AL 2000S/P
User Guide
AL2000S V9941
System Program
v. 3.07

FF-AUTOMATION OY

HEAD OFFICE

Eräkuja 2
01600 VANTAA
Tel. +358-9-5306310
Fax. +358-9-53063130
http://www.ff-automation.com
e-mail:info@ff-automation.com

SERVICE CENTER

Meijerikuja
37650 Valkeakoski
Tel. +358-3-5846390
Fax. +358-3-5846711

AL 2000S/P
User Guide
AL2000S V9941
System program
v. 3.07

FF-AUTOMATION OY

HEAD OFFICE

Eräkuja 2
01600 VANTAA
Tel. +358-9-5306310
Fax. +358-9-53063130
http://www.ff-automation.com
e-mail:info@ff-automation.com

SERVICE CENTER

Meijerikuja
37650 Valkeakoski
Tel. +358-3-5846390
Fax. +358-3-5846711

TABLE OF CONTENT

FF-AUTOMATION OY

HEAD OFFICE

Eräkuja 2
01600 VANTAA
Tel. +358-9-5306310
Fax. +358-9-53063130
http://www.ff-automation.com
e-mail:info@ff-automation.com

SERVICE CENTER

Meijerikuja
37650 Valkeakoski
Tel. +358-3-5846390
Fax. +358-3-5846711

1. GENERAL DESCRIPTION

1.1 Introduction

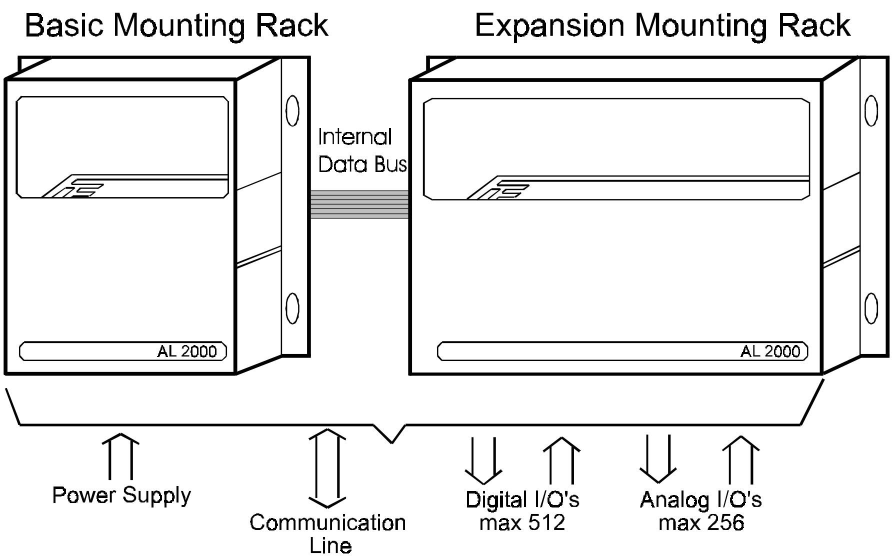
The AL2000 is a new modular PLC from FF-Automation, Finland. Due to it's modular design, different hardware configurations can easily be made for specific automation and data acquisition requirements. The AL2000 is designed to be used in medium and large scale applications.

The AL2000 can be easily programmed using an ordinary PC and the ALProWin programming software. ALPro has an extensive instruction language for PLC programming, with over 260 instructions available. Programming difficult procedures is made easy by the many instructions which have been developed for specific applications. The ALPro EIL (Extensive Instruction Language) is a highly efficient programming tool which is both easy to learn and simple to use.

For programmers who are familiar with ladder logic programming, ALProWin introduces a new and efficient way to write PLC programs. However, for programmers who prefer to work with ladder logic, an additional module is available for this purpose.

The AL2000L provides versatile communication capabilities via serial interface or Ethernet LAN.

AL2000 has been designed giving special attention to ease of use and rapid service. A thorough Self Diagnostics Function (SDF) continuously supervises the AL2000's operation. In the event that a fault in the system is discovered by the SDF, the user is immediately informed. A fault condition is displayed on the module by means of an LED, and sent via serial interface or Ethernet LAN to a supervising PC (if the user so chooses). This makes servicing the AL2000 both easy and rapid.



The AL2000 is versatile enough for the most demanding of applications!

The AL2000 can be used to realise a fully distributed control system. Remote units can be connected to a central AL2000 unit via a fast serial interface. It is only necessary to program the central AL2000 unit with all remote units operating as simple I/O units, however, for more demanding situations local programs can be written to the remote units. Typical applications for the AL2000 are:

* The Pulp and Paper Industry
* The Steel Industry
* Waste Water Treatment Plants
* Pumping Stations
* Saw Mills
* Regional Control Systems
* Remote Control Systems
* Water Treatment Plants
* Machine Manufacturing
* The Chemical Industry
* Building Automation
* Machine Control

The AL2000 together with control software such as 'PARAGON' or 'FCS', opens up unlimited opportunities for SCADA (Supervisory, Control And Data Acquisition) applications for every industrial control situation. The following is a summary of the advantages offered by the AL2000 system:

* Savings in mounting space due to the construction
* Flexibility of hardware configuration - You only buy what you need
* Robustness - the binary outputs are overload and short circuit protected
* Reliability - natural cooling, requires no fan
* Versatile communication using RS-232C, RS-422/485 serial interfaces and Ethernet LAN
* Powerful PID control capabilities
* Rapid "key hole" installation
* Easy service - efficient self diagnostics and detachable screw terminals
* I/O cables connect directly to each I/O module

Setting up the AL2000 system for a particular application is a simple and methodical procedure as outlined below:

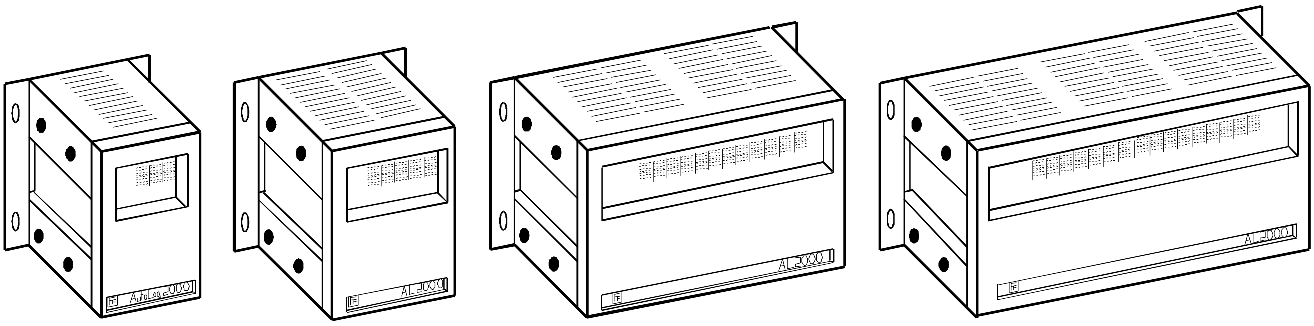
1. Determine the type of CPU-module, power supply units and the number and type of I/O modules required
2. Select the number and size of mounting racks (allowing for possible future expansion)
3. Select the length of flat cable required to connect the expansion mounting rack (if used)
4. Use the ALPro`Win software to define the hardware configuration (number and type of I/O modules and their respective slot numbers)
5. Select the communication protocol (e.g. MODBUS RTU)
6. Install the AL2000
7. Design and write the PLC program
8. Transfer the PLC program from the PC to the AL2000
9. Test the PLC program and correct possible errors
10. Save the PLC program onto disk
11. Make documentation for the PLC program (ALProWin can help with this)

1.2 Basic Design

The AL2000 is a modern PLC designed for general purpose industrial automation and data acquisition. The AL2000's modular design allows each mounting rack to be fitted with the CPU and I/O modules which best meet the specific demands of each application.

1.2.1 Mounting Rack

Five different sizes of mounting rack are available, with space for 3, 5, 11 or 16 I/O modules. All mounting racks are delivered with mother boards that connect the different I/O modules to the CPU. The modular design enables easy and rapid installation of different hardware configurations. The I/O cabling is connected to detachable screw terminals on the front edge of each I/O module. All I/O modules can be removed (or re-positioned in the rack) without disconnecting the I/O cables. The front cover of the unit hides the screw terminals but allows the LED indicators (indicating the status of each I/O) to be seen through a window.



1.2.2 Power Supply

Each mounting rack (except MR3 models) must be fitted with a power supply module and a power connecting module. The power supply cable is connected to the power connecting module. The power supply module provides the AL2000 with the voltages it needs. The MR3 includes a single-unit power supply and does not require separate power connecting and power supply modules.The MR3-mini does not require a power supply, power for this mounting rack come from the CPU2000P.

