Manual

SERCOS

Order no.:
VIPA SER-HB86E
Rev. 00/14

General Information on SERCOS

Software

The SERC-Master software is delivered complete with the required development environment preinstalled on an 80MV hard disk. The package also includes a PCMCIA-card containing an additional copy of the SERC-Master software and the respective libraries. (PCMCIA-cards must be ordered explicitly). We have also included a diskette with handling blocks for the PLC. At present these handler blocks are available for the PLC-115U (CPU941-CPU944B) and for the PLC-135/155(CPU928). (CPU945/CPU948 in preparation.)

Development environment

Directory C:\SERCOS contains all the modules that are required for the creation of a SERC-Master. The Borland-C V3.1 compiler is used for this purpose. You can use the TD386 debugger to test the output. For the use of the TD386 debugger it is necessary that the driver "thd386.sys" is installed in the "config.sys" file. The C:\SERCOS directory also contains two other directories that are called SOURCE and MASTER.

As delivered, the SERC-Master software is started automatically (see "autoexec.bat", "setup.bat" located in the root directory, C:\). When you wish to start a compilation you must quit from the SERC-Master software and change directory to the C:\SERCOS\SOURCE directory. Start Borland-C ("bc.exe") and the project file that is located in the same directory will be loaded automatically. At this point you can start the compilation or you can write your application software to create a special SERC-Master. The result is an executable file named "sercmst.exe" which is located in the directory C:\SERCOS\MASTER. This presents a windowed user interface when it is started.

Contents of the SOURCE directory:

sercmst.lib	Sercos-Master library (only used for linking purposes)
sercovl.lib	Sercos-Master overlay library (only used for linking purposes)
motctrl.c	C-language source module with entry points for your application
motctrl.bsp	C-language source containing an sample program explaining the use of MC
motctrl.h	Include-file containing the prototypes of the functions from motctrl.c
sercinit.h	Include-file containing all structure definitions for the SERCOS-Ring
saa.h	general include file
sercmst.prj	predefined Borland-C project file for the BC-IDE

Contents of the MASTER directory:

sercmst.exe	Sercos-Master
serc.txt	German text
serc.hlp	German help file
serc.fnk	Drive-specific parameter lists
tdconfig.td	Configuration settings for the virtual Turbo-Debugger TD386

Example of an installation of SERC-Master

It is assumed that 2 slaves are connected to the SERCOS-module and that they are ready for operation.

- Start SERC-Master
- Go to the menu item for "Ring-Slave configuration" and enter the manufacturer and the physical SERCOS-slave address for slave1.
- Change to the second slave by means of the "Page Up" key and repeat the entry for manufacturer and address for slave 2.

The ALT-D (default) key combination configures all slaves with the default settings for the selected manufacturer. At this time the operating mode, the telegram type and the configurable telegram are defined. You can edit these settings at a later stage by changing into the sub-menus for the different slaves. Some menu items may already access the respective SERCOS-Phase for communication with the slaves. Once all the slaves have been configured we recommend that you save the configuration. For this purpose you must change to the menu item "Save Ring configuration file". You can include the configuration file as a command line parameter when the SERC-Master is started. The configuration is loaded automatically and transferred to the PLC. As an alternative, you can also load the ring configuration file by means of menu item "Load ring configuration file". Once the ring has been configured you can set the phase of the ring. Change to "Service" menu or start "cyclic communications" by adding and enabling individual drives in the menu "Regulation- drive control". You can obtain additional information from online-help and from the manuals.

Tel.: +49 (91 32) 744-0
Fax.: +49 (91 32) 744-144
EMail: info@vipa.de
http://www.vipa.de

Hotline: +49 (91 32) 744-114

About this manual

This manual contains a description of the SER-BG86 module. The module provides an interface between the PLC and servo drives or decentralised I/O systems with SERCOS-interfaces.

The module can be installed into programmable logic controllers PLC-115 (without fan module) through PLC-188 as well as central or expansion units.

Three versions of the SERCOS-module are available:

- Standard module (BG86), a module with DIN 66025 interface e.g. as a replacement for WF- or IP

- OEM version
  Standard module without software intended for companies that wish to implement their own software.

- SERCOS interface subsystem (not yet available). Standalone positioning controller that is controlled from a master system via a standard interface.

Overview

Chapter 1: SERCOS master software

This chapter contains the description of the SERCOS master software that is required for the configuration of the SERCOS-ring, for defining parameters for slaves and for cyclic communications.

Chapter 2: SERCOS hardware description

This chapter contains an introduction of the SERCOS environment. This is followed by a description of the module together with the pin allocation, the location of jumpers and installation locations in the PLC. The chapter is concluded with a description of the BIOS-setup and the bus interface for the page frame operation of the PLC.

Chapter 3: SERCOS program interface

Contains a description of the SERCOS software SERC-Master that provides data communication facilities between the PLC, the digital drives and the decentralised periphery.

The chapter also contains a description of the motion-control-functions along with an overview of data structures.

Chapter 4: SERCOS handler modules

Chapter 4 contains a description of the SERCOS function blocks that you require for PLC programming along with a description of the data modules.

The chapter ends with an example of the calling structure for two drives in the SERCOS ring.

Contents

1 SERCOS Master Software

1 SERCOS Master Software

1.1 General

1.1.1 SERCOS Master Functions

The SER-BG86 module is based on a 486 processor coupled to the SERCON-410B interface controller. A VGA-Monitor and a keyboard provide the user interface.

The module is managed by a controller of the PLC or in Stand-Alone mode (in the VIPA Stand-Alone enclosure).

A special area of memory is reserved for communications between the SER-BG86 and the PLC. This memory area is referred to as Dual-Ported-RAM (DPR).

Master-slave communications is handled by a closed fibre-optic ring (FO). It is possible to control and administer up to a maximum of 8 slaves from a single master.

The operation in conjunction with the ring requires a synchronisation run. The phase run-up (phase 0-4) is a process that places a all participants in ready mode. A ring configuration must be completed before the phase run-up is started. The settings for this ring configuration can be defined in a configuration mask. Phase run-up is started automatically when a program requiring communications via the SERCOS ring is selected. The configuration of ident numbers can be performed in phases 2, 3 and 4. Any configuration data (required data) is transferred via the service channel in a "non-cyclic transfer" mode. Ident numbers are a representation of function registers in the slave that the slave supplies, depending on its functionality. The user has access to simple configuration options for monitoring or modifying these ident numbers. All ident numbers can also be configured from the PLC by means of the control program.

From phase 4 a "cyclic transfer" is executed in parallel with the "non cyclic transfer". The "cyclic transfer" is interrupt controlled and it satisfies the respective real-time conditions. In general, cyclic data consists of time critical data that is transferred in real time to the die slaves.

1.1.2 Quick overview of SERCOS master functions

The SERCOS-Master-Software provides the interface for the exchange of data between the PLC, digital drives and the decentralised digital/analogue periphery. You can connect up to a maximum of 8 slaves to a fibre optic ring (FOC). The cycle time depends on the number of drives and the volume of data. Data communications comprises configuration data, nominal values, actual values, status messages for cyclic operations and the data for the service channel.

1.1.2.1 Functions that can be initiated by the PLC

- Initiate phase run-up
- Set operating mode (turn drives on, turn drives off, enable, stop)
- Initiate change of operating mode
- Transmission of nominal values for the cyclic telegram
- Reception of actual values from the cyclic telegram
- Reception of status and error conditions
- Read or write ident numbers
- Initiate or clear certain commands
- Deactivate drives

1.1.2.2 Functions that can be initiated by the SERCOS module

Creation of a ring configuration for later transfer to the PLC (the ring configuration consists of the manufacturer, drive address, operating modes, telegram types....)

- Control the phase run-up
- Control change of operating mode
- Reception of status and error conditions and transfer to the PLC
- Data communication via service channel
- Read and write ident numbers
- Execution of commands
- Diagnostic functions for the display of errors and operating modes
- Deactivate drives
- Display of telegram data from the cyclic data communications
- Control of telegram data
- Defining drives (single-step operation)

1.1.3 Quick overview of commissioning

The following steps must be completed before the phase run-up may be started.
First, the cycle time must be defined.

1.1.3.1 Default settings for drives

Menu ring: enter the ring configuration for all drives.

If you are using the configuration telegram:
Menu ring: configure the telegrams for the respective drives.

1.1.3.2 Default settings for digital/analogue periphery

Menu ring: Enter ring configuration for all peripherals
Menu ring: Configure telegrams for the respective periphery (as a rule the telegram has already been pre-configured in the slave)
Menu ring: Configure IO-offsets for the respective slaves

This concludes the ring configuration and it should be saved to a file under any name.

At this point all the information required for phase run-up is available. The phase run-up is started automatically when one of the following menus is selected:

File menu:	Load/save drive parameters
Control menu:	Control drive parameters
Service menu:	Configure ident numbers phase 2,3,4 Ident number status phase 2,3,4
Diagnostics menu:	Display diagnostic points Plain text diagnostics Display command status Disable certain drives
Cycle menu:	Status cyclic telegrams Control of cyclic telegrams Alignment operation

Tab. 1-1: Menu during phase run-up

1.1.4 Phase run-up

Phase run-up starts in phase 0. You can only execute a return to phase 0 from a high level phase. The master software executes a phase run-up to the required phase automatically. The basic requirement for the phase run-up is a valid ring configuration. It is also possible to initiate the phase run-up from the PLC.

Procedure

PHASE 0	The master synchronisation telegram (MST) is received 10 times in succession to check whether the ring is closed, i.e. whether all the slaves are in repeat-mode. Change into PHASE 1 is initiated by the master.
PHASE 1	Connected devices are identified by means of drive addresses. Change into PHASE 2 is initiated by the master.
PHASE 2	Establish a connection with the different slaves, read communication parameters from all the slaves to determine communication properties, calculation of transmission slots, transfer of initialisation data to the slaves. As an option the phase run-up can be stopped at this point to allow the transfer of user data via the service channel. Change into PHASE 3 is initiated by the master.
PHASE 3	As an option the phase run-up can be stopped at this point to allow the transfer of user data via the service channel. Change into PHASE 4 is initiated by the master.
PHASE 4	The initialisation is complete, cyclic operation has been activated. This cyclic operation is executed under interrupt control in the background of the SERCOS-Master-Software. A transfer of user data via the service channel can be executed in parallel.

Tab. 1-2: Phases of the phase run-up

1.1.5 Linking to a PLC

The communication link between the PLC and the SERCOS-module is provided by a DPR (Dual-Ported-RAM). The SERCOS-Master processes time-critical tasks that must be completed in real-time. It is one of the primary tasks of the PLC to start functions that are executed within the SERCOS-Master or to respond to messages from the SERCOS-Master. These include starting of the phase run-up, setting enabling conditions for regulators, initiating service functions, reacting to error conditions reported by the SERCOS-Master, etc. All nominal data values, actual data, status values etc. can be retrieved by the PLC. The PLC can also generate nominal values directly and transmit these to the different drives.

The software interface of PLC consists of handler modules that support all necessary SERCOS-functions.

The following functions are provided by means of handler modules:

- Read ring configuration
- Drives can be turned on, turned off, stopped or released
- Phase run-up
- Change of operating mode
- Service channel functions for non cyclic operation
  (read, write ident number, commands)
- Write nominal values, read actual values in cyclic operation
- Exchange of memory data

Please refer to chapter 4 for more detailed information.

1.1.6 Motion-Control

The software interface on the SERCOS-Master consists of a functional unit that is referred to as Motion-Control (abbreviated MC).

The SERCOS-Master software is provided with an entry point to Motion Control. From the point of view of the programmer the MC consists of an empty function for which the contents can be defined by the application, i.e. by the application programmer. The MC-function is called from the SERCOS-Master software on a periodic basis (at a rate set by the cycle time) and it has a limited amount of processing time at its disposal.

This means that it represents the interface with respect to the regulation-specific process. The user is provided with a library containing the master-software. Users can enter their own "C" language functions into the body of the function. These must be linked to the library modules to generate an executable EXE-file. This process requires that a complete development environment (e.g. Borland-C) is available.

Please refer to chapter 3 for more detailed information.

1.1.7 Silicon-Disk and PCMCIA

The standard SERCOS-module is supplied complete with a silicon-disk (EPROM). The silicon-disk contains an operating system, the SERCOS-Master software and drivers for the PCMCIA-interface. The silicon-disk occupies one logical drive name. You can only read data from the silicon-disk, i.e. it is only possible to read and start programs from there. You can use the PCMCIA-interface if you wish to create files, e.g. a ring configuration or a status list with ident numbers. The PCMCIA-interface can be used to connect a PCMCIA hard disk or a PCMCIA FLASH-disk with an ATA interface.

1.2 Program structure

1.2.1 Main menu

These are the items on the main menu:

File

- Load ring configuration
- Save ring configuration
- Load drive parameters
- Save drive parameters
- Exit

Ring

- Set ring cycle time
- Configure slaves
- Transmit configuration

Control

- Drive control
- Drive shut-down

Service

- Ident. no. configuration phase 2
- Ident. no. configuration phase 3
- Ident. no. configuration phase 4
- Ident. no. status phase 2
- Ident. no. status phase 3
- Ident. no. status phase 4
- Command functions
- Print ident. no.

Diagnostics

- Display diagnostic items
- Plain text diagnostics

Cycle

- Slave status
- Slave control
- Defining slaves

Help

- Contextual help
- Help index
- Help on help
- Version

1.2.2 Status indicators

The bottom of the screen displays a list of indicators. These indicators provide information on the participation of a slave in the ring and about its status. Slaves appear here once the slave address has been included in the ring configuration menu. The status indicator is retrieved from the status word of the slave.

The following status conditions exist:

	No slave installed
N.BEREIT	Slave installed but not ready
BEREIT	Slave is ready for power on
LZBEREIT	Control section, power on ready
BETRIEB	Slave is operational
ZKLASSE1	Error occurred in status class 1
ZKLASSE2	error occurred in status class 2

Tab. 1-3: Status conditions of the slave

A description of the different menu items follows below.

1.2.3 Menu File

The following items are available from the sub-menu

- Load a ring configuration
- Save a ring configuration
- Load drive parameters
- Save drive parameters
- Exit

1.2.3.1 Loading, saving the ring configuration

Loading and saving of ring configurations is possible via dialogue windows. These dialogue windows display directory entries and they may be used to select a different drive or to select another file. You may also use wildcards to select file names.

The ring configuration consists of a binary file containing the information on all the slaves on the ring. A ring configuration that was prepared from the main menu item "Ring" may be saved under any file name. You can also load ring configurations that have previously been saved. The file name extension is pre-set to **".cfg"** . Before a new ring configuration is loaded the ring must be in phase 0 (see also chapter 1.2.5.2).

Ident number file – file name extension ".idn"

The service function "Monitor/configure ident numbers" is available to save a mask containing ident numbers to a file with any required name. The mask can contain a maximum of 12 ident numbers along with the respective operational parameters. The file name extension is pre-set to **".idn"**. You can also load a previously saved mask containing ident numbers and operational parameters to display the respective status.

1.2.3.2 Loading, saving drive parameters

You must specify the slave number before you can load or save ident numbers. Only files that were saved to a file by means of "Save ident numbers" can be loaded. These files contain a list of ident numbers. The list is also identified by an ident number that is saved in the slave. The Id. no is pre-set to 192 (list of ident numbers that must be saved). You may, however, save any list of ident numbers.

You may use any file name for lists of ident numbers containing drive parameters. The file name extension is pre-set to **".lst"**. Saved lists of ident numbers containing operational parameters can be transferred to any required slave.

Key	Function
Page ↑	Page forward to the next slave
Page ↓	Page back to the previous slave
ALT-A	Load an ident number file
ALT-I	Save the specified list into an ident number file

Tab. 1-4: Function summary load/save drive parameters

1.2.3.3 Exit (terminate program)

A check determines whether the ring is switched into phase 0 before the program is terminated. If the ring is switched to a higher phase number the drives are shut down (ring is set to phase 0) before the program is terminated.

1.2.4 Menu ring

The following sub-menu items are available

- Specify ring cycle time
- Configure slaves
- Transfer configuration

1.2.4.1 Specify ring cycle time

The cycle time of the ring (TScyc) may be modified. The default cycle time depends on the number of drives. The cycle time may only be modified in phase 0. Changes must be in 1 ms steps or in integer multiples of 1 ms.

1 to 2 drives	2 ms
3 to 4 drives	3 ms
5 to 6 drives	4 ms
7 to 8 drives	5 ms

Tab. 1-5: Default settings

1.2.4.2 Slave configuration

Here you can enter the manufacturers name, the physical address and the drive type. These parameters are obligatory for every slave. All other settings can be defined via the sub-menu "Default". Once "Default" has been activated the operating mode, telegram type, the drive telegram (AT) and the master data telegram (MDT) are configured depending on the manufacturer and the axis type. For this purpose a change to phase 2 is executed to determine telegram sizes etc. The number of slaves in the ring is determined automatically from the number of slave addresses that have been assigned. Every slave with an address other than 0 is identified as a participant on the ring.

Key	Function
ALT-D	Selection of default values for operating mode, telegram type and configurable telegram, depending on the manufacturer an the axis type (may be altered at a later stage)
ALT-E	Set operating mode
ALT-C	Set real-time status bits
ALT-T	Set telegram type
ALT-K	Telegram configuration
ALT-O	Configuration of I/O-offsets
ALT-M	Motion-control configuration

Tab. 1-6: Sub-menu to configure slaves

The configuration of the decentralised periphery requires the following steps, but only once:

- Erase control track
- Enter I/O-offsets
- Change to the menu for the configurable telegram and check or adjust the telegram data
- Save the ring configuration
- Start phase run-up up to phase 4
- Shut-down of drives (change to phase 0)
- Initiate a PLC restart so that the input and output areas may be acknowledged by the PLC.
- Restart phase run-up up to phase 4, periphery is ready.

These steps are necessary when one considers the re-configuration of a slave, e.g. when an I/O-module is inserted into a running system. Ted PLC can only recognise and enter any new modules into the control track after a restart.

It is important that overlaps of I/O-areas must not occur during the I/O-configuration and slaves require at least 1 word starting at an even offset address. Handler modules are only required for the purpose of enabling the slaves of digital/analogue I/Os! Communications with the I/O modules occurs in transparent mode.

1.2.4.2.1 Selection of operating mode

This menu item is used to assign of the type of regulation to the different operating modes.

The type of regulation is entered in hexadecimal. The type of regulation may consist of a number between 0 - FFFFh. In addition to the standard regulation types defined by SERCOS it is thus also possible to enter manufacturer specific types. When the enter key is depressed the name of the selected regulation type is displayed in plain text.

Refer also to **SERCOS interface specification** "Operating modes of drives".

The following 4 operating modes are available

- Main operating mode
- Auxiliary operating mode 1
- Auxiliary operating mode 2
- Auxiliary operating mode 3

The operating mode parameters are used to store the required operating modes (types of regulation) for the slave. A change between operating modes in cyclic operation is initiated by means of the control word in MDT.

Example

Main operating mode = velocity control (2h)

Auxiliary operating mode 1 = attitude control (3h)

In cyclic operation this provides the facility to switch between the main operating mode (velocity control) and auxiliary operating mode (attitude control).

All slaves that support ident numbers for operating modes can utilise any one of the types of regulation that are stored for the main operating mode and auxiliary operating modes 1, 2, or 3.

The switch between operating modes can be initiated directly by the PLC or by the SERCOS-Master-Software. (Refer also to the **SERCOS interface specification** "Operating modes of drives").

The "Motion Control operation" switch is activated by means of the "SPACE" key.

1.2.4.2.2 Real-time status bit settings

Real-time status bits are available where the slave provides support for real-time signals. Real-time status signal may be selected separately for every operating mode. The status of the real-time status bits is transferred to the PLC at the rate determined by the selected cycle time.

Two real-time status bits are provided in every slave. These may be used in conjunction with special allocations (see ident numbers S-0304, S-0306). Real-time status bits provide the master with a real-time indication of selected statuses or events that may have occurred in the slave. It does not make any sense to allocate more than one ident number at any one time. The operating data of this number represents a binary signal (1bit).

Example

S-0-0330	Message N-actual = N-nominal	velocity-actual = velocity-nominal
S-0-0331	Message N-actual = 0	velocity-actual within standstill window

If the slave does not support the ident numbers for real-time status bits it, for instance, is possible to enter ident number 13 (status class 3) into the cyclic telegram. This is also transferred to the PLC.

1.2.4.2.3 Setting the type of telegram

The telegram type determines the contents of the telegram. Here you may choose from preferred telegrams 0 to 6 and the configurable telegram.

Refer also to **SERCOS interface specification** "Defining the contents of telegrams".

An automatic change to phase 2 is executed for the configuration of telegrams. All the ident numbers that are accepted by the slave for the MDT or the AT telegram are displayed under "configurable ident numbers" (refer also to **SERCOS interface specification** "Ident number 188, 187"). One or more ident numbers may be selected from these ident numbers as the contents of the cyclic telegram for the cyclic data communications. Multiple ident numbers must be separated by means of a semicolon (;). Telegram content is entered as a list into ident numbers 24 or 16 (refer to **SERCOS interface specification** "Ident number 16, 24").

1.2.4.2.4 I/O-Offset configuration

I/Os are referred to as decentralised periphery in the SERCOS-Ring. The number of inputs and outputs is determined automatically during the phase run-up (in phase 2). You must only enter the offset of the inputs and the outputs.

During the phase run-up the input and output area with the respective width is entered into the control track of the PLC. The control track of a memory area of the PLC that is used to store the position of the periphery within the memory area.

1.2.4.2.5 Motion-Control configuration

Where a slave is used in "Motion-Control" mode (setting in operating mode menu) you must define the input parameters for this purpose here. A maximum of 8 input parameters can be defined. The length must also be specified as a restriction. A single parameter can consist of a word (16 bit) or a double word (32 bit).

The data length may contain:

"1"	parameter consists of one word
"2"	parameter consists of a double word

The format determines whether the parameter is shown in decimal-, hexadecimal- or a fixed-point representation.

The following are possible for the format:

"d", "D"	decimal representation
"h", "H"	hexadecimal representation
"f", "F"	fixed-point representation

The name and the unit are used for textual purposes and are optional.

1.2.4.3 Configuration transfer

The selected ring configuration must be transferred to the PLC. Only when this has been done is it possible to initiate communications for the exchange of cyclic data, etc. via the DPR.

1.2.5 Menu control

The following sub-menu items are available

- Drive control
- Drive shut-down

1.2.5.1 Drive control

Slaves may be controlled (on, off, halt) from the SERCOS-Master-Software. A condition for this is that the power on control is ready and that a phase run-up to phase 4 was execute. The status indicator displays "READY".

1.2.5.2 Drive shut-down

A drive shut-down is accompanied by a phase change into phase 0. This shuts the drives (slaves) down.

All drives must be shut down before you can change the ring configuration or before you can load a new ring configuration.

1.2.6 Menu service

The following sub-menu items are available

- Ident. no. configuration phase 2
- Ident. no. configuration phase 3
- Ident. no. configuration phase 4
- Ident. no. status phase 2
- Ident. no. status phase 3
- Ident. no. status phase 4
- Command functions
- Ident. no. print

1.2.6.1 General information on data communications via the service channel

The service channel is a functional unit providing access to the data in the slave. A data transfer via the service channel is always initiated and controlled by the master. The data that is transferred via the service channel consists of user data that is only transferred upon request and not in cyclic mode.

User data consists of parameters of the slave that may be changed and that are addressed by means of ident numbers. The data transfer via the service channel starts with phase 2. It is possible to change to one of the phases 2, 3 or 4 for configuration purposes or to monitor ident numbers (see chapter 1.2.6.3 and 1.2.6.4).

Non-cyclic data transfers provide the following operations:

- Initialisation of the SERCOS-interface by means of initialisation data
- Transfer of all the elements of a data block via the ident number
- Initialisation of the SERCOS-interface
- Starting commands
- Changing limit values
- Changing regulator parameters
- Detailed status messages from drives
- Diagnostic functions

1.2.6.2 Ident numbers

An ident number is a data block consisting of 7 elements (refer also to **SERCOS interface specification** "Data block structure").

