Name of company:

Address:

System Concerned:

System Identification:

Remarks:

Have you participated in a Festo Seminar? ☐ Yes ☐ No

FESTO

Electronic Department
Documentation

Please fill in the address of your nearest Festo office

Argentina:
Festo Pneumatic S.A.
Edison 2392
1640 Martinez –
Pcia. de Buenos Aires

Australia:
Festo Pty. Ltd.
179-187 Browns Road, P.O. Box 261
Noble Park, Vic. 3174 (Melbourne)

Austria:
Festo Maschinenfabrik Ges.m.b.H.
Lützowgasse 14
1141 Wien

Belgium:
Festo Pneumatic Belgium SPRL
101, Rue Colonel Bourg
1040 Bruxelles

Brazil:
Festo Máquinas e Equipamentos
Pneumáticos Ltda.
Av. Pereira Barreto, 1286
Cx. Postal 2554
09190 Santo André-SP

Canada:
Festo Inc.
215, Carlingview Drive
Rexdale (Toronto) Ontario
M9W 5X8

Denmark:
Festo ApS
Islevdalvej 180
2610 Rødovre

Federal Republic of Germany:
Festo KG
Postfach 6040
Ruiter Straße 82
7300 Esslingen 1

Finland:
Festo Oy Ab
P.O. Box 86
Valimotie 6
01510 Vantaa

France:
Festo E.U.R.L.
5, rue Montgolfier
93116 Rosny-sous-Bois Cedex

Great Britain:
Festo Ltd.
7, High Street
Teddington, Middlesex TW11 8EH
(London)

Hong Kong:
Festo Pneumatic Ltd.
6/F. New Timely Fty. Bldg.
497 Castle Peak Road
Kowloon, Hong Kong

Iran:
Festo Pneumatic SK
P.O. Box 1518/1485
Teheran

Italy:
Festo S.p.A.
Via Enrico Fermi, 36/38
20090 Assago (MI)

Please let us know how you find this manual in respect of the following points:

Was this manual

very helpful	helpful	satisfactory	
☐	☐	☐	for a system overview
☐	☐	☐	for commissioning the system
☐	☐	☐	for programming the system
☐	☐	☐	for using operands
☐	☐	☐	for maintenance
☐	☐	☐	for locating errors

Any suggestions? .....

.....

If you have any questions concerning our product, do not hesitate to contact your nearest Festo office.

Japan:
Festo K.K.
5993 Shin-yoshida cho
Kohoku-ku, Yokohama 223
Kanagawa

Korea:
Festo Korea Co. Ltd.
470-9 Garibong-dong, Kuro-ku
Seoul

Malaysia:
Festo Pneumatic Sdn. Bhd.
99L, Jalan Tandok
Off Jalan Bangsar, Kuala Lumpur

Mexico:
Festo Pneumatic S.A.
Apartado Postal 119
Av. Ceylán 3, Col. Tequesquinahuac
54030 Tlalnepantla (Mex. City)
Edo. de México

Netherlands:
Festo B. V.
Graaf Willem II Laan 16
2645 AJ Delfgauw (Delft)

Norway:
Festo AB
Østensjøveien 27
Oslo 6

Philippines:
Festo Inc.
La Fuerza Bldg.
2241 Pasong Tamo
Makati, Metro Manila

Singapore:
Festo GmbH & Co.
No. 28, Third Lok Yang Road
Jurong, Singapore 2262

South Africa:
Festo Proprietary Ltd.
22-26 Electron Avenue
P.O. Box 255
Isando 1600 (Johannesburg)
Transvaal

Spain:
Festo Pneumatic S.A.
Avda. de la Gran Vía, s/n
08908 Hospitalet de Llobregat
(Barcelona)

Sweden:
Festo AB
Stillmansgatan 1
P.O. Box 21038
20021 Malmö

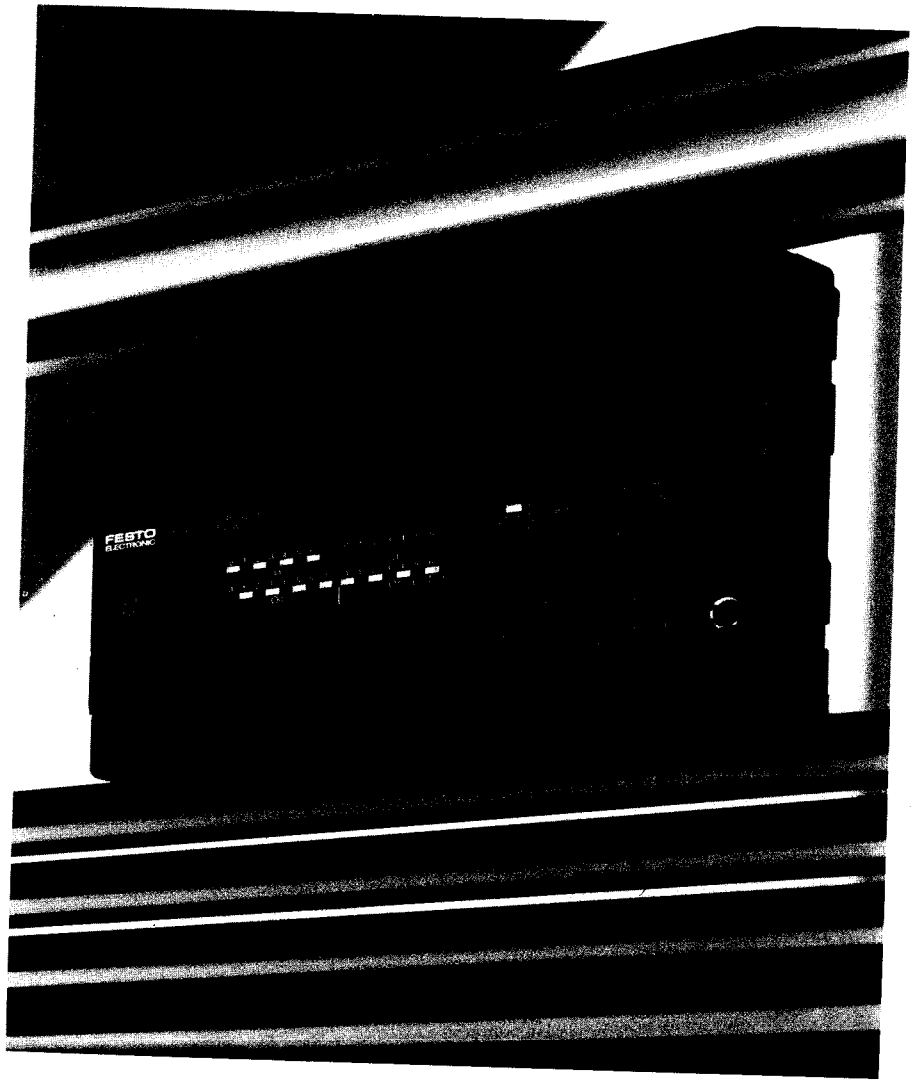
Switzerland:
Festo AG
Moosmattstrasse 24
8953 Dietikon-Zürich

Taiwan:
Festo Co., Ltd.
42 Sung Chiang Road
P.O. Box 32-60
Taibei 104 27

Thailand:
Technicom Engineering
18 Soi Pipat, Silom Road
Bangkok 10500

USA:
Festo Corp.
395 Moreland Road
Hauppauge, N Y. 11788

Festo FPC 202C
Program Logic Controller
Operating Manual



FESTO
ELECTRONIC

8392 GB

Author: A. Lücke

Editor: M. Holder

Translation: D. Smith

Type setting and layout: B. Durz

This operating manual contains the necessary information on installation, operation and maintenance of the FPC 202C. Instructions on programming the controller are also contained.

The following symbols and conventions have been adopted for the descriptions of commands and for programming instructions.

{ } Selection brackets

One of the elements contained within these brackets is to be selected.

< > Comment brackets

These brackets contain a description or substitute term for the variable to be entered.

[ ] Optional brackets

The information contained within these brackets can, if desired, be programmed.

Upper-case letters in the command name

The letters on the keyboard which represent command names are printed in upper-case letters in the text.
e.g.: NOT, RUN, . . .

Lower-case letters in the command name

The letters required to complete the full command name are printed in lower-case letters in the text.
e.g.: SET Output, DELete program, . . .

CONTENTS

CONTROL CHARACTERISTICS

The FPC 202C controller has been designed for small and medium control tasks. These tasks are programmed with a PC which must be connected to the controller. Programs and program modules are created in statement list (STL) and ladder diagram (LAD) with the aid of the programming software FST202C.

The controller offers the user:

Data memory:

- 32Kbyte RAM or
  32Kbyte EPROM

Memory organization:

- 8 user programs
- 8 program modules
- 256 (max.) function modules
- 4 libraries

Inputs/outputs:

- Controller
  16 inputs
  8 transistor outputs
  8 relay outputs (exchangeable relay)
- Extension device
  32 inputs and outputs per device
  3 devices can be connected
- Plug-on terminal strips for inputs and outputs

Operands:

- 32 timers (0.00s to 655.35s), either as impulse timers or timers with switch on/switch-off delay
- 32 counters (up to 65535 counting cycles), either as increment or decrement counter
- 64 registers
- 256 flags in 16 flag words

Further characteristics:

- Two programs/modules can run at the same time
- Compiled programs can be processed
- Programmable automatic start
- Lockable keyboard
- Integrated diagnostic system (command interpreter)
- Fuses for supply voltage and output stages
- Memory backup by battery
- Battery failure recognition
- Battery replacement without loss of data

STRUCTURE OF THE FPC 202C

There are covers for the following parts of the FPC 202C:

1) Input terminals
2) Output and supply terminals
3) Fuses and battery
4) Memory modules and plugs for extension device

All these covers can be removed after raising at the points marked with arrows.

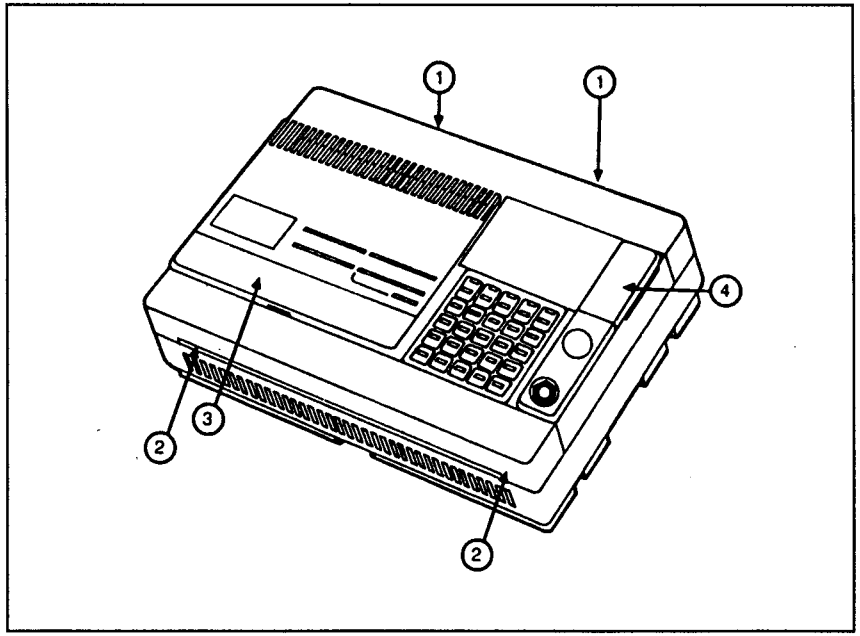


Fig. 1/1: Cover parts of FPC 202C

SYSTEM STRUCTURE

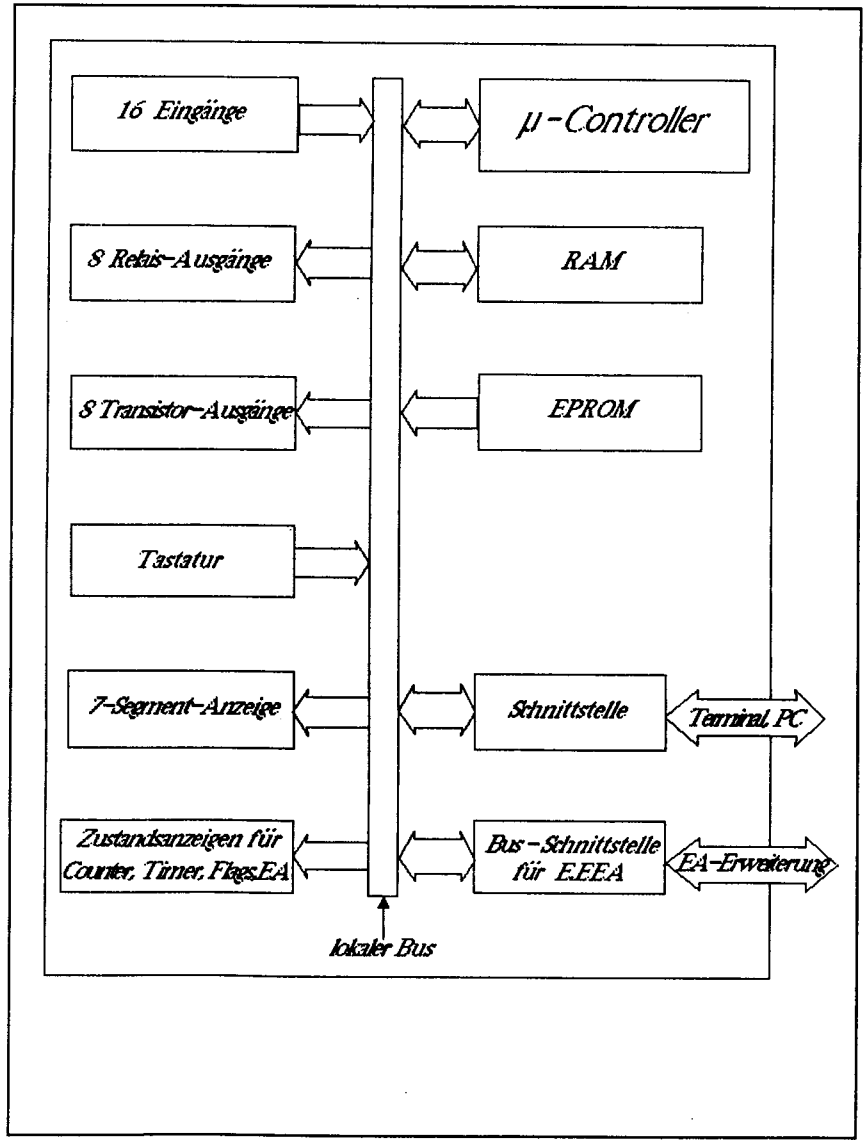
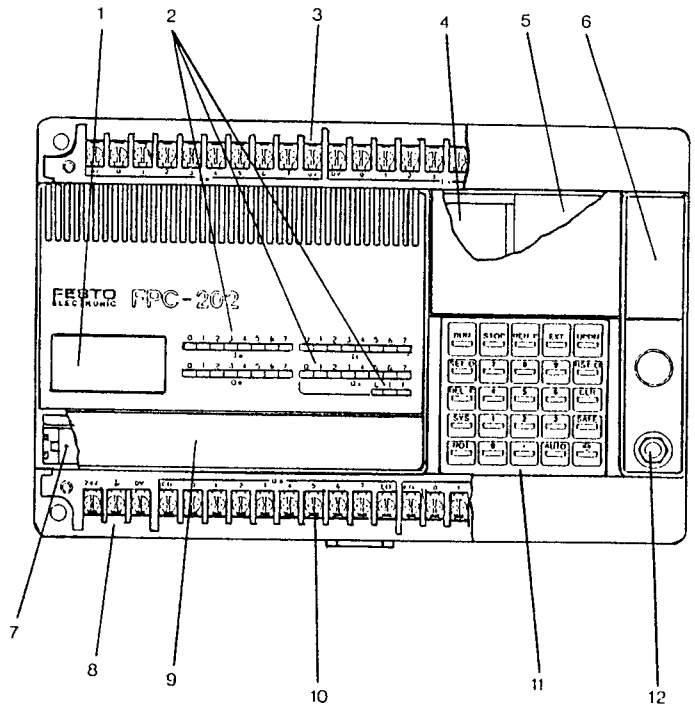


Fig. 1/2: System structure of the FPC 202C

LOCATION OF PARTS



1. 7-segment display
2. LED display for inputs/outputs, counters, timers, flags
3. 16 inputs
4. System memory
5. User EPROM
6. Bus plug for I/O extension device
7. Battery with backup connection
8. Connections for supply voltage
9. Fuses
10. 16 outputs
11. Keyboard
12. Diagnostic port (mod. 20 mA interface)

Fig. 1/3: External structure of the FPC 202C

EXTENSION OF THE INPUTS AND OUTPUTS

Up to three E.EEA 202 I/O extension devices can be connected via the bus plug under cover 4. In this way, the 32 inputs and outputs of the CPU can be extended to 128 inputs and outputs.