1.2.3 Internal Bus

The internal bus handles communication between the I/O modules and the CPU. When an expansion mounting rack is used, the internal bus is extended from the basic mounting rack to the expansion mounting rack using a flat cable. The internal bus provides fast communication and is designed to IEC-Standards IEC47B (CO8) for electrical features and (CO10) for mechanical features. Each I/O module is connected to the mother board using DIN41612 connectors. The expansion mounting rack does not require its own CPU since the internal bus is extended to serve the expansion mounting rack via a flat cable. The basic mounting rack and the expansion mounting rack (if fitted) are hereafter collectively referred to as the Basic Unit.

1.2.4 The CPU2000S Series Modules

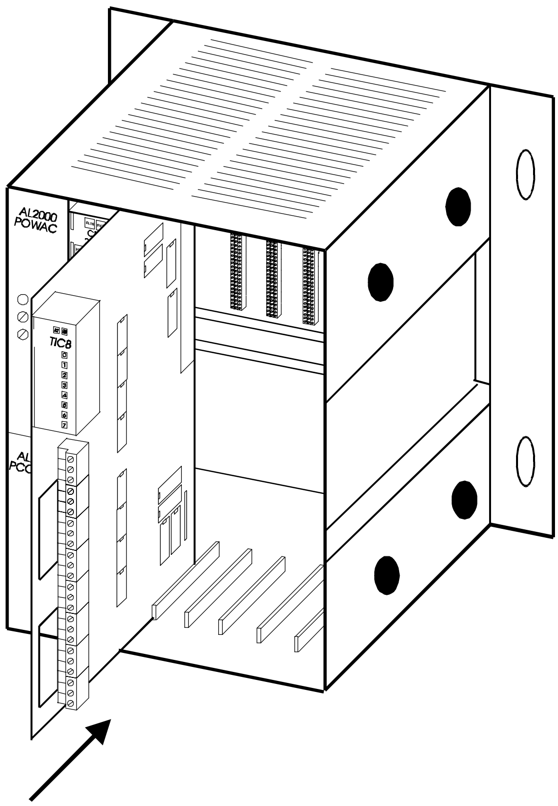
The CPU2000S is the basic general purpose CPU-unit for the AL2000. This CPU can control the basic unit which can house a maximum of 32 I/O modules. This basic unit can consist of one basic mounting rack and one expansion mounting rack. All CPU modules are equipped with three serial interface ports and an I^2C interface port for connection of a keypad/display unit. The CPU2000P is equipped with own power supply unit.

1.2.5 The CPU2000L Module

The CPU2000L is a powerful CPU based on the pentium 586 and is entirely PC compatible. The CPU2000L is equipped with two serial interfaces, an I^2C interface port for connection of a keypad/display unit, parallel printer port, Ethernet LAN and connectors for standard IBM-AT keyboard.

1.2.6 The I/O Modules

There are several modules available for both digital and analog inputs and outputs. These modules are slot-in cards which can be installed in any of the mounting racks and controlled by any of the slot-in CPU modules. All I/O modules are equipped with LED indicators which indicate the status of inputs and/or outputs and any error condition.



2. PROGRAMMING

2.1 Programming the AL2000

The AL2000 can be programmed using an normal PC running Windows 9x or NT and the AlproWin programming software. The AL Extensive Instruction Language (ALEIL) comprising over 260 commands is used for programming the AL2000. Due to the AL2000's versatility, pure ladder logic programming would not make efficient use of all its features, although AlproWin allows the use of ladder logic in addition to ALEIL, AlproWin features both on-line and off-line programming, graphical representation of variables and an excellent tool for documentation of PLC programs. For more detailed information concerning AlproWin please consult the AlproWin User Manual.

2.1.1 Programming with the AlproWin Software

Before programming the AL2000, the hardware configuration must be defined. This is done using the AlproWin software The PLC program can be written using the text editor in AlproWin or using any other word processor or editor and save file in ASCII mode. AlproWin compiles the PLC program for running on the AL2000. PLC programs are saved on disk for later use.

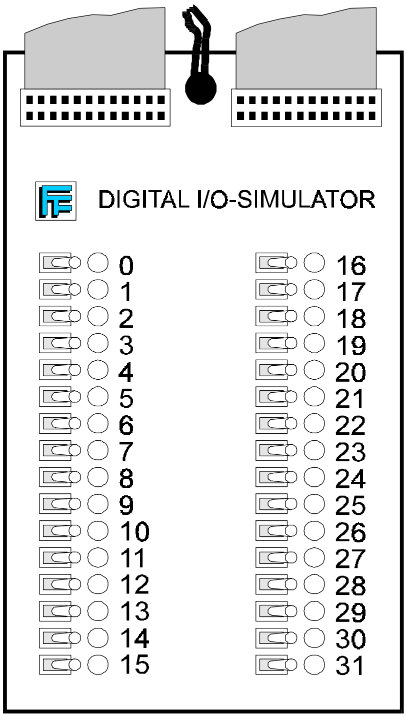
2.1.2 I/O Module Addresses

Each slot in the mounting rack has its own unique address which is automatically recognised by the I/O module installed in that specific slot. Two special registers in the I/O modules are provided for a test procedure done by the CPU-module. This permits the CPU to continuously supervise the function of the internal bus, check that the correct I/O modules are installed in the correct slots etc. The registers in the I/O modules also contain information about the type of the I/O module. This makes it possible for the CPU to fully check the I/O configuration. If an I/O module is accidentally replaced with the wrong type of I/O module (e.g. during repair/service), the CPU discovers and reports the mistake immediately upon power-up.

2.1.3 MODBUS

Using the standard MODBUS RTU protocol, AL2000 units can operate as masters driving other AL2000, AL14 - AL32 and AL20AN slave units, thus enabling large scale and distributed control systems to be realised. This is a standard feature on all AL2000 CPU modules. The MODBUS software included with AlproWin can be used to define the master/slave units and their respective addresses. The MODBUS protocol can also be used to communicate with PLCs made by other manufacturers and all supervisory software currently available.

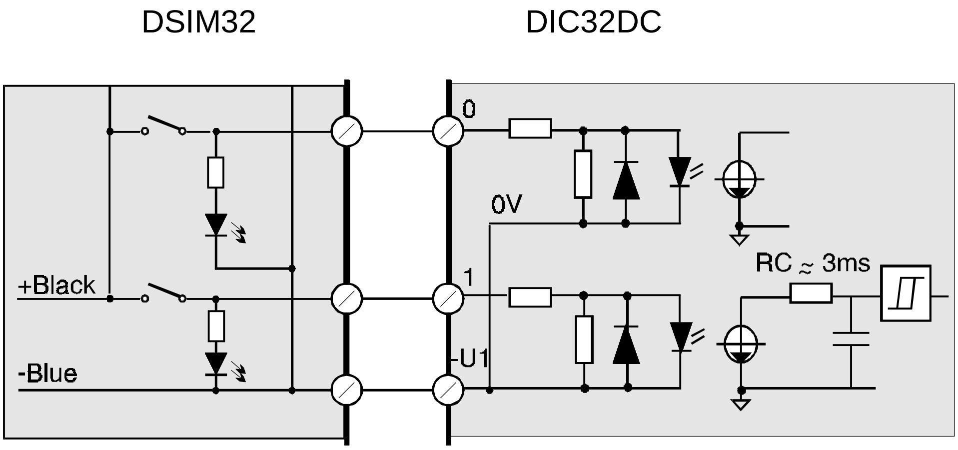
2.2 DSIM32 Digital I/O Simulator



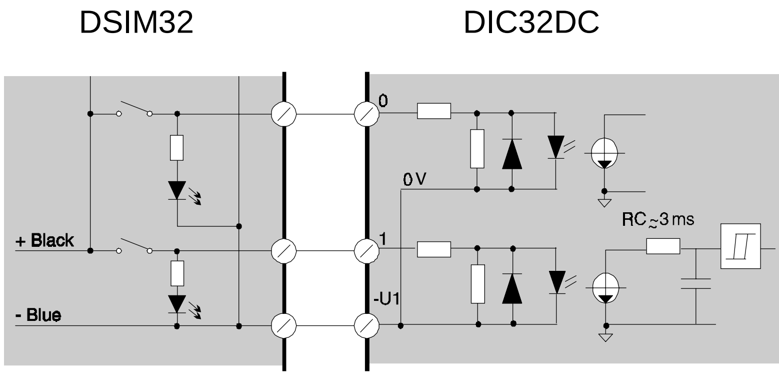
The DSIM32 digital I/O simulator can be used to test and debug PLC programs before they are used in the actual control process. This makes the testing process faster, easier and safer by avoiding the possibility of erronous PLC programs affecting the process. It is strongly recommended that all PLC programs be tested with the DSIM32 prior to use in the control process.