Element 1	Id. No.	Data block number
Element 2	Name	Data block name
Element 3	Attribute	Used for displaying element 7
Element 4	Unit	The units of element 7
Element 5	Minimum value	Minimum value of the data in element 7
Element 6	Maximum value	Maximum value of the data in element 7
Element 7	Data	Operating data

Tab. 1-7: Data block structure

The data block in the slave consists of a group of registers. Every slave has a separate set of ident numbers consisting of a certain sub-set of ident numbers that are defined in the respective **SERCOS interface specification** (refer also to the respective **SERCOS interface specification** "General parameters"). These ident numbers are also known as standard ident numbers (standard data "S"). A slave may also have certain product ident numbers (product data "P"). Every data block in the slave must provide elements 1, 3 and 7. Elements 2, 4, 5 and 6 are optional.

1.2.6.3 Ident number configuration

The operating data of slaves is identified by means of ident numbers. Every ident number is associated with a data block that may be read in phase 2, 3 or 4. Write operations can only be used on the operating data of the data block.

Commands may be initiated by writing an ident number which has been assigned to a command function. In this case the operating data is ignored. The integrated command controller automatically inserts the required entries into the operating data.

You can scroll through the "data" field horizontally.

Refer also to the respective **SERCOS interface specification** "Ident number structure", and "Data block structure".

1.2.6.4 Ident number status

In pases 2, 3 or 4 you can display a list of ident numbers if you want to monitor certain ident numbers . You may enter up to 12 ident numbers per page into the column with the heading Id. No. The range of an ident number is from 1 to 4095. You may specify standard ident numbers (type S) and product ident numbers (type P).

Key	Function
ALT-Z	Display ident numbers that are included in the mask
ALT-I	Display the ident number as a list
ALT-P	Configure ident numbers
ALT-F	Select an Id. No. function group
ALT-C	Erase entire page of ident numbers
ALT-N	Insert ident numbers into a page
ALT-T	Delete an ident number from a page
ALT-L	Load a page of ident numbers from a file
ALT-S	Save a page of ident numbers to a file

Tab. 1-8: Function summary Id. No. status

You may select type S or type P when you are using the list function. You can change the selection by entering type S or P into the first line under the column Id. No. When the list function is activated the displayed page will contain the new type. The display starts from the value that was entered into "Offset". When you depress the "function" key a list of function groups will be displayed.

You can scroll horizontally through the fields for "Name", "Date" and "Unit".

1.2.6.5 Function groups

A function groupis a list of ident numbers that perform a certain task.

You can make a selection from a list of function groups that is displayed. The selected group is loaded as ident number list.

For the "Commands" group an ident number list is loaded that is relevant for the execution of the respective command.

Key	Function
ALT-S	Set command
ALT-L	delete command

Tab. 1-9: Function summary of the "commands" group

1.2.6.6 Command functions

Commands are instructions sent by the master that initiate a sequence of events in the drive, e.g. return to reference location. A command has the structure of a data block that is referenced by means of an ident number. Bit 19 of element 3 (attribute) identifies a command. In the SERCOS-Master-Software a command is initiated when the respective ident number is written.

It is possible to enter multiple ident numbers (having command functions). You can place a tick on the write or delete command in the input fields for "write" and "delete" by means of the "SPACE" key. However, the write or delete command is only executed when the write or delete function is initiated. The status of the command can be interrogated by means of then ALT-Z key combination. The command status displays the allocation of the command channels for slaves S1 through S8.

Key	Function
ALT-S	Initiate the command (write)
ALT-L	Delete command
ALT-Z	Update command status

Tab. 1-10: Function summary command function

1.2.6.7 Printing ident numbers

The slave number must be selected before ident numbers can be printed . You can print any ident numbers from 1 - 65536 (default 1-4065) or a list of ident numbers that is stored in the slave. You can select between LPT1 or a file for the printout, the progress of the print operation is displayed at the bottom of the screen. As the print operation is executed in the background you may continue program operations while the printout is being created.

Key	Function
ALT-T	Print "from - to" (start and end is defined in the mask)
ALT-I	Print the ident numbers of the selected list
ALT-B	Abort an active print job

Tab. 1-11: Function summary printing

1.2.7 Menu diagnostics

The following submenu items are available

- Display diagnostic items
- Plain text diagnostics

1.2.7.1 Display diagnostic items

The diagnostic items correspond with the standard ident numbers for diagnostic purposes (refer to the respective **SERCOS interface specification** "status classes and diagnostics").

The following functions are available and they always refer to a slave:

Key	Function
ALT-Z	Display diagnostic items
ALT-R	Mask entries for status classes 2, 3
ALT-K	Reset status class 1, 2, 3

Tab. 1-12: Function summary diagnostics

1.2.7.2 Plain text diagnostics

The diagnostic telegrams of all the slaves are displayed (see also ident number 95).

Where errors are reported from status class 1, 2, 3 or from the manufacturers status class 2, 3 these errors are entered in plain text into the log.

1.2.7.3 Status classes

Status classes belong to the diagnostic telegrams of the slave (see the respective **SERCOS interface specification** "Status classes and diagnostics").

Status classes are divided into 3 categories.

Status class 1

Locks a drive. Provides the best-possible shut-down and subsequent torque release.

Status class 2

Preliminary shut-down warning

Status class 3

Signalling of operating modes

Status class 1 must be acknowledged via the service channel by initiating the command Reset-Status class 1 (ident number 99). The error bit is only cleared when all status class 1 errors have been removed.

The Master-Software displays status class 1 or 2 errors in the status indicators located at the bottom of the screen. Status classes of all slaves are reset in the main menu of the SERCOS-Master or individually in the diagnostic menu "Display diagnostic items".

1.2.8 Menu cycle

The following submenu items are available

- Slave status
- Slave control
- Slave definition

1.2.8.1 General information on cyclic data communications

The contents of the telegram for cyclic data communications is defined by the telegram type or the configurable telegram. For this purpose the ring configuration must be valid. Cyclic data communications starts in phase 4. The ring configuration must have been transferred to the PLC before cyclic communications can be established with the PLC. It is also not possible to change the configuration in phase 4. If a change should become necessary the ring must first be switched to phase 0 (drive shut-down). With regard to the program the cyclic data communications is executed under real-time conditions (interrupt controlled). The timing of the execution for nominal values and actual value processing is guaranteed by the SERCON410B. Due to the fact that cyclic data communications runs in background normal program operations (service channel-functions etc.) can take place in the foreground.

1.2.8.2 Slave status

The slave status displays the cyclic telegram data of all the slaves in the form of a list.

The representation of cyclic telegram data depends on the selection of the telegram type or on the configurable telegram. The data (MDT nominal value, AT actual value) for every ident number in the cyclic telegram is displayed on a new line. If the telegram should contain multiple ident numbers for a single slave these are listed one below the other.

Key	Function
Page ↑	Display the next page
Page ↓	Display the previous page

Tab. 1-13: Function summary slave status

1.2.8.3 Control of the cyclic telegrams

The representation of cyclic telegram data depends on the selection for the telegram type or the configurable telegram as well as the definition for Motion-Control in the ring configuration. Every slave is displayed on a separate page together with the Motion-Control parameters defined for the slave. If the telegram should contain multiple ident numbers for a single slave the telegram data for nominal and actual values are listed one below the other. Motion-Control is only active when the switch for "Motion-Control active" is set at the time when the operating mode is being selected. If Motion-Control is not active the drive follows the nominal values supplied by the PLC directly.

Key	Function
Page ↑	Display the next page
Page ↓	Display the previous page
ALT-S	Terminate the status display, you may control the cyclic nominal values
ALT-C	Control of MC, initiates the transfer of the MC nominal values
ALT-T	Control MDT, initiates the transfer of the MDT nominal values (direct control of the slave)

Tab. 1-14: Function summary control of cyclic telegrams

1.2.8.4 Defining drives

You can enter the definition for any individual slave when you have selected a slave number. Use the "ALT-E" key combination to activate the definition routine for the slave. The routine will first determine whether the slave is in standstill mode (S-0124). You can use the cursor keys to move the cursor in a positive (cursor right) or a negative (cursor left) direction. The cursor keys ↑, ↓ determine the speed of the movement. This can be in a range lying between 10 % and 200 % of the velocity pre-set for the definition operation. The positioning velocity is changed in steps of 10% and it is displayed as "override".

Key	Function
Page ↑	Page to the next slave
Page ↓	Page to the previous slave
ALT-E	Activate the definition mode

Tab. 1-15: Function summary defining drives

1.2.9 Menu Help

The following submenu items are available

- Contextual help
- Help index
- Help on help

The program is provided with an extensive help facility. This system contains contextual help and an index of the different help subjects.

The help text of the windows of contextual help refer to the current display where you are located when the function is activated.

2 SERCOS Hardware Description

2 SERCOS Hardware Description

2.1 Introduction

2.1.1 Information and history

SERCOS has become the communication interface for computerised numeric control machines. Initially, this serial bus for real-time communications was developed as a manufacturer independent standard interface, especially for drive control applications. Subsequently the SERCOS interface specifications were expanded to include peripheral equipment. In a common work group that included the VDW (Gemeinschafts-Arbeitskreis des Vereins Deutscher Werkzeugmaschinen-fabriken e. V.) and ZVEI (Zentralverband Elektrotechnik- und Elektronikindustrie e. V.) as well as renowned manufacturers of machine-tools, drives and numeric control machines have defined a uniform set of interface specifications. To provide a clear and unambiguous arrangement of the specification, drive related functions are separated from the input/output functions for the machine-tool periphery. For this reason the entire physical input/output process on the sensor-/actuator-level is processed by I/O field stations.

Except for drive control, a field station may be used to connect the entire machine-tool periphery to the SERCOS interface. A massive reduction in the required wiring becomes possible since the machine-tool periphery as well as drives can communicate with the controller on the next level via the same fibre-optic ring (this can naturally also be used separately!). Thus fibre-optic technology also has advantages for communications on the sensor/actuator level. Interference to signals is minimised, potential differences between the machine-tool periphery has no effect and complications due to electromagnetic compatibility are history. Cycle times guarantee that the process image available to the controller is always up to date.

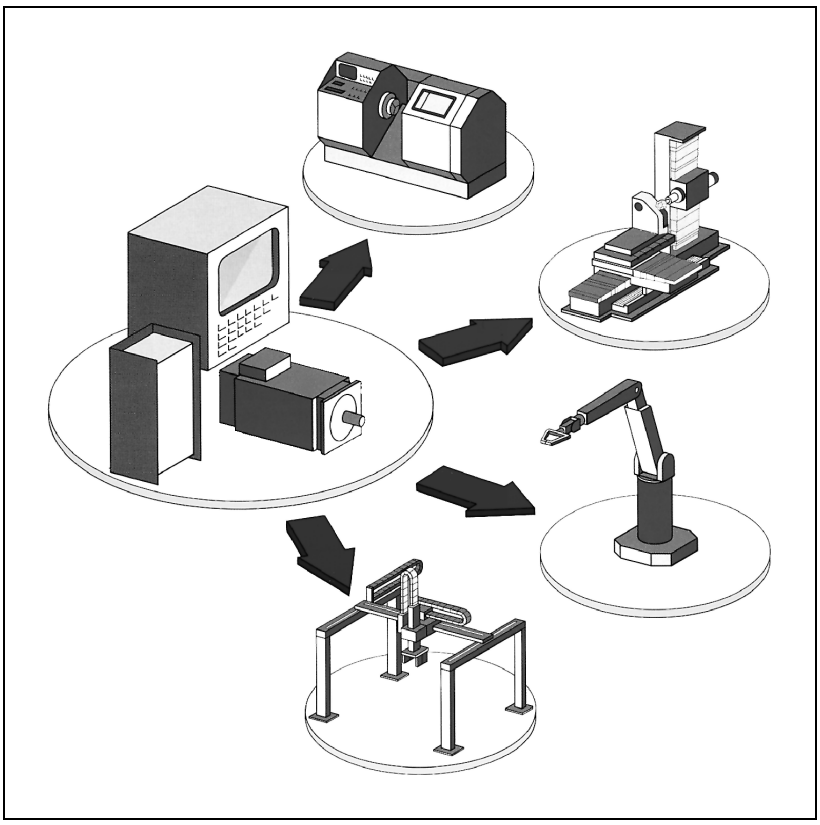


Fig. 2-1: SERCOS on site

2.1.2 Abridged topology

Physically the SERCOS interface is a Master/Slave-System with a ring-topology. A fibre-optic ring connects the transmitter of the master to the receiver of the first slave. This slaves transmitter is connected to the receiver of the next slave, and so on. A fibre-optic connection between the transmitter of the last slave and receiver of the master closes the ring.

Communications is based on a cyclic sequence of HDLC telegrams that are spaced at a fixte timing raster (”High-Level Data Link Control”). Only a single telegram is transmitted via the FO-link at any one time. The pacemaker for this race is the master: the master starts communication cycles at a fixed repetition rate ($1/T_{SCYC}$). Slaves re-transmit received telegrams after a short delay time (Repeater function). In the end the master will receive all telegrams, including his own. In this way the master can always detect line errors.

The cycle is started when the master transmits the master-synchronisation telegram (MST). This is followed by the slaves transmitting their actuaol values in the allocated drive telegrams (AT1, AT2 through ATm). This cycle is terminated by the master data telegram (MDT) from which the slaves retrieve information addressed to them. When the cycle time T_{SCYC} has expired the master transmits the next MST.

Telegrams use NRZI-coding (”Non-Return-to-Zero Inverted”) and the master transmits a filler signal containing clock-recovery transitions between telegrams. Every telegram is enclosed in a Begin-of-Frame (BOF)- and an End-of-Frame field (EOF). Every telegram also contains an 8 bit address field and a 16 bit CRC-field (”Cyclic Redundancy Check”). The significance of the remaining field in the telegram depends on the telegram type. Thus the MST only contains an 8-bit phase information. The respective field of an AT (drive telegram) consists of a data record. This is sub-divided into a fixed part containing the status word and the service-info word, and a configurable part for data that must be transferred on a cyclic basis. Finally, the MDT contains M-data records, one for every AT; in this case the status word is called control word.

Cycle times depend on the number of ATs, the data rate in the ring and the quantity of data that must be transferred. The number of drive telegrams is limited to 254 due to the 8 bit address field: address 0 is reserved for test purposes, the broadcast (”to all”) address 255 is located in the MST- and MDT-address field. The data rate is pre-set to 4 MBit/s. The SERCOS interface specifies the following cycle times: 63 µs, 125 µs, 500 µs and 1 ms and multiples of 1 ms up to a maximum of 65 ms.

2.1.3 Operation

The SERCOS interface ring SER-BG86 supplied by VIPA may be connected to simultaneously drive amplifiers from different suppliers as well as decentralised I/Os. In the latter case images of the SERCOS-ring I/Os are copied into the peripheral area of the PLC. This means that I/Os are initialised with the same I/O addresses as in the PLC-rack. In this manner decentralised peripheral components are available for access by PLC programs in the same way that local PLC peripherals are accessed. The VIPA SER-BG86 automatically detects peripheral modules connected to the SERCOS-ring and copies the respective images into the PLC.

At the moment the SERCOS interface can be used to connect up to 8 slaves to the fibre optic ring with a typical cycle time of 2 ms. The exact maximum number of slaves that each ring can manage depends on the cycle time, the selected data volume and the data rate (currently 4 MBit/s). The number of slaves per controller can be increased as required by means of additional fibre optic rings.

Data communications between the module and the PLC-CPU is performed by means of handler modules. You can install additional SERCOS-rings on each PLC by inserting further modules. Extensive diagnostics, for instance of the SERCOS-ring or the axis status and plain text telegrams about error conditions, diagnostic telegrams from the drives and I/O status indicator simplify commissioning and service. A printer interface is available for printing of parameters and program listings. Data is stored in a removable PCMCIA hard disk (PCMCIA 2.0, type I, II, III).

If you require further information you may wish to contact the society for the development of SERCOS (Fördergemeinschaft SERCOS interface e. V. Herseler Straße 31, D-50389 Wesseling, Tel. (02236/ 1517) or contact us by telephone.

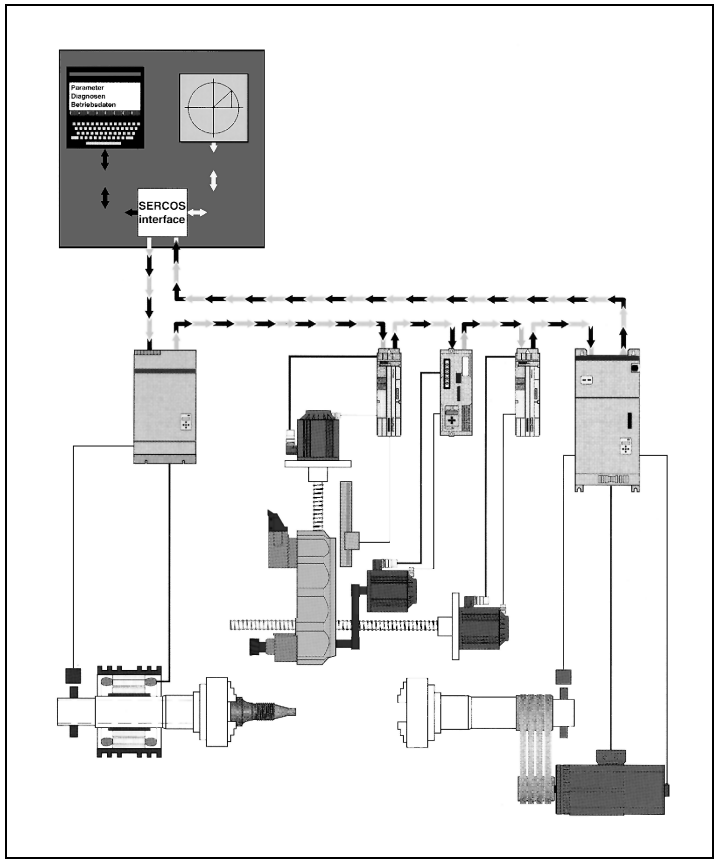


Fig. 2-2: Sample application: Feed axis

2.1.4 Application area

Main and servo drives on machinery, handling equipment and utility plants.

The digital SERCOS drive sets new standards with respect to functionality, dynamics, positioning and repeatability accuracy. In addition to the normal operating modes, these drives provide other functions like interpolation, electronic gearing, gantry-axis and electronic cams.

For this reason the VIPA SER-BG86 is suitable for the following applications:

- Transfer lines
- Conveyors
- Assembly lines
- Presses
- Roll feed ales
- Feed lines
- Woodworking machines
- Packing machines
- Rotary transfer machines
- Special multi-axle machines
- Gear wheel machine tools

2.1.5 Structure and operation of the PLC coupling

Dualport-RAM

The PC is connected directly to the back plane bus via a Dualport-RAM. On the PLC this Dualport-RAM is available as standard-CP-interface with 8 page frames. Data communications can be controlled by means of handler modules.

Communication software

The communications between the PC and the PLC can be controlled by means of handler modules that perform the actual exchange of data.

2.1.6 Block diagram of the SERCOS module

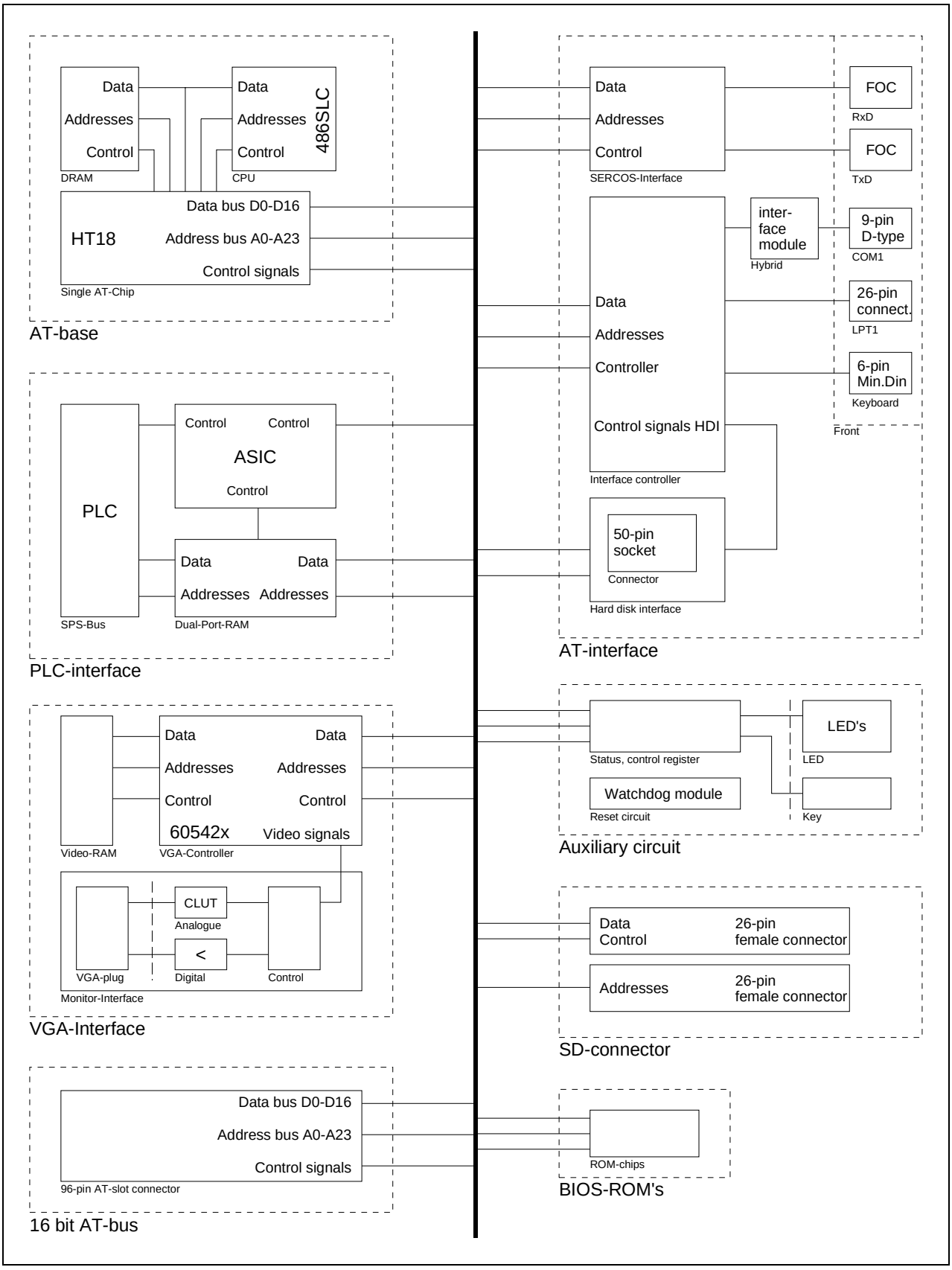


Fig. 2-3: Block diagram

2.1.7 Special components

Interface modules

The serial interface is configured by means of plug-in modules. COM1 is configured as an RS232C interface by default. An optional 20mA interface is also available.

Battery backup

Data retention for the BG86 is maintained by the battery in the PLC. The module is also fitted with a rechargeable lithium battery that can maintain the BG86 data while it is disconnected for up to 6 months.

PCMCIA interface:

The PCs PCMCIA interface is a socket for different credit-card sized expansion modules. This socket may be installed as an option on modules fitted with a PC slot. These modules are accessed via drivers. At present PCMCIA modules are available as modems, memory expansion modules, hard disks, network adapters etc.

The BG86 uses a CIRRUS-LOGIC PD6710 chip. The respective drivers are supplied by Phoenix Technologies.

To support the PCMCIA interface it is required that different PCMCIA drivers are installed in the "config.sys" file. The drivers listed below are already installed on ROM-Disk A: or hard disk C:

CNFIGNAM.EXE	(parameters from PCM.INI)
PCMSSCL.EXE	(install the socket-service)
PCMCS.EXE	(install the card-service)
PCMATA.SYS	(install logical-driver for PCMCIA ATA-disk)
PCM.INI	(configuration file with initialisation data)

Please refer to the "PhoenixCARD Manager Plus" version 2.21 manual if you require detailed information.

2.2 Hardware

2.2.1 General

3 versions of the SERCOS interface module are available :

- Standard module (BG86): a module with a DIN 66025 interface, e.g. WF- or IP-replacement

- OEM version
  standard module without software, for manufacturers who wish to install their own software.

