Selectron

Selectron Lyss Ltd.
Industrial Electronics
Bernstrasse 70
CH-3250 Lyss Switzerland
FAX 032 844 820 Telex 934 469
Telefon 032 856 161

SELECONTROL® PMC 30
System Description

C O N G R A T U L A T I O N S
****

You have made the right choice !

SELECONTROL® PMC

The future-orientated system
for industrial automation

For New-comers

1. Read carefully through sections 1 and 2. You will then learn the most important things about SELECONTROL® PMC.
2. In section 9.1, we have made a summary of what you have to do to prepare your SELECONTROL® PMC for programming.
3. The instruction set for programming via a terminal is given in section 5.1. Here you will also find the first of many programming examples.
4. When you want to know more about the instruction set, you will find a full description in section 6.
   The program examples will help you to learn and understand your SELECONTROL® PMC as quickly as possible.
5. If you enjoy team-work and do not have much time, why not sign up for one of our basic seminars?
   (No prior knowledge is required).

For Experts

1. You will find all you need to know about the hardware in section 3.
2. After studying section 6 thoroughly, you will be a real programming buff !
3. Sections 7 and 8 go into details of analog value processing and communications. You will be pleasantly surprised at just how simple these tasks are to master.
4. In our advanced seminars we can give you many useful tips concerning word-wise processing, arithmetic operations, jump routines, shift operations, code changing, analog value processing and communication.

For Connaisseurs

You already know all about SELECONTROL® PMC.

For you, we have summarized the most important points in tabular form at the beginning of section 6 as an aide-mémoire.

PART I

PART II

1 INTRODUCTION

1.1 General

Today, freely programmable automation systems are being put to work in every branch of industry.

Acting as the link between mechanics and electronics, such systems form the basis for the automation of machines, installations, procedures and processes.

SELECTRON LYSS AG has set itself the target of supplying its customers in the industrial electronics field with control and driver systems that are at the very forefront of technology.

The Controllers in the **SELECTRON** system cover a whole range of needs from the smallest, with a unit in Euro-board format, through compact controllers and up to complete modular systems.

The **SELECONTROL** Controllers can also be readily programmed by machine builders or constructors or by plant electricians without any special knowledge of electronics.

The user-friendly **SELECONTROL programming technique** enables programming to be carried out from any arbitrary documentation (e.g. flow diagram, function plan, procedure, switching plan time/path diagram, etc.).

The SELECONTROL programming technique is especially helpful in the programming of sequential procedures. Thanks to the freely-programmable step counters (for incremental sequences) and the possibility of making arbitrary jumps forwards and backwards, sequential programs can be easily and clearly structured. The operational security is considerably increased through the use of the SELECONTROL programming technique.

Highly efficient software packages are available for IBM PC's and compatible machines to provide a very convenient form of program and documentation preparation.

A further speciality of the SELECONTROL Controllers is the built-in **analysis and diagnostic center.** Routines, operational statuses and any fault indications are constantly displayed. Commissioning a system is simplified through the provision of these diagnostic possibilities. Production stoppages and down-times are reduced to a minimum.

Regularly-held **seminars** provide interested parties and users with practical training concerning the hardware and software aspects of SELECONTROL Controllers.

SELECTRON is also able to help the user with system support. An excellent knowledge of control techniques and automation, coupled with many years of experience, provide the foundation for competent **advice** for **planning** of economic solutions.

Our **world-wide sales and service organisation** is easily accessable to you and your customers.

1.2 SELECONTROL® PMC
The Controller with the window on the future

PMC is the abbreviation for **P**rogrammable-**M**ulti-**C**ontroller.

The SELECONTROL® PMC is much more than a conventional programmable controller with a memory.

The SELECONTROL® PMC is a future-orientated system for industrial automation purposes

The SELECONTROL® PMC is used for:

- **Control**
- **Measurement**
- **Regulation**
- **Positioning**
- **Communication**

in every branch of industry, such as:

Machine fabrication

- Tool machines
- Handling equipment
- Packing machines
- Special machines

Installation construction

- Production installations
- Manufacturing installations
- Assembly lines
- Conveyance and storage installations
- Test, check and sorting rigs

Control and switch-panel construction

- Heating, ventilation and air conditioning
- Environmental engineering
- Building automation.

The robust construction of the SELECONTROL® PMC enables it to be used under difficult environmental conditions.

The SELECONTROL® PMC extends the SELECONTROL® PLC system range at the higher performance end of the scale.

Use of the SELECONTROL® PMC becomes economical with as little as about 64 digital inputs/outputs or in even smaller systems when the processing analog data, arithmetic functions, communication facilities or text handling are required.

A system can be expanded up to a **maximum of 256 inputs/outputs** (16 modules).

Automation of complete installations and processes can be achieved with 1 master and up to 8 slave systems which together give **up to 2160 inputs/outputs.**

1.3 Features of the SELECONTROL® PMC

The most important features of this system are:

1) Advanced system technology

You can configure the optimum system to suit your requirements yourself, simply and economically.

The following processor modules are available to enable you to do this:

- Central processor unit for control, measurement and regulation, bit, byte and word processing, arithmetic operations
- Communications processor to link several SELECONTROL® PMC systems together and SELECONTROL® PMC to a SELEDATA® MPS positioning system.
- Digital and analog input/output modules as well as analog timers.

2) Conventional programming systems

A special programming unit is no longer required. You can enjoy convenient programming even on a commercially available non-intelligent terminal because the necessary firmware is incorporated into the SELECONTROL® PMC processor unit. Naturally, any **hand-held computer** or **PC** is also admirably suitable.

The SELECONTROL® PSU Programming and Service Terminal is available for you to use in an industrial environment.

It can be built into a front panel (splash-proof) and can be used as a text display and input terminal.

3) Freely-programmable LCD display

SELECONTROL® PMC shows what is happening! The built-in LCD display enables you to keep in touch with routines and operational statuses, elapsed times and counter states as well as much else besides. These diagnostic possibilities reduce production stoppages and down-times to a minimum.

4) User-friendly programming technique

A good program is transparent and clearly constructed. The SELECONTROL programming technique effectively supports this requirement. Complex procedures can be simply and clearly structured by using the **freely-programmable PMC routines** with the possibility of arbitrary jumps forwards and backwards.

Every command to the machine or installation being controlled corresponds to a part program and is assigned to an instruction in the routine (step counter).

Each new program step is only performed when the preceding one has been completed.

The **operational security** is considerably increased without having to program-in complicated interlocks.

The current program step can be continuously displayed on the LCD display.

If the conditions required to complete the next step have not been fulfilled (e.g. a limit switch has not been activated), the program halts at the corresponding step.

The problem can be quickly localized thanks to the step counter display and the LED interface display (**fault diagnosis**).

5) Functional assembly and cabling

The choice of the most suitable type of assembly is yours, e.g. **rear-panel mounting** via robust module supports or 19" **rack-mounting** through the use of the corresponding accessories. The unique connection technique facilitates **rational cabling**; wire up the terminal strip - plug in - switch on your SELECONTROL® PMC - go!

The SELECONTROL® PMC offers you even more . . .

Do you appreciate convenient programming?

Then use you **IBM PC** or any compatible computer for writing your programs. The highly efficient SELECONTROL® CAP 3000 software package will support you in preparing programs and documentation.

Are you considering communication?

- **Decentralized automation** of complete installations with exchanges between several systems.
- **Implementation and supervision** of machine functions or process procedures via terminals.
- **Print-out** of production and plant data as well as alarm messages with time and date.

The future-orientated SELECONTROL® PMC 30 offers you all these possibilities.

Do you want to control and drive?

We have the right solution for this, too - the SELEDATA® MPS positioning system.

1.4 Mode of operation

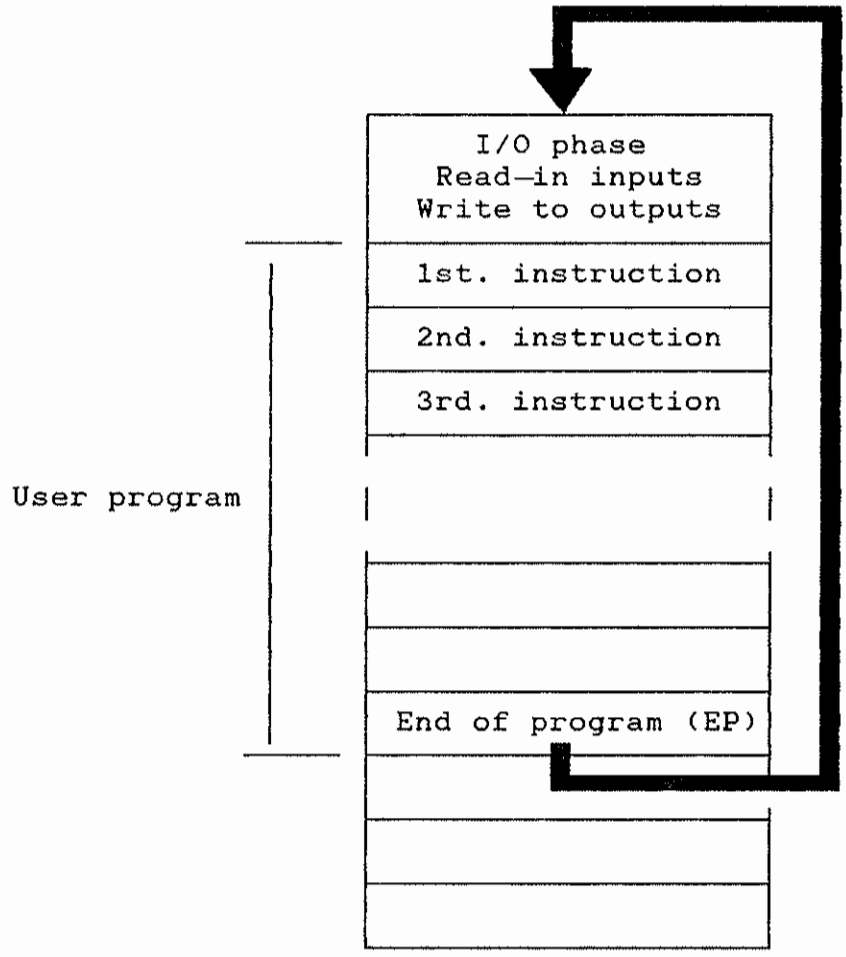
The SELECONTROL® PMC operates cyclically (scanner principle).

During the I/O phase at the beginning of each program cylce:

- the signal statuses at the inputs are read-in and stored in the input image buffers for the duration of the following program cycle, and
- in the meantime, the data that has been accumulated in the output image buffers are fed to the outputs.

The program is then run through sequentially; one instruction being executed after another according to the instruction list.

The data for the outputs is built-up continuously in the output image buffers.



The cycle time depends on the length of the program and the nature of the command instructions.

Program segments that require especially short reaction times are better handled by sub-routines.

2 System summary

You can put the optimum system configuration together yourself modularly, simply and economically.

You need the following for a functional system:
- a module mounting frame
- central processor unit
- I/O modules and analog timers, as necessary

The central processor unit is always inserted in the first slot on the left in the frame (as shown by the inscription).

The I/O modules and analog timers can be inserted arbitrarily in the other slots (00 ... 15).

The following range of modules is available for you to choose from:

2.1 Modules

Processor modules

CPU 30 **Central Processor Unit**
Central processor unit for bit, byte and word-processing as well as arithmetic operations.It is equipped with a freely-programmable LCD display and has an integrated power supply unit. A ROM 30 EPROM module is included.

CPU 31 **Central Processor Unit**
Central processor unit for bit and word-processing as well as arithmetic operations. It is equipped with a freely-programmable LCD display and has an integrated power supply unit.
Two 100kHz up/down counter inputs are provided or an incremental transducer with rotational sense indication can be connected.
A ROM 30 EPROM module is included.

TCC 30 **Telecommunication Controller**

Communications processor for **linking several SELECONTROL® PMC systems and to connect with the SELEDATA® MPS positioning system.**

TCC 31 **Telecommunication Controller**

Communications processor for data exchanges with computers and peripherals via standardized interfaces for the **logging of production and plant data** as well as printing out alarm messages with the date and time.

Digital input/output modules

DIM 30 Digital **Input Module**
16 electrically isolated inputs **18...36V dc**

DIM 31 Digital **Input Module**
16 electrically isolated inputs **90...240V ac/dc**

DIM 32 Digital **Input module**
16 electrically isolated inputs **12...60V ac/dc**

DOM 30 Digital **Output Module**
16 electrically isolated outputs **18...36V dc**
2A, short circuit protected

DOM 31 Digital **Output Module**
8 **Relay outputs** 250V ac

Analog timer modules

ATM 30 **Analog Timer Modul**
8 analog timers
Time range **30ms...10h**
4 timers externally settable

ATP 30 **Analog-Timer Potentiometer**
Potentiometer-set timer for the manual setting of analog times

Analog input/output modules

AIM 30 **Analog Input Module (0/+ range)**
8 analog, electrically isolated input channels

AIM 31 **Analog Input Module (+/- range)**
8 analog, electrically isolated input channels

Analog input circuits for AIM 30/31

AID 30	for 4 channels:	0 ... 20mA or -20 ...+20mA
AID 31	for 4 channels:	0 ...+10V or -10 ...+10V
AID 32	for 2 channels:	Pt100 (0...500mV)

AOM 30 **Analog Output Module (0/+ range)**
4 analog, electrically isolated output channels for output signals of 0 ... +10V. (Range modules AOD 30 are necessary for outputs of 0 ... +20mA).

AOM 31 **Analog Output Module (+/- range)**
4 analog, electrically isolated output channels for output signals of -10 ... +10V. (Range modules AOD 30 are necessary for outputs of -20 ... +20mA).

Range modules for AOM 30/31

AOD 30	for 2 channels:	0 ... +20mA (AOM 30) or -20 ... +20mA (AOM 31)

Accessories

BPU 08 **Back-Panel Unit**
Mounting for a central processor unit and **8 Modules** (128 inputs/outputs)

BPU 12 **Back-Panel Unit**
Mounting for a central processor unit and **12 Modules** (192 inputs/outputs)

RMA 12 **Rack-Mounting Accessory**
Adapts the BPU 12 for 19"-rack mounting

EPU 08 **Expansion Unit**
Extension mounting for the BPU 08 to enable the system to be expanded up to a maximum of **16 Modules** (256 inputs/outputs)

EXM 30 **Expansion Module**
Expansion module for the bus link between the BPU 08 and the EPU 08
Screened cable included

BSM 30 **Blind Space Module**
Dummy module to fill unused module slots

PSM 30 **Power Supply Module**
110/220V ac in 24V dc out, 1A

RAM 30 **RAM-Board** (32k x 8)
RAM-Memory for 4000
control instructions.
Buffer battery included.
With connector for
EPROM-Board and integrated
EPROM-Loader.

ROM 30 **EPROM-Board** (32k x 8)
EPROM-Memory for
4000 control instructions.

SIM 30 **Simulator**
Simulator for 16 inputs
with parallel interface for an encoder

MAL 30 **Module Address List**
Module identification strips
1 sheet of 12 identification strips
for labelling the inputs and outputs

CCA 30 **Data cable**
Cable with 9-pin socket (female)
for the connection of e.g. an **IBM-PC or SELECONTROL² PSU**
Length 1.5m

CCA 31 **Data cable**
Cable with 9-pin plug (male)
for the connection of e.g. a **Microscribe-Terminal**
Length 1.5m

CCA 32 **Data cable**
cable with 25-pin socket (female)
for the connection of e.g. a **Commodore PC**
Length 1.5m

CCA 33 **Data cable**
cable with 25-pin plug (male)
for the connection of e.g. an **NEC-Terminal**
Length 1.5m

2.2 Programming systems

PSU 30 **Programming and Service Unit**
For on-line programming and data input.
LCD display: 4 rows of 40 characters.
(A data cable CCA 30 is necessary for
on-line operation with SELECONTROL® PMC).

PSU 31 **Programming and Service Unit with memory**
For on and off-line programming
of the SELECONTROL PMC.
With integrated 32kByte RAM memory,
battery-buffered.
LCD display: 4 rows of 40 characters.
(A data cable CCA 30 is necessary for
on-line operation with SELECONTROL® PMC).

NEC 8201A **Programming Terminal**
Handheld Computer with 16k RAM-memory.
LCD-Display, 8 rows of 40 characters.
With communications program.

Convenient programming with a Personal-Computer

CAP 3000D **Demonstration diskette**
Computer aided programming
Software packet for
convenience programming and
documentation preparation with an **IBM-PC**
or compatible systems (MS-DOS).
(Limited to 100 instructions)

CAP 3000 **Program diskette**
Computer aided programming
Software packet for
convenience programming and
documentation preparation with an **IBM-PC**
or compatible systems (MS-DOS).
Operating guide can be in English,
French, German or Italian

CCP 30 **Computer Communication Program**
Program for the IBM-PC and compatible
Systems to use them as programming terminals

CCP 31 **Computer Communication Program**
Program for the Commodore VC/SX 64
for use as a programming terminal

2.3 General technical data

SELECONTROL PMC technical data	
Central processor	Bit, digit, byte and word processing. Arithmetic functions.
No. of **Inputs/outputs**	
- per system	max. 256 (16 modules)
- per network	2160 (1 master + 8 slaves)
User memory	32k x 8 (4000 instructions) Plug-in memory modules; buffered RAM or EPROM
Command execution time	2µs (e.g. AND Operation)
Data registers	1024 (8-bit) or 512 (16-bit)
Counters and timers	512 (range 0...9999)
Markers	368
Step counters	16 x 100 Steps
Fast counters	2 x 100kHz (Option: CPU 31)
System power supply	Integrated in CPU Supply voltage 18...36V dc
Programming interface	RS 232 C (V24)
Inputs with electrical isolation	
digital	16 per module, 18...36V dc, 16 per module, 90...270V ac/dc
analog	8 per module Range modules for: 0...+10V, 0...+20mA, Pt100 -10...+10V, -20...+20mA
Outputs with electrical isolation	
digital	16 per module, 18...36V dc, 2A short circuit protected
	8 relays per module 250V, 5A
analog	4 per module Outputs for 0...+10V (0...+20mA) -10...+10V (-20..+20mA)
Analog timers	8 per module Range: 30ms...10hrs (4 externally settable)

SELECONTROL PMC technical data	
Isolation category	Group C 250 (as per VDE 0110)
Protection class	IP 20
Operating temperature	0...55°C
Storage temperature	-40...+70°C
Humidity class	F (as per DIN 40040)
Mechanical requirements	Rigid in-situ mounting Light vibration permissible
Housing colour	Beige-grey RAL 7006

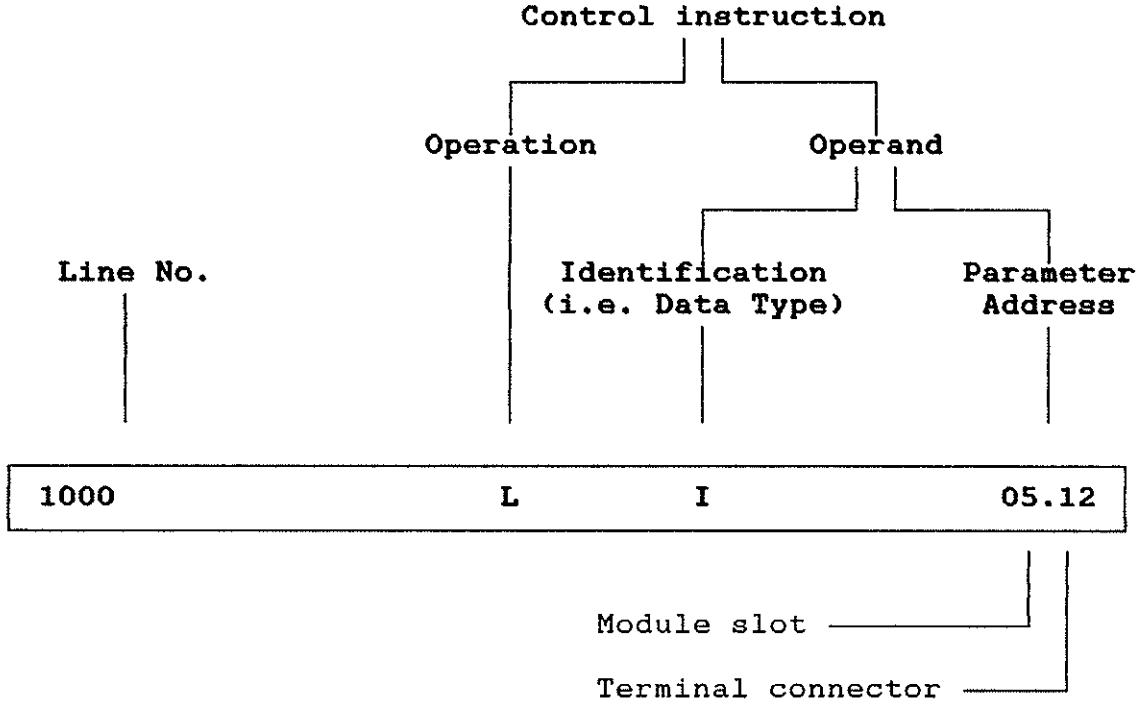
2.4 Programming technique

The program consists of a series of control instructions.

The format of the control instructions for the SELECONTROL® PMC is based on the recommedations of the **DIN 19239** standard.

A control instruction is made up of an operation part and an operand part; the operand part consisting of an identification and a parameter.

The mnemonic abbreviations used have been based on the English language.



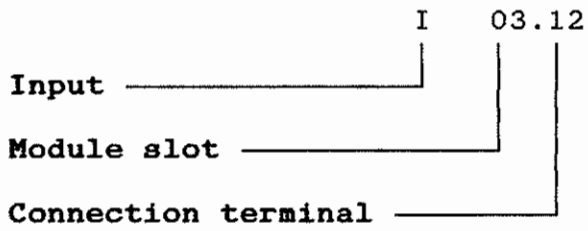
Line No.: 0000 to max. 4000

2.5. Operands

The operands are identified as follows:

Inputs	**I** nn.nn	(00.00 ... 15.15)
Outputs	**O** nn.nn	(00.00 ... 15.15)
Markers	**M** nn.nn	(16.00 ... 38.15)
Special markers	**M** 40.nn	(40.00 ... 40.15)
Step counters	**S** nn.kk	(00.00 ... 15.99)
Data registers	**D** nn.nn	(00.00 ... 15.63)
Constants	**K** 0kkkk	(00000 ... 0FFFF)
Extended data range	**R** nn.nn	(00.00 ... 15.63)

A parameter thus has a connotation as follows:



Input/outputs

The input and output modules can be inserted as required in **arbitrary slots.**

The SELECONTROL® PMC can handle a **maximum of 256 inputs and outputs (16 modules).**

The following operations can be carried out using these inputs and outputs:

- Logic operations (1-bit processing)
- Data transport operations
  (digit, byte and word operations)

Such operations are always carried out on the image buffers and not on the physical inputs and outputs themselves.

The image buffers are up-dated during the I/O phase.

Markers

The SELECONTROL® PMC is provided with **368 Markers.**

In addition, the unused inputs and outputs as well as the data registwers can be used as extra markers.

> Only very few markers are necessary for programming purposes thanks to the comprehensive instruction set.

The following operations can be carried out using the markers:

- Logic operations (1-bit processing)
- Data transport operations
  (digit, byte and word operationsn)

Special markers

The special markers contain system information:

Special marker	Meaning	Remark
M 40.00	Logical 1	Used for operations that have to be continuously executed (e.g. data transport)
M 40.01	1st cycle	Signals the first program cycle. Used for operations which only have to be executed during the first cycle (e.g. for resetting data registers upon warm start).
M 40.02	0.1s clock	Internal 10Hz timing pulses.
M 40.03	1.0s clock	Internal 1Hz timing pulses.
M 40.04	0.01s clock	Internal 100Hz timing pulses
M 40.05	Shift register (1-Bit)	Used for the 1-Bit rotation (left or right)
M 40.06	Short-cct	Outputs switched off because of a short circuit.
M 40.07	Arithmetic error	Signals arithmetic error e.g.: - division by 0 - operand with hex values - binary/decimal conversion giving an operand > 9999
M 40.08	Cycle time exceeded	Signals that the cycle time has exceeded 50ms. The time carry for 'TF' (TIMER FAST) will be incorrect.
M 40.09	Carry / Borrow	Carry-over information e.g.: - result of addition > 9999 - result of subtraction < 0000
M 40.13	Warmerr	Signalize in the first program cycle that the buffer battery of the RAM has no more capacity.
M 40.14*	Communication Error Flag	Data transfer Master-Slave incorrect
M 40.15*	Toggle Flag	Data transfer Master-Slave correct Changes after each transmission its status

*Additional markers only used for the communication version SELECONTROL® PMC 22 EC.

The appropriate marker is active (logical 1) when the corresponding situation is true.

Step counters

The SELECONTROL® PMC is equipped with 16 100-step counters.

> Complex procedures can be simply and clearly programmed in a structured manner using the step counters to help handle sequential routines.
>
> The construction of the program is completely unrestricted. Arbitrary jumps forwards and backwards may be made and parallel programs can be executed.
>
> The operational security is considerably increased through the use of sequential routines (using step counters).

The step counters can be used for the following operations:

- Logical read operations (L, A, O, XO)
  (e.g. A S 03.27 AND step counter 03 at step 27)

- Logical SET "S" operations
  (e.g. S S 03.28 SET step counterr 03 at step 28)

- Data transport operations
  "FTB" FETCH BYTE
  (FTB S 03.00 reads the current step from step counter 03)
  "STB" STORE BYTE
  (STB S 03.00 writes the data word (byte) as a step)

> When a step counter is set to a specific step (e.g. S 00.01) the previous step (00.00) is automatically deactivated.

The preceding step must not be deactivated by using the "R" (Reset) instruction.

Data registers (D-Registers)

The markers contain 1-bit logic signals (0 or 1) whereas the data registers can be used for **multi-bit data** (e.g. counter data in the range 0000 ... 9999).

The SELECONTROL® PMC is provided with 1024 8-bit data registers (D 00.00, D 00.01, ... D 15.63)

The contents of the **data registers D 15.56 ... D 15.63** are displayed on the central processor unit's LCD display.

The data registers can be accessed byte-wise (FTB, STB) or word-wise (FTW, STW)

A logic operation in connection with a data register checks the content of the register for the value 0000

Extended data range (R-Registers) / Access to the peripherals

The extended data range enables modules which involve large quantities of data (e.g. the analog input module, AIM, the analog output module, AOM, or the communications controller, TCC) to be handled.

A data capacity of 64 bytes is available to each of the module slots (R XX.00, R XX.01 ... R XX.63).

The two lowest bytes (R XX.00 and R XX.01) assigned to each slot correspond to the physical inputs or outputs.

> The extended data range (R-Registers) can be accessed byte-wise (FTB, STB) or word-wise (FTW, STW).

There is a considerable difference between the extended data range and the other types of data.

> Access to the R-Registers always relates to the current statuses at the inputs and outputs and not to the images.

This fact can be used, for example, to interrogate the inputs by means of a sub-routine several times during a program cycle.

Constants

Constants are used in the programming of SELECONTROL® PMC for the following purposes:

- **Arithmetic operations**
  4 basic operations (ADD, SUB, MUL, DIV) with 4-digit BCD constants constants (0000...9999)

- **Word logic de mots** (AW, OW, XOW)

- **Data transport operations** (FTW, FTB, FTD, FIR, FID, FTR)

- **Word-logic-operations** (AW, OW, XOW)

- **Comparison operations** (LT, EQ, GT, LTE, GTE)

- **Jumps** (JP, JCF, JCT, JS, LB, RET)
  as jump destinations with 2-digit BCD constants (00...99)

The constants are handled as 4-digit values in each case by adding preceding zeros (even for jump, digit or byte operations)

Formatted presentation: 00000...0FFFF

HEX-constants (A, B, C, D, E, F) always have to be entered with a preceding zero.

2.6 Instruction set

The instruction set conforms to DIN 19239 (English) and comprises the following operations:

Logical operations: reading

L	Load (If)
LN	Load Not (If Not)
A	And
AN	And Not
O	Or
ON	Or Not
XO	Exclusive-Or
XON	Exclusive-Or-Not
AB	And-Block
OB	Or-Block

Logic operations: writing:

=	Then
=N	Then Not
S	Set
R	Reset

Triggering:

TRG	Trigger

Counting operations:

CU	Count Up
CD	Count Down

Timing operationsn:

TF	Timer fast (0,1s)
TS	Timer slow (1,0s)

Data transport:

FTW	Fetch Word
FTB	Fetch Byte
FTD	Fetch Digit
STW	Store Word
STB	Store Byte
STD	Store Digit
SOH	Serial out hex
SOD	Serial out digit

Word operations:

AW	And Word
OW	Or Word
XOW	Exor Word

Comparison operations:

LT	Less than
LTE	Less than or equal
EQ	Equal to
GT	Greater than
GTE	Greater than or equal

Arithmetic operations:

ADD	Add
SUB	Subtract
MUL	Multiply
DIV	Divide
FTR	Fetch Aux.-Reg.

Jump Instructions:

JP	Jump unconditionally
JCF	Jump conditionally on false
JCT	Jump conditionally on true
LB	Jump destination label
JS	Jump to sub-routine
RET	Return

Shift operations:

SFL	Shift left
SFR	Shift right
RTR	Rotate right
RTL	Rotate left
BSU	Block shift up
BSD	Block shift down
BR	Block reset

Step counters:

INC	Increment, step + 1
DEC	Decrement, step - 1

Code changing:

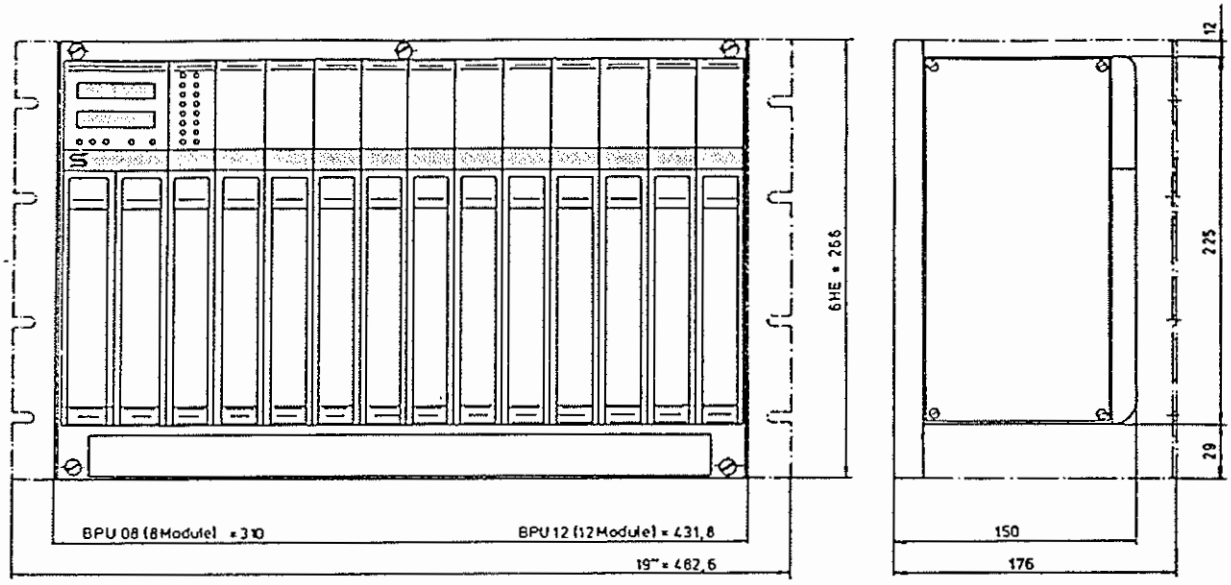
BID	Binary into BCD
DEB	BCD into binary

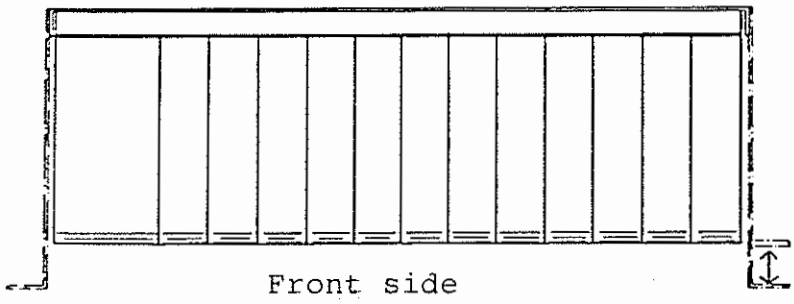
Table look-up:

LKP	Look-up truth table

Program organisation:

EP	End of program
NOP	No operation





Front side

adjustable

3 HARDWARE DESCRIPTION

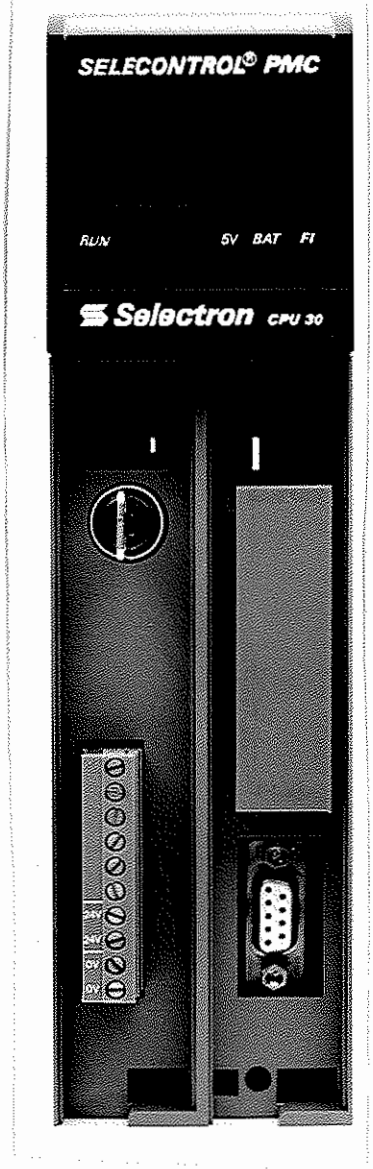
This section describes the hardware of the SELECONTROL® PMC.

The general technical specifications can be found in the section entitled "System Summary".