The DSIM32 can be used to simulate up to 32 digital inputs or outputs at a time, and is connected to the modules using convenient plugs which replace the detachable screw terminals making connection rapid and trouble free.

For input simulation, the switches are used to turn inputs on and off. LED indicators adjacent to the switches show the status of each input. Connections are as follows:



For input simulation, the switches are used to turn inputs on and off. LED indicators adjacent to the switches show the status of each input. Connections are as follows:



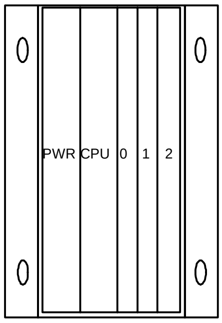
3. HARDWARE CONFIGURATION

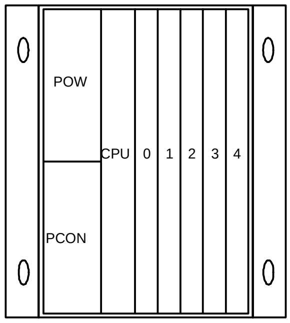
3.1 Selection of Mounting Rack

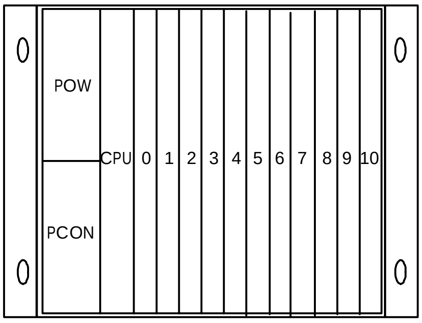
The number of the I/O modules and CPU type must be considered when selecting the mounting rack. If the process to be controlled is spread over a large area, considerable savings can be achieved by using the AL2000 to implement a distributed control system utilising several remote units. In this case only the central unit requires PLC programming, the remote units operate as Modbus RTU I/O units only.

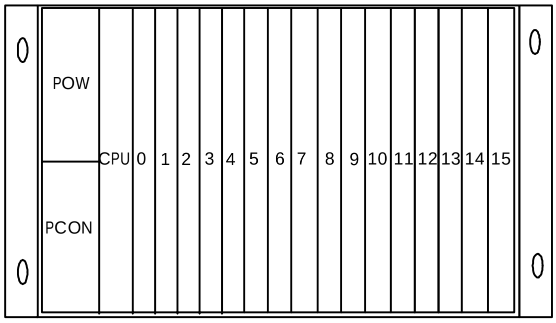
Five different mounting racks are available:

1. MR3-mini for up to 3 I/O modules (CPU2000SCP)
2. MR3 for up to 3 I/O modules
3. MR5 for up to 5 I/O modules
4. MR11 for up to 11 I/O modules
5. MR16 for up to 16 I/O modules









3.2 Selection of Power Input Module(s)

The AL2000 power supply is made up of two separate parts, the power connecting module and the power supply module. All mounting racks except the MR3 and MR3-mini must be fitted with both of these units. A different, single-unit power supply is included with the MR3 and the MR3-mini requires no power supply (the CPU2000SCP has its own power supply).
There are two types of power connecting module available, the PCON230 and the PCON24. These must be used in conjunction with the POWAC and POWDC power supply modules respectively:

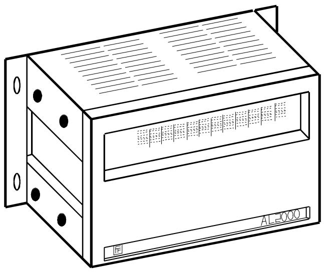
PCON230 and POWAC:	Input voltage	230VAC
	Suitable for	MR5, MR11, MR16
PCON24 and POWDC:	Input voltage	24VAC/DC
	Suitable for	MR5
PWR3:	Input voltage	230VAC or 24VAC/DC
	Suitable for	MR3 only

3.3 Modules Required for Each Unit

3.3.1 MR5/MR11/MR16 Basic Unit

The following modules must be installed in the basic mounting rack:

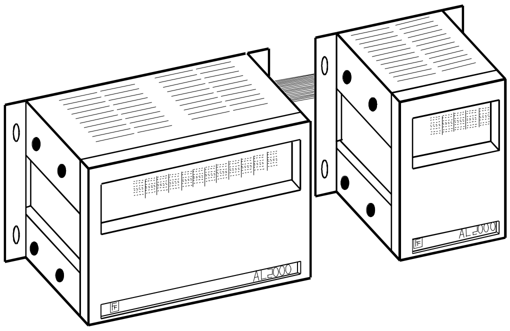
1. PCON power connection module
2. POW power supply module
3. CPU central processing unit
4. Additionally the required I/O modules.



3.3.2 MR5/MR11/MR16 Expansion Unit

The following modules must be installed in the expansion mounting rack:

1. PCON power connection module
2. POW power supply module
3. Additionally the required I/O modules.

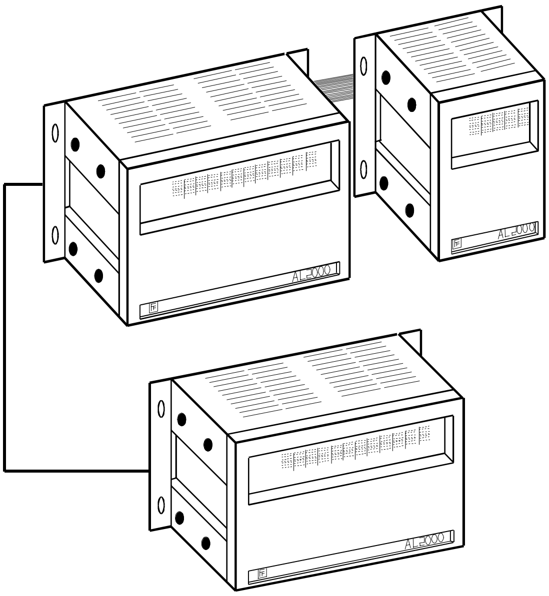


3.3.3 MR5/MR11/MR16 Remote Unit

The following modules must be installed in the remote mounting rack:

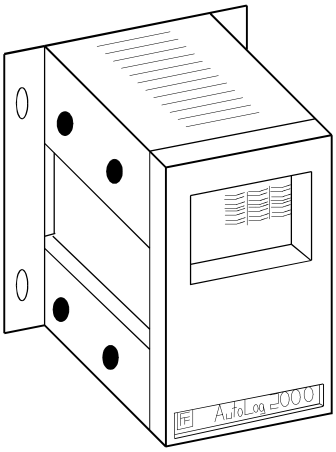
1. PCON power connection module
2. POW power supply module
3. CPU with DIP switch set to remote mode
4. Additionally the required I/O modules.

Remote units are connected to the basic unit via serial interface (SER3).



3.3.4 MR3 Mounting Rack

This unit is intended for use as a remote unit or a stand-alone unit for small applications. This unit cannot be expanded by adding other mounting racks, neither can it be used as an expansion mounting rack. The following modules must be installed in the MR3 for use as a basic unit:

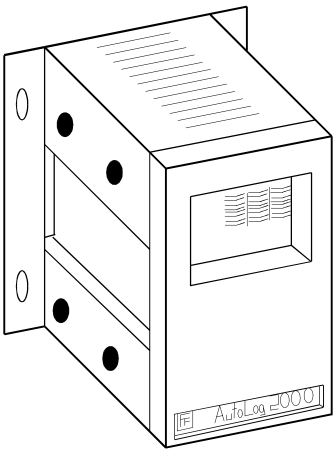


1. PWR3 power supply module
2. CPU central processing unit
3. Additionally the required I/O modules

Note: If this mounting rack is used as a remote unit, the CPU DIP switch must be set to remote mode

3.3.5 MR3-mini Mounting Rack

This unit is intended for use as a remote unit or a stand-alone unit for small applications. This unit cannot be expanded by adding other mounting racks, neither can it be used as an expansion mounting rack. The following modules must be installed in the MR3-mini for use as a basic unit:



1. CPU2000P central processing unit
2. Additionally the required I/O modules

The MR3-mini is powered by the CPU2000P module and therefore requires no seperate power supply. The CPU2000P, however, requires an input voltage of 24V.

Note: If this mounting rack is used as a remote unit, the CPU DIP switch must be set to remote mode

3.4 Hardware Configuration Set-up

The hardware is configured using the ALPro software. The software gives directions about where to place the I/O modules required by the application in the rack(s). Each I/O module has an internal address which the AL2000 operating system detects on power-up. No DIP-switches have to be set to determine the address of an I/O module.

