

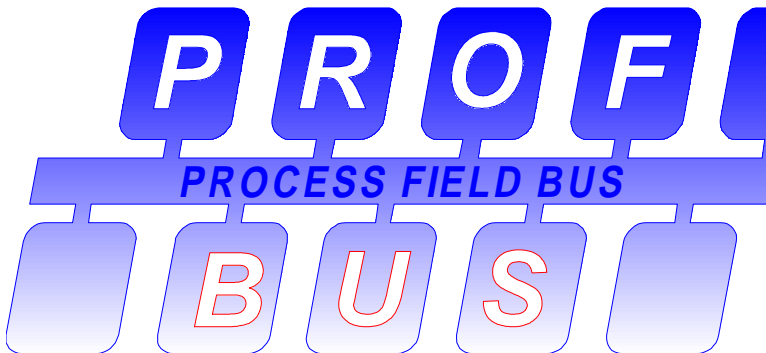
Kuhnke Electronics Instruction Manual

PC Control 645-500

PC Card PROFIBUS-FMS/-DP 500 kbit/s

E 405 GB

19 August 1997 / 67.121



This manual is primarily intended for use by design, project, and development engineers. 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 and modifications.

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

Table of contents

1. Purpose of PC Control 645-500	1-1
1.1. Transition from individual controllers to network systems	1-2
1.3. Remote programming via PROFIBUS	1-4
1.3. Summary of PROFIBUS features	1-5
1.4. PC Control 645-500 as a successor to KUAX 644	1-7
 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. Design	3-1
3.2. Power supply	3-2
3.3. Reset button	3-2
3.4. Coding switch	3-3
3.4.1. Addressing the (ISA-)PC interface	3-3
3.4.1.1. I/O range addressing	3-4
3.4.1.2. Memory range addressing	3-4
3.4.2. Setting the reset trigger	3-5

Table of contents

3.4.3. Setting the I/O wait states	3-5
3.4.4. Setting the interrupt "PC Control 645-500 -> PC"	3-6
3.5. Interfaces	3-7
3.5.1. V.24 interface	3-7
3.5.2. PROFIBUS interface	3-8
3.5.2.1. PROFIBUS cable connector	3-8
3.5.2.2. PROFIBUS bus termination	3-8
3.6. Memory	3-9
3.7. Ni-Cd accumulator	3-9
3.8. Installation	3-10
3.8.1. Installing the card	3-10
3.8.2. Measures for the prevention of address conflicts	3-11
3.8.2.1. I/O range addressing	3-11
3.8.2.2. Memory range addressing	3-12
 4. PC Control 645-500 used as PLC	 4-1
 4.1. Integration into PROFIBUS	 4-2
4.1.1. Creating a network	4-3
4.1.2. Defining PROFIBUS parameters	4-3
4.1.3. Defining the network stations	4-4
4.1.4. Defining the communication paths	4-5
4.1.5. Switching to KUBES	4-5
 4.2. Going on-line with KUBES	 4-6
4.2.1. Online V.24	4-7
4.2.2. Online PC	4-8
4.2.3. Online PROFIBUS	4-10
 4.3. Setting up the memory	 4-11
 4.4. PLC working method	 4-13

4.5. Operands	4-14
4.5.1. Local operands	4-14
4.5.1.1. Short description of local operands	4-15
4.5.1.2. Testing the accumulator functions	4-16
4.5.2. External operands as process image	4-17
4.5.2.1. Example 1: process image of Profi Control 680S (Slave)	4-18
4.5.2.2. Example 2: Process image of a controllers (master)	4-20
4.5.2.3. PROFIBUS messages	4-22
 4.6. Summary of commands	 4-23
4.6.1. Logical operations commands	4-23
4.6.1.1. Load commands	4-24
4.6.1.2. AND commands	4-25
4.6.1.3. OR commands	4-26
4.6.1.4. Exclusive-OR commands	4-27
4.6.1.5. Assignments and set commands	4-28
4.6.2. Arithmetic commands	4-29
4.6.3. Comparison commands	4-30
4.6.4. Shift and rotation commands	4-31
4.6.5. Byte and flag manipulation	4-32
4.6.6. Module calls	4-32
4.6.7. Jump commands	4-33
4.6.8. Copy and BCD commands	4-33
4.6.9. Programmable pulses, timers and counters	4-34
4.6.10. Special commands	4-35
4.6.11. Commands of the initialisation modules	4-35
4.6.12. Commands of the data modules	4-36
 4.7. Registers	 4-37
 4.8. Addressing	 4-38
4.8.2. Offset addressing	4-39
4.8.2.1. External operands used as offset	4-39
4.8.3. Types of addressing: summary	4-40

4.9. PLC reactions to errors and failures	4-41
4.9.1. Failures and errors summary	4-41
4.9.2. Undervoltage (supply, failure no. 2)	4-43
4.9.2.1. Without external supply	4-43
4.9.2.2. With external supply	4-43
4.9.3. Watchdog (program run time exceeded, failure #3)	4-45
4.9.4. PROFIBUS error (failures 4, 5, 6)	4-46
4.9.5. Checksum in user program (failure #8)	4-47
4.9.6. Hierarchy error (failure #9)	4-48
5. PC Control 645-500 used as programming card	5-1
5.1. What do you need?	5-2
5.2. Preparing the programming card	5-3
5.2.1. PC card settings	5-3
5.2.2. Setting as PROFIBUS card	5-4
5.2.3. Loading a C task into PC Control 645-500	5-6
5.3. On-line PROFIBUS	5-7
6. Communication with PC programs	6-1
6.1. PC interface driver	6-2
6.1.1. DOS program with process image interface	6-2
6.1.1.1. Components	6-2
6.1.1.2. Purpose of the program	6-2
6.1.1.3. Program functions	6-3
6.1.1.4. Practical examples	6-3
6.1.2. VisualBasic program with process image interface	6-4
6.1.2.1. Components	6-4
6.1.2.2. Purpose of the program	6-4
6.1.2.3. Program functions	6-4
6.1.2.4. Practical example	6-4
6.1.3. WINDOWS application with protocol interface	6-5
6.1.3.1. Components	6-5
6.1.3.2. Purpose of the program	6-5
6.1.3.3. Program functions	6-5
6.1.3.4. Practical examples	6-6
6.1.4. DOS program with protocol interface, VisiPro	6-7
6.1.4.1. Components	6-7
6.1.4.2. Purpose of the program	6-7
6.1.4.3. Program functions	6-8

6.1.4.4. Practical example	6-8
6.1.5.DOS program with protocol interface, TurboPascal	6-9
6.1.5.1. Components	6-9
6.1.5.2. Purpose of the program	6-9
6.1.5.3. Program functions	6-9
6.1.5.4. Practical example	6-9
6.1.6. DOS program with protocol interface -	
programming language link	6-10
6.1.6.1. Program components	6-10
6.1.6.2. Purpose of the program	6-10
6.1.6.3. Program functions	6-11
6.1.6.4. Practical example	6-12

A. Specifications A-1

A.1. Technical specifications A-1

A.2. Order specifications A-3

A.3. References to literature A-3

Index Index-1

Sales & Service

Table of contents

1. Purpose of PC Control 645-500

Automation more and more frequently relies on the use of PCs which are applied either for process visualization and data acquisition or even for control operations.

PC Control 645-500, a PLC in the shape of PC slot card, corresponds to this trend.

2 Models

- without external power supply:

<i>device</i>	<i>partnumber</i>
PCControl 645-500	645.425.01

- with 24 V DC external power supply:

<i>device</i>	<i>partnumber</i>
PCControl 645-500NT	645.425.02



In the following chapters we are using the term "PC Control 645-500" synonymously for both device types. Text passages making reference to the differences between the two are marked.

Tasks

- as a PLC, the device is used for process control relying on PROFIBUS for signal and data exchange with process-level components or other controllers
- if used in a PC environment, it interacts with the PC serving as process control system and interface between the process to be controlled on the one hand and process visualization and process data processing on the other
- as a PC card it can be used for the programming of Kuhnke controllers via PROFIBUS

1.1. Transition from individual controllers to network systems

Programmable logic controllers (PLCs) play an important role in industrial automation. There are three main reasons for this:

- they are universally applicable,
- programming is easy and comprehensible,
- they provide numerous means of testing and putting them into operation.

As problem-orientated 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 which have found acceptance in a wide range of action.

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 here perform on the process interfacing level whereas PCs are used for processing and managing large amounts of data on the control level.

Task separation leads to decentralisation

Integrating further components such as sensors and actuators produces network systems at the field level.

High-capacity interfaces and transmission lines are of vital importance because they are the hardware for communication between PLCs and further PLCs, other devices as well as PCs.

Advantages of decentralisation

- reduction of multicore cables,
 - material (cables, connectors..)
 - space (conduits, terminal blocks, switching cabinet)
 - installation (time, possibilities of errors)
- increased efficiency
- the program structure is similar to the object structure, this leading to more transparency
- reaction to problems (if one part of the systems fails other parts can continue to work)
- reduced setup times
- pre-testing of individual stations
- devices of different manufacturers can be combined

These advantages can only be fully effective if a standardised solution for all part-systems is available. This must offer an acceptable compromise for a vast majority of tasks.

Speed and reliability of data transfer and the design as an open system are of decisive importance here. The system to provide this compromise is PROFIBUS.

1.3. Remote programming via PROFIBUS

If programmed on-line, controllers are usually programmed locally via the RS232 (V.24) interface or, in the case of PC Control 645-500, via the PC bus. For programming, the programmer has to go to where the device is.

However, in distributed systems, controllers may be placed at locations very far from each other or at locations that are difficult to access. In extreme cases, there may be even several floors separating them. Think, for example, of wind power stations with a base station and a rotor station. Changing the rotor control program would require climbing up the tower.

As a new feature, it is now possible to program and put the following controllers into operation via PROFIBUS:

- Profi Control 680I
- Modu Control 657P
- KUAX 644
- PC Control 645-500

This not only **bridges the gaps** but also ensures that every PLC can be accessed from **one** location.

Another advantage of remote programming results from the fact that only **one single PC is required for programming**. To avoid having to permanently change the location of the programming PC it has been the usual method up to now to use a separate PC for every PLC. Consequently, in order to avoid getting lost in the network the network had to be up-dated time and again to react to changes of single projects, e.g. by integrating the sub-networks via VEBES.

This procedure has now been made redundant, except of course in cases where several stations have to be monitored simultaneously.

Please refer to chapter 5 to learn about the easy way of turning PC Control 645-500 into a programming PCB. While the hardware remains unchanged, the only prerequisite is to load a specific program.

1.3. Summary of PROFIBUS features

PROFIBUS is a field bus. Its name is an acronym of the term "Process Field Bus".

It was developed to network controllers (such as PC Control 645-500, Profi Control 680I, Modu Control 657P etc.) while providing an interface to the field level, i.e. to sensors and actuators (e.g. via decentralised input/output devices such as Profi I/O 690E, KUAX 680S).

PROFIBUS also interfaces to the control level where one or several central computers may be implemented to control the overall process.

Open communication

The principle of open communication is meant to guarantee interconnections between devices of different manufacturers. The standard for this field bus is set in Euronorm "EN 50 170, Volume 2, PROFIBUS".

Topology

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

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

Station address

Each PROFIBUS station is assigned its own station address under which it can be addressed by the other stations. Valid addresses are in the range of 0 to a maximum of 126. PC Control 645-500 has assigned its station address by VEBES, the network configurator.

VEBES

VEBES, the network operating software, is the PROFIBUS configurator for Kuhnke PROFIBUS masters. The program is used for defining bus parameters, creating station lists and setting up the communication paths between individual stations.

Bus protocols

PROFIBUS provides various protocols (profiles) to be able to react to different requirements.

PC Control 645-500 supports the following protocols:

- PROFIBUS-FMS (master)
- PROFIBUS-DP (class 1 master)
- and preliminary standard SINEC L2-DP (master)

As an all-protocol master, PC Control 645-500 even supports all of the protocols above in one single network, the advantage being that a great variety of communication partners can be integrated.

Baudrate and other bus parameters

PC Control 645-500 supports the following baudrates for communication via PROFIBUS:

9.6 kbit/s, 19.2 kbit/s, 93.75 kbit/s, 187.5 kbit/s, 500 kbit/s

All bus parameters for the network, including the baudrate, are set via VEBES and transmitted to PC Control 645 together with the KUBES project. Changing any hardware settings on the card itself is not required.



For a comprehensive description please refer to instruction manual PROFIBUS, E 365 GB.

1.4. PC Control 645-500 as a successor to KUAX 644

PC Control 645-500 meets all prerequisites to be called successor to KUAX 644. Even in fully configured networks can the latter be replaced by PC Control 645-500.

Make sure to observe the following general rules:

- The PC interface (ISA) is to be addressed in the I/O range (see ch. "3.4.1. Addressing ...").
- In the VEBES (version 3.0 or higher) "Define stations" dialog, replace device type 644 by 645-500.
- Use KUBES (version 5.10 or higher) to transfer the project into PC Control 645-500.
- Bus parameters and baudrate are not longer set via coding switches. This information is automatically written into the project by VEBES and then transferred by KUBES.