- SERCOS interface subsystem (not yet available). Standalone positioning controller that is controlled by a master system via a standard interface.

Tabular Overview

Module	Standard module	OEM version	Subsystem
RAM	4MB	8MB	4MB
FPU	no	Socket 80387	no
COM1	RS232C	RS232C (opt. 20mA)	20mA
Keyboard adapter	yes	no	yes
Firmware EPROM	1MB	no	1MB
Firmware RAM	1MB/4MB	no	1MB/4MB
Hard disk	no	340MB	no
PLC interface	yes	yes	no

Tab. 2-1: Tabular overview

2.2.2 Construction of the BG86 standard module

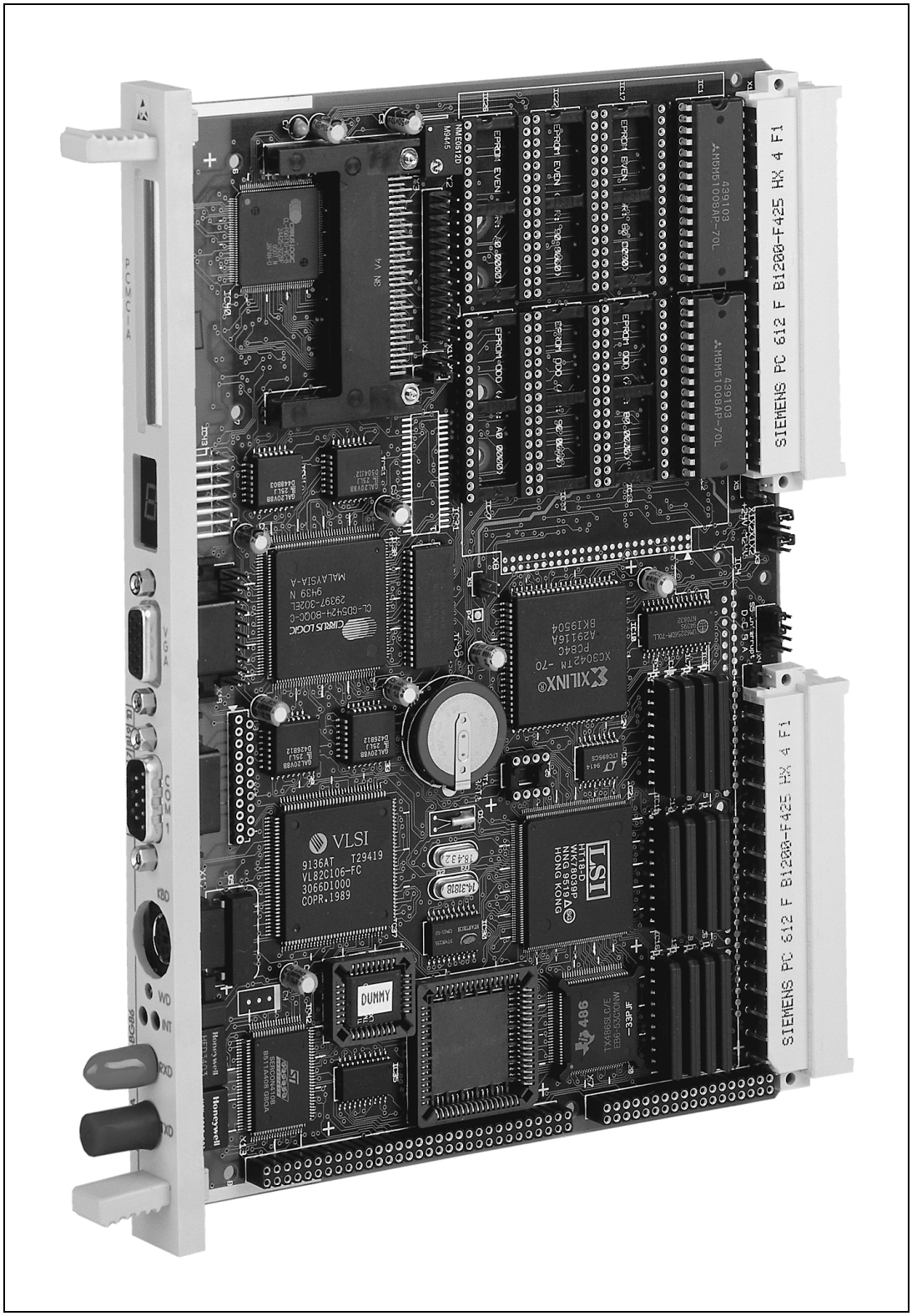


Fig. 2-4: Construction of the module

2.2.3 Controls and indicators on the front panel

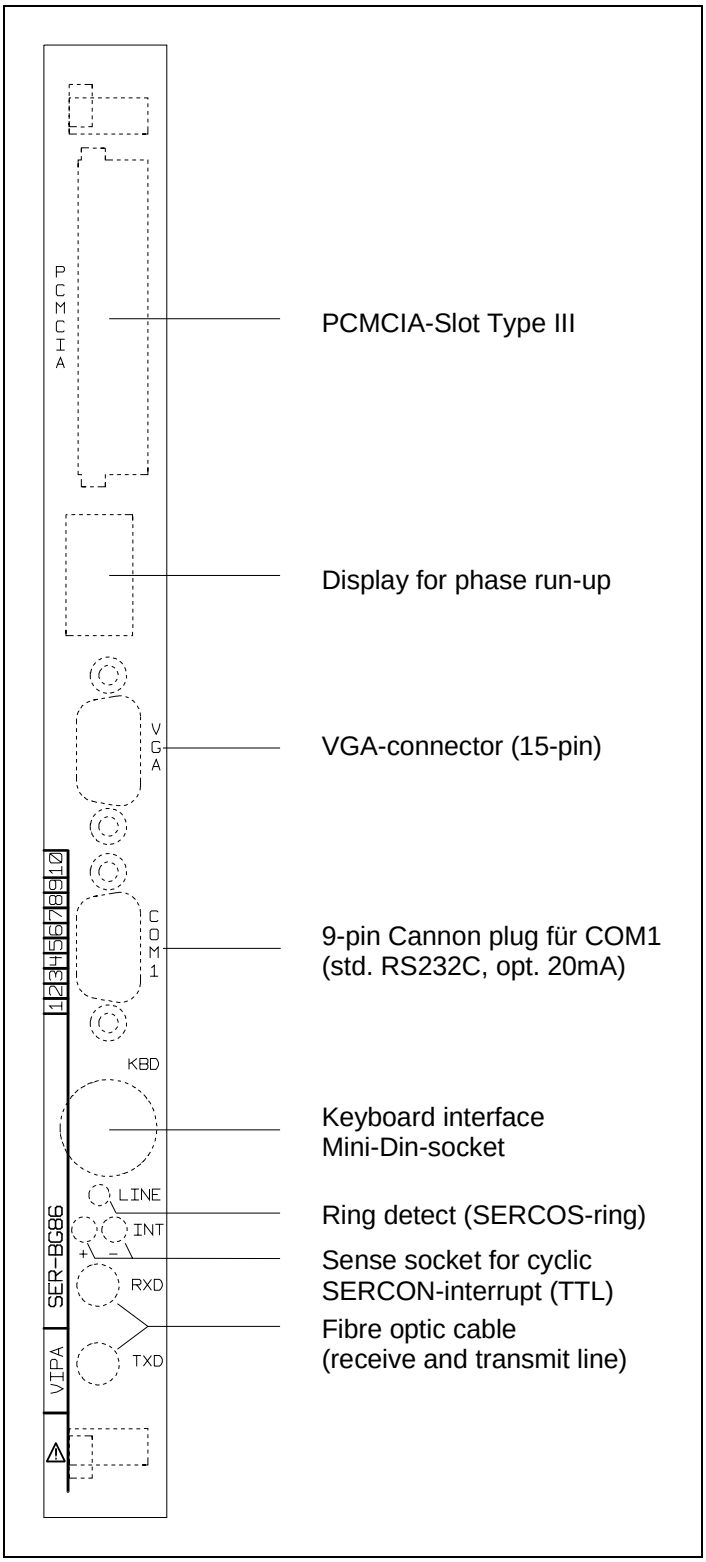


Fig. 2-5: View from the front of the module

The following elements are located on the module:

- CPU80486SLC and FPU80387SX socket
- SERCON module
- Main memory 4MB/8MB/16MB
- Keyboard interface
- VGA graphic adapter
- Socket for serial interface module (COM1)
- Parallel interface (LPT1) (optional with additional adapter)
- Socket for silicone disk (8 memory ICs)
- Externally accessible 16Bit-AT-bus
- 8K Dualport-RAM for the PLC as CP-interface
- Optional hard disk interface

2.2.4 Layout of jumpers and sockets

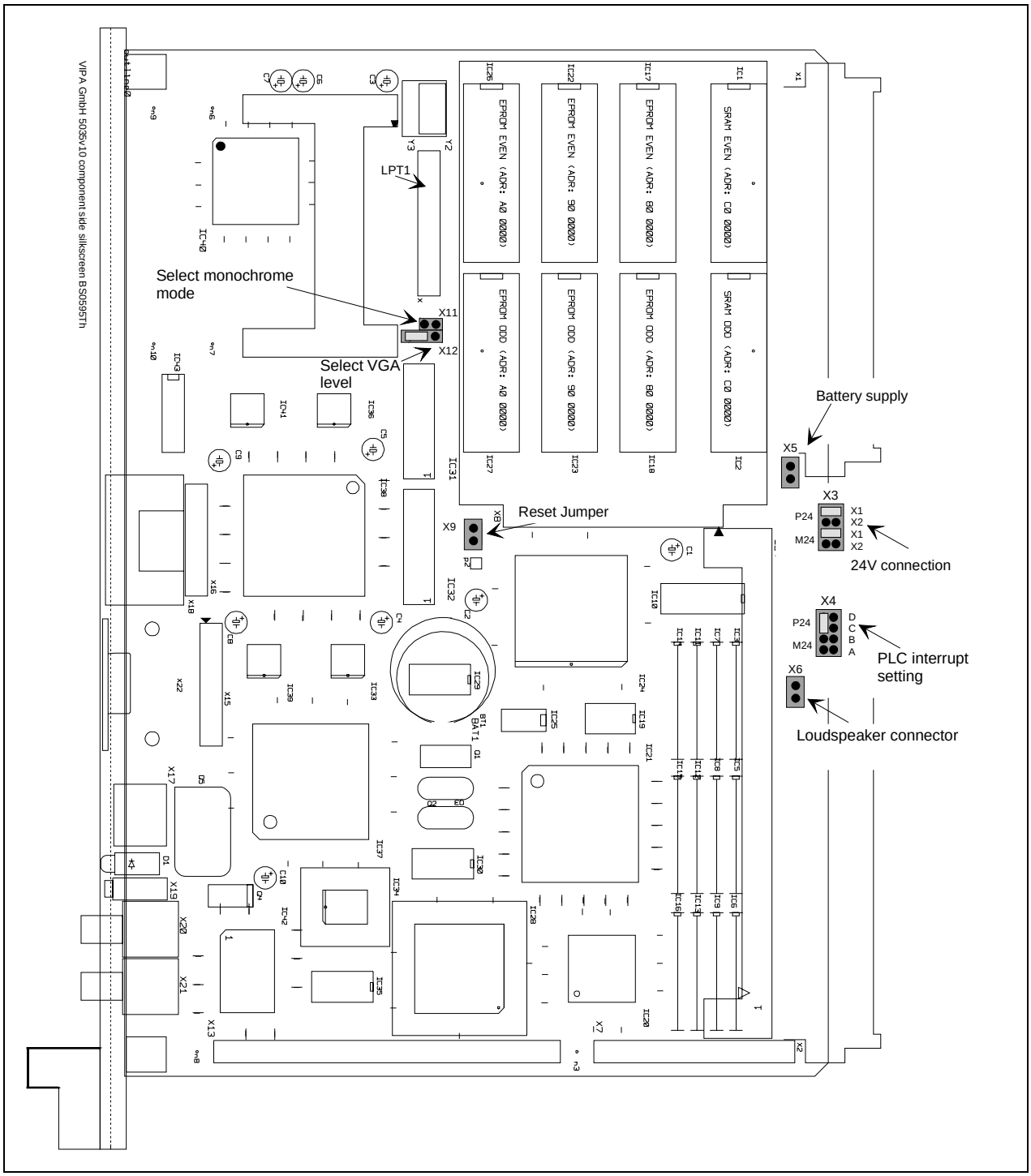


Fig. 2-6: Location of jumper

Jumper functions

Jumper	Description
X3	24 V circuit
X4	PLC-interrupt assignment to PLC-interrupt A,B,C or D. Default assignment: no jumpers installed
X5	UBAT external Jumper installed: battery voltage supplied by PLC
X6	Loudspeaker connector
X9	Reset jumper Jumper installed: module is being reset
X11	Monochrome mode Jumper installed: monochrome mode Jumper open : auto detect
X12	VGA level setting OFF : low 1 <–> 2 : high 2 <–> 3 : normal

Tab. 2-2: Jumper functions

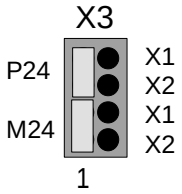
2.2.5 24V power supply configuration

Then BG86 must be connected to a 24V DC supply under the following conditions:

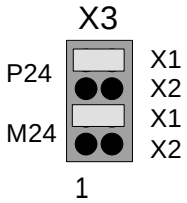
- for connection to an active 20mA interface

The 24 Volt supply is selected by means of jumpers X3:

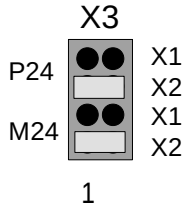
Jumper X3: 24V-supply off



Jumper X3: Supply voltage connected to upper back plane bus connector



Jumper X3: Supply voltage connected to lower back plane bus connector



Do not exceed the ratings of the power supply!

Overview of 5V and 24V power supply connections

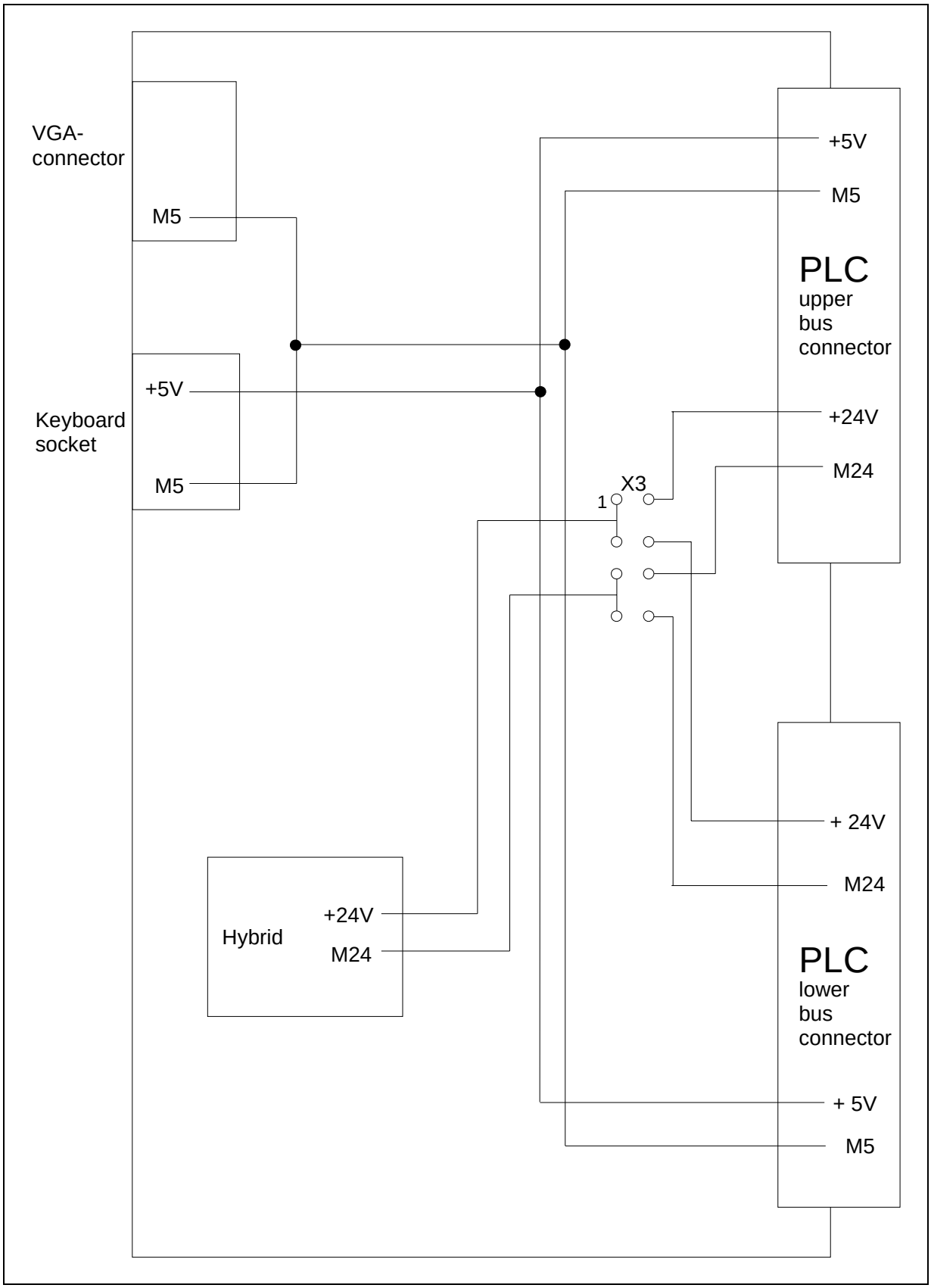


Fig. 2-7: 5V and 24V power supply connections

2.2.6 Installation of the mathematical co-processor (FPU)

The mathematical co-processor is installed by inserting the IC into the respective socket (IC19) (Attention: ensure proper orientation of the IC). The IC should only be installed by VIPA or properly trained technicians.

(Ensure compliance with the regulations for handling electrostatically sensitive components).

When the system is started the mathematical co-processor is detected automatically. Please refer to the manual or the diskette if you require more detailed information.

It is likely that the utilisation of the mathematical co-processor requires configuration of the application software (see application software documentation).

2.2.7 Plug-in locations for the BG86 in the PLC

The following schematic shows possible plug-in locations (indicated by an x) for the BG86 in the different PLC module racks.

Plug-in locations in the PLC-115U

Plug-in locations	PS	CPU	0	1	2	3	4	5	6	IM
Power supply module	X									
Central module		X								
VIPA-BG86 (in the CR 700-1)			X							
VIPA-BG86 (in the CR 700-2)			X	X	X	X	X	X		
VIPA-BG86 (in the CR 700-3)			X X	X X	X X	X	X	X		

In the PLC-115U the BG86 must be installed in an enclosed adapter.

Plug-in locations in the PLC-135U

Plug-in locations	3	11	19	27	35	43	51	59	67	75	83	91	99	107	115	123	131	139	147	155	163
Co-ordinator	X																				
R-, S-, M-processor		X	X	X	X																
Communication processors		X	X	X	X	X	X	X	X												
VIPA-BG86			X	X	X	X	X	X	X												

Plug-in locations in the PLC-155U

Plug-in locations	3	11	19	27	35	43	51	59	67	75	83	91	99	107	115	123	131	139	147	155	163
Co-ordinator	X																				
Central modules		X	X	X																	
Communication processors					X	X	X	X	X	X	X	X	X	X	X	X	X				
VIPA-BG86					X	X	X	X	X	X	X	X	X	X	X	X	X				

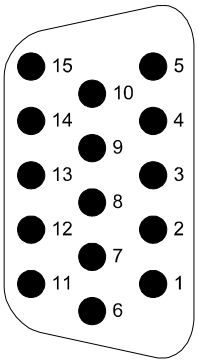
Plug-in locations in the EG-185U

Plug-in locations	3	11	19	27	35	43	51	59	67	75	83	91	99	107	115	123	131	139	147	155	163
Connecting units 300																					X
Connecting units 314, 318	X																				
VIPA-BG86			X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X			

2.3 Pin-assignment of sockets and plugs

The figures show the respective plugs or sockets when the module is located in the module rack. The top of the figure corresponds with the top of the plug.

2.3.1 D-type socket for connection to a monitor (15-pin)

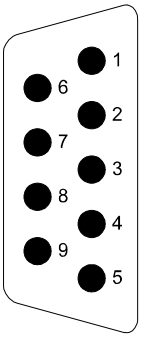


Pin-no.	VGA-colour		VGA-mono
1	Red		-
2	Green		Video
3	Blue		-
4	not connected		
5	GND		
6	GND red		
7	GND green		
8	GND blue		
9	n.c.		
10	GND Sync		
11	not connected		
11	not connected		
13	Hsync		
14	Vsync		
15	n.c.		

Tab. 2-3: D-type socket for connection to a monitor

The 15 pin high-density D-type socket is used to connect a monitor to the module. The socket has the same pin-assignment as a standard VGA adapter. Monitors with other pin-assignments require a special interface cable.

2.3.2 D-type plug with RS232C-interface (9-pin)



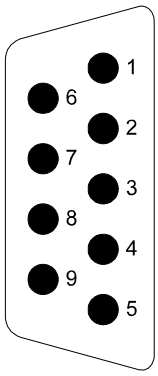
Pin-no.	Signal
1	DCD-
2	RXD
3	TXD
4	DTR-
5	GND
6	DSR-
7	RTS-
8	CTS-
9	RI-

Tab. 2-4: D-type plug with RS232C interface

The interface is connected to both types of handshake signal (RTS/CTS and DTR/DSR). The actual type of signal that is used depends on the wiring and the programming of the interface.

2.3.3 D-type plug with 20mA interface (9-pin)

The mode of this interface may be set to operate as active or passive 20mA interface. For the active mode the module must be connected to 24V DC. The 24Volt supply is configured by means of jumpers X3. (see chapter 2.2.5)



Pin-no.	Signal
1	n.c.
2	TXD+
3	S1+
4	RXD+
5	S2+
6	TXD-
7	S1-
8	RXD-
9	S2-

Tab. 2-5: D-type plug with 20mA interface

Passive 20mA interface:

Only 4 signals of the interface are used When the interface is used in passive mode.

Active 20mA interface:

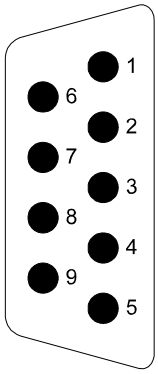
In active mode the S-signals are connected to power sources that must be connected to the data lines as required. In this case the module must be connected to a 24V-DC supply.

2.3.4 9 pin D-type plug with RS422/485 interface

This connector can be used as an RS422 interface, i.e. a point-to-point link with separate transmit and receive lines as well as an RS485 interface for a bus system with transmission and reception via the same line. In this case a bus-master controls the operating mode by means of the SEL-signal.

In each case TxD-lines and RxD-lines must be connected by means of screened twisted pair lines.

2.3.4.1 RS422 interface operation

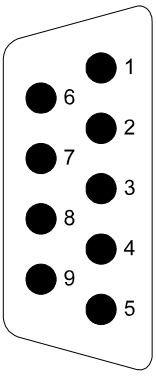


Pin-no.	Signal
1	n.c.
2	TXD+
3	n.c.
4	RXD+
5	n.c.
6	TXD-
7	n.c.
8	RXD-
9	n.c.

Tab. 2-6: RS422 interface operation

Pins shown as "n.c." are not used by the RS422 mode of operation. These pins must not be connected!

2.3.4.2 RS485 interface operation



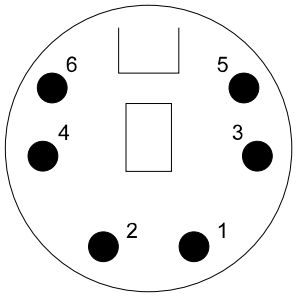
Pin-no.	Signal
1	n.c.
2	n.c.
3	SEL
4	RXD+/TXD+
5	SEL
6	n.c.
7	DTR
8	RXD-/TXD-
9	RTS

Tab. 2-7: RS485 interface operation

In RS485 mode only a single 2-wire line is connected to the plug. The SEL-signal is used to switch between transmit and receive operation. A logical 1 (5V) selects transmit mode and a logical 0 (0V) selects receive mode. The SEL-signal may be controlled via the DTR, the RTS or an external signal. This signal is selected by means of a link in the plug.