3.1 Processor modules

Central Processor Unit, CPU

The central processor unit is responsible for all tasks concerning control, measurement and regulation. It contains:



- the microprocessor for processing the bit, digit, byte and word operations as well as the arithmetic functions.
- a power supply unit for the internal system powering of the CPU and up to a maximum of 16 modules.
- a freely-programmable display for the presentation of operational statuses, process sequences, fault messages, counter contents, etc.
- a key switch to select the "RUN", "RUN-MONITOR" or "PROGRAM" operating mode.
- an RS 232C interface for connecting a programming system.
- a program memory for the firmware.
- a data memory for the process data.

Technical specifications

Central Processor Unit	CPU
Microcontroller	Intel 8031 (8-bit)
User memory	32k x 8-bit buffered RAM or EPROM for 4000 instructions
Internal memories – Operating system – Data memory	 32k x 8-bit EPROM 8k x b-bit RAM, buffered
Power consumption	4W (additional requirement per module 0,25W)
Processing time - per binary operation - per word operation	 2µs (AND operation) 14µs (FTW D XX.XX)
Markers	368 (1-bit)
Data registers	1024 (8-bit) or 512 (16-bit)
Counters and timers	512 in the register range Counting range 0000 ... 9999 (Times: 100ms ... 999.9s or 1s ... 9999s)
Address range	Module slots 00 ... 15 (16 modules)
No. of inputs/outputs	256 max. (16 modules)
Weight	800g (28oz)

Displays

The central processor unit has a freely-programmable LCD display with 2 rows of 8 characters.

The display can be used to show operating statuses, process sequences, error messages and the status of counters relating to the installation under control.

In the event of a system fault occuring, this is also shown on the display.

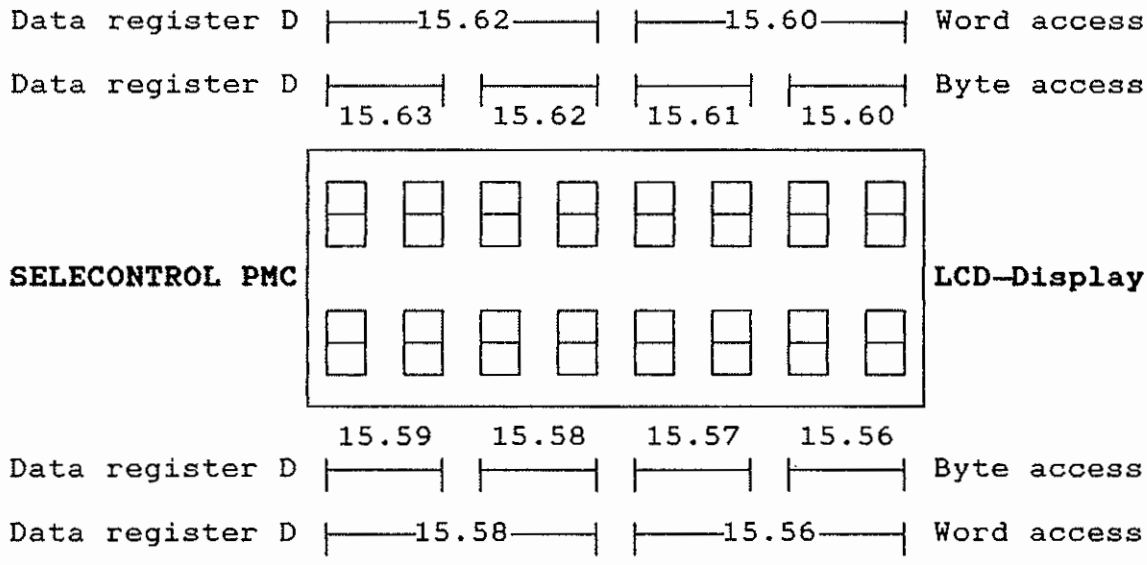
System fault display:

CPU system fault	Meaning
E - 01	Processor-RAM defective
E - 02	System-RAM defective
E - 03	Memory module defective
E - 04	No memory module present or memory module not inserted correctly
E - 05	EPROM badly programmed
E - 06	No valid program in EPROM
E - 07	Program overflow in EPROM
E - 08	No valid program in RAM
E - 10	Power supply missing(< 18V)

The central processor unit has to be repaired if either the fault message E - 01 or E - 02 appears.

Programming the display:

The LCD display is addressed via data registers.



The display data is in binary code:

Binary	Hex	Display
0 0 0 0	0	0
0 0 0 1	1	1
0 0 1 0	2	2
0 0 1 1	3	3
0 1 0 0	4	4
0 1 0 1	5	5
0 1 1 0	6	6
0 1 1 1	7	7
1 0 0 0	8	8
1 0 0 1	9	9
1 0 1 0	A	-
1 0 1 1	B	E
1 1 0 0	C	H
1 1 0 1	D	L
1 1 1 0	E	P
1 1 1 1	F	

The binary code enables the display to contain "-" and blank spaces which allows the displayed information to be clearly formatted.

Examples: E - 27 means: error message 27
3 - 19 means: step counter 3, counter content = 19

LED's indicate the operational status and error messages of the central processor unit.

Operational status indicator (green LED):

Indication	Meaning
RUN cont. light	Program running error-free ("RUN" mode)
RUN blinking	Program error (e.g. EP operation missing) or system fault as per LC-display
RUN extinguished	Operating mode "PROGRAM"
5V cont. alight	Internal 5V supply OK
5V extinguished	Internal 5V supply missing

Fault indicator (red LED):

Indication	Meaning
BAT alight	Buffer battery in the CPU is discharged The buffer battery must be exchanged if the "warm start" operating mode is required. In case of a discharged battery, the data remanence is no longer quaranteed. Process data loss is signalized with the special marker M 40.13 (Warmerr).
FI alight	System fault. Watch-dog has been activated. Re-start after power-down. A warm start is not possible.

Power supply

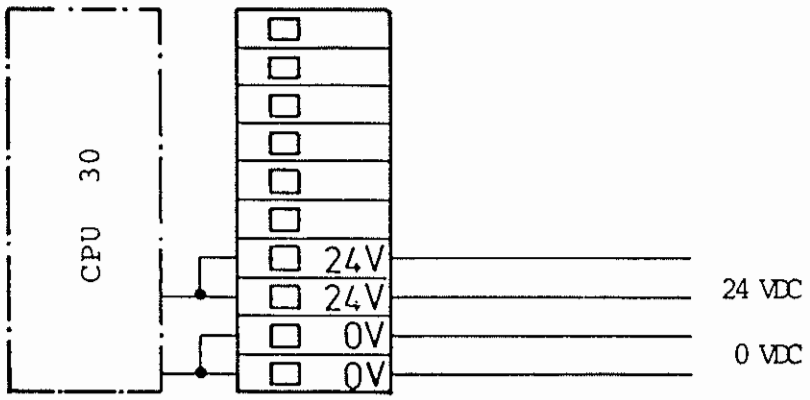
The power supply unit for supplying power to the internal system is integrated into the CPU.

Besides the CPU, a maximum of 16 modules can also be supplied.

Intelligent modules have their own power supply.

System supply	CPU
Supply voltage	24V dc (18 ... 36V dc)
Ripple	50% at 24V$_{rrr}$ (full-wave rectified, unsmoothed)
Output voltage	5V short circuit protected
Electr. isolation	None
Bridging time during power interruption	30ms min.
Buffer battery for data memory	Integrated. Buffering time: 10 years approx.

Connection diagram:



Operating mode switch (key-switch)

The key-switch is used to select the operating mode.

The following operating modes are possible:

"PROGRAM" **Program input**

A new program can be entered or an existing one can be altered.

A currently running program is interrupted.

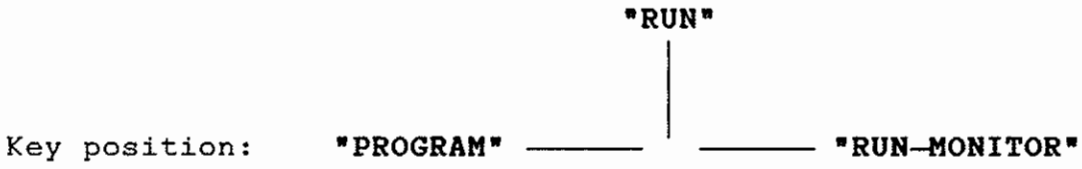
"RUN" **Program runs (continuous operation)**

No intervention can be made in the program that is running when in this mode.

"RUN-MONITOR" **The program runs**

Process data (e.g. counter contents) can be displayed and altered via the programming terminal while the scanner cycle runs.

This mode is useful, for example, during commissioning.



The key is removeable in the "RUN" and "RUN-MONITOR" positions.

Programming interface

A standard terminal or a personal computer can be connected to the RS 232C programming interface for inputting a program or to observe a process.

Technical specifications:

RS 232C programming interface	Technical data
Connector	9-pole, D-type, sub-min.
Electrical isolation	None
Baud rates	150 / 300 / 1200 / 4800 set by DIP switches
Level, unloaded	±5V

Pin assignment:

RS 232C programming interface	Pin assignment
$\overline{\text{TXD}}$	Pin 2
$\overline{\text{RXD}}$	Pin 3
0 V	Pin 7
CTS	Pin 5

Setting the Baud rate:

The Baud rate can be set by means of DIP switches at the rear of the central processor unit, thus:

Baud rate	Switch 1	Switch 2
150	ON	ON
300	OFF	ON
1200	ON	OFF
4800*	OFF*	OFF*

* Factory setting

Setting the data transmission format:

The data transmission format can be set-up by means of DIP switches at the rear of the central processor unit.

Transmission format	Switch 3
1 start bit/8 data bits/no parity/1 stop bit	ON
1 start bit/7 data bits/even parity/1 stop bit	OFF*

* Factory setting

RAM 30 RAM cassette
ROM 30 EPROM cassette

> Programming is only possible if a RAM board is inserted in the central processor unit

The user program is stored in a RAM during programming, testing and commissioning.

Memory size: 32k x 8-bit
4000 instructions max.

The RAM cassette is fitted with a buffer battery which gives a buffering time of approximately 6 years.

The program is stored in an EPROM for continuous use.

The EPROM cassette has to be plugged onto the RAM cassette to transfer the program from the RAM to the EPROM. The transfer is started by means of the programming terminal. No special EPROM loader is necessary.

It is also possible to copy the program from one RAM cassette to a second RAM cassette "R)".

Central Processor Unit CPU 31

The Central Processor Unit CPU 31 differs from the CPU 30 through the inclusion of two additional 100kHz counters.

These counter can be operated in two ways:
- as event counters (up/down counters)
- for connecting incremental encoders with rotational sensing

When a Central Processor Unit Type CPU 31 is used, no module must be inserted in slot 15.

Technical data

CPU 31	Counter inputs
Number of counters	2
Counting range	0000 ... 9999 (4-digit BCD)
Counting rate:	
- Event counter	100kHz max.
- Incremental	50kHz (pulse doubling) 50kHz (pulse quadrupling)
Duty ratio	1:1
Counter addresses	R 15.60 ... R 15.63
Access	Byte or word-wise
Electrical isolation	Yes (opto-isolators)
Reset	Under program-control
Read-out	Under program-control
Input voltage:	
- standard, nominal	+24V dc (18 ... 36V dc)
- optional	+12V dc (8 .... 18V dc)
- optional	+5V dc (4.5 ... 7V dc)
Input current, typ.	18mA (at 24V dc) 17mA (at 12V dc) 15mA (at 5V dc)
External connections	Front panel connector (together with the 24V system supply)
Weight	850g (30oz)

The contents of the counters are transferred into RAM during the I/O phase.

The values stored in this RAM remain unchanged throughout the following program cycle.

The value for the appropriate counter is read out of the RAM with the commands:

FTW R 15.60	or	FTB R 15.60 FTB R 15.61	Counter 0
FTW R 15.62	or	FTB R 15.62 FTB R 15.63	Counter 1

The special command, **STB R 15.59,** can be used to up-date the counter content data under program control during the current cycle, if necessary.

The special commands, **STB R 15.60** and **STB R 15.62** can be used to reset counters 0 and 1 respectively.

Operating modes

There is a DIP switch on the rear panel of the module to set the operating mode.

Switch 1	ON	Counter 0 as an event counter
	OFF	Counter 0 for incremental encoder input
Switch 2	ON	Pulse quadrupling, Counter 0 *
	OFF	Pulse doubling, Counter 0
Switch 3	ON	Counter 1 as an event counter
	OFF	Counter 1 for incremental encoder input
Switch 4	ON	Pulse quadrupling, Counter 1 *
	OFF	Pulse doubling, Counter 1

* Only effective in conjunction with incremental encoders.

Each counter can accept two (electrically isolated) input signals (A and B).

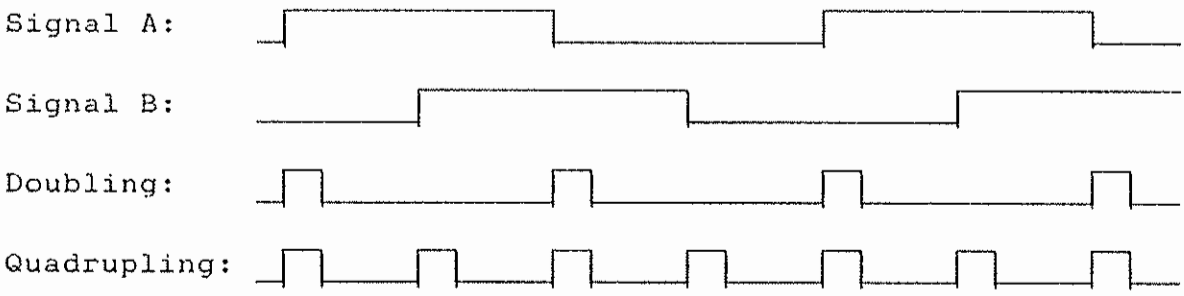
These signals have the following meaning in the two operating modes:

Event counting:
Signal A = Count (on positive edge)
Signal B = Counting direction:
- active: count up
- inactive: count down

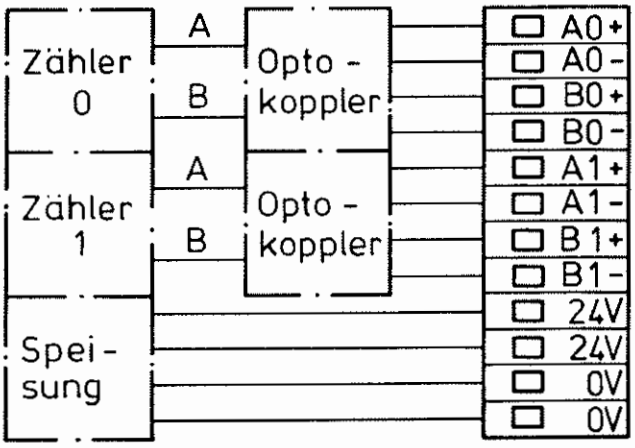
Incremental encoders: The A and B inputs serve to connect the 90° out-of-phase encoder signals. When working with incremental encoders, it is often desirable to attain twice the resolution of the encoders. This can be achieved by means of pulse doubling or quadrupling through the appropriate setting of switch 2 (or 4).

Doubling: Counting on the positive and negative edge of A

Quadrupling: Counting on the pos. and neg. edges of A and B

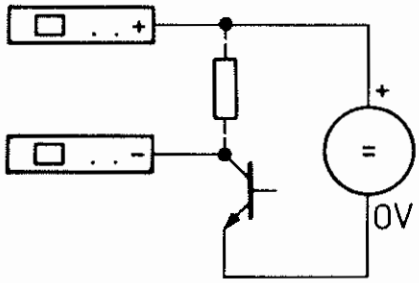


Connection diagram: Counter

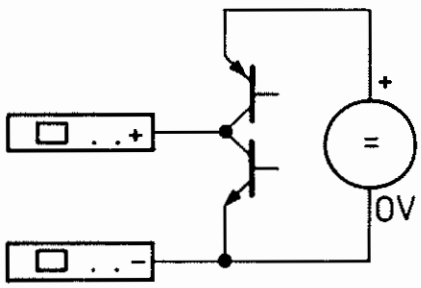


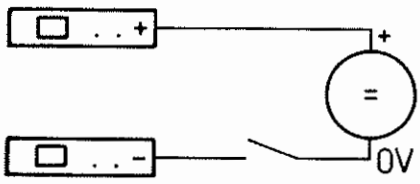
Connection modes: Encoder

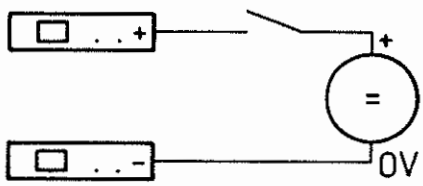
Open collector



Bipolar







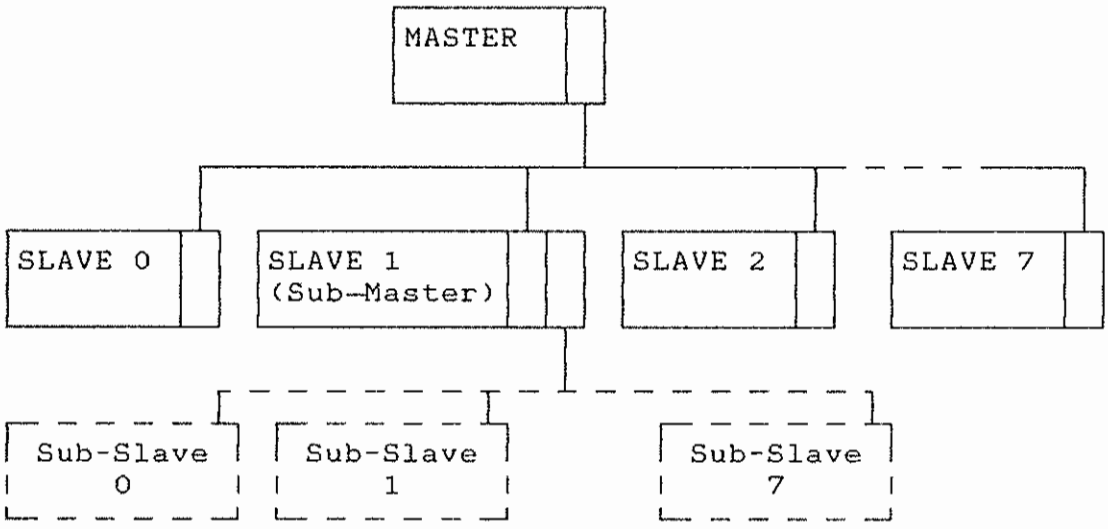
Telecommunications Controller TCC 30

The telecommunications controller TCC 30 enables an installation to be decentrally automated by linking together numerous systems:

SELECONTROL® PMC - SELECONTROL® PMC
SELECONTROL® PMC - SELEDATA® PPC

In such a network, a master can communicate with up to eight slave systems.

Each of these slaves can, in turn, act as master and communicate with a sub-network having up to eight sub-slave systems.



The hardware of the TCC 30 can be set-up to act as either a master or as a slave (0 ... 7).

Data is transfered via RS 485 interfaces with a transmission rate of 187.5 kBaud. A twisted-pair line serves as an economic transmission medium (RS 485 bus).

The maximum transmission distance is 800m (2633 ft).

Process data, i.e. the contents of data registers, are exchanged between the systems.

A maximum of 32 bytes can be transfered per slave and per direction.

Programming of the telecommunications controller, TCC 30, is described in Section 8.

Operating mode setting

A DIP code switch is mounted on the rear panel which is used for setting whether the module shall operate as a master or as a slave as well as for mode selection and the slave address.

Operation as master: Switch 1 -> OFF
Switch 2 -> ON

Mode	S.3	S.4	
0	ON	ON	1...8 slaves
1	ON	OFF	1...4 slaves
2	OFF	ON	1...2 slaves
3	OFF	OFF	1 slave

Operation as a slave: Switch 1 -> ON

Slave addr.	S.2	S.3	S.4
0	ON	ON	ON
1	ON	ON	OFF
2	ON	OFF	ON
3	ON	OFF	OFF
4	OFF	ON	ON
5	OFF	ON	OFF
6	OFF	OFF	ON
7	OFF	OFF	OFF

3.2 Input/Output Modules

DIM 30 Digital Input Module

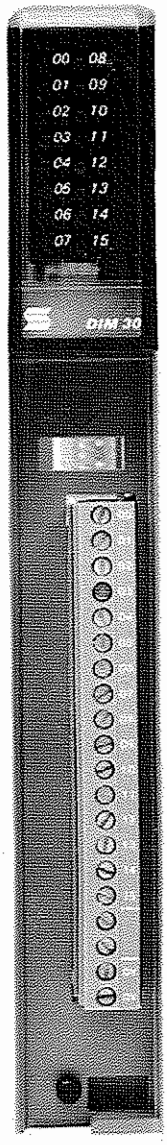
The digital input module, DIM 30, serves to connect up to 16 positive-going input signals (24V dc).

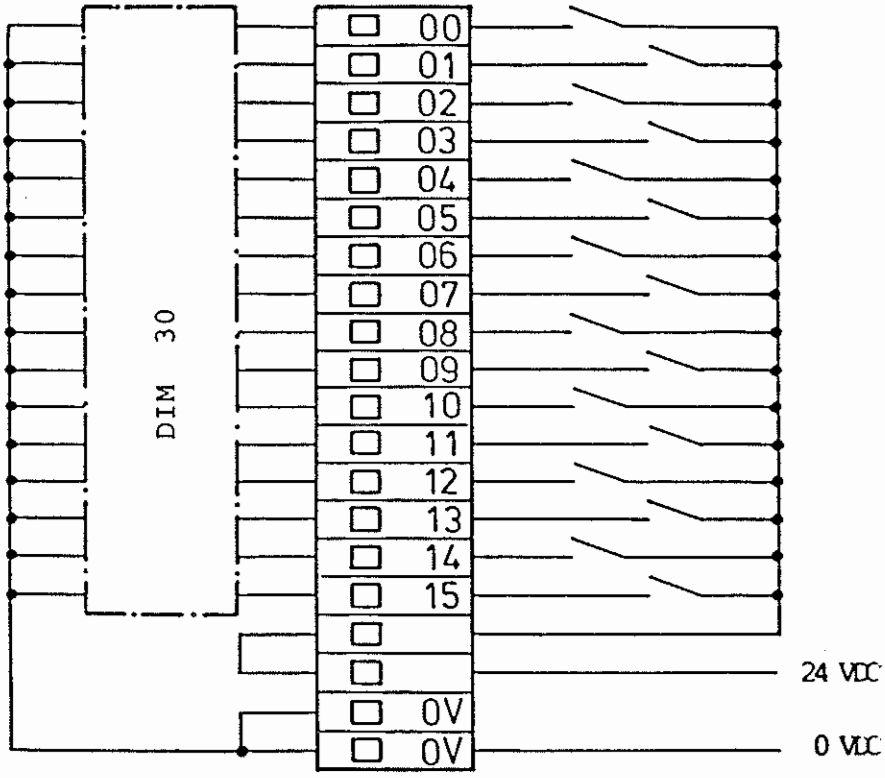
Technical specifications

Digital input module	**DIM 30**
Number of inputs	16
Electrical isolation	Yes (opto-isolators)
Input voltage - nominal value - permissible range	 +24V +18 ... +36V dc
Switching threshold - signal logical '0' - signal logical '1'	 < 5V dc > 13V dc
Input current, typ.	6mA (at 24V dc)
Input delay	< 5ms
Interface indicator	LED, green
Front panel connector	20-pole
Weight	400g (14oz)

The signal lines are connected via a 20-pole connector on the front panel.

Connection diagram inputs





Digital Input Module DIM 31

The Digital Input Module DIM 31 provides a means of connecting up to 16 110/240V ac signal sources. The module is also suitable for connecting positive or negative-going signal sources.

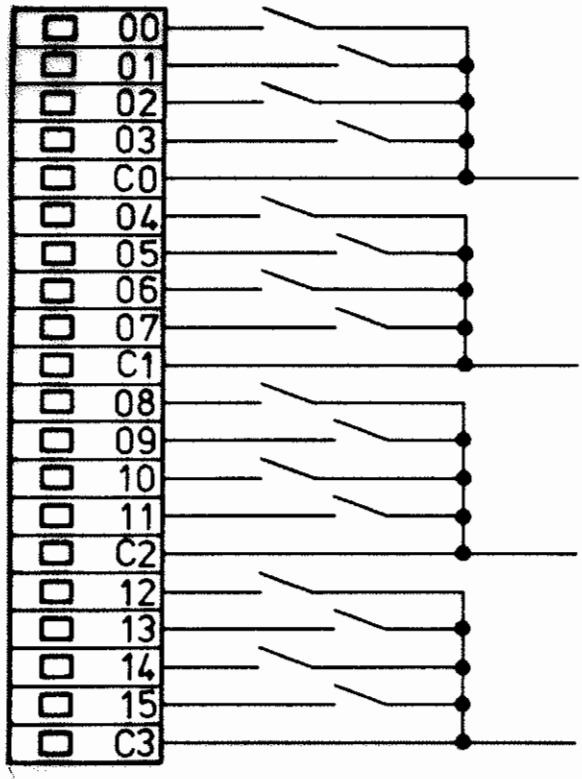
Technical data

Digital Input Module	DIM 31
Number of inputs	16
Electrical isolation	Yes (opto-isolators)
Input voltage	110...240Vac
	(+10 / -15%)
Switching thresholds:	
- Logical "0" signal	< 30Vac/dc
- Logical "1" signal	> 70Vac/dc
Input current, typ.	1.5mA (110Veff)
	3.0mA (240Veff)
Input delay	< 5ms
Interface indicator	LED, green
Front panel connector	20-pin
Weight	400 g

The 16 inputs are sub-divided into four electrically separated groups. Each group thus has four inputs with a common connection (Common C).

In this way, it is possible to connect positive-switching (PNP), negative-switching (NPN) as well as ac-switching signal sources to the same module.

Connection diagram: Inputs



Digital Input Module DIM 32

The Digital Input Module DIM 32 provides a means of connecting up to 16 12/48Vac/dc signal sources. The module is also suitable for connecting positive or negative-going signal sources.

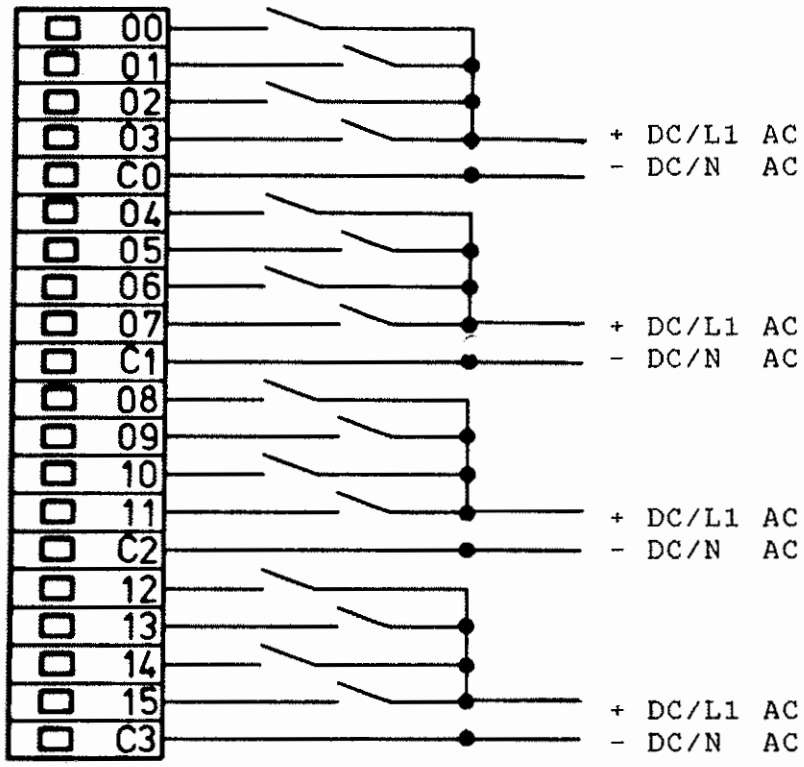
Technical data

Digital Input Module	DIM 32
Number of inputs	16
Galvanic isolation	Yes (opto-isolators)
Input voltage	12...48Vac
	(+15 / -15%)
Switching thresholds:	
- Logical "0" signal	< 2Vac/dc
- Logical "1" signal	> 5Vac/dc
Input current, max.	18mA (48Veff + 15%)
Input delay	< 5ms
Interface indicator	LED, green
Front panel connector	20-pin
Weight	400 g

The 16 inputs are sub-divided into four galvanically separated groups. Each group thus has four inputs with a common connection (Common C).

In this way, it is possible to connect positive-switching (PNP), negative-switching (NPN) as well as ac-switching signal sources to the same module.

Connection diagram: Inputs



Digital Output Module DOM 30

The digital output module, DOM 30, provides 16 positive-going (PNP) outputs rated at 24Vdc.

Technical specifications

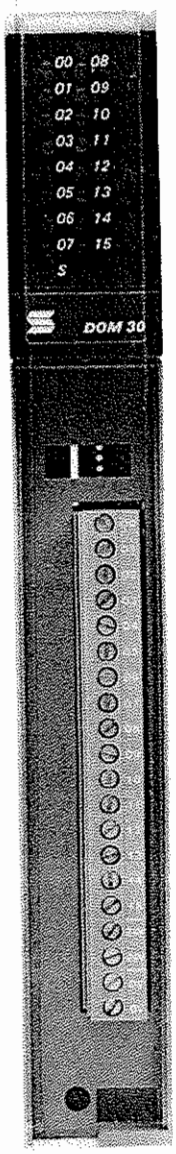
Digital Output Module	DOM 30
Number of outputs	16
Galvanic isolation	Yes (opto-isolators)
Output voltage - nominal value - permissible range	 +24Vdc +18...+36Vdc
Output current - per output, max.	 2A
Total current per module - at 20°C max. - at 55°C max.	 16A 10A
Short circuit protection	Yes (electronic)
Interface indicator	LED, yellow
Short circuit indicator	LED, red
Front panel connector	20-pin
Weight	600 g

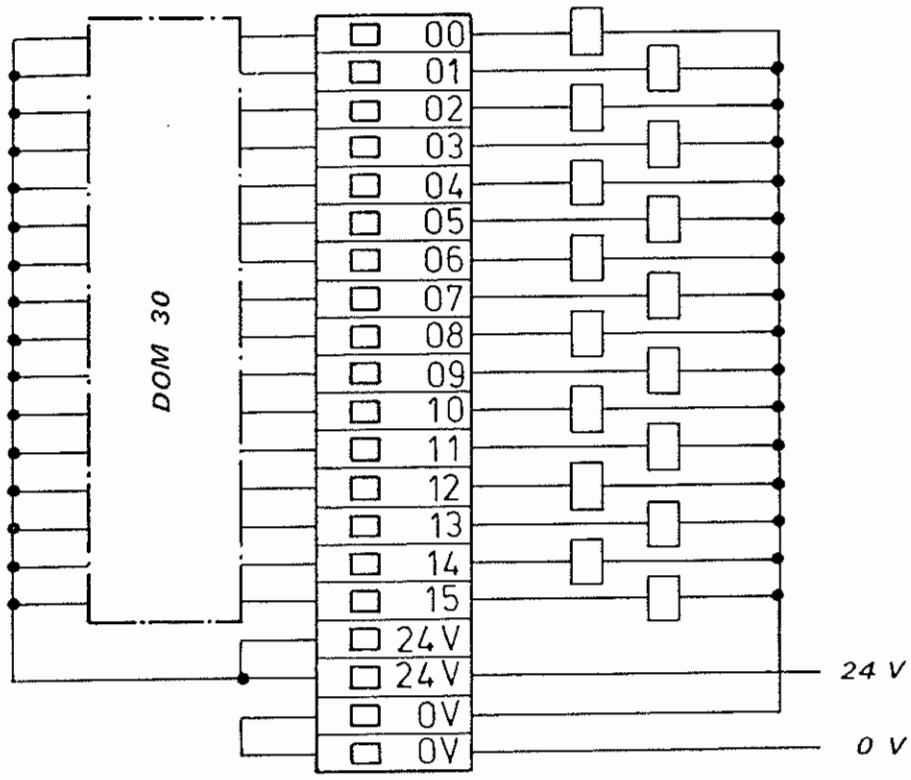
Note: In the event of a short circuit, all 16 outputs are switched off and the short circuit indicator (red LED) lights up. The 24V supply to the module has to be briefly interrupted to bring the module back into operation.

Each output is fitted with a protective diode to prevent damage to the module when working into inductive loads. In critical cases (especially with long feeder lines), it is recommended to fit an appropriate interference suppressor directly onto the load (interference source).

The output signal lines are connected via a 20-pin front panel mounted connector.

Connection diagram outputs





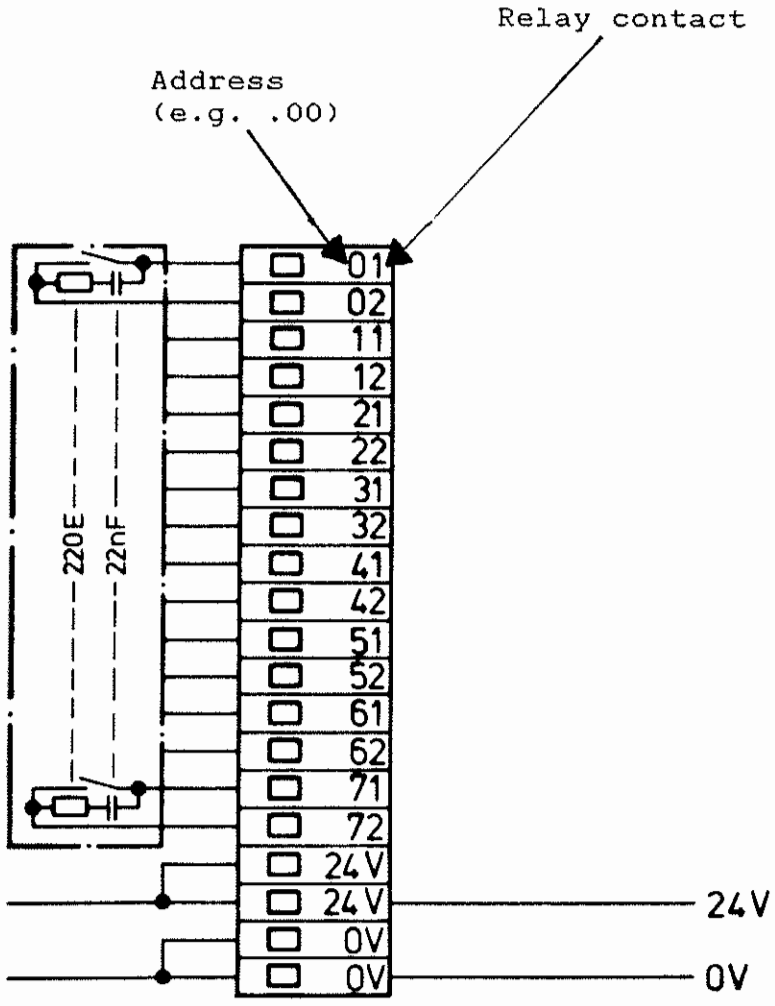
Digital Output Module DOM 31

The digital output module, DOM 31, has 8 relay outputs rated at 250V ac, 5A.