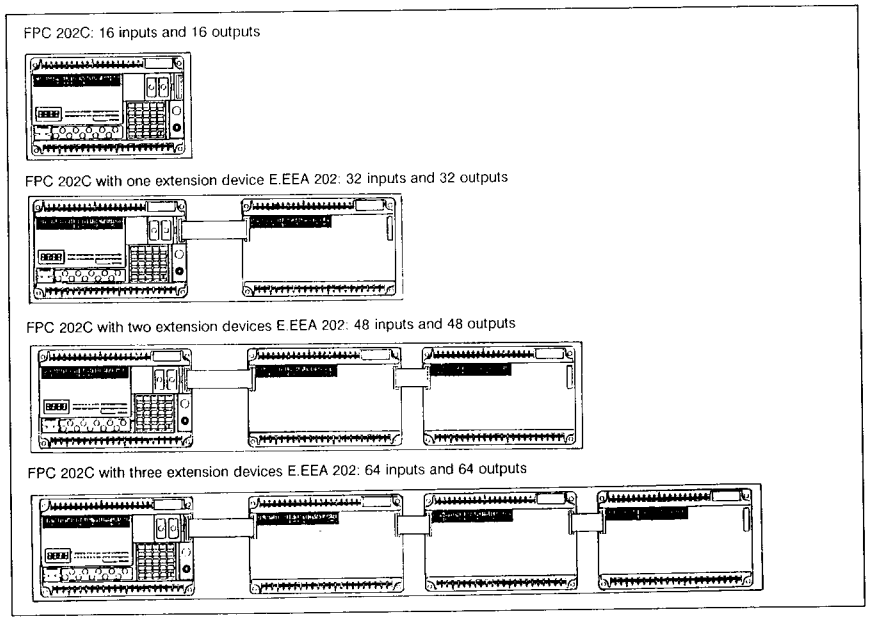


Fig. 1/4: Connecting the extension devices

CONNECTING DIALOG DEVICES

On the right next to the keyboard there is a modified 20mA current-loop interface. The FPC 202C can be connected to various devices via this interface.

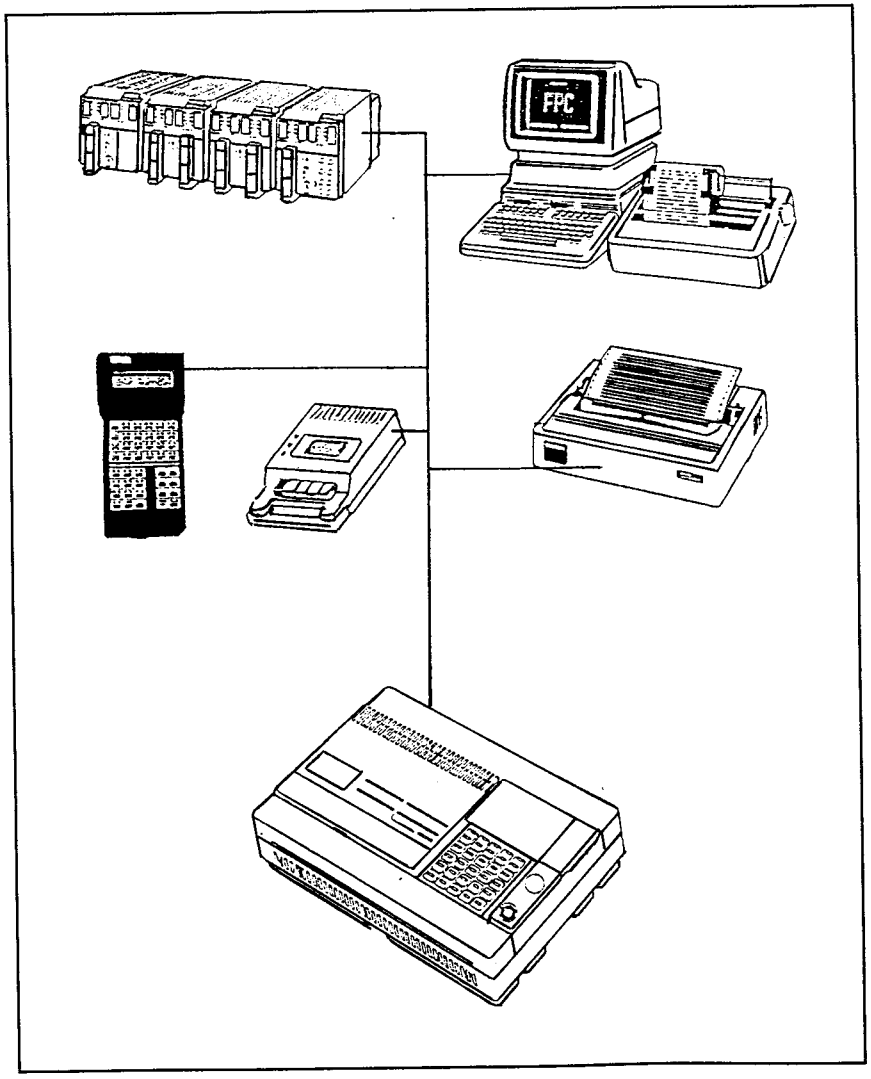


Fig. 1/5: Connecting dialog devices

Notes

2

INSTALLATION INSTRUCTIONS

MECHANICAL INSTALLATION

ELECTRICAL INSTALLATION

This chapter contains basic requirements for commissioning the FPC 202C

INSTALLATION INSTRUCTIONS

In order that the FPC 202C can function properly, the following requirements must be fulfilled:

Ambient parameters

Ambient temperature	between 0°C and 55°C • Check that there is sufficient room for air circulation. • If necessary, provide ventilation fan. • Do not install the controller over heater, transformer or load resistors. • Avoid fluctuations in temperature which can cause condensation.
Humidity (relative)	between 0% and 95% (non condensing)
Atmosphere	Avoid: • corrosive, potentially-explosive, extremely dusty atmosphere; • heavy contamination by salt and iron particles; • water and chemical sprays.
Mechanical stress	Avoid: • vibration and knocks of more than 3.36g.

Electromagnetic interference

Cables which can transmit electrical or electromagnetic interference, should not be placed nearer than 100mm to the housing.

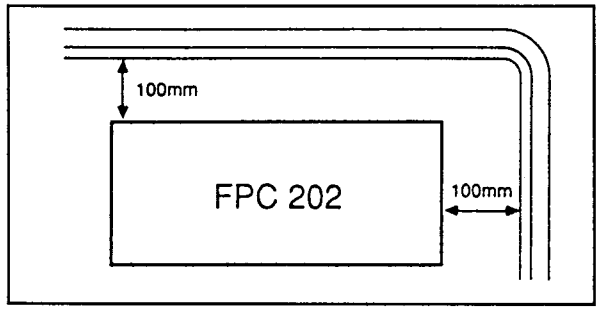


Fig. 2/1: Electromagnetic interference cable

Interference suppression measures

If the electrical elements controlled by the FPC 202C (electromagnetic relays, solenoid valves, etc.) create interference, suitable measures must be taken. Ideal elements for suppressing interference signals are varistors, which are suitable for both DC and AC currents.

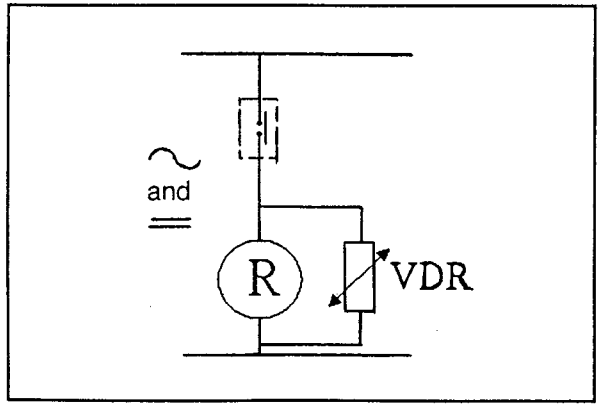


Fig. 2/2: Circuit for interference suppression

AC interference sources require a circuit for suppressing voltage peaks parallel to the coil (see Fig. 2/3).

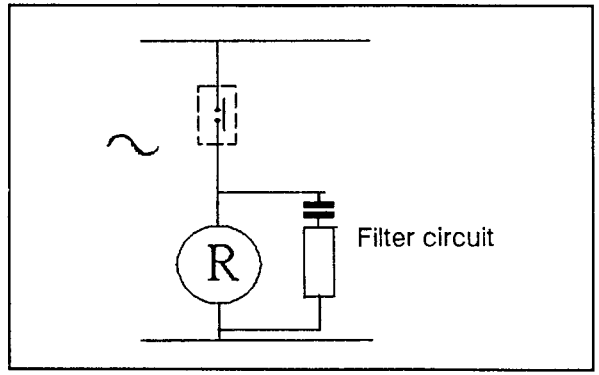


Fig. 2/3: AC interference source

DC interference sources can be suppressed by a circuit with resistor and free-wheeling diode (see Fig. 2/4).

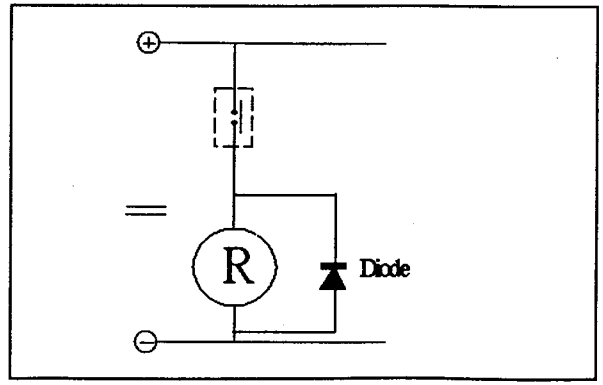


Fig. 2/4: DC interference source

Cables

To avoid the complicated laying of cables and to protect the cables used, we recommend the use of cable channels. If additional cables carrying 220V/10A and more are required by the machines or systems to be controlled, the following must be observed:

- 300mm minimum distance between parallel cable channels,
- insulation from the voltage supply by means of earthed metal plates.

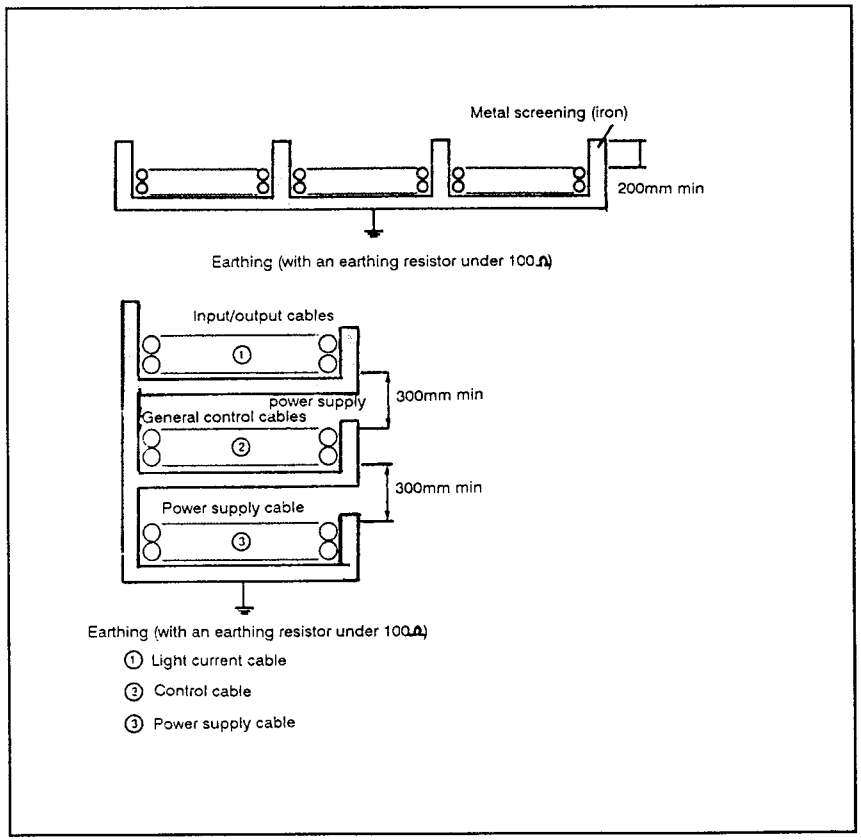


Fig. 2/5: Cabling of the controller

MECHANICAL INSTALLATION

Fitting possibilities

The FPC 202C can be fitted in the following ways:

- Front fitting
- On support rails

The requirements for these types of fitting are already incorporated in the FPC 202C housing. When fitted from the front, the FPC 202C is fastened directly to the four corners of the housing. When the controller is fitted on support rails, the guide rails in the floor of the housing should be used.

Front fitting

There are four holes for this type of fitting under covers 1 and 2 (see Fig. 1/1) in the corners of the housing. 45mm (M5) screws should be used here.

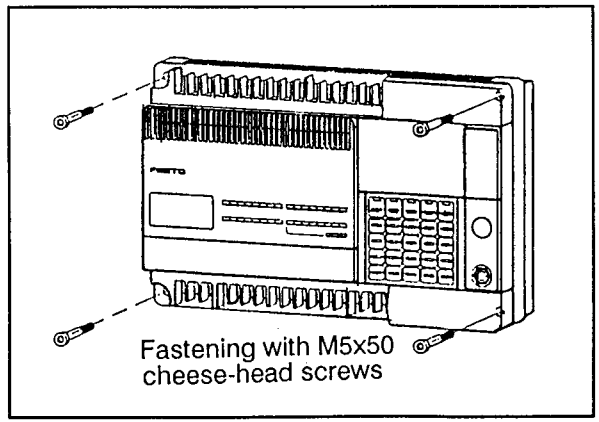


Fig. 2/6: Front fitting

Fitting on support rails

The guide rails and the red plastic slide on the floor of the housing are intended for this type of fitting.

Fitting

- Hang the controller by fitting the upper guide rail into the support rail.
- Press the lower edge of the controller gently against the support rail until it is heard to clip into position.

The spring plastic slide fastens the FPC 202C on the support rail.

Dismantling

- Insert a screwdriver into the lower horizontal slot of the red slide.
- Press the slide downwards.
- Gently raise and remove the controller.

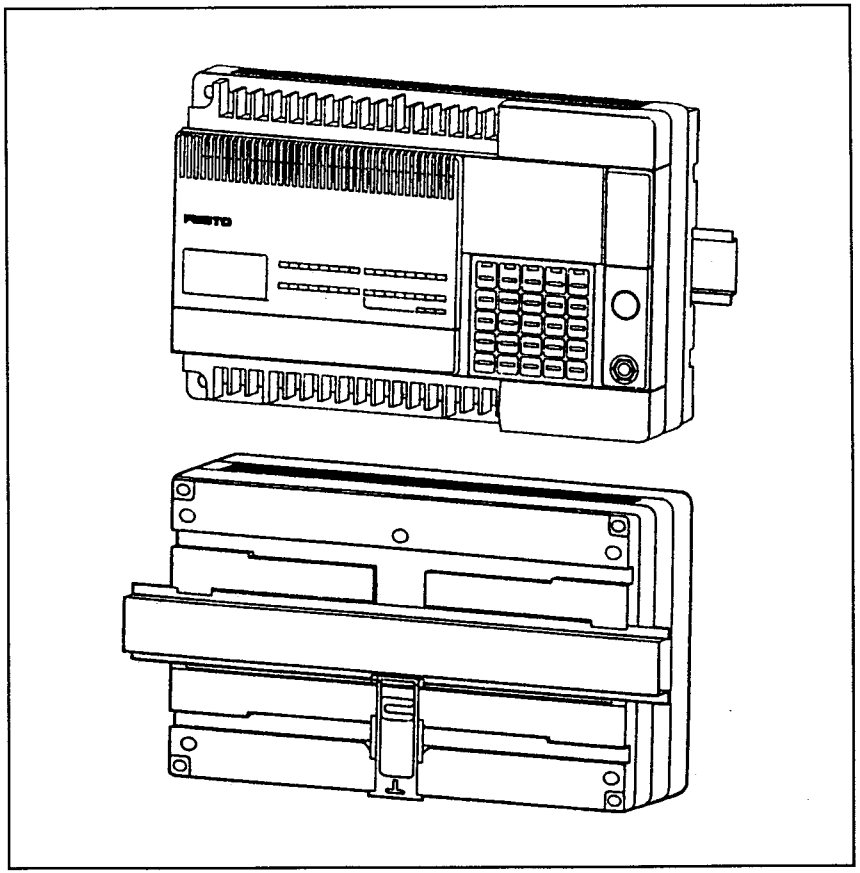


Fig. 2/7: Fitting on support rails

Replacing terminal strips

To enable peripheral devices (motors, sensors, etc.) to be connected more quickly and more flexibly, replaceable terminal strips can be plugged into the existing input/output connections.

Fitting the terminal strips

1. The relevant covers must be removed before the terminal strips are fitted:
   - Cover 1 (see Fig. 1/1) for input terminals
   - Cover 2 (see Fig. 1/1) for output and supply terminals
2. Plug in the appropriate terminal strip and tighten by equally turning the two knurled screws.
3. Replace the relevant cover.