The system program checks the configuration on every program cycle, this makes it impossible to accidentally replace an I/O module with one of the wrong type (e.g. during maintenance) and also ensures that no mistakes can be made in the hardware configuration. Select Configure from the ALPro main menu. Select the desired slot and make the following definitions:

1. Type of I/O module
2. Address of variables in the I/O map
3. On Error - action to be taken in the event that the hardware configuration is not the same as that defined in the ALPro Software: The options are:
   - STOP and RESET: execution of PLC program stops. All outputs are given the value 0
   - STOP: execution of PLC program stops. All outputs maintain their status
   - IGNORE: execution of PLC program continues
4. Type and measurement range of analog inputs (for analog I/O modules only)

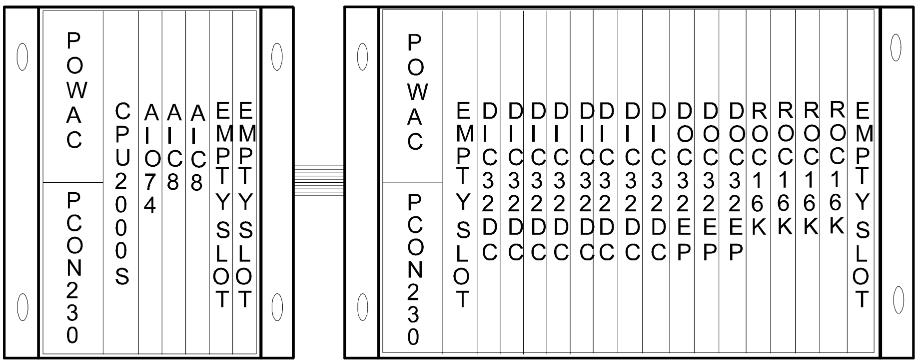
3.5 Example of Hardware Configuration

In a certain application the following number of I/O's must be handled:

232 x Digital Inputs (24VDC)
78 x Digital Outputs (24VDC)
58 x Digital Outputs (230VAC)
20 x Analog Inputs (4-20mA)
4 x Analog Outputs (4-20mA)

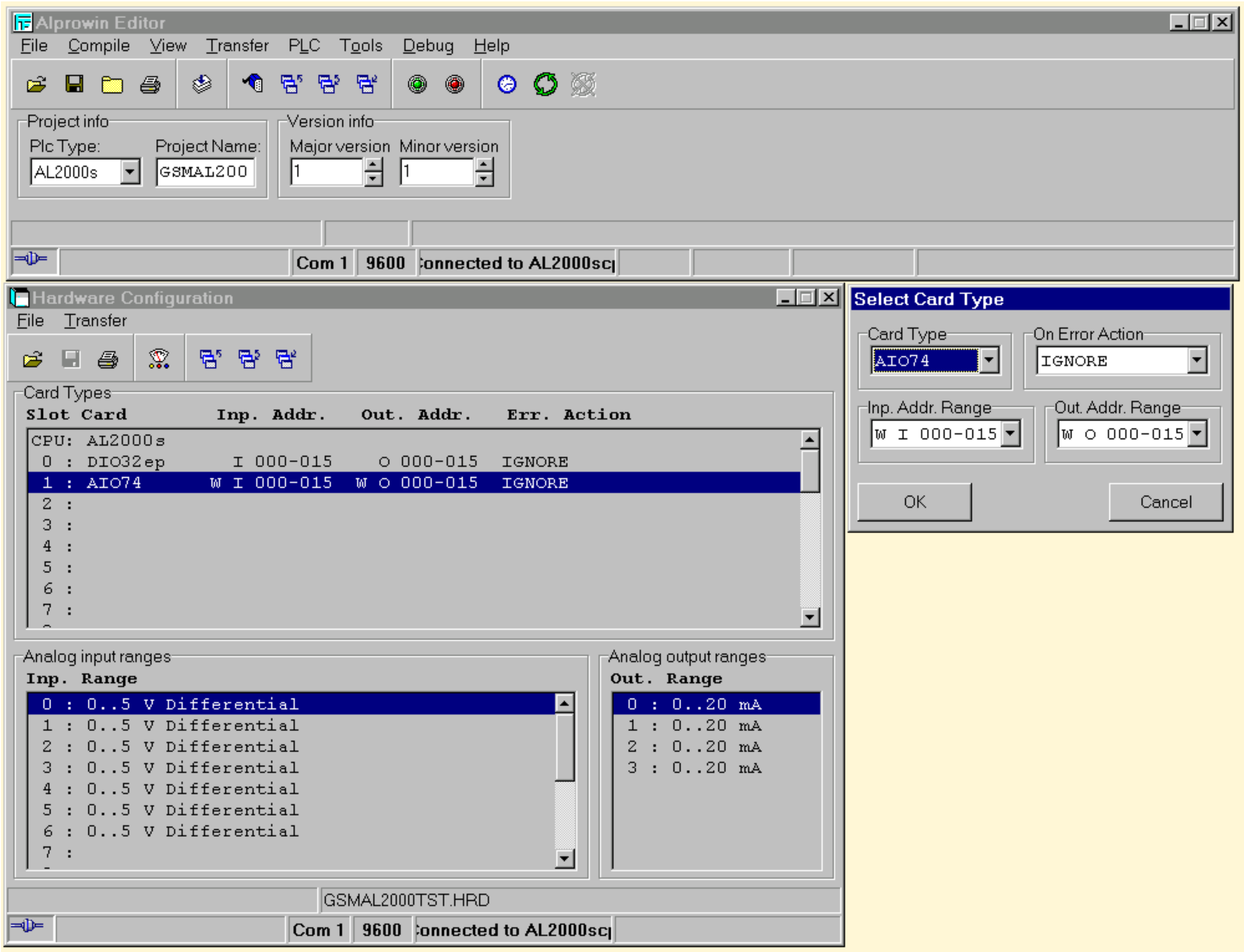
The following modules are required:

1 x CPU2000S
1 x MR5
1 x MR16
2 x POWAC
2 x PCON230
8 x DIC32DC
3 x DOC32EP
4 x ROC16
2 x AIC8
1 x AIO74
1 x Flat cable



3.5.1 Configuring the Modules

The required I/O modules and their respective slot positions must be configured in the ALPro software. This is done by selecting **Hardware Configuration** from the **View** menu.
The Configuration screen is shown below.

