H standing for Hex files, B for binary files, and xx representing the bank no.) and the path to the files' storage location.

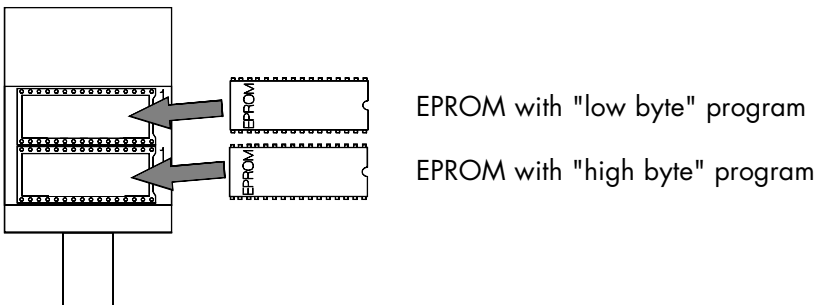
E.1.2.2. Programming EPROMs

Prerequisites

- The EPROM programming device must be capable of programming the required EPROM types (see "3.6.3. EPROM memory modules").
- It must be able to split the open file into an EPROM for the low byte (even) and one for the high byte (odd).

Procedure

- Connect the EPROM programming device to the PC
- Use the software installed in the EPROM programming device to transfer the Hex or Binary file(s) you just created to the programming device (if several files have been created make sure to transfer the separate banks in ascending order)
- Plug the first EPROM into the programming device
- Burn in the Low Byte (even) program, mark EPROM with "low byte" and unplug
- Plug the second EPROM into the programming device
- Burn in the High Byte (odd) program, mark EPROM with "high byte" and unplug
- Plug the EPROMs into the EPROM (or EPROM-RAM) memory module:



E.2. Programming Flash-EPROMs

Use KUBES to write the user program. Then transfer it to the Flash-EPROM via the controller's programming interface, test and start it.



KUBES first writes the module table into a specially reserved RAM area. The module table therefore depends on the accu's charging state.

Once you have completed your program you will therefore have to transfer the module table to the Flash-EPROM separately to make sure that the program no longer depends on the accu's charging state.

E.2.1. Using KUBES to copy the program to Flash-EPROM

Make sure that the program has been adjusted (main status bar displays "A") prior to copying the it to the Flash-EPROM. Then proceed as follows:

- Menu "EPROM"
- Command "Copy to Flash"

The following dialog pops up:



- Click on OK
- Wait until the dialog has disappeared

User program and module table are now safely stored in the Flash-EPROM.

E.2.2. Using "FLASH_UP" to program Flash-EPROMs

FLASH_UP allows you to transfer the user program to a controller equipped with a Flash-EPROM without requiring KUBES to be installed.

Profi Control 680I and Profi Control 680I-SL support this feature as from monitor version 6.01.

What do you need?

Before you transfer the user program be sure to have: a PC with a serial port, a programming cable (part no. 657.151.03), and a diskette with FLASH_UP and the new user program (as a binary file) on it.

A simple MS-DOS-PC is enough, you do not need to run MS-Windows.

E.2.2.1. Creating the FLASH_UP diskette

Creating EPROM files

First you use KUBES to program a controller, then the PLC program is stored as EPROM files:

- Start KUBES, open a project
- Go Online with the PLC, RESET PLC
- Transmit project to PLC
- Start PLC (to allow checksum calculation)
- RESET PLC
- Copy PLC program to Flash-EPROM
- Create EPROM files (type: binary file)

You will find the binary files in project directory
 \KUBES\<project name>:
 <project name>.BLE and <project name>.BHE.

Preparing the EPROM files

The EPROM files you just created need to be prepared for use with FLASH_UP. This is achieved by means of the MAKEBIN and MERGEBIN utility programs:

MERGEBIN

merges binary files <project name>.BLE and <project name>.BHE into <project name>.BIN.

Data preparation procedure:

- Start MERGEBIN:

 - Command: MERGEBIN<Enter>

- Specify drive containing the KUBES project

- Specify name of the network (for network project, else type <ENTER>

- Type in project name

You can create FLASH_UP diskettes automatically if

FLASH_UP is stored in the sub-directory as MERGEBIN.

Otherwise you will have to copy binary files <project name>.BHE and <project name>.BLE (located in the KUBES project directory) as well as FLASH_UP.EXE to a diskette.

E.2.2.2. Transferring the program to the controller



Please take note of the following warning:

If, for any reason, an error occurs while FLASH_UP is running, or if you quit the program by answering No to any of the security prompts, the contents of the PLC's user memory may be undefined. In that case, the PLC will not restart.

Normally, you should be able to remedy this situation by simply starting FLASH_UP again. However, if that proves to be impossible, you will have to run the KUBES error analysis routine.

The FLASH_UP diskette contains FLASH_UP.EXE and a binary file, <project name>.BIN.