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,
- statistic 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 to correspond to European standard EN 60742 (corresponds to VDE 0551)
- In case of power breakdowns or power fades: the program has to be structured in such a way as to create a defined state at restart that excludes dangerous states.
- Emergency switch-off installations have to be realised in accordance 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.
- The relevant standards and VDE regulations are to be observed in every case.
- Control elements have to be installed in such a way as to exclude unintended operation.
- Control cables have to be layed in such a way as to exclude interference (inductive or capacitive) which could influence the controller operation or 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!) detracted from their proper functioning.
- 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 determined 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

Electrostatic discharge, ESD
in accordance with IEC 801-2, 3rd degree of sharpness

Fast transient interference, Burst
in accordance with IEC 801-4, 3rd degree of sharpness

Irradiation resistance of the device, HF
in accordance with IEC 801-3, 3rd degree of sharpness

Immunity to damped oscillations
in accordance with IEC 255-4 (1 MHz, 1 kV)

2.5.3. Interference emission

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



If the controller is designed for use in residential districts, 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:

data signals, shielded
analogue signals, shielded

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

2.5.7. Location of installation

Make sure that there are no impediments due to temperatures, dirt, impact, vibrations 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 asking the manufacturer or supplier of the corresponding actuators for the relevant information.

3. Hardware

3.1. Design

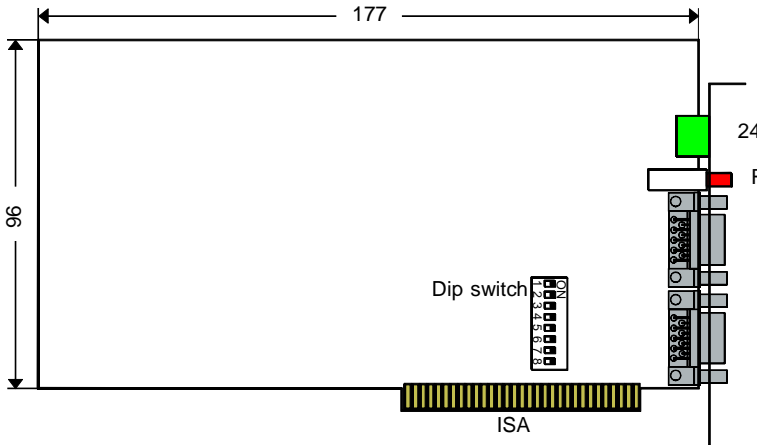


Fig.: PC Control 645, location of user components

The illustration above is a side and a front view of the device showing the location of the components that the user must be able to access:

User components

24 V DC	external power supply, 2-pin screw-type locking connector, only in PC Control 645-500 NT (645.425.02)
Reset	push-button for resetting the card
V.24	V.24 connector for programming and data communication, 9-pin female D-sub connector
Bus	PROFIBUS interface, 9-pin female D-sub connector
Dip switch	coding switches SW1...SW8, addressing, reset, wait-states and interrupt request
ISA	interface to PC bus

3.2. Power supply

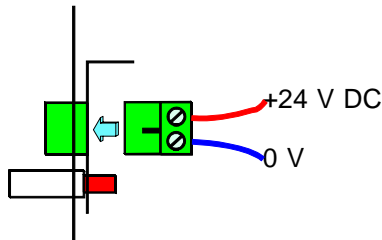
PC Control 645-500 (part no. 645.425.01)

- 5 V DC / 0.4 A from the PC via the ISA interface

PC Control 645-500NT (part no. 645.425.02)

either internal

- 12 V DC / 0.25 A from the PC via the ISA interface
- or external
- voltage: 24 V DC -20%+25% / 0.15 A
- connector: 2-pin screw-type locking connector



Due to the external power supply, the device can also be operated independently of the PC.



In case of power failures of the external power supply, the PC automatically takes over the supply without interruption, if the device is operated as a PC slot-in card and if the PC is on.

Due to the built-in **undervoltage monitoring function** it is possible to detect power breakdowns early. The monitoring function only works while the card is being **supplied externally exclusively**. Include suitable measures in your user program to ensure a safe status in case of failures (see ch. "4.9. Reactions to errors and failures").

3.3. Reset button

The card is reset by pushing the red Reset push-button (see ch. "3.4.2. Reset").

Card operation is restarted after releasing the Reset button.

3.4. Coding switch

The 8-pin coding switch (see ch. "3.1. Illustration", DIP switch) has several settings:

3.4.1. Addressing the (ISA-)PC interface

The ISA interface enables PC Control 645-500 to communicate with the PC. For this purpose, PC Control 645-500 is equipped with a dual-port RAM whose size varies according to the type of addressing (see table):

<i>addressing</i>	<i>dual-port RAM</i>
I/O range	1 kbyte
memory range	2 kbyte

Use switch SW4 to set the address range – I/O range or memory range – via which PC communication is carried out. The starting addresses within the set range are defined by switches SW1...SW3:

Type of addr.			SW4					
Address range			off			on		
SW1	SW2	SW3	I/O range addressing			Memory range addressing		
			<i>see ch. "3.4.1.1..."</i>			<i>see ch. "3.4.1.2..."</i>		
off	off	off	0x0120	-	0x1F60	0xD000	-	0xD07F
on	off	off	0x2120	-	0x3F60	0xD200	-	0xD27F
off	on	off	0x4120	-	0x5F60	0xD400	-	0xD47F
on	on	off	0x6120	-	0x7F60	0xD600	-	0xD67F
off	off	on	0x8120	-	0x9F60	0xD800	-	0xD87F
on	off	on	0xA120	-	0xBF60	0xDA00	-	0xDA7F
off	on	on	0xC120	-	0xDF60	0xDC00	-	0xDC7F
on	on	on	0xE120	-	0xFF60	0xDE00	-	0xDE7F

Default setting:

SW1=off, SW2=off, SW3=off, SW4=off

continued overleaf



3.4.1.1. I/O range addressing

Sixteen address fields of 64 kbyte each are used in the PC I/O range. That totals 1 kbyte.
These address fields are not located immediately one after the other. The inter-location space of 448 kbyte each can be used by other hardware such as graphics adapters, interfaces, etc.

The sixteen 64 kbyte fields used are distributed as follows (depending on the starting address):

Example for setting 0x0120:

	0x0..								0x1..							
from	..120	..320	..520	..720	..920	..B20	..D20	..F20	..120	..320	..52	..720	..920	..B2		
to	..160	..360	..560	..760	..960	..B60	..D60	..F60	..160	..360	..56	..760	..960	..B6		



please also refer to ch. "3.8.2. Measures for the prevention of address conflicts"

3.4.1.2. Memory range addressing

In this case the memory space used occupies a consecutive range of 2 kbyte.



please also refer to ch. "3.8.2. Measures for the prevention of address conflicts"

3.4.2. Setting the reset trigger

A reset has the following effects:

- the program run is executed
- communication via PROFIBUS, V.24 and ISA interfaces is interrupted
- all non-remanent markers (in PLC operation) are reset (value "0")

Reset triggers

Always possible:

- push Reset button on PC Control 645
- switch on external power supply (24 V DC) if internal supply via the PC is not activated
- switch on PC if there is no external supply

Applies only if **SW5=on**:

- push PC Reset button
 - switch PC power supply on or off
- this will reset the card even if it is externally supplied*

Default setting:

SW5=off



A so-called warm boot of the PC using key combination <Ctrl>+<Alt>+ does not reset PC Control 645.

3.4.3. Setting the I/O wait states

So-called wait states are used for I/O range addressing (SW4=off) to ensure safe PC access to the dual-port RAM of PC Control 645-500. Use switch SW6 to define whether PC-BIOS wait states are used or whether they are shortened:

SW6	off	I/O wait states	as in BIOS, default setting
	on		shortened (only if SW4 = off)

Basic setting:

SW6=off

3.4.4. Setting the interrupt "PC Control 645-500 -> PC"

PC Control 645 is capable of triggering an interrupt in the PC (PC-IRQ 3 or PC-IRQ5). This is a practical feature if information coming from PC Control 645 is to be processed directly in the PC.

Use switches SW7 and SW8 to make the interrupt settings:

SW7	SW8	Function	
off	off	no interrupt, default setting, compulsory setting for KUBES (ONLINE PC)	
on	off	PC-IRQ3	only allowed if appropriate interrupt routine exists in the PC
off	on	PC-IRQ5	
on	on	illegal setting	

Default setting:

SW7=off, SW8=off

Prerequisites for interrupt operation

To be able to react to an interrupt, the PC must have an appropriate interrupt routine.

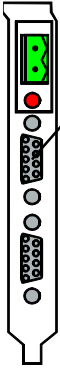


If this is not the case, interrupts will infallably cause the PC to crash as soon as one of switches SW7 or SW8 is on.

Always choose the default setting, if you want KUBES to communicate with PC Control 645-500 with option "ONLINE PC" (via the ISA interface).

3.5. Interfaces

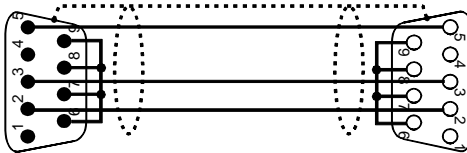
3.5.1. V.24 interface



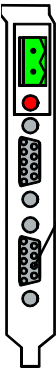
The V.24 interface is located on the upper of the two D-Sub connectors. It can be used for various tasks such as programming via an external PC, communication with barcode scanners, scales, systems for process visualisation, etc. The interface is a female 9-pin connector with the following pin wiring:

<i>pin</i>	<i>function</i>
casing	shield
2	TxD
3	RxD
5	Gnd

The following cable is to be used to establish PC connections for programming etc (part no. 657.151.03):



3.5.2. PROFIBUS interface

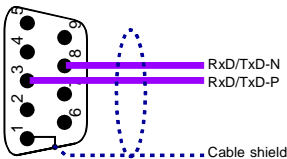


The PROFIBUS interface is located on the lower of the two D-Sub connectors. It is a female 9-pin connector with the following wiring:

<i>pin</i>	<i>function</i>
1 and casing	shield
3	RxD/TxD-P
5	Data-Gnd
6	VP (+5 V)
8	RxD/TxD-V

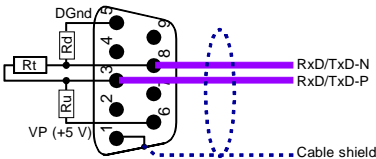
3.5.2.1. PROFIBUS cable connector

Connectors 3 and 5 are to be connected 1:1 (without crossover) to establish a connection to the communication partners:



3.5.2.2. PROFIBUS bus termination

PROFIBUS is established as a line. Both terminating stations are to be connected to a bus termination:



<i>resistor</i>	<i>type A cable</i>	<i>type B cable</i>
R _t	220 Ω	150 Ω
R _u	390 Ω	390 Ω
R _d	390 Ω	390 Ω

3.6. Memory

PC Control 645-500 is equipped with large memory ranges:

- flash EPROM, 512 kbyte *)
For: programs and permanent data
- RAM, 256 kbyte *)
For: variable data, operand ranges
- dual-port RAM, 2 kbyte
For: data transfer to the PC via ISA interface



**) If PC Control 645-500 is used as PLC, a certain part of the memory is reserved for the monitor program. Please refer to chapter "4.3. Setting up the memory" to learn how much user memory is available.*

3.7. Ni-Cd accumulator

PC Control 645-500 has a Ni-Cd accumulator for buffering defined RAM areas. In the PLC these are the remanent markers, i.e. bit and byte ranges that are not reset in Reset events (see above).

Max. buffer time is about 36 days at an ambient temperature of 0...40 °C.



Due to different lengths of time spent in the warehouse the charging state of the accumulator is undefined at the time of delivery.

Please supply the device with voltage for at least 72 hours to charge the accumulator.



In chapter "4.5.1.2. Testing the accumulator functions" you will find a program suggestion for testing the accumulator functions.

3.8. Installation

PC Control 645-500 can be used in all normal IBM compatibles equipped with ISA slots.

3.8.1. Installing the card

Please proceed as follows:

- set the coding switches as you need them for your intended application (see ch. "3.4. Coding switches").
- switch off the PC power supply and pull out the cable to be on the safe side
- make sure that the card is not externally supplied with 24 V DC either
- open the PC casing
- choose a free ISA slot
- remove the metal slot cover



*Never put the card in or take it out while the PC is running.
You might otherwise ruin the device.*

- push PC Control 645-500 into the slot with the PC connector strip (ISA) facing down and the metal slot panel out
- screw the slot panel in where the metal slot cover was before
- close the PC casing
- restart the PC
 - > PC Control 645-500 is started automatically as it is supplied via the PC.
- provide an external 24 V DC power supply if required ("3.2. Power supply").
- plug in the bus cable (see ch."3.5.2. PROFIBUS interface")

3.8.2. Measures for the prevention of address conflicts

PC Control 645-500 is used in various PC configurations. It is mandatory to take certain precautions to avoid provoking address access conflicts with other PC slot cards.

There are two basic address ranges within the PC address space that can be used: the I/O range and the memory range (see ch. "3.4.1. Addressing of...").