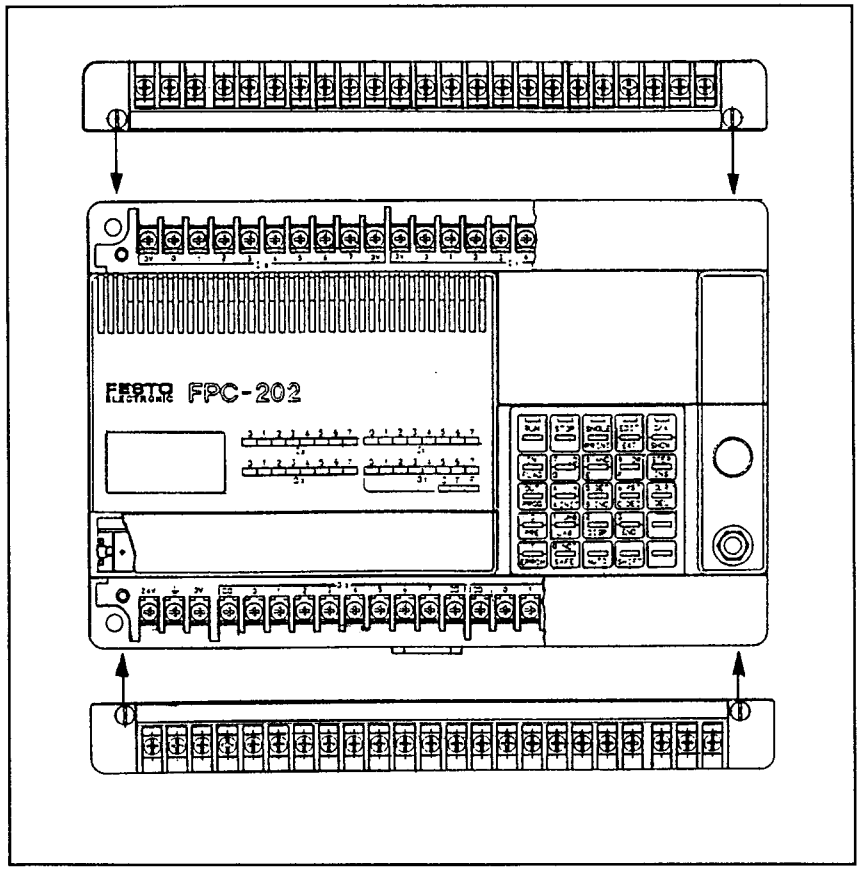


Fig. 2/8: Fitting the terminal strips

Notes

ELECTRICAL INSTALLATION

Memory modules

In the standard version of the FPC 202C a fixed RAM module with 32Kbyte serves as a user memory. Memory space can be extended by means of an EPROM module with a 32Kbyte memory. In this case, the RAM or EPROM memory module serves as the user memory.

Fitting the EPROM memory module

Please observe the precautionary measures for dealing with MOS components.

1. Switch off the 24V DC supply voltage.
2. Remove the right-hand upper housing cover of the controller.
3. Insert the user EPROM as follows:
   - Place the EPROM in the right-hand socket.
   - The U-shaped module marking must lie above the appropriate socket marking.
   - Press in the memory module carefully.
4. Check that the EPROM is positioned correctly.
5. Set the memory type on the switch specially provided for this purpose.

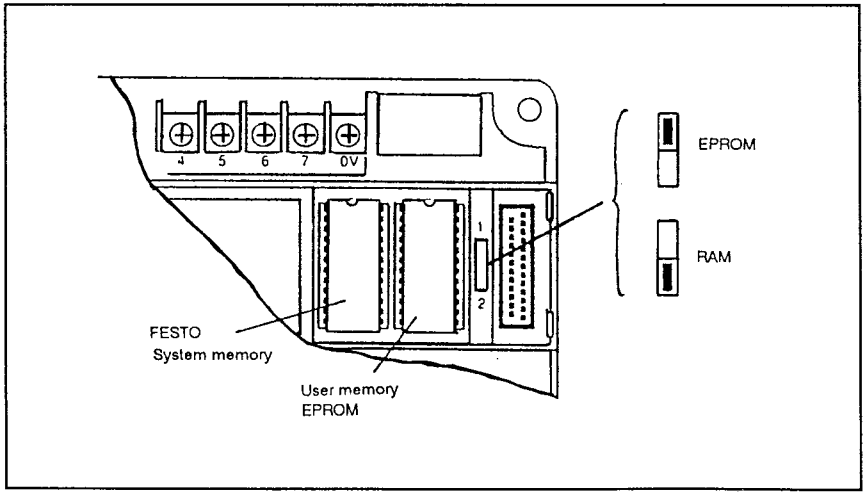
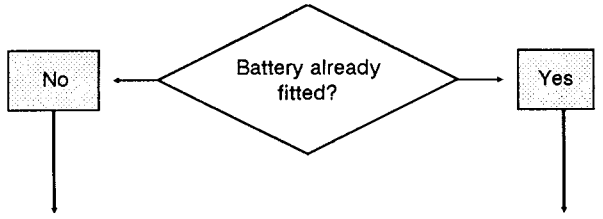


Fig. 2/9: Fitting EPROM and setting memory type

Battery

The contents of the RAM memory module (user memory) are saved by an applied voltage. This is carried out automatically when the supply voltage remains switched on. If the voltage fails or if it is switched off, the battery voltage automatically saves the memory contents. When used at 25° C, the battery has a service life of 5 years as from the date of manufacture.

The RAM backup battery should be fitted as follows:



Fitting the battery

- Remove cover 3 (see Fig.1/1)
- Fit the battery with correct polarity

Replacing the battery

- See Chapter 7

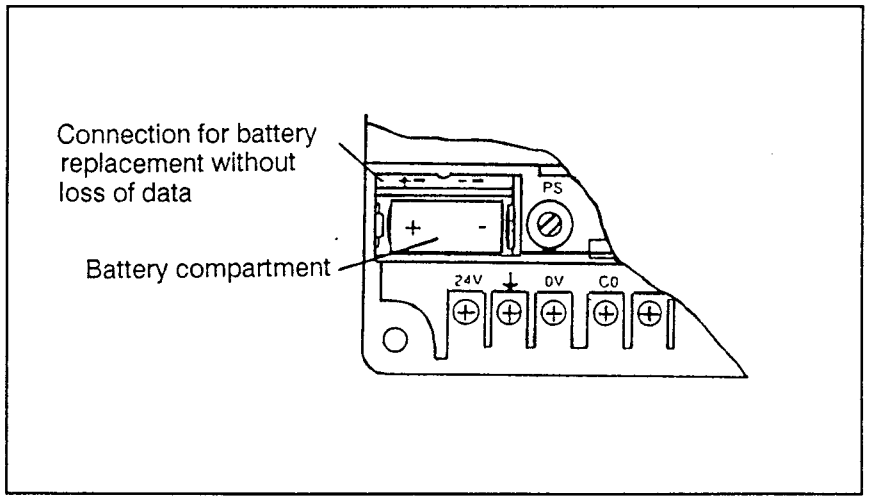


Fig. 2/10: Inserting the battery

Supply voltage

A supply voltage with the following specifications is required for operating the FPC 202C:

Supply voltage rated value tolerance range	 DC 24 V - 25 %; + 25 %
Current capacity typical maximum	 165 mA 390 mA
Power consumption maximum	 7.2 W

The suppply voltage is applied via the terminals provided under cover 2 (see Fig. 1/2).

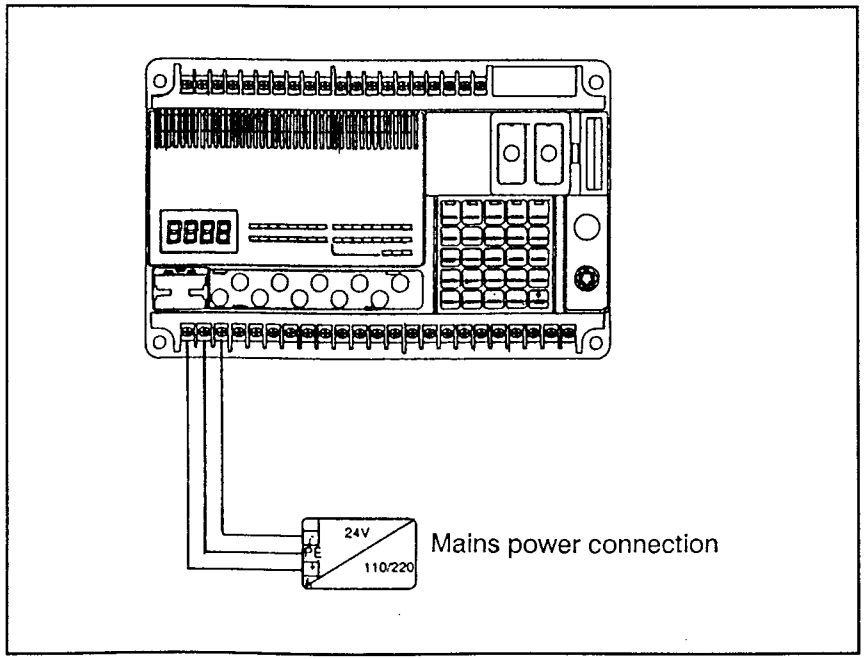


Fig. 2/11: Applying the supply voltage

Applying the supply voltage

Both the power unit and the controller must be switched off before the terminals are connected.

1) Connect the following:
   - the earth cable of the power unit with the 0V-terminal of the controller,
   - the earth connections of the power unit with the FPC 202C (mains filter improvement),
   - the positive pole of the power unit with the 24V-terminal of the controller.
2) Connect the power unit to the mains. The red LED above the STOP key should now light up and Error 51 (Er51 = Battery is empty, missing or was missing) should appear on the display.
3) If this is not the case:
   - switch off the supply voltage again,
   - check the cable connection between the power unit and the controller,
   - check the PS fuse under cover 3 (see Fig. 1/1).

Inputs and outputs

The FPC 202C has 16 inputs, 8 relay outputs and 8 transistor outputs. Sensors and actuators are connected via the appropriate terminal strips.

Assigning the inputs and outputs

The terminal strip for the inputs is assigned as follows:

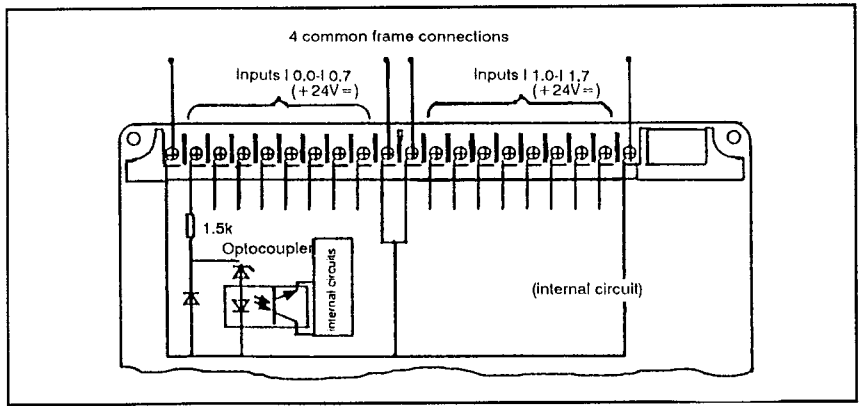


Fig. 2/12: Assigning the inputs

The terminal strip for the outputs is assigned as follows:

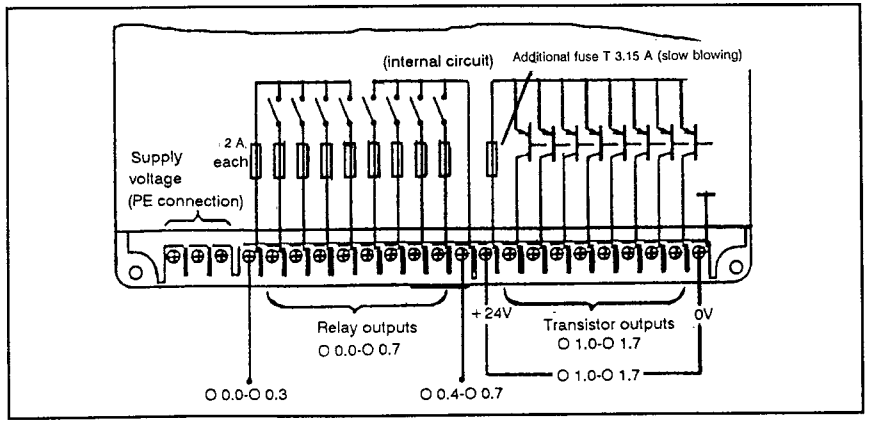


Fig. 2/13: Assigning the outputs

Circuitry of the inputs and outputs

In order that the control signals can be received, the sensors and actuators must be connected in accordance with the circuit diagram below.

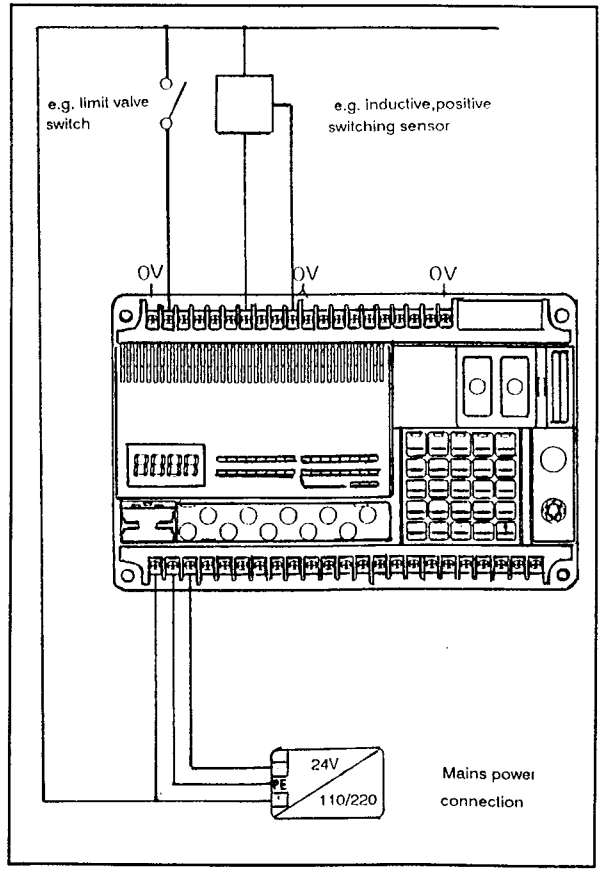


Fig. 2/14: Connecting sensors

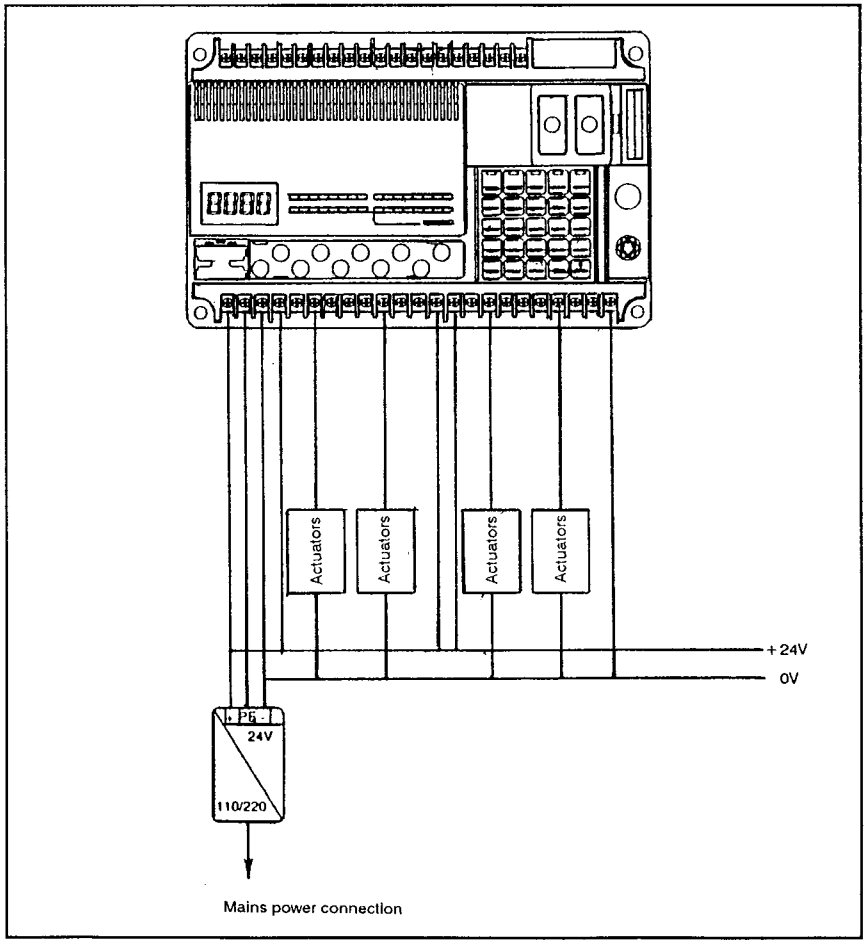


Fig. 2/15: Connecting actuators

3

KEYBOARD

DISPLAYS

SAVING PROGRAMS ON EPROM

KEYBOARD

The keys on the FPC 202C keyboard are assigned at least once. The commands are executed by pressing the appropriate key and then the return key.