Program transfer procedure:

– Start FLASH_UP from the diskette:

Commands: A:<Enter>

FLASH_UP<Enter>

FLASH_UP will execute the following operations. It:

– searches for a PLC connected to one of the PC's serial ports

– searches the diskette for a binary file;

FLASH_UP stops and outputs an error message if it is unable to find a PLC or a binary file

– RESETs the PLC

– deletes the old PLC program

– transfers the new PLC program (the binary file) to the PLC

– updates the PLC's Flash-EPROM

– restart the PLC

The PLC RESET and START operations, which might become dangerous if there is a machine connected to the controller, are to be explicitly confirmed by responding to the extra security prompts.

Index

A

- accumulator
 - of the CPU 4-21
- address mnemonic 4-22
- addressing 4-22
 - types of 4-24
- analogue inputs and outputs 4-4
- AND commands 4-8
- arithmetics commands 4-12
- assignments and set commands 4-11

B

- BCD commands 4-16
- bus terminator
 - active B-8
 - passive B-7
- byte and flag manipulation 4-15

C

- cable routing and wiring 2-7
- checksum D-9
- coding switch 3-14
 - cover 3-1
- commands
 - overview 4-5
- comparison commands 4-13
- copy commands 4-16
- counters 4-4, 4-17
 - description of operands 4-4

D

- danger 2-2
- data memory
 - in the memory module 3-18

- data module

- commands 4-19
- decentralisation 1-2
- device bus connector 3-29
- device with 8 modules 3-6
- dimensions 3-2

E

- earth 3-10
- electromagnetic compatibility 2-5
- electrostatic discharge 2-5
- EMC 2-5
- emergency stop 2-3
- EPROM memory modules 3-21
- EPROM-Dateien
 - Namen E-3
- EPROM-RAM memory modules 3-23
- EPROMs
 - program distribution E-1
 - programming E-1
 - KUBES > 4.00 E-2
 - KUBES 4.00ß E-4
- error byte "ERR00.00" D-2
- error handling D-1
- error marker 4-4
 - description of operands 4-4
- errors overview D-1
- ESD 2-5
- exclusive OR commands 4-10

F

- failure
 - LED D-2
- failure indication 3-28

Index

Flash-EPROMs

programming E-6

FLASH_UP

programming Flash-EPROMs E-7

H

hardware 3-1

basic device 3-1

hierarchy error D-10

high contact voltage

danger caused by 2-2

I

I/O modules 3-29

information / cross reference 2-2

initialisation module

commands 4-18

inputs 4-3

analogue 4-4

description of operands 4-3, 4-4

digital 4-3

installation 3-2

carrier rail 3-4

to be observed 2-3

wall 3-3

interference emission 2-6

particular sources of

interference 2-8

J

jump commands 4-16

L

light emitting diodes

status and failure indication 3-28

LOAD commands 4-7

logical operations commands 4-6

M

maintenance

to be observed 2-4

markers 4-3

description of operands 4-3

"ERR00.00" 4-4

memory module 3-17

method of operation 4-1

module calls 4-15

modules

quantity 3-29

multi-tasking system 5-9

N

network system 1-2

O

offset addressing 4-22

exceeding the operand range 4-22

operands

overview 4-2

OR commands 4-9

outputs 4-3, 4-4

analogue 4-4

description of operands 4-3, 4-4

digital 4-3

overvoltage D-4

P

part numbers A-3

power supply

device with 8 modules 3-6

input/output modules 3-7

PROFIBUS

bus branch B-6

- bus terminator B-7
 - active B-8
 - passive B-7
- cable B-1
- cable length B-1, B-3
- cable type B-1
- cable type B B-3
- connecting stations B-5
- connector B-5
- D-Sub connector B-6
- horizontal cable outlet B-6
- installation B-1
- T-lines B-2, B-3
- transfer rate B-1, B-3
- vertical cable outlet B-6
- PROFIBUS error D-8
- PROFIBUS-680I
 - data consistency 5-3, 5-9
 - KUBES modules
 - OS_CRIT 5-3, 5-9
 - OS_CRIT 5-3, 5-9
 - validity of external operands 5-11
- PROFIBUS-680I-SL
 - communication parameters 6-8
 - configuration 6-12
 - diagnostic data 6-18
 - device-specific 6-15
 - standard 6-18
 - diagnostic information 6-13
 - standard 6-13
 - mailbox 6-20
 - Kuhnke 6-20
 - master - slave communication
 - data communication 6-17
 - error message 6-18
 - initialisation 6-16
 - Min. TSDR 6-8

- output control settings 6-10
- parameterisation data 6-8
- watchdog 6-9
- program memory 3-18, 4-1
- programmable pulses 4-17
- project planning
 - to be observed 2-3
- pulses 4-17

R

- rack 3-2
- RAM memory modules 3-19, 3-26
- references to literature C-1
- registers 4-21
- reliability 2-1
- resistance to interference 2-5
- rotation commands 4-14

S

- safety 2-1
- separate controller 1-1
- servicing
 - to be observed 2-4
- shift commands 4-14
- short circuit D-3
- signal lines 3-9
- software 4-1
- special commands 4-18
- station address
 - label 3-1
- status and failure indication
 - via LEDs 3-28
- status LED 3-1

T

- target group 2-1
- task change 5-10

Index

technical data A-1

terminal block

power supply 3-1

signals 3-1

timers 4-3, 4-17

description of operands 4-3

U

undervoltage D-5

user program memory 3-17

W

watchdog D-7