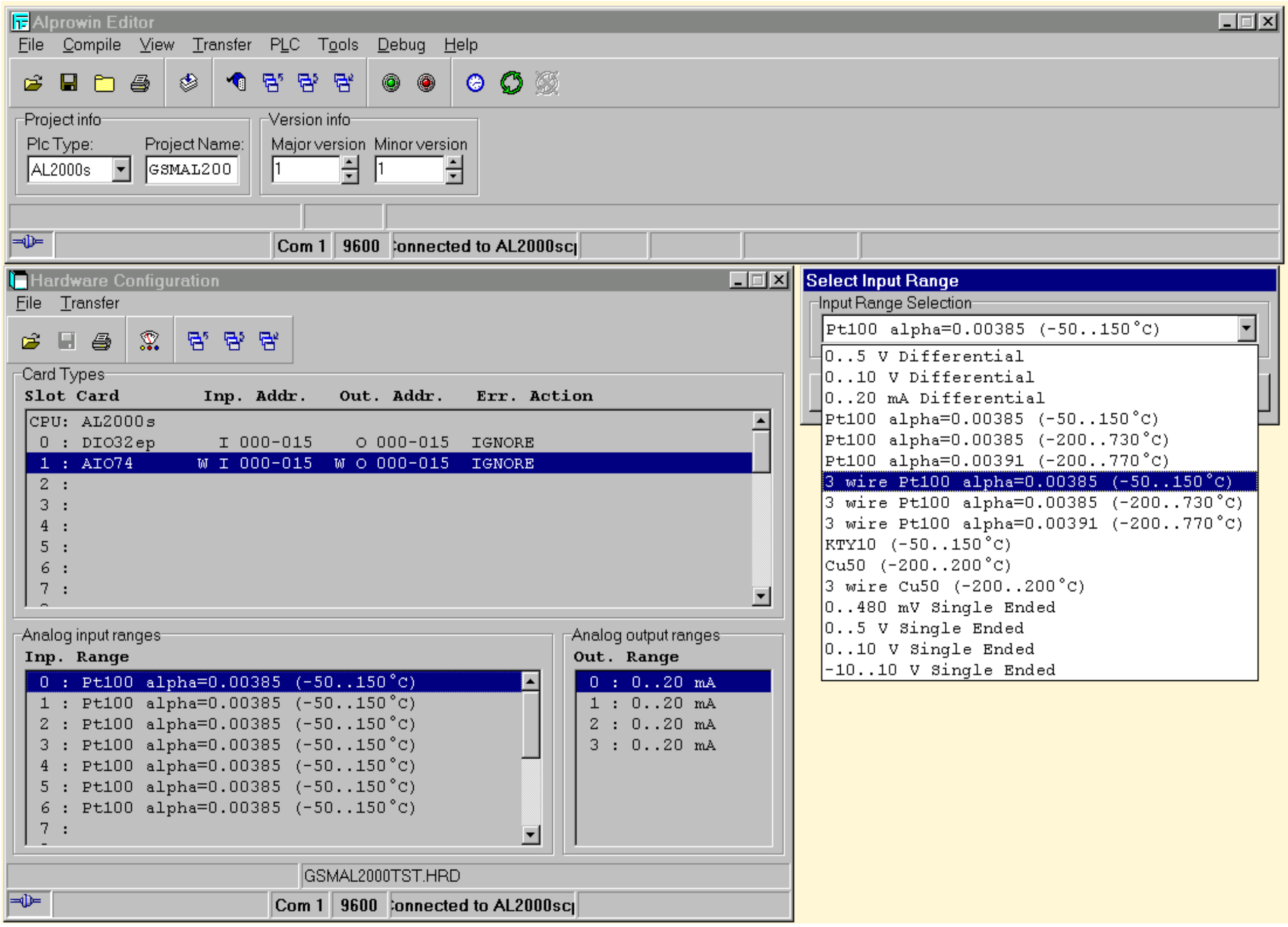


The required I/O modules can be configured as follows:

1. Select the desired slot using the up and down arrow keys
2. Select the desired type of card from the Card menu
3. Select On Error from the menu
4. Select the action to be taken in the event of an error in the card

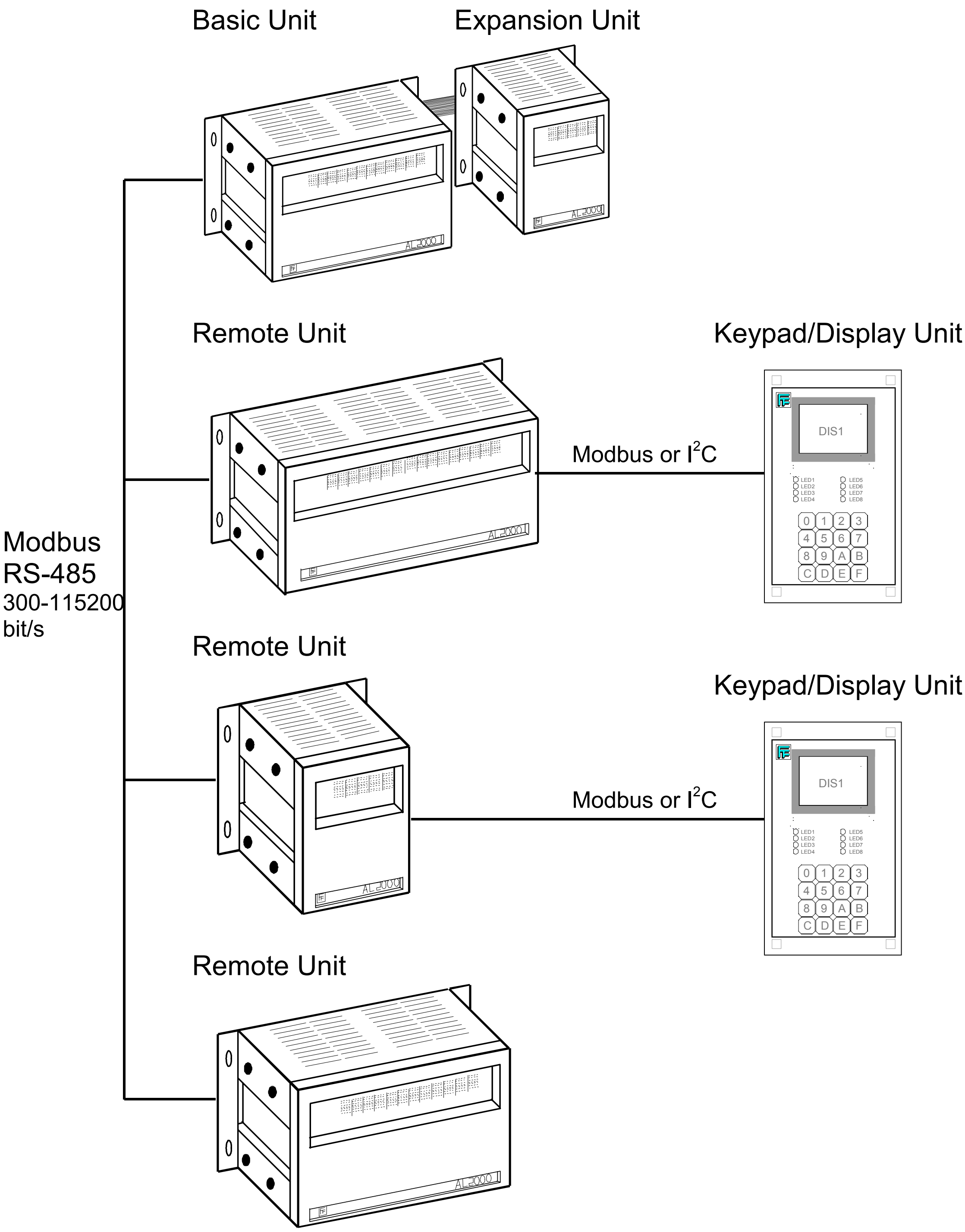
3.5.2 Setting the Range of Analog Inputs

The range of analog inputs can be set by selecting Range from the menu. The following screen is then displayed:



4. DISTRIBUTED CONTROL SYSTEM

All CPU modules are capable of controlling AL2000 remote units in addition to the basic unit (each having a maximum of 32 I/O modules). Each remote unit may consist of one basic mounting rack and one expansion mounting rack. The MODBUS RTU protocol is also supported all CPU modules, enabling the AL2000 to be connected to other PLCs and computers via serial interface. Using MODBUS remote units a fully distributed control system can be realised.



4.1 The MODBUS RTU Protocol

Using the MODBUS RTU protocol, the AL2000 can be connected to other PLCs (AL2000, AL100, AL20AN, AL32, AL16, AL14) and/or a computer via serial interface. The AL2000 can function as MODBUS Master or MODBUS Slave to other units.

Several MODBUS masters can be used in the same AL2000 system. One AL2000 can operate as a master to another AL2000, which in turn can operate as master to a third AL2000. This allows large distributed control systems to be realised.

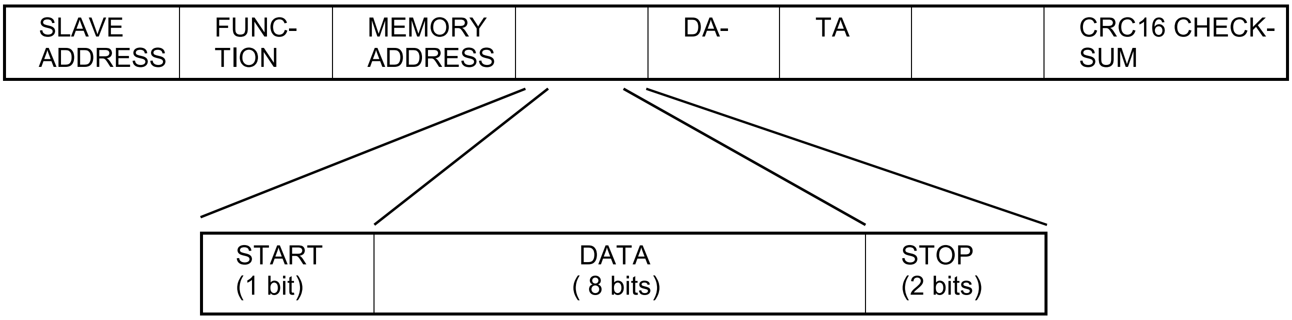
Using MODBUS to connect the AL2000 to a computer running software such as PARAGON, opens up unlimited opportunities for SCADA (Supervisory, Control And Data Acquisition) applications.

4.1.1 Error Checking

All MODBUS communication is subject to the following error checks by both Master and Slave units:

- CRC16 Checksum
- Address Validity Check
- Framing Check
- Command Validity Check
- Message Length Check
- Message Integrity Check

Messages may be transmitted at rates of 300-115kbaud. The composition of the message frames is shown below:



Modbus messages are launched from master either based on pollrate defined in the message or if user has defined conditional messages, these are launched from PLC-program with FCN instruction. The Slave unit ignores all invalid messages. If, after transmitting a message, the Master does not receive a valid response from a slave, it will time-out after 500 ms and re-send the message. If the Master still does not receive a valid response, it will ignore all further messages destined for that slave for 10 seconds. On the next run through the message list the Master will attempt to send messages to that slave again.