3.8.2.1. I/O range addressing

PC Control 645-500 completely analyses the I/O address (16 bit). This gives you the possibility of safely differentiating between up to 8 PC Control 645-500 within any one PC.

The address set via the coding switch is the starting address of an address range of 16x64 Byte (=1 kbyte).



Make sure to absolutely avoid overlapping address ranges if other PC slot cards are used that are also addressed via the I/O address range.

If these include PC slot cards that do not fully code the I/O address (16 bit) non-conflict conditions may prove to be impossible to create. Use the memory range instead in such cases.

3.8.2.2. Memory range addressing

The address set via the coding switch is the starting address of a 2 kbyte address range (segment addressing).



Keep this memory range reserved for PC Control 645-500 to avoid any address conflicts.

For this purpose, include exclusive system entries in all system files calling up memory managers.

It depends on your PC and the operating system you are running what entries have to be made.

Example for memory address D000

- start an ASCII editor program
 - under DOS: e.g. EDIT.EXE
 - under Windows: e.g. SYSEDIT.EXE
- load the CONFIG.SYS system file and add the following parameters to command line EMM386 : X=D000-D07F
e.g.:
DEVICE=C:\DOS\EMM386.EXE X=D000-D07F
- load the Windows SYSTEM.INI system file and add the following line to section [386Enh]:
EmmExclude=D000-D07F

4. PC Control 645-500 used as PLC

At this point we're assuming that your PC Control 645-500 has been coded correctly and installed in the PC as well as that it is being supplied with voltage. If this is not the case please first follow the instructions given in chapter "3.8. Installation".

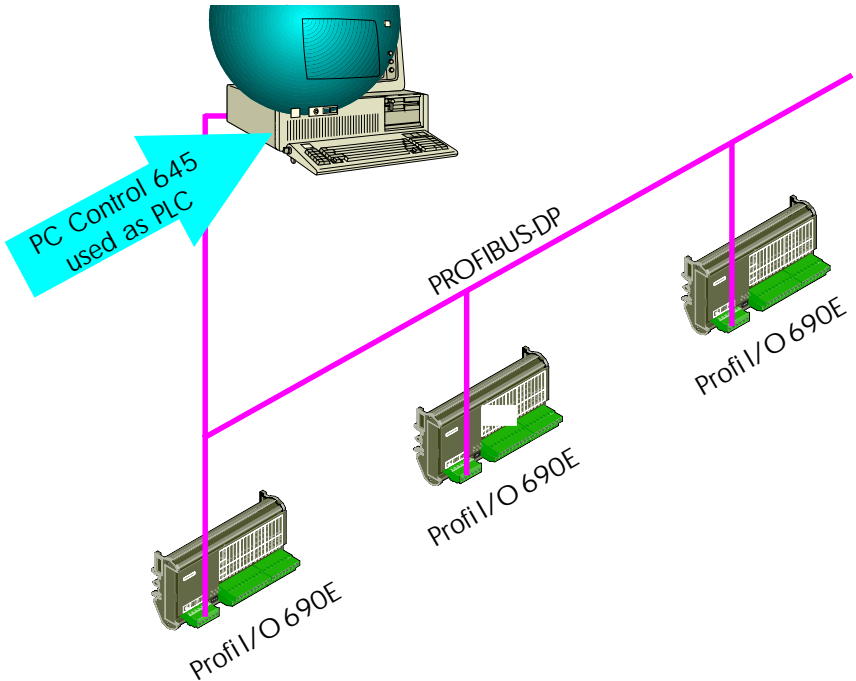


Fig.: PC Control 645-500, installed in a PC and networking with 3 decentralised I/Os via PROFIBUS-DP.

4.1. Integration into PROFIBUS

Programmable logic controllers are used to control machines, plants etc. To be able to do so it has to exchange data with the process components which is usually done via inputs and outputs.

PC Control 645-500 has no local I/Os. It was designed for data exchange via PROFIBUS. It is therefore to be integrated into a PROFIBUS network where other devices exist to decentrally reading from or outputting to the I/Os.

Prerequisites

- KUBES version 5.10 or higher
- VEBES version 3.0 or higher

Before PC Control 645-500 can be operated on PROFIBUS it is to be programmed **in a network project** and **not in a single project** in the first place.

This means that project planning should follow a certain sequence:

- 1 use VEBES to create a network
- 2 set the PROFIBUS parameters
- 3 define all network stations
- 4 set the communication paths between all stations
- 5 switch to KUBES
- 6 use KUBES to write the PLC user program always with the external operands (PROFIBUS) in mind.



These steps will be explained in more detail on the following pages. We are assuming that you know how to work with VEBES and KUBES.

4.1.1. Creating a network

Start VEBES, the Kuhnke network operating software, and create a new network:

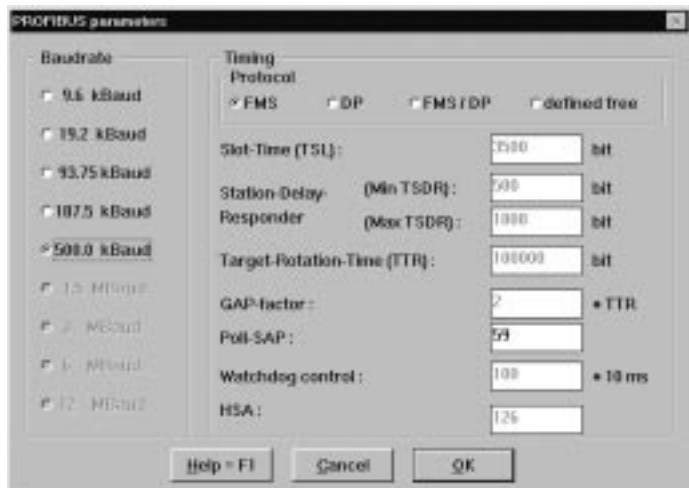
- "Network" menu
- option "Create..."



4.1.2. Defining PROFIBUS parameters

Define the PROFIBUS parameters as required (here: PROFIBUS-FMS, standard parameters)

- "Line" menu
- option "Bus parameters..."



4.1.3. Defining the network stations

Enter PC Control 645-500 and all other devices that are to be networked via the same bus line as stations (in this case: PC Control 645-500 and KUAX 680S)

- "Line" menu
- option "Stations..."

Station/project	Addr	Station types	bit/byte	byte
MASTER_1	1	645-500	32	32
SLAVE_1	2	680 S	16	16

Settings

- Station / project: name of the station
If the device entered here is a master, the name you enter will also become the project name under KUBES.
- Addr.: PROFIBUS station address
- Device type: type of station
- Configuration: number of signals sent via the bus

4.1.4. Defining the communication paths

Define the communication paths (in this case: connection via process image in both directions)

- "Line" menu
- option "Connections - Process image..."



4.1.5. Switching to KUBES

Switch to KUBES "with project". The project name is the same as the station name allocated to PC Control 645-500.

- "Switch to KUBES" menu
- option "with project"



Use KUBES to write the user program. Refer to chapter "4.5.2. External operands as process image" to learn how decentralised inputs and outputs are integrated in the user program.

4.2. Going on-line with KUBES

It is a prerequisite that an on-line connection to the PLC be made via KUBES so that transferring programs, setting the memory, commissioning and testing of programs is possible.

The on-line connection can be made in various ways:

- Online V.24
is the connection from the V.24 interface of PC Control 645-500 to the COM_n port of the PC
- Online PC
is a connection made via the PC bus (ISA)
- Online PROFIBUS
is a connection via PROFIBUS, a PC Control 645-500 is used as programming card in the programming PC

On the following pages you will find a description of how the on-line connections are established by KUBES. Data is written into the "PROFISOF.INI" system file which is stored in the Windows root directory.

4.2.1. Online V.24

Choose this type of on-line connection if PC Control 645-500 is not plugged into the PC on which KUBES is running.

Plug one end of programming cable with part no. 657.151.03 into the female connector reserved for that purpose on PC Control 645-500, and the other end into the PC COM port that was defined for that during KUBES installation (see "PROFISOF.INI" file in the Windows directory).



For interface and programming cable information see ch. "3.5.1. V.24 interface"

V.24 settings under KUBES

- "PLC" menu
- option "Online settings - V.24 parameters"



- PC interface: COM1 - COM2
- transfer rate: 9600...57600 baud
- parity check: even - **odd** - no

The suggested settings result in the following entries in PROFISOF.INI:

SPS=COM1 :
V24-PARAM=9600,0,8,1

To establish the on-line connection

- "PLC" menu
- option "ONLINE - V.24"

4.2.2. Online PC

Choose this type of on-line connection if PC Control 645-500 is installed in the PC on which KUBES is running.

This is the fastest connection as it directly accesses the PC bus via the ISA-PC interface.

PC card settings under KUBES

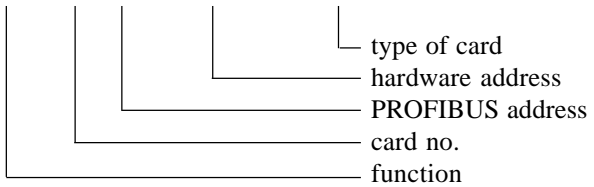
- "PLC" menu
- option "Online settings - PC cards"



- type of card: 645-500
Choosing "none" and "accepting" the choice causes all card entries in PROFISOF.INI to be cleared
- card no.: 1...8
You can use up to 8 cards on any one PC
- PROFIBUS addr. 0...125
Enter the same PROFIBUS station number that was assigned to PC Control 645-500 during VEBES network configuration (see ch. "4.1.3. Defining the network stations")
- hardware address 0120...E120 (I/O range)
D000...DE00 (memory range)
This setting has to be the same as the setting of the coding switches (see ch. "3.4.1. Addressing the (ISA-)PC interface").
- accept:
Click here to accept the settings chosen for your card. This will cause an entry in PROFISOF.INI to be made.

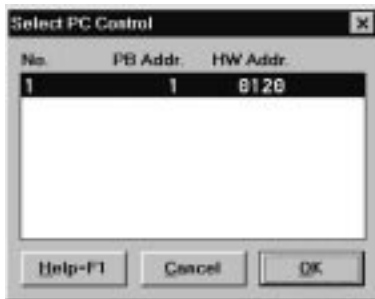
Entries in PROFISOF.INI

AT_SPS_1=1,0x0120,645



To establish the on-line connection

- "PLC" menu
- option "ONLINE - PC"



This box lists all PC Control 645-500 that were previously entered in PROFISOF.INI (see previous page and above). Choose the card to which the on-line connection is to be made.

4.2.3. Online PROFIBUS

Choose this type of connection if

- the PLC to be programmed (PC Control 645-500) is connected to a PROFIBUS network
- and if a PROFIBUS card is installed in the PC that is used by KUBES to access the same PROFIBUS network.

This card may be a PC Control 645-500 which is not used as PLC but one on which a PROFIBUS task is running (P645_IO.Tnn or P645_MEM.Tnn).



In this case please read chapter "5. PC Control 645-500 used as programming card".

4.3. Setting up the memory

Apart from the operands (see ch. "4.5. Operands") the following memory types and capacities are available for the user:

- flash EPROM, 320 kbyte
for: program and permanent data
- RAM, 112 kbyte
for: variable data

The memory space is divided up by use of the corresponding KUBES programming software functions.

Prerequisites:

- open project
- KUBES on-line
- PLC stop (options "Stop" or "Stop and reset")

While in on-line mode, choose option "Online settings" from the "PLC" menu. The following dialogue will be displayed:



program
range

distribution	type of memory	size
banks 0...4	flash EPROM	256 kbyte
banks 5...6	RAM	112 kbyte



The memory distribution in the illustration above is shown without data memory. Read the next page to learn how to reserve data memory.

To reserve data memory

Data stored in the flash EPROM cannot be modified while the PLC is running. Data stored in the RAM, however, can be modified.

Proceed as follows:

- Decide where you want to store the data and how big you want the data ranges to be.
- In the dialog box (see below), enter a value for the program range in the selected bank that is smaller by the same amount as the intended size of the data memory.
- > The memory range reserved for data is shown as hexadecimal address range.

Example:



In the example above, a RAM data range of 32 kbyte have been reserved in bank 5. It is located in address range \$8000...\$ffff.

Like this you can also reserve data memory in several banks.

4.4. PLC working method

The microprocessor for the user program receives its instructions from two memory sources:

- > monitor program memory
- > user program memory

The monitor program contains all system features of the controller PC Control 645-500. It is delivered with the controller.

The user program is written by the user based on the functionality of the KUBES programming software.

As it is only practical to run PC Control 645-500 when networking with other devices, it has to be configured as a station in a PROFIBUS network. This configuration task is taken care of by VEBES, the network operating software.

In the following chapters you will find all information required for writing user programs for PC Control 645-500.



The way how to create a network configuration or how input the user program does not lie within the scope of this manual. For the relevant information please refer to the manuals listed below:

- E 327 GB *Beginner's Manual KUBES*
 - E 315 GB *Beginner's Manual VEBES*
- as well as to the appropriate on-line help systems*
- and*
- E 417 GB *Programming Manual for Kuhnke PLCs*

4.5. Operands

PC Control 645-500 differentiates between two types of operands, i.e. "local" and "external" ones.