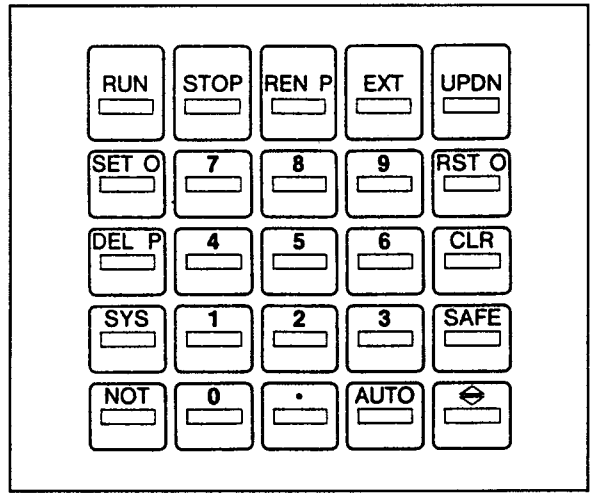
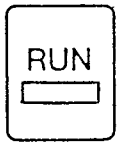


Fig. 3/1: FPC 202C keyboard

Return key

Entries are concluded with the return key (except for the commands CLR and STOP). This key is also called <CR> (carriage return).

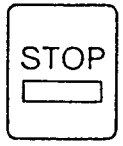
The individual commands



Starting and stopping programs

User programs are started manually with the RUN command. The program number must first be specified. If the program number is not specified, the program with the lowest number will be started.

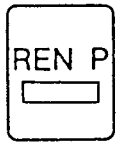
Input format:
RUN [program number] <CR>
program number = {0, . . . , 7}



The STOP command does not require a parameter and must not be concluded with <CR>.
The STOP command stops all active programs and modules.

Input format:
STOP

Renaming and deleting programs



An existing program can be renamed with the REName command.
The new program number must not be assigned already.

Input format:
REN P <old program number> <CR>
<program number> <CR>
program number = { 0,..., 7}

DEL P

The DELete command deletes an existing program. This command is protected against unintentional deleting. The user must press the <CR> key twice in order to execute the command.

Input format:

Del P <program number> <CR>
<CR>

program number = { 0, . . . , 7}

Example

Entry	Display	Comment
DEL P 6	6	Command and program entry
<CR>	dP 6	Command check
<CR>		Delete program no.6

Setting and resetting outputs

SET O

The SET Output command sets a specified output from 0 to 1-signal. Setting an output already set has no effect.

Input format:

SET O<word number>.<bit number><CR>

Word number = { 0, . . ., 7}

Bit number = { 0, . . ., 7}

Example

Entry	Meaning
SET O 1.6 <CR>	Output 6 in output word 1 is set

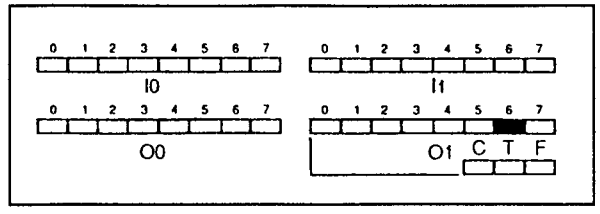


Fig. 3/2: SET O 1.6

The ReSeT Output command resets an active output from 1 to 0-signal. Resetting an inactive output has no effect.

Input format:

RST O <word number>.<bit number><CR>

Word number = {0, . . . , 7}

Bit number = {0, . . . , 7}

Example

Entry	Meaning
RST O 3.4	Output 4 in the 3rd. output word (1st. extension device) is reset.

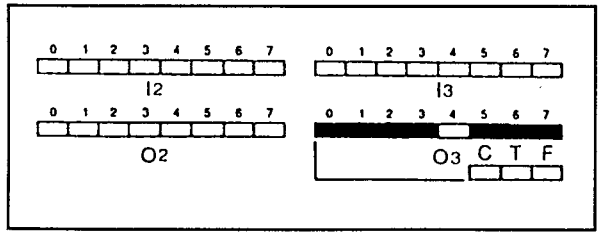
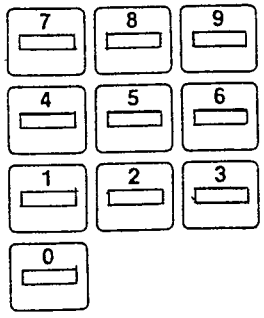


Fig. 3/3: RST O 3.4

Number block



Digits and numbers must be entered for:

- Program number

  Example

  RUN 5 <CR>

  REN P 7 <CR> 6 <CR>

  DEL P 4 <CR> <CR>

- Outputs

  Example

  SETO6.7 <CR>

  RSTO4.1 <CR>

- Keyboard locking and unlocking

  Example

  .3009 <CR>

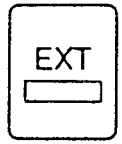
- Baud rate

  EXT4 <CR>

  UPDN2 <CR>

  UPDNNOT9 <CR>

Data transfer via the 20mA interface



The command interpreter of the controller is accessed by means of the EXTernal command (see Chapter 5).

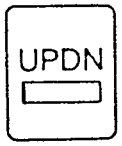
Input format:

EXT[<Baudrate>]<CR>

Baudrate = {1,2,3,4,6,9}

whereby:
- 1 = 1200 Bd
- 2 = 2400 Bd
- 3 = 300 Bd
- 4 = 4800 Bd
- 6 = 600 Bd
- 9 = 9600 Bd (default)

Download - Upload



The UPDN key is assigned with two commands (see also Chapter Program saving on EPROM).

With the DowNload command, data, i.e. for saving on EPROM, is transferred by the controller to an external device. Data transfer is made via the 20mA interface.

Input format:
UPDN<baudrate><CR>
Baud rate = See EXT command

Example

UPDN9<CR>

Prepare a data transfer to an external device with a baud rate of 9600 Baud

The UPload command loads data from an external device into the controller.
Data transfer is made via the 20mA interface.

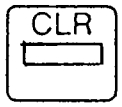
Input format:
UPDNNOT<Baudrate><CR>
Baud rate = See EXT command

Example

UPDNNOT2<CR>

Load data into the controller with a baud rate of 2400 Baud.

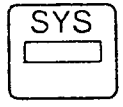
Deleting incorrect entries and error messages



The CLeaR command is used for correcting incorrect entries and deleting error messages, provided these have not already been concluded with <CR>. Although error messages are deleted with CLR, the cause of the error is not necessarily eliminated.

Input format:
CLR

Initializing function units



The SYStem command deletes all:

- timer states, words, preselects
- counter states, words, preselects
- flags
- registers
- inputs and outputs

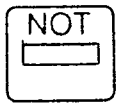
Input format:
SYS

Resetting all outputs

With the SAFE command, all connected outputs are reset (from 1 to 0-signal).

Input format:
SAFE<CR>

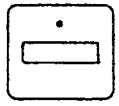
Negation or shift function



The NOT key serves as shift key for the AUTO and UPDN keys which are both assigned with two functions.

Input format:
See AUTO or UPDN command

Divider and locking



This key has two functions:

- Divider between the word number and the bit number with the commands SET Output and ReSeT Output.

Input format:
SETO<wordnumber>.<bitnumber>

- Initiating the locking function.

Input format:
See Chapter: Keyboard locking

Automatic program start

The user programs can be started manually (via RUN) or automatically (via AUTO). The AUTO command only becomes effective when the supply voltage is reapplied.

Activating the automatic program start:

Input format:
AUTO<CR>

Deactivating the automatic program start:

Input format:
AUTO NOT <CR>

Keyboard locking

As a protection against unauthorized use of the FPC 202C, a code can be used to lock the keyboard.

Locking

1) The controller must be in the STOP or RUN status
2) The locking function is selected with the key:

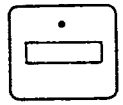
3) The following appears on the display:

4)

Input format:
<numbercode> <CR>
Number code = {1, ... ,9999}

Unlocking

1) The controller must be in the STOP or RUN status
2) The unlocking function is selected with the key:



3) The following appears on the display:

4)

Input format:
<lockingcode> <CR>
Locking code = number sequence specified for locking

Reading the locking code

The locking code can be read with the H$ command under memory address $F580. See Chapter 5: Command interpreter.

DISPLAYS

Communication with the FPC 202C is shown in most cases by the various display possibilities.

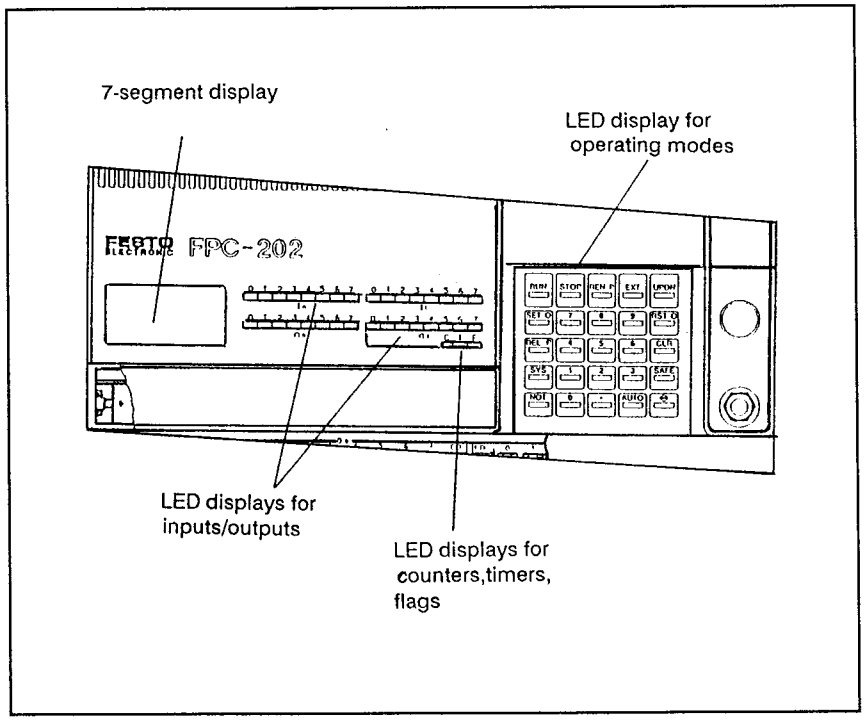


Fig. 3/4: Display possibilities

7-segment display

In addition to error messages, the following commands or controller states are shown on the display of the FPC 202C:

Display	Meaning / Comment
AUTO \NAUT	The automatic program start is programmed/deleted
EXT	The default value and, if necessary, the baud rate are specified for use with the EXT command.
REN/DEL P	A program with specified number is renamed/deleted.
RUN	The program number and type are specified with the appropriate program when the RUN command is entered. The display remains dark whilst the program is running.
SAFE	All connected outputs are deleted.
SET/RST O	The outputs including the word and bit numbers are set/reset.
UPDN	With the UPDN command, a hexadecimal count is made per 1Kbyte up to F during data transmission.
CodE	Locking and unlocking is initiated, selected with divider key

LED displays for inputs and outputs

The signal at an input or output at a particular moment is shown on the relevant input and output LEDs.

LED displays above the keyboard

The five LEDs above the keyboard show the status of the relevant key command.

LED for ... alight	Meaning / Comment
RUN and STOP	The system is always in one of these two states.
REN P and DEL P	This LED lights up when either of the commands REName program or DELete program is activated.
EXT	This LED lights up when the test system is activated.
UPDN	This LED lights up during the UP/DOWNLOAD of the valid user memory.

Notes

PROGRAM SAVING ON EPROM

For data saving, a commercially-available EPROM programmer can be connected to the diagnostic port of the FPC 202C.

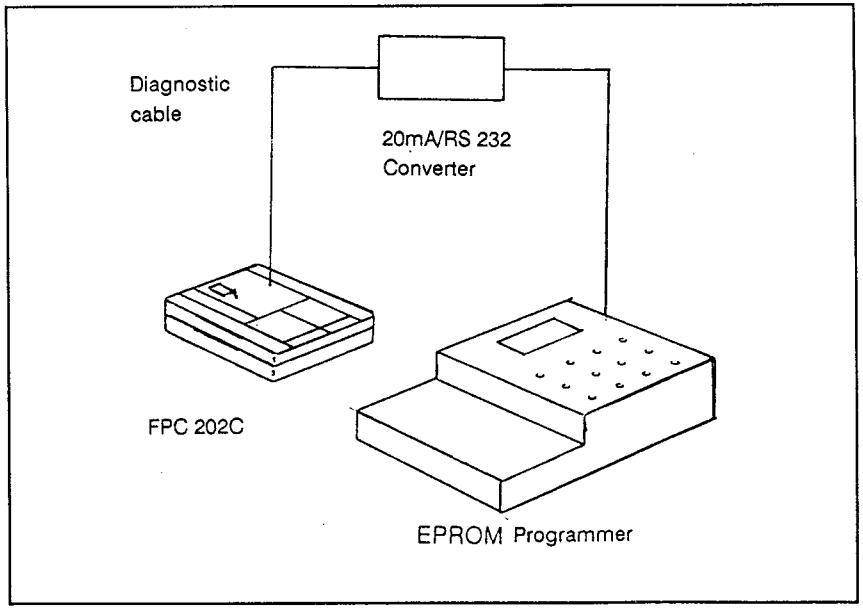


Fig. 3/5: EPROM programming

The controller can be connected to the programmer by means of:

Converter
D.AS-DIAG/RS 232-1
TN 80401

in connection with:

Diagnostic cable
E.KDI-20
TN 8325

The FPC 202C outputs the data of the user memory in INTEL-HEX format with relative addressing. The Xon/Xoff protocol (software handshake) is used here. Only the assigned part of the user memory is transferred to the EPROM programmer.

Before the EPROM programmer is accessed by the FPC 202, the programmer must be set for reception (see the operating instructions with the EPROM programmer).

Accessing the programmer:

UPDN[<baudrate>]<CR>

Baud rate = {1,2,3,4,6,9}

whereby:

1 = 1200 Bd
2 = 2400 Bd
3 = 300 Bd
4 = 4800 Bd
6 = 600 Bd
9 = 9600 Bd (default)

Data transmission can be interrupted at any time by:

EXT

A counter runs in the controller display as a visual check that data is being transmitted correctly. This is incremented for every 256 bytes transmitted.

4

PROGRAMMING SOFTWARE

OPERANDS

TASK STRUCTURE OF THE OPERATING SYSTEM

PROGRAMMING SOFTWARE

The FPC 202C is programmed with the FESTO programming software FST 202C. This software must be loaded onto a personal computer which is connected to the controller.

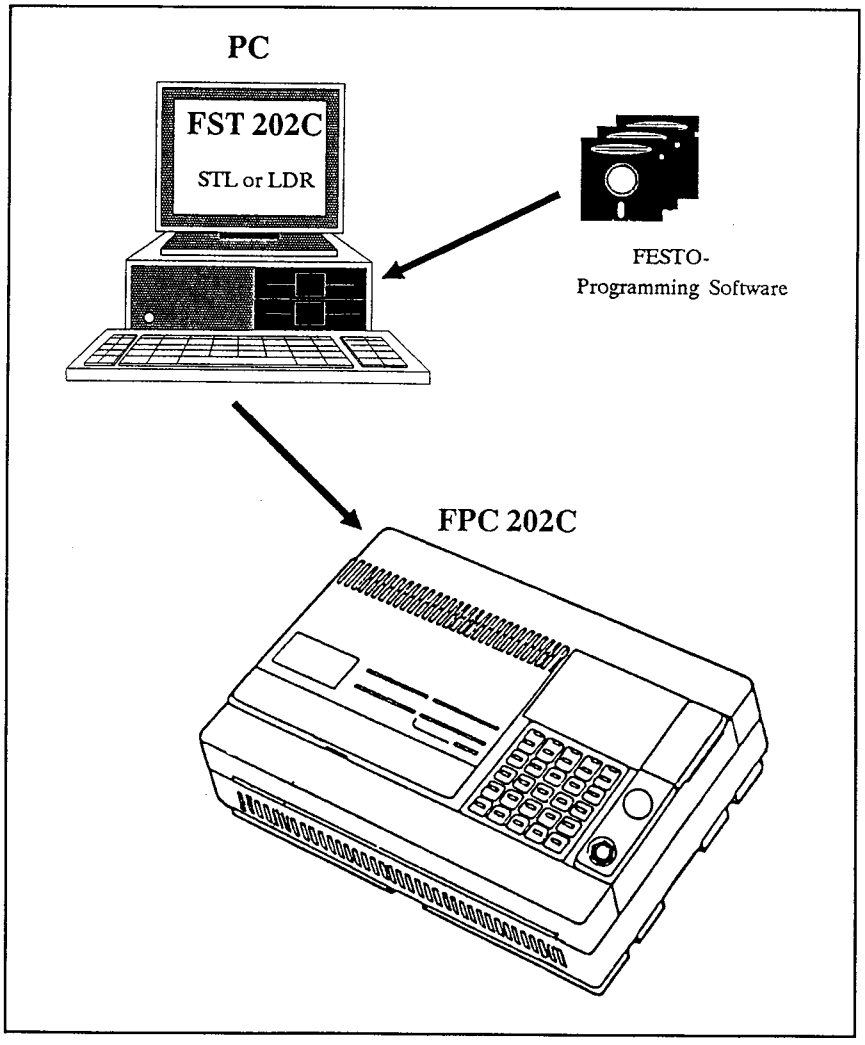


Fig. 4/1: Programming the FPC 202C

The two programming languages statement list (STL) and ladder diagram (LAD) can be used to create user programs and program modules. A description of the programming requirements, programming languages and techniques are to be found in the manuals:

- FESTO Software Tools - Statement list of the FPC 202C - Manual FST 202C

and

- FESTO Software Tools - Ladder diagram of the FPC 202C - Manual FST 202C.