Pins shown as "n.c." are not used by the RS422 mode of operation. These pins must not be connected!

2.3.5 Mini-DIN-socket for keyboards (6-pin)



Pin-no.	Signal	Description
1	KBDATA	
2	n.c.	not connected
3	GND	Ground
4	+5V	Supply voltage for keyboard
5	KBCLK	

Tab. 2-8: Mini-DIN-socket for keyboard

Note:

AT-type keyboards with automatic selection of XT- and AT-mode are not compatible with the BG86. This type of keyboard will not be recognised and return a keyboard error. Where possible you should set these keyboards to AT-mode.

2.3.6 Two-part AT-Bus sockets

2.3.6.1 X13

B	A	
GND	SA0	31
OSC	SA1	30
+5V	SA2	29
BALE	SA3	28
T/C	SA4	27
DACK2-	SA5	26
IRQ3	SA6	25
IRQ4	SA7	24
IRQ5	SA8	23
IRQ6	SA9	22
IRQ7	SA10	21
CLK	SA11	20
RFSH	SA12	19
DRQ1	SA13	18
DACK1	SA14	17
DRQ3	SA15	16
DACK3	SA16	15
IOR-	SA17	14
IOW-	SA18	13
SMEMR-	SA19	12
SMEMW-	AEN	11
GND	IOCHRDY	10
n. c.	SD0	9
0WS	SD1	8
n. c.	SD2	7
DRQ2	SD3	6
n. c.	SD4	5
IRQ9	SD5	4
+5V	SD6	3
RESET	SD7	2
GND	IOCHK	1

2.3.6.2 X7

D	C	
GND	SD15	18
MASTER-	SD14	17
+5V	SD13	16
DRQ7	SD12	15
DACK7-	SD11	14
DRQ6	SD10	13
DACK6-	SD9	12
DRQ5	SD8	11
DACK5-	MEMW-	10
DRQ0	MEMR-	9
DACK0-	LA17	8
IRQ14	LA18	7
IRQ15	LA19	6
IRQ12	LA20	5
IRQ11	LA21	4
IRQ10	LA22	3
IOCS16-	LA23	2
MEMCS16-	SBHE	1

2.3.7 PLC base connector (48-pin male multipoint connector)

2.3.7.1 Base connector X1

d	b	z	
nc	M	+5V	2
UBAT	PESP	nc	4
ADB12	ADB00	/CPKL	6
ADB13	ADB01	/MEMR	8
ADB14	ADB02	/MEMW	10
ADB15	ADB03	/RDY	12
IRA	ADB04	DBO	14
IRB	ADB05	DB1	16
IRC	ADB06	DB2	18
IRD	ADB07	DB3	20
/BAU115	ADB08	DB4	22
/NAU115	ADB09	DB5	24
nc	ADB10	DB6	26
DSI	ADB11	DB7	28
+24V	BASP	M24V	30
nc	M	nc	32

2.3.7.2 Base connector X2

d	b	z	
nc	M	+5V	2
nc	nc	nc	4
nc	nc	nc	6
nc	nc	nc	8
nc	nc	nc	10
nc	nc	nc	12
nc	nc	/NAU	14
nc	nc	/BAU	16
nc	nc	nc	18
nc	nc	nc	20
/TxDSN	nc	nc	22
nc	nc	nc	24
nc	/RxDSN	nc	26
nc	nc	nc	28
nc	nc	M24V	30
nc	M	+24V	32

2.3.8 Socket for interface modules, X15

DCD	1		2	+5V
SubD1	3		4	+24V
SubD6	5		6	DSR
SubD2	7		8	DTR
SubD7	9		10	RxD
SubD3	11		12	CTS-
SubD8	13		14	RTS-
SubD4	15		16	TxD
SubD9	17		18	M24
SubD5	19		20	GND
RI	21			

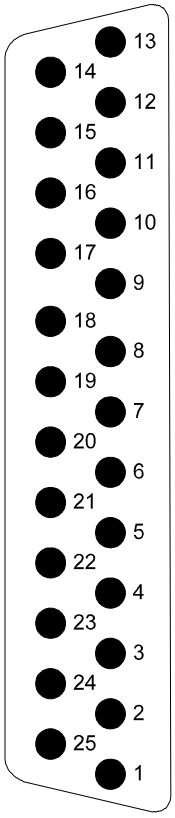
2.3.9 Socket (X8) with IDE-interface for hard disk (50-pin)

2	-	-	1
4	-	-	3
6	-	-	5
8	GND	RESET-	7
10	D8	ID7	9
12	D9	D6	11
14	D10	D5	13
16	D11	D4	15
18	D12	D3	17
20	D13	D2	19
22	D14	D1	21
24	D15	D0	23
26	-	GND	25
28	GND	-	27
30	GND	IOW-	29
32	GND	IOR-	31
34	ALE	-	33
36	GND	-	35
38	IOCS16-	INT	37
40	-	A1	39
42	A2	A0	41
44	CS1-	CS0-	43
46	GND	LED-	45
48	+5V	+5V	47
50	+5V	GND	49

2.3.10 Male plug X10 (26-pin)

26	-	SLCT	25
24	GND	PE	23
22	GND	BUSY	21
20	GND	ACK-	19
18	GND	PD	17
16	GND	PD6	15
14	GND	PD5	13
12	GND	PD4	11
10	GND	PD3	9
8	SLCTIN-	PD2	7
6	INIT-	PD1	5
4	ERR-	PD0	3
2	AFD-	STB-	1

This interface can be brought out to the front panel by means of an optional cable. In this case a 25-pin D-type socket provides the LPT1-CENTRONICS interface with the following assignment:



Pin-no.	Signal	Pin-no.	Signal
1	STB-	14	/AFD
2	PD0	15	/ERR
3	PD1	16	/INIT
4	PD2	17	/SLCTIN
5	PD3	18	GND
6	PD4	19	GND
7	PD5	20	GND
8	PD6	21	GND
9	PD7	22	GND
10	/ACK	23	GND
11	BUSY	24	GND
12	PE	25	GND
13	SLCT		

Tab. 2-9: Pin-assignment Centronics interface

2.4 BIOS-description and system programming

2.4.1 System organisation

The BG86 contains the firmware that provides the basic functions for the available hardware - called BIOS. The BG86 BIOS has many parts:

QUADTEL-BIOS:	This section contains the usual interfacing functions for the standard components of a compatible AT-type PC.
VIPA-BIOS:	VIPA has provided additional functions: these consist of theinitialisation and control of the hardware interface for the PLC, additional drivers for the silicon-disk and the memory-card including boot functions and otherfunctions to ensure that the module starts properly if the system components configuration should have been lost.
Cirrus-Logic-VGA-BIOS:	This BIOS contains the control functions for the display.

Some BIOS parameters may be customised by means of a SETUP function provided by the firmware. The default system-setup is pre-defined by VIPA in accordance with the system configuration and it is maintained for app. 1500 hours by a rechargeable battery when the module is disconnected from the power supply. The module uses a set of default settings to allow the module to start properly when the above settings are lost due to a discharged battery.

The user can customise the system configuration by means of the SETUP function.

2.4.2 BIOS-SETUP

When the system is turned on the monitor screen will display the BIOS-version. After this the BIOS tests the system components followed by a memory test. When the tests have been completed the system attempts to boot from drive A, if available, and then from drive C if this is available.

The SETUP function is only accessible while the system starts by depressing the <F2>-key. This key may be pressed when the respective request is displayed on screen.

2.4.2.1 VIPA-BIOS-SETUP

At first the VIPA-SETUP is displayed with the serial number of the module (hexadecimal) and fields for the password, the page frame configuration and the silicon-disk selection. This window also displays a warning if the system configuration was lost (battery discharged). Information displayed at the bottom of the screen show the keys that can be used for the SETUP function:

```
                    VIPA-Setup für Kachelinterface und Silikondisk V0.3
-------------------------------------------------------------------------------------------

                     Seriennummer der Baugruppe:  0305



       Paßwort ändern:

       Aktuelle Kachelnummer:    32-39

       Laufwerk A:             FLOPPY-Disk





-------------------------------------------------------------------------------------------
  ↑↓ Cursor bewegen     F5,F6  auswählen      F10 übernehmen      ESC Setup verlassen
```

Translation:

Seriennummer der Baugruppe = Module serial number

Paßwort ändern = Change password

Aktuelle Kachelnummer = Current page frame number

Laufwerk A = Drive A

Cursor bewegen = Cursor control

auswählen = select

übernehmen = accept

Setup verlassen = Quit from setup

You can enter a password into the **Passwort field** of the setup screen (8 ASCII characters). A blank SETUP field indicates that no password has been entered. In this case you may access the other setup fields. You can enter a password by typing sequence of up to 8 ASCII characters into the password field followed by the Enter key. When you hve entered a password the colour of the password field is changed from a white field to black. In this case the user can only access the setup functions to change the password or other system parameters in the SETUP windows after he has entered the valid password. The password is deleted if the user should press the Enter key in the password field without entering any ASCII characters.

In the **Page-frame field** the base address of the PLC page-frame can be entered from 0 ... 248 in steps of 8. The selected area is activated immediately when the BG86 starts and it is then accessible to the PLC.

The user can specify other drive types (e.g. silicon-disk) as boot drives for drive A in the SETUP-field "**Drive A**". To enable this setting you must make a selection for the diskette drive A in the second part of the SETUP.

You can install different silicon-disks on the BG86 and the chip-type silicon-disk as well as the memory-card silicon-disk are directly available on the module (optional):

- The chip-type silicon-disk (IC3, IC4) is set to address C00000 hex in hardware. These ICs can consist of SRAM's, EPROM's or PEROM's.
- Die memory-card silicon-disk (memory-card socket) is set to hardware address 800000 hex. SRAM-cards and OTP-cards are available as memory-cards.
- Additional silicon-disks are available in the form of plug-in adapters. The respective addresses can be defined on the adapter.

One of these silicon-disks may be designates as boot drive. The user must define the address and the type of memory - RAM or ROM - for the respective boot-drive:

- ROM at base address 80 0000 hex	Silicon-disk board ROM
- RAM at base address 80 0000 hex	Silicon-disk board ROM
- ROM at base address C0 0000 hex	Silicon-disk ROM
- RAM at base address C0 0000 hex	Silicon-disk ROM

Example:	VIPA-SETUP	Quadtel-SETUP
Boot from 1.44MB-diskette	diskette	1.44MB
The module does not support booting from diskettes.		
Boot from EPROM-chip silicon-disk	ROM at C00000	1.44MB
Boot from SRAM-memory-card silicon-disk	RAM at 800000	1,44MB

A diskette drive is not installed. For the silicon-disk operation disk drive 0 must be entered as a 1.44MB!

It is, however, necessary that a bootable silicon-disk is available when the boot-procedure should occur from the silicon-disk. Settings are saved when you quit from the SETUP-window by means of the <ESC>-key.

The password is saved immediately when the enter-key is pressed.

2.4.2.2 QUADTEL-BIOS selection menu

The following menu is displayed when you quit from the VIPA-BIOS setup:

```
      Extended BIOS Software
 Copyright 1989-92 Quadtel Corp.
 ____________________________________
     Setup
     Extended BIOS Features
     System Information
     System Security
     Park Fixed Disks
     Diagnostics
     Format Fixed Disk
 ----------------------------------
 ↑↓  Move  Enter Select  Esc Exit
```

System settings in the different windows presented by "Setup" and "Extended BIOS-Features" should only be attempted by advanced users. These settings do not normally require changes. The menu item "System Information" returns information about the QUADTEL-BIOS. A second boot-protection password can be defined in the menu item "System Security". If you should enter a password into this location the system will only boot when the respective password has been entered. Menu item "Park Fixed Disks" is not required by the specific hard drive used in the system. Menu item "Diagnostics" provides a number of tests for system components. Menu item "Format Fixed Disk" provides an option to format the hard disk. This function must never be used on the hard disks supplied with the system.

2.4.2.3 QUADTEL-BIOS-Setup

This **BIOS Setup** provides the following options (the figure shows default settings):

```
      Extended BIOS Setup - Copyright 1989,90,91 Quadtel Corporation
____________________________________________________________________________

  Current Date: [09/14/1995]          Video System:   [EGA / VGA  ]
  Current Time: [11:22:10]

  [  640K] System Memory              Power Up Speed: [Fast  ]
  [ 3072K] Extended Memory            BIOS Shadow:    [System in RAM]
      96K  Shadow Memory                              [Video in RAM ]
  [  288K] EMS Memory                 Wait States:    [0, All Banks ]

  Internal Floppy:  [Enabled      ]   Internal COM A: [Off       ]
  Internal IDE:     [Enabled      ]   Internal COM B: [Off       ]
                                        Internal LPT:   [Off       ]
  Diskette Drive 0: [1.44 MB, 3 1/2]
  Diskette Drive 1: [Not Installed ]

  Fixed Disk 0: Type:[User] CY:[ 705] HD:[14] ST:[17] LZ:[   0] WP:[   0]
  Fixed Disk 1: Type:[None]
----------------------------------------------------------------------------
  ↑↓   Move        F5  Previous Value      F9   Automatic Configuration
  F1   Help        F6  Next Value          F10  Save Configuration
  Esc  Exit
____________________________________________________________________________
```

The following system parameters can be changed by means of the BIOS-Setup:

- Date
- Time
- System start up password
- Type of diskette drive 1 (360KB, 720KB, 1.2MB, 1.44MB)
- Type of diskette drive 2 (360KB, 720KB, 1.2MB, 1.44MB)
- Type of hard disk 1 (1 .. 47)
- Type of hard disk 2 (1 .. 47)
- Type of Video-interface
- Size, type and arrangement of the system memory
- Shadow-RAM for system-BIOS
- Shadow-RAM for VGA-BIOS

Setup parameters should only be changed by advanced users. It is normally not necessary to change the system setup.

The control keys for the system setup are shown at the bottom of the screen. The <F1>-key returns help informat. The <F10>-key saves the current configuration and the <ESC>-key can be used to quit from the BIOS-Setup

2.4.2.4 Extended BIOS-Features

The **Extended BIOS** offers the following additional features (the figure shows default settings):

```
    Extended BIOS Features - Copyright 1989-92 Quadtel Corp.
____________________________________________________________________
      Auto-park Disk: [No ]         Keyboard Click: [Yes     ]
          Quick Boot: [No ]         Keyboard Delay: [1/4 Sec ]
        Screen Saver: [Disabled]    Keyboard Rate : [30/Sec  ]
                                  Numlock Boot State: [Auto    ]
--------------------------------------------------------------------
 ↑↓    Move     F5  Previous Value   F9   Auto Configuration
 Esc   Exit     F6  Next Value       F10  Save Configuration
```

You can enter additional system parameters into the Extended BIOS Features window. These setup parameters should only be changed by advanced users. It is not normally necessary to change these settings.

The control keys for the system setup are shown at the bottom of the screen. The <F1>-key returns help information. The <F10>-key saves the current configuration and the <ESC>-key can be used to quit from the BIOS-Setup.

2.4.2.5 AT ROM Diagnostics

You can test nearly all the system components by means of the integrated **AT ROM Diagnostics**:

```
AT ROM Diagnostics V3.11 - Copyright 1989,90,91 Quadtel Corp.
_____________________________________________________________________________

   Continuous: [No ]   Stop on error: [Yes]   Echo log to LPT1: [No ]
_____________________________________________________________________________


          [P] System Board                 [P] Monochrome Parallel
        [101] Keyboard Controller          [N] Primary Parallel
     [  640K] System Memory                [N] Secondary Parallel
     [ 3072K] Extended Memory              [P] Primary Serial
     [1.44 M] Diskette Drive 0             [P] Secondary Serial
     [1.44 M] Diskette Drive 1
          [P] Fixed Disk 0
          [N] Fixed Disk 1


-----------------------------------------------------------------------------
   ↑↓   Move        F5  Previous Value      F9   Test Present Devices
   F1   Help        F6  Next Value          F10  Test Selected Device
   Esc  Exit
_____________________________________________________________________________
```

You can test individual system components by means of the menu item "Diagnostics". The control keys for the system setup are shown at the bottom of the screen. The <F1>-key returns help information. The <ESC>-key can be used to quit from the diagnostic window.

2.4.2.6 CMOS-RAM-Statusbyte (CMOS-RAM address 4E)

The CMOS-RAM contains additional memory areas between 40 and 4F. VIPA utilises these addresses for BIOS extensions and for the VIPA-SETUP and they are protected by a checksum. Users must not change the contents of these RAM-locations.

One exception is byte 4E - the VIPA-CMOS-Statusbyte. This byte has the following assignments:

Bit 0	0	System date and system time are OK
	1	System date and system time were lost
Bit 1	0	System configuration is OK
	1	System configuration was lost, default table was loaded
Bit	2-6	not used
Bit 7	0	must be set to 0

Procedure:

The VIPA-BIOS extensions for the BG86 ensure that the module operates properly even if the system configuration was lost (battery discharged). The correct system configuration can be loaded under DOS. Only the date and time must be supplied by the user if the configuration was lost. Bit 0 in the VIPA-CMOS-Status-register is used for this purpose. The user can interrogate or set this bit from an application program.

The user may set this bit to 1 when he has corrected the date and time settings. This byte is not included in the checksum.

2.4.2.7 ROM-SETUP

The VIPA-BIOS has facilities for a ROM-SETUP. In this case the CMOS-RAM is only used to buffer date and time. The system setup data is saved in the system-EPROM.

The table for the ROM-SETUP is located in BIOS-EPROM at addresses

F000:7800 through F000:784E (hex). This table contains the standard SETUP-table which is used for recovery if the battery should fail.

A control flag with the functions below is located at address F000:784F (hex)

FF When the system boots the SETUP is read from CMOS-RAM. The ROM SETUP and the DIP-switch S2 are only used if the battery should have failed

00 The system will always use the ROM-SETUP when it boots. Changes to the SETUP are only possible by exchanging the BIOS-EPROM's.

(Please submit your custom setup enquiry to VIPA)

2.4.3 Address assignments, interrupts and DMA-channels

2.4.3.1 Memory-address assignments

Addressing range	Size	Description
000000 - 09FFFF	640KB	DRAM main memory
0A0000 - 0BFFFF	128KB	Display memory
0C0000 - 0C7FFF	32KB	VGA-BIOS
0C8000 - 0CFFFF	32KB	PLC-interface data area
0D0000 - 0D0FFF	4KB	SERCOS chip DPR
0D1000 - 0DFFFF	60KB	PCMCIA memory window
0E0000 - 0E7FFF	92KB	unused memory (for instance EMS)
0E8000 - 0EFFFF	32KB	Extension BIOS, VIPA setup, utilities, drivers
0F0000 - 0FFFFF	64KB	BIOS
100000 - 3FFFFF	4MB	Expanded memory (main memory)
800000 - 8FFFFF	1MB	ROMDisk 1
900000 - 9FFFFF	1MB	ROMDisk 2
A00000 - AFFFFF	1MB	ROMDisk 3
C00000 - CFFFFF	1MB	RAMDisk 1

Tab. 2-10: Memory-address assignments

2.4.3.2 I/O-Address assignment

Addressing range	Description
000-01F	DMA controller 1
020-03F	Interrupt controller 1
040-05F	Timer 8254
060	Keyboard controller
061	Port B register, PPI, 8255
062-06F	Keyboard controller
070-07F	Real-time clock and CMOS-RAM, NMI-mask
080-08F	DMA Page register
090-091	DMA Map register
092	Alternate Gate A20 and Hot Reset
093-09F	DMA Map register
0A0-0BF	Interrupt controller 2
0C0-0DF	DMA controller 2
0F0	Clear Math Coprocessor Busy
0F1	Reset Math Coprocessor
0F8-0FF	Math Coprocessor
102-104	VGA Controller
1EC-1EF	EMS- and Control-Register
1F0-1F8	Hard disk
270-277	Control circuitry and Watchdog
280	VIPA-CP Status register
281	VIPA PLC Status register
282	VIPA-control register
284	7 Segment display
285	Hardware Identifikation register (serial EPROM)
288-28F	VIPA PLC Interface register (see bus-interface, chap.2.5)
2F8-2FF	COM2
378-37B	LPT1 (available only after appropriate configuration of the combo)
3B0-3BB	VGA-controller
3BC-3BF	LPT1
3C0-3DF	VGA-controller
3F0-3F7	FD-controller
3F8-3FF	COM1
46E8	VGA controller

Tab. 2-11: I/O address assignments

2.4.3.2.1 Description of the VIPA Status registers (280)

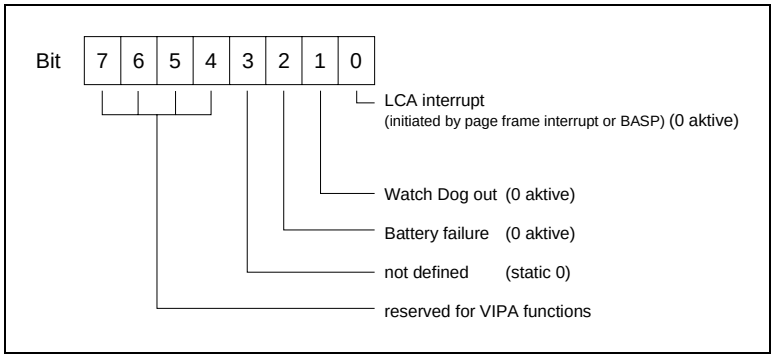


Fig. 2-8: VIPA Status register (280)

2.4.3.2.2 Description VIPA PLC Status register (281)

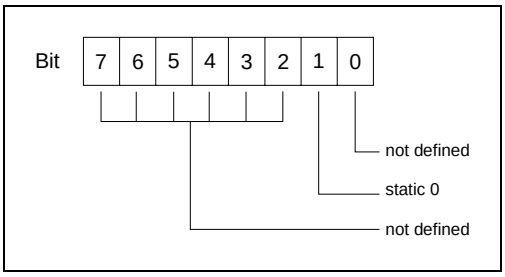


Fig. 2-9: VIPA Status register (281)

2.4.3.2.3 Description VIPA control register (282)

When the system has booted and after a reset the watchdog is turned off and it can be turned on and off again by means of software.

After power-on the watchdog must be triggered at a rate that is less than once every 1,6 seconds. A programmed transition is used for triggering. If the watchdog timer is not retriggered within the 1,6 seconds it will initiate a system reset (depending on the status of the CONTROL-register bit 0). The state of the watchdog in the STATUS register can be interrogated by means of software.