Technical specifications

Digital Output Module	DOM 31
Number of relay outputs	8
Galvanic isolation	Yes (individual channels)
Nominal voltage rating	250Vac
Output current - per output max.	5A/250Vac 250mA/220Vdc 800mA/24Vdc
Interface indicator	LED, yellow
Front panel connector	20-pin
Weight	600 g

Connection diagram outputs



Analog Input Module AIM 30

The analog input module can accept up to 8 analog input signals and convert them into digital values.

Technical specifications

Analog Input Module	AIM 30
Input configuration	Selectable per 4 channels using a range module: 0...20mA, 0...+10V, Pt 100
No. of settable channels	4 or 8
Galvanic isolation	Yes, (opto-isolators)
Meas. principle settable	Single channel or pairs of channels differentially
Resolution	11-bit (binary coded) 2047 units = nominal value
Error message (over-range)	Yes, (over 2047 units)
Conversion principle	Integrated voltage/time
Integration time	20ms (optimum interference voltage suppression)
Conversion time - 4 input channels - 8 input channels	50ms/channel 0.2s 0.4s
Input impedance: +20mA +10V Pt 100	25Ω 50kΩ 10MΩ
Basic error limit Absolute error limit (0°C...55°C)	0.2% 0.5%
Connection mode	2-wire/channel
Integrated current source	2mA
Supply voltage	+24Vdc, +18...+36Vdc)
Front panel connector	20-pin
Weight	450 g

Range modules

Various range modules are available to complement the analog input module:

AID 30	0...20mA	(4 channels)
AID 31	0...+10V	(4 channels)
AID 32	0...500mV (Pt 100)	(2 channels)

A 2mA current source is incorporated in the module for driving up to four Pt 100 devices.

Each range module contains the input circuitry for four single channels. This provides the opportunity to configure the eight channels of the analog input module in two different groups of four channels each, e.g.:

4 channels 0...20mA plus
+4 channels 0...10V

Operating modes and measurement techniques

The module can be operated in two different modes:

A: **Automatic multi-channel operation**

B: **Single channel/multi-channel switchable by software**
This operating mode can be useful for special time-critical applications.

Note:
Programming and the processing of the data are fully desribed in **Section 7, analog value processing.**

The two following **measurement techniques** can be employed:

Single: The value is taken relative to ground (GND) (max. 8 single channels)

Differential: The difference between two adjacent channels is measured (max. 4 differential channels)

The module also has the possibility of being switched from 8-channel to 4-channel operation. In this way, the conversion time can be shortened if only four channels are in use.

Setting-up

There is a DIP switch on the circuit board of the module for setting it up:

Switch 1	ON* OFF	- Automatic multi-channel operation - Single/multi-channel operation with switch-over under software control
Switch 2	ON* OFF	- Single - Differential
Switch 3	ON* OFF	- 8-channel single / 4-channel diff. - 4-channel single / 2-channel diff.
Switch 4		Not used

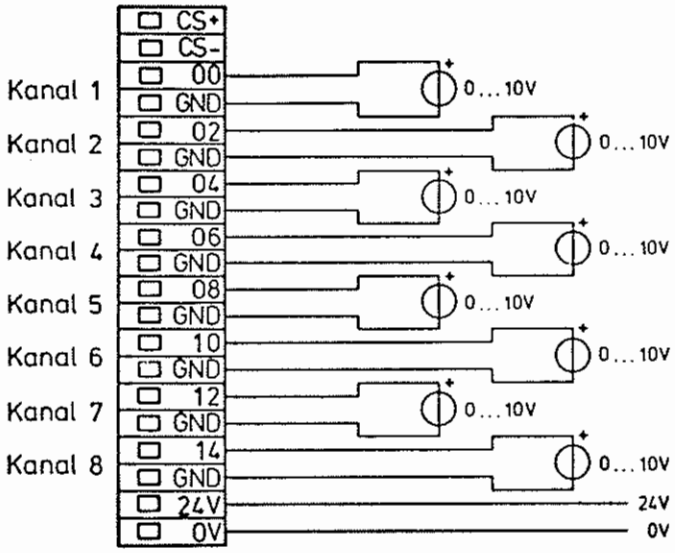
* Factory setting

The conversion times are:

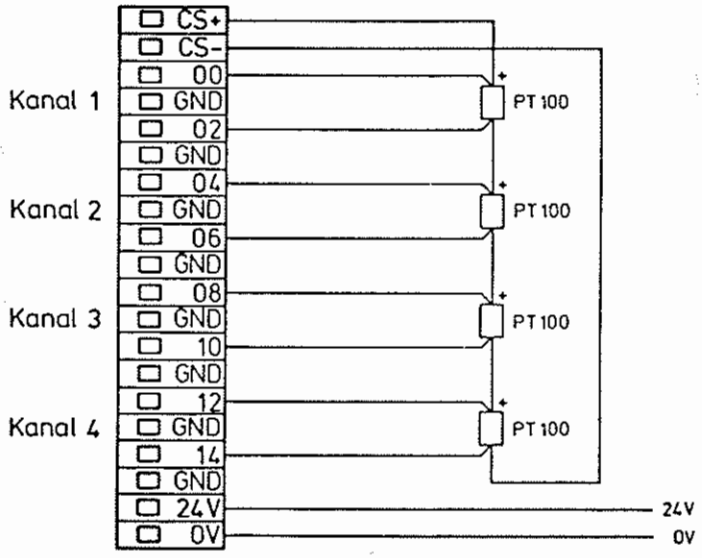
- Multi-channel operation: 8 ch. single (4 ch. diff.) 0.4s
  4 ch. single (2 ch. diff.) 0.2s
- Single channel operation: 0.05s

Connection diagram

Example: 8 channels single / 0...10V

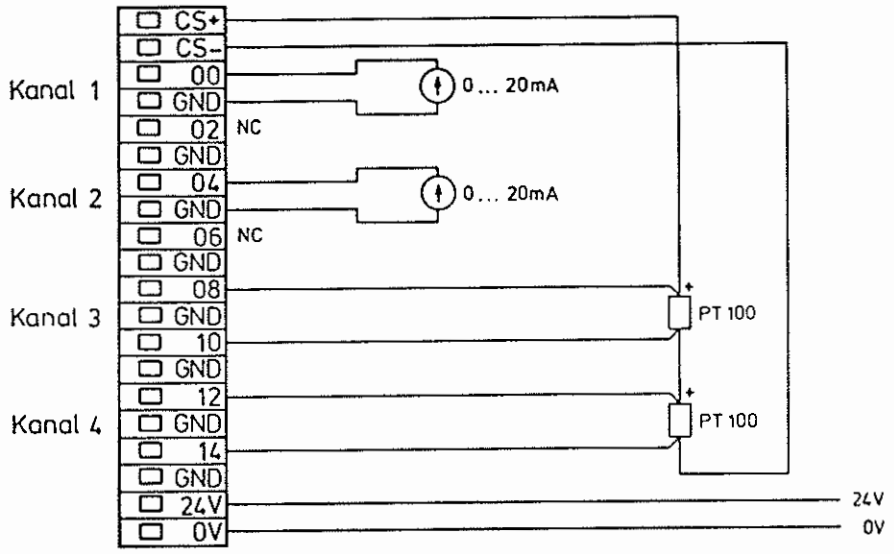


Example: 4 x Pt 100 / 4 channels differential



Important: Current sources are **always** wired to ground (GND) even for differential measurements. The adjacent channel is not connected for differential measurements.

Example: 2 channels 0...20mA
2 channels 0...500mV (Pt 100
(4 channels differential)



NC: No Connection

Analog Input Module AIM 31

Technical specifications

Analog Input Module	AIM 31
Input configuration	Selectable per 4 channels using a range module: -20...+20mA, -10...+10V
No. of settable channels	4 or 8
Galvanic isolation	Yes, (opto-isolators)
Meas. principle settable	Single channel or pairs of channels differentially
Resolution	11-bit (binary coded) 2047 units = nominal value
Error message (over-range)	Yes, (over 2047 units)
Conversion principle	Integrated voltage/time
Integration time	20ms (optimum interference voltage suppression)
Conversion time - 4 input channels - 8 input channels	50ms/channel 0.2s 0.4s
Basic error limit Absolute error limit (0°C...55°C)	0.2% 0.5%
Connection mode	2-wire/channel
Integrated current source	2mA
Supply voltage	+24Vdc, +18...+36Vdc
Front panel connector	20-pin
Weight	450 g

Range modules:

AID 30	-20...+20mA	(4 channels)
AID 31	-10...+10V	(4 channels)

For setting-up information, see AIM 30.

Analog Output Modules AOM 30/31

The analog output modules convert digital information into four analog values.

Output levels:

AOM 30: 0...+10V
(0...+20mA with range module AOD 30)

AOM 31: -10...+10V

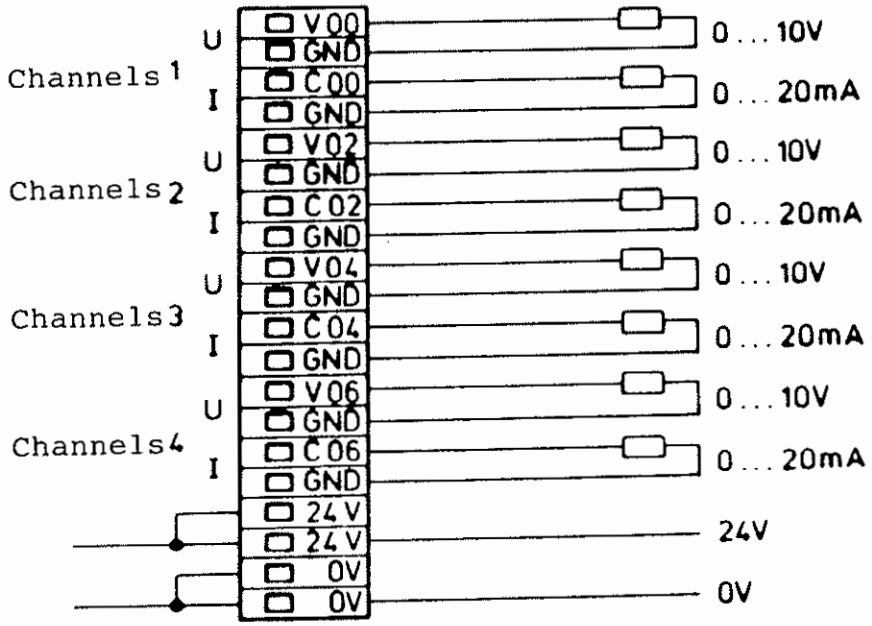
Technical specifications

Analog output module	AOM 30 / 31
Output levels: AOM 30	0...10V (RLmin = 2k) 0...20mA (RLmax = 250E)
AOM 31	-10...+10V
Number of channels	4
Electrical isolation	Yes, (opto-isolators)
Resolution	12-bit (binary coded) 4095 units = nominal value
Conversion principle	R-2R-network
Conversion time	< 50µs
Basic error limit	0.3 %
Absolute error limit (0°C...55°C)	0.5 %
Supply required	+24Vdc
Front panel connector	20-pin

For each pair of voltage outputs, two proportional current outputs (0...20mA, AOM 30) can be produced additionally by use of the AOD 30 range module.

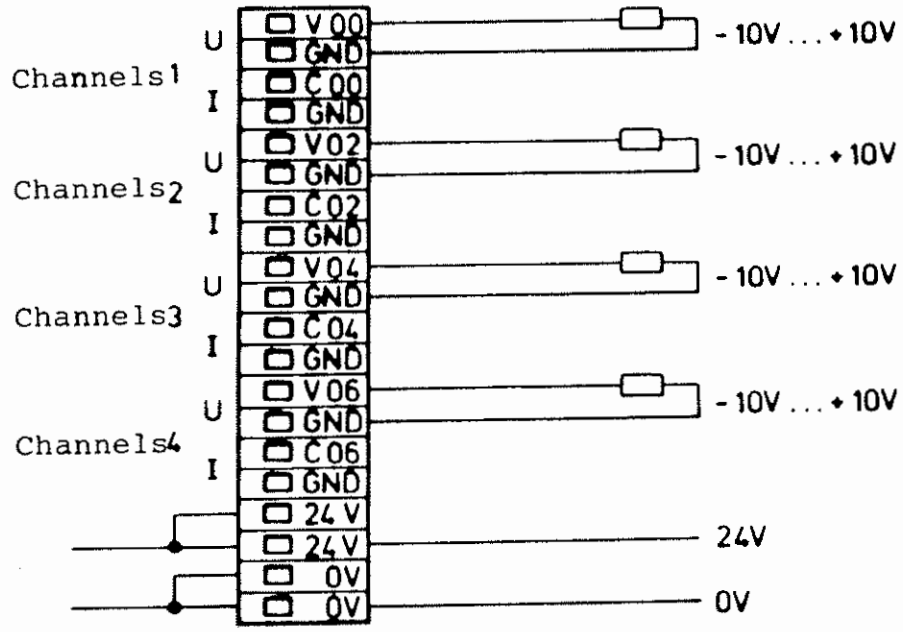
Connection diagram

4 Channels 0...10V
4 Channels 0...20mA (with range modules AOD 30)



Connection diagram

4 Channels -10...+10V



3.3 Special Modules

ATM 30 Analog Timer Module

The analog timer module, ATM 30, contains 8 settable analog timers.

Technical specifications

Analog timer module	**ATM 30**
Number of timers	8
Settable time range	30ms ... 10 hours
Settable sub-ranges	7 (DIP switches)
Fine adjustment	Front-mounted potentiometers
External potentiometers	4 max. (as required)
Supply voltage	Internal
Status indicator	LED, green
Front panel connector	5-pin
Weight	400g (14oz)

The required sub-range can be set-up on the circuit board by means of the DIP switches 1 .. 3 inside the module.

Time range setting:

Time range	Switch 1	Switch 2	Switch 3
30...150ms*	ON*	ON*	OFF*
0.1...1.0s	OFF	OFF	OFF
1.0...10s	OFF	OFF	ON
0.1...1.0min	ON	OFF	OFF
1.0...10min	ON	OFF	ON
0.1...1.0hr	OFF	ON	OFF
1.0...10hr	OFF	ON	ON

* Factory setting.

The potentiometers for fine time adjustments are accessible at the front panel.

The timers, T4 ... T7, can be set internally or externally as required. The choice is made by means of DIP switch 4.

Selection: internal/external time setting

Time setting	Switch 4
Internal*	ON
External	OFF

* Factory setting.

The timers are started under software control and their statuses can be interrogated. They run as long as the start condition remains in force or until the set time has elapsed.

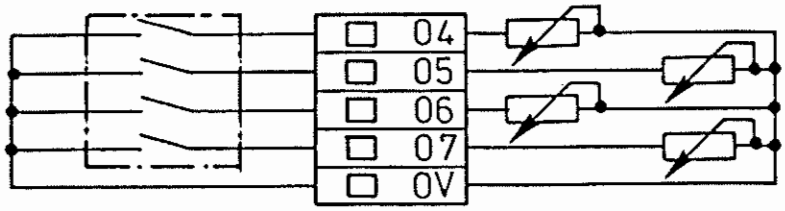
The **LED indicators** have the following meanings:

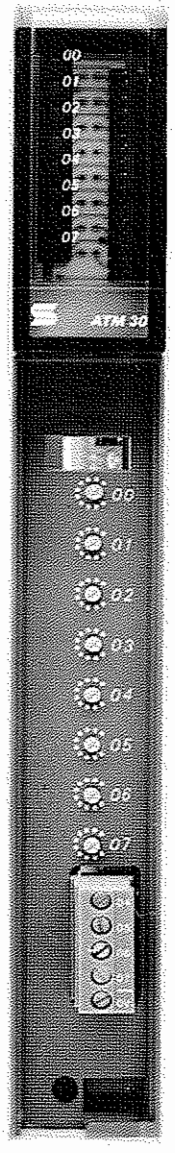
LED extinguished	Timer set but not started
LED blinking	Timer running
LED lighted	Time has elapsed and start condition is active or timer is not programmed

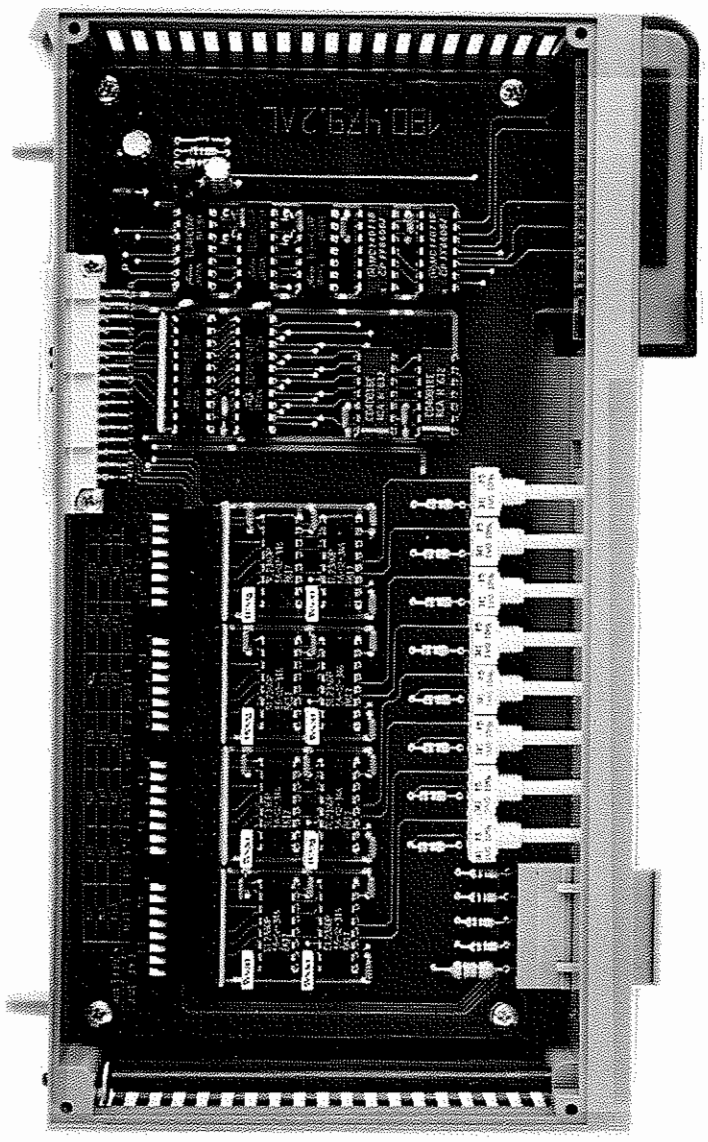
The potentiometer set, ATP 30, must be ordered separately.

Connection diagram
external potentiometer

- Potentiometer 1MΩ
  linear
- Use screened cable
  (max. length 5m approx.)
  Connect the screen only to
  the GND of SELECONTROL PMC
- Set internal potentiometer
  to minimum setting





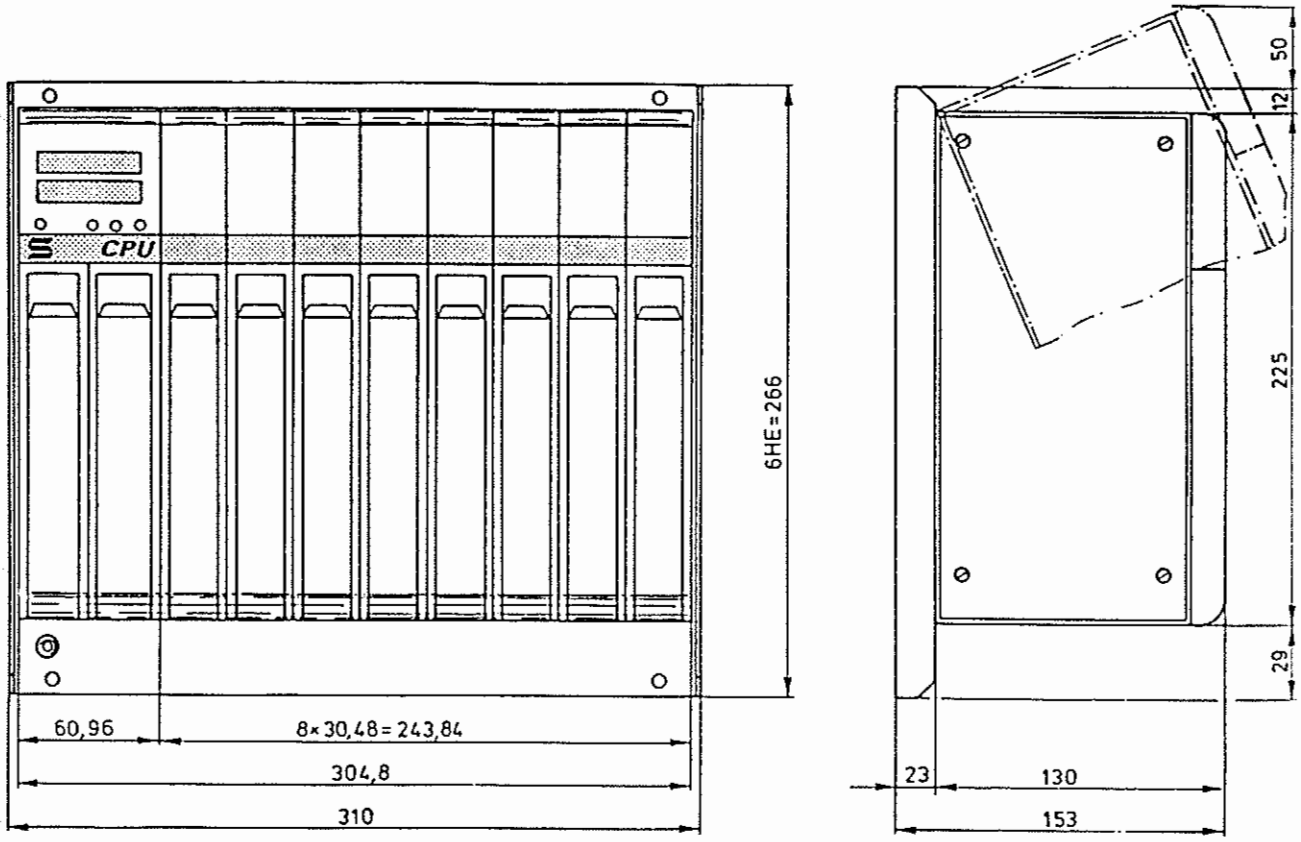


3.4 Accessories

BPU 08 Back-panel Unit for 8 modules

The back-panel unit, BPU 08, has slots for one CPU central processor unit and eight further modules.
(Maximum 128 inputs/outputs).

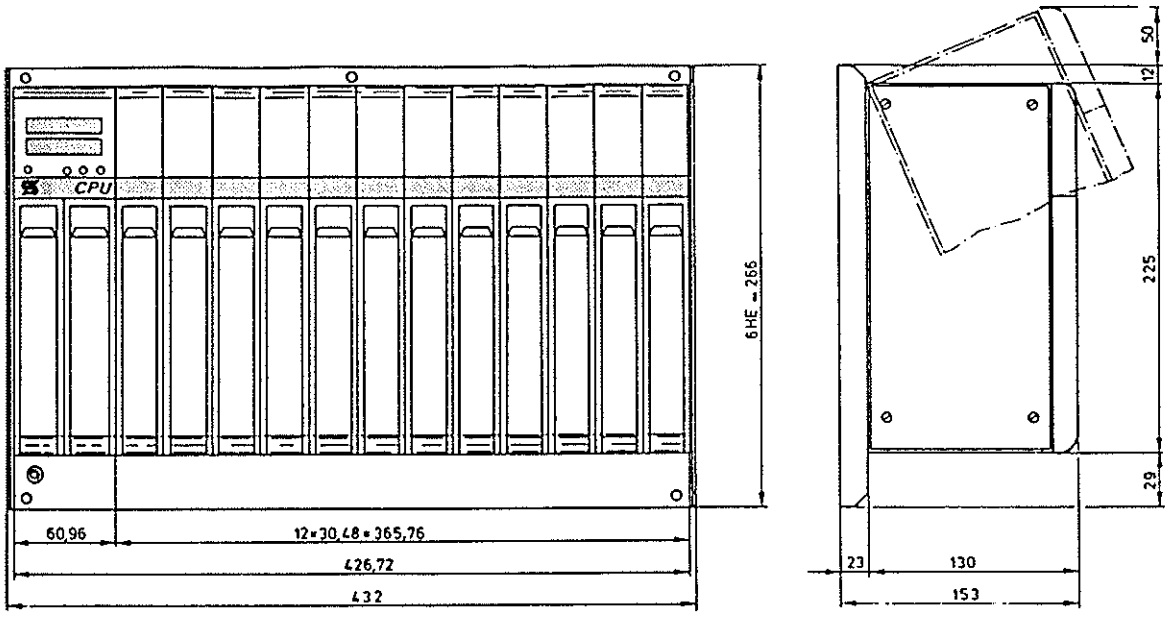
Dimensions:



BPU 12 Back-panel Unit for 12 modules

The back-panel unit, BPU 12, has slots for one CPU central processor unit and a further 12 modules.
(Maximum 192 inputs/outputs).

Dimensions:



The back-panel unit, BPU 12, can be mounted in a 19" rack.

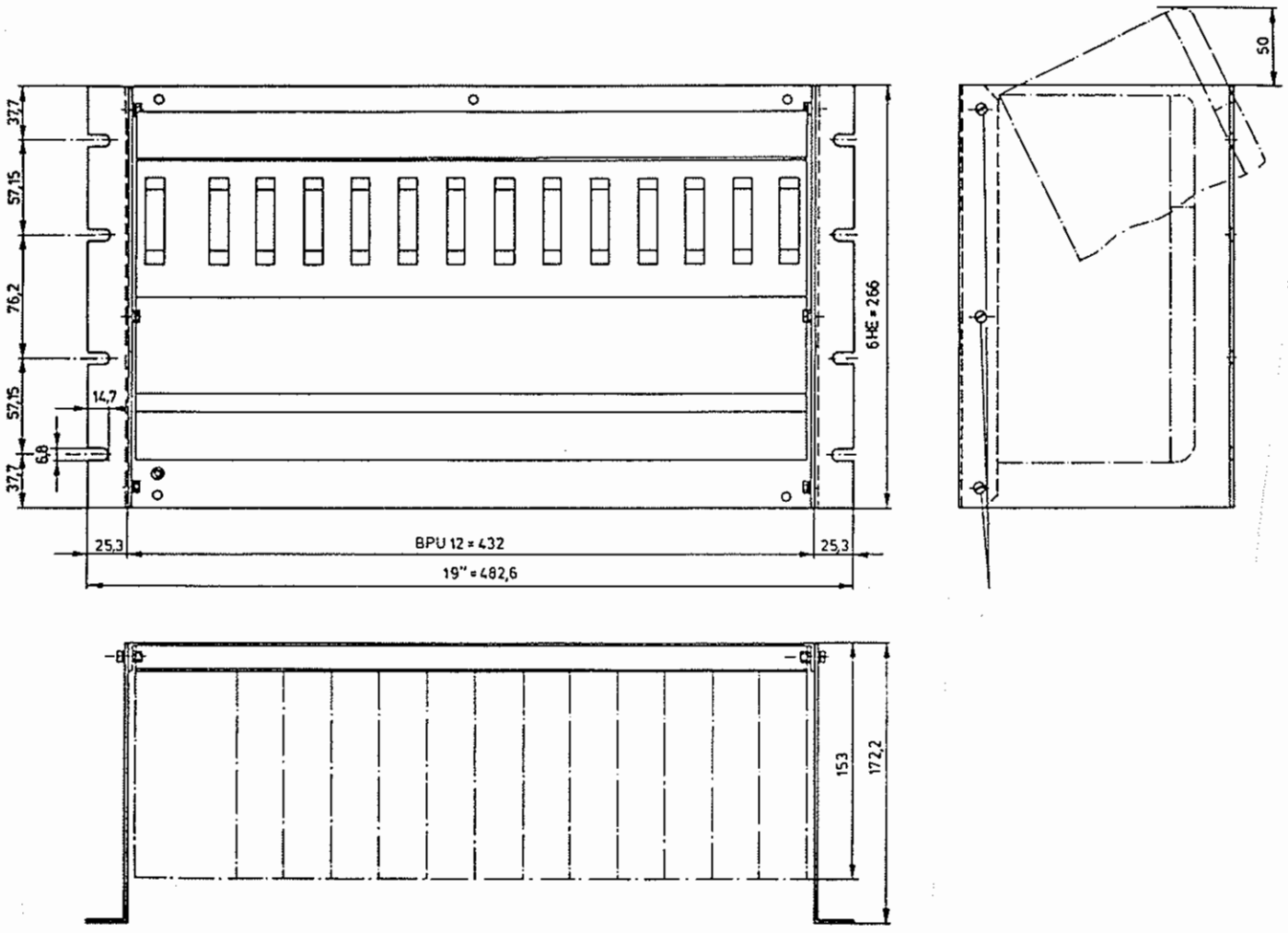
The accessory kit, RMA 12, is necessary for rack-mounting.

RMA 12

The rack-mounting accessory kit, RMA 12, contains all that is needed to mount a BPU 12 in a 19" rack.

The installation depth can be adjusted as required.

Dimensions:



EPU 08 Expansion Unit for 8 modules

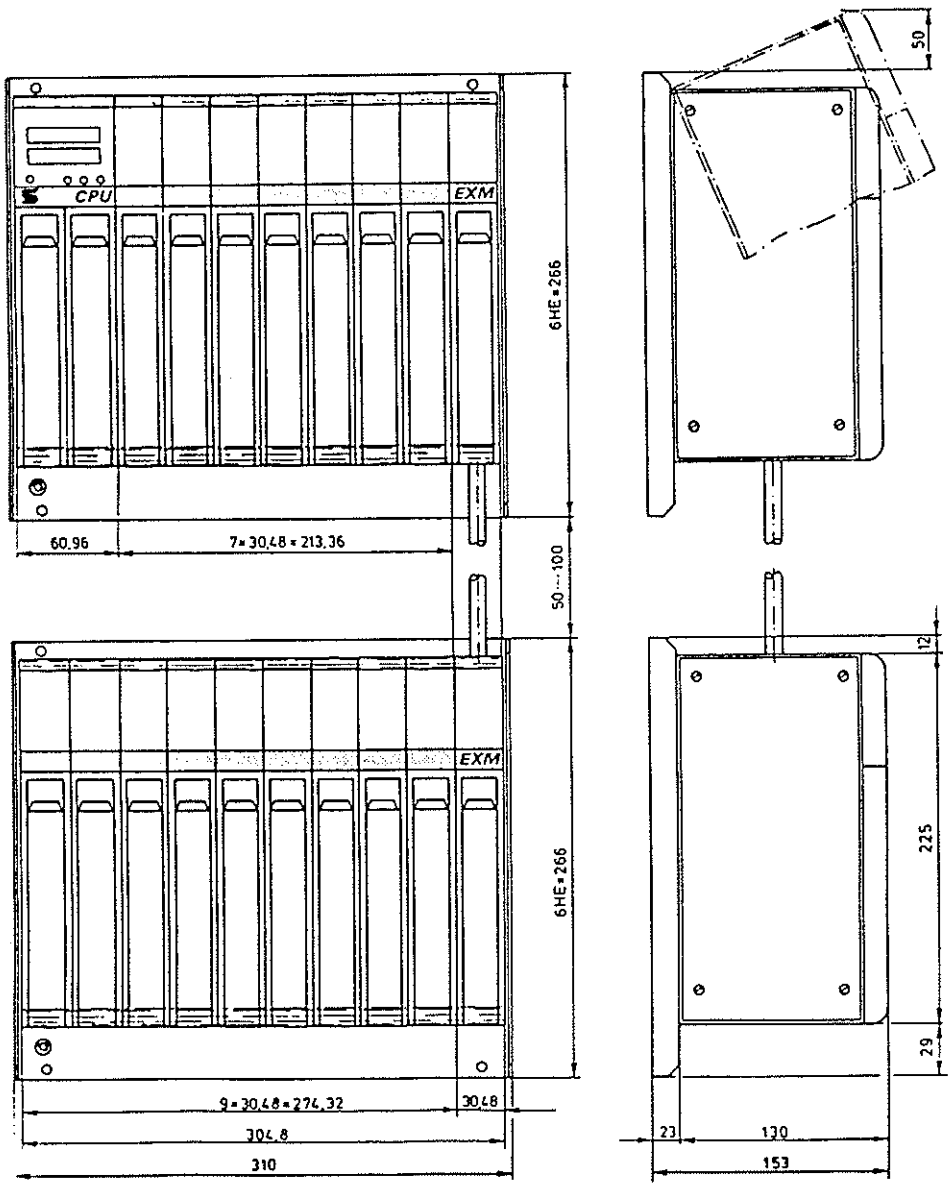
The expansion unit, EPU 08, is only ever used in conjunction with the back-panel unit BPU 08.

A system can be expanded to 16 modules (256 I/O's) in this way.

The expansion module, EXM 30, is needed to provide the link.

The module address, 07, moves from the back-panel unit BPU 08 to the expansion unit EPU 08

Dimensions:



EXM 30 Expansion Module

The expansion module is used to link the bus through from the back-panel unit to the expansion unit, EPU 08.

The expansion module must be inserted in the extreme right-hand slot of both the back-panel unit BPU 08 and the expansion unit EPU 08

The bus linking cable is doubly screened for security against interference.

BSM 30 Blind Space Module

The BSM 30 module is empty and is intended for filling empty module slots.