OPERANDS

The table below contains the operands managed by the operating system of the FPC 202C.

Number	Meaning	Designation	Parameter	Comment
64	Inputs	I	{0-7}.{0-7}	Single-bit operand, 8 input words with 8 inputs each
8	Input words	IW	{0-7}	Multibit operand
64	Outputs	O	{0-7}.{0-7}	Single-bit operand, 8 output words with 8 outputs each
8	Output words	OW	{0-7}	Multibit operand
256	Flag	F	{0-15}.{0-15}	Single-bit operand, 16 flag words with 16 flags each
16	Flag words	FW	{0-15}	Multibit operand
1	Init flag	FI	1	Single-bit operand, MI only with LAD
32	Counters	C	{0-31}	Single-bit operand
32	Counter preselects	CP	{0-31}	Multibit operand
32	Counter words	CW	{0-31}	Multibit operand
32	Timers	T/TE/TA	{0-31}	Single-bit operand, all timers can be programmed as impulse timers T, switch-on delay timers TE, switch-off delay timers TA
32	Timer preselects	TP	{0-31}	Multibit operand
32	Timer words	TW	{0-31}	Multibit operand

Number	Meaning	Designation	Parameter	Comment
8	Programs	P	{0-7}	Single-bit operand
8	Program modules	CPM	{0-7}	The CPM are created on the PC in STL or LAD, they are stored in the user memory.
64	Registers	R	{0-63}	Multibit operand
256	Function modules	CFM	{0-255}	The CFM are supplied with the operating system
1	Error	E	1	Single-bit operand
1	Error word	EW	1	Multibit operand
40	Special operands	FU	{0-23} and {32-47}	Multibit operand

Modules

To facilitate programming and to relieve user programs, frequently required command sequences can be programmed in so-called modules (subprograms).

A distinction is made between program modules and function modules, depending on whether these modules are to be programmed by the user himself or whether they are already defined in the operating system.

Both types of modules can be accessed from within an active program. When the module has been processed, the accessing program is then continued from the point at which it was interrupted.

Program modules

Program modules can be created in statement list or ladder diagram with the FESTO programming software. Maximum 8 modules can be stored in the user memory of the FPC 202C.

Function modules

The FPC 202C operating system manages maximum 256 (0 to 255) function modules. These modules are already defined. They are stored in the operating-system EPROM.

Notes

TASK STRUCTURE OF THE OPERATING SYSTEM

The FPC 202C has an operating system capable of multitasking, i.e. it can accomplish several tasks at the same time.

For programming, this means that two of the eight user programs of the controller can be processed at the same time. If one module is also accessed from each of the active programs, programs and modules will then be processed quasi-parallel.

Program processing

A program is started by means of the RUN command (e.g. P1).

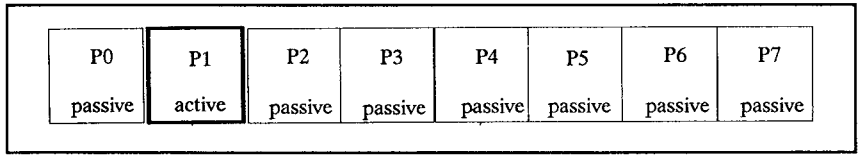


Fig. 4/2: Program start

A second program (e.g. P4) can be started from this active program:

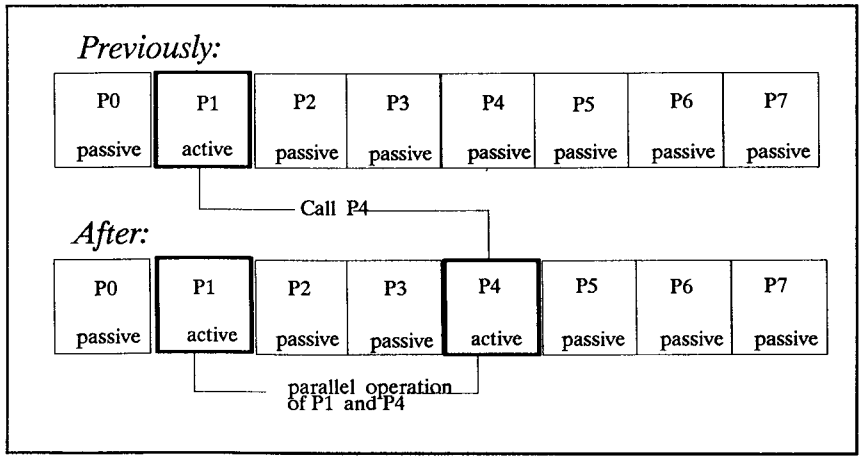


Fig. 4/3: Parallel program processing 1

Both user programs are now processed one after the other. With compiled programs, the processing change takes place after a complete cycle.

A cycle consists of:

- with LAD: a program cycle (= complete LAD program)
- with STL: a program step (from STEP ... to STEP ...)

Maximum two programs can be processed at the same time. If a further program is accessed from within one of the active programs, the accessing program and the newly set program will run parallel, the third will be stopped automatically.

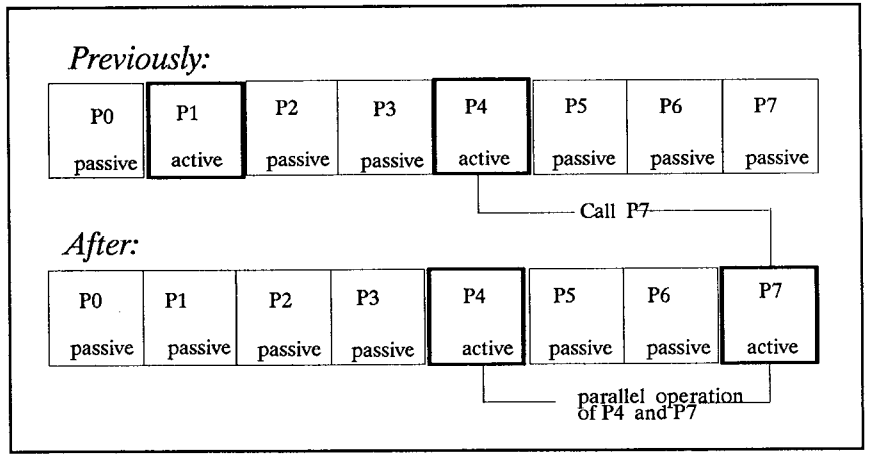


Fig. 4/4: Parallel program processing 2

Module processing

A program module or a function module is accessed in the same way as a subprogram. When the program module or function module has been processed, the user program is continued from the point at which it was interrupted.

Notes

5

SUMMARY

OPERATING INSTRUCTIONS

DESCRIPTION OF COMMANDS

OVERVIEW

The command interpreter is a software packet within the operating system of the FPC 202C and enables the controller to be externally operated. It forms the logical interface between the diagnostic port of the FPC 202C and the operating system in the CPU.

It serves for:

- test, servicing and diagnostic purposes

as well as for:

- (limited) remote control of the FPC 202C.

The command interpreter enables programs, memory areas and function units to be controlled externally.

Please note that, although programs may be entered in English, the German abbreviations for parameters must be used in the command interpreter. The German parameter definitions are listed on page 5-11.

The command interpreter offers the following facilities:

Programs:

- can be started and stopped,
- can be loaded,
- can be deleted together or individually,
- can be modified.
- The checksum can also be formed.

Function units:

- can be modified and displayed.

Memory areas:

- can be displayed, i.e. individual or all programs, the directory or all stored data.

OPERATING INSTRUCTIONS

Connection to a dialog device

Before the command interpreter can be used, the FPC 202C must be connected to a dialog device.

The user can choose between:

- a commercially-available PC
- a commercially-available terminal
- the FESTO hand-held terminal E.TAB

The diagnostic port serves as interface for the FPC 202C. This is a modified 20mA current-loop interface.

Characters are transmitted asynchronously in duplex mode. The baud rate can be adapted to the dialog device when the command interpreter is accessed. The character set which the command interpreter accepts corresponds to the ASCII code (= ISO 7-bit code, DIN 66003).

Transmission format:

1 start bit
8 data bits
1 stop bit
No parity bit

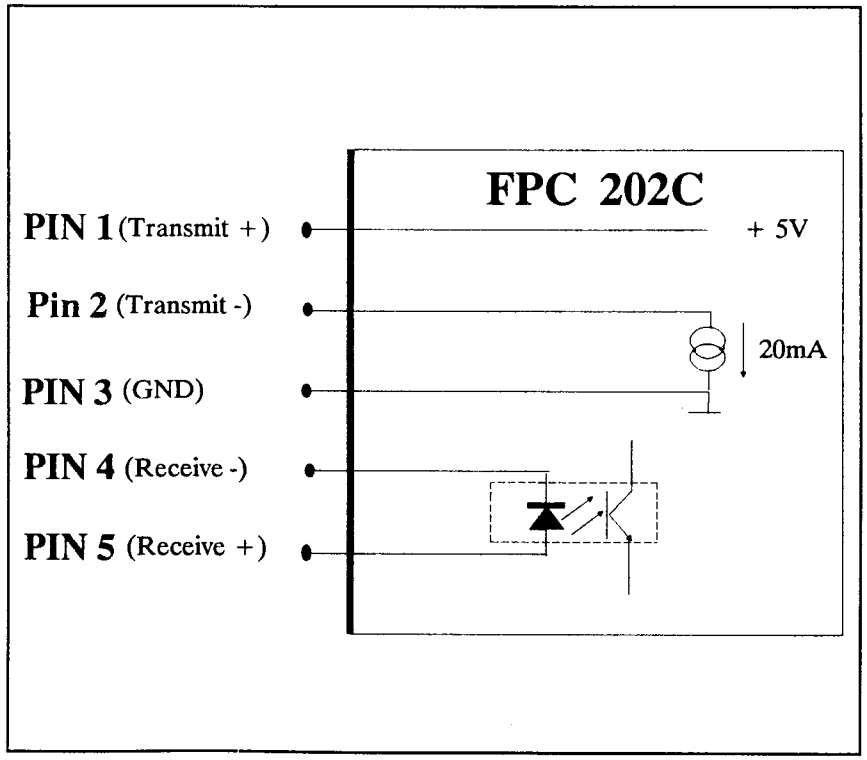


Fig. 5/1: Connections of the 20mA current-loop interface

The controller is connected to the dialog device via the:

Converter
D.AS-DIAG/RS 232-1
TN 80401

in connection with:

Diagnostic cable
E.KDI-20
TN 8325

Accessing the command interpreter

When one of the dialog devices has been connected to the FPC 202C, the actions listed below must be carried out in the sequence shown before the command interpreter can be accessed:

- Apply operating voltage to the controller.
- Switch on the dialog device (current supply of the E.TAB display via diagnostic cable from the controller).
- Access the command interpreter.

a) Via the controller

If no baud rate is specified, the last setting or, if switched on again, the "default" value is valid.
The baud rate can be set when the command interpreter is accessed.

Input format:

EXT < [baudrate] > < CR >

Baudrate = {1,2,3,4,6,9}

whereby:

1200 Bd	=	1
2400 Bd	=	2, for the E.TAB
300 Bd	=	3
4800 Bd	=	4
600 Bd	=	6
9600 Bd	=	9 default

b) Via dialog device

The dialog device should be set at the correct baud rate (default: 9600 Bd).
Alternatively, the FPC 202C can be adapted to the baud rate of the dialog device.

```
Input format:
CTRL T
```

```
The command interpreter responds with:
FPC-202 V3.0
> _
```

The command interpreter can be accessed with CTRL T even when the controller registers an error. However, this error must be eliminated before any further activity (see Chapter 6).

Exiting the command interpreter

The command interpreter can be exited with either the controller or the dialog device.

- Via the controller:

```
Input format:
EXT <CR>
```

- Via the dialog device:

```
Input format:
X <CR>
```

When this command has been executed, the controller can be addressed again via the integrated keyboard.

Command structure

A command consists basically of a letter for the command identifier, almost always a parameter and, if necessary, a memory address.
The entry can be in upper-case or lower-case letters.
A command is concluded with <CR> and the character sequence
<CR><LF><><><DC1>.

Input format:
<command letter> [<parameter>]
[<memory address>]

Del, Backspace or CTRL H can be used to delete incorrect entries before they are concluded with <CR>.

Command identifier

The letters shown below denote commands.

Command letter and command	Explanation
H = HEX-DUMP (= Selective hex. memory) D = DISPLAY (= Display operands)	Display memory contents/operands
C = CHECKSUM (= Checksum)	Form checksum
M = MODIFY (= Modify operands)	Modify
Z = Allocate memory space N = NULLIFY (= Delete programs) and operands) Y = INITIALIZE directory	Memorymanagement
W = WRITE (= Output data) L = LOAD (= Load data)	Data saving
R = RUN (= Start)	Start program
S = STOP (= Stop)	Stop program
T = Interrogate keyboard	Check keyboard
X = EXIT (= Leave)	Exit command interpreter

Parameter definitions

Different parameters must be used, depending on which operand, file or memory area a command is to be used.

Parameter	Meaning
C	Complete memory contents without free user memory
S	Depending on previous command, available user memory or system interrogation
D	Depending on previous command, directory or display format
K	Configuration
Y<n>	Hexadecimal display of task control block no. n
L{A/K/F}	Library
H	Handshake format
P<m>	Program no. m
B<m>	Program module no. m
E<m>.<m>	Input no. m.m
EW<m>	Input word no. m
A<m>.<m>	Output no. m.m
AW<m>	Output word m
M<x>.<x>	Flag no. x.x
MW<x>	Flag word no. x
Z<y>	Counter no. y
ZW<y>	Counter word no. y
ZV<y>	Counter preselect no. y
T<y>	Timer no. y
TW<y>	Timer word no. y
TV<y>	Timer preselect no. y
R<z>	Register no. z

n = {0,...,2}	y = {0,...,31}	A = Statement list (STL)
m = {0,...,7}	z = {0,...,63}	K = Ladder diagram (LAD)
x = {0,...,15}		F = External library

Memory addresses

Either a 32Kbyte RAM or a 32Kbyte EPROM serves the controller as user memory. The following memory addresses are valid ($ indicates hexadecimal address).

Memory area	Address range
System memory	$0000 - $7FFF
User memory	$8000 - $EFFF
Data memory	$F000 - $FFFF

Load format

Data is read or loaded from the command interpreter via the diagnostic port in "INTELHEX format". With this data format each line contains both its memory address as well as its checksum. In this way, serial data transmission is made more reliable.

A program file consists of any desired number of information lines in the form:

<length> <AA> <type> <data> <check byte>

Length : Number of data bytes per line (two-figure in hex. form).

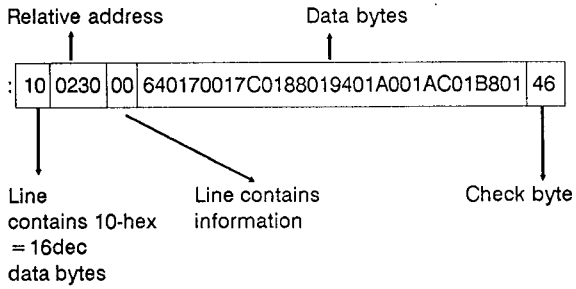
AA : Start address of the first data byte (four-figure).

Type : 00 = the line contains information
01 = the line is the last to be transmitted and contains no information. Record is an "end record".

Data : Data bytes (each two-figure)

Check byte : Check byte of all bytes in the line.

Example



Notes

DESCRIPTION OF COMMANDS

This chapter contains an explanation of the individual commands and, in some cases, appropriate examples.

Show memory contents

By means of a suitable dialog device connected to the controller, the information stored in the FPC 202C can be shown on a display.