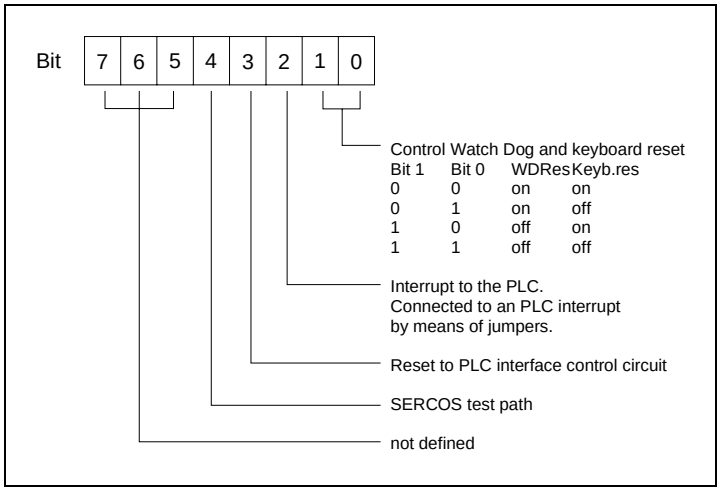
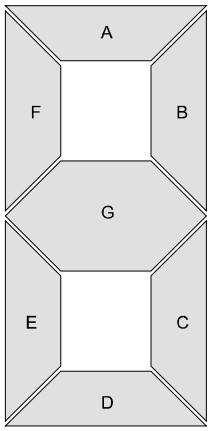


Fig. 2-10: VIPA control register (282)

2.4.3.2.4 Description of 7-segment display (284)



Bit 0 Segment A
Bit 1 Segment B
Bit 2 Segment C
Bit 3 Segment D
Bit 4 Segment E
Bit 5 Segment F
Bit 6 Segment G
Bit 7 Decimal point
0: Segment off
1: Segment on

Fig. 2-11: Description 7 segment display (284)

2.4.3.3 Interrupt assignments

Interrupt	System	Description
IRQ0	08	Timer Output 0
IRQ1	09	Keyboard (output buffer full)
IRQ2	0A	Interrupt from interrupt controller 2 (IRQ8-IRQ15)
IRQ3	0B	COM2
IRQ4	0C	COM1
IRQ5	0D	unused
IRQ6	0E	FD-controller
IRQ7	0F	LPT1
IRQ8	70	Real-time clock
IRQ9	71	SERCOS interrupt 1
IRQ10	72	PCMCIA-Socket
IRQ11	73	SERCOS interrupt 0
IRQ12	74	PLC interrupt (page frame interrupt and BASP-interrupt)
IRQ13	75	Co-processor
IRQ14	76	Hard disk
IRQ15	77	unused

Tab. 2-12: Interrupt assignments

Example for the interrogation of an interrupt on the module:
a hardware interrupt IRQ12 (software interrupt 74) can be triggered on the BG86 by

- an PLC access to element 1023 of the respective active page frame
- BASP activation on the back plane bus (depending on DIP-switch S1/3)

The respective interrupt-vector is located at address 1D0 - 1D3 (hex). The interrupt must be cleared in the interrupt-service-routine by means of an access to the interrupt-reset-register (LCA-register)

2.4.3.4 DMA-channel assignments

Channel	Description
CH0	-
CH1	-
CH2	FD-controller
CH3	-
CH4	Cascade for controller 1
CH5	-
CH6	Reserved for additional VIPA board
CH7	-

Tab. 2-13: DMA channel assignments

2.5 Bus-interface V.10 for PLC page frame operation

The PLC page frame interface consists of a dual port RAM structure with a memory size of 16 KB. This RAM is accessible to both sides. Accesses are controlled by a LCA (logic cell array) located on the SERCOS-module. The LCA may be configured to operate in common page frame mode (16 page frames of 1 KB) or in linear addressing mode, i.e. the PLC may access RAM in common page frame mode, in linear mode or in the peripheral area. The SERCOS-module may access RAM in the memory addressing area.

2.5.1 PLC-interface LCA registers

The PC can access the PLC-interface LCA registers at I/O addresses 288h-28Fh. PLC access is restricted to the common page frame mode under PLC-address FEFF (page frame select register).

Name	Access	Description
RAM size register	PC R/W bit 0-3 I/O address: 288	Defines the RAM size for the back plane bus. Value RAM size 0000 1K (default) 0001 2K 0011 4K 0111 8K 1111 16K
RAM area select register	PC R/W bit 0-3 I/O address: 289 PLC W bit 0-3 address: FEFF	Defines the RAM addressing area specifying the PC address where the data is saved. Memory address area: C800:0 - C800:3FFF Default access: PLC addr. FEFF
PLC start address register	PC R/W bit 0-7 I/O address: 28A	Depending on the size of the linear address area this defines the start address in the PLC addressing area where the PLC RAM should be inserted. To define the PLC start address only the most significant 6 bits of the start address are required. The following are available: size linear addr. Area setting step size 1 KB 0000 - FC00 400 2 KB 0000 - F800 800 4 KB 0000 - F000 1000 8 KB 0000 - E000 2000 16 KB 0000 - C000 4000 Default value: F4(hex)
Page frame bank select register	PC R/W bit 0-3 I/O address: 28B PLC W bit 4-7 address: FEFF	In page frame mode the PLC enters the page frame number into the page frame select register FEFF. If the value of the page frame bank select register matches the value of the page frame bank compare register the PLC RAM is available as a page frame. The addressing range where the PC can access the data of the page frame RAM is also defined by the PLC page frame select register in the RAM area select register. Default value: 0000
Name	Access	Description

Page frame bank compare register	PC R/W bit 0-3 I/O address: 28C	Specifies the page frame bank to which the PC can react as page frame RAM. (see Description page frame bank select register). Default value: 0010(bin)
PLC status register	PC R/W bit 0-3 I/O address 28D	Status register : page frame function Value Function Bit 1 Bit 0 0 0 conventional page frame operation 0 1 reserved 1 0 linear addressing 1 1 reserved Status register: interrupt Value Interrupt Bit 3 Bit 2 BASP Page frame 0 0 enabled enabled 0 1 enabled disabled 1 0 disabled enabled 1 1 disabled disabled Default value: 1000(bin)
Interrupt Request/Status Register	PC R/W bit 0-2 I/O address 28E	This register is used simultaneously as interrupt request and as interrupt status register. The interrupt status is returned with any read access. The interrupt flags are cleared by a write access. Read a register: Value Interrupt status Bit 1 Bit 0 BASP Page frame 0 0 inactive inactive 0 1 inactive active 1 0 active inactive 1 1 active active Bit 2 : BASP status Write a register: Value Interrupt request Bit 1 Bit 0 BASP Page frame 0 0 no change no change 0 1 no change request 1 0 request no change 1 1 request request BASP is not changed by a write access to bit 2.
LCA version number	PC R bit 0-4 I/O address 28F	returns the current version number of the LCA (10)

Tab. 2-14: PLC-Interface LCA register

2.5.2 Setting the LCA to page frame operation

The contents of the two registers must be changed to set the LCA to page frame operation. These registers are the page frame bank compare register (I/O 28C) and the PLC start address register (I/O 28A). After the reset the LCA operates by default under the PLC page frame number 32-47 (page frame bank 2). At this point the page frame interrupt is active and the BASP interrupt is masked.

Page frame bank	PLC page frame number	Page frame bank compare register (28C)	PLC start address register (28A)
0	0-15	0	F4
1	16-31	1	F4
2	32-47	2	F4
3	48-63	3	F4
4	64-79	4	F4
5	80-95	5	F4
6	96-111	6	F4
7	112-127	7	F4
8	128-143	8	F4
9	144-159	9	F4
10	160-175	A	F4
11	176-191	B	F4
12	192-207	C	F4
13	208-223	D	F4
14	224-239	E	F4
15	240-255	F	F4

Tab. 2-15: Setting the LCA to page frame operation

2.5.3 Dual Port RAM address assignments in page frame operation

Page frame no.	bus address	PC address
n+15	F400-F7FF	0C800:0000-0C800:03FF
n+14	F400-F7FF	0C800:0400-0C800:07FF
n+13	F400-F7FF	0C800:0800-0C800:0BFF
n+12	F400-F7FF	0C800:0C00-0C800:0FFF
n+11	F400-F7FF	0C800:1000-0C800:13FF
n+10	F400-F7FF	0C800:1400-0C800:17FF
n+9	F400-F7FF	0C800:1800-0C800:1BFF
n+8	F400-F7FF	0C800:1C00-0C800:1FFF
n+7	F400-F7FF	0C800:2000-0C800:23FF
n+6	F400-F7FF	0C800:2400-0C800:27FF
n+5	F400-F7FF	0C800:2800-0C800:2BFF
n+4	F400-F7FF	0C800:2C00-0C800:2FFF
n+3	F400-F7FF	0C800:3000-0C800:33FF
n+2	F400-F7FF	0C800:3400-0C800:37FF
n+1	F400-F7FF	0C800:3800-0C800:3BFF
n	F400-F7FF	0C800:3C00-0C800:3FFF

n = 0,16,32,48,64,80,96,112,128,144,160,176,192,208,224,240

Tab. 2-16: Dual Port RAM address assignment in page frame operation

2.5.4 Setting the LCA to operate in linear addressing mode

PLC status register (28D) bit 1 must be set whilst bits 0,2,3 must remain unchanged.

1. The same value must be written into page frame bank select register (28B) and page frame bank compare register (28C). Preferably 0000H.
2. The RAM size register (288) must be set to match the size of the linear addressing area in the PLC. (Refer to register description for values).
3. The PLC start address must be entered into the PLC start address register (28A). It must be noted that the PLC start address can only be changed in relation to the RAM size of the linear addressing area. I.e. the following settings are available for the PLC start address:

RAM size	PLC start address setting
1 KB	0000,0400,0800,0C00,1000,...,F000,F400,F800,FC00
2 KB	0000,0800,1000,1800,2000,...,F000,F800
4 KB	0000,1000,2000,3000,4000,...,E000,F000
8 KB	0000,2000,4000,6000,8000,A000,C000,E000
16 KB	0000,4000,8000,C000

5. Select the addressing area in the page frame RAM by means of the addressing area select register (289). Depending on the size of the linear addressing area the page frame RAM may be divided into different portions.

RAM size	Number of RAM areas
1 KB	16 areas
2 KB	8 areas
4 KB	4 areas
8 KB	2 areas
16 KB	1 areas

2.5.5 Dual Port RAM address assignment in linear addressing mode

RAM size	PLC-address	RAM area	PC-address
1 KB	Startaddr. - Startaddr. + 3FF	0	0C800:0000-0C800:03FF
		1	0C800:0400-0C800:07FF
		2	0C800:0800-0C800:0BFF
		3	0C800:0C00-0C800:0FFF
		4	0C800:1000-0C800:13FF
		5	0C800:1400-0C800:17FF
		6	0C800:1800-0C800:1BFF
		7	0C800:1C00-0C800:1FFF
		8	0C800:2000-0C800:23FF
		9	0C800:2400-0C800:27FF
		10	0C800:2800-0C800:2BFF
		11	0C800:2C00-0C800:2FFF
		12	0C800:3000-0C800:33FF
		13	0C800:3400-0C800:37FF
		14	0C800:3800-0C800:3BFF
		15	0C800:3C00-0C800:3FFF
2 KB	Startaddr. - Startaddr. + 7FF	0,1	0C800:0000-0C800:07FF
		2,3	0C800:0800-0C800:0FFF
		4,5	0C800:1000-0C800:17FF
		6,7	0C800:1800-0C800:1FFF
		8,9	0C800:2000-0C800:27FF
		10,11	0C800:2800-0C800:2FFF
		12,13	0C800:3000-0C800:37FF
		14,15	0C800:3800-0C800:3FFF
4 KB	Startaddr. - Startaddr. + 1000	0,1,2,3	0C800:0000-0C800:0FFF
		4,5,6,7	0C800:1000-0C800:1FFF
		8,9,10,11	0C800:2000-0C800:2FFF
		12,13,14,15	0C800:3000-0C800:3FFF
8 KB	Startaddr. - Startaddr. + 2000	0 - 7	0C800:0000-0C800:1FFF
		8 - 15	0C800:2000-0C800:3FFF
16 KB	Startaddr. - Startaddr. + 4000	0 - 15	0C800:0000-0C800:3FFF

Tab. 2-17: Dual Port RAM address assignment in linear addressing mode

2.5.6 Interrupt processing

The LCA analyses the page frame interrupt (write access to the last page frame address) and the BASP interrupt (inhibit command output). The page frame interrupt is triggered when a write access operation to the last page frame occurs. The BASP interrupt is set when the BASP signal changes state from "RUN" to "STOP". Both interrupts are connected to CP interrupt line 12.

When an interrupt occurs the interrupt status bits indicate which interrupt (page frame or BASP interrupt) was responsible for the event. Interrupts must be acknowledged by means of a write access to the interrupt request/status register which resets the interrupt.

Both interrupts can be masked. After the reset the page frame interrupt is enabled and the BASP interrupt disabled.

The relevant registers for interrupt processing are the PLC status register (addr. 28D, bit 2 and 3 for masking) as well as the interrupt request/status register (addr. 28E, bit 0 and 1 status and request). The function of these registers is described in the LCA register description.

2.5.7 I/O addressing

The LCA can react as an I/O device via the PLC peripheral and link flag area. To ensure that accesses are discriminated the LCA manages a control track for read and write accesses of the PLC in the I/O area. If the respective control bit of the I/O byte is set the LCA acknowledges the PLC request by means of READY and makes the data byte available as a RAM cell. If the control bit is not set the READY response to a PLC request will not be issued. The control bit is bit 0 of the respective CPU address.

IO area	PLC address	CPU address : C800 offset
P-periphery write (128 byte)	F000-F07F	5C00-5C7F
A-periphery write (128 byte)	F080-F0FF	5C80-5CFF
Q-periphery write (256 byte)	F100-F1FF	5D00-5DFF
Link flag write (512 byte)	F200-F3FF	5E00-5FFF

Tab. 2-18: Control track address map (WRITE access)

IO area	PLC address	CPU address : C800 offset
P-periphery read (128 byte)	F000-F07F	4C00-4C7F
A-periphery-read (128 byte)	F080-F0FF	4C80-4CFF
Q-periphery-read (256 Byte)	F100-F1FF	4D00-4DFF
Link flag read (512 Byte)	F200-F3FF	4E00-4FFF

Tab. 2-19: Control track address map (READ access)

IO area	PLC address	CPU address : C800 offset
P periphery write (128 byte)	F000-F07F	7C00-7C7F
A periphery-write (128 byte)	F080-F0FF	7C80-7CFF
Q periphery write (256 Byte)	F100-F1FF	7D00-7DFF
Link flag write (512 Byte)	F200-F3FF	7E00-7FFF

Tab. 2-20: Data byte address map (WRITE access)

IO area	PLC address	CPU address : C800 offset
P periphery read (128 Byte)	F000-F07F	6C00-6C7F
A-periphery read (128 Byte)	F080-F0FF	6C80-6CFF
Q-periphery read (256 Byte)	F100-F1FF	6D00-6DFF
Link flag read (512 Byte)	F200-F3FF	6E00-6FFF

Tab. 2-21: Data byte address map (READ access)

Please note

I/O addressing is not active if linear addressing is selected as the main operating mode of the LCA, as the I/O area of the PLC can not be used.

If the main operating mode of the LCA is set to page frame mode the IO addressing is active.

2.5.7.1 Example for I/O addressing

You want to write and read periphery bytes 0 and 1.

You must complete the following steps for this purpose:

- From the PC you must set the control bits for read and write accesses

  WRITE control track: PC address C800:5C00 value 1 (periphery byte 0)
  PC address C800:C5C01 value 1 (periphery byte 1)

  READ control track: PC address C800:4C00 value 1 (periphery byte 0)
  PC address C800:4C01 value 1 (periphery byte 1)

- All other control bits must be set to 0 to avoid duplicate addressing.

The module will acknowledge the request via periphery bytes 0 and 1 and make the following RAM cells available to the PLC as well as the PC:

periphery byte 0 write access: PC address C800:7C00
periphery byte 1 write access: PC address C800:7C01
periphery byte 0 read access: PC address C800:6C00
periphery byte 1 read access: PC address C800:6C01

2.5.7.2 PLC-Dual Port RAM interface address map

PC base address: C800h

Offset	Contents
7FFF-7C00	I/O data bytes write access (1 KB)
7BFF-7000	Unused data area (only accessible to the PC, 3 KB)
6FFF-6C00	IO data bytes read access (1 KB)
6BFF-6000	Unused data area (only accessible to the PC, 3 KB)
5FFF-5C00	I/O control track for write access (1 KB) only the least significant bit is used
5BFF-5000	Unused data area (only accessible to the PC, 3 KB)
4FFF-4C00	I/O control track for read access (1 KB) only the least significant bit is used
4BFF-4000	Unused data area (only accessible to the PC, 3 KB)
3FFF-0000	Page frame area or linear addressing (16 KB)

Tab. 2-22: PLC-Dual Port RAM interface address map

3 SERCOS Programming Interface

3 SERCOS Programming Interface

3.1 General

3.1.1 SERC-Master with programming interface

The SERC-Master software provides data communication interfaces between the PLC, digital drives and decentralised peripherals (slaves). You may connect up to eight slaves to a fibre optic ring (FOC). The maximum number of drives that are controlled via a fibre optic ring depends on the cycle time and the volume of data. The data communication interface between the PLC and the SERCOS-module SERC is provided by the DPR (Dual-Ported-RAM). Data consists of configuration data, nominal values, actual values, status messages for cyclic operation and data for the service channel transfer.

Function summary

PLC

Initiate phase run-up
Set operating mode (turn drives on, turn drives off, enabled, stop)
Initiate a change of operating mode
Transmission of nominal values for the cyclic telegram
Reception of actual values from the cyclic telegram
Reception of status and error conditions
Read or write of ident numbers
Initiate or clear certain commands
Deactivate drives

SERCOS-module

Prepare the ring-configuration and transfer it to the PLC (the ring configuration consists of the manufacturer, drive address, operating modes, telegram types...)
Control of the phase run-up
Control change of operating mode
Reception of status and error conditions and transfer to the PLC
Data communication via service channel
Read or write ident numbers independently or via the PLC
Execution of commands
Diagnostic functions for the display of errors and operating modes
Deactivate drives
Display of telegram data from the cyclic data communications
Control of telegram data
Defining drives (singel-step operation)

Please refer to the online help function for a detailed description of the SERC-Master software (a manual is being prepared).

The SERC-Master software is a system that is controlled via an SAA user interface. The foreground process provides the functions shown above. These are augmented by interrupt functions that may be accessed by the application programmer. You receive the SERC-Master software in the form of a library together with C-language source files consisting of function shells where you can insert your application program. Therefore these empty functions are your programming interface an they are called Motion-Control (abbreviated MC).

For this purpose the SERC-Master software is available in the following form:

SERC.EXE program as a library (.lib) containing the functions described above;

C-language files with function bodies for the programming interface as source files (.c) and header-files (.h).

Fig. 3-1 shows function unit MC embedded into the SERC-Master software. The buffers shown in the Motion-Control-Box are available to each and every slave.

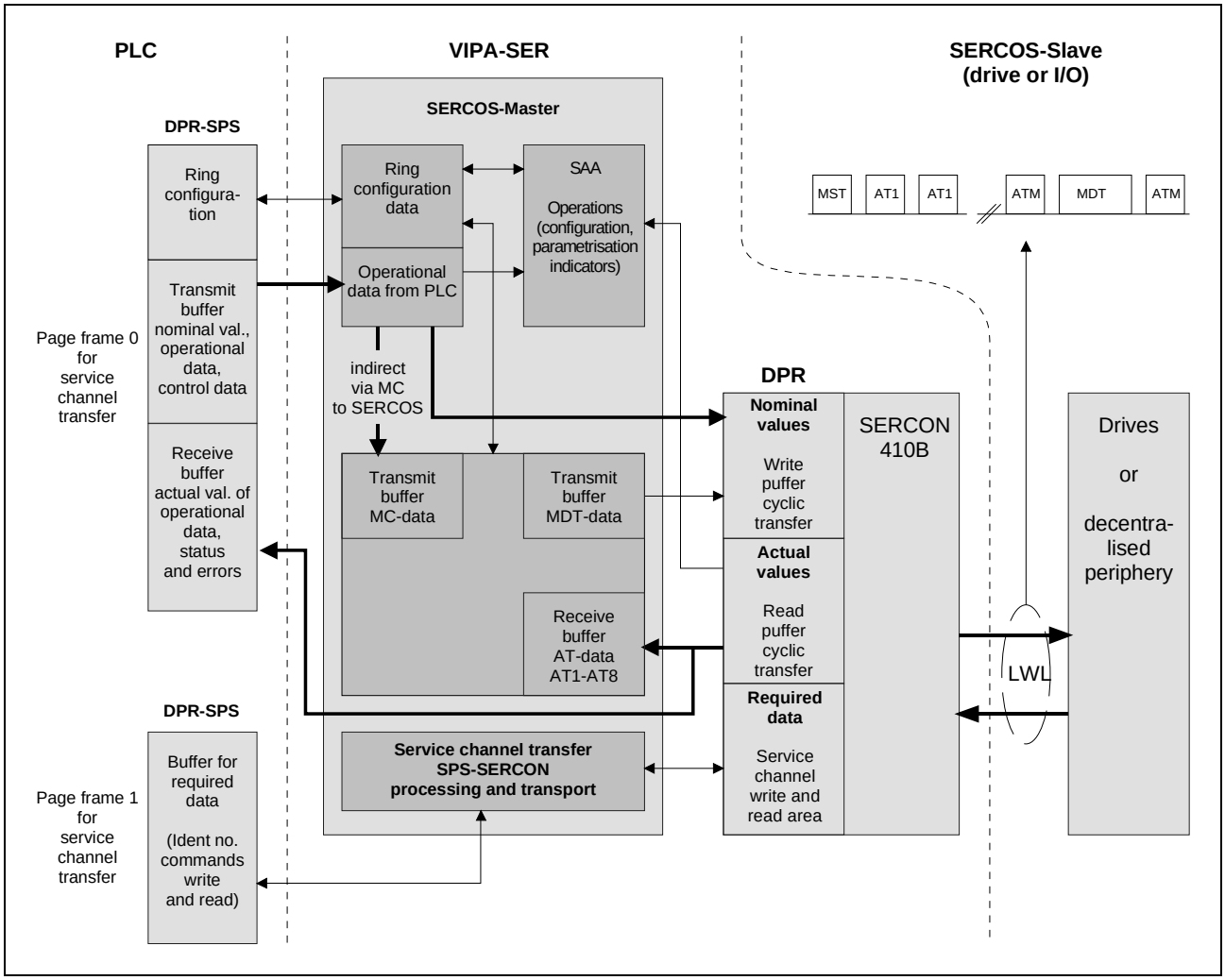


Fig. 3-1: Function unit MC

3.2 Motion-Control

3.2.1 General

Motion-Control is used to provide structured telegram buffer displays. A telegram buffer is an area in memory that you must use to transfer telegram data into the MDT (master data telegram) and to read telegram data from the AT (drive telegram). Your Motion-Control program can use the Telegram buffer MC (Motion-Control) as a parameter area. However, it is left to your discretion whether or not you wish to use this area.

3.2.2 Motion-Control configuration

You must assign a Motion-Control-Box to each and every slave that should be controlled by means of Motion-Control. You must define the input parameters that are required by the Motion-Control-Box (MCB). You can define a maximum of 8 input parameters per slave. The name and the unit are used for documentation purposes, the format is used for the representation of the data. The only mandatory entry is the data length. Parameters can consist of a word (16 bit) or a double word (32 bit). In floating point representation the length determines the number of digits that follow after the comma.

The configuration is entered into the SERC-Master software under menu item "Ring configuration - slave configuration - MC".

The following are valid entries for data length:

"1": parameter consists of a single word (2 digits after the comma in floating point format).

"2": parameter consists of a double word (4 digits after the comma in floating point format).

Note: you must cater for the number of digits after the decimal comma where MC-data must be transferred directly into MDT-data.

The format determines whether the parameters is shown as a decimal number, hexadecimal number or as a floating point number.