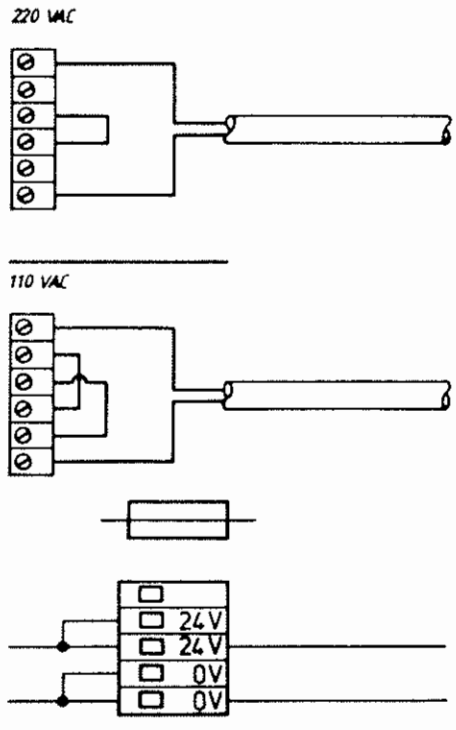
PSM 30 Power Supply Module

The power supply module, PSM 30 (engage 1 module slot)provides the 24V supply for the central processor unit, CPU 30, and also simulates the in- puts and outputs for the program test.

Technical specifications:

Power supply module	PSM 30
Input voltage	110Vac or 220Vac
Output voltage	24Vdc
Output current	1A
Connections - Mains: 110/220Vac - 24Vdc	 Internal terminal strip Front panel 5-pin connector
Weight	1000 g

Connection diagram



SIM 30 Simulator

The 16 inputs of a module can be simulated with the simulator, SIM 30.

The simulator is plugged directly onto the module in place of the connector terminal strip.

The connector terminal strip can also be plugged onto the simulator. In this case, the inputs (signal sources/simulator switches) are connected in parallel i.e. linked together. This configuration is mostly used during commissioning of an installation where individual signal sources have to be simulated.

MAL 30 Module Address List

These are module identification strips with space provided to write-in an identification of the inputs and outputs. The strips can be attached to the inside or the outside of the terminal strip cover as appropriate.

LOC	
MOD	
00	
01	
02	
03	
04	
05	
06	
07	
08	
09	
10	
11	
12	
13	
14	
15	
24V	
24V	
0V	
0V	

LOC: module slot
MOD: module type

Data cable CCA

The cable is used to transfer data between the programming equipment and the central processor unit.

The programming devices have differing connectors and, therefore, the following selection of cables is available:

CCA 30: Cable with 9-pin socket (female)
(for connecting e.g. an IBM-PC or SELECONTROL® PSU)

CCA 31: Cable with 9-pin plug (male)
(for connecting e.g. to a Microscribe terminal)

CCA 32: Cable with 25-pin socket (female)
(for connecting e.g. a Commodore PC 10 or PC 20)

CCA 33: Cable with 25-pin plug (male)
(for connecting e.g. an NEC terminal)

The cables have a length of 1.5m (5 feet)

RS 232C programming interface on the CPU
Pin assignment in the 9-pin D-type plug:

Pin	Signal
2	$\overline{\text{TXD}}$
3	$\overline{\text{RXD}}$
5	CTS
7	GND (0V)

4 PROGRAMMING SYSTEMS

4.1 Standard terminals

Because of its refined firmware, the SELECONTROL² PMC can be readily programmed with commercially available, non-intelligent terminals.

Such a terminal must comply with the following requiements:

- ASCII character set
- RS 232C interface port (V24)
- Display at least 2 lines of 20 characters

Hand-held, home and personal computers can also be used as a programming device.

In hand-held computers (e.g the NEC PC 8201A), a communications program is already included in the standard software and only needs to be initiated.

Home computers and personal computers have to transfer the ASCII characters via an RS 232C (V24) interface and need to have an appropriate communications program installed in them.

Communications programs are available on diskettes for the following systems:

CCP 30: for IBM-PC and compatibles

CCP 31: for Commodore VC/SX 64

The communications programs, CCP 30 for the IBM-PC and compatibles) and CCP 31 for Commodore VC/SX 64) are started as thus:

1) Place diskette in drive A

2) Start the program by typing "TERMINAL" and pressing "ENTER"

Always clear the memory in your computer before loading the CCP 30 program

The COMDRIVE file will only be loaded if it has not been loaded previously in another program.

Data transmission to and from the controller is not possible if the correct COMDRIVE file has not been loaded from the CCP.

The NEC PC-8201A hand-held computer is put into operation and initialized as follows:

NEC PC-8201A

1) Put the "BACK-UP POWER" switch on the PC underside to "ON"

2) Connect the RS 232C interfaces on the SELECONTROL® PMC and the PC-8201A together with data cable SELECONTROL CCA 33.

3) Switch the power on for the
   - SELECONTROL® PMC (24V dc)
   - NEC PC-8201A
     (8.5V from a mains adapter or 6V from a battery)
     Main switch on the right side of the terminal to "ON"

4) Set the contrast on the PC-8201A
   The choice of operating modes is displayed on the screen:
   "BASIC" "TEXT" "TELCOM"

5) Place the cursor on "TELCOM", press the "->" key and then "ENTER"

6) Press the "F4" key. The display shows: **Telcom:Stat**

7) Input the data transmission format.
   The data transmission format must agree with that set on the central processor unit, CPU 30.
   The transmission rate must be set to **300 Baud.**
   SELECONTROL® PMC S1: OFF, S2 ON
   S3: OFF (data transmission format)
   Input at the terminal: 3E71NN
   Display shows: Telcom: **Stat 3E71NN.** Press "ENTER"

8) Press **"F5"**. Display shows: **Term**

9) Press **"F2"**. **"FULL"** is displayed

10) Press **"F3"** until **"Echo"** disappears

11) Press "ENTER"

The terminal is now ready for programming.

The operating mode switch (key-switch) on the SELECONTROL® PMC must be in the "PROGRAM" position before programming begins.

4.2 SELECONTROL® PSU
Programming and Service Terminal

General

The SELECONTROL® PSU is an ASCII terminal with dust and splash proof keyboard which makes it especially suitable for industrial applications. It is used for programming, data input and text display and can be built into front panels or operating consoles (IP 65 for front panel mounting). Communication with the controller is via an RS 232C interface. The LC-display can display up to 4 lines with up to 40 characters each with a character height of 5 mm.

The SELECONTROL® PSU is available in different versions:

SELECONTROL® PSU 30
ASCII terminal for off-line programming with 1 RS 232C interface.

SELECONTROL® PSU 31
ASCII terminal for programming in either on-line or off-line mode with 1 RS 232C interface.

SELECONTROL® PSU 31P
ASCII terminal for programming in either on-line or off-line mode with 1 RS 232C interface and 1 parallel printer interface (CENTRONICS).

Operating instructions see manual

4.3 SELECONTROL® CAP 3000
(Computer Aided Programming)

1. Introduction

The highly refined SELECONTROL® CAP 3000 software package enables very convenient programming and documentation preparation on IBM-PC's or compatible systems (running with MS-DOS operating system).

Operating instructions see manual

5 OPERATING INSTRUCTION SET

The central processor unit, CPU, contains the necessary firmware for programming via standard terminals.

The operating command set of the SELECONTROL® PMC (comparable to the MS-DOS operating system of a PC) contains the following handy functions:

In the "PROGRAM" mode

(Key switch in the position PROGRAM)

In the 'basic state', the free program lines can be written to directly or they can be inserted (INSERT command).

Help menu (INFORMATION)	I)
Set the current program line (GOTO)	G)
Display step-wise	
Clear program lines (DELETE)	D)
Display program lines (LIST)	L)
Print out the program (PRINT)	P)
Search for program lines (FIND)	F)
Fill the look-up table (TABLE)	T)
Display the program header (HEADER)	H)
Clear program (NEW) with security code N)1234	N)
Copy program from RAM to EPROM (SAVE) with security code S)1234	S)
Copy program from EPROM to RAM (RESTORE) with security code R)1234	R)
Select "COLD START" after power up	C)
Select "WARM START" after power up	W)
Define a personal code-word for program protection (USERIDENT)	U)

In the "RUN-MONITOR" mode
(Key switch in the position RUN-MONITOR)

Display the process data (VIEW) V)

Modify the process data (MODIFY) M)

Programming is carried out 'on-line' (programming terminal connected to the CPU) in the echo mode, i.e.

- The control commands are input via the programming terminal (unformatted).
- After pressing >ENTER<, the central processor unit checks the command for syntax errors.
- If the input is correct, the control command is formatted and returned to the programming terminal via the RS 232 (V24) interface.
- The input line is over-written with the formatted instruction and the next line number is displayed.

An **acoustic error warning** is given if the input is incomplete or not permissible. The same line number is again shown so that the input can be renewed. The incorrect input is shown one line higher in the unformatted form.

The SELECONTROL PMC firmware supports the following functions for **correcting the inputs:**

"BACK SPACE" : Moves the cursor to the left. The erroneous input can be over-written or deleted.

"DELETE" : The character to the left of the cursor is deleted.

5.1 Operating command set in the "PROGRAM" mode

Help menu (INFORMATION)

Operating command: **I)**

The operating command I) brings up a help menu on the screen.

```
Line No.
XXXX  I)       Operating command I)
```

After striking >ENTER<, all the operations that are permissible for programming the SELECONTROL® PMC are displayed.

Additionally, it is also possible to display which operands (identification + parameter) can be used in conjunction with a particular operation.

```
Line No.
XXXX  I) L     Operating command I)
               followed by an operation e.g. "L" (load)
```

After striking >ENTER<, all the operands (identification + parameter) are listed which can be used in conjunction with the selected operation.

The I) operating command makes the programming of the SELECONTROL PMC easier for you

Writing to free program lines

An input is always assigned to the program line (line number) shown by the cursor.

All spaces and all leading zeros within a control instruction can be omitted (with the exception of leading zeros for hex constants)

The complete control instruction is over-written with a formatted version including the spaces and leading zero after the >ENTER< key has been struck.

The system then brings up the next line number so that programming can be continued.

```
Line No. Instruction

0000     LI 0.0          Input without spaces and leading zeros

after >ENTER<

0000     L  I 00.00      Formatted input
0001                     Next line number
```

An acoustic warning is given if the input is erroneous and the control instruction is not formatted. The system comes back with the same line no. so that the instruction may be corrected.

```
Line No. Instruction


0000     A 0.1           Erroneous input
                         (no operand identification "I" INPUT)

after >ENTER<

0000     A 0.1           Acoustic warning
0000                     Same line number for renewed input
```

Now input the following program:

```
Line No. Instruction    Remark

0000     L   I 00.00    If input 00.00
0001     A   I 00.01    And input 00.01
0002     O   I 00.03    Or input 00.03
0003     =   O 05.00    Then output 05.00
0004     EP              End of program
0005
```

Set new line number (GOTO)

Operating command: G)

By using the GOTO command, a jump can be made to any arbitrary line number within the program.

```
Line No.  Command

0005      G)2              Command G) with new line number

after G) and >ENTER<

0002      O   I 00.03      Previous instruction
0002                       Line number for new input
                           (automatic INSERT)
```

If a line number is selected that is outside the current program, the first free line number will be automatically be shown instead.

```
Line No.  Command

0002      G)1987           Command G) with new line number

after G)1987 and >ENTER<

0005
```

If the program is to be structured by use of spaces (NOP), then these spaces must be specifically input.

It is not necessary to insert many spaces (NOP) since control instructions can be subsequently added at any time (the NOP's lengthen the cycle time).

Insertion of control instructions (INSERT)

An **automatic INSERT** is executed if a GOTO command directs a new instruction to a line number at which there is already an instruction.

The new instruction is written at the specified line number and the next and all following instructions are moved down by one line right through to the end of the program.

```
Line No.  Instruction

0000      L   I 00.00    Previous program
0001      A   I 00.01
0002      O   I 00.03
0003      =   O 05.00
0004      EP
0005

after G)

0002      O   I 00.03    Previous instruction at line 0002
0002      A I 0.2        New instruction

after >ENTER<

0002      A   I 00.02    New program
0003      O   I 00.03
0003
```

Display step-wise

The successive line number and the corresponding instruction are displayed each time the **>ENTER<** key is pressed.

A step-by-step listing is thus possible.

```
Line No. Instruction

0003      O    I 00.03
0003

after >ENTER<

0003      O    I 00.03
0004      =    O 05.00
0004

after >ENTER<

0003      O    I 00.03
0004      =    O 05.00
0005      EP
0005
```

Clearing instructions (DELETE)

Operating command: D)

This command deletes the current instruction and moves all the other instructions up right through to the program end.

```
Line No.  Instruction

0000      L   I 00.00
0001      A   I 00.01
0002      A   I 00.02
0003      O   I 00.03
0004      =   O 05.00
0005      EP

after G)2

0002      A   I 00.02      Previous instruction
0002      D)               DELETE

after >ENTER<

0002      O   I 00.03
0002
```

> Control instructions cannot be over-written. An instruction must be consciously DELETED (otherwise INSERT always has priority).

Display the instruction list (LIST)

Operating command: **L)**

The instruction list is shown from (and including) the current line number.

The listing procedure can be interrupted and then continued by striking any key. Pressing >ENTER< terminates the procedure.

```
0000      L)                Operating command L)

after >ENTER<

Line No.  Instruction

0000      L    I 00.00      Program
0001      A    I 00.01
0002      O    I 00.03
0003      =    O 05.00
0004      EP                Last instruction

END OF LISTING              System message

0005                        First free line number
```

Print-out of the instruction list (PRINT)

Operating command: **P)**

The instruction list is printed out from (and including) the current line number.

A "Program header" with page numbering is also output.

The print-out is shifted towards the center of the page to achieve a better arrangement. The print-out can be interrupted by striking any key and then be restarted by striking any other key.

The procedure can be terminated by striking >ENTER<.

Search for control instructions (FIND)

Operating command: **F)**

The search begins at (and includes) the current line number where the P) command occurs and continues to the end of the program.

A search can be made for the following types of data:

F) 00.03	Search for parameter
F) I 00.03	Search for an operand (data type + parameter)
F) O I 00.03	Search for a complete instruction

Line No.

0000 L I 00.00	Start of search procedure
0000 F)0.3	Command F) and parameter sought

after >ENTER<

FIND *** * 00.03	Search for a parameter
0002 O I 00.03	First line number with parameter sought
FIND NEXT? "Y"	Entering "Y" continues the search
	Entering any other character terminates the search
FIND XXX X 00.03 STRING NOT FOUND	The sought for instruction has not been found up to the end of the program. The cursor is at the program line at which the search was started.

Filling the look-up table (TABLE)

Operating command: T)

The look-up table is used for code changing. (See Section on "Programming": Operation LKP).

The operating command, T), is used to fill the look-up table.

Line No.

0003	T) nn.mm	Entry in the line nn
TABLE	nn.mm	(00 ... FF) in the table
0003		
		Value mm (00 ... FF)

No line number is occupied in the program by an entry in the table.

Display of the program header (HEADER)

The system set-up can be shown with the operating command H).

Line No.

XXXX H) Operating command H)

after >ENTER<, the following system settings are shown:

1) LKP table (code changing table)

2) System setting
   - warm/cold start
   - data transmission Baud rate and format

The HEADER output can be interrupted and restarted by striking any key. Pressing the >ENTER< key terminates the output.

Clear the program (NEW)

Operating command: N)1234.

The PMC program can be cleared by using the operating command N) followed by the security code 1234

Line No.

XXXX N)1234 Operating command N) + security code 1234

after >ENTER<, the system reports:

PROGRAM CLEARED

0000 Line No. 0000

The operating command N)1234 has the effect of completely re-initializung the user-program memory, i.e.:

- clearing the user program
- clearing the look-up table
- setting the system to cold start

Copy the program from:
RAM to EPROM memory (SAVE)
EPROM to RAM memory (RESTORE)

Operating command: **S)1234** (copy from RAM to EPROM)

Operating command: **R)1234** (copy from EPROM to RAM)

The RAM board must be plugged into the central processor unit, CPU, for these commands. The EPROM board is plugged onto the RAM board.

The operating command S)1234 starts the copying procedure from RAM to EPROM (SAVE)

The operating command R)1234 starts the copying procedure from EPROM to RAM (RESTORE)

The RESTORE command also enables a program to be copied from RAM to RAM whereby the program in the outermost RAM board is loaded into the CPU.

The whole program is always copied right up to the EP including the look-up table

Operating mode selection
Cold start / Warm start

Operating command: C) – Cold start

Operating command: W) – Warm start

All the process data are cleared (initialized to zero) when the system is powered-up in the cold start mode.

> When the system is powered-up in the warm start mode, all the process data in the data registers (D 00.00...D 15.63) from when the system was switched off are retained intact.

The warm start is maintained if power failures occur or upon switching from run to program mode or vice-versa.

The start-up behaviour is signalled with the special marker M 40.13 (WARMERR).

The special marker is set to logical 1 or to logical 0 in the first program cycle under the following circumstances respectively:

Upon selecting a cold start

- Start-up with initialised process data M 40.13 = logical 0.

Upon selecting a warm start

- Start-up with correct process data M 40.13 = logical 0.
- Start-up with process data loss M 40.13 = logical 1.

Program protection (USERIDENT)

Operating command: U)

A security code can be entered in the program mode by use of the operating command U). This security code is activated when the programming mode is left.

A return to the programming mode requires the input of this security code, otherwise every instruction (list, print, etc.) will be ignored.

A string of 1 to 4 characters (figures, symbols, letters) is permissible as the USERIDENT (approx. 16 million combinations).

The USERIDENT can be deleted in the programming mode by means of U) and >ENTER<.

5.2 Operating command set in the "RUN-MONITOR" mode

Process data can be displayed and altered during a scanner cycle when the system is in the "RUN-MONITOR" mode (key switch on the CPU turned to the right.

The intervention always occurs between the I/O phase and the processing of the user program.

In the "RUN-MONITOR" mode, the programming terminal shows:

R)

The user program cannot be altered in the "RUN-MONITOR" mode. Only process data (e.g. counter contents) can be shown and be altered.

Display of process data (VIEW)

Operating command: **V)**

Process data can be **displayed** in the **"RUN-MONITOR" mode** by use of the command: V)

The following types of data can be displayed:

Data type	Display
I nn.nn	Binary status (1/0)
O nn.nn	Binary status (1/0)
M nn.nn	Binary status (1/0)
S nn.00	Step counter status (2 digits)
D nn.nn	D-register content (4 digits)
R nn.nn	R-register content (4 digits)

R	V)	I 00.00	Displays status of I 00.00
0001			Status is constantly up-dated
R	V)	D 07.00	Displays content of D 07.00
3798			Value is constantly up-dated
R	V)	S 13.00	Displays content of step counter S 13.00
0039			The current step is continually shown

The display is not up-dated after >ENTER< has been struck.

The operating command, V), is used to display process data that is not shown on the LCD display of the CPU

Modification of the process data (MODIFY)

Operating command: **M)**

Process data can be altered (modified) in the **"RUN-MONITOR" mode** by using the command: **M)**.

The user-program is not affected by using this command.

The following types of data can be modified:

Data type			Status
I	nn.nn	K 1/0	to logical 1/0
O	nn.nn	K 1/0	to logical 1/0
M	nn.nn	K 1/0	to logical 1/0
S	nn.00	K kk	to the value kk
D	nn.nn	K kkkk	to the value kkkk
R	nn.nn	K kkkk	to the value kkkk

The statuses of data types I, O and M are only modified for one scanner cycle. Thereafter, the effective value is automatically assumed by the system.

R	M)	I 00.00 K1	Input 00.00 is set to logical 1 for one program cycle
R	M)	D 15.60 1987	Data register is set a value of 1987 (Displayed on the LCD display)
R	M)	S 13.00 K67	Step counter 13 is set to a value of 67

The operating command, M), is used during the program test and commissioning e.g. for:

- **presetting the step counters step-by-step and checking the outputs**
- **alteration of counter contents and elapsed times**

6 SELECONTROL® PMC programming technique

6.1 Introduction

The SELECONTROL® PMC can be programmed in the form of instruction lists directly from arbitrary documentation such as:

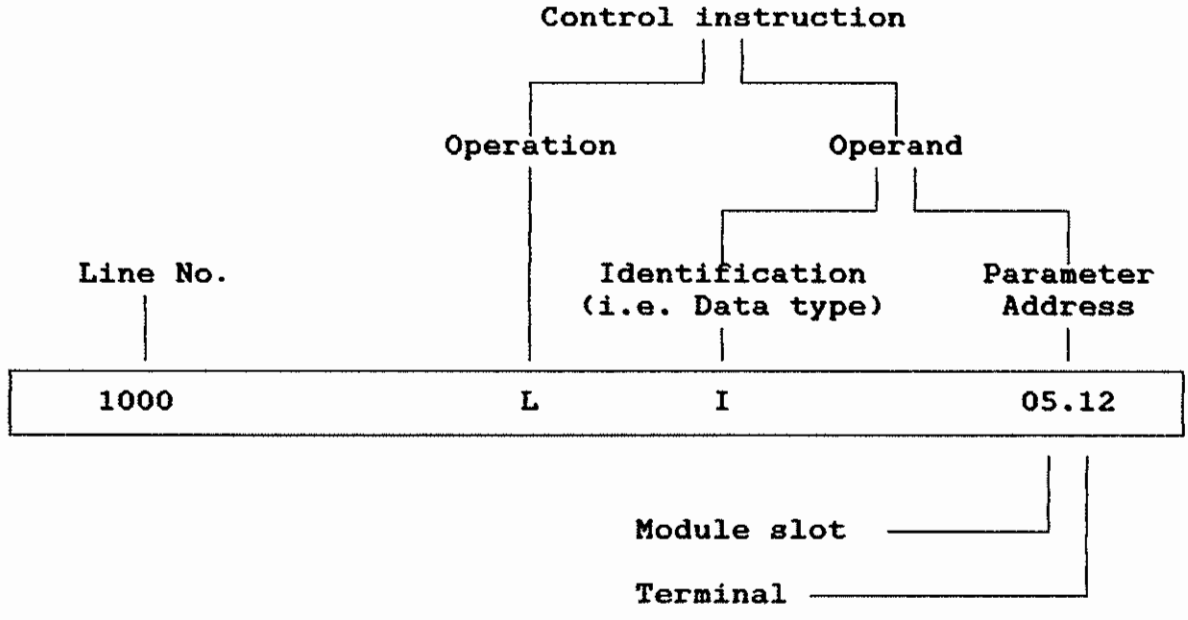
- **Sequence plans**
- **Function plans**
- **Time/path diagrams**
- **Contact switching plans**

The highly refined instruction set includes 58 different operations.

Thanks to the convenience of the instruction set, even complex automation tasks with shift registers, word-processing, arithmetic operations and code changing can be simply and quickly programmed.

The complete instruction set is listed in the following tables.

The tables show which operands can be used with the various operations.



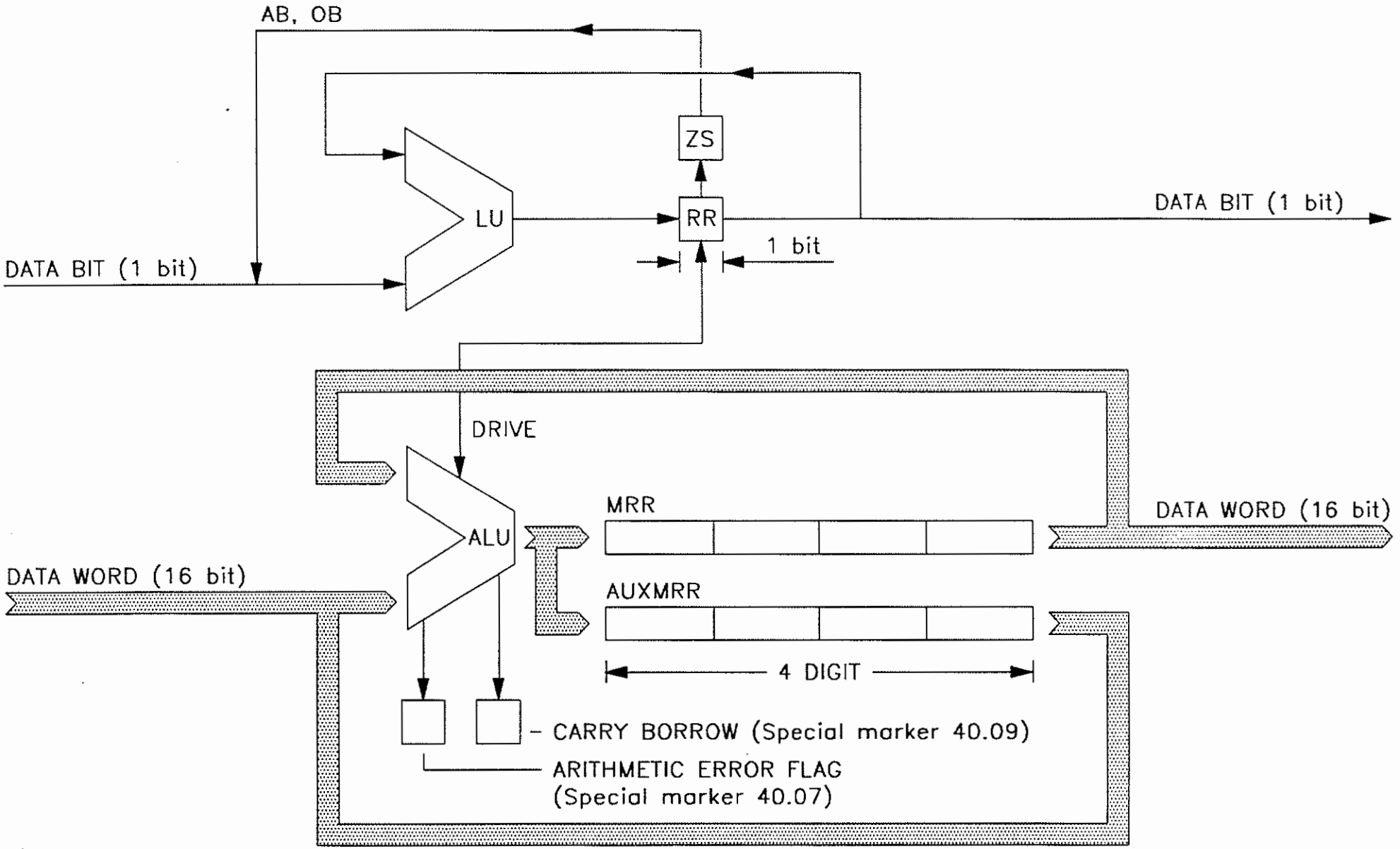
OPERATION	OPERAND								
	INPUT I 00.00 ...00.15	OUTPUT O 01.00 ...15.15	MARKER M 16.00 ...38.15	STEP CNR S 00.00 ...15.99	DATA REGISTER D 00.00 ...15.63	EXTEND. DATA RNG R 00.00 ...15.63	CONST. K 00000 ...0FFFF	IMPLICIT	REMARKS
Logic operations - reading									
L	X	X	X	X	X				Interrogate the
LN	X	X	X	X	X				data registers
A	X	X	X	X	X				for the value 0000
AN	X	X	X	X	X				(evens only)
O	X	X	X	X	X				
ON	X	X	X	X	X				
XO	X	X	X	X	X				
XON	X	X	X	X	X				
AB								X	
OB								X	
Logic operations - writing									
=	X	X	X						
=N	X	X	X						
S	X	X	X	X					Reset data
R	X	X	X		X ---	--------	--------	--------	registers (evens only)
Triggering									
TRG		X	X						
Counting op'ns									Data registers
CU					X				- even
CD					X				- even
Timing op'ns									Data registers
TF					X				- even
TS					X				- even
Data transport									Data registers
FTW	X	X	X		X	X	X		- even
FTB	X	X	X	X	X		X		- all
FTD	X	X	X				X		
FID					X				- even
FIR						X	X		- even
									Data registers
STW	X	X	X		X	X			- even
STB	X	X	X	X	X	X			- all
STD	X	X	X						
SID					X				- even
SIR						X			- even
MDD							X		
MDR							X		
MRD							X		
MRR							X		
SOH							X		
SOD							X		
Word operations									Data registers
AW					X		X		- even
OW					X		X		- even
XOW					X		X		- even

OPERATION	OPERAND								
	INPUT I 00.00 ...15.15	OUTPUT O 00.00 ...15.15	MARKER M 16.00 ...38.15	STEP CNR S 00.00 ...15.99	DATA REGISTER D 00.00 ...15.63	EXTEND. DATA RNG R 00.00 ...15.63	CONSTANT K 00000 ...0FFFF	IMPLICIT	REMARKS
Compare op'ns									Data registers
LT					X		X		- even
LTE					X		X		- even
EQ					X		X		- even
GT					X		X		- even
GTE					X		X		- even
Arithmetic op'ns									Data registers
ADD					X		X		- even
SUB					X		X		- even
MUL					X		X		- even
DIV					X		X		- even
FTR								X	
Shift op'ns									
SFL								X	
SFR								X	
RTR								X	
RTL								X	
BSU					X				
BSD					X				
BR					X				
Step counters									
INC				X					
DEC				X					
Code changing									
BID								X	
DEB								X	
Look-up table									
LKP								X	
Jumps									
JP							X		
JCF							X		
JCT							X		
LB							X		
JS								X	
RET								X	
Prog. org'n									
NOP								X	
EP								X	

Special Markers

The special markers have the following meanings:

Spec. marker	Meaning	Remark
M 40.00	Logical 1	Used for operations that have to be continuously executed (e.g. data transport)
M 40.01	1st cycle	Signals the first program cycle. Used for operations which only have to be executed during the first program cycle (e.g. for resetting data registers upon a warm start)
M 40.02	0.1s clock	Internal 10Hz timing pulses
M 40.03	1.0s clock	Internal 1Hz timing pulses
M 40.06	Short-cct.	Signals outputs switched off because of a short circuit
M 40.07	Arithmetic error	Signals an arithmetic error e.g. - division by 0 - operand with hex value - binary/decimal conversion giving an operand > 9999
M 40.08	Cycle time exceeded	Signals that the cycle time has exceeded 50ms. The time carry for the "TF" (TIMER FAST) will be incorrect.
M 40.09	Carry / borrow	Carry-over information e.g.: - result of addition > 9999 - result of subtraction < 0000



LU = LOGIC–UNIT
ALU = ARITHMETIC–LOCIG–UNIT (activated only if drive present, RR=1)
RR = RESULT–REGISTER
ZS = INTERMEDIATE MEMORY
MRR = MULTIBIT–RESULT–REGISTER
AUXMRR = AUXILIARY–MULTIBIT–RESULT–REGISTER (for arithmetic operations)

QUICK REFERENCE CARD: PMC 1

Nmem.	Drive	RRnew	Intermed. Store (ZS)	MRRnew	AUXMRR	If jumped over (JP,JCT,JCF)	Arithmetic error M 40.07	Carry / Borrow M 40.09	Remarks
L	No	Data	RRold	MRRold	No change		No change	No change	
LN	No	$\overline{\text{Data}}$	RRold	MRRold	No change		No change	No change	
A	No	Data· RRold	No change	MRRold	No change		No change	No change	A combination with a data register (e.g. L D 14.38)
AN	No	$\overline{\text{Data}}$· RRold	No change	MRRold	No chnage	Commands will be	No change	No change	results in the content of the data register being
O	No	Data+ RRold	No change	MRRold	No change	executed: this results,	No change	No change	compared with zero (RR = 1 if content = 0)
ON	No	$\overline{\text{Data}}$+ RRold	No change	MRRold	No change	however, in only the loss	No change	No change	
XO	No	Data⊕ RRold	No change	MRRold	No change	of RR and CC.	No change	No change	
XON	No	$\overline{\text{Data}}$⊕ RRold	No change	MRRold	No change		No change	No change	
AB	No	ZS· RRold	No change	MRRold	No change		No change	No change	
OB	No	ZS+ RRold	No change	MRRold	No change		No change	No change	
=	No	RRold	No change	MRRold	No change		No change	No change	
=N	No	RRold	No change	MRRold	No change	These commands are	No change	No change	
R	If RR=1	RRold	No change	MRRold	No change	not executed.	No change	No change	RD nn.nn clears the content of the data registers
S	If RR=1	RRold	No change	MRRold	No change		No change	No change	
TRG	RR= ⌐	RR=1	No change	MRRold	No change	Executed	No change	No change	Needs an auxiliary marker TRG M XX.XX
INC S	If RR=1	RRold	No change	MRRold	No change		No change	No change	The step counter is incr. or decr. in each cycle (as long as RR = 1).
DEC S	If RR=1	RRold	No change	MRRold	No change		No change	No change	
JP	No			MRRold	No change		No change	No change	
JCT	If RR=1	Gets lost when a JUMP is executed.	Gets lost when a JUMP is executed.	MRRold	No change	These commands are not executed.	No change	No change	These commands process the following program instructions as 'empty' lines until the corresponding LB K kk or until EP
JCF	If RR=1			MRRold	No change		No change	No change	
LB	No			MRRold	No change		No change	No change	
JS	No	Gets lost	Gets lost	Depends on the content of the sub-routine			No change	No change	JS calls program line 0000 to RET as a sub-routine.
RET	No					Executed	No change	No change	Faster SCANNER in SCANNER.