- HEX-DUMP command: Show memory
- DISPLAY command : Show operands

Show memory

The H command can be used to display memory areas in hexadecimal form.
The H command can be aborted with the <CR> key.

User memory

Input format	Meaning
H$ <AA>	Display a memory line
H$ <AA>-<EA>	Display memory area
H$ <AA> <length>	Display memory area
AA = start address, EA = end address, Addresses from $8000-$FFFF	

When this command has been concluded with <CR>, the command interpreter delivers 16 data bytes in pairs on the same line. Any desired number of such lines can be shown.

Example 1

```
H$8088
8088: 19D0 0BFF 0507 2106 2205 2109 220A 21D0
```

Example 2

```
H$8000-8030
8000: A01C 5383 1480 5882 0000 0000 0000 0000
8010: 0000 0000 90A5 DA01 0500 16D0 800A 0AFF
8020: 1940 0BFF 0501 2100 210F 21D0 0AFF 19D0
8030:A1
```

System memory

Input format	Meaning
HS $ <AS>	Display a memory line
HS$<AS>-<ES>	Display a memory area
H$<AS><length>	Display a memory area
AS = start address, ES = end address, Addresses from $0000-$7FFF	

Example 1

```
HS$0000
0000: 0201 0002 3F02 FFFF FFFF FF02 041D FFFF
```

Example 2

```
HS$0100-0120
0100: 787F E4F6 D8FC 7581 5112 0139 E535 20E6
0110: 0B75 503A F5F0 1203 F785 F0J5 E5JE 20E6
0120: 0B
```

Example 3

```
HS$1500 0020
1500: 9480 F990 F20C 743A F0A3 7431 F0A3 7430
1510: F0A3 E912 15AE E812 15AE 7430 F0A3 7430
```

The user can display the whole memory area including the directory, the directory alone or each individual file.

Input format	
Memory area	Meaning
HC	Display all programs, program modules and libraries
HD	Display directory
HP { 0,..., 7}	Display programs nos. 0 to 7
HB { 0,..., 7}	Display program modules nos.0 to 7
HL {A, K, F}	Display libraries
HY { 0, 1, 2}	Display task control blocks 0, 1, and 2
HK	Display configuration 1 Only CPU (min. structure) 2 CPU plus 1 extension device 3 CPU plus 2 extension devices 4 CPU plus 3 extension devices

Display operands

The DISPLAY command can be used to display the states and contents of the operands as well as the current status of the programs. The command interpreter always responds on the input line.

Single-bit operands

Display status (0-signal or 1-signal) of selected operand.

Input format		Message of command interpreter
Timer	DT<z>	DT<z>={0/1}
Counter	DZ<z>	DZ<z>={0/1}
Inputs	DE<x>.<x>	DE<x.x>={0/1}
Outputs	DA<x>.<x>	DA<x.x>={0/1}
Flags	DM<y>.<y>	DM<y.y>={0/1}
x={0,...,7},y={0,...,15},z={0,...,31}		

Example 1

Display status of single-bit operand output 0.6

DA0.6=1

>_

Example 2

Display status of single-bit operand input 1.5

DE1.5=0

>_

Multibit operands

The following enquiries are possible for showing the contents of multibit function units:

Input format		Message of command interpreter
Input words:	DEW<x>	DEW<x> = {$HEXADEC./SIGNEDDEC./DEC.}
Outputs words:	DAW<x>	DAW<x> = {$HEXADEC./SIGNEDDEC./DEC.}
Flag words:	DMW<w>	DMW<w> = {$HEXADEC./SIGNEDDEC./DEC.}
Timer words:	DTW<y>	DTW<y> = {$HEXADEC./SIGNEDDEC./DEC.}
Counter words:	DZW<y>	DZW<y> = {$HEXADEC./SIGNEDDEC./DEC.}
Timer preselect:	DTV<y>	DTW<y> = {$HEXADEC./SIGNEDDEC./DEC.}
Counter preselect:	DZV<y>	DZW<y> = {$HEXADEC./SIGNEDDEC./DEC.}
Error word:	DF	DF = {$HEXADEC./SIGNED DEC./DEC.}
Registers:	DR<z>	DR<z> = {$HEXADEC./SIGNEDDEC./DEC.}
x = {0,...,7}, w = {0,...,15}, y = {0,...,31}, z = {0,...,63}		

Special operands

A) Display format

Input format	Message of command interpreter
D D	>DD={S/D/H}
S = signed decimal (default), D = decimal, H = hexadecimal	

B) Handshake format

Input format	Message of command interpreter
D H	>DH={0/1}
0 = without handshake, 1 = with handshake	

C) Program

Input format	Message of command interpreter
D P <x>	>DPx = <program type>, <program length>, <program status>, [,<current prog. step no.> [,<module number>, <current module step no.>]
x = {0,..., 7)	

Program type
The program type is given in code form according to the following bit pattern (in decimal or hexadecimal form).

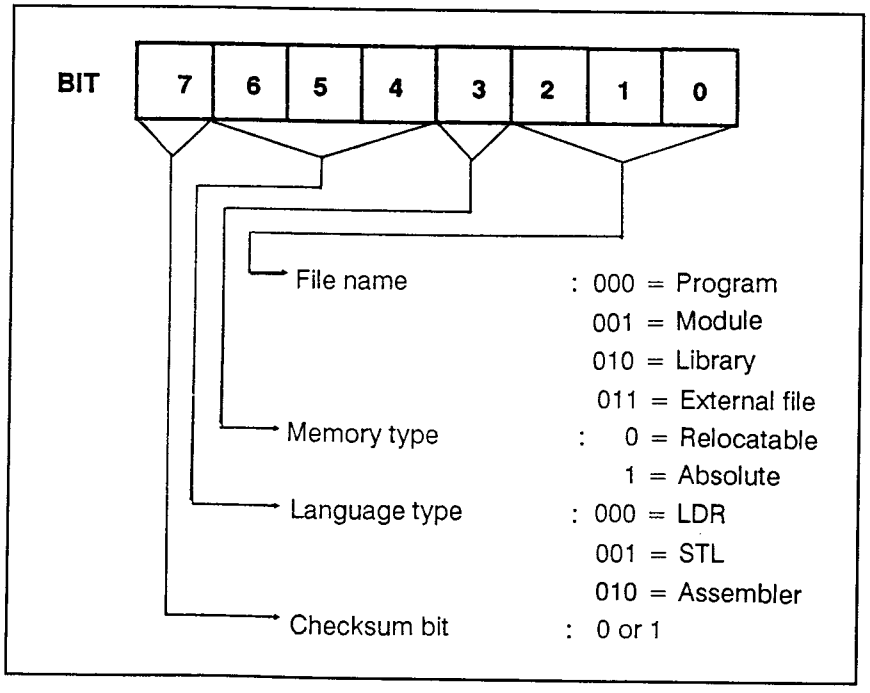


Fig. 5/2: Coding the program type

Example

Relocatible STL module with checksum bit.

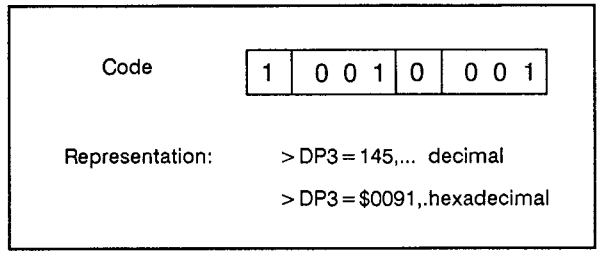


Fig. 5/3: Example of program type

Program Length
The program length specifies the number of data bytes in a program.

Program status
Program status = 0 means "Program is inactive",
Program status = 1 means "Program is active".

Program step no.
The current program step number (0 to 255) is only shown if the program is active (status = 1).
LAD program: no. 0
STL program: no. 0-255

Module no. and module step no.
If an active program processes a module, the module number (0 to 7) and the module step no. (0-255) are shown.

D) Module

Input format	Message of command interpreter
DB <x>	>DB<x> = <module type>, <module length>
x = {0,..., 7}	

Module type
See program type

Module length
The module length specifies the number of data bytes in a module.

E) Library

Input format	Message of command interpreter
DL {A/K/F}	>DL{A/K/F} = <library length>

Library length
The library length specifies the number of data bytes in a library.

F) Free user memory

Input format	Message of command interpreter
DS	>DS=$<available memory area>

Available user memory
The DS command is used to show the size of the available user memory either in decimal, hexadecimal or signed decimal form.

Form checksum

The command interpreter enables the user to check any area of the user memory.

Checksum and check byte

The checksum and the check byte of a memory area is formed with the C command and shown on the display.

User memory

Input format:	Meaning
C $ <AA>	Checksum of a memory line
C $ <AA>-<EA>	Checksum of a memory area
C $ <AA> <length>	Checksum of a memory area
AA = Start address, EA = End address, Addresses from $8000-$FFFF	

When the command is concluded with <CR>, the command interpreter delivers the relevant information on the same line.

Example 1

C$8000=00064C:B4

Example 2

C$80AF-81EB=00600C:F4

Example 3

C$820C 0080=002E79:87

The checksum is shown in six-figure form, the check byte in two-figure hexadecimal form.

System memory

Input format:	Meaning
CS $ <AS>	Checksum of a memory line
CS $ <AS>-<ES>	Checksum of a memory area
C $ <AS> <length>	Checksum of a memory area
AS = Start address, ES = End address, Addresses from $0000-$7FFF	

In the same way, the appropriate checksum can be formed for the complete memory area, for the directory alone or for each individual file.

Input format:	Meaning
CC	Checksum of all programs, program modules, libraries
CD	Checksum of the directory
CP {0,...,7}	Checksum of programs nos. 0 to 7
CB {0,...,7}	Checksum of module nos. 0 to 7
CL {A/K/F}	Checksum of libraries

Example 1

CP2=001500:00

Example 2

CLK = 02F300:00

The check byte of the complete memory area, of the directory or of each individual file must always be 00. If the check byte is not 00, the interrogated memory area has an error.

Modifying

With the command interpreter the user can display information on the programmed function units and, if necessary, correct them.

Modifying operands

The contents or states of the operands can be modified with the M command. The operand can also be displayed.

- In order to modify an operand directly without displaying it first, the user should enter the desired value after the command and conclude with <CR>.

  Example

  >MAW1=255

- If the user wishes to look at the contents or the status of the operand first, he should enter <CR> after the command. The command interpreter delivers the current value. The new value should then be entered after the colon and concluded with <CR>.

  Example

  >MAW1=255:126

The values can be entered in decimal, hexadecimal and signed decimal form (see display format).

Single-bit operands

The following entry is required for modifying the status of operands:

Input format:		
Outputs	:	MA <x>.<x>
Flag status	:	MM <y>.<y>
Timer status	:	MT <z>
Counter status	:	MZ <z>
x = {0,...,7}, y = {0,...,15}, z = {0,...,31}		

Example

```
>MA1.6 = 1 : 0
```

Status before modifying

Status after modifying

Multibit operands

The following entries are required for modifying the contents of multibit operands:

Input format:		
Output word	:	MAW<x>
Flag word	:	MMW<y>
Timer word	:	MTW<z>
Timer preselect	:	MTV<z>
Counter word	:	MZW<z>
Counter preselect	:	MZV<z>
Register	:	MR<u>
Error	:	MF
x = {0,...,7}, y = {0,...,15}, z = {0,...,31}, u = {0,...,63}		

Example

>MZW0=9662:xxxxx

Current counter value

New counter value

Special operands

A) Display format

Input format:
MD = {D/S/H}
D = decimal, S = signed-decimal, H = hexadecimal

B) Handshake format

Input format:
MH = {0/1}
0 = without handshake, 1 = with handshake

Memory management

As soon as programs are to be deleted or loaded into the controller, commands are required in order that the memory can be organized.

Memory allocation

The Z command serves as preparation for loading a program, a module or a library. It is a prerequisite for enabling a program or module to be loaded, since the LP/LB command can only be used with the program or module in the last position in the user memory.

Create program/module

Input format:
ZP <program number>, <program length> ZB <module number>, <module length>
Program or module no. = {0,..., 7} Program or module length = Length of file in bytes

The Z command is used to create a program/module which does not already exist. If a program/module with the selected number already exists, this file will be moved to the final position in the user memory where it remains operable.

Command interpreter message:
>ZP <program number>, <program length> $ address between $8000 and $FFFF or >E9: MEMORY FULL

The program will not fit into the user memory.

Create library

Input format:	
Z L <library type>, <library length>	
Library type	A (= statement list) K (= Ladder diagram) F (= External library)
Library length =	Length of library in bytes

The ZL command must be given before each LL command. Using the length specified, a check is made to see if there is sufficient memory space available and, in the case of an already existing library, whether it is the correct version.

Command interpreter message:
>ZL<library type>, <library length>
$ address between $8000 and $FFFF

The library does not exist already and can be loaded with the LL command.

or

>ZL <library type>, <library length>
$ address between $0000 and $7FFF

The library exists already and must not therefore be loaded into the controller.

or

```
> E8: ACCESS ERROR
```

1. The library is an incorrect version.
2. 4 libraries have been loaded already.
3. REName program or DELete program is active.

or

```
> E9: MEMORY FULL
```

The library does not fit into the user memory.

Delete user memory

A separate command must be used, depending on whether the user wishes to delete all files (programs, program modules, libraries) at the same time, or just a single file.

The user should check that he is using this command correctly before pressing the return key, because the Y command deletes the whole directory and the files in the user memory are therefore no longer available.

A) Delete all files

Input format:	Message of command interpreter:
Y	>DEL ALL ? [Y/N]:_
Input format:	
Y <CR> if everything is to be deleted <CR> providing the Y command is not to be used	

B) Delete individual programs

Input format:
NP <program number>
Program number = {0,..., 7}

C) Delete individual modules

Input format:
NB <module number>
Module number = {0,..., 7}

D) Delete individual libraries

Input format:	Message of command interpreter:
NL<library type>	>DEL ALL ? [Y/N]:_
Library type = {A/K/F}	
Input format:	
Y <CR> if everything is to be deleted <CR> providing the Y command is not to be used	

Delete operands

A) Delete all flags

Input format:
N M

B) Delete all registers

Input format:
N R

C) Delete all timers

Input format:
N T

D) Delete all counters

Input format:
N Z

Data saving

For data saving purposes, it is absolutely necessary, on the one hand, to be able to read the programs in a controller in order to store them externally; on the other hand, it must also be possible to load externally stored programs into the FPC 202C. The command interpreter offers different commands for this purpose.

Read data

With the W command, a specified area of the user memory can be read in load format (= INTELHEX format, see Chapter 1.3) via the diagnostic port. The address of the data lines output with the W command is relative, i.e. it refers to the start address of the relevant memory area.

Input format:	Meaning
W $ <AA>	Reading a memory line
W $ <AA> <EA>	Reading a memory area
W $ <AA> <L gth>	Reading a memory area
AA = Start address, EA = End address Addresses from $8000-$FFFF	

The user can therefore display the complete memory area, the directory alone or each file individually.

Input format:	Meaning
WC	Reading all programs, Program modules and libraries
WD	Reading the directory
WP {0,...,7}	Reading programs nos. 0 to 7
WB {0,...,7}	Reading module no. 7
WL {A/F/K}	Reading the library

Load data

With the L command (Load data), the user can load files (programs, program modules, libraries) from the dialog device into the memory of the FPC 202C.

Before the L command is used, the new program must be loaded into the memory with the Z command, or moved to the end of the valid user memory.

Input format:	Meaning
L$<AA>	Loading a memory line
AA = Start address from $0000-$7FFF	

In this way, the complete memory area, the directory alone or each file can be loaded individually.

Input format:	Meaning
LC	Loading the complete memory area
LD	Loading the directory
LP{0,...,7}	Loading programs nos. 0 to 7
LB{0,...,7}	Loading modules nos.0 to 7
LL	Loading a library

The criteria for aborting is the end record (otherwise hard reset), then from the terminal <DC1> or <CR> (with or without handshake).

The command interpreter transfers the data in INTELHEX format.

Start program

RUN

The R command must be used if an operable program is to be activated in the controller.

Input format:
R

The R command alone starts the first operable program entered in the directory.

If a special program is to be started, this must be specified directly.

Input format:
RP<programnumber>
Programnumber = {0,...,7}

Stop program

STOP

The S command stops all running programs.

Input format:
S

Keyboard test

With the T command, the keyboard of the FPC 202C can be interrogated for test purposes. This command can be aborted with <CR>.

Input format:
T<CR>

When this command is concluded with <CR>, the number of a key pressed on the FPC 202C keyboard will be transmitted to the interface.