The following are valid for the format:

d, D Decimal representation

h, H Hexadecimal representation

f, F Floating point representation

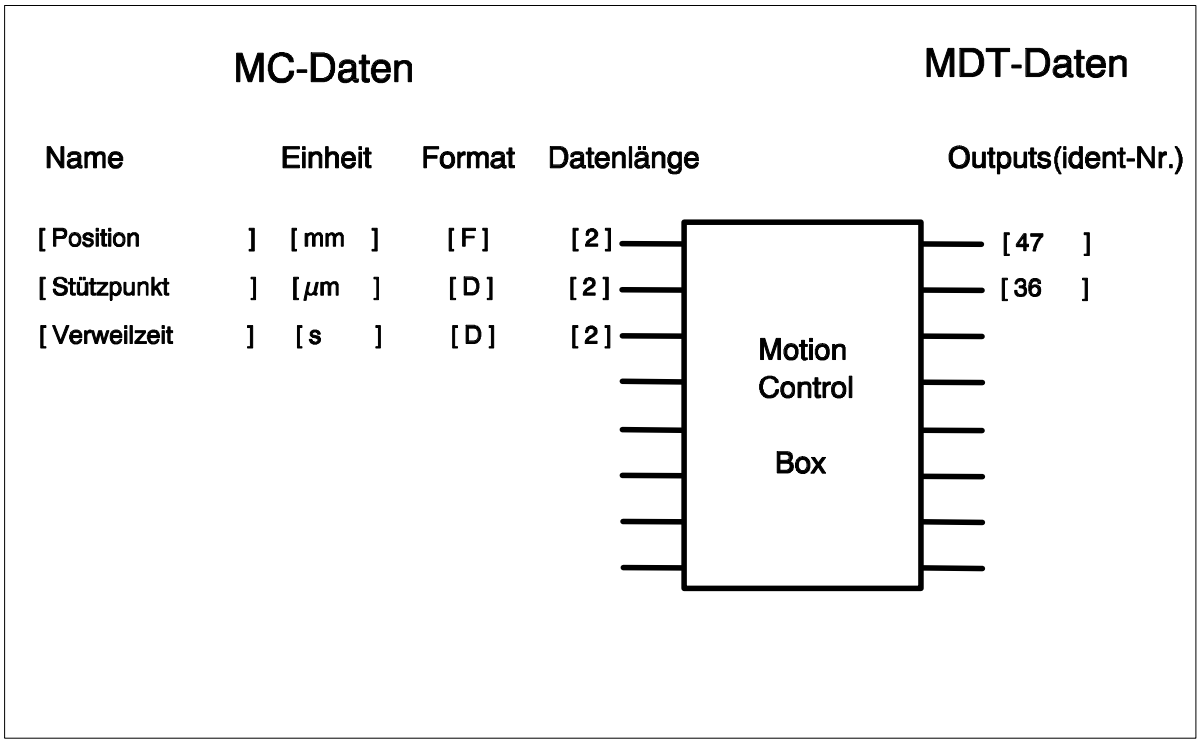


Fig. 3-2: MC-data configuration for a slave

Translation:

```
Einheit = Units
Datenlänge = Length of data
Stützpunkt = Fulcrum
Verweilzeit = Dwell time
```

3.2.3 Motion-Control status and control

All data of the cyclic telegram from the different slaves is displayed in the SERC-Master software special masks in under the menu "Cycle - control of cyclic telegrams" and you may change them here.

Fig. 3-3 shows the Motion-Control-Box with your configuration. The display also shows the actual values from the drive telegram (AT). To change MC-data or MDT-data (i.e. to control these) you must quit from the status display and move the cursor into the respective data fields. Here you can enter changes by overwriting existing data or inserting new data. Press "ALT-C" to accept any modified MC-data and to have submit the modifications for processing by your Motion-Control program. Modified MDT-data is accepted by means of the "ALT-D" keys and it is transferred directly into the master telegram (MDT).

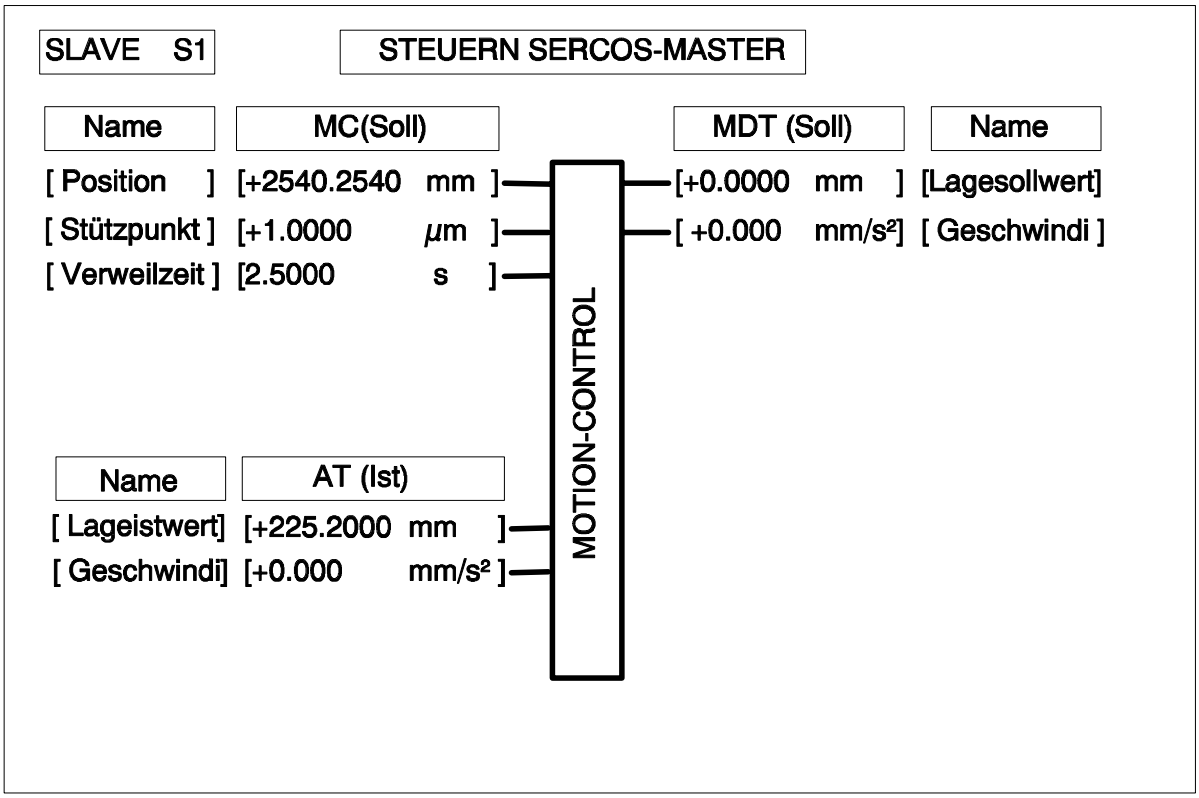


Fig. 3-3: Motion-Control Box

Translation: STEUERN = CONTROL, (Soll) = (nominal), Lagesollwert = Nominal location, Stützpunkt = Fulcrum, Geschwindi = Velocity, Verweilzeit = Dwell time, Lageistwert = Actual location, Geschwindi = Velocity

We have also provided a freely definable MC-control word and an MC-status word for general function purposes (e.g. synchronisation of PLC telegram data). The PLC can overwrite the MC-control word and this data can be used in your MC-program. You can use the MC-status word in your MC-program to transfer status information to the PLC.

You can find information on the data buffer, status and control words in chapter 3.4.

3.3 Functions

3.3.1 Description of the Motion-Control functions

MC-functions are supplied in the form of empty function shells. The function shells are functions that may be accessed once or on a cyclic basis from the Master-Software.

Programming requires the C-language (Borland-C). Where the timing is critical you may also use the 8086-Assembler language directly.

The following functions are available:

- void motion_control_START (void)

The "motion_control_START()" function is executed once at the end of phase 3. This function can be used for initialisation purposes, etc. This function does not place any overhead on the SERCOS-cycle.

- void motion_control_ZYK (void)

The "motion_control_ZYK()" function is executed cyclically from within the foreground program. This function may be used for routines where timing is critical. This function does not place any overhead on the SERCOS-cycle.

- void motion_control_INT(unsigned long Abgleichmaske)

The "motion_control_INT()" function is initiated by the periodic interrupt issued by the "SERCON410B" (interface-controller). The function is initiated at the start of every SERCOS-cycle. The processing of this function must be completed before the next telegram is received from the master.

Access and administration activities require a maximum of 0,5 ms. Therefore the available processing time is the cycle time minus 0,5ms. This function calculates the nominal values for every drive and enters them into the data buffers described below. As the programmer you are responsible for terminating the function in good time.

The **Co-ordination mask** (Abgleichmaske) consists of a 32Bit mask. Only the lower 8 bits of this mask are used and allocated to one slave each. This mask indicates to the MC-function that an actual value has changed, i.e. the function must assume that actual values have been modified (e.g. after a return to reference location).

3.3.2 The source-module "motctrl.c"

All the functions for Motion-Control are contained in the module-file "motctrl.c", the file "motctrl.h" contains the respective prototypes and the file "saa.h" contains general definitions.

Extract from the file "motctrl.c"

```
#include "saa.h"             /* Only significant for the defines WORD, BYTE */
#include "sercinit.h"        /* Contains the structure along with all information */
#include "motctrl.h"         /* Contains the prototypes of the function for Motion-Control */



/***************************************************************************
        Initialisation-routine
        Executed once at phase run-up time at the end of phase 3
        May be used for once-off initialisation purposes
***************************************************************************/
void Motion_Control_START(void)
{

}



/***************************************************************************
        IDLE-Routine
        Is initiated cyclically by the foreground program
        Causes no additional overhead to the interrupt handler
***************************************************************************/
void Motion_Control_ZYK(void)
{

}



/***************************************************************************
         Interrupt-Handler
         Initiated via the SERCOS-HW-Interrupt in the SERCOS-cycle

***************************************************************************/
void Motion_Control_INT(unsigned long Abgleichmaske)
{

}
```

3.4 Data structures

3.4.1 Description of the telegram data buffers

Three telegram data buffers are allocated to every slave (drive). These three data buffers contain nominal value parameters from the PLC (MC), nominal values for the master data telegram (MDT) and actual values from the drive telegram (AT) that was transferred by a slave. You determine the contents of the telegram data buffer for the master data telegram or the drive telegram by means of the telegram type you have selected. The telegram type is part of the configuration data and it is configured in the SERC-Master software.

The telegram data for telegram buffer MC, the number and the type of parameter(s) are also configured in the SERC-Master software. Every buffer has a maximum length of 16 words. The maximum number of buffer entries is 8 parameters or ident numbers for every buffer. Every parameter or ident number can consist of a word (16 bit) or a double word (32 bit).

MC-Data

The **MC-Data** buffer contains parameters that can be modified by the PLC or by your application program.

MDT-Data

The **MDT-Data** buffer contains nominal data for a slave that is supplied by your application. This data is automatically entered into the SERCOS-master data telegram.

AT-Data

The **AT-Data** buffer contains actual values from the SERCOS-drive telegram received from a slave. This data can be read and processed by your application.

Status and control words

MC-status and control words are also available. These consist of one control word for every drive and one status word as well as one control word for higher level purposes.

All the telegram buffers, status and control words are part of the structure that is described in chapter 3.4.2.

3.4.2 Access to data puffers and ring data

The volume and the length of the data in telegram buffers MDT and AT results from the selected telegram type. If you should use SERCOS-priority telegrams 0-6 the format of the telegram data is fixed. You may also configure your own telegrams. Here you may select any one of the available ident numbers that are offered by the slave (drive) in ident number lists 16 and 24. For the purpose of the program the telegram buffers are 16bit data fields that are included in a structure.

Description of the relevant structure data in structure "AKT"

Telegram data buffer

```
AKT -> Teledaten . MDT [0 - MAXTEL_LEN]                          MDT- telegram buffer
AKT -> Teledaten . AT [0 - MAXTEL_LEN]                           AT- telegram buffer
AKT -> Teledaten . MOTION_CONTROL[0 - MAXTEL_LEN]                MC- telegram buffer
```

Ring data

```
AKT -> ATCount                      Number of drives in the ring
AKT -> TScyc                        SERCOS-cycle time in µs
AKT -> Phase                        Phase number
AKT -> Hersteller [0 - MAXAT]       (0=ABB,1=Bosch,2=Indramat 2_1,3=Indramat 2_2, 4=Baumüller,
                                    5=Bautz, 6=Lütze, 7=Sonstige)
AKT -> modi .   ....                Operating modes for the different drives
AKT -> KonfTele .   ....            Lists of configurable telegrams
AKT -> SPS_offset_eing [0 - MAXAT]  Offset of the inputs in Lütze station for all slaves
AKT -> SPS_offset_ausg [0 -  MAXAT] Offset of the outputs in Lütze station for all slaves
AKT -> MDTSteuer [0 - MAXAT]        Control word of the cyclic master telegram
AKT -> ATStatus [0 - MAXAT]         Status word of the cyclic master telegram
AKT -> MDTInfo [0 -  MAXAT]         Info word of the cyclic master telegram
AKT -> ATInfo [0 - MAXAT]           Info word of the cyclic drive telegram
AKT -> MCStatus [0 - MAXAT]         Status word that can be modified from the MC for
                                    each and every slave
AKT -> MCStatuswort                 Status-word for the entire MC
AKT -> MCSteuerwort                 Control-word for the entire MC
AKT -> zk .    ...                  Images of the condition classes
```

The maximum number of drives is specified in "MAXAT" and it is defines as 8 in "sercinit.h".

3.4.3 Access to general PLC data

You can configure PLC data as required and this is available for general use. PLC data consists of 512 bytes and it is accessed by means of the variable name SPS. For the purpose of the program the variable SPS represents an "unsigned char far*" pointer. This means that the data in this location is byte-oriented.

Extract from motctrl.h:

```
static unsigned char far*         SPS = (unsigned char far*)MK_FP(0xC800,0x3800)
```

If you must transfer word-based data in accordance with the Motorola-format to the PLC you must reverse the Low/High bytes before you enter them into the PLC data area. This may then be followed by entering a word into the PLC data area by means of, for instance the function "memcpy".

Example

```
unsigned int swapword, var1;
.
.
var1=0xAA55;
swapword=swab_word (var1); /* Your function to rotate the bytes in var1)
/* swapword now contains 0x55AA */
memcpy (SPS+4,&swapword, sizeof (int) ); /* write starting with the fourth byte */
.
```

It is your responsibility to transfer consistent data into the PLC or to the SERCOS-side. We suggest the following procedure:

Install an area on the SERCOS-side that defines the type of data ant the respective offset that must be transferred into the PLC. Here you should then enter the respective transfer parameters when the PLC should acquire data. On the PLC you cyclically read area 1 via FB4 (see chapter 4.1.4) to determine whether data is available for collection from the SERCOS-card.

Install a second area where the PLC can enter the data that was transferred.

On the SERCOS-card you then analyse this data cyclically to determine whether data was transferred by the PLC.

This mechanism is only capable of transferring one contiguous data area per cycle.

Example

Area 1	**From SERCOS to PLC**
SPS+0 (DW n)	Offset data to PLC (0-255)
SPS+1 (DW n)	Data volume for PLC (1-255)
SPS+2 (DW n+1)	Handshake (bit 0 - data must be retrieved by the PLC, reset by the PLC after transfer was successful)
SPS+3 (DW n+1)	unused
Area 2	**From PLC to SERCOS**
SPS+4 (DW n+2)	Offset data from PLC (0-255)
SPS+5 (DW n+2)	Data volume to PLC (1-255)
SPS+6 (DW n+3)	Handshake (bit 0 - data must be retrieved by SERCOS, reset by SERCOS after transfer was successful)
SPS+7 (DW n+3)	unused
Area 3	**Net data area**
SPS+8 (DW n+4)	from this point onwards net-data may be transferred.

3.4.4 The include-file "sercinit.h"

The header file "sercinit.h" contains all the relevant structures and data about the SERCOS-ring. This header file must be included in every C-language module that must access this data by means of a "#include" statement.

Extract from the header file "sercinit.h"

```
/**********************************************/
/* SERCOS limit */
/**********************************************/

#define MAXAT                ( 8 )  /* Max. number of drives in a ring */
#define MAXRING              ( 8 )  /* Max. number of rings */
#define MAXTEL_LEN           ( 16 ) /* Max. length of a telegram in words */
#define MAX_DATEN            ( 8 )
#define MAXGRUPPE            ( 8 )
/*
Sub-structures that are also available for the "Motion-Control" module

The "R" signifies:    data must only be read !
Die "W" signifies:    data can be read and written !
*/

struct Telegramdata
{
  WORD AT [MAXAT][MAXTEL_LEN];              /* "R" actual value from SERCOS for SPS,MC...*/
  WORD MOTION_CONTROL[MAXAT][MAXTEL_LEN];   /* "R" nominal values from SPS to MC*/
  WORD MDT[MAXAT][MAXTEL_LEN];              /* "W" nominal values from MC to SERCOS */
};

struct Operatingmode
{
  WORD Hauptbetriebsart [MAXAT];     /* "R" the main operating mode (0-FFFF) */
  WORD Nebenbetriebsart1[MAXAT];     /* "R" the auxiliary operating mode (0-FFFF) */
  WORD Nebenbetriebsart2[MAXAT];     /* "R" the auxiliary operating mode (0-FFFF) */
  WORD Nebenbetriebsart3[MAXAT];     /* "R" the auxiliary operating mode (0-FFFF) */
  WORD Ist_Betriebsart [MAXAT];      /* "R" the actual operating mode (0-3) */
                                     /*(0=main operating mode ... 3=auxiliary operating mode 3) */
  WORD MotionControl[MAXAT];         /* "R" operation via Motion Control */
};

struct KonfigTelegramm
{
  WORD KonfLenMDT[MAXAT];                   /* "R" Data length of MDT-conf. msg. (no. of IdNo.s)*/
  WORD KonfInhaltMDT[MAXAT][MAXTEL_LEN];    /* "R" List of all ident numbers for the DT */
  WORD KonfLenAT[MAXAT]; /* "R" Data length of the AT-conf.telegram (no. of ident numbers) */
  WORD KonfInhaltAT [MAXAT][MAXTEL_LEN]; /* "R" list of all ident numbers for the AT */
};

struct Zustandsklassen
{
  WORD zk1 [MAXAT];                  /* "R" Image of condition class 1 */
  WORD zk2 [MAXAT];                  /* "R" Image of condition class 2 */
..WORD ss_status[MAXAT];             /* "R" Image of interface status */
  WORD hersteller_zk1[MAXAT];        /* "R" Image of manufacturer condition class 1 */
  WORD hersteller_zk2[MAXAT];        /* "R" Image of manufacturer condition class 2 */
  BYTE diagnosticmsg_AT[MAXAT][80];  /* "R" Image of the diagnostic telegrams (first 80 char.)*/
};
```

```
struct MST_AT
{
  struct       Telegramdata   Msgdata;       /* "R(W)" nominal data, actual data */

  WORD  ATCount;                             /* "R" number of drives */
  WORD  Tscyc;                               /* "R" SERCOS cycle time in µs */
  WORD  Phase;
  BYTE  Hersteller [MAXAT];                  /* "R" name of drive manufacturer as a value */
  WORD  Telegrammart [MAXAT];                /* "R" telegram types of the drives */
  struct Betriebsart  Modi;                  /* "R" operating modes for drives */
  struct KonfigTelegramm     KonfTele;       /* "R" List of configurable telegrams */

  BYTE  SPS_offset_eing [MAXAT];             /* "R" offsets for digital input stations */
  BYTE  SPS_offset_ausg [MAXAT];             /* "R" offsets for digital output stations */

  WORD  MDTSteuer [MAXAT];                   /* "R" control word  SERCOS (from SPS)*/
  WORD  ATStatus [MAXAT];                    /* "R" control word  AT (only valid in phase4)*/
  WORD  MDTInfo [MAXAT];                     /* "R" Info word     AT (only valid in phase4)*/
  WORD  ATInfo [MAXAT];                      /* "R" status word   AT (only valid in phase4)*/

  WORD  MCStatus [MAXAT];             /* "W" status word overwritten in Motion Control */
  WORD  MCStatuswort                  Status-word for the entire MC
  WORD  MCSteuerwort                  Control-word for the entire MC

  struct Zustandsklassen     zk;     /* "R" images of the condition classes of the drives */
.
.
}
.
.
extern struct MST_AT *AKT;
.
.
```

3.5 Example of three drives in the SERCOS ring

The ring includes three slaves. Slave 1 and slave 2 are controlled by Motion-Control. The PLC sends parameters to the MC-buffer for this purpose. The MC (application program) processes the parameters and produces nominal values for the MDT. Nominal value data is then entered into the MDT-buffers in the MC.

Slave 3 receives nominal values for the MDT (no operation via MC) directly from the PLC. For this purpose the MDT-buffer is overwritten directly. Slave 3 therefore follows the nominal values of the PLC.

SLAVE 1

- Operation via Motion-Control
- Telegram type is priority telegram 5 (ident numbers 47 and 36 for the MDT)
  (ident numbers 51 and 40 for the AT)

Parameters for MC: Par1: 2 words
Par2: 2 words
Par3: 1 word
Par4: 1 word

SLAVE 2

- Operation via Motion-Control
- Telegram type is the configurable telegram

Configurable telegram consists of :

MDT:	Id.Nr. xx:	2 words
	Id.Nr. yy:	2 words
AT:	Id.Nr. zz:	2 words

Parameters for MC: Par1: 1 word
Par2: 2 words
Par3: 2 words

SLAVE 3

- Drive follows nominal values of the PLC directly
- Telegram type is priority telegram 2 (ident number 36 for the MDT)
  (ident number 40 for the AT)

Buffer Slave 1

MC-data-buffer	MDT-data-buffer	AT-data-buffer
(Msg-data.MOTION _CONTROL)	(Msg-data.MDT)	(Msg-data.AT)
Par1 (low word)	Id.No. 47 (low word)	Id.No. 51 (low word)
Par1 (high word)	Id.No.47 (high word)	Id.No. 51 (high word)
Par2 (low word)	Id.No. 36 (low word)	Id.No. 40 (low word)
Par2 (high word)	Id.No. 36 (high word)	Id.No. 40 (high word)
Par3	unused	unused
Par4	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused

Tab. 3-1: Buffer Slave 1

Puffer Slave 2

MC-data-buffer	MDT-data-buffer	AT-data-buffer
(Msg.data.MOTION _CONTROL)	(Msg.data.MDT)	(Msg.data.AT)
Par1	Id.-No. xx (low word)	Id.No. zz (low word)
Par2 (low word)	Id.-No. xx (high word)	Id.No. zz (high word)
Par2 (high word)	Id.No. yy (low word)	unused
Par3 (low word)	Id.No. yy (high word)	unused
Par3 (high word)	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused

Tab. 3-2: Buffer Slave 2

Puffer Slave 3

MC-data-buffer	MDT-data-buffer	AT-data-buffer
(Msg.data.MOTION _CONTROL)	(Msg.data.MDT)	(Msg.data.AT)
unused	Id.No. 36 (low word)	Id.No. 40 (low word)
unused	Id.No. 36 (high word)	Id.No. 40 (high word)
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused
unused	unused	unused

Tab. 3-3: Puffer Slave 3

4 SERCOS Handler modules

4 SERCOS Handler modules

4.1 Function blocks

4.1.1 FB1 - initialisation of page frame addressing areas

```
BAUSTEIN#FB1
BSTNAME #INIT
BIB     #102
BEZ     #SSNR D:KY        Base page frame number
BEZ     #DBST B           Number of the control-DB
BEZ     #AZSL D:KF        Number of slaves if init-data is pre-set
                          by PLC
BEZ     #PAFE A:BY        Error byte
```

The initialisation module is called cyclically from OB1. After start-up, this module checks whether the initialisation data has been received from the SERCOS-board or whether they were transferred by the PLC (for the time being initialisation data must always be supplied from the SERCOS-side). The module uses this data to calculate up-to-date addresses in the page frame for data communications between the PLC and slaves.

Scratch bytes used

```
MB 234 - MB 255
```

Designators and designator values:

```
BEZ     #SSNR  D:KY  base page frame number (0-240)
BEZ     #DBST  B     number of the control-DBs (DB2 - DB255)
BEZ     #AZSL  D:KF  number of slaves when initialisation data is read
                     from the PLC
               = 0   when initialisation occurs from the SERCOS-side
BEZ      #PAFE A:BY  error byte (MB 0-199)
               = 0   no error
               <>    0 Error

               3 = base page frame not divisible by 16
               4 = control-DB not present
               5 = initialisation data in page frame not up to date
               6 = user is present on SERCOS-config-mask
               7 = undefined handshake value
               8 = slave-no. not in a valid range
               9 = SERCOS module not ready
```

Sample call

```
  :SPA FB 1         calculation of offset-addresses in the page frame
NAME #INIT
SSNR =KY 0,0        page frame number 0
DBST =DB 10         control-DB(10).
AZSL =KF +0         SERCOS-side initialises the page frame
PAFE =MB 198        save error nummer in MB 198
```

4.1.2 FB2 - synchronisation between PLC and SERCOS

```
BAUSTEIN #FB2
BSTNAME  #SYNCHRON
BIB      #102
BEZ      #SSNR D:KF
BEZ      #DBST B
BEZ      #PAFE E:BY
```

This module is used for the synchronisation between the PLC and the SERCOS-module and it is initiated at start-up or when a re-start occurs. At the start of the function block a delay loop is accessed to allow for the start-up delay of the SERCOS-module. The delay time is app. 30 sec. After this period a synchronisation flag is set in the page frame to indicate that the PLC is ready. From this point onwards the SERCOS-module can enter initialisation data into the page frame.