RR: Result register
MRR: Multi-bit result register

QUICK REFERENCE CARD: PMC 2

Mnem.	Drive	RRnew	Intermed. store(ZS)	MRRnew	AUXMRR	If jumped over (JP,JCT,CF)	Arithmetic error M 40.07	Carry / Borrow M 40.09	Remarks
FTW	If RR=1	RRold	No change	Data	No change		No change	No change	
FTB	If RR=1	RRold	No change	Data 0...7	No change		No change	No change	
FTD	If RR=1	RRold	No change	Data 0...3	No change		No change	No change	
STW	If RR=1	RRold	No change	MRRold	No change	These	No change	No change	
STB	If RR=1	RRold	No change	MRRold	No change		No change	No change	
STD	If RR=1	RRold	No change	MRRold	No change		No change	No change	
AW	If RR=1	RRold	No change	Data· MRRold	No change	commands	No change	No change	
OWN	If RR=1	RRold	No change	Data+ MRRold	No change		No change	No change	
XOW	If RR=1	RRold	No change	Data⊕ MRRold	No change		No change	No change	
ADD	If RR=1	RRold	No change	Data+ MRRold	No change	are	Set if characters from A...F occur	Set if sum > 1000	Carry-in is ignored
SUB	If RR=1	RRold	No change	MRRold- Data	No change			Set if difference = 0	Borrow-in is ignored
MUL	If RR=1	RRold	No change	MRRold* Data	Result Highp.			Set if product > 10000	
DIV	If RR=1	RRold	No change	MRRold/ Data	Rest	not	Set after division by 0	No change	
FTR	If RR=1	RRold	No change	AUXMRR	No change		No change	No change	
SFL	If RR=1	RRold	No change	SFL (MRRold)	No change		No change	No change	
SFR	If RR=1	RRold	No change	SFL (MRRold)	No change	executed	No change	No change	
RTL	If RR=1	RRold	No change	RTL (MRRold)	No change		No change	No change	
RTR	If RR=1	RRold	No change	RTR (MRRold)	No change		No change	No change	
BID	If RR=1	RRold	No change	BID (MRRold)	No change		Set if source > 2710H	No change	
DEB	If RR=1	RRold	No change	DEB (MRRold)	No change		Set if source contains characters A...F	No change	

QUICK REFERENCE CARD: PMC 3

Nmemonic	Drive	RRnew	Intermediate store (ZS)	MRRnew	AUXMRR	If jumped over (JP,JCT,JCF)	Arithmetic error M 40.07	Carry / Borrow M 40.09	Remarks
LT	If RR=1		No change	MRRold	No change		No change	No change	
LTE	If RR=1		No change	MRRold	No change		No change	No change	
EQ	If RR=1	RRold* decision-res.	No change	MRRold	No change		No change	No change	
GT	If RR=1		No change	MRRold	No change	These	No change	No change	
GTE	If RR=1		No change	MRRold	No change	commands	No change	No change	
CU	RR= ⌐	Carry 9999-0	No change	MRRold	No change	are	Set if characters A...F occur	No change	Timers that have been jumped over continue to run on.
CD	RR= ⌐	Carry 0-9999	No change	MRRold	No change	not		No change	
TF (10Hz)	RR= ⌐	RRold	No change	MRRold	No change	executed		No change	
TS (1Hz)	RR= ⌐	RRold	No change	MRRold	No change			No change	A timer that has been jumped over cannot, however, be started.
BSU	If RR=1	RRold	No change	Shift out	No change		No change	No change	
BSD	If RR=1	RRold	No change	Shift out	No change		No change	No change	
BR	If RR=1	RRold	No change		No change		No change	No change	
LKP	If RR=1	RRold	No change	MRR:=LKP*	No change		No change	No change	
NOP	No	RRold	No change	No change	No change	These commands are executed in any case.	No change	No change	
EP	No	RR = 0	ZS = 0	MRR = 0	AUXMRR=0		M 40.07=0	M 40.09=0	

* The two lowest digits in the MRR are over-written by the values from the table. The two highest digits in the MRR remain unchanged.

Instruction execution times

Because a microcontroller is used (as the instruction execution device), the execution time for each instruction is dependent on its complexitity. The execution times of the instructions are therefore extremely various. Besides this, the execution time for an individual instruction can also vary.
Various factors, such as:

- drive active/inactive
- size of the parameter to be processed
- instructions jumped over

affect the execution time of the instruction concerned.

Typical execution times are:

Logical instruction (bit-addressable I/O range):	2.....5µs
Word instruction:	2....22µs
Arithmetic instruction:	10...600µs

Auxiliary program to determine the program throughput per second

The display (e.g. D15.60) shows the number of times that the program is executed per second if the following program is added to an existing user program.

Program

Line No.	Operation	Operand	Remark
	L	M 40.00	
	FTW	D 15.54	
	ADD	K 00001	
	STW	D 15.54	
	L	M 40.03	
	TRG	M 38.15	Unused marker
	FTW	D 15.54	
	STW	D 15.60	Free display register
	R	D 15.54	
	EP		

6.2 Logical operations

The operation set complies with DIN 19239 (English).

Operation "L", "LN":

A network is always opened with an "L" operation (Load/If) or with an "LN" operation (Load Not/If Not).

The "L" (Load) operation corresponds to a closing contact.

The "LN" (Load Not) negation corresponds to an opening contact.

Operation "A", "AN":

The "A" (And) operation corresponds to the series connection of a closing contact.

The "AN" (And Not) negation corresponds to the series connection of an opening contact.

Operation "O", ON":

The "O" (Or) operation corresponds to the parallel connection of a closing contact.

The "ON" (Or Not) negation corresponds to the parallel connection of an opening contact.

Operation "=", "=N":

The "=" (Then) operation has the effect of assigning a logical conclusion to an operand (e.g. an output).

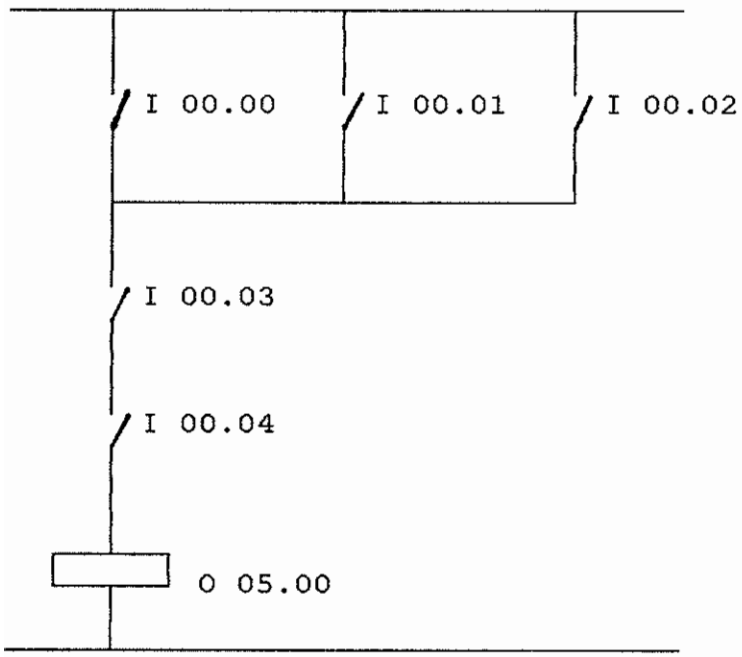
The "=N" (Then Not) negation corresponds to the assignment of an inverse logical result.

In contrast to the "S" (Set) operation, the assignment "=" is always dependent on the instantaneous result of the logical connection.

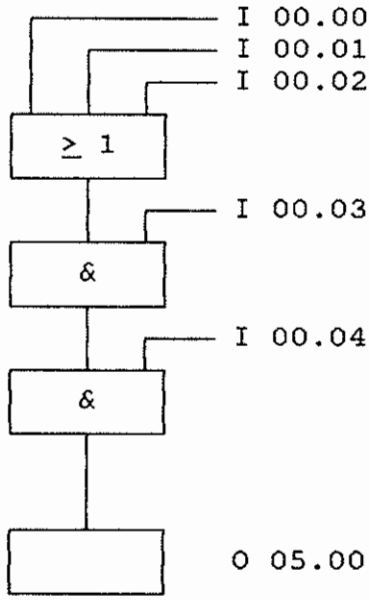
An operand may only be assigned with "=" or "=N" once in a program.

Programming example:

Contact switching plan



Function plan



Program

Line No.	Operation	Operand	Remark
0000	L	I 00.00	If input
0001	O	I 00.01	Or input
0002	O	I 00.02	Or input
0003	A	I 00.03	And input
0004	A	I 00.04	And input
0005	=	O 05.00	Then output
0006	EP		End of program

Operation "AB":

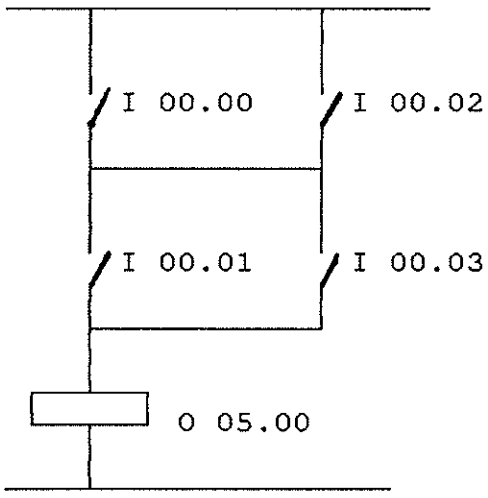
The **"AB"** (And Block) operation serves to simplify programming by replacing brackets.

> **In every "L" and "LN" operation, the preceeding logical result is temporarily stored.**
>
> **In "AB" and "OB" operations, the temporarily stored logical results can be used for further logical decisions.**

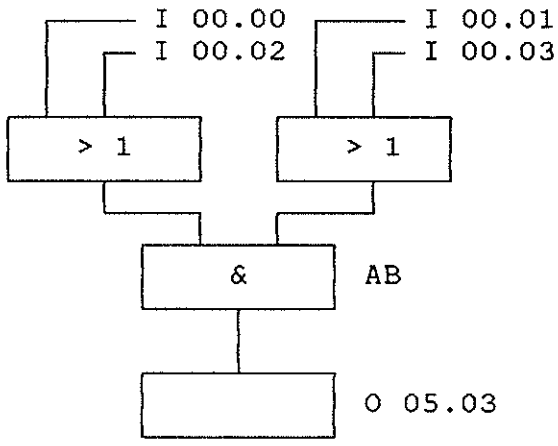
"AB" operations are used without operands.

Program example:

Contact switching plan:



Function plan:



Program

Line No.	Operation	Operand	Remark
0000	L	I 00.00	If input
0001	O	I 00.02	Or input
0002	L	I 00.01	If input
0003	O	I 00.03	Or input
0004	AB		And Block (implicit)
0005	=	O 05.03	Then output
0006	EP		End of program

Operation "OB":

The "OB" (Or Block) function serves to simplify programming by replacing brackets.

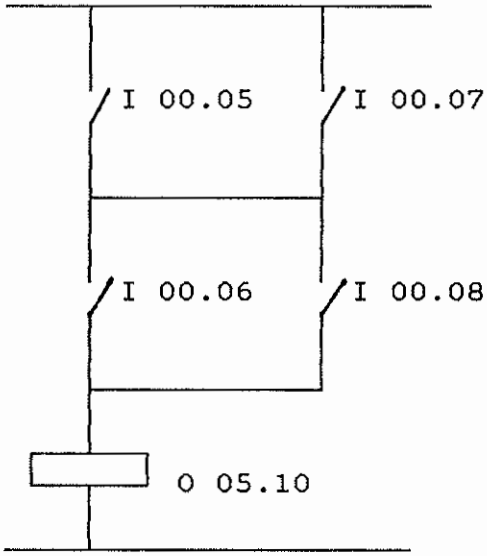
In every "L" or "LN" operation, the preceeding logical result is temporarily stored.

In "AB" or "OB" operations, the temporarily stored logical results can be used for further logical decisions.

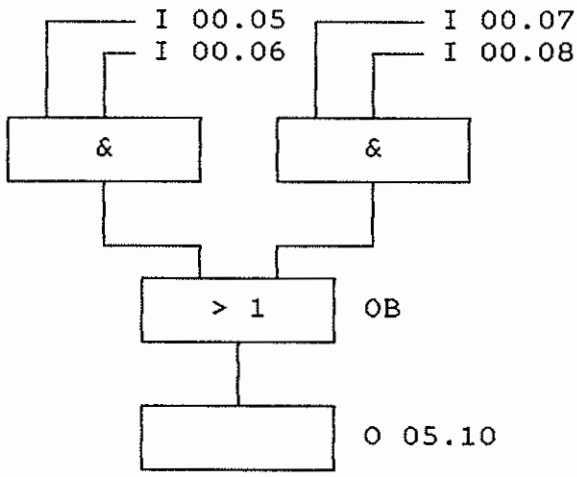
The "OB" operation is used without operands.

Program example:

Contact switching plan:



Function plan:



Program

Line No.	Operation	Operand	Remark
0000	L	I 00.05	If input
0001	A	I 00.06	And input
0002	L	I 00.07	If input
0003	A	I 00.08	And input
0004	OB		Or Block
0005	=	O 05.10	Then output
0006	EP		

Operations "S', "R"

The appropriate operand, e.g an output, is set (logical "1") if the logical result up to the corresponding "S" (SET) operation is logical "1".

The operand remains set until such times as the logical result for the corresponding "R" (RESET) operation is logical "1".

The combination of the "S" and "R" operations corresponds to a (latching) flip-flop.

In contrast to the "=" operation, the operand remains set after an "S" operation until the right conditions for an "R" operation occur.

In view of the sequential program execution, the dominant operation of the two ("S" or "R") is always that which occurred as the last in the program sequence.

When a step counter is set to a specific step (e.g. S 00.01) the prior step is automatically rendered inactive.

The prior step does not have to be reset with an "R" operation.

Program example:

Line No.	Operation	Operand	Remark
0000	L	I 00.00	If input
0001	S	O 05.00	Set output
0002	L	I 00.01	If input
0003	R	O 05.00	Reset output
0004	EP		

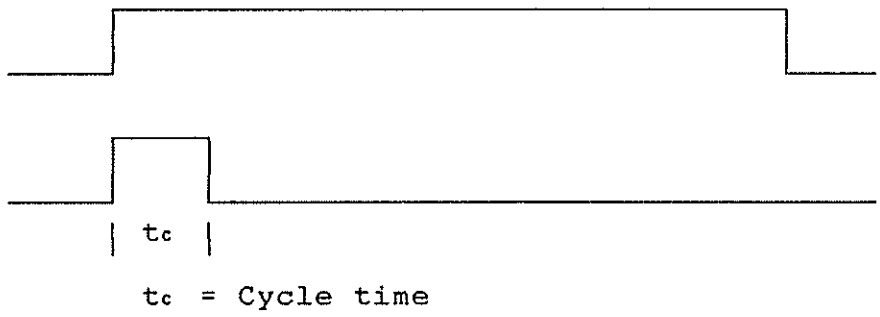
Operation "TRG"

The TRIGGER **"TRG"** operation serves to produce a pulse. The pulse is the length of a program cycle and is triggered from the leading edge of an input signal.

The pulse is formed using an auxiliary marker (M XX.XX) or an output (O XX.XX).

Program example:

Function diagram



t_c = Cycle time

Program

Line No.	Operation	Operand	Remark
0000	L	I 00.04	If input
0001	TRG	M 16.01	Auxiliary marker
0002	=	M 16.01	Then marker (pulse)
0003	EP		

6.3 Analog Timers

The analog timers are programmed in exactly the same way as logic functions.

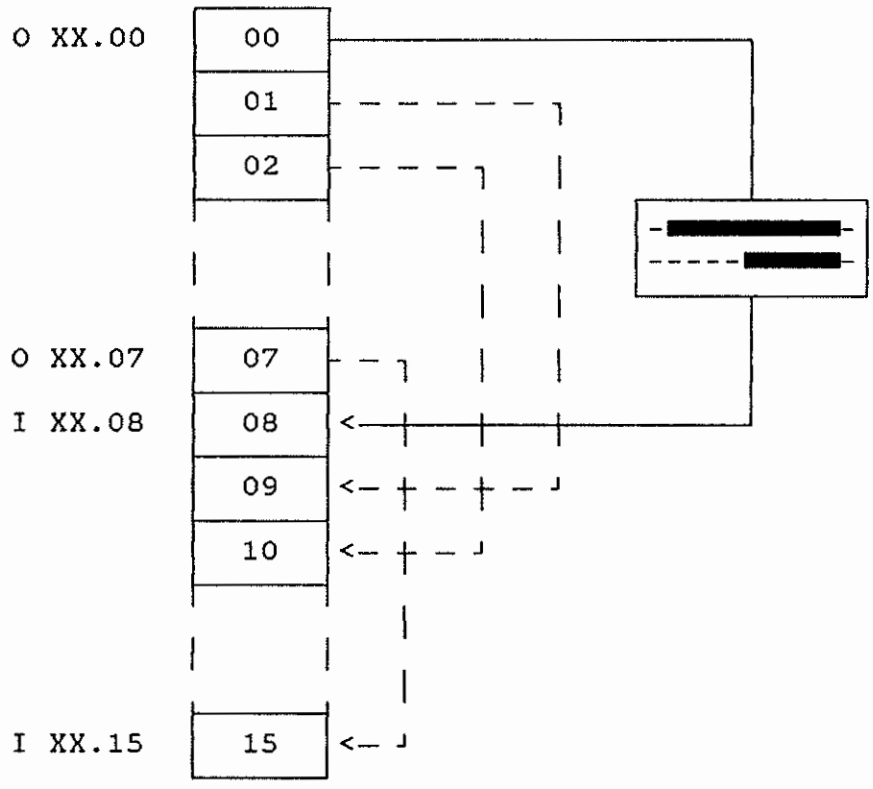
The time period is started by an instruction being sent to the outputs (O XX.00 ... O XX.07).

The inputs (I XX.08 ... I XX.15) are activated when the time has elapsed.

The input remains active after the time has elapsed so long as the output stays active.

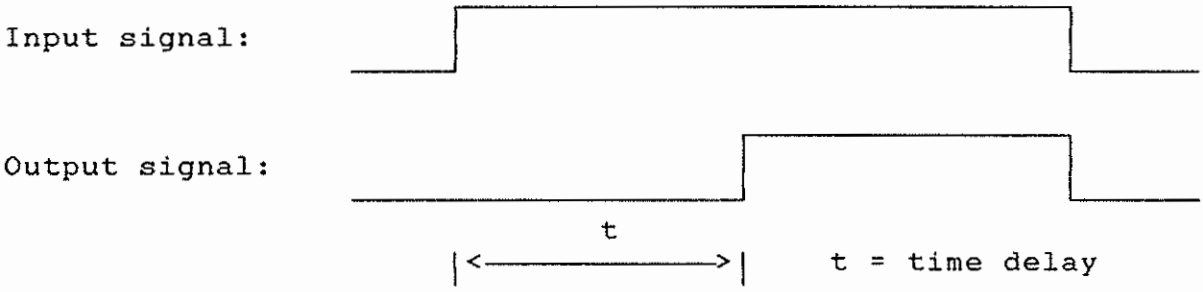
XX.00	start timer 0	→	XX.08 interrogate timer 0
XX.01	start timer 1	→	XX.09 interrogate timer 1
XX.02	start timer 2	→	XX.10 interrogate timer 2
XX.03	start timer 3	→	XX.11 interrogate timer 3
XX.04	start timer 4	→	XX.12 interrogate timer 4
XX.05	start timer 5	→	XX.13 interrogate timer 5
XX.06	start timer 6	→	XX.14 interrogate timer 6
XX.07	start timer 7	→	XX.15 interrogate timer 7

XX = module slot



Program example:

Time diagram



Program

Line No.	Operation	Operand	Remark
0000	L	I 00.50	If input
0001	=	O 06.01	Then start analogue timer
0002	L	I 06.09	If time has elapsed
0003	=	) 05.00	Then output
0004	EP		

6.4 Counters

Operations:

"CU"	**–**	**COUNT UP**
"CD"	**-**	**COUNT DOWN**

The SELECONTROL® PMC has provision for 512 4-decade up/down counters.

The "CU" and "CD" operations are always used together with a data register as the operand.

Only even-numbered data registers can be used as operands.

The counters are triggered by the positive-going edge of the counting signal.

The data register contains the current counter content.

The contents of this data register, and therefore the status of the counter, can be altered in the "RUN-MONITOR" mode.

Program example:

4-decade up-counter (0000 ... 9999)

Line No.	Operation	Operand	Remark
0000	L	I 00.00	If input
0001	CU	D 15.60	Count upwards
0002	EP		

4-decade down counter (9999 ... 0000)

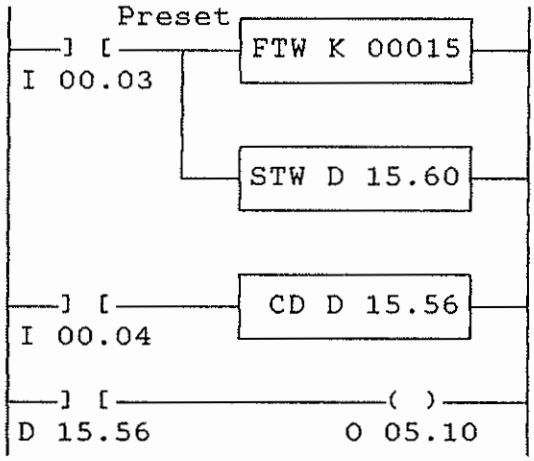
Line No.	Operation	Operand	Remark
0000	L	I 00.01	If input
0001	CD	D 15.60	Count downwards
0002	EP		

The counters can easily be cascaded, e.g. to obtain an 8-decade upwards counter (00000000 ... 99999999).

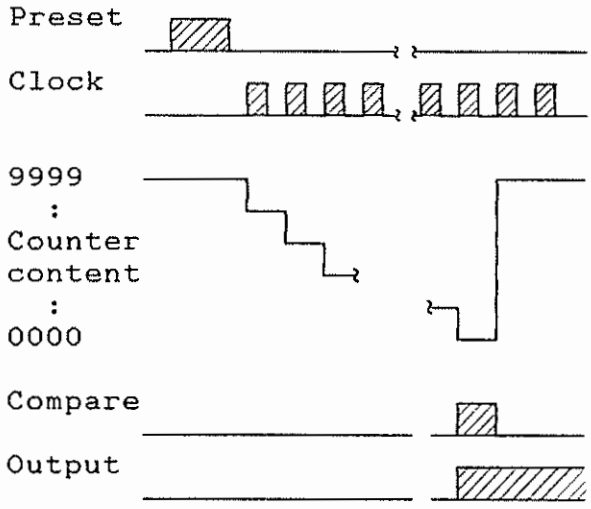
Line No.	Operation	Operand	Remark
0000	L	I 00.00	If input
0001	CU	D 15.60	Count upwards
0002	CU	D 15.62	Carry
0003	EP		

Downwards counter with preset input and output comparison at the value 0000.

Circuit



Timing diagram



Line No.	Operation	Operand	Remark
0000	L	I 00.03	If preset input
0001	FTW	K 00015	Read constant (preselect value)
0002	STW	D 15.56	Write to data register
0003	L	I 00.04	If counter input
0004	AN	D 15.56	And Not content = 0000
0005	CD	D 15.56	Count down data register
0006	L	D 15.56	If content = 0000
0007	=	O 05.10	Then output
0008	EP		

The operations L, LN, A, AN, O, ON, XO or XON using a data register as the operand have the effect of comparing the register content with 0000 (RR = "1" if the data register content = 0000).

6.5 Digital timers

Operations:

"TF"	TIMER FAST	**(0.1s pulses)**
"TS"	TIMER SLOW	**(1.0s pulses)**

The SELECONTROL® PMC has **512 digital timers** available for programming.

The operations "TF" and "TS" are always used together with a data register as the operand.

Only even-numbered data registers can be used as operands.

The data register is loaded with the preselected value for each positive-going edge of the input signal (drive).

A further positive-going edge to the input signal has the effect of reloading the data register with the preselected value (time resetting).

The data register is decremented with 0.1s pulses by the "TF" operation (TIMER FAST) and with 1s pulses by "TS" (TIMER SLOW). Decrementing ends when the content of the data register reaches 0000.

Decrementing continues down to 0000 even if the input conditions are no longer valid or if a jump is made over the program part.

The data register contains the remaining time. In the "RUN-MONITOR" mode, the contents of the register and therefore the remaining time can be altered.

The operations L, LN, A, AN, O, ON, XO, XON using the data register as the operand, have the effect of comparing the contents of the data register with 0000 (RR = "1" if the data register content = 0000).

The **accuracy of the digital time delays** is:

- +0.0/-0.1s in "TF" TIMER FAST operations (0.1s pulses)
- +0.0/-1.0s in "TS" TIMER SLOW operations (1.0s pulses)

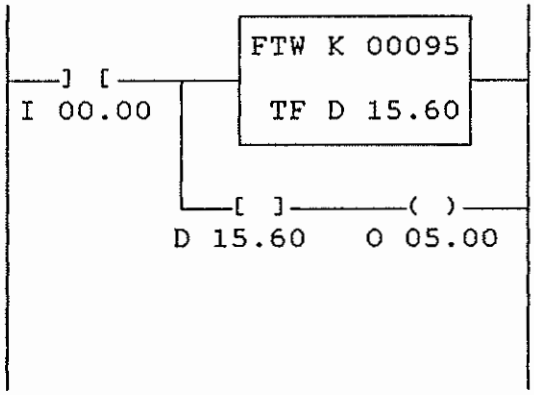
When the cycle time is longer than 50ms, the time carry in "TF" operations will be false and the time delay lengthened.

This condition is signalled by the special marker, M 40.08, (time in excess of the cycle time).

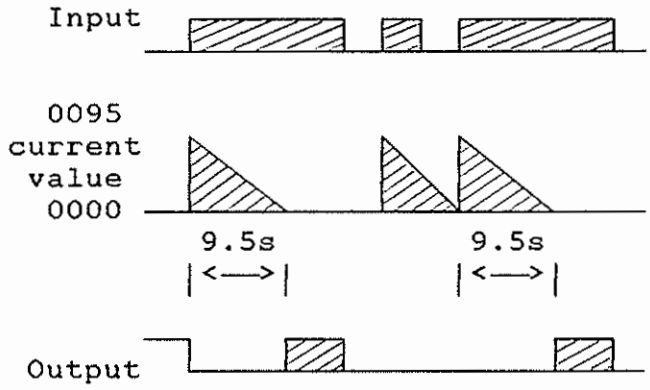
Programming examples

Switch-on delay

Circuit



Timing diagram



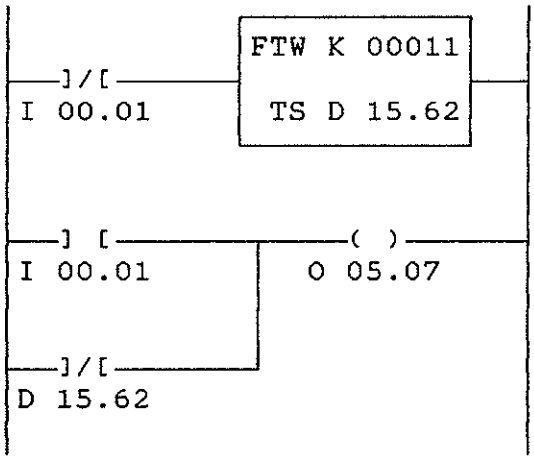
Program

Line No.	Operation	Operand	Remark
0000	L	I 00.00	If input
0001	FTW	K 00095	Read constant (preselected value)
0002	TF	D 15.60	Decr. with 0.1s pulses
0003	A	D 15.60	And data reg. value 0000
0004	=	O 05.00	Then output
0005	EP		

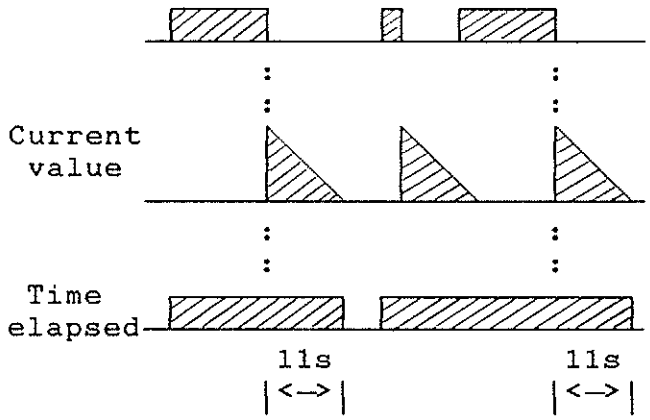
Operands type K or D can be used as the preselected value.

Switch-off delay

Circuit



Timing diagram



Program

Line No.	Operation	Operand	Remark
0000	LN	I 00.01	If Not input
0001	FTW	K 00011	Read constant (preselected value)
0002	TS	D 15.62	Decr. with 1.0s pulses
0003	L	I 00.01	If input
0004	ON	D 15.62	Or Not data register
0005	=	O 05.01	Then output
0006	EP		

Digital timers can also be formed by using special markers (time rate) together with counters.

Special marker M 40.02 (0.1s rate)

Special marker M 40.03 (1.0s rate)

Example: Switch-on delay of 9.5s with up-counter.

Line No.	Operation	Operand	Remark
0000	L	I 00.00	If input
0001	A	M 40.02	And rate = 0.1s
0002	CU	D 15.60	Count upwards
0003	L	I 00.00	If input
0004	FTW	K 00095	Read constant
0005	LT	D 15.60	Compare whether K is less than D
0006	=	O 05.00	Then output
0007	EP		

The time elapse can be stopped by interrupting the start conditions.

The time does not elapse if the program part is jumped over.

The elapsed time can be displayed with "CU" COUNT UP.

The remaining time can be displayed with "CD" COUNT DOWN.

6.6 Step counters (sequential routines)

The freely programmable sequential routines or step counters in the SELECONTROL® PMC are key elements for:

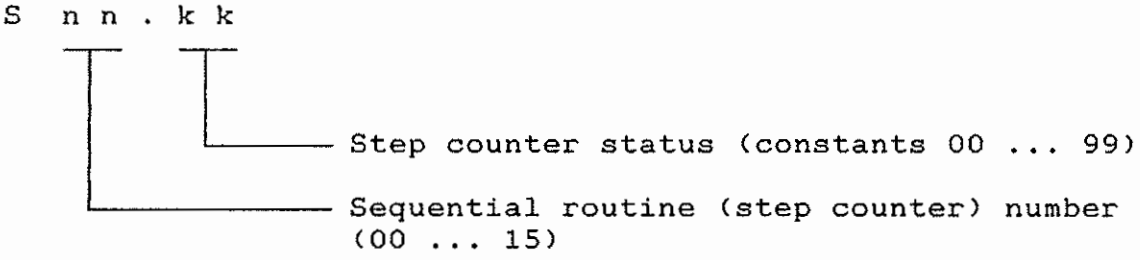
- simple programming of complex sequences
- fault diagnostics and fault reporting
- transparant structuring of programs

Sequential representations with function plans based on the DIN path/time diagrams, Grafcet, flow diagrams or Graph 5 form ideal starting points for direct translation into sequence (or step counter) diagrams.

Organisation of the sequential routines:

Example: 16 sequences of 99 steps each

Format of the operands:



The following address range results:

- step counter S 00 step 00 ... 99 to

- step counter S 15 step 00 ... 99

Access to the sequential routines

Operations for interrogating the step counter content and for setting a specific step count (logical operations).

L	S nn.kk	(or LN)	If step counter (SC)
A	S nn.kk	(or AN)	And step counter
O	S nn.kk	(or ON)	Or step counter
S	S nn.kk		Set step counter

Program example:

Line No.	Operation	Operand	Remark
0000	L	I 00.05	If input
0001	A	S 10.29	And SC 10 to step 29
0002	S	S 10.30	Set SC 10 to step 30
0003	EP		

Only one of the steps (00...99) in a step counter can be active at a time i.e. be at "logical 1". When a new step is set to "logical 1, the previously active step is automatically set to "logical 0" i.e. it is deactivated.

This feature makes sure that the status of a step counter is always meaningful. Use of step counters in linked sequences ensures optimum security.

Practical application of sequential routines

- sequential processes
- processes with flow diagram structures (branching, wait-loops, jumps forwards/backwards, etc.)
- status and alarm displays or fault reports in processes
- fault supervision and diagnostic routines

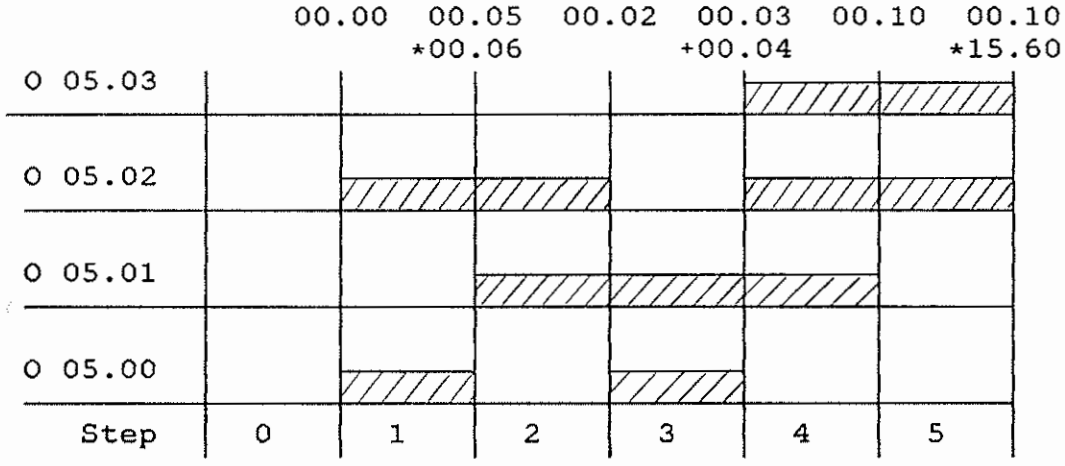
Any process sequence can be sub-divided into individual, defined stages that can be logically linked together.

The advantage of sequential routines is that the individually definable stages can be directly transposed into program steps for the process control.

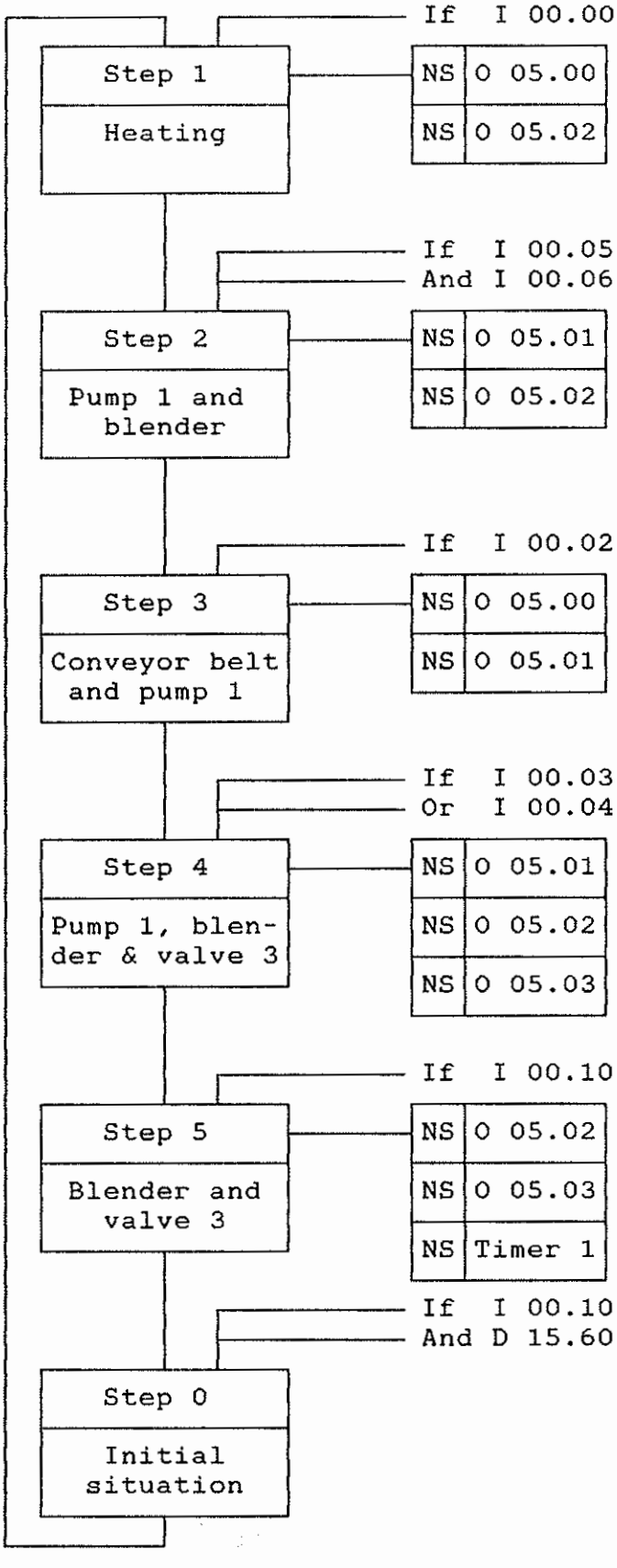
Program example:

The following process which is represented by a path/step diagram and an execution plan can be converted very simply into a program with the SELECONTROL® PMC programming technique.