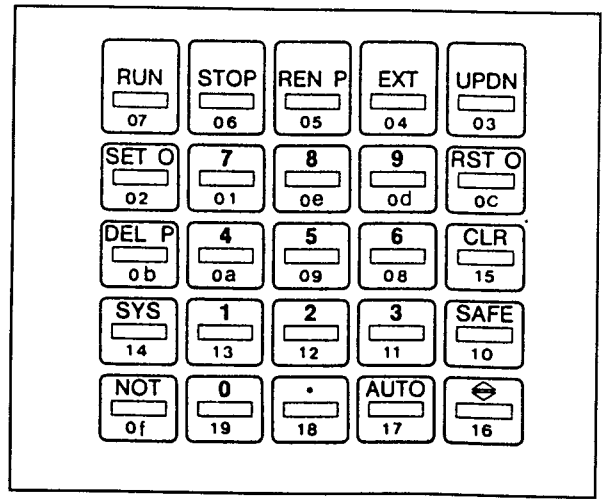


Fig. 5/4: Key numbers

Notes

6

ERROR MESSAGES OF THE OPERATING SYSTEM

ERROR MESSAGES OF THE COMMAND INTERPRETER

ERROR MESSAGES OF THE OPERATING SYSTEM

The error messages of the operating system are shown below.

Error message on display (hexadecimal)	Brief definition
Er01	Checksum test incorrect, user memory and remanent memory area deleted.
Er05	EPROM user memory defective or missing.
Er10	Program directory empty, program does not exist.
Er11	Program does not exist or incorrect.
Er12	Program number cannot be modified, because program with desired number already exists.
Er17	No memory space for further programs.
Er18	Loading error, user memory and remanent memory area deleted.
Er25	Program cannot be renamed, deleted, created or moved, because of EPROM operation.
Er50	Battery capacity almost finished, battery must be replaced.
Er51	Battery empty, missing, has been missing, user memory and remanent memory area deleted.
Er55	Remanent memory area deleted, controller can be operated with EPROM and without battery.

Error treatment

The error messages on the controller display must be acknowledged with the CLeaR key. The controller can then be used again, but the cause of the error has not been eliminated.

ERROR MESSAGES OF THE COMMAND INTERPRETER

Error messages of the command interpreter are shown on the connected PC as follows:

```
Audible signal
<error message>
>_
```

The following error messages can appear:

Error message	Meaning
E1: SYNTAX ERROR	Incorrect command or incorrect command
E2: ADDR. ERROR	The end of the user memory has been reached when loading data.
E3: INPUT ERROR	Incorrect or impermissible sign when entering data.
E6: PR NOT FOUND	The file specified is not in the memory.
E7: BAD CHECKSUM	Checksum error ascertained when loading a file.
E8: ACCESS ERROR	Access to memory areas which are not permitted or which do not exist.
E9: MEMORY FULL	Not enough space in user memory.

Error treatment

If, when the user accesses the command interpreter, he thinks there is an error in the controller, he can check this from the terminal and correct the error.

Display error

Errors can be shown in the command interpreter with the command "Display error".

Input format:	Message of command interpreter
DF<CR>	DF = <xx>
xx = error number of the FPC 202C (see above)	

Modify error

All errors must be eliminated before the command interpreter commands can be executed.

Display error

Input format:	Message of command interpreter
MF<CR>	MF = <xx>:
xx = error number of the FPC 202C (see above)	

Delete error

Input format:
MF = <xx>:0<CR>
or
MF = 0 <CR> without expecting error number

Notes

7

BATTERY REPLACEMENT

FUSES

OUTPUT RELAYS

BATTERY REPLACEMENT

In the event of a power failure, the RAM contents are saved by a lithium battery (service life: five years at 25°C).

A monitoring circuit controls the charge status of the battery when the FPC 202C is switched on. If there is a fault, it will be shown on the display.

- Er50
  Battery capacity is running out. Battery must be replaced.
- Er51
  Battery is empty, missing or was missing; i.e. battery voltage has possibly failed due to supply voltage not being applied.

Battery replacement without loss of data

If the supply voltage is applied:

- the battery can simply be replaced. Attention must be paid to the polarity.

If the supply voltage is not applied:

- the auxiliary voltage for data saving should be applied. Attention must be paid to the polarity.
- Specification:
  V_{min} = 2.8V
  V_{max} = 3.7V
- Connection:
  There are two flat plugs above the battery compartment for connecting the auxiliary voltage.
- The battery should be replaced with the auxiliary voltage applied.

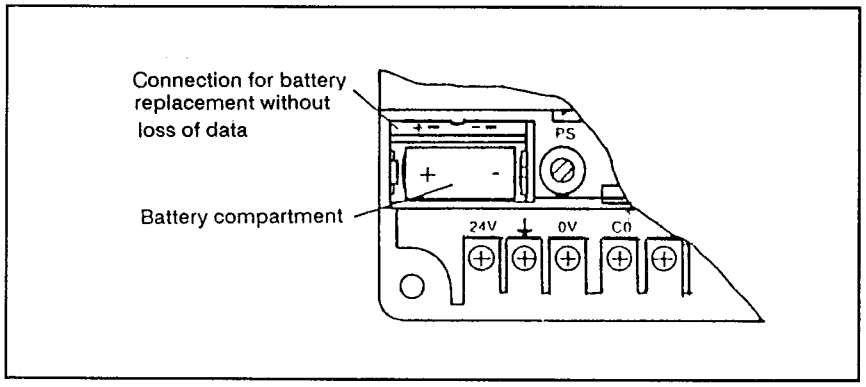


Fig. 7/1: Replacing the battery

FUSES

The FPC 202C contains blow fuses for:

- supply voltage
- relay outputs (each output individually fused)
- transistor outputs (all outputs commonly fused)

In addition, the controller has an electronic short-circuit and overload fuse for the transistor outputs.

Blow fuses

The blow fuses of the FPC 202C are on the right next to the battery compartment. Access can be made to the individual fuses by removing cover (3) (see fig. 1/1) (*).

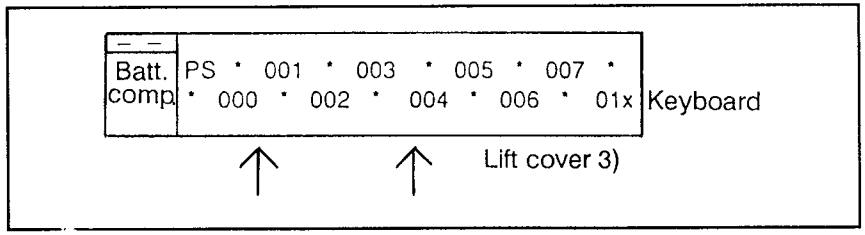


Fig. 7/2: Fuses of the FPC202C

The fuses are marked in order that they can be distinguished (see Fig. 7/2). Further details on the fuses used are to be found in the table below.

Designation of fuses	Value in A	Explanations
PS	T 1.6	Power supply / fuse for supply voltage
O00	T 2.0	Output 0.0 / fuse for relay output 0
O01	T 2.0	Output 0.1 / fuse for relay output 1
O02	T 2.0	Output 0.2 / fuse for relay output 2
O03	T 2.0	Output 0.3 / fuse for relay output 3
O04	T 2.0	Output 0.4 / fuse for relay output 4
O05	T 2.0	Output 0.5 / fuse for relay output 5
O06	T 2.0	Output 0.6 / fuse for relay output 6
O07	T 2.0	Output 0.7 / fuse for relay output 7
O1x	T 3.15	Output 1.0 ... 1.7 / fuse for the transistor outputs 0 to 7
T means: "slow-blowing" fuse		

When changing the fuses:

- With RAM, insert battery for saving data
- Switch off the voltage supply.

Short-circuit and overload fuse

The transistor outputs are protected with a high-speed short-circuit and overload fuse. If there is a short circuit or an overload of one or several transistor outputs, all the transistor outputs will be switched off.

The outputs can be switched on again in one of two ways:

- by switching off the supply voltage to the transistor outputs,
- by briefly removing the 3.15A fuse (see Fig. 7/2) of the transistor outputs.

OUTPUT RELAYS

The relays for each of the 8 relay outputs can be replaced individually.

The relays should always be replaced by Service personnel.

Replacing the relays

- Disconnect the controller from the voltage supply.
- Remove the base plate of the controller housing by loosening 6 screws (to loosen screw 5, the red slide must be pushed up by lifting the terminal strip).

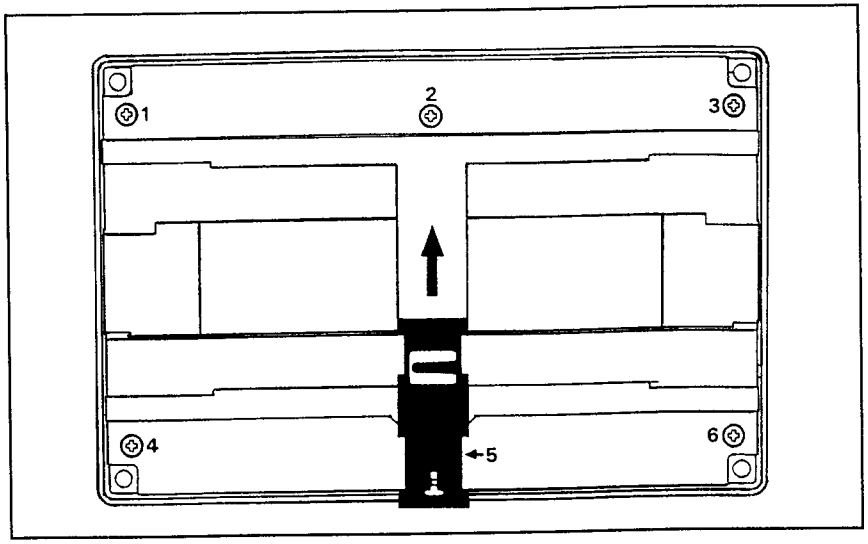


Fig. 7/3: Dismantling the baseplate

- Remove the PC board carefully from the housing. Do not disconnect the other PC board (flat ribbon cable).

- Replace the relay.

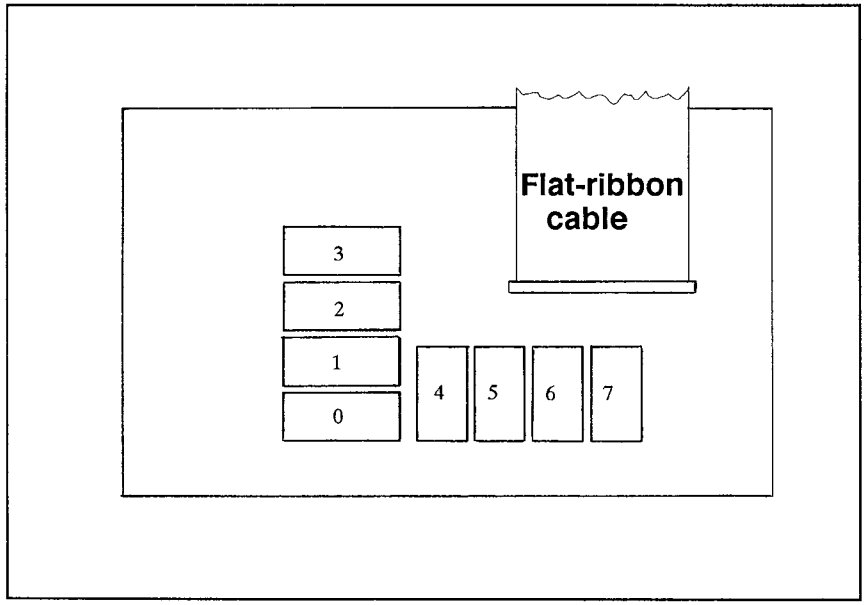


Fig. 7/4: Positioning the relays

- Remove the defective relay from the socket.
- Insert and press in the new relay (if correctly positioned, the pins of the relay will fit exactly into the socket connections.

- Insert the PC board into the housing again.
- Replace the base of the housing and tighten the screws equally.

8

OVERVIEW OF THE I/O EXTENTION DEVICE

CONNECTION TO CPU

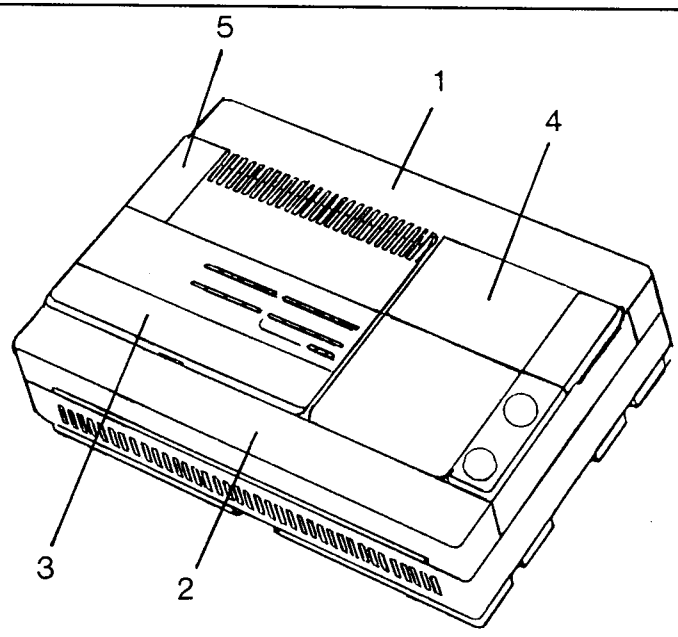
I/O EXTENSION STAGES

OVERVIEW OF I/O EXTENSION DEVICE

The FPC 202C has 32 input and output stages. By connecting up to three E.EEA 202 I/O extension devices, the user can increase the number of I/O stages to 128.

Structure of the I/O extension device

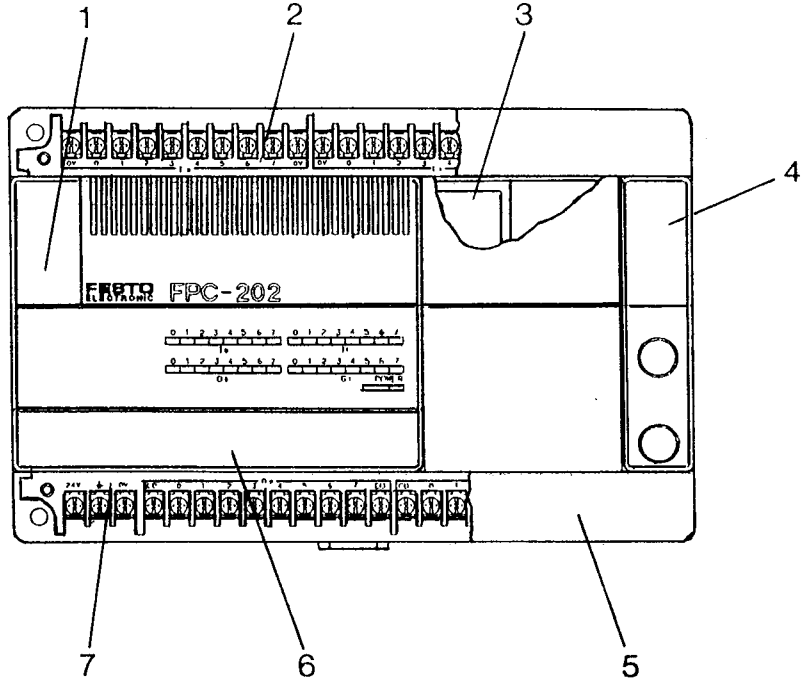
There are five covers in the housing of the E.EEA 202 I/O extension device. They can be raised at the points marked with arrows.



1 Input terminals
2 Output and supply terminals
3 Fuses
4 Plug and address switch for extension device
5 Plug for flat-ribbon cable

Fig. 8/1: Cover parts of the extension device

Location of parts



1 Bus plug to CPU
2 16 inputs
3 Address switch
4 Bus plug to I/O extension devices
5 16 outputs
6 Fuses
7 Supply voltage

Fig. 8/2: External structure of the extension device

Notes

CONNECTION TO CPU

Flat ribbon plug

The I/O extension device is connected to the CPU via the:

Flat ribbon cable
008395 BUS CABLE
E.SSK-202

or

Flat ribbon cable
018131 BUS CABLE (LONG)
E.KBL-202

The flat ribbon cable should be connected as follows:

- Remove cover 4 of the CPU (see Fig. 1/1).
- Remove cover 5 of the I/O extension device (see Fig. 8/1).
- The bus connection must be made when all devices involved are switched off.
- The plugs of the flat ribbon cable are protected against torsion.

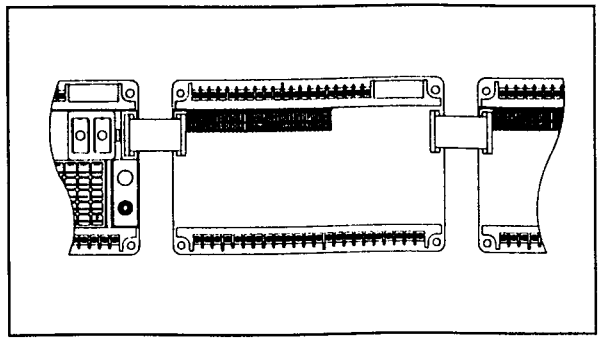


Fig. 8/3: Flat ribbon connection

Supply voltage

A voltage supply with the following specifications is required for operating the extension device.