4.5.1. Local operands

Local operands are immediately available in the controller.

Summary

Group	Input	Function	Type	Qty.	Input range		Comment
					from	to	
M	M_____	markers	bit	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	
O	O_____			256	O00.00	O15.15	see "Inputs and outputs" on next page
SO	SO_____			256	SO00.00	SO15.15	
R	R_____	remanent markers		256	R00.00	R15.15	buffered by accumulator *)
SR	SR_____			256	SR00.00	SR15.15	
BM	BM_____	byte markers	byte	256	BM00.00	BM15.15	
SBM	SBM_____			256	SBM00.00	SBM15.15	
BO	BO_____			256	BO00.00	BO15.15	see "Inputs and outputs"
DB0	DB0_____	byte markers to be used as "data processing ranges" for data modules		256	DB000.00	DB015.15	these operands can also 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_____	remanent byte markers	byte	256	BR00.00	BR15.15	buffered by accumulator *)
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	
AO	AO_____	word markers	word	128	AO00.00	AO07.15	
SLG	SLG_____	DP-RAM markers	byte	256	SLG00.00	SLG15.15	DP-RAM buffer storage (for PC data)
SLH	SLH_____			256	SLH00.00	SLH15.15	
C	C_____	progr. counters	word	32	C00.00	C01.15	
PT	PT_____	progr. timers		128	PT00.00	PT07.15	
PC	PC_____	progr. clock	byte	4	PC00.00	PC00.03	
PP	PP_____	progr. pulse	bit	128	PP00.00	PP07.15	
PL	PL00.00 PL00.01	logical 0 logical 1		1 1	PL00.00 PL00.01		
ERR	ERR00.00	system error	byte	1	system error messages (see appendix "D. Reactions to failures and errors")		

*) buffered by Ni-Cd accu on the module

4.5.1.1. Short description of local operands

Inputs and outputs

You can connect no local inputs and outputs to PC Control 645-500. You can, however, use the output addresses as additional markers.

Bit, byte, word markers

There are bit markers, byte markers (8 bit) and word markers (16 bit) available for storing (marking, noting) current data. The contents (values) of remanent markers are not changed by RESET or restart. All other markers are cleared at restart.

Timers

PC Control 645-500 has, as a standard, 128 software timers (PT00.00-PT07.15). The time range is from 10 ms - 65535 s. These timers can be programmed with raising or falling delay or as clock pulse or pulse generators respectively. If required they can be remanent.

Counters

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

DP-RAM markers

Address ranges SLG and SLH are used as process chart memory for data exchange processes between PC and PLC, e.g. for process visualisation. They are used in combination with KUBES module PA_MOVE.

System error marker "ERR00.00"

Once system errors have been detected, they are written into byte operand (8 bit) "ERR00.00" by the monitor program. They can be read and analysed by the user program.

Operands

4.5.1.2. Testing the accumulator functions

The built-in accumulator serves buffering remanent operands. It is possible to test its functions via the user program. With KUBES running, proceed as follows:

- Choose two free directly consecutive byte markers (e.g. BR15.14 and BR15.15).
- Input a bit pattern into these before the first start;
suggestion: BR15.14=aa, BR15.15=55 (hexadecimal values)



Call up the "Display address range" (command ^F5)

- Include a program at the beginning of your Organisation Module with the purpose of checking the bit pattern at every start. There are various possible reactions to errors.

Program suggestion

```
; Accumulator function test
; -----
      LD      ACCU_TS1      BR15.14 ; (remanent byte marker for accu test)
      CMPD    $55AA          ; does bit pattern still exist?
      JP=     START         ; yes -> start program

; reaction to errors
; for example:
; - PLC reset (used in this case)
; - increment one byte marker (for KUBES' dynamic display)
; - have a (decentralised) output flash
; - generate text for terminal display
; - warning message on process visualisation screen
; - message to other station(s) via PROFIBUS

ERROR   RESET              ; reset PLC if accu discharged

; start user program
; -----
START   NOP
```

4.5.2. External operands as process image

At this point we will only describe PROFIBUS communication via the process image.



Block transfer communication is described in the PROFIBUS manual, E 365 GB.

External operands are read as inputs or controlled as outputs via PROFIBUS. There are various sources and destinations:

- decentralised input/output devices (e.g. Profi Control 680S)
- other programmable logic controllers
- positioning controllers
- etc.

As a window to PROFIBUS, there is a 496 byte RAM memory available to PC Control 645-500 for the external operands. This memory represents the process image of the available external operands.

VEBES creates a list of the external operands

VEBES (see instruction manual E 315 GB, VEBES) is used for network configuration. This process creates projects for all participating Kuhnke controllers which are then programmed / edited with the KUBES functionality.

During network configuration with VEBES, the programmer determines the communication partners and the amount of communication objects (external operands).

VEBES adds these external operands to the list of operands in the corresponding project and distributes them within the process image memory. During programming, they can then be addressed like local operands. It is also possible to use mnemonics in the symbol table.

PROFIBUS updates the process image

During operation, PROFIBUS automatically takes care of transmitting data between communication partners and process image as often as possible. As this is done at certain time intervals and controlled by interrupts, up-dating may also occur

Operands

while a (program) module is being processed.



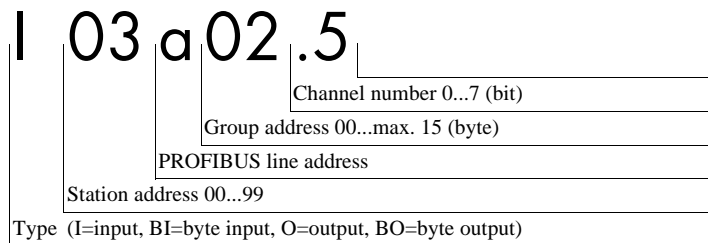
This is why process image data may be different at the beginning and the end of a module.

This does not apply to interrupt and timer modules.

4.5.2.1. Example 1: process image of Profi Control 680S (Slave)

Profi Control 680 is a decentralised input and output device. The size of the process image depends on its configuration. The modules are plugged into the chassis. Each module occupies one byte in the process image.

Designation of operands



- Type

These are the inputs and outputs that Profi Control 680S is equipped with:

- Inputs are read via PROFIBUS
- Outputs are controlled via PROFIBUS

- Station address

Address of the PROFIBUS communication partner. PC Control 645-500 can basically communicate with all partners with an address between 00 and 99.

- PROFIBUS line address

Always "a"..

- Group address

The group address is identical with the module number (modules plugged into Profi Control 680S are numbered through from left to right. Inputs: I00...04/07, outputs: O00...04/07).

- Channel number

Only applies to bit addressing. The channel number corresponds to the channel number of the module, i.e. to the input or output.

Addressing

- Bit addressing (for addressing one channel)

The operand is addressed by "I" (input) or "O" (output). The desired bit is selected by the channel number.

Examples: L I03a02.5 ;read external input
 = M01.02 ; and write into local marker

 L M00.03 ;read local marker
 = O02a01.7 ; and write into external output

- Byte addressing (for addressing all channels of a module)

Address 1 byte (8 bit) by one command. Inputs are addressed with "BI", outputs with "BO". There is no indication of a channel number.

Examples: L BI03a02. ;read 2 external byte inputs
 = BM01.00 ; and write into byte marker

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

- Word addressing (for addressing all channels of 2 modules)

Address 1 word (2 byte or 16 bit) by one command. Inputs are addressed with "BI", outputs with "BO". The channel number is not indicated.

Examples: 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

Operands

4.5.2.2. Example 2: Process image of a controllers (master)

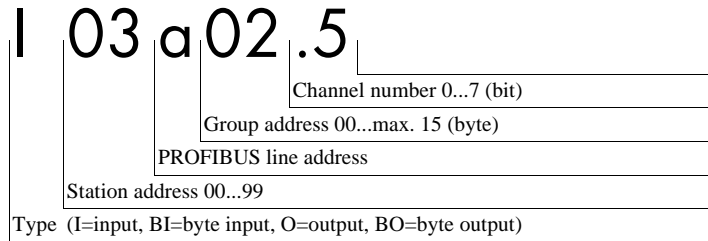
In the process image memory of the master, communication partners corresponding to the controller profile by default occupy 16 byte of the available 496 byte:

- 8 byte for input signals (groups 00 ... 07)
- 8 byte for output signals (groups 00 ... 07)



A second master in the network is only supported by the PROFIBUS-FMS protocol.

Designation of operands



- Type

The type signifies the relation between operand and controller:

- Inputs are read via PROFIBUS
- Outputs are actuated via PROFIBUS

- Station address

Address of the PROFIBUS communication partner. The PC Control 645-500 can basically communicate with all partners with an address between 00 and 99.

- PROFIBUS line address

Always "a".

- Group address

A group has a size of one byte.

- Channel number

Selects one bit of the indicated group (see below "Bit addressing")

Addressing

The operands in the process image can be addressed by byte or by word. Either type uses the same memory addresses, however.

- Bit addressing

Only possible in the first 4 byte (groups 00 to 03). This type of addressing is used for transferring binary signals of inputs, outputs or markers.

The operand is addressed by "I" (input) or "O" (output). The required bit is selected by the channel number.

Examples: L I03a02.5 ;read external input
 = M01.02 ; and write into local marker

 L M00.03 ;read local marker
 = O02a01.7 ; and write into external output

- Byte addressing

Address 1 byte (8 bit) by one command. Inputs are addressed with "BI", outputs with "BO". There is no indication of a channel number.

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 (address all channels of 2 modules)

Address 1 word (2 byte or 16 bit) by one command. Inputs are addressed with "BI", outputs with "BO". The channel number is not indicated. Processing of words is only possible with even group addresses (00, 02, 04...).

Examples: 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

4.5.2.3. PROFIBUS messages

PC Control 645-500 can receive three types of PROFIBUS messages (and partly also send them). During network configuration, VEBES reserves specific operand ranges for these messages. These are

Messages:	Operand range:
- Error messages	PFxx.yy
- Status messages	PSaxx.yy
- Event notifications	PEaxx.yy



If external operands are linked in a program (e.g. inputs of a Profi I/O 690E), it has to be taken special care of their validity. If, for example, the bus connection is interrupted the received value may no longer correspond to the real status of the input. In order to avoid wrong switching operations as a result we urgently recommend to carefully analyse the above-mentioned PROFIBUS messages in the program.



See instruction manual PROFIBUS, E 365 GB, for an extensive description of these messages.

4.6. Summary of commands

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

Programming of external operands

The execution times contained in the following tables are valid if a connection between the relevant PC Control 645 and the external participant has been established.



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

4.6.1. Logical operations commands

Logic commands are the commands for logical operations between operands. This includes assignments of results.

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



For further information on programming, use of the commands and programming examples please refer to our Programming Manual, E 417GB.

Summary of commands

4.6.1.1. Load commands

Com- mand	Operand (example)	Byte	Time [μs]	Function	
L	M00.00	4	0.45	Load	1bit address
	BM00.00	4	0.45		8bit address
	100	4	0.3		8bit constant
	M00.00[10]	8	0.75		1bit address with constant offset
	M00.00[BM01.00]	14	3		1bit address with variable offset
	BM00.00[10]	8	0.75		8bit address with constant offset
	BM00.00[BM01.00]	14	3		8bit address with variable offset
	I03a00.0				decentralised 1bit address
	BI03a00.				decentralised 8bit address
LN	M00.00	6	0.6	Load Nega- tion	1bit address
	BM00.00	6	0.6		8bit address
	M00.00[10]	10	0.9		1bit address with constant offset
	M00.00[BM01.00]	16	3.2		1bit address with variable offset
	BM00.00[10]	10	0.9		8bit address with constant offset
	BM00.00[BM01.00]	16	3.2		8bit address with variable offset
	I03a00.0				decentralised 1bit address
	BI03a00.				decentralised 8bit address
LD	BM00.00	4	0.45	Load 16bit	even 8bit address (00.00+00.01)
	BM00.01	10	3		odd 8bit address (00.01+00.02)
	AO00.00				16bit address
	1000	4	0.3		constant
	BM00.00[10]	16	3		8bit address with constant offset
	BM00.00[BM01.00]	22	5.3		8bit address with variable offset
	BI03a00.				decentralised 8bit address (00.+01.

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

4.6.1.2. AND commands

Com- mand	Operand (example)	Byte	Time [μs]	Function	
A	M00.00	4	0.45	AND	1bit address
	BM00.00	4	0.45		8bit address
	100	4	0.3		8bit constant
	M00.00[10]	10	0.9		1bit address with constant offset
	M00.00[BM01.00]	16	3.2		1bit address with variable offset
	BM00.00[10]	10	0.9		8bit address with constant offset
	BM00.00[BM01.00]	16	3.2		8bit address with variable offset
	I03a00.0				decentralised 1bit address
	BI03a00.				decentralised 8bit address
AN	M00.00	8	0.75	AND Nega- tion	1bit address
	BM00.00	8	0.75		8bit address
	M00.00[10]	12	1.1		1bit address with constant offset
	M00.00[BM01.00]	18	3.3		1bit address with variable offset
	BM00.00[10]	12	1.1		8bit address with constant offset
	BM00.00[BM01.00]	18	3.3		8bit address with variable offset
	I03a00.0				decentralised 1bit address
	BI03a00.				decentralised 8bit address
AD	BM00.00	6	0.6	AND 16bit	even 8bit address (00.00+00.01)
	BM00.01	10	2.6		odd 8bit address (00.01+00.02)
	AO00.00				16bit address
	1000	4	0.3		constant
	BI03a00.				decentralised 8bit address (00.+01.