Time/path diagram:



Function plan



Program sequence

```
0000  L  I 00.00
0001  A  S 00.00
0002  S  S 00.01

0003  L  I 00.05
0004  A  I 00.06
0005  A  S 00.01
0006  S  S 00.02

0007  L  I 00.02
0008  A  S 00.02
0009  S  S 00.03

0010  L  I 00.03
0011  O  I 00.04
0012  A  S 00.03
0013  S  S 00.04

0014  L  I 00.10
0015  A  S 00.04
0016  S  S 00.05

0017  L  I 00.10
0018  A  D 15.60
0019  A  S 00.05
0020  S  S 00.00
```

Program part a) represents the pure programmed sequential routine. The structure is simple and transparent i.e. the conditions for changing from one step to the next are clearly defined. The sequence can only be executed in the order shown, any other non-programmed step is thus automatically excluded. This type of programming ensures the required security into the running of the process and makes an analysis of errors in the sequence very easy.

In the logical operations relating to the step conditions, the AND function together with the current step must be placed at the last position immediately before the assignment of the next step (the so-called 'boss' function) so that the step sequence is arranged in a non-ambiguous manner.

All that is missing now is the program part for the output assignment and the Timer 1:

Program:
Output assignment

```
0021   L   S 00.01
0022   O   S 00.03
0023   =   O 05.00

0024   L   S 00.02
0025   O   S 00.03
0026   O   S 00.04
0027   =   O 05.01

0028   L   S 00.01
0029   O   S 00.02
0030   O   S 00.04
0031   O   S 00.05
0032   =   O 05.02

0033   L   S 00.04
0034   O   S 00.05
0035   =   O 05.03

0036   L   S 00.05
0037   A   D 15.60
0038 FTW   K 00005
0039 STW   D 15.60
0040   L   M 40.03
0041  CD   D 15.60
```

Operations with access to the step counter content (status):

FTB S nn.00 The content of step counter nn is read and can be displayed.

STB S nn.00 Sets the step counter.

Program example:

Line No.	Operation	Operand	Remark
0000	L	I 00.00	If input
0001	FTB	K 00027	Read constant
0002	STB	S 00.00	Write to SC 00, step 27
0003	L	M 40.00	If special marker (log. 1)
0004	FTB	S 00.00	Read step counter status
0005	STB	D 15.60	Display counter status
0006	EP		

The current step of a step counter can be read with the operation FTB S nn.00 and be shown on the LCD display.

Incrementing and decrementing step counters:

INC	S nn.00	Increments the step counter
DEC	S nn.00	Decrements the step counter

The operations INC and DEC raise and decrease the step counter content by 1 respectively.

Program example:

Line No.	Operation	Operand	Remark
0000	L	M 40.03	If clock = 1.0s
0001	TRG	M 16.00	Trigger aux. marker
0002	INC	S 15.00	Increments SC 15
0003	L	S 15.11	If SC 15 is at step 11
0004	S	S 15.00	Set SC 15 to step 00
0005	L	M 40.00	If marker is log. 1
0006	FTB	S 15.00	Read counter status
0007	STB	D 15.60	Write data reg. (display)
0008	EP		

The display shows the steps 0 ... 10 at 1s intervals.

6.7 Data transport

Besides the logical operations (bit operations) the SELECONTROL® PMC also processes digit, byte and word operations.

Type of operation	Processing width
Bit operation	1 bit
Digit operation	4 bits
Byte operation	8 bits
Word operation	16 bits

Operations:

"FTW"	-	Fetch word
"FTB"	-	Fetch byte
"FTD"	-	Fetch digit

"STW"	-	Store word
"STB"	-	Store byte
"STD"	-	Store digit

The inputs, outputs and markers can be accessed bit, digit, byte or word-wise.

Inputs Outputs Markers	Logic	Data transport		
	Bit	Digit	Byte	Word
XX.00	XX.00	D XX.00	B XX.00	W XX.00
XX.01	XX.01			
XX.02	XX.02			
XX.03	XX.03			
XX.04	XX.04	D XX.04		
XX.05	XX.05			
XX.06	XX.06			
XX.07	XX.07			
XX.08	XX.08	D XX.08	B XX.08	
XX.09	XX.09			
XX.10	XX.10			
XX.11	XX.11			
XX.12	XX.12	D XX.12		
XX.13	XX.13			
XX.14	XX.14			
XX.15	XX.15			

Data transport operations are used in conjunction with registers (D or R-registers).

The data registers have a parallel function to the markers since they accept multi-bit data (e.g. counter contents).

The registers can be accessed byte or word-wise.

Register	8 bits		Byte access	Word access
XX.00			I B XX.00	W XX.00
XX.01			I B XX.01	
XX.02			I B XX.02	W XX.02
XX.03			I B XX.03	
XX.04			I B XX.04	W XX.04
XX.05			I B XX.05	
XX.62			I B XX.62	W XX.62
XX.63			I B XX.63	

Data transport operations are only executed if the corresponding drive is at logical 1

Program example:

Line No.	Operation	Operand	Remark
0000	L	I 00.00	If input
0001	FTW	I 00.00	Read word module 00
0002	STW	D 15.60	Write to data register
0003	STW	O 05.00	Write word module 05
0004	EP		

The binary data word from module 00 will only be read if the drive, I 00.00, is active.

The value will be written into data register 15.60 and be shown on the display in binary code.

The binary data word will be sent to the output module 05.

If the special marker, M 40.00, (logical 1) is used as the drive, the data transport operation will be performed continuously.

Program example:

Line No.	Operation	Operand	Remark
0000	L	M 40.00	If marker is logical 1
0001	FTW	I 00.00	Read word from module 00
0002	STW	D 15.60	Write to data register
0003	STW	O 05.00	Write word to module 05
0004	EP		

DATA OUTPUT via the RS 232 serial programming interface

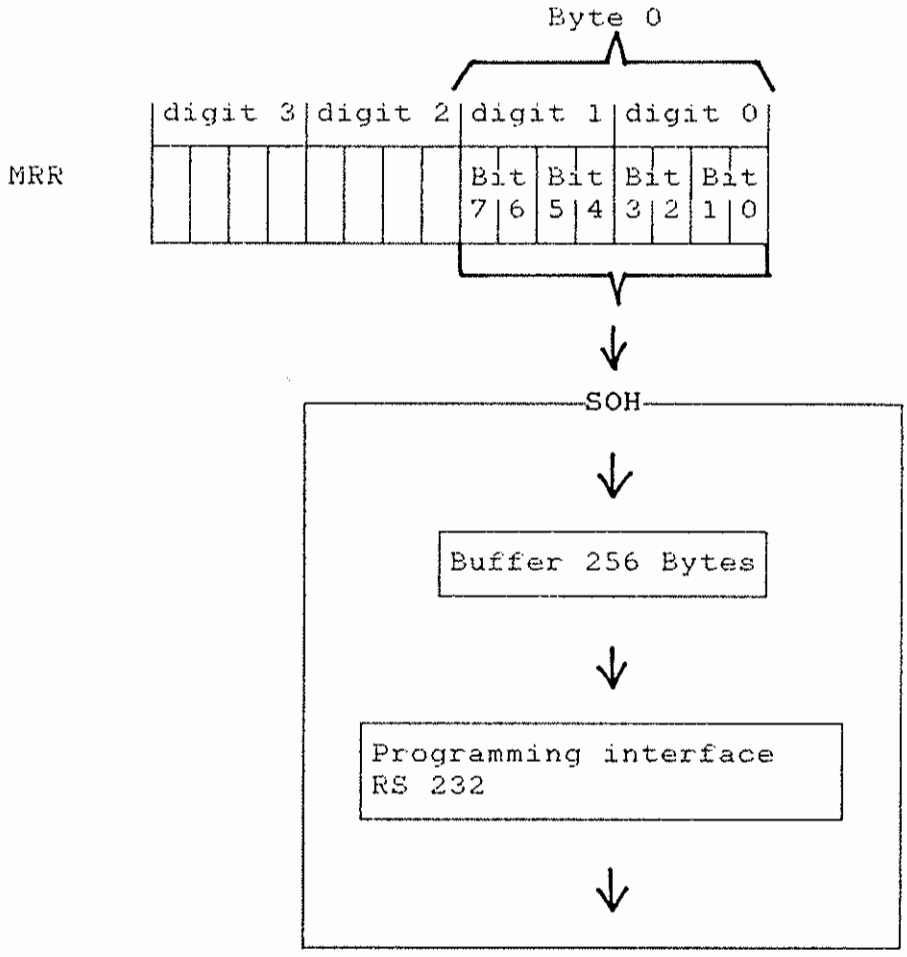
Operations:

"SOH" serial out hex	(Byte 0 of the MRR is written into the buffer without being converted).
"SOD" serial out digit	(Digit 0 of the MRR is converted from numerical to ASCII before being written into the buffer).

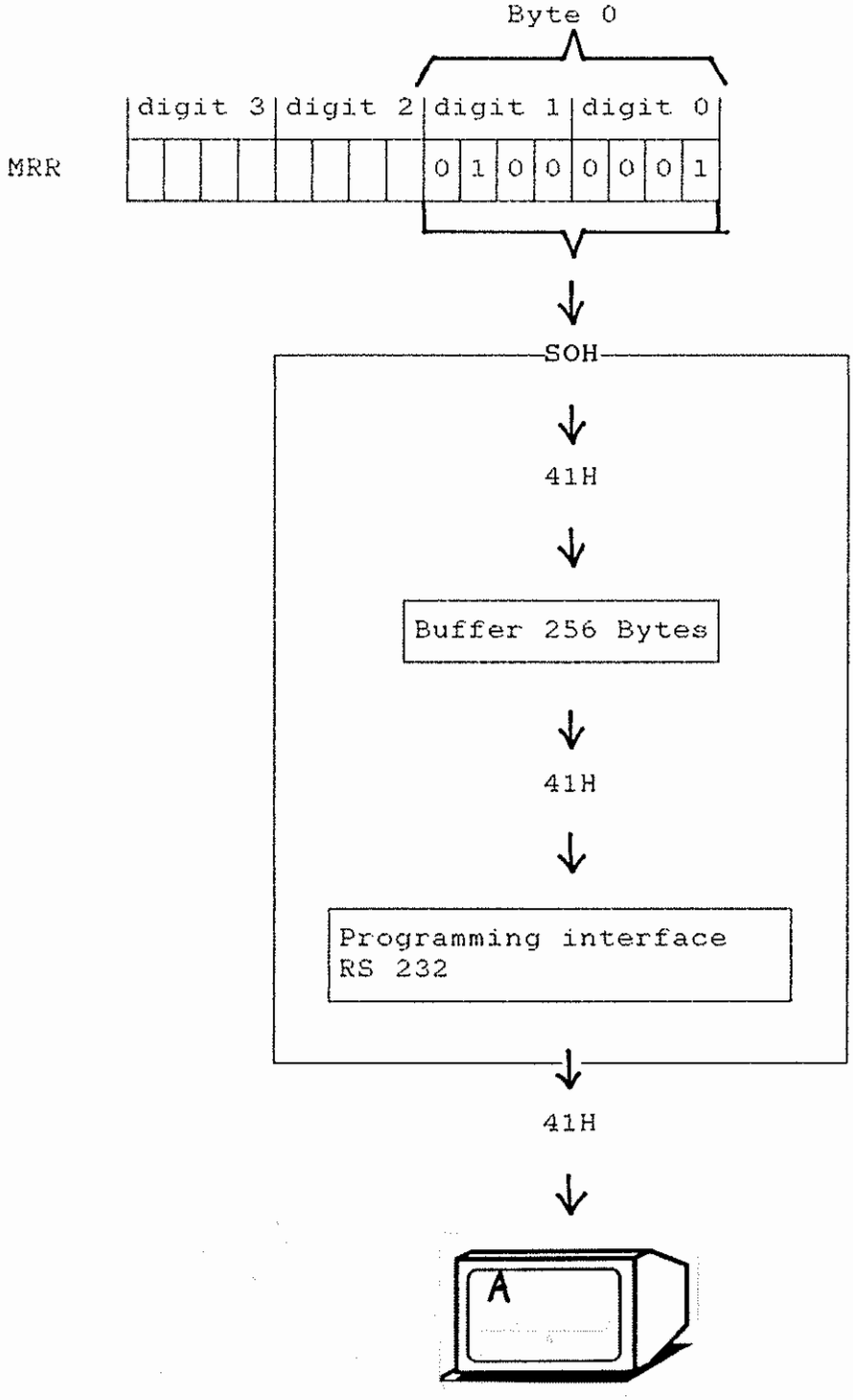
Data is written from the multibit result register (MRR) into the output buffer (FIFO) with the commands "SOH" and "SOD". Up to a maximum of 256 characters can be written into the output buffer. The data is passed automatically from the output buffer to the RS 232 programming interface.
The buffer overflow is drive-controlled.

"SOH" Command

The Command transfers the 8 least significant bits that are in the MRR (7-bit ASCII 00H...0FFH) to the buffer without conversion. One character is output to the serial interface by the system at the end of each cycle.
The command is only executed if the drive is at logical "1". The drive is influenced by the command execution as follows:
- Drive = 1: if the transfer to the buffer has been completed.
- Drive = 0: if the transfer to the buffer could not be completed e.g. if the buffer is full.


Byte 0
digit 3
digit 2
digit 1
digit 0
MRR
Bit 7
Bit 6
Bit 5
Bit 4
Bit 3
Bit 2
Bit 1
Bit 0
SOH
Buffer 256 Bytes
Programming interface
RS 232

"SOH" Example


Byte 0
digit 3
digit 2
digit 1
digit 0
MRR
0 1 0 0 0 0 0 1
SOH
41H
Buffer 256 Bytes
41H
Programming interface
RS 232
41H
A

"SOD" Command

The "SOD" command converts the lowest digit that is in the MRR (OH...FH) into the corresponding ASCII value and transfers the chaacter to the output buffer. One character is output to the serial interface by system at the end of each cycle (via the FIFO-OUT).
The command is only executed if the drive is at logical "1".
The drive is influenced by the command execution as follows:
- Drive = 1: if the transfer to the bufffer has been completed.
- Drive = 0: if the transfer to the buffer could not be completed e.g. if the buffer is full.

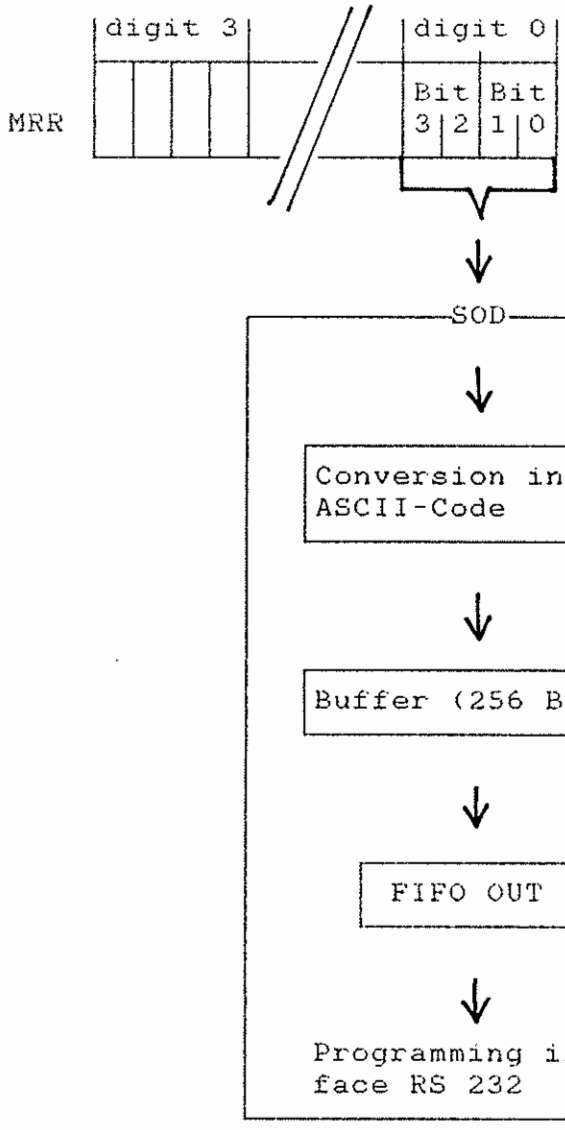
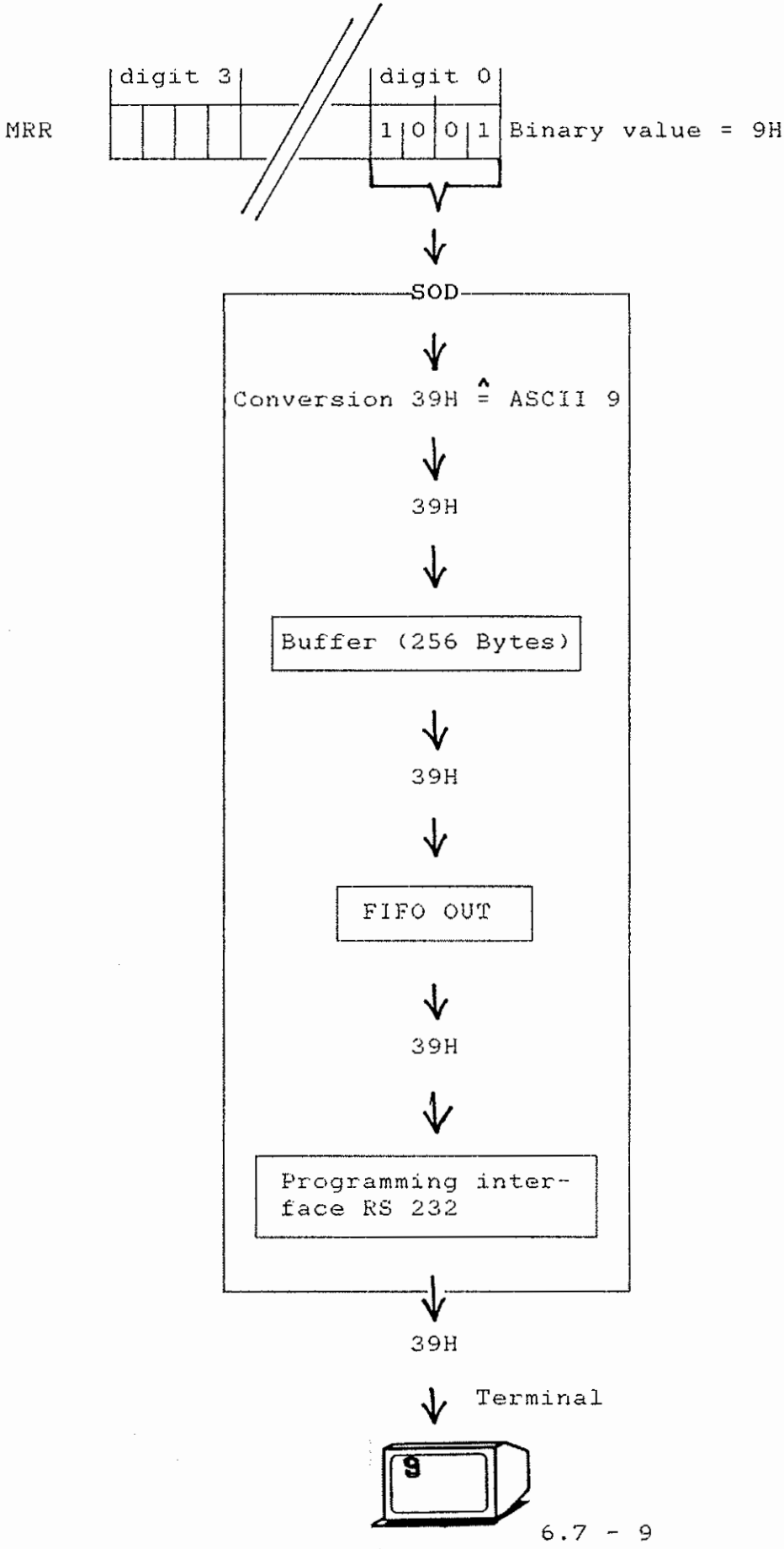


Table for conversion into ASCII code

0 H conversion 30H ≙ ASCII 0
1 H conversion 31H ≙ ASCII 1
.
.
.
9 H conversion 39H ≙ ASCII 9
A H conversion 41H ≙ ASCII A
.
.
.
F H conversion 46H ≙ ASCII F

Example "SOD"


digit 3
digit 0
MRR
1 0 0 1
Binary value = 9H
SOD
Conversion 39H ^= ASCII 9
39H
Buffer (256 Bytes)
39H
FIFO OUT
39H
Programming inter-
face RS 232
39H
Terminal
9

6.8 Word logic operations

A handy extension to the 1-bit logic operation is provided by the 16-bit word operations.

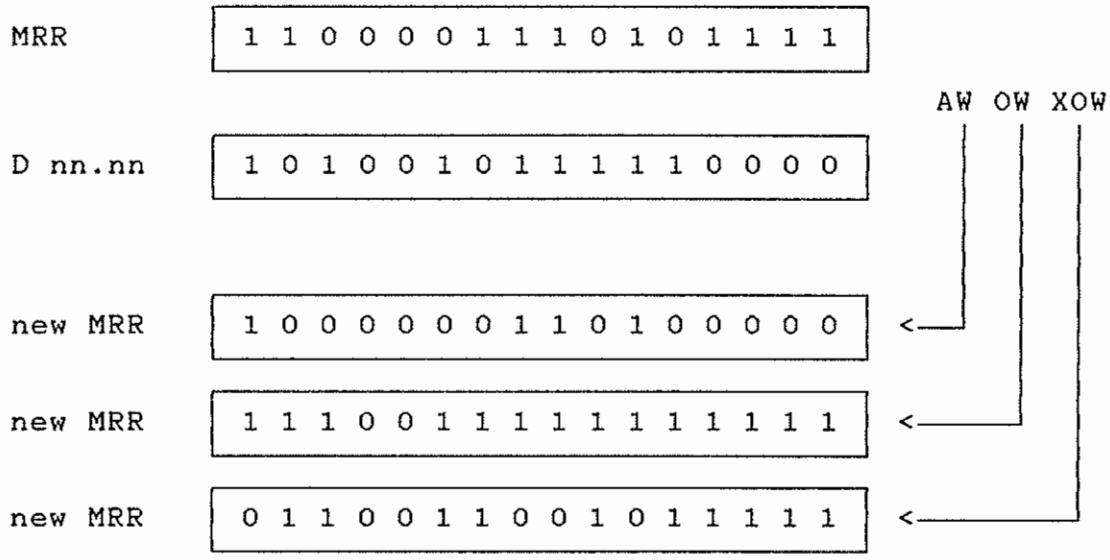
"AW"	**-**	**AND word**
"OW"	**-**	**OR word**
"XOW"	**-**	**EXCLUSIVE OR word**

These logic operations operate on the following operands:

Data registers		D nn.nn
Constants		K kkkkk

Syntax:	AW	D nn.nn	(or AW K kkkkk)
	OW	D nn.nn	(or OW K kkkkk)
	XOW	D nn.nn	(or XOW K kkkkk)

Example:



The logic operations are performed between the individual corresponding bits and the final result is placed in the multi-bit result register (MRR).

Uses:

- Masking (comparisons, masking and checking of bit patterns)
- Auxiliary arithmetic operations
- Supervision of input variables (digital 16-bit values via digital or analog input modules)
- Clearing the multi-bit result register, MRR (AW K 00000)

Program example:

The statuses at inputs I 00.00 ... I 00.15 have to be compared with the statuses at inputs I 01.00 ...I 01.15.

Any differences should activate the appropriate output O 05.00 O 05.15.

Line No.	Operation	Operand	Remark
0000	L	M 40.00	If marker = logical 1
0001	FTW	I 00.00	Read word from module 00
0002	STW	D 15.60	Write to data register
0003	FTW	I 01.00	Read word from module 01
0004	XOW	D 15.60	EXOR (difference)
0005	STW	O 05.00	Write word to module 05
0006	EP		

6.9 Comparison operations

Operations:

"LT" – LESS THAN

"LTE" – LESS THAN OR EQUAL

"EQ" – EQUAL

"GT" – GREATER THAN

"GTE" – GREATER THAN OR EQUAL

The **"LT"** (**"LTE"**) operations examine the value in the multi-bit result register to see whether it is less than (less than or equal) to the value of a constant or a data register.

The **"EQ"** operation examines the value in the multi-bit result register to see whether it is the same as the value of a constant or a data register.

The **"GT" ("GTE")** operations examine the value in the multi-bit result register to see whether it is greater than (greater than or equal) the value of a constant or a data register.

The comparison operations are only executed if their drive is at logical 1.

The special marker, M 40.00, (logical 1) causes the comparison to be performed continuously.

Besides comparisons of BCD values, it is also possible to compare hex values.

Program example:

Line No.	Operation	Operand	Remarks
0000	L	I 00.00	If counter input
0001	CU	D 15.60	Count upwards data reg.
0002	L	M 40.00	If marker = logical 1
0003	FTW	D 15.60	Read data register
0004	GTE	K 00005	Compare whether content of D 15.60 $\geq$ 5
0005	=	O 05.00	Then output
0006	EP		

The output 05.00 will be set if the counter status (D 15.60) reaches or exceeds the value 5.

The "EQ" EQUAL operation can be cascaded to compare values that are greater than 4-bits wide.

Program example:

Line No.	Operation	Operand	Remarks
0000	L	M 40.00	If marker = logical 1
0001	FTW	D 15.60	Read data register
0002	EQ	K 04298	Examine whether content of D 15.60 = 4298
0003	FTW	D 15.62	Read data register
0004	EQ	K 05397	Examine whether content of D 15.62 = 5397
0005	=	O 05.00	Then output
0006	EP		

Output 05.00 will be activated when the data registers contain the BCD value of 53,974,298.

6.10 Arithmetic

The SELECONTROL® PMC is capable of performing the four basic mathematical operations with 4-digit operands.

Operations:

"ADD"	-	**Addition**
"SUB"	-	**Subtraction**
"MUL"	-	**Multiplication**
"DIV"	-	**Division**

A constant (K kkkkk) or a data register (D nn.nn) serves as the operand.

All the operations refer to the integer value in the first quadrant (whole positive values).

The mathematical operations are only executed if their drive is at logical 1.

The mathematical operations will be performed in every program cycle as long as the drive remains at logical 1.

For this reason, the **"TRG" TRIGGER** operation is used in the following program example.

Program example: addition

Line No.	Operation	Operand	Remarks
0000	L	I 00.00	If input
0001	TRG	M 16.00	Trigger aux. marker
0002	FTW	D 15.60	Read data register
0003	ADD	K 01234	Add constant
0004	STW	D 15.60	Write to data register
0005	EP		

The **special marker, M 40.09** (carry/borrow), is set if the result of an addition is greater than 9999 or the result of a subtraction is less than 0000.

This special marker retains its status until the next mathematical operation is performed.

> **Addition and subtraction can be performed with arbitrary values.**
>
> **Any carry or borrow sets the special marker M 40.09.**

Program example: addition with carry.
The constant 1234 has to be added to the value in the data register D 15.60. The carry amount is to be written inot data register D 15.62.

Line No.	Operation	Operand	Remarks
0000	L	I 00.00	If input
0001	TRG	M 16.00	Trigger aux. marker
0002	FTW	D 15.60	Read data register
0003	ADD	K 01234	Add constant
0004	STW	D 15.60	Write to data register
0005	A	M 40.09	And special marker carry
0006	FTW	D 15.62	Read data register
0007	ADD	K 00001	Add constant (carry)
0008	STW	D 15.62	Write to data register
0009	EP		

Program example: subtraction with borrow.
The constant 1250 has to subtracted from the value 27500 each time the input I 00.00 is activated.

Line No.	Operation	Operand	Remarks
0000	L	I 00.15	Preset if input
0001	FTW	K 07500	Read constant
0002	STW	D 15.60	Write to data register
0003	FTW	K 00002	Read constant
0004	STW	D 15.62	Write to data register
0005	L	I 00.00	If input
0006	TRG	M 16.00	Trigger aux. marker
0007	FTW	D 15.60	Read data register
0008	SUB	K 01250	Subtract constant
0009	STW	D 15.60	Write to data register
0010	A	M 40.09	And special marker borrow
0011	FTW	D 15.62	Read data register
0012	SUB	K 00001	Subtract constant
0013	STW	D 15.62	Write to data register
0014	EP		

Program example: multiplication

Line No.	Operation	Operand	Remarks
0000	L	I 00.00	If input
0001	FTW	K 00021	Read constant
0002	MUL	K 00022	Multiply by constant
0003	STW	D 15.60	Write to data register
0004	EP		

If the result of a multiplication is greater than 9999, the more significant figures (10^4 ... 10^7) are written into an auxiliary register.

This additional register can be read with the operation "FTR" FETCH AUXILIARY REGISTER and the data can be processed further.

Program example: multiplication with a result over 9999.

Line No.	Operation	Operand	Remarks
0000	L	I 00.00	If input
0001	FTW	K 02100	Read constant
0002	MUL	K 00222	Multiply by constant
0003	STW	D 15.60	Write to data register
0004	FTR		Read aux. register
0005	STW	D 15.62	Write to data register
0006	EP		

Program example: division.

Line No.	Operation	Operand	Remarks
0000	L	I 00.00	If input
0001	TRG	M 16.00	Trigger aux. marker
0002	FTW	K 07654	Read constant
0003	DIV	K 00100	Divide constant
0004	STW	D 15.60	Write to data register
0005	FTR		Read aux. register (rest)
0006	STW	D 15.56	Write to data register
0007	EP		

The special marker, M 40.07, is set if one of the operands is outside the BCD range or if the divisor in 0000 (zero).

6.11 Jump operations

The use of step counters is described in Section 6.6 of the SELECONTROL programming technique. This possibility enables freely programmable PMC sequential routines to be prepared with arbitrary jumps backwards and forwards. The jump operations which follow are additional aids to program construction. They make it possible to skip parts of the program.

"JP" - JUMP UNCONDITIONALLY
"JCT" - JUMP CONDITIONALLY ON TRUE
"JCF" - JUMP CONDITIONALLY ON FALSE
"LB" - LABEL (jump destination)
"JS" – JUMP TO SUB–ROUTINE
"RET" – RETURN (from jump address)

The jump operations affect the cycle time; lengthening or shortening it as the case might be.

Conditioned jumps

JCT (jump conditionally on true)

The jump is only made if the preceding logical operation has produced a true (i.e. logical 1) result.

The **jump address** is given by a label **LB K 000kk** (k = 00 .. 99).

Example:

Line No.	Operation	Operand	Remarks
xxxx	JCT	K 000kk	Jump to label LB K 000kk
:			:
:			:
:			:
:			Skipped program part
:			:
:			:
:			:
yyyy	LB	K 000kkk	Jump address: label LB K 000kk

JCF (jump conditionally on false)

The jump is only made if the preceding logical operation has produced a false (i.e. logical 0) result.

The **jump address** is given by a label **LB K 000kk** (k = 00 .. 99).

Maximum number of labels (LB) is: 100 (k = 00000...00099)

Uncondition jumps

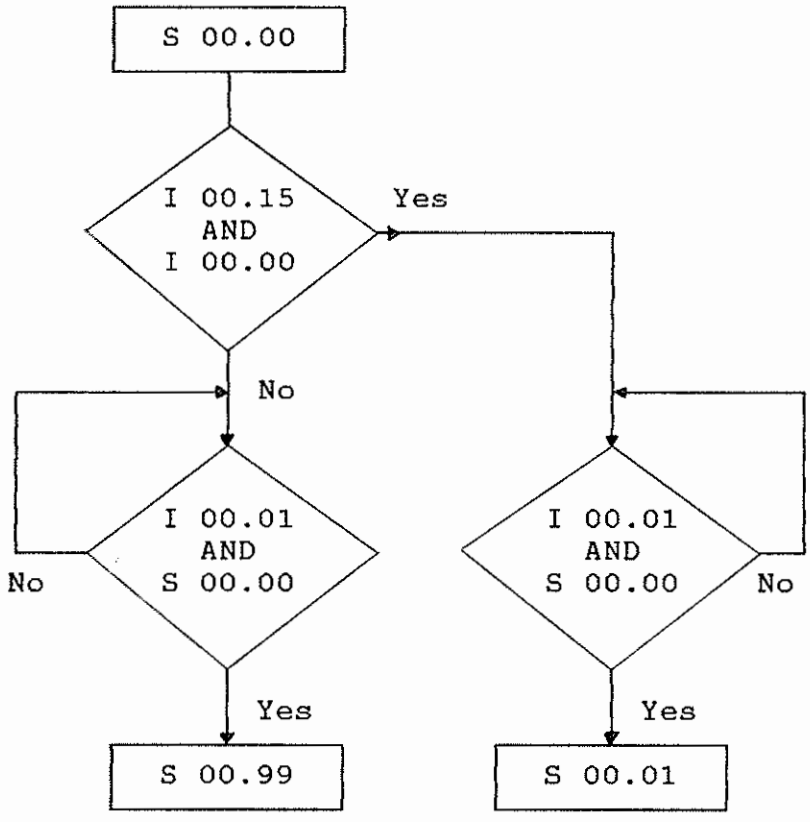
JP (jump)

This jump is made, irrespective of the result of the preceding logical operation, to a jump address given by a label (LB) in the program.