Supply voltage	
Rated value	DC 24 V
Tolerance range	- 25%; + 25%
Current capacity	
typical	250mA
maximum	350mA
Power consumption	
maximum	6.4W

Applying the supply voltage

1) The bus connection to the CPU and to other extension devices must be fitted before the supply voltage is applied.
2) Minimum diameter of earth cable: 1.5mm^2
3) Maximum length of the connection cable to the terminals: 600mm
4) The voltage supply to the extension device via the CPU is shown in Fig. 8/4.

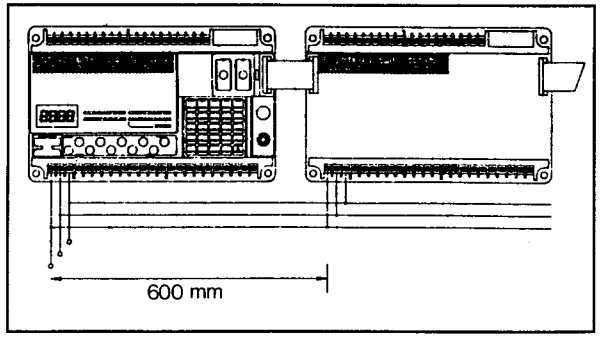


Fig. 8/4: Voltage supply

Address switch

Up to three I/O extension devices can be connected to the FPC 202C. The address of each device can be set with two slide switches. The first extension device must always be connected directly to the main device.

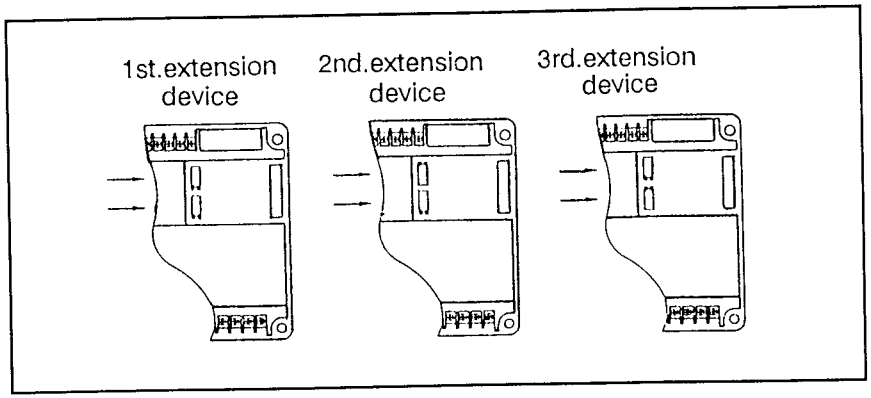


Fig. 8/5: Address switch

Installation and maintenance

Installation

The I/O extension devices are fitted in the same way as the main device (see Chapter INSTALLATION - Fitting methods).

As the bus cable is flexible, the individual devices can be from 5 mm to 450 mm apart.

Maintenance

The battery, the fuses and the output relay are replaced in the same manner as those of the main device (see Chapter MAINTENANCE).

I/O EXTENSION STAGES

Programming instructions

The inputs and outputs of the extension device are programmed in the same manner as those of the main device. The only difference is the extended numbering of the appropriate operands.

The CPU has 16 inputs in two input words. These input words have the numbers 0 and 1. The same applies to the outputs.

Each extension device also has two input and output words, which, related to the main device, are addressed consecutively.

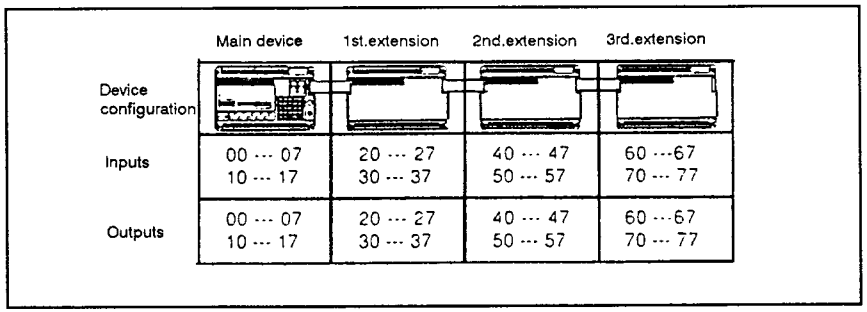


Fig. 8/6: Addressing the I/O stages

Notes

A

HARDWARE DATA

Operating requirements

In order to guarantee interference-free operation, cables which can transmit electrical or electro-magnetic interference should not be placed nearer than 100 mm to the housing.

Dimensions and weight

Dimensions:	240mm wide/158mm high/60mm deep
Weight:	1165 g

Insulation

Insulation group A as per VDE 0110	Leakage path and air gap within the internal system area.
Insulation group C as per VDE 0110	• Input and output area. • Insulation of internal system from input/output area.

Immunity to interference

Spark suppression as per VDE 0871 limit curve B	
interference impulses	2.5kV sym.-asyn. interference impulses tr = 5ns, tp = 100ns

Data of the interfaces and connections

Supply voltage

Supply voltage Rated value Tolerance	 DC 24 V -25%; +25%
Current capacity typical maximum	 165 mA 390 mA
Power consumption maximum	 7.2 W
Fuse primary	 1.6A (slow-blowing)

Outputs

Relay outputs

- All 8 relays exchangeable
- Each ouptut fitted with 2 A fuse
- Max. switching load:
  250V(AC)/ 2A,
  24V(DC)/ 2A
- Electrical service life: $3 \cdot 10^5$ switching cycles under ohmic load
- When inductive loads are connected, a suitable spark suppressor must be fitted

Transistor outputs

- Galvanic insulation via optocoupler
- High-speed short-circuit protection
- Additional fuse T 3.15 A (slow blowing)
- Protective diode against overvoltages when inductive loads are switched off
- If there is a short circuit or overload at one or several transistor outputs, all outputs will be switched off

- The outputs are switched on again by:
  a) switching off the supply voltage of the transistor outputs
  b) by briefly removing the 3.15 A fuse (01X) of the transistor outputs
- Voltage supply of the transistor outputs: 18.5 - 30 V
- Output currents:
  a) max. output current for an output: 2.0 A
  b) max. combined current for all 8 outputs: 2.0 A
  c) Sum of all output currents ...
     ... at 20°C: 2.0 A
     ... at 55°C: linear falling to 1.5 A
- Max. output voltage for an output at 24 V supply voltage:

Loading current	Output voltage
1.5 A	23.0 V
0.5 A	23.0 V
1.5 A	22.5 V

Inputs

- Galvanic insulation via optocoupler
- Voltage signal for logic 1:
  + 15 V to + 35 V
- Voltage signal for logic 0:
  - 30 V to + 7 V
- Input current at 24 V:
  11 mA typical
- Input current at 25 V:
  5 mA typical

Diagnostic connection

Signals are transmitted via this connection in a modified current-loop procedure. A 5 V (+- 5%) supply with a max. current loading of 200 mA is available for a device to be connected.

Pin assignment of the diagnostic port:

Pin	Designation
1	5 V and transmitted data positive terminal
2	Transmitted data negative terminal
3	0 V
4	Received data negative terminal
5	Received data positive terminal

Flat-ribbon connector

Further information on the bus plug for the flat-ribbon cable is to be found in Chapter "I/O extension device".

Data memory

User memory	Number of statements
32 K EPROM plug-in memory above socket	approx. 4300
32 K RAM integrated memory	approx. 4300

Backup time for RAM memory:
5 years providing an appropriate backup battery is used.

OPERANDS

Number, syntax and parameters of operands see Chapter "Programming instructions".

SUPPLY VOLTAGE FAILURE

If the supply voltage drops below 18V DC, a power failure is recognized. The checksum of the directory is then formed and stored in the user memory. The controller reacts differently depending on the duration of the power failure:

Failure time	Consequences
<30ms	No effects
>30ms <40ms	The display is switched off for the duration of the failure. When the voltage is switched on again, the controller functionsnormally.
>40ms	The outputs are switched off. When voltage is switched on again, controller enters STOP status, provided an AUTO start has not been programmed.
The above applies to a voltage drop from 24V to 0V.	

The bridgeable failure time varies with the level of the supply voltage before the failure. A low supply voltage results in shorter failure periods than with high supply voltages. This is due to the low loading status of the internal power unit capacitors.

Notes

B

HARDWARE DATA

Operating requirements

In order to guarantee interference-free operation, cables which can transmit electrical or electro-magnetic interference should not be placed nearer than 100 mm to the housing.

Dimensions and weight

Dimensions:	240mm wide/158mm high/60mm deep
Weight:	1140 g

Insulation

Insulation group A as per VDE 0110	Leakage path and air gap within the internal system area.
Insulation group C as per VDE 0110	• Input and output area. • Separation of the internal system from the input/output area.

Immunity to interference

Spark suppression as per VDE 0871 limit curve B	
Interference impulses	2.5kV sym. asyn. interference impulses tr = 5ns, tp = 100ns

Data of the interfaces and connections

Supply voltage

Supply voltage	
Rated value	DC 24 V
Tolerance	-25%; +25%
Current capacity	
typical	250 mA
maximum	350 mA
Power consumption	
maximum	6.4 W
Fuse	
primary	1.6A (slow-blowing)

Outputs

Relay outputs

- All 8 relays exchangeable
- Each fitted with 2 A fuse
- max. switching load:
  250V(AC)/ 2A,
  24V(DC)/ 2A
- Electrical service life: $3 \cdot 10^5$ switching cycles under ohmic load
- When inductive loads are connected, a suitable spark suppressor should be fitted.

Transistor outputs

- Galvanic insulation via optocoupler
- High-speed short-circuit protection
- Additional fuse T 3.15 A (slow blowing)
- Protective diode against overvoltages when inductive loads are switched off
- If there is a short circuit or overload on one or several transistor outputs, all the outputs will be switched off

- The outputs are switched on again by:
  a) switching off the supply voltage of the transistor outputs.
  b) by briefly removing the 3.15 A fuse (01X) of the transistor outputs
- Voltage supply of the transistor outputs: 18.5 - 30 V
- Output currents:
  a) max. output current for an output: 2.0 A
  b) max. combined current for all 8 outputs: 2.0 A
  c) sum of all output currents ...
     ... at 20°C: 2.0 A
     ... at 55°C: linear falling to 1.5 A
- Max. output voltage for an output at 24 V supply voltage:

Load current	Output voltage
1.5 A	23.0 V
0.5 A	23.0 V
1.5 A	22.5 V

Inputs

- Galvanic insulation via optocoupler.
- Voltage signal for logic 1:
  + 15 V bis + 35 V
- Voltage signal for logic 0:
  - 30 V bis + 7 V
- Input current at 24 V:
  11 mA typical
- Input current at 25 V:
  5 mA typical

OPERANDS

The I/O extension device has the following operands:

- 8 relay outputs
  (with exchangeable relays)
- 8 transistor outputs
- 16 inputs

C

ORDER DATA

Product	Designation	Type	Part number
Program logic controller FPC 202C	CONTROLLER	E.FPC202	008381
Operating system EPROM compiler	SYSTEM EPROM 202C	E.SEC-202	018135
Operating system EPROM interpreter	SYSTEM EPROM 202I	E.SEI-202	018134

User memory module	MEMORY, EPROM	E.EPR-32K	008323
Backup battery for RAM memory	BATTERY, LITHIUM	E.BAT-HMIG	008351

Extension device for the I/O stages	EXTENSION, I/O	E.EEA202	008391
Short bus cable	BUS CABLE	E.SSK-202	008395
Long bus cable	CABLE, BUS (LONG)	E.KBL-202	018131

Exchangeable output relay	OUTPUT RELAY	E.REL-202-8	008325
Exchangeable terminal strips	TERMINAL STRIPS	E.KLM-202	008385

Diagnostic cable	CABLE, DIAGNOSTIC	E.KDI-20	008325
Converter for 20mA-V24	CONVERTER	D.AS-DIAG/ RS 232-1	80401

Notes

EXECUTION TIMES

The cycle time depends on the configuration of the device (the number of I/Os connected, program length, number of active tasks) and can be calculated by adding the basic load time to the actual program processing time. The basic load time here represents the time required by the operating system.

I/O update	
Main device	8 µs
Main device + 1 extension	145 µs
Main device + 2 extensions	197 µs
Main device + 3 extensions	249 µs
Timer update	
Timer is inactive	3 µs
Timer is inactive	22 µs
Timer is active (no transfer)	19 µs
Timer is active (no transfer)	31 µs
Process dispatcher)	9 µs
10ms interrupt	45 µs
Assign program (after processor change)	29 µs

The execution times of individual or of several commands can be calculated with the aid of tables C.1 to C.12. Please note that the times have a basic quartz frequency of 12MHz. However, the FPC202C is driven by an 11.059 MHz quartz. This means that the execution times specified for individual or several commands must be corrected by the factor 1.085.

N.B.: The abbreviation LIB indicates the use of a library function
FE0-23 Field bus flag
FE32-47 Special FU (parameter transfer)

Commands for program organization

Function	Times min	Times max	Number of bytes
Edge evaluation	25	29	8 (LIB)
Edge interrogation rel	2	4	7
Edge interrogation abs	2	7	10
NOP		0	0
PW		13	6 (LIB)
Task		14	3 (LIB)
change (suspend)		2	3
jump to	22	27	8 (LIB)
start of step		2	3
if	2	3	4
then rel	3	5	7
then abs	2	3	5
otherwise rel	3	5	8

rel = implemented in the program with relative jump

abs = implemented in the program with absolute jump

D

Single-bit load commands

Function	Times min	Times max	Number of bytes
Load input	1		2
Load input neg.	2		3
Load output	1		2
Load output neg.	2		3
Load flag	5		6
Load flag neg.	6		7
Load counter	2		4
Load counter neg.	3		5
Load timer	2		4
Load timer neg.			
Load program	5		6
Load program neg.	6		7
Load error	11	12	11
Load error neg.	12	13	11
Load init. flag	1		2
Load init. flag neg.	2		3

Single-bit instructions

Function	Times min	Times max	Number of bytes
To output	2		2
To output neg.	3		4
To flag	8		7
To flag neg.	9		9
To counter	4		6
To counter neg.	5		8
To timer	4		6
To timer neg.	5		8

Bit execution commands

Function	Times min	Times max	Number of bytes
Set output LAD	2	3	4
Set output ATL		1	2
Set flag LAD	2	10	9
Set flag STL		8	7
Set counter		11	12
Set counter with		22	22
Set timer		18	17
Set timer with		19	19
Move output STL	4	6	12
Move output STL		10	17

Function	Times min	Times max	Number of bytes
Set output LAD	2	3	4
Set output STL		1	2
Reset flag LAD	2	7	8
Reset flag STL		8	7
Reset counter		2	3
Reset timer		11	10
Reset timer		8	7

Multibit load commands

Function	Cycles	Number of bytes
Load EW	3	5
Load AW	3	5
Load MW	9	9
Load ZV	10	8
Load ZW	10	8
Load TV	10	8
Load TW	10	8
Load register	9	9
Load constant	4	7
Load FU32-47	15 17	16
Load FU0-23	9	8
Load FW	9	8

Multibit instructions

Function	Times min max	Number of bytes
To AW	2	3
To MW	9	9
To ZV	10	8
To ZW	12	11
To TV	10	8
To TW	12	11
To register	9	9
To FU32-47	14	16
To FU0-23	9	8
To FW	9	8

Increment / Decrement

Function	Times min	Times max	Number of bytes
Inc/Dec AW	1		2
Inc MW	9	15	14
Dec MW	10	16	15
Inc ZV	15	21	8 (LIB)
Dec ZV	16	22	8 (LIB)
Inc ZW	28	44	15
Dec ZW	27	31	12
Inc register	9	15	14
Dec register	10	16	15

Logical links of single-bit operands

Function	Times min	max	Number of bytes
and direct	2		2
and block	16	17	26
or direct	2		2
or block	16	17	26
exor direct	7	8	11
exor block	19	20	31
not	1		1

Logical links of multibit operands

Function	Times min	max	Number of bytes
and direct	8		10
and block	36		36
or direct	8		10
exor block	36		36
exor direct	8		10
exor block	36		36

direct = Operation is executed directly
Block = Expression is interlaced, a stack operation must be carried out

Arithmetical links of multibit operands

Function	Times min	Times max	Number of bytes
Addition	10		10
Subtraction	11		11
Multiplication a * b:			
a,b positive	86		
a or b negative	112		7 (LIB)
a,b negative	104		
Division a / b:			
a,b positive	525		
a or b negative	551		7 (LIB)
a,b negative	540		

Comparison of multibit operands

Function	Times min	Times max	Number of bytes
=	8	13	16
<	7		12
< =	8	11	13
>	7	10	12
> =	8	11	13
< >	8	14	16

Special operations

Function	Times min	max	Number of bytes
SWAP	3		3
SHL	7		7
SHR	6		6
ROL	8		8
ROR	8		8
CPL	6		6
NEG	8		10
DUD	387	436	3 (LIB)
DED	221		3 (LIB)

Notes