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

Summary of commands

4.6.1.3. OR commands

Com- mand	Operand (example)	Byte	Time [μs]	Function	
O	M00.00	4	0.45	OR	1bit address
	BM00.00	4	0.45		8bit address
	100	4	0.3		8bit constant
	M00.00[10]	10	0.9		1bit address with constant offset
	M00.00[BM01.00]	16	3.2		1bit address with variable offset
	BM00.00[10]	10	0.9		8bit address with constant offset
	BM00.00[BM01.00]	16	3.2		8bit address with variable offset
	I03a00.0				decentralised 1bit address
	BI03a00.				decentralised 8bit address
ON	M00.00	8	0.75	OR Nega- tion	1bit address
	BM00.00	8	0.75		8bit address
	M00.00[10]	12	1.1		1bit address with constant offset
	M00.00[BM01.00]	18	3.3		1bit address with variable offset
	BM00.00[10]	12	1.1		8bit address with constant offset
	BM00.00[BM01.00]	18	3.3		8bit address with variable offset
	I03a00.0				decentralised 1bit address
	BI03a00.				decentralised 8bit address
OD	BM00.00	6	0.6	OR 16bit	even 8bit address (00.00+00.01)
	BM00.01	10	2.6		odd 8bit address (00.01+00.02)
	AO00.00				16bit address
	1000	4	0.3		constant
	BI03a00.				decentralised 8bit address (00.+01

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

4.6.1.4. Exclusive-OR commands

Com-mand	Operand (example)	Byte	Time [µs]	Function	
XO	M00.00	4	0.45	Exclusive OR	1bit address
	BM00.00	4	0.45		8bit address
	100	4	0.3		8bit constant
	M00.00[10]	10	0.9		1bit address with constant offse
	M00.00[BM01.00]	16	3.2		1bit address with variable offse
	BM00.00[10]	10	0.9		8bit address with constant offse
	BM00.00[BM01.00]	16	3.2		8bit address with variable offse
	I03a00.0				decentralised 1bit address
	BI03a00.				decentralised 8bit address
XON	M00.00	8	0.75	Exclusive OR Negation	1bit address
	BM00.00	8	0.75		8bit address
	M00.00[10]	12	1.1		1bit address with constant offse
	M00.00[BM01.00]	18	3.3		1bit address with variable offse
	BM00.00[10]	12	1.1		8bit address with constant offse
	BM00.00[BM01.00]	18	3.3		8bit address with variable offse
	I03a00.0				decentralised 1bit address
	BI03a00.				decentralised 8bit address

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

no alteration defined flag altera
flag alteration

Summary of commands

4.6.1.5. Assignments and set commands

Com-mand	Operand (example)	Byte	Time [μs]	Function	
=	M00.00	4	0.45	Assign-ment	1bit address
	BM00.00	4	0.45		8bit address
	M00.00[10]	8	0.75		1bit address with constant offset
	M00.00[BM01.00]	14	3		1bit address with variable offset
	BM00.00[10]	8	0.75		8bit address with constant offset
	BM00.00[BM01.00]	14	3		8bit address with variable offset
	O03a00.0				decentralised 1bit address
	BO03a00.				decentralised 8bit address
=N	M00.00	8	1.4	Assign-ment Nega-tion	1bit address
	BM00.00	8	1.4		8bit address
	M00.00[10]	12	1.8		1bit address with constant offset
	M00.00[BM01.00]	18	4.9		1bit address with variable offset
	BM00.00[10]	12	1.1		8bit address with constant offset
	BM00.00[BM01.00]	18	3.3		8bit address with variable offset
	IO3a00.O				decentralised 1bit address
	BO03a00.				decentralised 8bit address
=D	BM00.00	4	0.45	Assign-ment 16bit	even 8bit address (00.00+00.01)
	BM00.01	18	3.6		odd 8bit address (00.01+00.02)
	AO00.00	16	3		16bit address
S	M00.00	12	1.2	conditional set 1bit address	
R	M00.00	10	0.9	conditional reset 1bit address	
=1	M00.00	16	1.65	unconditional set 1bit address	
=0	M00.00	16	1.5	unconditional reset 1bit address	

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

4.6.2. Arithmetic commands

Com- mand	Operand (example)	Byte	Time [μs]	Function	
ADD	BM00.00	4	0.45	Addition 8bit	8bit address
	BI03a00.00				external 8bit address
	100	4	0.3		8bit constant
ADDD	BM00.00	4	0.45	Addition 16bit	even 8bit address (00.00+00.01)
	BM00.01	10	3		odd 8bit address (00.01+00.02)
	AO00.00				16bit address
	1000	4	0.3		16bit constant
SUB	BM00.00	4	0.45	Subtraction 8bit	8bit address
	BI03a00.00				external 8bit address
	100	4	0.3		8bit constant
SUBD	BM00.00	4	0.45	Subtraction 16bit	even 8bit address (00.00+00.01)
	BM00.01	10	3		odd 8bit address (00.01+00.02)
	AO00.00				16bit address
	1000	4	0.3		16bit constant
MUL	BM00.00	8	3	Multipli- cation 8bit	8bit address
	BI03a00.00				external 8bit address
	100	10	3		8bit constant
MULD	BM00.00	8	3	Multipli- cation 16bit	even 8bit address (00.00+00.01)
	BM00.01	12	3.5		odd 8bit address (00.01+00.02)
	AO00.00				16bit address
	1000	8	3		16bit constant
DIV	BM00.00	8	4	Division 8bit	8bit address
	BI03a00.00				external 8bit address
	BM00.00	8	3.8		8bit constant
	BM00.00	8	4	Division	even 8bit address (00.00+00.01)
	BM00.01	12	3.5		odd 8bit address (00.01+00.02)

Summary of commands

4.6.3. Comparison commands

Com- mand	Operand (example)	Byte	Time [μs]		Function
CMP	BM00.00	4	0.4	Compare	8bit address
	100	4	0.3	8bit	8bit constant
CMPD	BM00.00	6	0.6	Compare 16bit	even 8bit address (00.00+00.01)
	BM00.01	10	2.6		odd 8bit address (00.01+00.02)
	AO00.00				16bit address
	10000	4	0.45		16bit constant
CMP=	BM00.00	16	1.3	Compare 8bit	8bit address
	100	16	1.1	if "="	8bit constant
CMPD=	BM00.00	18	1.4	Compare 16bit if "="	even 8bit address (00.00+00.01)
	BM00.01	22	3.3		odd 8bit address (00.01+00.02)
	AO00.00				16bit address
	10000	16	1.3		16bit constant
CMP<>	BM00.00	16	1.3	Compare 8bit	8bit address
	100	16	1.1	if "<"	8bit constant
CMPD<>	BM00.00	18	1.4	Compare 16bit if "<"	even 8bit address (00.00+00.01)
	BM00.01	22	3.3		odd 8bit address (00.01+00.02)
	AO00.00				16bit address
	10000	16	1.3		16bit constant
CMP<=	BM00.00	16	1.3	Compare 8bit if "<" or "="	8bit address
	100	16	1.1		8bit constant
CMPD<=	BM00.00	18	1.4	Compare 16bit if "<" or "="	even 8bit address (00.00+00.01)
	BM00.01	22	3.3		odd 8bit address (00.01+00.02)
	AO00.00				16bit address
	10000	16	1.3		16bit constant
CMP>=	BM00.00	16	1.3	Compare 8bit if ">" or "="	8bit address
	100	16	1.1		8bit constant

4.6.4. Shift and rotation commands

Com- mand	Operand (example)	Byte	Time [μs]	Function	
LSL	Accu	2	0.3	Logical shift left	in accu
LSLM	BM00.00	10	1.1	8bit	in 8bit address
LSR	Accu	6	0.45	Logical shift right	in accu
LSRM	BM00.00	10	1.1	8bit	in 8bit address
LSLD	Accu	2	0.15	Logical shift left 16bit	in accu
LSLDM	BM00.00	10	1.1		in even 8bit address (00.00+00.
	BM00.01	14	5.4		in odd 8bit address (00.01+00.
	AO00.00				in 16bit address
LSRD	Accu	2	0.15	Logical shift right 16bit	in accu
LSRDM	BM00.00	10	1.1		in even 8bit address (00.00+00.
	BM00.01	14	5.4		in odd 8bit address (00.01+00.
	AO00.00				in 16bit address
ROL	Accu	2	0.15	Roll left	in accu
ROLM	BM00.00	10	1.1	8bit	in 8bit address
ROR	Accu	10	0.75	Roll right 8bit	in accu
RORM	BM00.00				in 8bit address
ROLD	Accu	2	0.15	Roll left 16bit	in accu
ROLDM	BM00.00	10	1.1		in even 8bit address (00.00+00.
	BM00.01	14	4.5		in odd 8bit address (00.01+00.
	AO00.00				in 16bit address
RORD	Accu	20	2.1	Roll right 16bit	in accu
RORDM	BM00.00	26	2.1		in even 8bit address (00.00+00.
	BM00.01	34	5.1		in odd 8bit address (00.01+00.
	AO00.00				in 16bit address

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

Summary of commands

4.6.5. Byte and flag manipulation

Com- mand	Operand (example)	Byte	Time [μs]	Funktion	
INC	BM00.00	4	0.45	Increment 8bit value by 1	8bit address
INCD	BM00.00	10	1.1	Increment 16bit value by 1	even 8bit address (00.00+00.
	BM00.01	14	5.4		odd 8bit address (00.01+00.0
DEC	BM00.00	4	0.45	Decrement 8bit value by 1	8bit address
DECD	BM00.00	10	1.1	Decrement 16bit value by 1	even 8bit address (00.00+00.
	BM00.01	14	5.4		odd 8bit address (00.01+00.0
CLR	BM00.00	4	0.45	Clear	8bit address
NOP		2	0.15	Dummy operation	
SEC		2	0.15	Set Carry bit = 1	
CLC		2	0.15	Clear Carry bit = 0	

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

4.6.6. Module calls

Com- mand	Operand (example)	Byte	Time [μs]	Function
JPP	Program module	14	10	jump to program module
JPCP	Program module	18	10.5*2)	conditional jump to program module if ye
JPF	Function module	18	*1)	jump to function module
JPCF	Function module	26	*1)*2)	conditional jump to function module if ye
JPK	KUBES module	18	*1)	jump to KUBES module
JPCK	KUBES module	26	*1)*2)	conditional jump to KUBES module if ye
JPINIT	Init. module	14	10	jump to initialisation module

*1) Normal: 17 μs, plus 1 μs per additional parameter *2) 0.6 μs if no jump

*) Influence of (C)arry and (Z)ero bit: -- yes no alteration defined flag alterationonn unde
++ alteration

4.6.7. Jump commands

Com- mand	Operand (example)	Byte	Time [μs]	Function
JP	Jump label	4	0.3	unconditional jump
JPC	Jump label	8	0.6	conditional jump if yes (log. 1)
JPCN	Jump label	8	0.6	conditional jump if no (log. 0)
JP=	Jump label	4	0.3	jump if equal
JP<>	Jump label	4	0.3	jump if unequal
JP<	Jump label	4	0.3	jump if smaller
JP>	Jump label	4	0.3	jump if greater
JP<=	Jump label	4	0.3	jump if smaller or equal
JP>=	Jump label	4	0.3	jump if greater or equal
JPCS	Jump label	4	0.45	jump if Carry bit = 1
JPC	Jump label	4	0.45	jump if Carry bit = 0
JPZS	Jump label	4	0.45	jump if Zero bit = 1
JPZC	Jump label	4	0.45	jump if Zero bit = 0
JP+	Jump label	4	0.45	jump if positive (two's complement)
JP-	Jump label	4	0.45	jump if negative (two's complement)

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

4.6.8. Copy and BCD commands

Com- mand	Operand (example)	Byte	Time [μs]	Function
C1T8	I00.00	16	12	copy 1bit addresses into 8bit accu
C8T1	O00.00	16	10.6	copy 8bit accu into 1bit addresses
C1T16	I00.00	16	21.5	copy 1bit addresses into 16bit accu
C16T1	O00.00	16	19.2	copy 16bit accu into 1bit addresses
BINBCD3	Accu	4	5.2	binary-to-BCD conversion (3 decades)
BCDBIN3	Accu	4	5.1	BCD-to-binary conversion (3 decades)

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

Summary of commands

4.6.9. Programmable pulses, timers and counters

Com man	Operand (example)	Byt	Time [μs]	Function
=	PP00.00	16	5	programmable pulse at positive edge
=N	PP00.00	16	5	programmable pulse at negative edge
L A O	PP00.00	4	0.45	logical operation with pulse signal
LN AN ON	PP00.00	6 8 8	0.6 0.75 0.75	logical operation with pulse signal, negated
=	PT00.00:100*10ms:R:R	14	14	progr. time with constant time value (log.1=On)
=	PT00.00:BM00.00*10ms:R:R	20	17.5	progr. time with var. time value (log.1=On) **)
=N	PT00.00:100*10ms:R:R	18	15	progr. time with constant time value (log.0=On)
=N	PT00.00:BM00.00*10ms:R:R	24	18.5	progr. time with variable time value (log0=On)
=TH	PT00.00	14	11.5	time halt (current value is kept)
L A O	PT00.00	4	0.45	logical operation with timer output
LN AN ON	PT00.00	6 8 8	0.6 0.75 0.75	logical operation with timer output, negated
LD	PT00.00	4	0.45	load residual timer value
=	C00.00:100:F:R	14	9.6	progr. counter with const. counter val. (log.1=C
=	C00.00:BM00.00:F:R	20	10	progr. counter with var. counter val. (log.1=On
=N	C00.00:100:F:R	18	10.5	progr. counter with const. counter val. (log.0=C
=N	C00.00:BM00.00:F:R	24	11	progr. counter with var. counter val. (log.0=On
=C	C00.00	14	10	clock pulse transfer (counting)
L A O	C00.00	4	0.45	logical operation with counter output
LN AN ON	C00.00	6 8 8	0.6 0.75 0.75	logical operation with counter output, negated
LD	C00.00	4	0.45	load actual counter value

4.6.10. Special commands

Com- mand	Operand	Byte	Time [μs]	Function
O_OFF	-			no function in PC Control 645-500
O_ON	-			no function in PC Control 645-500
RESET	-			reset all non-remanent outputs, markers, counters and timers and stop program run

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

4.6.11. Commands of the initialisation modules

The initialisation modules are a special variety of modules.
None of the commands described previously in this chapter can
be used here. On the other hand can the following commands
only be used with the initialisation modules.