The PC's MODBUS COM port may be initialised from DOS using: **MODE COM1 9600,8,2,N,P**

4.1.2 Addresses of Inputs and Outputs

All I/O points have addresses, these are used by MODBUS. The address assignment can be seen on the ALPro configuration screen's I/O map. I/O addresses are assigned in blocks of 16, therefore there will be unused addresses if an I/O module requires less then 16 addresses. For an AIO74 module the used/unused addresses are:

AIO74 Inputs	0	1	2	3	4	5	6	-	8	9	10	11	12	13	14	-
AIO74 Outputs	0	1	2	3	-	-	-	-	-	-	-	-	-	-	-	-

Unused addresses are represented by the '-' symbol

4.1.3 Serial Port Configuration

The AL2000 has 3 serial interfaces of which all can be configured to function as MODBUS Slave and MODBUS Master (independently of one another). These are:

SER 1, SER 2	RS-232
SER 3	RS-485

The serial ports must be configured for the required baud rate and MODBUS master/ slave mode. This is done by initialising the following register outputs:

MODBUS modes	SER 1, SER 2, SER 3	MODBUS Slave/MODBUS Master
Baud Rates	SER 1, SER 2, SER 3	0,3/1,2/2,4/4,8/9,6/19,2 kbit/s 28,8/38,4/57,6/115,2 kbit/s

Maximum cable length for RS-232 is 15m for 9600 bit/s
Maximum cable length for RS-485 is 1200m for 9600 bit/s

4.2 Setting Up A MODBUS Master

A MODBUS master can be set up on SER1 and/or SER2 and/or SER3 by following the steps below.

Example: Set The Serial Port 2 to Modbus MASTER, baud rate 9600,N,8,1

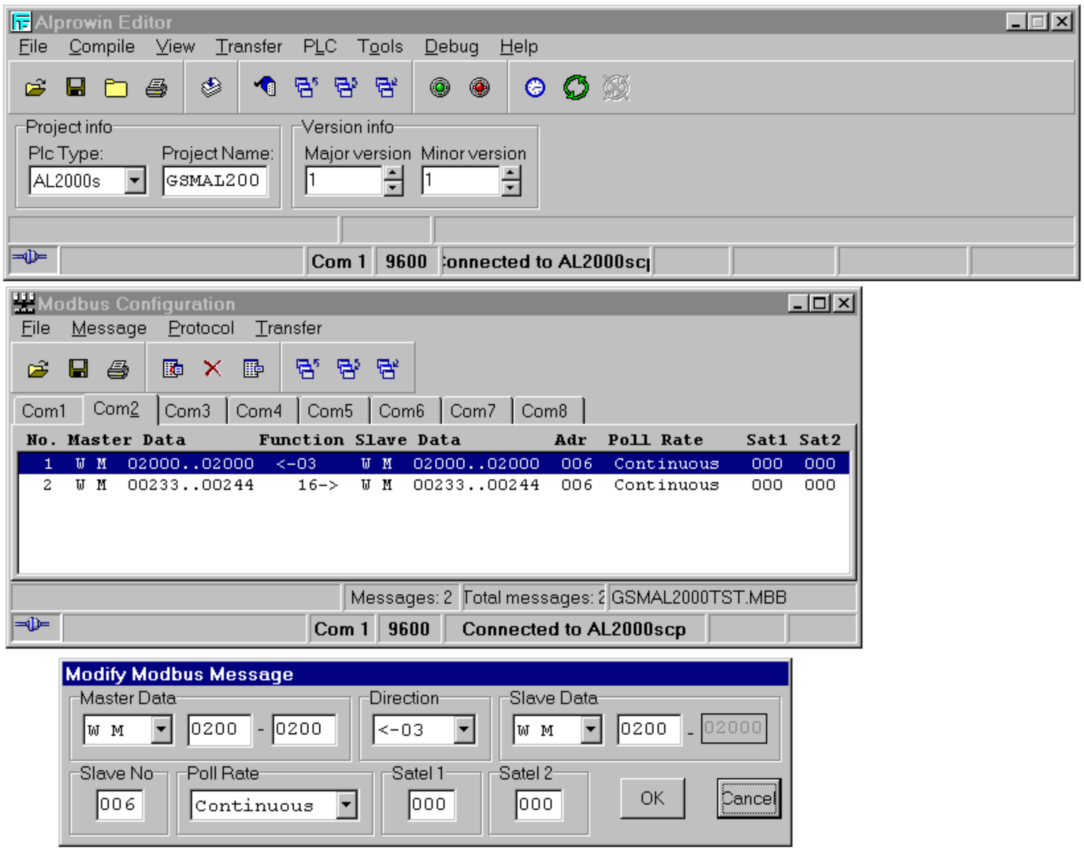
```
STR    R  C    5
EQ     R  O  215        ; set SER2 to MASTER mode
STR    R  C    4
EQ     R  O  229        ; set SER2 baud rate to 9600
STR    R  C    0
EQ     R  O  219        ; set SER 2 data configuration to 8,N,1
```

See chapter 9.3 Register Output Configuration for available baud rates, communication modes and data configuration modes.

4.2.1 Configure the MODBUS Master Messages

In one transmission, several input or output values can be transferred between master and slave. This is called a message. A separate definition must be made for each master/ slave message (max. 63 per serial port) stating what information is to be sent.

MODBUS Master message definitions should be made using the Modbus Configuration program in the ALPro software. The serial interface to be configured should be selected first (SER2/SER3-MODBUS), after which message definitions can be made as follows:



Master data Variable type and address range in the Master where data is to be written to or read from
Direction Arrow defines whether operation is read from slave (<-03) or write to slave (16->).
Slave data Variable type and start address in the Slave where data is to be written to or read from
Slave No Address of slave where data is to be written to or read from. Address 0 is used for broadcast messages. Broadcast messages are received by all slaves, but the slaves do not respond to these messages
Poll rate Defines the update rate for this message. Possible poll rates are :
Continuous ;update message as fast as possible
1 second, 10 seconds, 30 seconds, 60 seconds
Conditional ;launch message from PLC-program with FCN-instruction.
Satel1, Satel2 Satel codes are used when Satel radio modems are used as repeaters. (not added yet).

In the above example message 1 would read outputs 0-15 from slave 1 and place them in the masters memory at addresses 0-15.

After all messages have been defined, the configuration information must be transferred to the AL2000. This is done by selecting 'Send File' from the 'File' menu. Transfer can be verified by selecting 'Verify' from the menu.

The following functions are available in messages:

Read/write operation	**Modbus read**	**Mod-bus write**	**Address offset**	**Control sw. Address**	**FCS sw. Address**
Binary output (O)	01	05, 15	0	00001	SDO 0001
Binary memory (M)	01	05, 15	1024	01025	SDO 1025
Binary memory (GM)	01	05, 15	2048	02049	SDO 2049
Binary memory (BM)	01	05, 15	3072	03073	SDO 3073
Binary input (I)	02		0	10001	SDI 0001
Word input (W I)	04		1024	31025	SAI 1025
Register output (R O)	03	06, 16	0	40001	SAO 0001
Register memory (R M)	03	06, 16	1024	41025	SAO 1025
Word output 0 ->1023 (W O)	03	06, 16	2048	42049	SAO 2049
Word memory 0 ->1023 (W M)	03	06, 16	3072	43073	SAO 3073
Register general memory (R GM)	03	06, 16	4096		
Word general memory (W GM)	03	06, 16	5120	45121	SAO 5121
Word output 1024 -> 2047 (W O)	03	06, 16	6144	46145	SAO 6145
Word memory 1024 -> 4095 (W M)	03	06, 16	7168	47169	SAO 7169
Word output 2048 -> 4095 (W O)	03	06, 16	10240		
Word memory 4096 -> 16275 (W M)	03	06, 16	12288		

4.3 Setting Up A MODBUS Slave

A MODBUS slave can be set up on SER1and/or SER2 and/or SER3 by following the steps described below.

Example: Set The Serial Port 3 to SLAVE, baud rate 9600,N,8,1, slave address = 2

```
STR    R  C    4
EQ     R  O  216          ;set SER3 to SLAVE mode
STR    R  C    4
EQ     R  O  217          ;set SER3 baud rate to 9600
STR    R  C    0
EQ     R  O  218          ; set SER 3 data configuration to 8,N,1
STR    R  C    2
EQ     R  O  243          ; set Slave address to 2
```

See 4.1.3. Serial Port Configuration for available baud rates

The slave address can be from 1 to 255. If the PLC operates as a slave unit on two or three serial lines, the slave address is the same for all slave channels.
Address 0 is reserved for broadcast messages. A slave cannot have this address.