Designator and designator values

```
BEZ     #SSNR  D:KF  number of the base page frame(0-240)
BEZ     #DBST  B     number of the control-DBs  (2-255)
BEZ     #PAFE  BY    = 0 no error
               <>    0 an error has occurred:
               3     = base page frame not divisible by 16.
```

Scratch bytes used

```
MB 251 - MB 255
```

4.1.3 FB3 - transfer nominal value and actual value

```
BAUSTEIN#FB3
BSTNAME #DAT_ACHS
BIB     #102
BEZ     #SSNR D:KY  base page frame number
BEZ     #LAV  E:BY  Slave number
BEZ     #DBST B     number of the control-DB
BEZ     #DBDA E:BY  number of the slave-DB
BEZ     #PAFE A:BY  error byte
```

This FB is used to transfer nominal values for an axis to the SERCOS-board or to read actual values from the SERCOS-module. The actual values also include the status word from the drive. Furthermore the real-time control bits are also transferred.

Nominal values and actual values

The transfer of nominal values requires that the respective data is entered into the **Slave-DB** from data word 35. This can consist of a maximum of 16 words of nominal values. The definition and the number of nominal values depends on the configuration of the axis. Once all the nominal data has been entered the bit to "transmit nominal value" (D 1.0 in the **Slave-DB**) must be set. The nominal data is then transmitted to the board and the transmission bit is cleared when the next FB call is issued. In the opposite direction the actual value is read into the **Slave-DB** and saved from DW 51 when the FB is called while the bit "actual value is valid" is set.

Real-time control bits

When the real-time status bits for a slave have been configured the transfer of these bits is also handled by this module.

Status word from AT

The control word from the drive telegram AT is read with every call to the function block. The control word consists of the operating mode and the drive ready signal (DW3 and DW4 in the **Slave-DB**).

Scratch flags used

```
MB 230 - MB 255
```

Designator and range of values for designators:

```
BEZ     #SSNR  D:KY  base page frame no.     0-240
BEZ     #SLAV  D:KF  slave number            1-8
BEZ     #DBST  B     DB-no. of control-DB    DB 2-255
BEZ     #DBDA  E:BY  DB-no. of data-DB       DB 2-255
BEZ     #PAFE  A:BY  error byte              MB 0-199
               =  0  no error
               <> 0  error

               3 =   base page frame not divisible by 16
               4 =   control-DB not present
               9 =   address in data record not valid
               10 =  slave-no. less than or equal to zero
               11 =  slave-no. larger than quantity
               12 =  type out of range
               13 =  address out of range
               14 =  slave is an I/O-station
               15 =  data-DB does not exist
               16 =  data-DB too short
               17 =  drive not ready
```

Sample call

```
        :L  KB 2
        :T  MB 10    slave number in MB 10
        :
        :L  KB 20
        :T  MB 11    number of the data-DB in MB 11
        :
        :SPA FB 3    nominal/actual values from a slave
NAME    #DAT_ACHS    read or write (SLAVE parameter)
SSNR    =KY 0,0      page frame number 0
SLAV    =MB 10       number of the required slave
DBST    =DB 10       number of the control-DB (DB 10)
DBDA    =MB 11       number of the data-DB
PAFE    =MB 197      save error in MB 197
        :
```

4.1.4 FB4 - transfer of general nominal/actual values

```
BAUSTEIN #FB4
BSTNAME  #DAT_SPS
BIB      #102
BEZ      #SSNR D:KF        base page frame number
BEZ      #DB   E:BY        DB containing parameter data
BEZ      #ABDW E:BY        start of parameter data
BEZ      #DAT1 E:BY        offset of the data (0-255)
BEZ      #ANZ  E:BY        number of parameters for the transfer
BEZ      #R/W  E:BI        read/write parameters
BEZ      #PAFE A:BY        error byte
```

This FB is only of importance in conjunction with the SERCOS programming version.

The FB is used to transfer general data between the PLC and the SERCOS-board. The general data consists of 512 bytes and it is stored in a DB in the PLC. The SERCOS-board provides a memory area of 512 bytes where an image of the general data is stored.

Write

Before the write operation is initiated the data must be entered into data block DB at the designator contained in start parameter ABDW. The start of the data (DW0-255) and the number of data bytes (1-256) must be entered into designators DAT1 and ANZ. Designator R/W must be set to 1 to indicate a write operation.

Read

Data is saved from the start designator ABDW for the number of data bytes contained in designator ANZ into the specified data block DB. Designator R/W must be set to 0 to indicate a read operation.

A suitable mechanism must be devised for the transfer of consistent data. See chapter 3.4.3.

The PLC transfers data in Motorola-format (L-byte at most significant, H-byte at least significant address). On the SERCOS-side the data must be converted to Intel-format.
(L-byte at least significant address, H-byte at most significant address). For the programmer this means that the L-bytes and the H-bytes in a word must be swapped.

Scratch flags used

```
MB 248 - MB 255
```

Designators and ranges of values for designators

```
BEZ     #SSNR  D:KY        base page frame number (0-240)
BEZ     #DB    E:BY        number of the DB (DB2-255)
BEZ     #ABDW  E:BI        offset in the DB (0-255)
BEZ     #DAT1  E:BI        data offset (0-255)
BEZ     #ANZ   E:BY        number of data words (1-256)
BEZ     #R/W   A:BI        read(0) or write(1) data
BEZ     #PAFE  A:BY        error byte
                           0 = no error
                           3 = base page frame not divisible by 16
                           4 = DB does not exist
                           5 = short DB
```

Sample call

```
        :
        :L   KB 10         DB10 is the data-DB
        :T   MB 10
        :L   KB 5          write-data is located as from DW5
        :T   MB 11
        :L   KB 20         data transfer starts at offset 20
        :T   MB12
        :L   KB 2          the number of data bytes to be transferred
        :T   MB 13
        :
        :SPA FB 4

NAME    #DAT_SPS
SSNR    =KF 0        base page frame number
DB      =MB 10       MB containing the DB-number
ABDW    =MB 11       MB containing the offset in the DB
PAR1    =MB 12       MB containing the start offset
ANZ     =MB 13       MB cont. the number of data bytes for the transfer
R/W     =M  1.7      read/write
PAFE    =MB 197      save error in MB 197
   :
```

4.1.5 FB5 - ring configuration

```
BAUSTEIN #FB5
BSTNAME  #RING
BIB      #102
BEZ      #SSNR D:KY  base page frame number
BEZ      #DBST B     number of the control-DB
BEZ      #DBDA B     number of the slave-DB
BEZ      #LESR E:BI  read/write data
BEZ      #FRG  E:BI  FB must accept ring configuration
BEZ      #PAFE A:BY  error byte
```

This function block transfers ring data from the page frame into slave-DB's or from slave-DB's into the page frame (for the time being the ring configuration data must always be pre-set from the SERCOS-side). If bit "LESR" contains "0" the module checks whether current data is available from the SERCOS-module at which point it transfers the different axis data items (manufacturer, address, main operating mode, etc.) into the slave-DB's of the respective axis. If bit "LESR" contains a "1" the module enters the axis data into the page frame.

In case of multiple drives the data block numbers are incremented continuously from the DB-no. entered into designator DBDA. For instance, if designator DBDA contains DB20 that DB 22 is used as slave-DB for drive 3.

Scratch flags used

```
MB 240 - MB 255
```

Designator and designator values

```
BEZ     #SSNR  D:KY  base page frame number 0-240
BEZ     #DBST  B     number of the control-DB (DB 2 - 255)
BEZ     #DBDA  B     number of the slave-DB (DB 2 - 255)
BEZ     #LESR  E:BI  "0" = read ring configuration, "1" =
                     write ring configuration
BEZ     #FRG   E:BI  module must accept the ring configuration
BEZ     #PAFE  A:BY  error byte MB 0-199
               =  0  no error
               <> 0  error

               3 = base page frame not divisible by 16
               4 = control-DB not present
               5 = initialisation data in page frame not up to date
               6 = user located in configuration-mask
               7 = handshake has an undefined value
               8 = slave-no. out of range
```

Sample call

```
        :SPA FB 5    write or read ring configuration for all slaves
                     in the ring
NAME    #RING
SSNR    =KY 0,0      page frame number 0
DBST    =DB 10       number of the control-DB (DB 10)
DBDA    =DB 20       number of the first slave-DB  (continuous from 20)
LESR    =M 5.0       bit for reading or writing ring data
FRG     =M 5.1       bit to enable processing of the FB
PAFE    =MB 195      save error in MB 195
```

4.1.6 FB6 - axis configuration

```
BAUSTEIN #FB6
BSTNAME  #ACHSE
BIB      #102
BEZ      #SSNR D:KY  base page frame number
BEZ      #SLAV E:BY  slave number
BEZ      #DBST B     number of the control-DB
BEZ      #DBDA B     number of the slave-DB
BEZ      #PAFE A:BY  error byte
```

FB 6 transfers axis-related configuration data from a page frame into the slave-DB or from a slave-DB into the page frame (at present the axis configuration data must always be pre-set by the SERCOS-side). The bit "Enable config-data" (D 0.12) in the slave-DB is the enable bit for this FB. The block is only executed when this bit is set to "1". If the bit "Write config-data" (D 0.13) in the **Slave-DB** is "0" this block checks whether the SERCOS-module has up-to-date axis data and transfers the different axis data items (manufacturer, address, main operating mode, etc.) into the data block specified as designator DBDA. If the bit "Write config-data" (D 0.13) in the slave-DB is "1" the block writes the axis data into the page frame.

Scratch flags used

```
MB 239 - MB 255
```

Designator and designator values

```
BEZ     #SSNR  D:KY  base page frame number ( 0-240)
BEZ     #SLAV  E:BY  slave number that must be processed
BEZ     #DBST  B     number of the control-DB (DB 2-255)
BEZ     #DBDA  B     number of the slave-DB (DB 2-255)
BEZ     #LESR  E:BI  "0" for read config,"1" for write config
BEZ     #FRG   E:BI  enable processing of the block
BEZ     #PAFE  A:BY  error byte (MB 0-199)
               0     no error
               <> 0  error

               3 = base page frame not divisible by 16
               4 = control-DB not present
               5 = initialisation data in page frame not up to date
               6 = user located in configuration-mask
               7 = handshake has an undefined value
               8 = slave-no. out of range
```

Sample call

```
        :L   KB 1
        :T   MB 10   save slave number in MB 10
        :
        :SPA FB 6    read or write configuration data for a slave
NAME    #ACHSE
SSNR    =KY 0,0      page frame number 0
SLAV    =MB 10       slave number in MB 10
DBST    =DB 10       control-DB (DB 10)
DBDA    =DB 20       slave-DB  (DB 20)
PAFE    =MB 194      save error in MB 194
   :
```

4.1.7 FB7 - drive control

```
BAUSTEIN#FB7
BSTNAME #ANTR_EIN
BIB     #102
BEZ     #SSNR D:KF  base page frame number
BEZ     #SLAV E:BY  slave number
BEZ     #DBST B     number of the control-DB
BEZ     #DBDA E:BY  number of the slave-DB
BEZ     #STSP A:BI  control voltage
BEZ     #PAFE A:BY  error byte
```

The purpose of this block is to turn a drive on or off. The control bits in DR0 of slave-DB can be used to turn drives on/off, enable/disable or to stop or start a drive. In addition the control voltage can be connected as soon as the drive returns "ready" in the drive status word.

Designator and range of values for designators

```
BEZ     #SSNR  D:KY  base page frame no.( 0-240)
BEZ     #SLAV  D:KF  slave number (1-8)
BEZ     #DBST  B     number of the control-DB (DB 2-255)
BEZ     #DBDA  B     number of the slave-DB (DB 2-255)
BEZ     #PAFE  A:BY  error byte (MB 0-199)
               =  0  no error
               <> 0  error

               3 =   base page frame not divisible by 16
               4 =   control-DB not present
               9 =   address in data record not valid
               10 =  slave-no. less than or equal to zero
               11 =  slave-no. larger than quantity
               13 =  address out of range
               15 =  slave-DB does not exist
```

Scratch flags used

```
MB 248 - MB 255
```

Sample call

```
                :L   KB 1
                :T   MB 10   slave number in MB 10
                :
                :L   KB 20
                :T   MB 11   number of the slave-DB in MB 11
                :
                :SPA FB 7    turn slave on,stop or enable the slave (via
                             control word)
NAME    #ANTR_EIN
SSNR    =KF +0       page frame number 0
SLAV    =MB 10       slave number in MB 10
DBST    =DB 10       control-DB ( DB 10)
DBDA    =MB 11       number of the slave-DB in MB 11
STSP    =A 7.7       connect control voltage via A 7.7
PAFE    =MB 8        save error in MB 8
   :
```

4.1.8 FB8 - phase change and change of operating mode

```
BAUSTEIN #FB8
BSTNAME  #WECHSEL
BIB      #102
BEZ      #SSNR D:KF  base page frame number
BEZ      #SLAV E:BY  slave number
BEZ      #DBST B     number of the control-DB
BEZ      #DBDA E:BY  number of the slave-DB
BEZ      #PAFE A:BY  error byte
```

The purpose of this block is to initiate a phase change or to select a new operating mode.

Phase change

A different phase can be selected by means of the control bit "Execute a phase change" (D 0.8) in the slave-DB. The required number of the specific phase must first be entered into DW 31 of the slave-DB. This function performs a phase change on all slaves in the ring, i.e. the change into another phase can be initiated from any slave-DB.

Change of operating mode

The control bit "Execute change of operating mode" (D 0.10) of the slave-DB initiates a change to a new operating mode. The required operating mode must first be entered into DW 32 of the slave-DB. The number of the timer that should be used to monitor the timeout for the operation must be entered into DW 33 of the slave-DB.

Designators and ranges of values for designators

```
BEZ     #SSNR  D:KY  base page frame number (0-240)
BEZ     #SLAV  D:KF  slave number (1-8)
BEZ     #DBST  B     number of the control-DB (DB 2-255)
BEZ     #DBDA  B     number of the slave-DB (DB 2-255)
BEZ     #PAFE  A:BY  error byte (MB 0-199)
               =  0  no error
               <> 0  error

               3 =   base page frame not divisible by 16
               4 =   control-DB not present
               5 =   slave-DB does not exist
               6 =   illegal phase number
               7 =   error during phase run-up
               9 =   address in data record not valid
               10 =  slave-no. less than or equal to zero
               11 =  slave-no. larger than quantity
               13 =  address out of range
               14 =  invalid operating mode
               15 =  error during change of operating mode
```

Scratch flags used

```
MB 248 - MB 255
```

Sample call

```
                :L   KB 2
                :T   MB 10   slave number in MB 10
                :
                :L   KB 21
                :T   MB 11   number of the data-DB in MB 11
                :
                :SPA FB 8    phase change or change of operating mode for a
                             slave

        NAME    #WECHSEL
        SSNR    =KF +0       page frame number 0
        SLAV    =MB 10       slave number in MB 10
        DBST    =DB 10       control-DB (DB 10)
        DBDA    =MB 11       number of the slave-DB in MB 11
        PAFE    =MB 190      save error in MB 190
           :
```

4.1.9 FB9 - Service channel transfer

BAUSTEIN #FB9

```
BSTNAME #SERVICE
BIB     #102
BEZ     #SSNR D:KF  base page frame number
BEZ     #DBST B     control-DB
BEZ     #DBDA B     slave-DB
BEZ     #PAFE A:BY  error byte
```

This block is used to read or write production data from/to a slave. Data is addressed by means of an ident number (see SERCOS-specification). Service channel data is transferred between the PLC and the SERCOS-module by means of the control bits "Service read" (D 1.10) or "Service write" (D 1.8) in the **Slave-DB**.

Read data

The slave-no. (DW 112) and the ident number (DW 113) must be entered into a slave-DB before the bit "Read service data" is set. Following this the bit "Read service data" (D 1.10) can be set. In this case only the attribute (from DW 144) and the production data (from DW 156) is entered into the slave-DB. The attribute indicates the structure of the production data. If you require more detailed information you must set further control bits in DL 1 (D 1.12 "Read name" , D 1.13 "Read unit", D 1.14 "Read min/max-value").

Write data

The slave-no (DW 112) and ident number (DW 113) as well as the production data (from DW 156) must be entered into a slave-DB before the bit "Write service data" is set. This is then followed by setting the bit "Write service data" (D 1.8).

When an error occurs during the read or write operation the initiation bit for write or read is cleared and an error number is returned via the PAFE designator. This corresponds to the SERCOS-error message of the service channel.

Designators and ranges of values for designators

```
BEZ     #SSNR  D:KY  base page frame number (0-240)
BEZ     #DBST  B     number of the control-DB (DB 2-255)
BEZ     #DBDA  B     number of the slave-DB (DB 2-255)
BEZ     #PAFE  A:BY  error byte (MB 0-199)
               =  0  no error
               <> 0  error

               3 = base page frame not divisible by 16
               4 = control-DB not present
               5 = slave-DB does not exist
               9 = address in data record not valid
```

Scratch flags used

MB 248 - MB 255

Sample call

:SPA FB 9 read or write service data from a slave

```
NAME    #SERVICE
SSNR    =KF +0       page frame number 0
DBST    =DB 10       control-DB (DB 10)
DBDA    =DB 20       slave-DB (DB 20)
PAFE    =MB 189      save error in MB 189
   :
```

4.1.10 FB10 - Commands

```
BAUSTEIN #FB10
BSTNAME  #KOMMANDO
BIB      #102
BEZ      #SSNR D:KF
BEZ      #DBST B
BEZ      #DBDA B
BEZ      #PAFE A:BY
```

This block is used to transfer commands to a slave. The control bits in the **Slave-DB** "Initiate a command" (DR 2) or "Clear a command" (DR 5) can be used to transfer different commands to multiple slaves simultaneously. For slave 1 the number of the command must be entered into DW 96 of the **Slave-DB** and then the bit "Initiate command" (D 2.0) must be set. The command numbers for subsequent slaves must be entered into DW 97 through DW103 and the respective bits for "Initiate command" (D 2.1 to D 2.7) must be set. If a command should not be executed and no error number is returned the bit "Clear a command" (D 5.0 - D 5.7) can be used to reset the command for the respective drive. The initiation bit "Initiate command" (D 2.0) must be cleared before the bit "Clear command" D 5.0 is set.

Designators and ranges of values for designators

```
BEZ     #SSNR  D:KY        base page frame number (0-240)
BEZ     #DBST  B           DB-no. of control-DB (DB 2-255)
BEZ     #DBDA  B           DB-no. of slave-DB (DB 2-255)
BEZ     #PAFE  A:BY        error byte (MB 0-199)
               =  0        no error
               <> 0        error

               3 =         base page frame not divisible by 16
               4 =         control-DB not present
               5 =         slave-DB does not exist
               9 =         address in data record not valid
               20 - 27 =   error during the transfer of the command
                           (20 = slave 1, 27 = slave 8)
```

Scratch flags used

```
MB 248 - MB 255
```

Sample call

```
        :SPA FB 10   issue or clear commands to a slave
NAME    #KOMMANDO
SSNR    =KF +0       page frame number  0
DBST    =DB 10       control-DB (DB 10)
DBDA    =DB 20       slave-DB (DB 20)
PAFE    =MB 188      save error in MB 188
   :
```

4.1.11 FB11 - single step operation

```
BAUSTEIN   #FB11
   BSTNAME #TIPP
   BIB     #102
   BEZ     #SSNR D:KY        base page frame number
   BEZ     #SLAV E:BY        slave number
   BEZ     #DBST E:BY        number of the control-DB
   BEZ     #TIP+ E:BI        single step operation, plus
   BEZ     #TIP- E:BI        single step operation, minus
   BEZ     #GES+ E:BI        increase velocity
   BEZ     #GES- E:BI        decrease velocity
   BEZ     #TIME T           key-pressed supervision
   BEZ     #PAFE A:BY        error byte
```

This module is used to position an axis in single step operation. The number of the slave must be entered into designator SLAV. The direction and the velocity can be controlled by means of four binary signals. Single step operation and direction are activated by means of designators TIP+ or TIP-. While in single step mode the velocity can be increased or decreased by means of designator GES+ or GES-. The designator TIME defines the timer for the delay time. The purpose of the delay time is to clear the velocity when single step operation has been turned off.

Scratch flags used

MB 248 - MB 255

Designators and ranges of values for designators

```
BEZ     #SSNR  D:KY        base page frame number (0-240)
BEZ     #SLAV  E:BY        slave number (1-8)
BEZ     #DBST  E:BY        number of the control-DB (DB2-255)
BEZ     #TIP+  E:BI        single step operation, plus
BEZ     #TIP-  E:BI        single step operation, minus
BEZ     #GES+  E:BI        increase velocity
BEZ     #GES-  E:BI        decrease velocity
BEZ     #TIME  T           key-pressed supervision (T0-T127)
BEZ     #PAFE  A:BY        error byte (MB0-MB199)
                     3 =   base page frame not divisible by 16
                     4 =   control-DB not present
                     9 =   address in data record not valid
                     10 =  slave-no. less than or equal to zero
                     11 =  slave-no. larger than quantity
                     21 =  no operating mode
                     22 =  no drive stand still
                     23 =  invalid drive number
```

Sample call

```
        :L   KB 3
        :T   MB 10
        :SPA FB 11   transfer or clear commands for a slave
NAME    #TIPP
SSNR    =KY 0,0      page frame number  0
SLAVE   =MB 10       slave number entered into MB10
DBST    =DB 10       control-DB (DB 10)
TIP+    =E 2.0       axis must be positioned in +direction
TIP-    =E 2.1       axis must be positioned in negative direction
GES+    =E 2.2       increase velocity
GES-    =E 2.3       decrease velocity
TIME    =T34         key-pressed timer
PAFE    =MB 197      save error in MB 197
   :
```

4.1.12 FB12 - write/read NC-parameter

```
BAUSTEIN #FB12
BSTNAME  #PARAME
BIB      #102
BEZ      #SSNR D:KF        base page frame number
BEZ      #DB   E:BY        DB containing parameter data
BEZ      #ABDW E:BY        start of parameter data
BEZ      #PAR1 E:BY        number of the first parameter (0-127)
BEZ      #ANZ  E:BY        number of parameters for the transfer
BEZ      #R/W  E:BI        read/write parameters
BEZ      #PAFE A:BY        error byte
```

Parameters are variables for use in NC-programs. The purpose of the FB12 block is to pre-define or to read these parameters.

A maximum of 128 parameters (P0 through P127 with a length of one double word each) can be read or written by a single call to the function.

Parameters can contain nominal values, actual values as well as logical data. Logical parameter data must be seen as parameters containing a comparison value that is used by the NC-program to control the execution of loops, branch instructions, etc.

SPS	Serc-NC	
0	0.0000	
1	0.0001	
10	0.0010	less than 0,5 corresponds to 0 (FALSE) in a
100	0.0100	logical test in the NC
1000	0.1000	
.	.	
4999	0.4999	
---	---	---
5000	0.5000	
10000	1.0000	
100000	10.0000	
1000000	100.0000	larger than or equal to 0,5 corresponds to 1(TRUE) in
10000000	1000.0000	a logical test in the NC
100000000	10000.0000	
1000000000	100000.0000	
2147483647	214748.0000	

Tab. 4-1: Parameter representation (32 bit)

Write

Before a write command is issued the parameter data must be entered into data block DB in the data word contained in start parameter ABDW. The start of parameter data (0-127) and the number of parameters (1-128) must be entered into designators PAR1 and ANZ. The designator R/W must be set to 1 for write.

Read

Parameters are saved in the specified data block DB from start designator ABDW for the number of data bytes in designator ANZ. Designator R/W must be set to 0 for read.