Operand	Data type	Value	Byte	Time [μs]	Function
M00.00	BIT	1 1,0,1,1,... [16],1			Write logical value into 1bit address Write logical values into 1bit address and Write logical value into 1bit address and
BM00.00	BYTE	75 1,18,0,125... [8],128 "KUHNKE"			Write value (dec.) into 8bit address Write values into 8bit address and follow Write value into 8bit address and 7 follow Write text into 8bit address and following
BM00.00	WORD	19285 1,18,0,125... [8],13283			Write value (dec.) into two 8bit addresses: Write values into two 8bit addresses and Write value into two 8bit addresses and 7
O00.00		19285 1,18,0,125... [8],13283			Write value (dec.) into 16bit address Write values into 16bit address and follow Write value into 16bit address and 7 follow
BM00.00	TEXT	"KUHNKE" "KUHNKE", "MALENTE"...			Write text as from the stated address Write texts as from the stated address
BM00.00	ADR	FBM00.00			Write intermediate code address of opera into two 8bit addresses or into one 16bit ; (application for special KUBES modules
AO00.00					

4.6.12. Commands of the data modules

Com- mand	Operand	Byte	Time [μs]	Function
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 (= 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) intoDBx00.00...15.15(x = value 0...7 in by
StoreDB	x,<name>	12	150	store DBx00.00...15.15 (x = 0...7) in data modul <name>
	byte1,<name>			store DBx00.00...15.15 (x = value 0...7 in byte1) data module <name>
	x,byte2			store DBx00.00...15.15 (x = 0...7) in data module (y = value 1...255 in byte2)
	byte1,byte2			store DBx00.00...15.15 (x = value 0...7 in byte1) data module number y (y = value 1...255 in byte2)

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

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

4.7. Registers

As a matter of size, there are three types of operands in the PC Control 645-500:

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

The accumulator in the CPU of PC Control 645-500 can thus be used as 1bit, 8bit or 16bit register.



Please do not confuse the term "accu(mulator)" in the software part which stands for a general-purpose register in the processor and the term as it is used in the hardware part where 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 triggered by commands that carry a "D" as their last character.



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

4.8. Addressing

The value of the operand can be assigned in two different ways:

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

Address mnemonics

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

For example, "LBM00.00" symbolises the loading of the contents of a memory location which carries the mnemonic name "BM00.00".

4.8.2. Offset addressing

It is possible to indicate an offset together with the absolute address of a local operand. The address is then made up by adding absolute address and offset.

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 corresponding 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 M00.00[5]	is the same as	L M00.05
= O01.00[6]	is the same as	= O01.06

4.8.2.1. External operands used as offset

Offset addressing of external operands (PROFIBUS inputs) is not possible.

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

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

Example: L BR01.00[BIO3a00.]

4.8.3. Types of addressing: summary

The load command is used to give a summary of the different types of addressing.

Load contents of an operand		
L	M00.00	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	M00.00[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	M00.00[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	M00.00[BIO3a00.]	1bit address
L	BM00.00[BIO3a00.]	8bit address
LD	BM00.00[BIO3a00.]	16bit address

4.9. PLC reactions to errors and failures

PC Control 645-500 monitors itself. Any occurring errors or failures are reported and lead to reactions by the PLC, depending on their seriousness.

Errors and failures are numbered through from 1 to max. 255. They can be signalised in various ways:

4.9.1. Failures and errors summary

No.	Failure / Error Type	Type of indication			
		LED if exists	Error byte ERR00.00	PROFIBUS EVENT	m
1					
2	undervoltage	yes	2	5	
3	watchdog		-	2	
4	no communication		4	no	
5	bus failure		5		
6	baud rate		6		
7	-				
8	checksum	yes	8	1	
9	hierarchy		9	4	
10					
11					
12					
13	voltage (2) ok again		13		

Legend for the failures/errors summary

Error byte "ERR00.00"

The number of the error/failure is written into an error byte (ERR00.00) so that it can be analysed in the user program:

Example: L ERR00.00
 C8T1 000.00 ;binary display to 8 outputs

PROFIBUS (Event)

The error/failure inc. its relevant number is passed on to PROFIBUS as event notification. Events are internally sorted by priority. If there are several messages for simultaneous transmission, the event allocated highest priority (= lowest priority number) will be transmitted first.

Interrupt module [no.]

The error/failure triggers an interrupt (IRQ). This causes the monitor program to immediately call up the assigned interrupt module.



In the following sections, various types of failures/errors are described and suggested reactions are explained.

4.9.2. Undervoltage (supply, failure no. 2)

4.9.2.1. Without external supply

PC Control 645-500 is supplied by the PC with 5 V. If the voltage supply falls below 4.7 V, PC Control 645-500 immediately triggers a RESET. All unbuffered data will get lost.

There will be **no error message or failure indication**.

4.9.2.2. With external supply

Supply voltage: 24 V DC -20%/+25%

There is a two-stage reaction by the built-in voltage monitoring function to the falling below certain limiting values:

1st stage

Cause

Supply voltage c. < 19 V

Reaction

- interrupt module no. 17 will be activated
- the program run will not be interrupted as yet



Buffered operands (markers, timers, counters) may be involuntarily reset if the user program is continued to be processed at this stage. This could be caused by decentralised inputs that might detect a 0 signal due to the undervoltage condition.

Indication

- LED "failure" flashes (if exists)
- error byte "ERR00.00" is set to 2

Reactions to errors and failures

2nd stage

1st alternative

Cause

Supply voltage rises back to 24 V DC \pm 20%

Reaction

- "failure" LED will extinguish
- error byte "ERR00.00" will be reset
- the program run will be continued without interruption

2nd alternative

Cause

Supply voltage continues to fall, c. < 8.5 V

Reaction

- 5 V system voltage will be interrupted
- => - STOP: the program run will be stopped
 - RESET: outputs, error byte (ERR00.00) and unbuffered markers, timers and counters will be reset
 - all LEDs off

Remedy

- as a precaution, in the user program, to save buffered operands: Processing of the user program should be interrupted until the 2nd stage of the cause has been reached or until a realistic delay period has elapsed.

program example for interrupt module no. 17:

```
      WAIT      5      ;wait 5 * 10 ms=50 ms
      L         ERR00.00 ;scan error byte
      CMP       2      ;still undervoltage?
      JP<>      RETURN  ; jump if not
      RESET
      RETURN    NOP    ; RESET and Stop PLC
```


4.9.3. Watchdog (program run time exceeded, failure #3)

Cause

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

Indication

- "failure" LED flashes (if exists)
- KUBES outputs failure message in plain text
- event notification via PROFIBUS

Reaction

- STOP: program run will be stopped
- RESET: outputs and unbuffered markers, timers and counters will be reset
- external outputs (PROFIBUS) off

Remedy

- change the program architecture to achieve shorter run times
- restart the controller:
 - by hardware: switch supply off and back on again,
 - or
 - by software: KUBES RESET command followed by the RUN command

Reactions to errors and failures

4.9.4. PROFIBUS error (failures 4, 5, 6)

Cause

communication with the communication partners is interrupted by:

- PROFIBUS cable broken,
- problem lies with communication partner,
- wrong station address set,
- invalid baudrate set ...

Reaction

- the values in the external inputs are no longer current values

Notification

- error byte "ERR00.00" will be set to 4, 5 or 6

Remedy

- find and remove cause of problem

4.9.5. Checksum in user program (failure #8)

During program generation, a checksum (CS) is generated over the entire user program memory according to a certain algorithm.

Cause

When starting the controller, the monitor re-calculates the checksum and compares it to the stored value. If the result is unequal, the monitor recognises a failure.

Reaction

- the controller will not start

Notification

- error byte "ERR00.00" will be set to 8

Remedy

- find and remove the cause of the failure
- use the Transmit command of the KUBES PLC menu to transmit the project again to the controller

4.9.6. Hierarchy error (failure #9)

Program calls and other module calls must not exceed certain hierarchy limits (see chapter "3. Software modules" of programming manual E 417 GB).

During programming, the controller reports a hierarchy error when it receives a program. At this stage, this is only a warning that there could be an error. Only at start-up does the controller check whether there really is a hierarchy error. If there is not, the error indication disappears again.

Cause

When switching the controller on or after the KUBES start command, the monitor program checks whether there is a hierarchy error (a module calls up the module by which itself was called up or the nesting depth exceeds 5 levels)

Reaction

- the controller will not start

Indication

- "failure" LED flashes (if exists)
- error byte "ERR00.00" will be set to 9

Remedy

- find and remove the cause of the error
- use the Transmit command of the KUBES PLC menu to transmit the project again to the controller

5. PC Control 645-500 used as programming card

At this point we're assuming that your PC Control 645-500 has been coded correctly and installed in the PC as well as that it is being supplied with voltage. If this is not the case please first follow the instructions given in chapter "3.8. Installation".

The following controllers can be programmed via PROFIBUS:

- Profi Control 680I
- ModuControl 657P
- PC Control 644
- PC Control 645-500

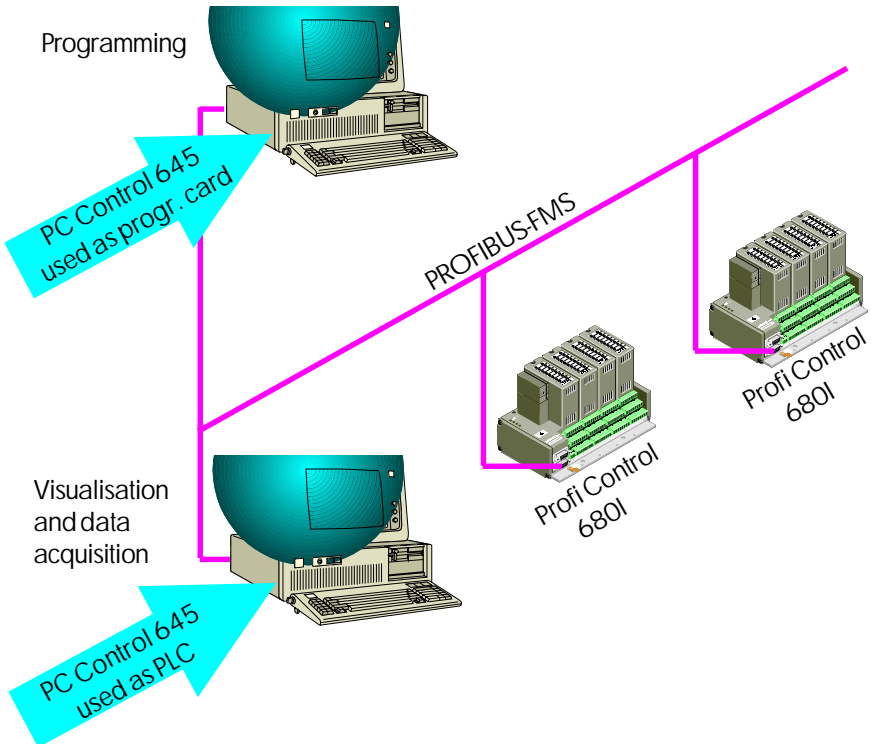


Fig.: PC Control 645-500, installed in the programming PC as programming card, has access - via KUBES - to 3 PROFIBUS-FMS masters connected to a PROFIBUS-FMS network

5.1. What do you need?

- KUBES (version 5.10 or higher)
- PC Control 645-500 (monitor version 6.0 or higher), installed in the PC
- disk "On-line PROFIBUS with 645-500"
part no. 645.506.01
 - KUBES project "P645_IO" with C task "P645_IO.T01"
if you are addressing the card via the I/O range
 - or
 - KUBES project "P645_MEM" with C task "P645_MEM.T01"
if you are addressing the card via the memory range

Function

PC Control 645-500 has a DP-RAM interface to the PC bus. KUBES, via its "Online PC" function and drive KUBES17.EXE, processes the entire KUBES protocol via this interface. Furthermore, PC Control 645-500 has a PROFIBUS interface. A PROFIBUS task in the PLC operating system supports this interface in processing all PLC-PROFIBUS functionality. If PC Control 645-500 is used as programming card, the PLC task will be deactivated and the PROFIBUS task will be used for processing the KUBES protocol. Switching over to this task is done by running C task P645_IO.T01 (in the case of I/O addressing) or P645_MEM.T01 (in the case of memory addressing) in PC Control 645-500. This C task also ensures that a programming connection to the selected Kuhnke PLC is created via PROFIBUS making use of the KUBES15.EXE driver's functionality. This driver is called up by choosing KUBES' Online Profibus command.