4.4 Using MODEMs with MODBUS

AL2000 RS-232 Serial channels supports modem control/status signals.

	Bit 0	Bit 1	Bit2	Bit3	bit4	Bit5	Bit6	Bit7
SER1 R O 57 SER2= RO 58	0=system program controls RTS 1= user controls RTS & DTR	RTS to modem	DTR to modem		DSR from modem	RI from modem	DCD from modem	CTS from modem

Two wire, carrier, MODEMs are normally used on standard 'dial-up' telephone lines. MODEMs which conform to the following CCITT recommendations are suitable for use with the AL2000 system:

V.22 (1200 baud)
V.22bis (1200/2400 baud)
V.32 (4800/9600 baud)

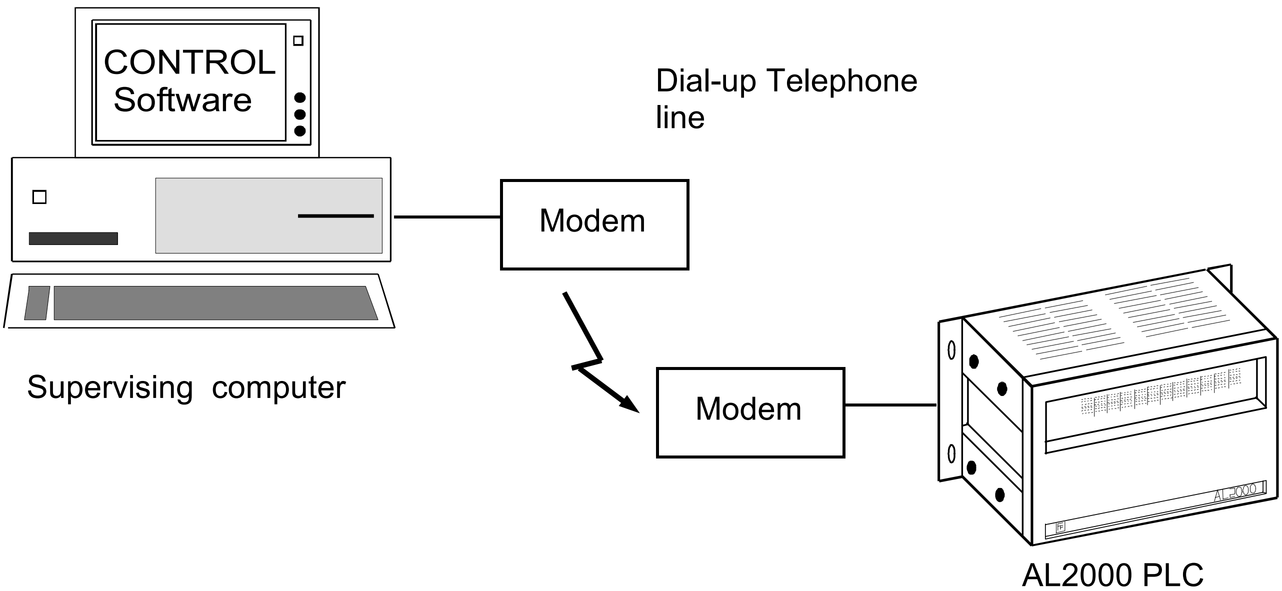
Using an auto-dialing MODEM which recognises 'AT' instructions, remote AL2000 units can dial-up and send data to a supervising computer.

The 'PRT' instruction is used to dial telephone numbers:

```
PRT ("ATDP 90844992",<CR>)
```

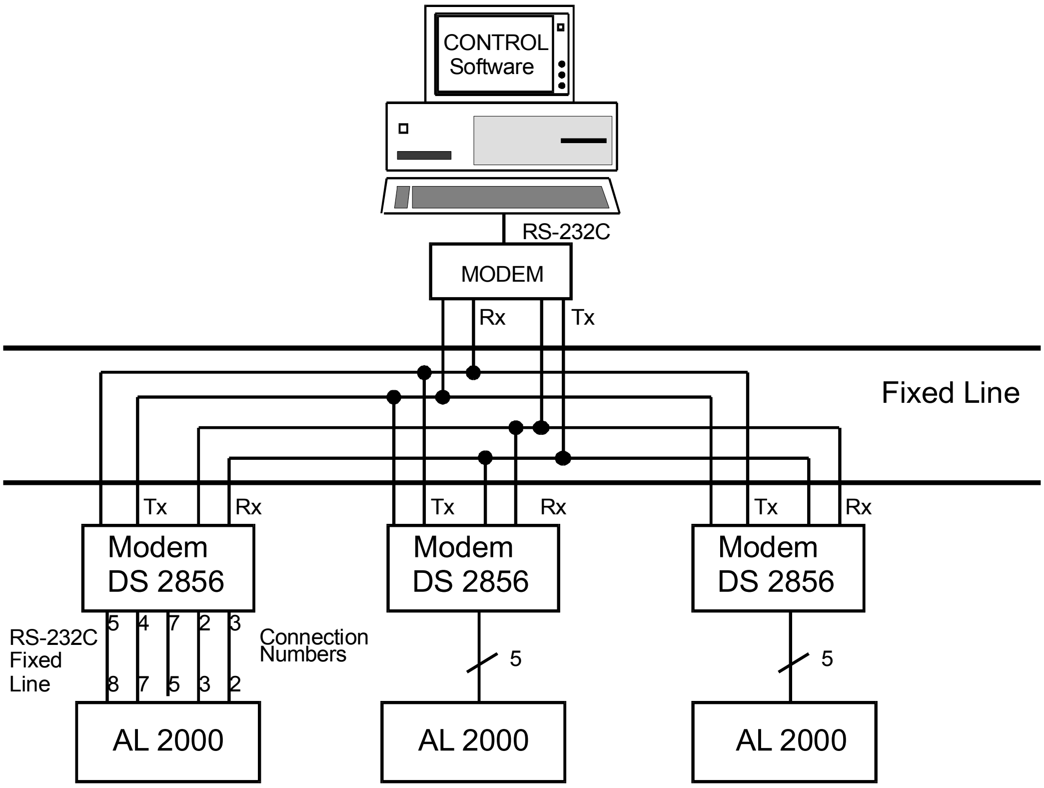
The MODEM reports a connection using the 'CONNECT' response. For the specific format of 'AT' instructions, consult the MODEM manual.

Data transfer rates of over 1200 baud, using telephone lines, can give rise to frequent errors. If 'PRT' instructions are used to transfer data, error correcting MODEMs should be used as data transfer errors may lead to operational problems. However, if an error correcting protocol like MODBUS is used, error correcting MODEMs are not necessary.



Only approved MODEMs should be connected to the telephone network!

Carrierless MODEMs are usually used over short distances (maximum cable length, 15km). Typically a four wire system is used in order to realise a network.



Carrierless MODEMs can be used for point-to point connection.

MODEMs are usually mains powered. If a battery back-up is required, the MODEM must be capable of being powered by a battery.

When a MODEM is used, the following definitions should be made in the PLC.

	Data Transfer Protocol	**Baud Rate**	**Data format in terminal mode**
SER1	R O 213 2 or 4	R O 214 0,1,2,3 or 4	R O 210 0 to 4
SER2	R O 215 0 or 4	R O 229 0,1,2,3 or 4	R O 219 0 to 4
SER3	R O 216 0 or 4	R O 217 0,1,2,3 or 4	R O 218 0 to 4

See also: REGISTER MEMORIES, OUTPUTS AND VARIABLES.