Example:

Line No.	Operation	Operand	Remarks
xxxx	JP	K 000kk	Jump to label LB K 000kk
:			:
:			:
:			:
:			Skipped program part
:			:
:			:
:			:
yyyy	LB	K 000kk	Jump address label LB K 000kk

Program example:



Program example:

Line No.	Operation	Operand	Remarks
0000	L	I 00.15	If input
0001	A	I 00.00	And input
0002	JCT	K 00001	Jump if true
0003	L	I 00.10	If input
0004	A	S 00.00	And step counter
0005	S	S 00.01	Set step counter
0006	JP	K 00002	Unconditional jump
0007	LB	K 00001	Jump address label 1
0008	L	I 00.01	If input
0009	A	S 00.00	And step counter
0010	S	S 00.99	Set step counter
0011	LB	K 00002	Jump address label 2
0012	L	M 40.00	If marker = logical 1
0013	FTB	S 00.00	Read step counter
0014	STB	D 15.60	Write to data register
0015	EP		

When the inputs I 00.15 and I 0.00 are logical 1, the program part 0003 - 0006 will be skipped and only the part 0007 - 0011 will be executed. If the result of the operation I 00.15 AND I 00.00 is logical 0, the program part 0003 - 0005 will be executed and an unconditional jump will be made from address 0006 to the address 0011.

It is assumed here that at the start of the program I 00.10 is at logical 1 and that the step counter 00.xx is at step 00. In the first case above, the display of D 15.60 would show step 99 and in the second case the display would show 01.

Program example:

In the RUN–MONITOR mode, a value in data register D 15.56 that is greater or less than 50 has to be input by using the M) MODIFY command [M) D 15.56 K kkkk]. If the value is less than 50, +50 has to be added to it and the result stored in the data register D 15.60. If the value is over 50 it can be stored in data register D 15.60 directly.

Display window:

a)

```
00 00 00 83
00 00 00 83
```

Example for K > 50 (here K = 83)

b)

```
00 00 00 77
00 00 00 27
```

Example for K < 50 (here K = 27)

Program

Line No.	Operation	Operand	Remarks
0000	L	I 00.00	If input
0001	FTW	D 15.56	Read data register
0002	LT	K 00050	If content D < 50
0003	JCF	K 00001	Jump to LB K1 if D > 50
0004	L	I 00.00	If input
0005	ADD	K 00050	Add constant
0006	STW	D 15.60	Write to data register
0007	JP	K 00002	Jump to LB K2 unconditionally
0008	LB	K 00001	Label LB K1
0009	L	I 00.00	If input
0010	STW	D 15.60	Write to data register
0011	LB	K 00002	Label LB K2
0012	EP		

After every jump command, a DRIVE (RR = 1) must be established again for drive-dependent operations. The result register (RR) changes its status during the execution of a jump routine. Its status is therefore different when the jump address is reached.

Establish a new DRIVE after every jump if DRIVE-DEPENDENT OPERATIONS follow.

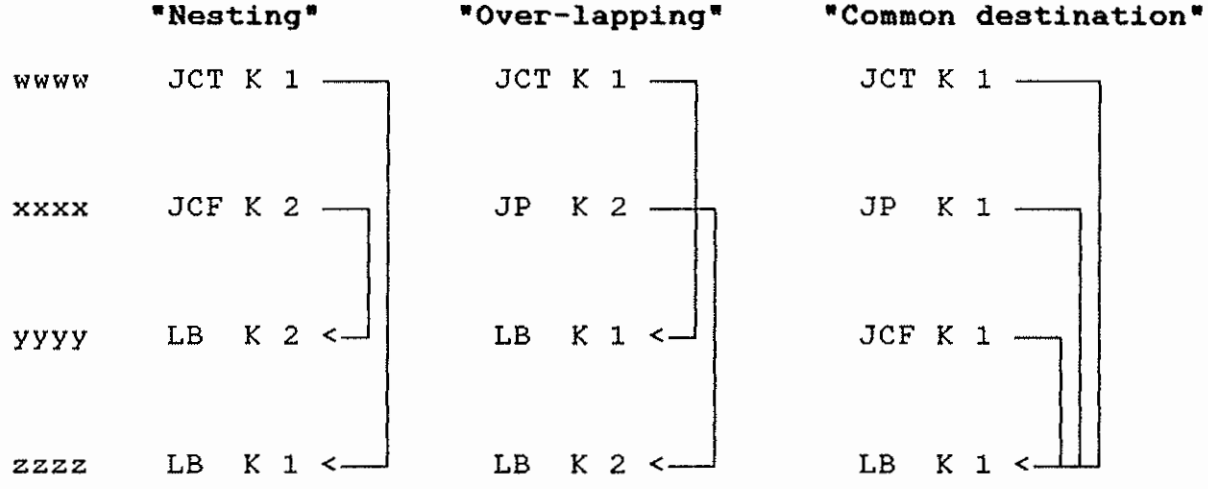
1-bit commands in the skipped part of the program are executed while multi-bit commands are checked but not executed. Instructions such as =, =N, S, STW, STB, STD, etc. are not executed.

Jump routines affect the program cycle time.

Only forward jumps can be made with the JP, JCT and JCF commands. Backward jumps would conflict with the scanner principle.

EP (end of program) and RET (end of sub-routine) instructions cannot be jumped over.

"Nesting" and over-lapping of jump routines as well as common destination addresses (label LB) are all basically possible.



Jump to a sub-routine

JS (Jump Sub-routine)

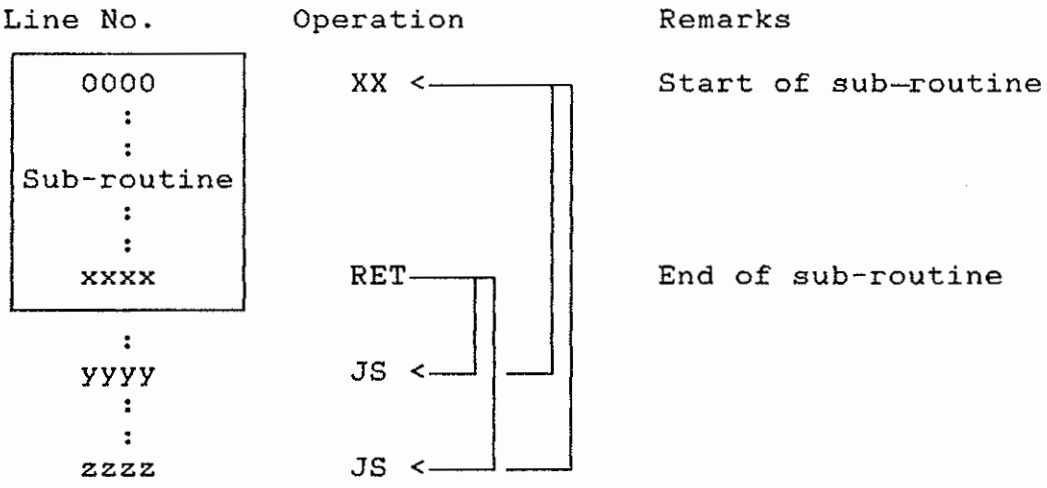
The JS (Jump Sub-routine) has the effect of causing a jump to be made to the line number 0000.

Basically, only one sub-routine can be included.

The start of a sub-routine in always 0000. The end of the sub-routine is indicated by a RET (return) instruction.

RET causes a jump to be made to the next program line following that from which the JS (Jump Sub-routine) operation was started.

Example:



The sub-routine can be called any number of times. The cycle time will, however, be extended.

Use of the jump sub-routine command:

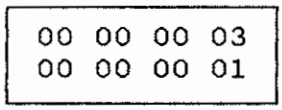
Time-critical processes can be controlled by use of the sub-routine. The sub-routine acts as a fast scanner within the main scanner i.e. it is a short program that can be called at the critical moment.

Example:

Line No.	Operation	Operand	Remarks
0000	L	I 00.00	If input
0001	AN	M 16.00	And Not marker
0002	FTW	D 15.60	Read data register
0003	ADD	K 00001	Add constant
0004	STW	D 15.60	Write to data register
0005	RET		End of sub-routine
0006	NOP		
0007	JS		Jump to sub-routine
0008	NOP		
0009	JS		Jump to sub-routine
0010	NOP		
0011	L	I 00.00	If input
0012	AN	M 16.00	And Not marker
0013	FTW	D 15.56	Read data register
0014	ADD	K 00001	Add constant
0015	STW	D 15.56	Write to data register
0016	S	M 16.00	Set inhibiting marker
0017	EP		

Once the input, I 00.00, has been activated the sub-routine (0000 - 0005) will be executed three times after which the data register D 15.60 will have the value 3. After the last part of the program (0006 - 0017) has been executed, the value in register D 15.56 will be 1 and the inhibiting marker will be set. The operations FTW, ADD and STW will not be executed in any following program cycles.

Display:



The RET (return) instruction cannot be jumped over.

6.12 Shift operations

The shift operations operate directly on the multi-bit result register (MRR).

The shift operations have the effect of shifting the contents of the MRR register by 1 digit to the left or right.

A shift operation is only executed if the corresponding drive is at logical 1.

Operations:

"SFL"	**–**	**SHIFT LEFT**
"SFR"	**-**	**SHIFT RIGHT**
"RTL"	**-**	**ROTATE LEFT**
"RTR"	**-**	**ROTATE RIGHT**

Uses:

- Data manipulations of all kinds e.g. multiplication or division of values by 10, 100 or 1000
- Formatting the contents of registers
- Operations with masks (in conjunction with AW "AND WORD" and OW "OR WORD"
- Formatting the LCD display

SFL (Shift left)

This operation has the effect of moving the contents of the multi-bit result register (MRR) by 1 digit to the left.

It should be noted that the most significant digit (10^3) will be lost.

Example:

4	7	8	3

Status of MRR **before** an **SFL** operation

7	8	3	0

Status of MRR **after** an **SFL** operation

Program example:

The content of data register D 15.60 is to be shifted by two places to the left and the result stored in register D 15.56.

Line No.	Operation	Operand	Remarks
0000	L	I 00.00	If input
0001	AN	M 16.00	Inhibit marker
0002	FTW	D 15.60	Load data register
0003	SFL		Shift by 1 digit left
0004	SFL		Shift by 1 digit left
0005	STW	D 15.56	Store in data register
0006	S	M 16.00	Set inhibit marker
0007	EP		

Check on the example:

An arbitrary value (e.g. 5678) is loaded into the data register D 15.60 in the RUN-MONITOR mode.

If input I 00.00 is activated, the value shown for the content of register D 15.56 would be 7800.

Display:

```
0 0 0 0  5 6 7 8
0 0 0 0  7 8 0 0
```

SFR (Shift right)

This operation has the identical effect as SFL except that the shift is to the right.

Example:

4	7	8	3

Status of MRR **before** an **SFR** operation

0	4	7	8

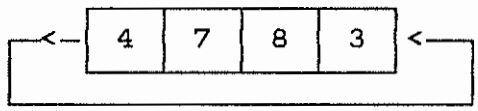
Status of MRR **after** an **SFR** operation

The most or least significant digit is lost after each SFL or SFR operation respectively!

RTL (Rotate left)

This operation has the effect of rotating the contents of the MRR by 1 digit to the left.

This shifts the most significant digit (10^3) to the position previously occupied by the least significant digit (10^0), thus:



Status of the MRR **before** an RTL

7	8	3	4

Status of the MRR **after** an RTL

Program example:

The contents of data register D 15.60 have to be rotated by 2 positions to the left and the result stored in register S 15.56

Line No.	Operation	Operand	Remarks
0000	L	I 00.00	If input
0001	AN	M 16.00	Inhibit marker
0002	FTW	D 15.60	Read data register
0003	RTL		Rotate by 1 digit left
0004	RLT		Rotate by 1 digit left
0005	STW	D 15.56	Store in data register
0006	S	M 16.00	Set inhibit marker
0007	EP		

Check on the example:

An arbitrary value (e.g. 5678) is loaded into the data register D 15.60 in the RUN-MONITOR mode.

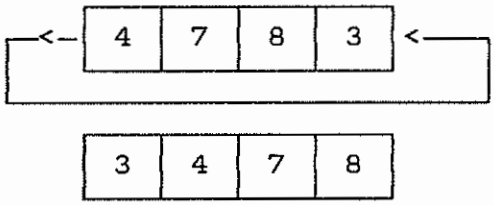
If input I 00.00 is activated, the value shown for the content of the data register would be 7856.

Display:

0 0 0 0	5 6 7 8
0 0 0 0	7 8 5 6

RTR (Rotate right)

This operation has the identical effect as RTL except that the rotation is to the right.



Status of the MRR **before** an RTR

status of the MRR **after** an RTR

6.13 Block-shift operations

"BSU" - Block shift up
"BSD" - Block shift down
"BR" - Block reset

These operations shift a data block of 64 bytes (32 words) by one word unit (i.e. by 2 bytes or 16 bits).

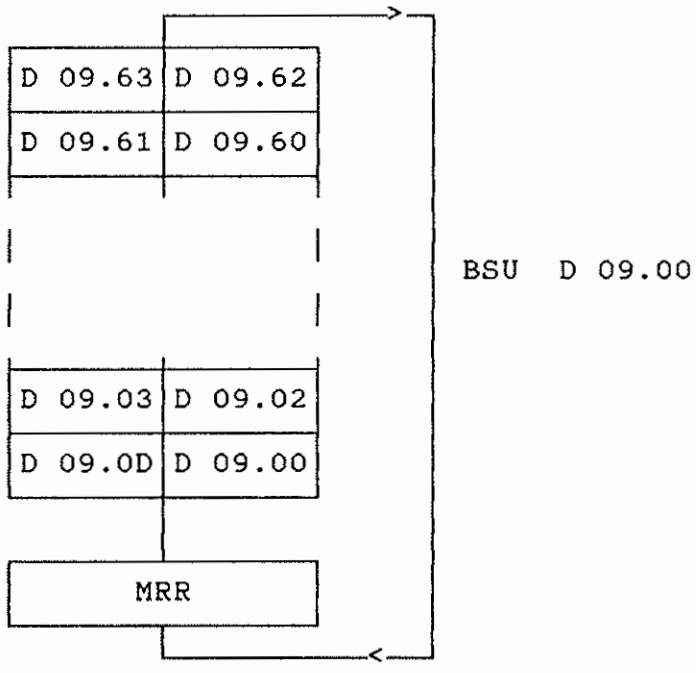
The effect of these operations is to move the contents of the MRR into the first word of the defined block while the last word of the defined block is transfered to the MRR or vice-versa.

Syntax: BSU D nn.00
BSD D nn.00

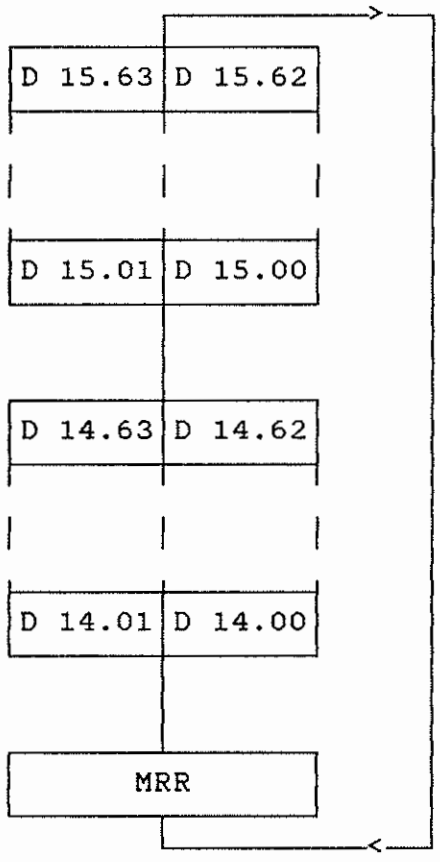
whereby nn.00 represents the data block from 00.00 to 15.00.

The operations only operate on data register blocks and are only executed if the corresponding drive is at logical 1.

BSU (Block shift up)



The operations can be cascaded so that whole block groups can be shifted.



Uses:

Setting up FIFO data register blocks (First In / First Out) for the chronological-sequential storage or output of data.

Typical applications:

- Cyclic processing of incoming external data (binary or hexadecimal as well as BCD values.

- chronological storage of addresses in sorting tasks.

Example:

The value "3579" has to be continuously shifted through the data register block, D 15.00 ..... D 15.62, at a 1s clock rate (i.e. a round-shift).

Line No.	Operation	Operand	Remarks
0000	L	I 00.00	If input
0001	A	M 40.03	And marker 1s clock
0002	TRG	M 16.00	Trigger aux. marker
0003	=	M 16.01	Then marker (pulse)
0004	L	M 16.01	If marker
0005	FTW	D 15.62	Read data register in MRR
0006	BSU	D 15.00	Block-shift upwards
0007	EP		

Check on the program:

Preset the data register, D 15.56, to K 3579 in the RUN-MONITOR mode by means of the operating command M).

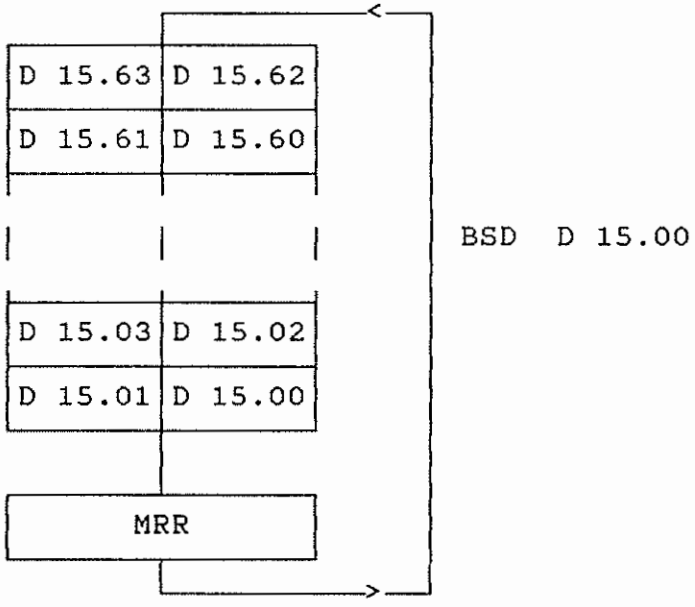
If input I 00.00 is activated, the value of 3579 will be moved through the defined data register block, D 15.00, and will appear in the display window every 32 seconds.

0	0	0	0
0	0	0	0

3	5	7	9
0	0	0	0

BSD (Block shift down)

The BSD operation has the effect of shifting the data register block downwards i.e. in the direction of the decreasing line number values.



BR (Block Reset) : Clearing a data register block

This operation sets all the values in the defined data register block to 0000. It is only executed if the corresponding drive is at logical 1.

Syntax: BR D 15.00

Effect:

All the data registers in the block from D 15.00 to D 15.63 will be reset to 0000.

6.14 Code-changing operations

BID	-	**Binary into BCD**
DEB	-	**Decimal into binary**

These operations operate on the 4-digit result register, MRR, and are only executed when the corresponding drive is at logical 1.

The value from the MRR is put back into the MRR after being converted.

0	0	0	1	1	0	0	1	0	1	1	0	1	1	1	1	MRR **before** operation **BID**

0	1	1	0	0	1	0	1	0	0	0	1	0	0	0	1	MRR **after** operation **BID**

Uses:

BID: Conversion of binary values (e.g. from analog input modules) into decimal values for further processing with arithmetic functions.

DEB: Conversion of decimal values into binary values for output purposes (e.g. via output modules) or to binary-based peripherals.

Program example:

Conversion from binary into decimal form:

Line No.	Operation	Operand	Remarks
0000	L	M 40.00	If marker = logical 1
0001	FTW	I 00.00	Read input module 00
0002	BID		Convert binary into BCD
0003	STW	D 15.60	Display (decimal
0004	STW	O 05.00	Write to output module 05
0005	EP		

Conversion decimal into binary:

Line No.	Operation	Operand	Remarks
0000	L	M 40.00	If marker = logical 1
0001	FTW	I 00.00	Read word, BCD
0002	STW	D 15.60	Display (decimal)
0003	DEB		Convert code
0004	STW	O 05.00	Output (binary)
0005	EP		

Important

The special marker, M 40.07 (arithmetic error), is set if the decimal equivalent exceeds the value of 9999 as a result of a code changing operation.

6.15 Table look-up operation

The LKP operation provides a simple and fast means of accessing a table having 256 freely assignable places for 2-digit values.

The operation requires its drive to be at logical 1.

Organisation of the table:

Table addresses									
00 ... 07	00	00	00	00	00	00	00	00	
08 ... 0F	00	00	00	00	00	00	00	00	
10 ... 17	00	00	00	00	34	00	00	00	Address 14 Value 34
18 ... 1F	00	00	00	00	00	00	00	00	
20 ... 27	00	00	00	00	00	00	00	00	
28 ... 2F	00	00	79	00	00	00	00	00	Address 2A Value 79
30 ... 37	00	00	00	00	00	00	00	00	
38 ... 3F	00	00	00	00	00	00	00	00	
...									
E0 ... E7	00	00	00	00	00	00	00	00	
E8 ... EF	00	8F	00	00	00	00	00	00	Address E9 Value 8F
F0 ... F7	00	00	00	00	00	00	00	00	
F8 ... FF	00	00	00	00	00	00	00	00	

Use of the LKP table:

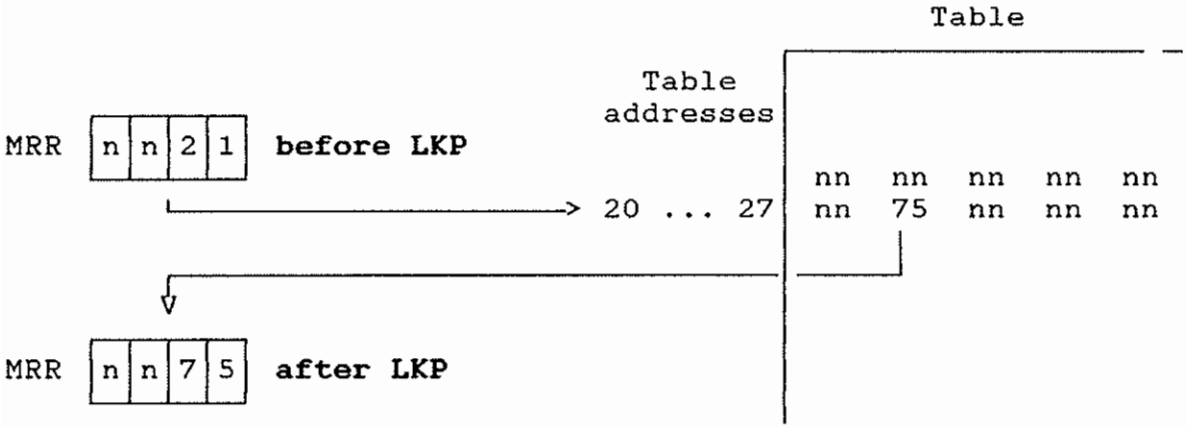
- Code changing of 8-bit (2-digit) values such as BCD into 7-segment code or BCD into Grey-code, etc.

- Simple transformation of, for instance, linear functions.

Operational manner of the LKP operation

The two lowest digits in the MRR contain the table addresses before the operation.

After the operation, these two lowest digits of the MRR contain the value from the table.



The table addresses (e.g. as 21 in the above example) can originate from data registers, step counters or input modules.

Example:

FTB S 12.00 The current step in step counter number 12 supplies the address for the table.

Filling the table

The table can be filled in the "PROGRAMMING" mode via the programming terminal or from the SELECONTROL CAP 3000 programming system.

Operating command: **T)** XX.nn

For example: T) 1A.43 has the following effect:

		Table				
Table address 1A -	00 ... 07	nn	nn	nn	nn	nn
	08 ... 0F	nn	nn	nn	nn	nn
	10 ... 17	nn	nn	nn	nn	nn
	18 ... 1F	nn	nn	43	nn	nn
	20 ... 27	nn	nn	nn	nn	nn

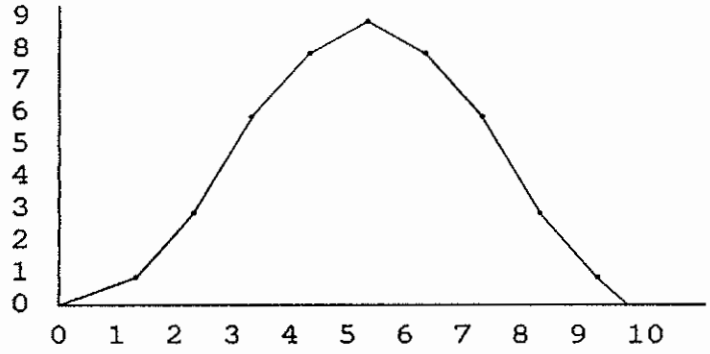
Displaying the table

The operating command **H)** in the PROGRAMMING mode has the effect of causing a listing of the contents of the table to be displayed on the programming system.

Deleting the table

The operating command **N)1234** clears the whole program and deletes the contents of the table.

Program example:



Task:
The shape shown above should be 'followed' at 1s intervals using a step counter and with the aid of the look-up table.
Display: SC via reg. D 15.60, table-value via reg. D 15.56.

Solution:
Increment the step counter up to a count of 10 using the 1s clock. The steps form the addresses for the table where the corresponding values 0, 1, 3, 6, 8 and 9 are stored.

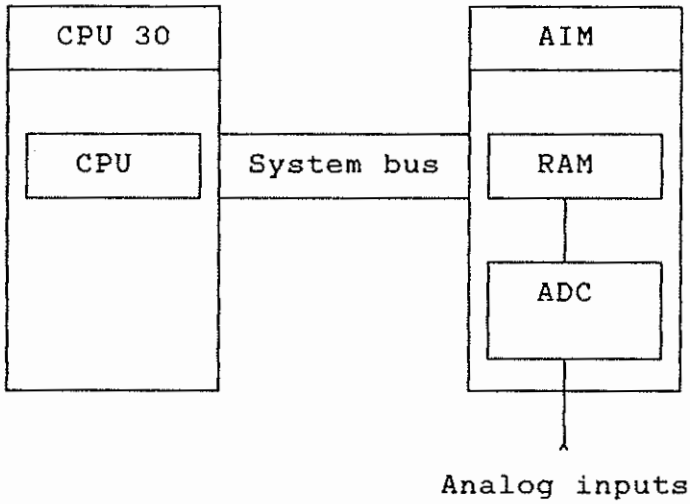
Line No.	Operation	Operand	Remarks
0000	L	I 00.00	If input
0001	A	M 40.03	And marker, 1s clock
0002	TRG	M 16.00	Trigger
0003	INC	S 00.00	Increment SC
0004	L	I 00.00	If input
0005	FTB	S 00.00	Read step counter
0006	LKP		Code conversion
0007	STB	D 15.56	Display table value
0008	L	S 00.10	If step 10
0009	S	S 00.00	Set step 00
0010	L	M 40.00	If marker = logical 1
0011	FTB	S 00.00	Read step counter
0012	STB	D 15.60	Display the step
0013	EP		

7 ANALOG VALUE PROCESSING

7.1 AIM 30/31 Analog Input Modules

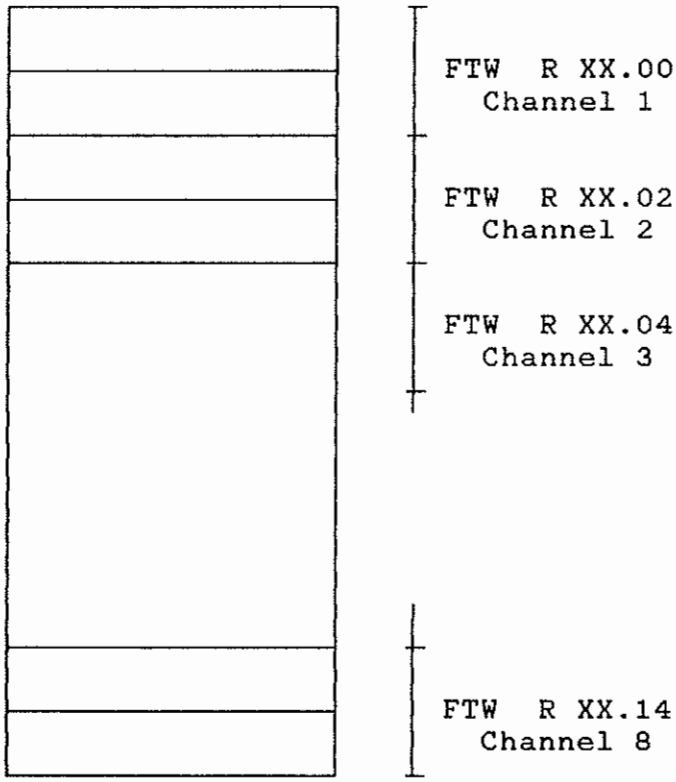
The analog input modules convert analog quantities (voltages, currents, etc.) into digital information. This digital information is copied into the RAM memory of the module during the I/O phase. In this way, the data at the input remains constant during a program cycle which is what is wanted of an "input".

Functional principle:



The RAM memory represents the expanded data range. This means that the value can be read by accessing the R-register for the module slot concerned.

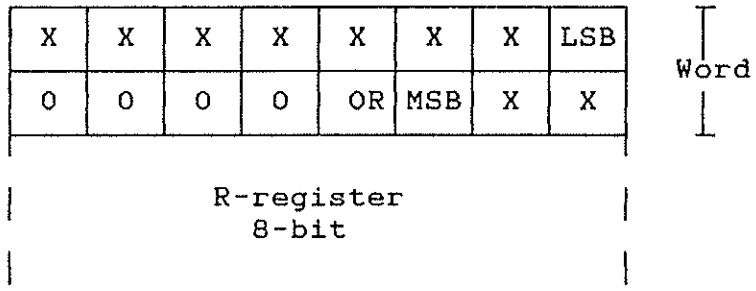
Register/channel assignment:



Note: In **differential** measurements, the converted data is stored in the word register, R, having the lower of the addresses of the channels concerned.

Diff. channel 1 - FTW R XX.00
Diff. channel 2 - FTW R XX.04
Diff. channel 3 - FTW R XX.08
Diff. channel 4 - FTW R XX.12

The ADC converts the input into 11-bit data. A 16-bit word register is assigned to each channel.



LSB: least significant bit
MSB: most significant bit
OR: over-range

The analog input is converted into "offset binary" code:

Smallest measurement - 0 (eleven times '0')
Largest measurement - 7FF (eleven times '1' = 2047 decimal)

If the 12th. bit is set (OR), this means that the input signal lays outside the measurement range. The highest four bits of the word register are always '0'.

Program example:

The converted input from the first channel has to be shown on the display in decimal form. Additionally, the output 05.00 should be set if the input signal exceeds the measurement range.

Line No.	Operation	Operand	Remarks
0000	L	M 40.00	If marker = logical 1
0001	FTW	R 03.00	Read R-register (AIM 30)
0002	STW	D 15.60	Write to display
0003	BID		Binary/decimal conversion
0004	GT	K 02047	In case input exceeds measurement range
0005	=	O 05.00	Then output
0006	EP		

The operating modes

The various module settings are described in Section 3 (Hardware description). The programming in the two operating modes will now be described in this section.

Automatic multi-channel operation
The channels are cyclically interrogated and the input converted in this mode. The required data can be read-in with the command: FTW R nn.mm, where nn is the module slot and mm is the channel address.

Single/multi-channel operation with change-over by software
In time-critical supervision applications, the channel can be defined that should be converted. Any fluctuation in the input signal can be recognized more quickly in this way.

The required channel can be defined with the special command: STB R nn.mm, where nn is the module slot and mm is the channel address (00, 02, 04,...14).

Multi-channel operation is implemented with the command:
TB R nn.16

Program example:
In a machine-control application, only the channel having the address 04 should be considered at process step 14 (step counter 00). The output 05.00 should be set if the value exceeds 1000.

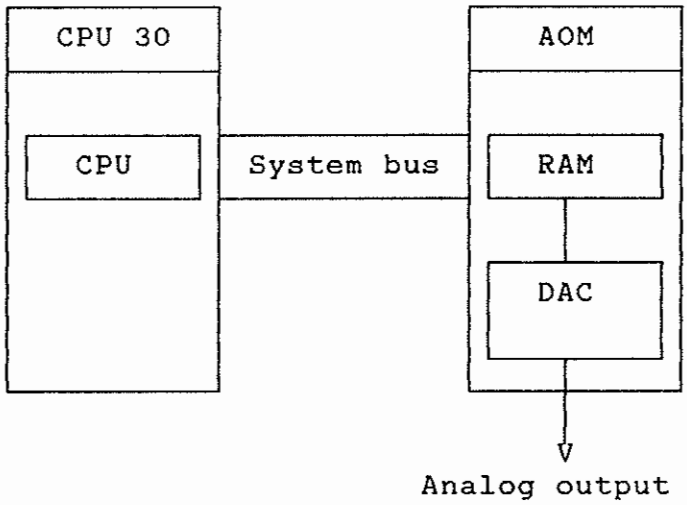
A return to normal multi-channel operation should be made in the following step.

Line No.	Operation	Operand	Remarks
0122	L	S 00.14	If step = 14
0123	STB	R 03.04	Then single channel (04)
0124	FTW	R 03.04	Measurement value
0125	GT	K 01000	In case measurement value exceeds 1000
0126	=	O 05.00	Then output
0127	L	S 00.15	If step = 15
0128	STB	R 03.16	Then multi-channel op'n

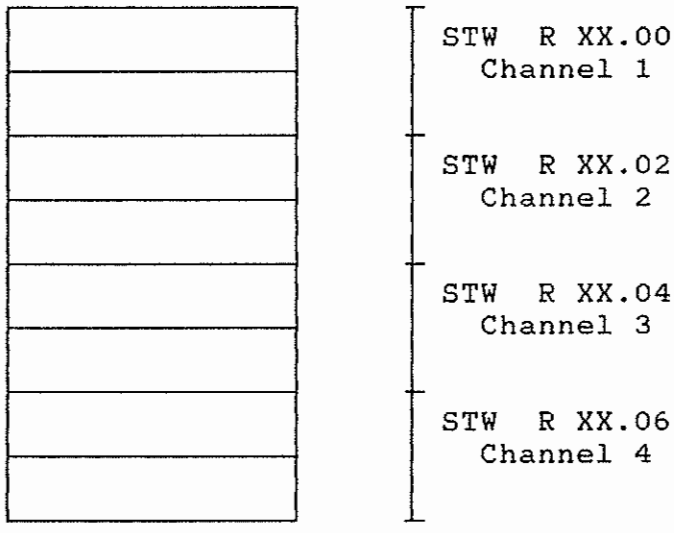
After switching on the power supply or after switching to another channel, the module contains unusable or old data. The module only delivers correct information after the first conversion period. The program must take this into account, for example, by introducing suitable delays.