Remote programming via PROFIBUS is also possible in Kuhnke controllers that are used in single projects but that are equipped with a PROFIBUS interface.

5.2. Preparing the programming card

Before you can use PC Control 645-500 to do some programming via PROFIBUS you will have to use KUBES to prepare the device for it.

5.2.1. PC card settings

- "PLC" menu
- option "Settings" - PC cards"



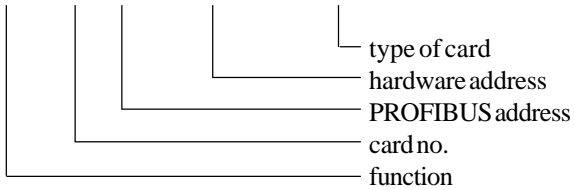
- type of card: 645-500
Choosing "none" and pressing "accept" would cause any existing card settings in PROFISO.INI to be cleared.
- card no.: 1...8
you can use up to 8 cards in any one PC
- PROFIBUS addr. 0...125
- hardware address 0120...E120 (I/O range)
D0000...DE000 (memory range)
this setting has to be the same as the one of the coding switches (see ch. "3.4.1. Addressing the (ISA-)PC interface").
- accept:
Click here to accept the settings chosen for your card. This will cause an entry in PROFISO.INI (system file stored in the Windows root directory) to be made.



overleaf

Entries in PROFISOF.INI

AT_SPS_1=1,0x0120,645

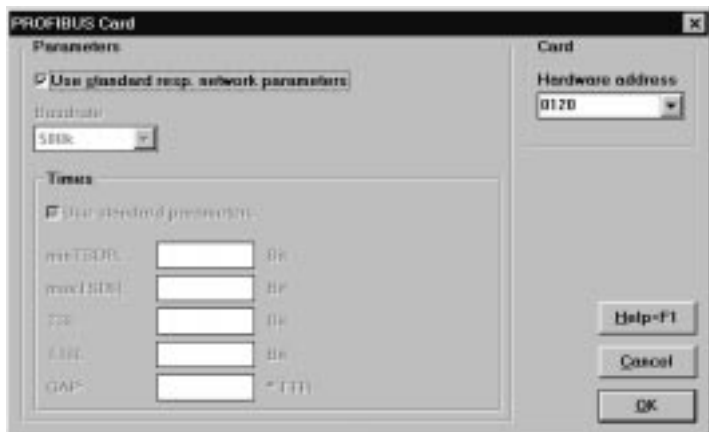


This is a complete entry for use as PLC. For use as programming card you need to input the type of card as the driver software for the various PC cards (645-500, 644...) reacts differently.

5.2.2. Setting as PROFIBUS card

You can set only one of the PC cards as PROFIBUS card (for programming).

- "PLC" menu
- option "Settings - PROFIBUS card"



next page

- use standard or network parameters
if you choose this option KUBES will automatically set all parameters:
 - if no network has been opened: standard PROFIBUS parameters
see table below "Standard FMS bus parameters"
 - if a network has been opened: PROFIBUS parameters set in VEBES
- baudrate
can only be set if the "standard or network parameters" have not been used
- times
can only be changed if the "standard or network parameters" have not been used
 - use standard parameters
sets the times corresponding to the set baudrate

Recommended action

Set the same parameters that are also being used in the network.

Standard FMS bus parameters

Parameter	Baudrate [kbit/s]				
	9,6	19,2	93,75	187,5	
min_TSDR	30 Bit	60 Bit	125 Bit	250 Bit	
max_TSDR	50 Bit	100 Bit	250 Bit	500 Bit	
TSL	120 Bit	200 Bit	500 Bit	1500 Bit	
GAP	2	2	2	2	
TTR	10000 Bit	15000 Bit	30000 Bit	50000 Bit	

5.2.3. Loading a C task into PC Control 645-500

The Software required to turn a PC Control 645-500 into a programming card is transferred to the card's memory as C task by means of the appropriate KUBES functions.

On the disk (part no. 645.506.01) you will find a directory called "KUBES" containing 2 projects. They contain the required C task and the necessary memory settings:

- project "P645_IO" for all cards to be addressed via the I/O range of the PC
- project "P645_MEM" for all cards to be addressed via the memory range of the PC

What do you have to do?

The following steps are to be carried out as you are used to with all "normal" KUBES projects:

- Copy project "P645_IO" or "P645_MEM" from the floppy disk onto the hard disk
- Open the project and Transmit to PC Control 645-500
- Start PC Control 645-500

To do a **function check** you can now call up the Display Address Range and (dynamically) monitor byte marker BM06.00. The value permanently alternates between "3" and other values.

- Close the project.

PC Control is now set up to work as a programming card as long as the program remains unchanged.

5.3. On-line PROFIBUS

Prerequisites

- A PC Control 645-500, set up as a programming card as described in chapter 5.2, has been installed in the PC
- KUBES, version 5.10 or higher, is running
- Via PROFIBUS-FMS, the programming card is connected to:
 - a network consisting of Kuhnke controllers
 - a single Kuhnke controller equipped with a PROFIBUS-FMS connector

Proceed as follows:

- Open the project for the controller that KUBES is to communicate with
- Start on-line communication:

- "PLC" menu
- option "Online - PROFIBUS"

The following warning message will be displayed:



- Cancel the process by choosing "no"
- Continue preparing the on-line connection by choosing "yes"



overleaf

The following dialog will be displayed:

- Master
valid option for all standard Kuhnke controllers
- Connection - from PROFIBUS card
PROFIBUS address of the programming card. If, in VEBES, a "programming PC" was entered as station for the network, KUBES will automatically take over the PROFIBUS address of this PC at this point.

Please note the following limitation for PROFIBUS addresses of the programming card:

- it must be no larger than the "Highest Station Address" of the network (see VEBES, bus parameters, HSA)
- it must be different from the PROFIBUS addresses of all network stations

- Connection - to PLC
PROFIBUS address of the controller to which the on-line connection is to be established
- extended check of PB card address
If this box is checked, KUBES will check whether the PROFIBUS address set for the programming card (see "Connection - from PROFIBUS card") is the same as the entry for the PC card (see "5.2.1. PC card settings, PROFIBUS address"). If this is not the case, the program will output an error message after "OK".
- OK
Create on-line connection.



6. Communication with PC programs

If used as PC card, PC Control 645-500 can be addressed directly via the PC bus.

The dual-port RAM of PC Control 645-500 is a physical memory of a size of 1 kbyte or 2 kbyte. At the side of an IBM-AT compatible computer it is addressed via the I/O or the memory range (see ch. "3.4.1. Addressing...").

The dual-port RAM is sub-divided into 2 channels with permanent address ranges:

- KUBES protocol range (is processed by the monitor program in the PLC)
- process image range (is created by KUBES module PA_MOVE in the PLC program)

Both channels can be used for the implementation of PC applications. Thus PC Control 645-500 **is to be preferably used with the following control concept:**

- Man-Machine-Interface, i.e. visualisation and operation via PC
- PLC program runs on the PC but independently of the PC program
- decentralised processes, interconnected via PROFIBUS
- optional extension of functions via PC (CNC module, input of measuring values ...)
- optional extension and networking via PROFIBUS



Due to the multitude and heterogeneity of the visualisation programs available on the market (both DOS and WINDOWS based software), Kuhnke support users by supplying drivers, demo and test programs, C sources and documentation as applications or the case may require.

6.1. PC interface driver

Some of the drivers described below are exclusively designed for use with PC cards addressed via the PC I/O range.

6.1.1. DOS program with process image interface

Program name: PAMOVDEM.EXE

6.1.1.1. Components

- Driver PAMOVDEM.EXE
- and sources DPRAM.C, DPRAM.H, PAMOVDEM.C, PAMOVDEM.H
- test program PAMOVTST.EXE, making use of driver PAMOVDEM.EXE.
- KUBES module PA_MOVE
- documentation

6.1.1.2. Purpose of the program

The program is an example for accessing the dual-port RAM via an application running on the PC from within the C programming language. Applicable compilers for the source code are the Visual C/ C++ Compiler by Microsoft and the C Compiler by Borland (use file "DPRAM.H" to define constant "MICROSOFT" for the Microsoft and constant "BORLAND" for the Borland compiler resp.). If at all possible, the program works only with standard ANSI-C calls. File "DPRAM.C" contains all routines for accessing the dual-port RAM. The constant and function prototype definitions used for this functionality are stored in file "DPRAM.H". The only functions that may not correspond to the ANSI standard are the calls for port accesses(inp, outp). These may have to be adjusted (by defining the constants "BORLAND" or "MICROSOFT", see above).

The main program is contained in "PAMOVDEM.C" with the definition of constants and function prototypes again being stored in "PAMOVDEM.H".

The program "PAMOVST.EXE" is an example for the application of "PAMOVDEM". It serves monitoring the input and output boxes for the process image values as well as monitoring the synchronisation and status bytes. Using this program you can read and write specific values.

What you need on the PLC side is KUBES module PA_MOVE. It ensure synchronised, dual-ported connection between marker ranges SLG/SLH and dual-port RAM her.

6.1.1.3. Program functions

The program PAMOVDEM.EXE is a simple DOS application which can, however, also run in a DOS box in WINDOWS.

Use the following call-up command to start it:

pamovdem.exe pppp or **pamovtst.exe pppp**

where: **pppp** = hexadecimal port starting address

The program generates a screen mask where first of all the version and the basic port address used are shown. Then it reads the values in the output box, displays them as hexadecimal and ASCII values and copies them to the input box. Successful writing operations in the input box are confirmed by the message "values written into input box". Any errors occurring are displayed.

As compared to "PAMOVDEM", "PAMOVST" has additional data editing functions.

Pressing "ESC" quits the program.

6.1.1.4. Practical examples

Customised user interfaces for machines and systems with fast direct data access.

6.1.2. VisualBasic program with process image interface

6.1.2.1. Components

- PAMOVDEM.BAS: VISUALBasic® source code
PAMOVBAS.TXT: the same for ASCII.
- PAMOVDEM.FRM: VISUALBASIC® dialog sources
PAMOVFRM.TXT: the same for ASCII.
- PAMOVDEM.MAK: VISUAL-BASIC® project file for
PAMOVDEM.BAS
- PAMOVDLL.DLL: DLL with accessing functions to the DP-
RAM.
- PAMOVDLL.LIB: library for VISUAL-C compiler
- PAMOVDEM.EXE: demo program using driver
PAMOVDEM.DLL (with VisualBasic compiler created from
PAMOVDEM.BAS).
- KUBES module PA_MOVE

6.1.2.2. Purpose of the program

as described in chapter 6.1.1.2 but for VISUAL-BASIC®.

6.1.2.3. Program functions

PAMOVDEM.EXE is a WINDOWS application program. Prerequisite for running it is the file PAMOVDEM.DLL stored in the same directory as well as VBRUN300.DLL in the \WINDOWS\SYSTEM directory.

Start it like any other WINDOWS program, by double-clicking on PAMODEM.EXE in File Manager.

6.1.2.4. Practical example

Customised user interfaces for machines and systems with fast direct data access.

6.1.3. WINDOWS application with protocol interface

6.1.3.1. Components

- driver KUBES17.EXE
- test program TEST.EXE making use of driver KUBES17.EXE
- sources TEST.C, TEST.RC, DEFINES.H, MESSAGE.H
- documentation, protocol description

6.1.3.2. Purpose of the program

TEST.EXE, a program running under WINDOWS, is an example for how to use KUBES' V.24 driver KUBES5.EXE as well as dual-port RAM driver KUBES17.EXE.

These drivers are WINDOWS applications of their own (interface servers) that are controlled by the program using them (interface clients, usually "KUBES1.EXE") by specific Kuhnke WINDOWS messages and that, vice versa, use the same method to transmit data such as error messages back to the client program.

The following are available for adaptation to the desired application: test.c, test.rc, defines.h, messages.h

6.1.3.3. Program functions

Use TEST.EXE to monitor PLC variables BM00.00/.01 and M00.00...M00.15 of PC Control 645-500.