Scratch flags used

```
MB 248 - MB 255
```

Designators and ranges of values for designators

```
BEZ     #SSNR  D:KY        base page frame number (0-240)
BEZ     #DB    E:BY        number of the parameter-DB (DB2-255)
BEZ     #ABDW  E:BI        offset in the DB (0-255)
BEZ     #PAR1  E:BI        parameter number (0-127)
BEZ     #ANZ   E:BY        number of parameters (1-128)
BEZ     #R/W   A:BI        read(0) or write(1) parameters
BEZ     #PAFE  A:BY        error byte

                     3 =   base page frame not divisible by 16
                     4 =   DB does not exist
                     5 =   short DB
```

Sample call

```
        :A   DB 10
        :L   KH 0001       save a value of 6.9905 in DD5
        :T   DW 5
        :L   KH 1111
        :T   DW 6
        :L   KH 2710       save a value of 1.0000 (or logical 1) in DD7
        :T   DW 7
        :L   KH 0
        :T   DW 8
        :
        :L   KB 10         DB10 is the parameter-DB
        :T   MB 10
        :L   KB 5          parameter data starts from DW5
        :T   MB 11
        :L   KB 20         data transfer to start from parameter 20
        :T   MB12
        :L   KB 2          number of parameters for the transfer
        :T   MB 13
        :
        :SPA FB 12
NAME    #PARAME
SSNR    =KF 0              base page frame number
DB      =MB 10             MB containing the DB-number
ABDW    =MB 11             MB containing the offset in the DB
PAR1    =MB 12             MB containing the number of the first
                           parameter
ANZ     =MB 13             MB containing the number of parameters that
                           must be transferred
R/W     =M  1.7            read/write
PAFE    =MB 197            save error in MB 197
   :
```

4.1.13 FB13 - load and start NC-programs

```
BAUSTEIN #FB13
BSTNAME  #PROGRAMM
BIB      #102
BEZ      #SSNR       D:KF        base page frame number
BEZ      #DBAR       E:BY        DB used
BEZ      #ABDW       E:BY        offset of the program name in the DB
BEZ      #LANG       E:BY        length of the program name
BEZ      #PRNR       E:BY        program number
BEZ      #STEU       E:BY        control of the NC-program
BEZ      #PAFE       A:BY        error byte
```

On the SERCOS-module NC-programs can be saved as DOS files. The SERCOS-module can simultaneously execute up to a maximum of 8 NC-programs. Every program is associated with an axis-group. A program for an axis-group is started by means of a call to function block FB13. A single call to a function block can only load or start a single NC-program for one axis-group. The program name of the NC-program that must be loaded must first be entered into the data block DBAR. The offset where the name starts in the data block and the length of the name must be entered into designators ABDW and LANG respectively. Once the program name has been saved in the data block parameter STEU can be used to execute the load and start of the NC-program.

DW	Description	DW	Description
0	Status byte / control byte for program axis-group 1	8	parameter error (PAFE) axis-group 1
1	axis-group 2	9	axis-group 2
2	axis-group 3	10	axis-group 3
3	axis-group 4	11	axis-group 4
4	axis-group 5	12	axis-group 5
5	axis-group 6	13	axis-group 6
6	axis-group 7	14	axis-group 7
7	axis-group 8	15	axis-group 8

Tab. 4-2: Structure of data block DBAR

Starting from data word 16 you can enter up to 8 names for up to 8 NC-programs. Program names correspond with the DOS-conventions and consists of a maximum of 8 characters. Extensions are not permitted. The SERCOS-module always uses a default extension of ".din".

The control bytes in the data block (DBAR DR0 to DR7) correspond to the designator STEU. Designator PRNR defines which program (axis-group) should be controlled.

Assignment of the control byte (right-hand data word)

Dec.

1 Load a program

2 Start a program

3 Load a program and start it afterwards

Assignment of the control byte (left-hand data word)

Dec.

0 No NC-program loaded

1 NC-program terminated manually

2 NC-program terminated by run-time error

3 NC-program running

4 NC-program located at start of program (after loading)

5 NC-program terminated without error

6 NC-program waiting in a programmed Halt (path condition M01)

7 NC-program was stopped manually at a record boundary

Scratch flags used

MB 239 - MB 255

Designators and ranges of values for designators

```
BEZ     #SSNR       D:KF        base page frame number(0-240)
BEZ     #DBAR       E:BY        DB used (DB 2 - 255)
BEZ     #ABDW       E:BY        offset of the name (DW 16 - 254)
BEZ     #LANG       E:BY        length of the name (1 - 8 characters)
BEZ     #PRNR       E:BY        axis-group (0 or 1 - 8)
BEZ     #STEU       E:BY        control byte
                                1 = load NC-program
                                2 = start NC-program
                                3 = load and start NC-program
BEZ     #PAFE       A:BY        error byte
                                0     no error
                                <> 0  error
                                3 =   base page frame not divisible
                                      by 16
                                4 =   DB does not exist
                                5 =   DB offset invalid
                                6 =   name too long

                                10 = NC-program not valid
                                11 = no access
                                12 = NC-program does not exist
                                13 = program read error
                                14 = error in NC-program
                                15 = not enough memory to load program
```

Sample call

```
          :L   KB13          use DB13
          :T   MB104
          :L   KB16          name located from DW16
          :T   MB105
          :L   KB8           name has 8 characters (maximum)
          :T   MB106
          :L   KB1           axis-group 1
          :T   MB107
          :L   KB3           load and start program
          :T   MB108
          :SPA FB 13
NAME      #PROGRAMM
SSNR      =KF 0              base page frame number
DBAR      =MB 104            MB containing the DB-number
ABDW      =MB 105            MB containing the offset in the DB
LANG      =MB 106            MB containing the length of the name
PRNR      =MB 107            axis-group for which a program must be
                             loaded
STEU      =MB 108            control byte for loading and starting
                             a program
PAFE      =MB 197            save error in MB 197
   :
```

If it is required that multiple programs are started simultaneously in a PLC cycle it is possible to pre-allocate the control bytes for the respective axis-group directly in the data block. Subsequently the call to FB13 can be issued using program number 0 in designator PRNR. One condition is that all NC-programs have already been loaded.

4.2 Data blocks

4.2.1 Control data block

The control data block contains general data that is not specific to a certain slave. The programmer must define an area of 256 words for the control data block. These 256 data words must be pre-set to KH0000. During the configuration of the following SERCOS-handler modules the number of the control-DB must be specified in designator DBST:

FB1	INIT	
FB2	SYNCHRON	
FB3	DAT_ACHS	(DAT_AXIS)
FB5	RING	
FB6	ACHSE	(AXIS)
FB7	ANTR_EIN	(DRIVE_ON)
FB8	WECHSEL	(CHANGE)
FB9	SERVICE	
FB10	KOMMANDO	(COMMAND)
FB11	TIPP	(SINGLE-STEP)

Structure of the control data block

```
DW0             summary bit: clear condition classes for all slaves by
                means of D 0.15 (8000hex)
DW1             internal
DW2             number of slaves in the SERCOSring
DW3             cycle time in the SERCOSring
DW4             ACTUAL phase
DW5             operating mode (the actual SERCOS operating mode of all slaves)
DW6             nominal phase (the required phase)
DW7             internal
DW8             internal
DW9             internal
DW10            error when clearing the condition class (see drive-service-
                info)
DW11            internal
DW12            MC-status word (status word from SERCOS or MC-program)
DW13            MC-control word (control word from PLC)
DW14            watchdog
                The analysis of the watchdog can occur on the PLC. DR14
                contains the cyclic watchdog, DL14 the interrupt watchdog. The
                cyclic watchdog operates as a counter and it is triggered by
                the foreground program. The interrupt watchdog is only active
                as of phase4 and counts at the frequency of the selected SERCOS
                cycle time.
DW15            internal
DW16            internal
.
.               internal
.
```

4.2.2 Slave data block

The slave data block contains all the data, status and control information that is exchanged between a specific slave and the SERCOS-module. One slave data block is required per slave. The programmer must define an area of 256 words for the slave data block. These 256 data words must be pre-set to KH0000. Slave data blocks can be allocated in one continuous sequence or separately, one per slave. The configuration of slave data blocks is performed by means of function blocks FB5 or FB6. If slave data blocks should be located at consecutive addresses it is only necessary to specify the number of the first slave data block in designator DBDA when the call to FB5 (RING) is issued. FB6 (ACHSE) must be used if the slave-DB for every slave must be freely addressable. This block expects the number of the required data block in the designator DBDA. In this case a call to FB6 must be issued separately for every slave. The call to F5 can be omitted.

The Serc-Master-Software on the SERCOS-module enters the configuration data into the Dual-Ported-RAM. From here the configuration data is read from FB5 or FB6. The SERCOS-module then transfers the configuration data automatically into the Dual-Ported-RAM when the call to the Serc-Master-Software is issued.

Slave data block structure

Status and control data

```
DW  0     DL      control bits phase change,change of operating mode,config.
                  read/write
          DR      control bits for the drive on/off/enable/halt

DW  1     DL      control bits for nominal values /real-time status bits
          DR      transfer control bits for the service channel
DW  2     DL      acknowledgement "command has already been issued "
          DR      control bits initiate command

DW  3     DL      current operating mode
          DR      required operating mode

DW  4     DL      SERCOS-status byte
          DR      SERCOS-control byte

DW  5     DL      reserved
          DR      clear control bits for the command

DW  6             Reserve
DW  7             Reserve
DW  8             Reserve
DW  9             Reserve
```

Configuration data

```
DW 10            drive manufacturer
DW 11            address
DW 12            main operating data
DW 13            auxiliary operating mode 1
DW 14            auxiliary operating mode 2
DW 15            auxiliary operating mode 3
DW 16            type of telegram
DW 17            data length for run
DW 18            real-time signal 1
DW 19            real-time signal 2
DW 20            handshake
DW 21            control word
DW 22            status word
DW 23            condition class 1
DW 24            condition class 2
DW 25            interface status
DW 26            manufacturer c. c. 1
DW 27            manufacturer c. c. 2
DW 28            real-time control bits
DW 29            MC-status word of the slave
DW 30            unused
DW 31            new phase for phase change
DW 32            new operating mode for change of operating mode
DW 33            timer-no. for time limit during the change
```

Nominal/actual value data

```
DW 35 to DW 49   nominal value for drive (16 words max.)
DW 51 to DW 67   actual value from drive (16 words max.)
DW 70 - 89       reserved
```

Command functions (see also FB10 commands)

```
DW  2     DL      acknowledgements "Command already issued "
          DR      control bits initiate the command
DW 90 to DW 195   data for command functions
                  DW92              command error word
                  DW93              slave number where the error has occurred
                  DW96  to DW103    command number for the 8 slaves
                  DW104 to DW111    command acknowledgements for the 8 slaves
```

Service functions (see also FB9 service channel transfers)

```
DW 112 to DW195  data for service-functions
                  DW112             slave-number for service function
                  DW113             ident-number for service function
                  DW114 to DW143    name (114,115 = length-parameter)
                  DW144 to DW145    attribute
                  DW146 to DW151    unit (146,147 - length-parameter)
                  DW152 to DW153    min. value
                  DW154 to DW155    max, value
                  DW156 to DW195    operating data (156,157 - length-parameter
                                    for variable operating data)
```

Description of the different bits

Drive connection

```
D 0.0            turn on control voltage
D 0.1            turn drive on
D 0.2            turn drive off
D 0.3            drive halt
D 0.4            drive restarted after halt
D 0.5            turn off drive enable
D 0.6            unused
```

Phase run-up

```
D 0.8            execute a phase change
D 0.9            acknowledgement for "new phase accepted"
```

Change of operating mode

```
D 0.10           execute a change of operating mode
D 0.11           acknowledgement for "new operating mode accepted "
```

Transfer the configuration to SERCOS (not supported by the SERCOS-side at present)

```
D 0.12           enable config. data
D 0.13           write config. data
```

Transfer nominal values

```
D 1.0            transmit nominal values located from W35 onwards
```

Real-time status bits

```
D 1.6            real-time bit 1 from drive
D 1.7            real-time bit 2 from drive
```

Service channel transfer

```
D 1.8            write service data
D 1.9            acknowledgement for "service data was written"
D 1.10           read service data
D 1.11           internal
D 1.12           read the name of the production data
D 1.13           read the unit of the production data
D 1.14           read min/max value of the production data
```

Command transfer

```
D 2.0            initiate command slave 1
D 2.1            initiate command slave 2
D 2.2            initiate command slave 3
D 2.3            initiate command slave 4
D 2.4            initiate command slave 5
D 2.5            initiate command slave 6
D 2.6            initiate command slave 7
D 2.7            initiate command slave 8

D 2.8            status commando running slave 1
D 2.9            status commando running slave 2
D 2.10           status commando running slave 3
D 2.11           status commando running slave 4
D 2.12           status commando running slave 5
D 2.13           status commando running slave 6
D 2.14           status commando running slave 7
D 2.15           status commando running slave 8

D 5.0            command clear slave 1
D 5.1            command clear slave 2
D 5.2            command clear slave 3
D 5.3            command clear slave 4
D 5.4            command clear slave 5
D 5.5            command clear slave 6
D 5.6            command clear slave 7
D 5.7            command clear slave 8
```

SERCOS control word of the drive (generated by the PLC from the control bits)

D 4.0	**D 4.1**	nominal operating mode
0	0	main operating mode
0	1	auxiliary operating mode 1
1	0	auxiliary operating mode 2
1	1	auxiliary operating mode 3

```
D 4.2              reserved
D 4.3              reserved
D 4.4              reserved
D 4.5              drive halt
                   0 = drive halt
                   1 = drive enabled
D 4.6              drive enabled
                   0 = not enabled
                   1 = drive enabled
D 4.7              turn drive on
                   0 = drive off
                   1 = drive on
```

SERCOS status word from a drive

D 4.8	**D 4.9**	actual operating mode
0	0	main operating mode
0	1	auxiliary operating mode 1
1	0	auxiliary operating mode 2
1	1	auxiliary operating mode 3

```
D 4.10             reserved
D 4.11             change of condition class 3
                   0 = no change
                   1 = change occurred
D 4.12             change of condition class 2
                   0 = no change
                   1 = change occurred
D 4.13             drive locked, error in condition class 1
                   0 = no error
                   1 = drive is locked due to an error situation
```

D 4.14	**D 4.15**	ready
0	0	drive not ready for power on
0	1	drive is ready for power on
1	0	drive controller and power supply ready
1	1	drive operational

4.3 Sample call for two drives in the SERCOS ring

OB1

Three additional modules are required to implement the call in the OB1. This example uses **FB20**, **FB21** and **FB23**. FB20 is initiated once per cycle and it provides ring and axis initialisation, the service channel transfer and command processing functions.

FB21 is initiated once per slave. It requires parameters for the slave number, the number of the slave-DB and a byte for single-step operation.

FB23 is responsible for the control of calls to the NC-program. FB23 contains the routines to load and start up to eight NC-programs.

```
          BAUSTEIN#OB1
          BIB     #6054
00000             :
00002             :SPA FB 20
          NAME #SERCRING
00006             :
00008             :SPA FB 21
          NAME #SLAVES
0000C             :
0000E             :SPA FB 23
          NAME #PROGALL
00012             :
00014             :BE
```

OB21

```
          BAUSTEIN#OB21
          BIB     #3054
00000             :
00002             :SPA FB 2
          NAME    #SYNCHRON
          SSNR    =KF +0
          DBST    =DB 10
          PAFE    =MB 198
0000C             :
0000E             :BE
```

FB20

All modules called from FB20 supervise the different control bits that are supplied by the SERCOS-module for a possible re-initialisation of the ring- and axis data and to control the required service channel transfers that are initiated by the PLC.

```
          BAUSTEIN #FB20
          BSTNAME  #SERCRING
          BIB      #10084
0000A             :
0000C             :SPA FB 1    calculate offset address in page frame
          NAME    #INIT        transfer config-data into the control DB(10)
          SSNR    =KY 0,0
          DBST    =DB 10
          AZSL    =KF +0
          PAFE    =MB 198
00018             :
0001A             :SPA FB 5    write or read the ring configuration (for all
                               slaves)
          NAME    #RING
          SSNR    =KY 0,0
          DBST    =DB 10
          DBA     =DB 20
          LESR    =M 5.0
          FRG     =M 5.1
          PAFE    =MB 198
0002A             :
0002C             :
0002E             :SPA FB 9    read or write service data from a slave
          NAME    #SERVICE
          SSNR    =KF +0
          DBST    =DB 10
          DBDA    =DB 20
          PAFE    =MB 198
0003A             :
0003C             :SPA FB 10   initiate or clear commands in a slave
                               (data is located in the
          NAME #KOMMANDO       slave-DB,the command number in DW96, the slave no.
                               in D2.0-D2.15)
          SSNR    =KF +0
          DBST    =DB 10
          DBDA    =DB 20
          PAFE    =MB 198
00048             :
0004A             :BE
```

FB21

FB21 contains the calls for two slaves located in the ring. Additional slaves may be entered at the end of the module by means of a call and configuration of FB22. No changes are required for FB22.

```
          BAUSTEIN #FB21
          BSTNAME #SLAVES
          BIB      #16105
0000A             :
0000C             :L     KB 1
0000E             :T     MB 20
00010             :L     KB 20       WORK-DB 20
00012             :T     MB 21
00014             :
00016             :SPA FB 22         CALL FOR SLAVE 1
                 NAME #SLAVE_N
                 SLAV =MB 20
                 DBNR =MB 21
                 TIPP =MB 22
00020             :
00022             :
00024             :L     KB 2        SLAVE 2
00026             :T     MB 20
00028             :L     KB 21       WORK-DB 21
0002A             :T     MB 21
0002C             :
0002E             :SPA FB 22         CALL FOR SLAVE 2
                 NAME #SLAVE_N
                 SLAV =MB 20
                 DBNR =MB 21
                 TIPP =MB 23
00038             :
0003A             :                  ADD PARAMETERS FOR ADDITIONAL SLAVES HERE
0003C             :
0003E             :BE
```

FB22

Module FB22 is called once per cycle and for every slave from the module FB21. The slave number and the slave-DB number are transferred in designators SLAV and DBNR. All modules called in FB22 supervise the entire set of drive-related functions that can be initiated by the PLC or by the SERCOS-module.

```
          BAUSTEIN#FB22
          BSTNAME  #SLAVE_N
          BIB      #10084
          BEZ      #SLAV E:BY
          BEZ      #DBNR E:BY
          BEZ      #TIPP E:BY

0001E             :L   =SLAV
00020             :T   MB 10
00022             :L   =DBNR
00024             :T   MB 11
00026             :L   =TIPP
00028             :T   MB 12
0002C             :SPA FB 6
          NAME    #ACHSE       read or write configuration data for a slave
          SLAV    =MB 10
          SSNR    =KY 0,0
          DBST    =DB 10
          DBDA    =MB 11
          PAFE    =MB 198
0003C             :SPA FB 7
          NAME    #ANTR_EIN    slave turn on,stop enable (via control word)
          SSNR    =KF +0
          SLAV    =MB 10
          DBST    =DB 10
          DBDA    =MB 11
          STSP    =M 7.7
          PAFE    =MB 198
0004E             :SPA FB 3    read or write nominal/actual values from a slave
          NAME    #DAT_ACHS
          SSNR    =KY 0,0
          SLAV    =MB 10
          DBST    =DB 10
          DBDA    =MB 11
          PAFE    =MB 198
00060             :SPA FB 8    change of operating mode for a slave
          NAME    #WECHSEL
          SSNR    =KF +0
          SLAV    =MB 10
          DBST    =DB 10
          DBDA    =MB 11
          PAFE    =MB 198
00072             :SPA FB 11   single-step operation for a slave
          NAME    #TIPP
          SSNR    =KY 0,0
          SLAV    =MB 10
          DBST    =DB 10
          TIP+    =M 12.0
          TIP-    =M 12.1
          GES+    =M 12.2
          GES-    =M 12.3
          TIME    =T 64
          PAFE    =MB 198
0008A             :BE
```

FB23

Module FB23 contains the entry points for loading and starting a specific NC-program. You can find the exact allocation of data block DBAR in chapter 4.1.13.

```
          BAUSTEIN#FB23
          BSTNAME #PROGALL
          BIB     #16105
0000A             :
0000C             :L     KB 13       use DB13
0000E             :T     MB 104
00010             :L     KB 16       name located from DW16
00012             :T     MB 105
00014             :L     KB 0        name has 8 characters
00016             :T     MB 106
00018             :L     KB 1        axis-group 1
0001A             :T     MB 107
0001C             :L     MB 110      control of load and start
0001E             :T     MB 108
00020             :
00022             :
00024             :SPA FB 13
          NAME #PROG_N
          SSNR =KF +0
          DBAR =MB 104
          ABDW =MB 105
          LANG =MB 106
          PRNR =MB 107
          STEU =MB 108
          PAFE =MB 197
0003C             :
0003E             :BE
```

Appendix

Appendix

A Technical Data

Basic module

Supply voltage	+5 V +/-5%
Current consumption (no options)	1,8 (BG86)
	2,3 (BG86F)
Load voltage for options	24 V DC +/-10%
Processor	CPU80486SLC
Clock frequency	33 MHz
Main memory	4 MB with parity
Video interface	VGA, 16 Bit, 512 KB, max. 1024 x 768 pixels
	Connector for VGA-, RGB-monitor (length of cable for BAS and RGB: 250 m)
Keyboard	Standard AT type
Serial interfaces	COM1: RS-232 (V.24)
Parallel interfaces	LPT1 (Centronics via standard interface plug)
Hard disk interface	Standard IDE
Chip based silicon disk	Sockets for 8 DIP-ICs, 32 pin
CP-Interface	16 page frames of 1 K x 8
Watchdog	triggerable
Copy protection	Serial number and copy protection circuitry
AT-Bus	ISA-96 bus, 16 Bit
System BIOS	QUADTEL according to VIPA specifications
VGA-BIOS	Cirrus Logic according to VIPA specifications

Dimensions

Height	233.4mm
Depth	160,0mm
Space requirements	1 plug-in location (BG86)
	2 plug-in locations (BG86F)

Environmental conditions (without options)

	Operational	Storage/transport
Temperature	0 °C to 55 °C	-20 °C to 70 °C
Temperature variations	20 °C/h	20 °C/h
Relative humidity	95 % at 25 °C	95 % at 25 °C
Altitude above sea level	-300 m to 3.300 m	-300 m to 13.000 m

Hard disk option (OEM version only)
Environmental conditions:

	Operational	Storage/transport
Temperature	5 °C to 50 °C	-40 °C to 70 °C
Temperature variations	20 °C/h	20 °C/h
Relative humidity	10 % to 90 %	10 % to 90 %
Altitude above sea level	-300 m to 3.300 m	-300 m to 13.000 m
Shock		
1/2 Sine, 11ms	5 G	100 G
Vibration		
1 Octave/min., 10-400Hz	1 G	5 G

B Initialisation data during phase run-up

The initialisation data consists of certain communication parameters that are required for the statrt-up of the respective phase. In phase 2 initialisation data is transferred as essential data via the service channel. (refer also to Sercos specification "Data terms", "Initialisation data".)

Ident-numbers that are accessed in phase 2	
reading	**writing**
S-0003	S-0001
S-0004	S-0002
S-0005	S-0006
S-0088	S-0089
S-0090	S-0008
S-0096	S-0007
	S-0009
	S-0010
	S-0015
	S-0032
	S-0033
	S-0034
	S-0035

One exception is the drive supplied by "Bautz". All ident-numbers of the drive of this manufacturer are initialised when it is started. It is not possible to configure the "Bautz" drive in phase 4.

C Operating system error messages during phase run-up

All errors and messages are displayed as plain text by the Sercos master software.

An exception to the above statement are messages that are generated when an error occurs during phase run-up. These errors are displayed in hexadecimal and explained in the following table.

Error-Code	Error message
0x0100	Not enough memory
0x0101	Interrupt vector error
0x0102	Handshake timeout cyclic telegram
0x0103	Drive telegram omitted
0x0104	Phase error (incorrect phase)
0x0105	Incorrect drive address
0x0106	Incorrect ring
0x0107	Illegal number of drives
0x0108	Invalid drive number
0x0109	Overflow Sercos-dual-ported-RAM
0x010a	Service channel not initialised
0x010b	Service channel transfer not completed
0x010c	No service channel transfer data
0x010d	Error during the calculation of T2
0x010e	Error during the calculation of T3
0x010f	Error during the calculation of T4
0x0110	Error during the calculation of TEND
0x0111	No service channel data
0x0112	Handshake timeout during service channel communications
0x0113	Not ready for service channel communications
0x0114	Not ready for BUSY-timeout communications

D Index of figures

E Table index

F Index

M. Stich