7.2 AOM 30/31 Analog output modules

The analog output modules convert digital data into analog quantities such as voltages, currents, etc. The digital values are written to the the R-register for the corresponding module slot. Conversion to analog quantities takes place immediately, i.e. while the user program is running.



Register/channel assignment



The DAC converts a 12-bit word into the equivalent analog value.

Range of values: 0000 ... 0FFF (hex)

The following conversions relate to the decimal system. A calculated decimal value therefore has to be converted to a binary value before it can be fed to an analog output module by an STW R XX.XX instruction.

The DEB (decimal/binary) operation carries out the conversion.

D Digital value	Analog output quantity 0...+10V	0...+20mA	Analog output quantity −10...+10V	-20...+20mA
0000_{10}	0.0V	0.0mA	-10.0V	-20.0mA
2048_{10}	+5.0V	+10.0mA	0.0V	0.0mA
4095_{10}	+10.0V	+20.0mA	+10.0V	+20.0mA

The values are calculated as follows:

Range 0...+10V : $D = 409.5 \times V$ (AOM 30)

Range 0...+20mA : $D = 204.75 \times I$ (AOM 30)

Range -10...+10V: $D = 204.75 \times (V+10)$ (AOM 31)

Range -20...+20mA: $D = 102.375 \times (I+20)$ (AOM 31)

D = digitial value
V = output voltage in V
I = output current in mA

Program example:

When the input 00.00 is not set, a value of +3V should be generated by channel 1 of the analog output module in slot 7 and channel 2 of the same module should generate a value of +4mA.

Ranges: Channel 1: 0...+10V; Channel 2: 0...+20mA

Channel 1: $D = 409.5 \times V = 409.5 \times 3 = 1228_{10}$

Channel 2: $D = 204.75 \times I = 204.75 \times 4 = 819_{10}$

Line No.	Operation	Operand	Remark
0000	LN	I 00.00	If input is not set
0001	FTW	K 01228	Read constant
0002	DEB		Decimal/binary conversion
0003	STW	R 07.00	Write to channel 1
0004	FTW	K 00819	Read constant
0005	DEB		Decimal/binary conversion
0006	STW	R 07.02	Write to channel 2
0007	EP		

8 COMMUNICATION

Several SELECONTROL® PMC controllers can be connected together in a network by use of the communications controller, TCC 30. In such a network, one TCC 30 is always defined as the master to which up to eight slaves may be connected. Data transfers between the communications controllers is fully autonomous and imposes no additional burden on the programmer.

The individual TCC 30 modules exchange various data blocks via the R-registers. The definition of the function of each of these blocks is fixed (see following table). A difference is made between R-registers for incoming and outgoing data. They are also assigned to the respective controller (master or slave). The programmer therefore only has to write the task for the required data transfer in the transmitted data block and to read the appropriate data block from the received data.

Data transfers are always between master and slave. Direct communication between slaves is not possible.

Program example

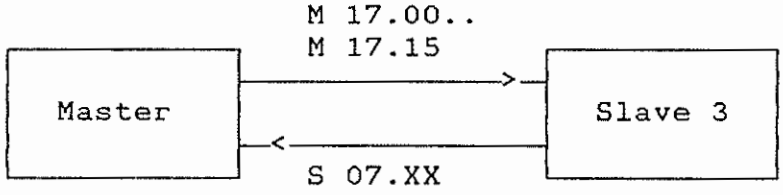
A network of two SELECONTROL® PMC's is assumed:

- 1 master
- 1 slave (slave number 3)

The communication tasks are:

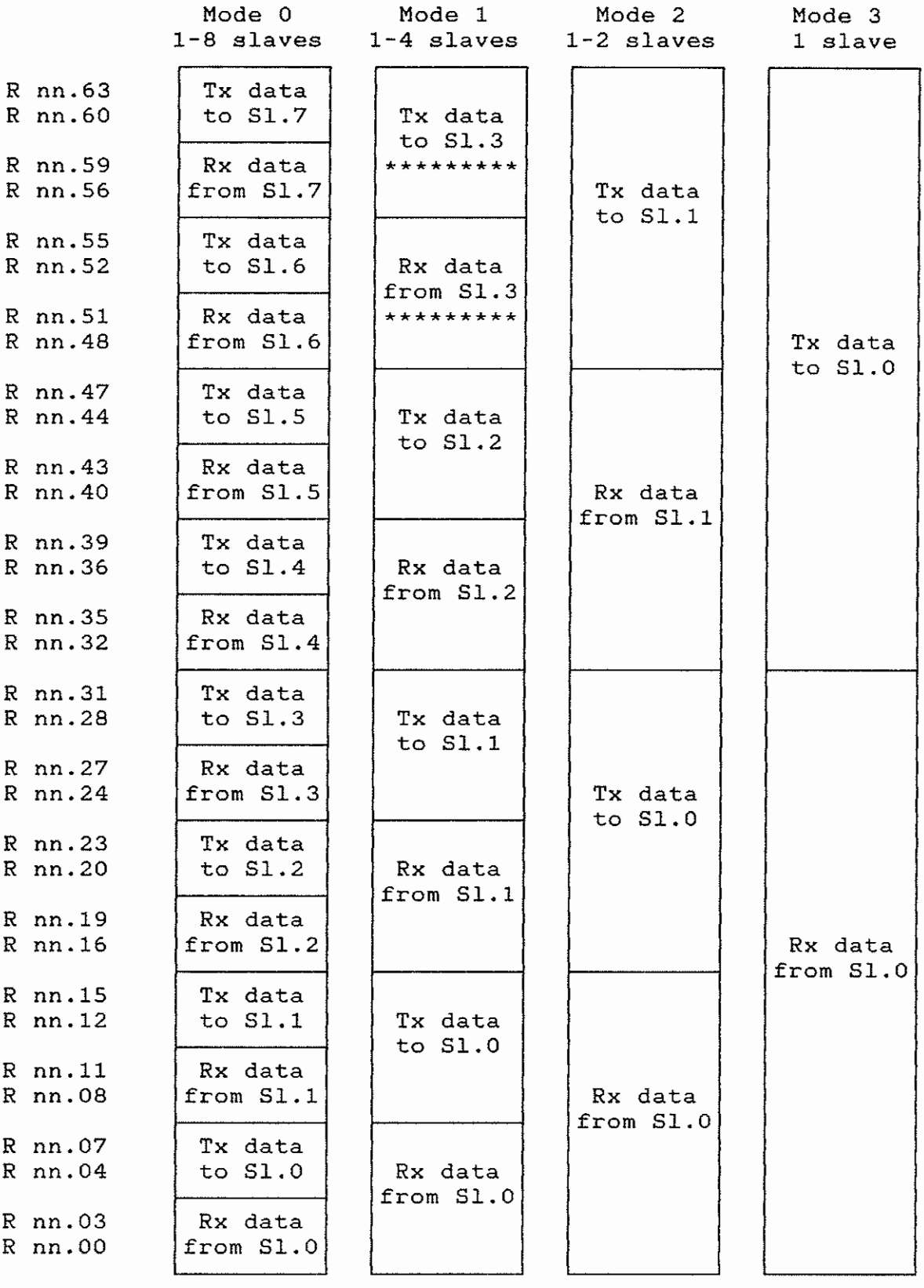
- The contents of step counter 7 in slave 3 has to be shown on the display of the master

- Outputs 04.00 to 04.15 in the slave shall have the same state as markers 17.00 to 17.15 in the master

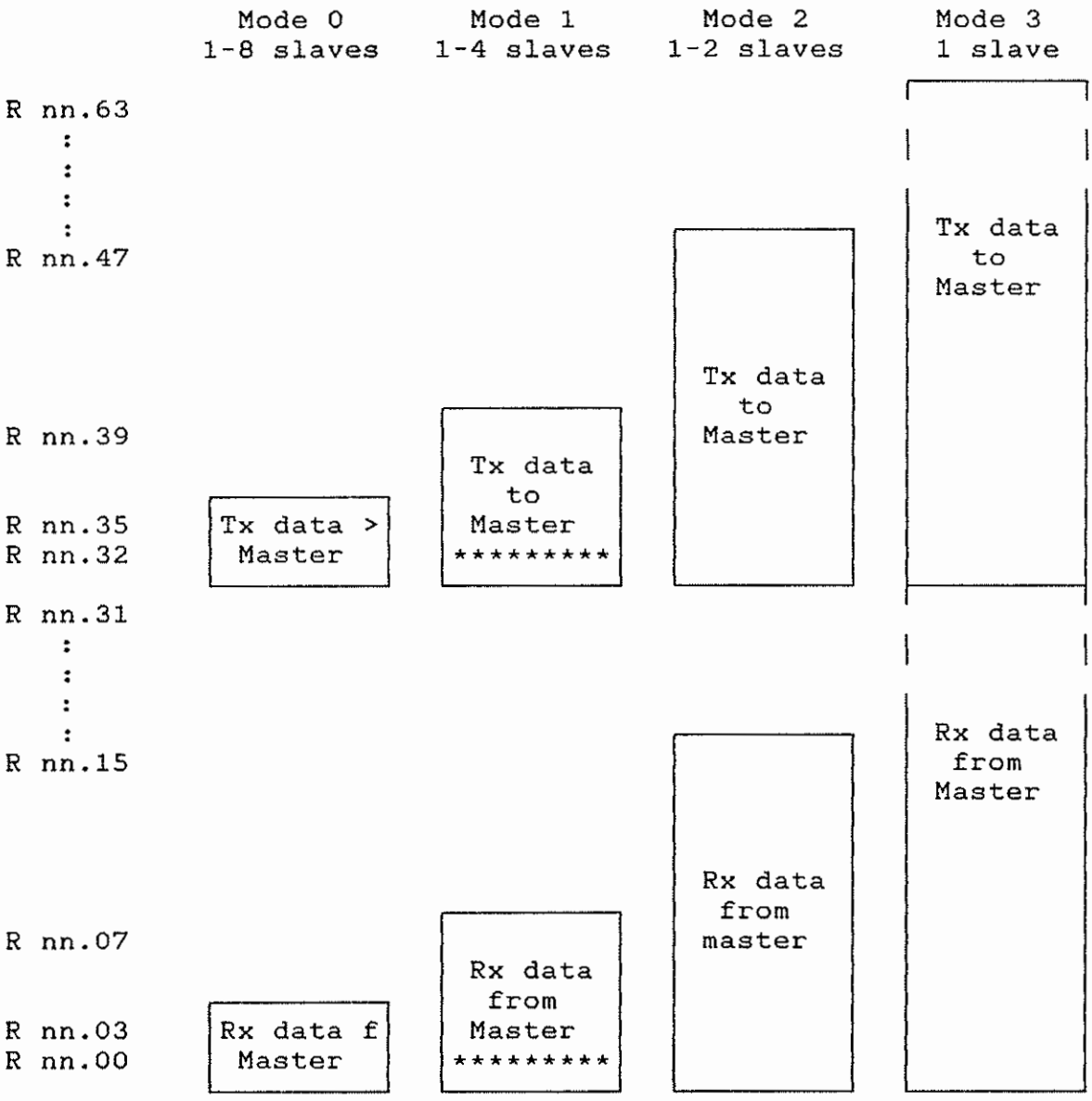


The registers used in this example are identified in the following table by ***.

Data format in the master:



Data format in the slave:



Master program:

Line No.	Operation	Operand	Remarks
			Markers M 17.00 to M 17.15
0100	L	M 40.00	Logical 1
0101	FTB	R nn.48	Read slave 3 data (SC)
0102	STB	D 15.60	Output for display
0103	FTW	M 17.00	Read 16 markers
0104	STW	R nn.56	Data output to slave 3 (16 markers)
0105	EP		

nn = module address in master TCC 30

Slave 3 program:

Line No.	Operation	Operand	Remarks
0050	L	M 40.00	Logical 1
0051	FTB	S 07.00	Read SC 7
0052	STB	R xx.32	Output data to master (SC status)
0053	FTW	R xx.00	Read data from master
0054	STW	O 04.00	Output to O 04.00 - 04.15
0055	EP		

xx = module address in slave 3 TCC 30

Communication supervision

Besides the R-registers, the TCC 30 also has 16 inputs in the I/O range. These 16 signals are sub-divided into 8 toggle flags and 8 error flags (master). The programmer can use these aids to follow and supervise data exchanges between the various pieces of equipment.

The toggle flag
When a data packet is exchanged between the master and a slave, the corresponding toggle flags in both units are complemented. This status change can be detected by the user program and the exchange can thus be recognized.

The error flag
The error flag is set to logical "1" for approximately 250ms if there is a transmission error, i.e. the respective partner unit has received erroneous data (in such cases, the toggle flag is not complemented).

Meaning of the error LED's
When in operation as the master, a brief lighting of a LED indicates that a transmission error has occurred. The LED number (Error 00 ... 07) corresponds to the address of the slave concerned.

A transmission error can also occur while sending a data packet to the master when in operation as a slave.

Memory map of the toggle and error flags

Master:

I nn. 15 - Error slave 7
I nn. 14 - Error slave 6
I nn. 13 - Error slave 5
I nn. 12 - Error slave 4
I nn. 11 - Error slave 3
I nn. 10 - Error slave 2
I nn. 09 - Error slave 1
I nn. 08 - Error slave 0
I nn. 07 - Toggle slave 7
I nn. 06 - Toggle slave 6
I nn. 05 - Toggle slave 5
I nn. 04 - Toggle slave 4
I nn. 03 - Toggle slave 3
I nn. 02 - Toggle slave 2
I nn. 01 - Toggle slave 1
I nn. 00 - Toggle slave 0

Slave:

I nn. 08 - Error
I nn. 00 - Toggle

9 INSTALLATION, CABLING AND COMMISSIONING

9.1 Commissioning for programming

It is recommended that the following points be adhered to exactly during the first commissioning and programming.

1) Assemble the system modularly.
   The modules can be inserted in arbitrary slots.
   The following system configuration is recommended for programming the examples given in this handbook:

   DIM 30 : slots 00 and 01 ATM 30 : slot 06
   DOM 30 : slots 04 and 05 PSM 30 : slot 07

   - Insert the modules in the back panel unit
   - Place the CPU in the first slot on the left
   - Insert other modules as appropriate
   - Screw modules firmly into place

2) Connect the simulator
   - Remove the connector strip
   - Insert the simulator in place of the connector strip

3) Wire-up the 24V dc system supply
   - Connect the CPU 30 to +24V and 0V
   - Connect the DIM 30 to +24V anf 0V, possibly via simulator
   - Connect the DOM 30 to +24V and 0V
     (The LED's at the outputs only light when the system supply is connected)

4) Connect the central processing unit and the programming terminal together using data cable CCA 3x

5) Insert the RAM board into the CPU
   - The space is occupied by a factory-installed EPROM board
   - A RAM board is necessary for programming
   - Replace EPROM board (ROM 30) by RAM board (RAM 30)

6) Switch on 24V dc system supply
   - The LCD display of the PMC lights up

7) Turn operating mode switch to "PROGRAM" mode (9 o'clock)

8) Switch on the programming terminal and initialize it for communication as given in the terminal handbook
   - See the section 'Programming systems' if using an NEC 8201A terminal with a SELECTRON software package
   - The terminal shows "00000" when ready

9) Program

9.2 Installation

You can select the form of installation that is most suitable for you.

Rear-panel mounting

The following back panel units are available:

- BPU 08 : Back panel unit for
  - Central processor unit and
  - 8 modules of choice

- BPU 12 : Back panel unit for
  - Central processor unit and
  - 12 modules of choice

- EPU 08 : Expansion unit
  For use with the BPU 08 for the maximum system configuration of up to 16 modules

19" rack mounting

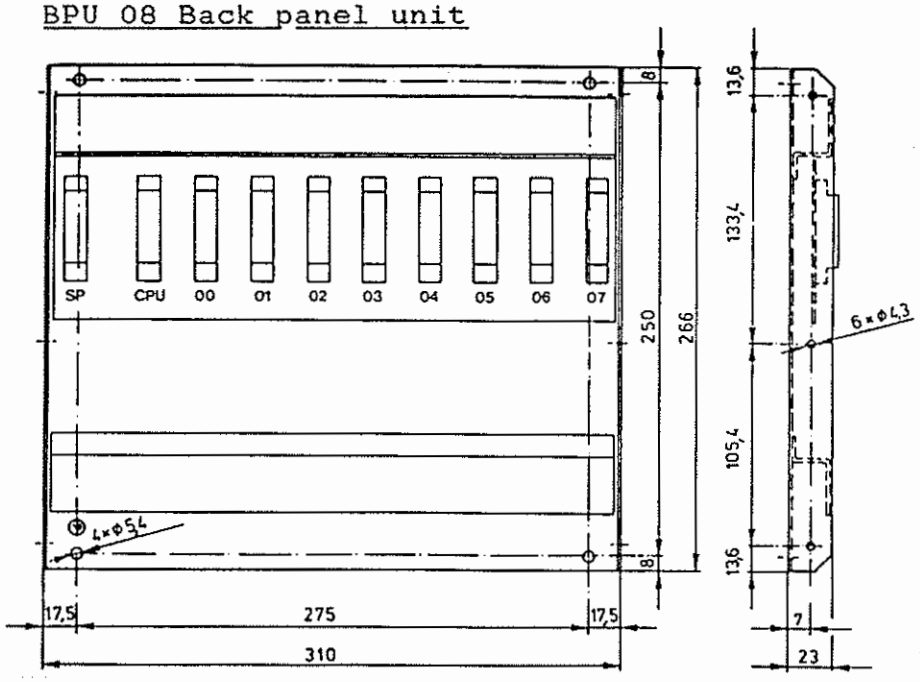
The following material is available:

- BPU 12 : Back panel unit for
  - Central processor unit and
  - 12 modules of choice

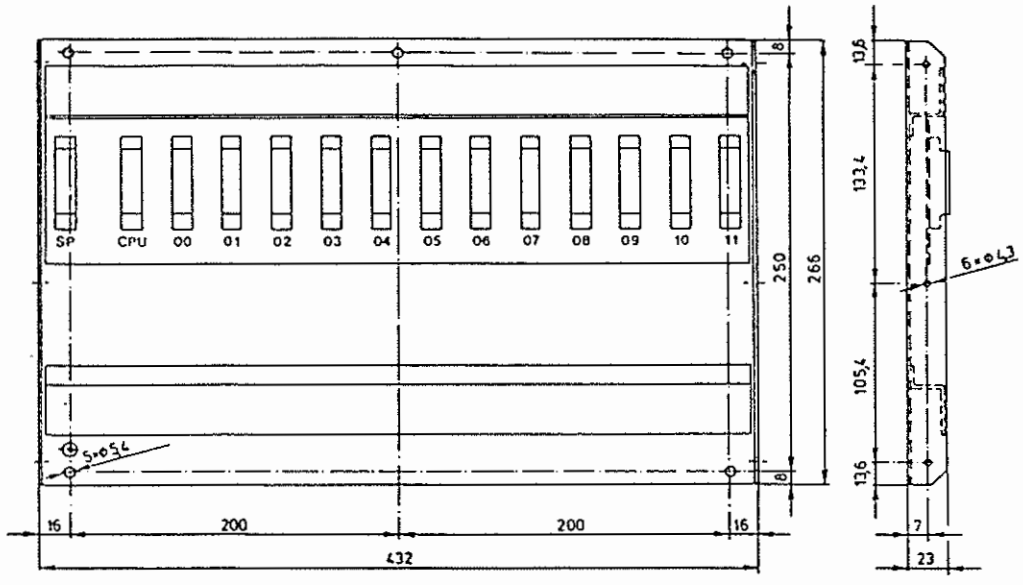
- RMA 12 : Rack-mounting accessory
  Accessory kit for installing a BPU 12 in a 19" rack

9.3 Dimensions

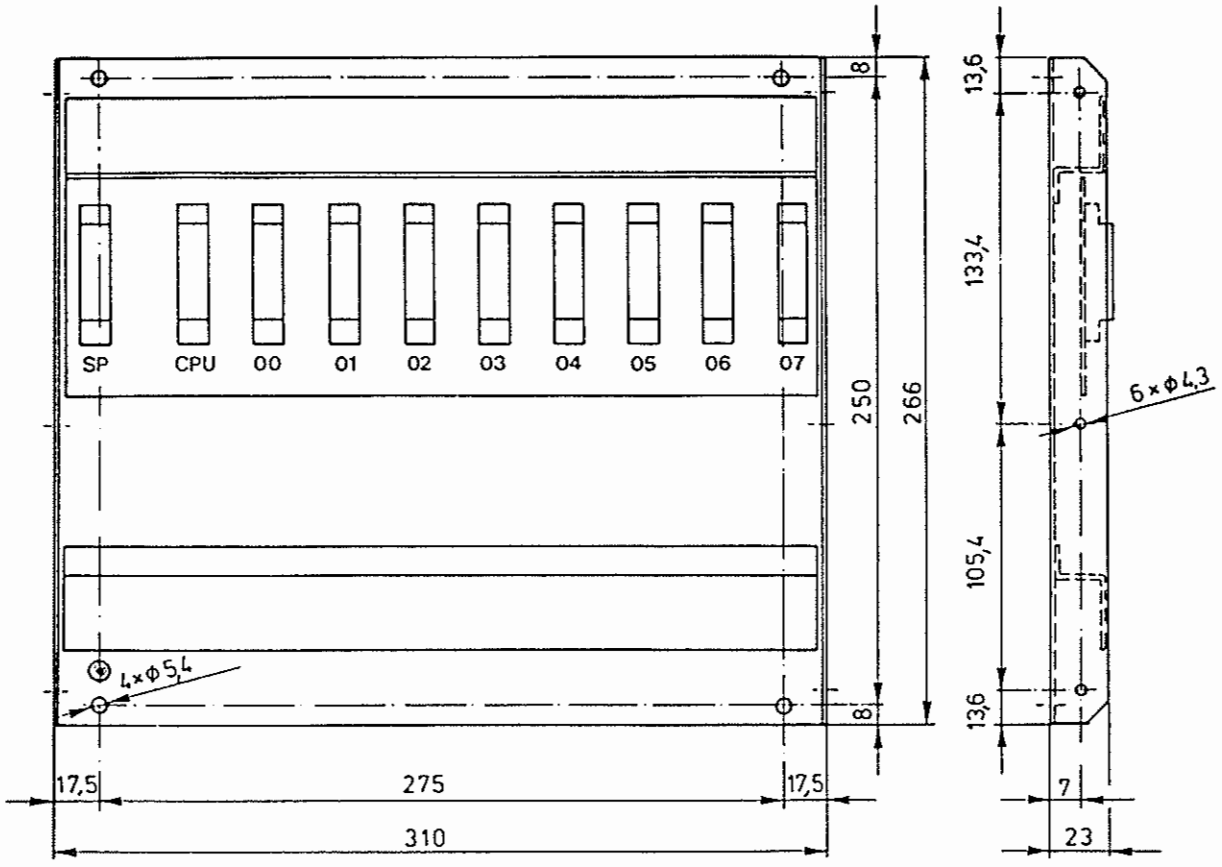
BPU 08 Back panel unit

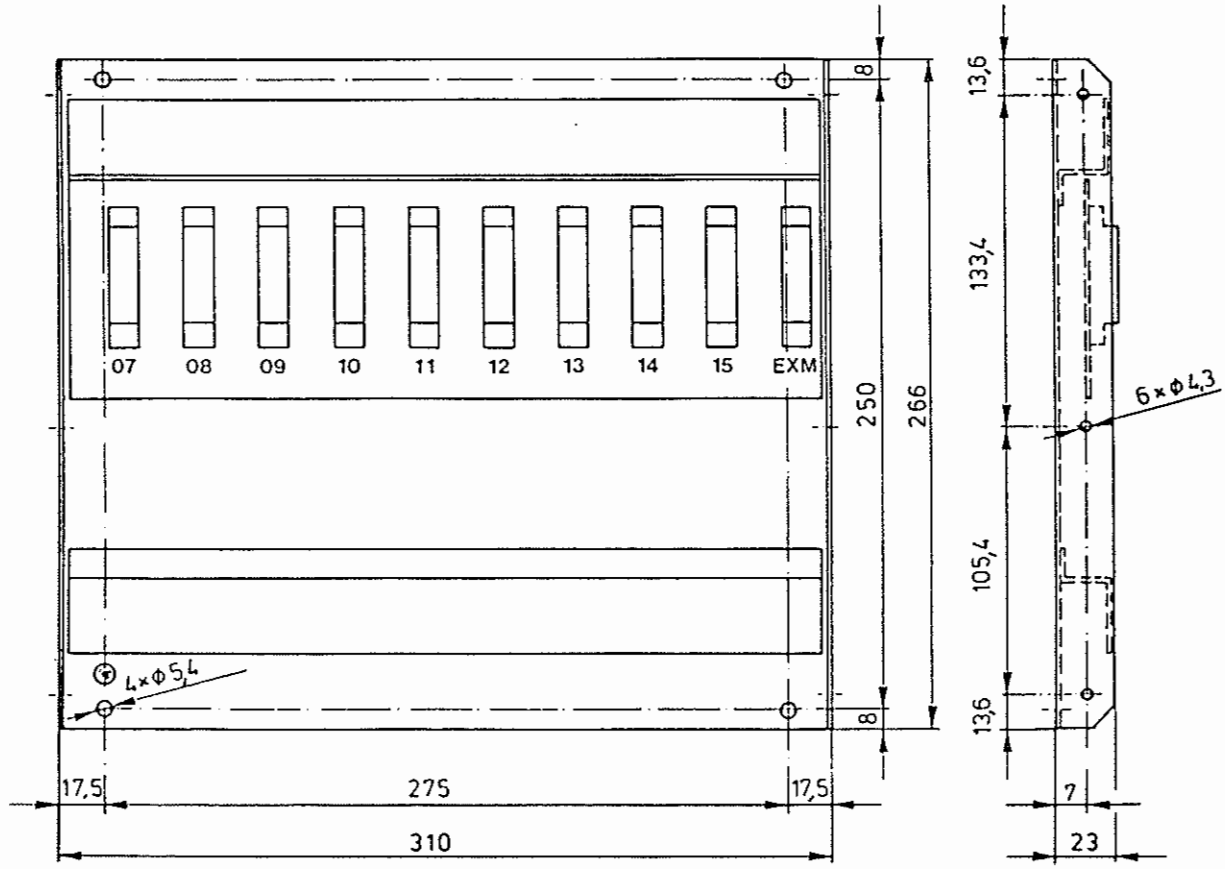


BPU 12 Back panel unit



BPU 08 and EPU 08 Back panel unit and Expansion unit





9.4 Cabling

The connection technique adopted with front mounted, plug-in terminals enables the cabling to be carried out logically.

The inputs and outputs can be wired with 1mm² stranded wire.

> Because of the electrical isolation provided, each module can be wired to a different supply.

The supply voltage can be daisy-chained across from one module to another. Two terminals are provided per module for the +24V and 0V connections. These connections are internally linked.

The connection diagrams for the individual modules are to be found in the section 'Hardware description'.

9.5 Commissioning (On-line operation)

The SELECONTROL® PMC can be switched to the RUN-MONITOR mode for commissioning purposes.

The RUN-MONITOR mode enables data registers (counters and timers), step counters, markers, R-registers, inputs and outputs to be controlled and influenced via the programming system.

It should be noted, however, that the program cannot be altered in this on-line mode. Only the running of the program can be changed by altering the various parameters in the operands mentioned above.

This possibility provides on the one hand optimum convenience during commissioning while on the other hand guaranteeing the highest security.

Uses:

- Checking-out the program that has been developed
- Commission the system controlled by the SELECONTROL® PMC
- Supervision of the process execution.

Display of process data

Operating command: V) (VIEW)

The required operand can be displayed by using the command: V).

V) Operand

The following types of data can be displayed in this way:

```
I nn.nn   (binary status)
O nn.nn   (binary status)
M nn.nn   (binary status)
```

(The special markers, M 40.nn, cannot be displayed)

```
S nn.00   (step counter status, 2 digits)
D nn.nn   (data register content, 4 digits)
R nn.nn   (extended data range, 4 digits)
```

The content of an R-register can only be displayed if a digital input module DIM 30/31 or an analog input module AIM 30 is inserted in the corresponding module slot.

Example:

```
R V)I00.08
00001          Status 1 "logical 1" from input I 00.08 is shown

R V)S10.00
00047          Step 47 of SC 10 is shown to be active

R V)D15.60
07385          Content of data register D 15.60 is shown as 7385
```

The display of the content of the operand specified does not just show the status at the moment it was called but shows a continuously up-dated display.

Striking the >ENTER< key terminates the display and enables a new operand to be called.

Modifying the process data

Operating command: M) MODIFY

An operand can be modified with the M) command:

M) Operand Constant kkkkk

The following types of data can be modified in this way:

```
I nn.nn K 1/0   (setting to logical 1 or 0)
O nn.nn K 1/0   (setting to logical 1 or 0)
M nn.nn K 1/0   (setting to logical 1 or 0)
```

(The special markers, M 40.nn, cannot be altered)

These "non-memorized" types of data (I/O/M) retain the modified status only during the following cycle.

Thereafter, the effective status is restored again during the next I/O phase.

Only one operand at a time can be modified with M).

```
S nn.00 K kk    (setting to step kk)
D nn.nn K kkkk  (setting to value kkkk)
R nn.nn K kkkk  (setting to value kkkk)
```

The content of an R-register can only be displayed if a digital output module DOM 30/31 or an analog output module AOM 30 is inserted in the corresponding module slot.

The "memorizing" types of data (S and D) retain their modified status until changed by the running program (assuming the program is so constructed) or by another modify command setting a different value.

Example:

```
R M)I00.08K1
R                Input I 00.08 is set to logical 1 for the
                 duration of the next pgogram cycle

R M)S10.00K47
R                Step counter 10 is set to step 47

R M)D15.60K7385
R                The content of data register D 15.60 is
                 set to a value of 7385
```

Useful tip:

Often, constant values (constants K kkkk) have to be used in a program that can only be defined during practical operation. Typical examples are the digital timers.

It is recommended to put these times into data registers during commissioning instead of setting them as constants so that they can be adjusted as easily as possible if necessary.

Program example:

Instead of a digital time set as a constant like this:

```
L    I 00.00    (Drive)
FTW  K 00060    Preset time - 60 secs
TS   D 15.60    Clock pulse - 1 sec
```

the following program part could be selected:

```
L    I 00.00    (Drive)
FTW  D 00.00    Preset time register
TS   D 15.60    Clock pulse - 1 sec
```

In this way, the optimum preset time value can be found during commissioning by the "successive approximation" technique using M) D00.00 K kkkk to load the preset time register. As soon as the program is proved and commissioning is finished, the definitive preset time value can be written-in as a constant.

10 SYSTEM CHARACTERISTICS

This section deals with the system characteristics during:

- Commissioning (powering-up the system)
- Interruption of the supply
- System fault

10.1 System characteristics during commissioning

When the power is first switched on, all the outputs, counters, timers and registers are set to 0 (zero).

10.1 Interruption of the power supply

Brief interruptions in the power supply of up to 30ms have no effect on the operation of the controller. The program continues to run when the power returns.

The special marker, M 40.01 (1st. cycle) is not set.

If the power interruption is longer, the current program cycle will continue to its end.

Thereafter, the current process data is stored in the data memory of the CPU.

A warm or a cold start can be set by software.

Unless programmed otherwise, a cold start is always performed.

10.2 Power supply interruption

Short power interruptions up to 30 ms do not influence the operation of the system. The program continues normally after the power returns.

The special marker 40.01 (1 cycle) is not set.

The current program cycle is still completed even in the event of a longer power supply interruption.

Thereafter, the current process data is stored in the remanent data register (D 00.00 ... D 15.63).

> Warm start or cold start can be set by software.
>
> If no warm start is programmed, a cold start always happens.

Cold start

All outputs, counters, timers and registers are set to 0 (zero) when a cold start is made.

Warm start

In the event of a power interruption, the content of the data registers, D 00.00 ... D 15.63 , (i.e. the process data) remains stored. The content of these registers is then used to continue running the program once power has been restored.

A partial set or reset can be performed under program control in the first program cycle (by reference to the special marker M 40.01)

> The program carries on after power has been restored from the point that it was at when the power failed.

No warm start is possible under the following circumstances:

- If the watch-dog has been triggered
  (FI-LED on the CPU alight)

- After data loss due to power failure
  (System fault E-09 shown on the LC-display)

10.3 System fault

The system supervises the length of the cycle.

The WATCH DOG reacts if the **cycle time exceeds about 100ms** and

- resets the CPU
- switches off the outputs
- lights the FI-LED.

After this, only a cold start is possible

When the cycle time exceeds 50ms, the time-carry for the "TF" TIMER FAST will be erroneous but the controller continues to run.

The special marker, M 40.08, flags this condition.

11 ANALYSIS AND FAULT DIAGNOSIS

The SELECONTROL® PMC shows what is happening or why something is not happening.

Process sequences, counter statuses, error messages and much more can be shown on the freely programmable LCD display of the SELECONTROL® PMC.

These diagnostic possibilities mean that production stoppages, and down-times can be reduced to a minimum.

The sequence display together with the LED interface display even make remote diagnostics possible.

If a case arises where the correct condition to enable the program to go on to the next step is not satisfied (e.g. a limit switch has not been tripped), the sequence will halt at the corresponding place. The problem can be quickly localized because the step counter status can be displayed.

Experience has shown that about 95% of all faults occur externally to the controller in automated installations.

This fact is due to the considerably heavier demands placed on encoders, actuators, drives and cabling, etc.

It therefore becomes obvious to use the central processor unit directly for sequence analysis and fault diagnosis.

The SELECONTROL® PMC will even show system faults within itself e.g. a fault in the central unit, the memory or other internal problems (see Section 3, Hardware description: central processing unit).

The availability of the installation as a whole is thus considerably increased even further.