The following are required to operate the test program: TEST.EXE, KUBES17.EXE and/or KUBES5.EXE stored in the same directory that was entered as PATH in the [KUBES] section of PROFISOF.INI.

Start the program from this directory by the following command:

```
\windows\win test
```

6.1.3.4. Practical examples

Visualisation software running under WINDOWS such as InTouch®.

6.1.4. DOS program with protocol interface, VisiPro

This program is available for I/O range addressing as well as for memory range addressing (see ch. "3.4.1. Addressing..."). It consists of several files.

The filenames are differentiated by replacing variable "xxx" by

- IO for I/O range addressing
- MEM for memory range addressing

Example for file driver "VISI_xxx.EXE":

- VISI_IO.EXE for I/O range addressing
- VISI_MEM.EXE for memory range addressing

6.1.4.1. Components

- driver VISI_xxx.EXE
- test program xxx_TEST.EXE making use of driver VISI_xxx.EXE.
- documentation

6.1.4.2. Purpose of the program

VisiPro® is a software package for process visualisation supplied by INOSOFT. Use the extensive functionality of this package to edit and display your data. VisiPro® relies on universal drivers that maintain data exchange with other PC applications.

VISI_xxx.EXE is the link between VisiPro's® universal driver and PLC PC Control 645-500.

6.1.4.3. Program functions

VISI_xxx.EXE runs as TSR program, i.e. you only load it once and it then occupies c. 18 kbyte of memory.

Always start VISI_xxx.EXE before starting VisiPro®. A batch routine would be a practical suggestion.

Use the following syntax for **installation**:

visi_xxx [(PLC no.)] [(driver no.)]

Successful and unsuccessful installation are reported by corresponding messages.

Repeat the above command to unload the program.

After launchint VisiPro, this program's universal driver will, via the KUBES protocol interface, access the addresses of PC Control 645-500 defined in the visualisation software.

6.1.4.4. Practical example

Visualisation software running under DOS, e.g. VisiPro®.

6.1.5.DOS program with protocol interface, TurboPascal

This program is available for I/O range addressing as well as for memory range addressing (see ch. "3.4.1. Addressing..."). It consists of several files.

The filenames are differentiated by replacing variable "xxx" by

- IO for I/O range addressing
- MEM for memory range addressing

Example for file driver "VISI_xxx.EXE":

- VISI_IO.EXE for I/O range addressing
- VISI_MEM.EXE for memory range addressing

6.1.5.1. Components

- driver VISI_xxx.EXE
- Pascal program VISIDEMO.PAS making use of driver VISI_xxx.EXE.
- ditto generated as VISI.UNI, UNITDEMO.PAS

6.1.5.2. Purpose of the program

The purpose of the program is to suggest solutions for data exchange with PC Control 645-500 to users creating their PC programs using TurboPascal.

6.1.5.3. Program functions

VISI_xxx.EXE is run as TSR program, i.e. you load it once and then it occupies c. 18 kbyte of memory.

VISIDEMO.PAS contains call-up routines for VISI_xxx.EXE.

6.1.5.4. Practical example

TurboPascal program written for data exchange with PC Control 645-500.

6.1.6. DOS program with protocol interface - programming language link

This program is available for I/O range addressing as well as for memory range addressing (see ch. "3.4.1. Addressing..."). It consists of several files.

The filenames are differentiated by replacing variable "xxx" by

- IO for I/O range addressing
- MEM for memory range addressing

Example for file C source program "xxx.EXE":

- VISI_IO.EXE for I/O range addressing
- VISI_MEM.EXE for memory range addressing

6.1.6.1. Program components

xxx.C C source program

e.g. CLIPxxx.LIB: C library program for embedding in CLIPPER.

6.1.6.2. Purpose of the program

The purpose of the program is to support users relying on one of the many widely spread programming languages therefor having to make use of the default software interfaces of these programming languages (for visualisation) in developing data exchange procedures with PC Control 645-500.

6.1.6.3. Program functions

The whole program is based on a software module written in the "C" language.

Generally speaking, there are only three exported functions between this module and the application software. These have been implemented as follows:

Test whether a PLC can be accessed in the PC

BOOL CHECKxxx (int NoOfPLC)

This routine requires an "int" type parameter indicating the number of the PLC being searched for. The value of the parameter is 1.

A logical value will be returned: 1 if the search was successful, 0 if it was not.

Get data from PLC

unsigned FROMxxx (char * address, unsigned quantity, unsigned char *pdestination)

The following three parameters are required:

- char * address:
 - long pointer to a string containing the PLC address as from which the data to be got are stored
- unsigned int quantity:
 - number of byte to be read
- unsigned char *pdestination:
 - long pointer to the memory defined for storing the downloaded data (reserved size >= quantity!)

Write data to PLC

unsigned TOxxx (char * address, unsigned quantity, unsigned char *psource)

The following three parameters are required:

- char * address:
long pointer to a string containing the PLC address as from which the data are to be written
- unsigned int quantity:
number of byte to be written
- unsigned char *psource:
long pointer to the memory containing the data to be written

Return values for getting/writing data

An "int" value with the following definitions is returned for the reading/writing functions:

- 0, // okay
- 1, // wrong PLC or no response
- 2, // address too large
- 3, // address string not found
- 4, // no synchronisation with PLCS
- 5, // parameter error for call-up = PARAM_ERROR


6.1.6.4. Practical example

At present, the following program connections are supported by the described software interface:

- C programs (in general)
- VisiPro (visualisation)
- Clipper (dBase compiler)
- (Turbo)Pascal (in general)
- BASIC (in general)

A. Specifications

A.1. Technical specifications

Type	plug-type PC module		
Weight	158 g		
Application	PLC in the PC use in other casing possible if supplied externally		
Networking via PROFIBUS			
protocols	- PROFIBUS-FMS - PROFIBUS-DP - SINECL2-DP (preliminary DP standard)		
baudrates	9.6/19.2/93.75/187.5/500 kbit/s		
Microprocessor	80C167		
Memory	- flash EPROM:	512 kbyte	
	- RAM:	256 kbyte	
Accumulator	NiCd 3.6 V		
buffer time:	~ 36 days (0...40 °C)		
recharge time	≤ 72 h		
	<i>Due to different warehouse times the charging state of the accumulator is undefined when delivered</i>		
Admissible ambient conditions			
storage temperature	-25...+70 °C		
ambien temp. during operation	0...55 °C		
relative humidity	50...95%		

Appendix

Power supply

- PC Control 645-500 (645.425.01) 5 V DC/0.4 A from the PC
- PC Control 645-500 NT (645.425.02) either internal
 - 12 V DC/0.25 A from the PC
 - or external
 - 24 V DC -20%+25%/0.15 A



Uninterrupted change-over from internal and external supply and vice versa!

Interfaces

- 1 PROFIBUS interface (RS 485) up to 500 kbit/s
- 1 V.24 (programming) interface up to 57.6 kbit/s
- 1 ISA interface to the PC bus via dual-port RAM, 1 or 2 KByte

Programming

- programming device IBM-PC (or compatible)
- operating system Windows 3.n, Windows 95
- programming software - KUBES (user program)
 - VEBES (PROFIBUS network)
- programming interface - ISA bus
 - V.24
 - PROFIBUS

A.2. Order specifications

PC Control 645-500
without external supply 645.425.01

PC Control 645-500 NT
with external supply 24 V DC 645.425.02

A.3. References to literature

Instruction manual E 315 D

VEBES

Network operating software for PROFIBUS

Kuhnke GmbH, Malente

Instruction manual E 327 D

KUBES

Kuhnke user software for programmable logic controllers

Kuhnke GmbH, Malente

Instruction manual E 365 D

PROFIBUS

Kuhnke GmbH, Malente

Programming manual E 417 D

Kuhnke GmbH, Malente

EN 50 170 Volume 2, PROFIBUS

PROFIBUS standard

PROFIBUS

The fieldbus for industrial automation

Klaus Bender (Ed.)

Carl Hanser Verlag and Prentice Hall

ISBN 13-012691-8 (hbk)

Appendix

Index

A

- accumulator 3-9
 - function test 4-16
- addr. 4-4
- address
 - mnemonics 4-38
- address conflicts 3-11
- addressing 3-3, 4-38
 - bit 4-21
 - byte 4-21
 - word 4-21
- ambient conditions A-1
- application A-1
- arithmetic commands 4-29
- assignments 4-28

B

- bank 4-11
- baudrates A-1
- BCD commands 4-33
- block transfer 4-17
- bus parameters
 - standard 5-5
- bus protocol 1-6
- byte manipulation 4-32

C

- C task
 - P_645_IO 5-6
 - P645_MEM 5-6
- cable routing and wiring 2-7
- card no. 4-8
- checksum 4-47
- coding switch 3-1, 3-3

commands

- summary 4-23
- communication
 - open 1-5
- communication path 4-5
- comparison commands 4-30
- configuration 4-4
- copy commands 4-33
- counters 4-15, 4-34

D

- danger 2-2
- data memory 4-12
- data modules
 - commands 4-36
- decentralisation 1-2
- design 3-1
- device type 4-4
- DIP switch 3-1
- DP-RAM markers 4-15
- drivers 6-2
- dual-port RAM 3-9

E

- electromagnetic compatibility 2-5
- electrostatic discharge 2-5
- EMC 2-5
- emergency stop 2-3
- EN 50 170 A-3
- error byte "ERR00.00" 4-42
- ESD 2-5

Index

F

failures and errors summary 4-41
flag manipulation 4-32
flash EPROM 3-9, 4-11

H

hardware 3-1
hardware address 4-8
hierarchy error 4-48
 during program generation 4-48
high contact voltage
 danger caused by 2-2

I

I/O range 3-3, 3-4, 3-11
I/O wait states 3-5
individual controller operation 1-2
information / cross reference 2-2
initialisation modules
 commands 4-35
inputs 4-15
installation 3-10
 to be observed 2-3
interface
 PROFIBUS 3-1
 to PC bus 3-1
 V.24 3-1
interfaces 3-7, A-2
interference emission 2-6
 particular sources of interference 2-8
interrupt 3-6
ISA 3-1

J

jump commands 4-33

K

KUAX 644 1-7
KUBES 4-2, 4-5

L

Line address 4-20
literature A-3
logical operations commands 4-23

M

maintenance
 to be observed 2-4
markers 4-15
memory 3-9, A-1
 setting up 4-11
memory range 3-3, 3-4, 3-12
microprocessor A-1
models
 PC Control 645-500 1-1
module calls 4-32
monitor program 4-13

N

network
 create 4-3
 stations 4-4
Ni-Cd accumulator 3-9

O

offset 4-39
on-line
 PC 4-6, 4-8
 PROFIBUS 4-6, 4-10
 V.24 4-6, 4-7

- operands
 - 16bit 4-37
 - 1bit 4-37
 - 8bit 4-37
 - external 4-17
 - local 4-14
- operating system A-2
- order specifications A-3
- outputs 4-15

P

- PAMOVDEM.EXE 6-2
- parity check 4-7
- PEaxx.yy 4-22
- PFxx.yy 4-22
- PLC 4-1
 - working method 4-13
- power breakdown 3-2
- power supply 3-2, A-2
 - external 3-1
- process image 4-17
 - controller 4-20
 - Profi Control 680S 4-18
- PROFIBUS
 - bus termination 3-8
 - cable connector 3-8
 - features summary 1-5
 - integration of the PLC 4-2
 - interface 3-8
 - messages 4-22
 - parameters 4-3
- PROFIBUS addr. 4-8
- PROFIBUS card 5-4
- PROFIBUS error 4-46
- PROFIBUS standard A-3
- PROFISOF.INI 4-6
- programming card 5-1
- programming device A-2
- programming interface A-2
- programming software A-2

- project planning
 - to be observed 2-3
- protocols A-1
- PSaxx.yy 4-22
- pulses 4-34
- purpose
 - PC Control 645-500 1-1
- push-button
 - for resetting the card 3-1

R

- RAM 3-9, 4-11
- reactions to errors and failures 3-2
- reactions to failures and errors 4-41
- registers 4-37
- reliability 2-1
- remote programming 1-4
 - KUAX 644 1-4
 - Modu Control 657P 1-4
 - PC Control 645-500 1-4
 - Profi Control 680I 1-4
- reset 3-1
- reset trigger 3-5
- resistance to interference 2-5
- rotation commands 4-31

S

- safety 2-1
- segment addressing 3-12
- servicing
 - to be observed 2-4
- set commands 4-28
- shift commands 4-31
- special commands 4-35
- station / project 4-4
- station address 1-5
- SW4 3-3
- system error marker 4-15

Index

T

- target group 2-1
- technical specifications A-1
- timers 4-15, 4-34
- topology 1-5
- transfer rate 4-7
- type of card 4-8
- types of addressing 4-40

U

- undervoltage 4-43
- undervoltage monitoring 3-2
- user components 3-1
- user program 4-13

V

- V.24 interface 3-7
- validity
 - of external operands 4-22
- VEBES 1-6, 4-2, 4-17
- VisiPro 6-7
- VisualBasic 6-4

W

- watchdog 4-45
- weight A-1