The following is a program example using MODEM communication.

```
STR     R    C  000
EQ      R    O  212                    ;select English instruction set
STR     R    C  000
EQ      R    O  215                    ;select MODEM on SER2
EQ      R    O  219                    ;select terminal mode SER2:data configuration
STR     R    O  001
EQ      R    O  229                    ;select baud rate - 1200 Bd
STR          S  000                    ;establish connection
AND          I  000
STEP         S  001
STR          S  001                    ;dial number
PRT     ("ATE0V0DP844992",<CR>)
STEP         S  002
EQ      R   RO  232
EQ      R   RO  233
NEXT         S  002     030 ;time-out
NEXT         S  003     060
STR          S  004
STEP         S  001
STR     R    O  232                    ;check that connection is established
EQU     R    C  013
AND          S  002
STEP         S  010
STR     R    O  233
EQU     R    C  049
AND          S  010
STEP         S  011
STR          S  011                    ;data transfer
PRT     ("AL2000 is sending some data",<CR>)
STEP         S  055
NEXT         S  055     060
STR          S  056
STEP         S  012
NEXT         S  012     003            ;terminate connection (hang-up)
STR          S  013
PRT           ("+++")
STEP         S  014
NEXT         S  014     003
STR          S  015
PRT        ("ATH0",<CR>)
STEP         S   16
EQ      R   RO  232
EQ      R   RO  233
NEXT         S  016     005            ;PLC waits for next connection attempt
STR     R    O  232
EQU     R    C  013
AND          S  017
STEP         S  019
STR          S  019
STEP         S  000
NEXT         S  017     005            ;if hang-up is unsuccessful then hang-up again
STR          S  018
STEP         S  012
STOP
```

AT	speed, character length and parity
D	dial phone number
P	pulse dialing
T	tone dialing
E0	disable echo
V0	digit responses

5. OPERATIONAL/TECHNICAL DATA

All AL2000 components are furnished with identification labels which state the code, serial number and name of the quality control inspector. On most components these stickers can be found on the bus connectors. A sample identification label is shown below:

FF-AUTOMATION OY Made In Finland	Code 902203 Serial 01059300164	DCL Inspected By

5.1 Power Supply

Each mounting rack (except MR3-mini) must be equipped with its own power supply. The 230 VAC power supply can be used with all mounting racks, however, the 24 VDC/AC power supply can only be used with the MR5 mounting rack.

The external power supply cables are connected to the power connecting module and the power supply module generates the voltages required by the AL2000.

Supply Voltage	Power Connecting Module	Power Supply Module	Mounting Rack
230 VAC	PCON230	POWAC	MR5, MR11, MR16
24 VAC/DC	PCON24	POWDC	MR5
24 VAC/DC 230 VAC	PWR3		MR3 only

The protective ground must be connected to the ground terminal on the mounting rack as shown below.

(The left bottom corner of the AL2000)

CAUTION: In order to prevent electric shock, disconnect mains supply before making any power connections.

5.1.1 PCON24 Power Connecting Module

The PCON24 is a power connecting module for supply voltages of 24VDC or 24VAC. The PCON24 drives the POWDC-25W power supply module and can also supply an auxiliary voltage of 24VDC (0.8A) for powering sensors or an AL32 PLC etc.

NOTE! The PCON24 can only be used in the small MR5 mounting rack.

Supply voltage: 20-32 VDC or
20-26 VAC

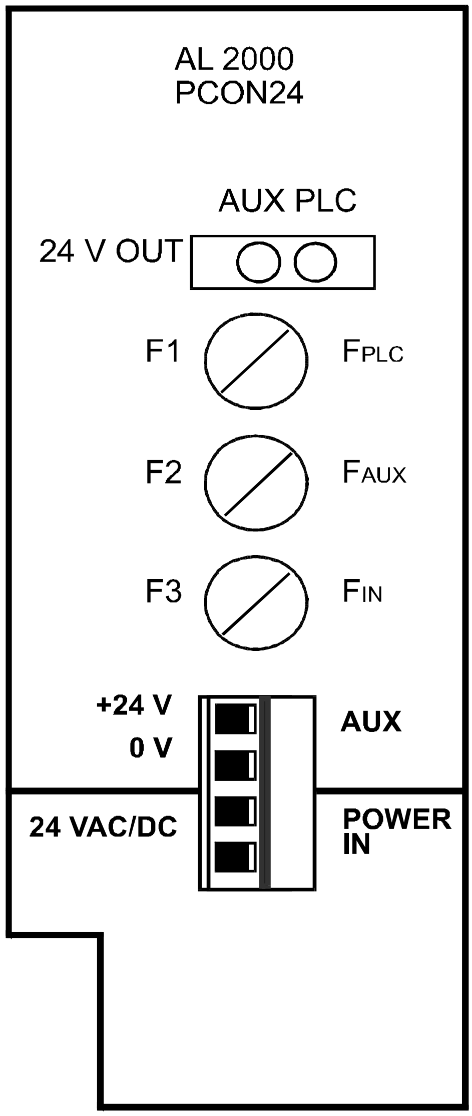
POWER IN The supply voltage should be connected to these screw terminals.
AUX Auxiliary voltage of 24VDC (0.8A) is available for external use from these terminals.

There are three fuses in the PCON24 module:

F1 T1.6A slow-blow fuse for supply to the AL 2000
F2 T1A slow-blow fuse for auxiliary power supply
F3 T2.5A slow-blow fuse for supply to the PCON24

There are two LED indicators on the PCON24 module:

AUX (green) Auxiliary supply ON
PLC (green) Supply to POWDC ON



CAUTION: In order to prevent electric shock, disconnect mains supply before making any power connections.

5.1.2 PCON230 Power Connecting Module

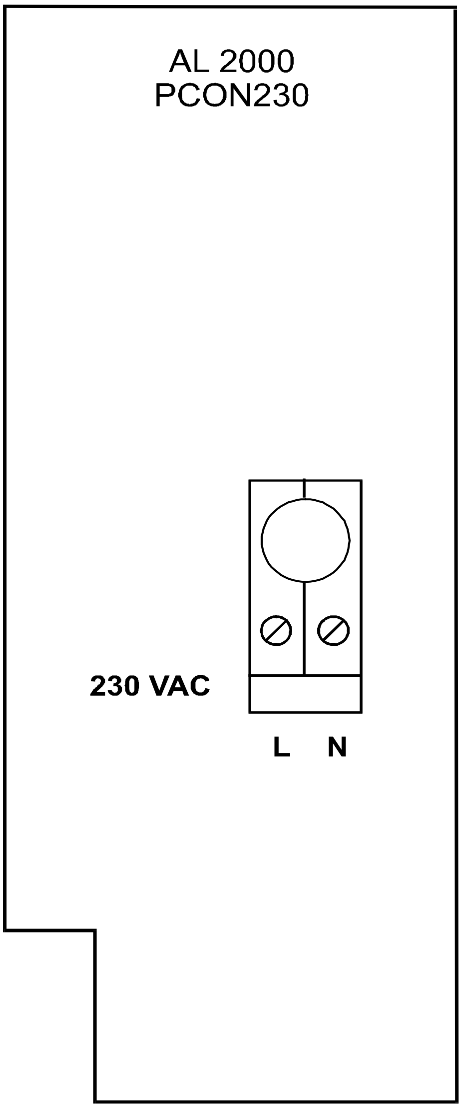
The PCON230 is a power connecting module for supply voltages of 230 VAC. The PCON230 drives the POWAC power supply module.

Supply voltage: 180-265 VAC

L and N The supply voltage should be connected to these screw terminals.

The PCON230 module is equipped with a high frequency filter to remove interference, and protection against excessive input voltages.

The PCON230 is fitted with a T2.5A slow-blow fuse for short circuit protection.



CAUTION: In order to prevent electric shock, disconnect mains supply before making any power connections.

5.1.3 POWDC-25 W Power Supply Module

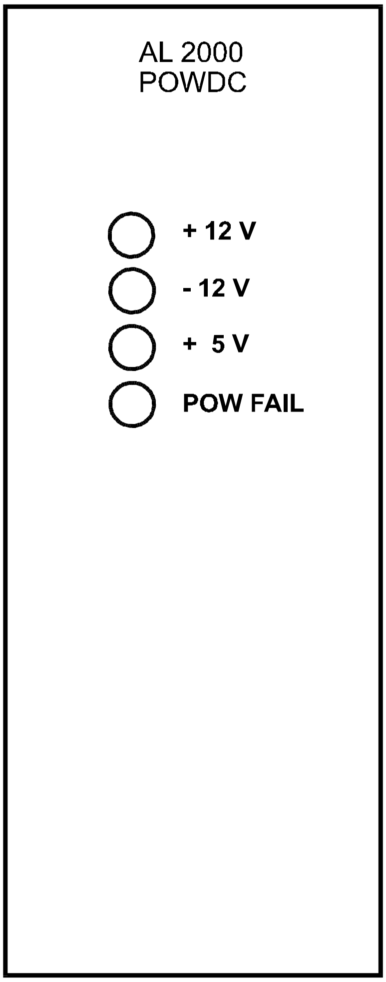
The POWDC-25W is a power supply module designed for use with the PCON24 power connecting module and supplies the following voltages to the AL2000:

+5V (3A)
+12V (0.6A)
-12V (0.6A)

There are four LED indicators in the POWDC module:

+12 V (green)	+12V supply ON
-12 V (green)	-12V supply ON
+5 V (green)	+5V supply ON
POW FAIL (red)	Supply voltage too low

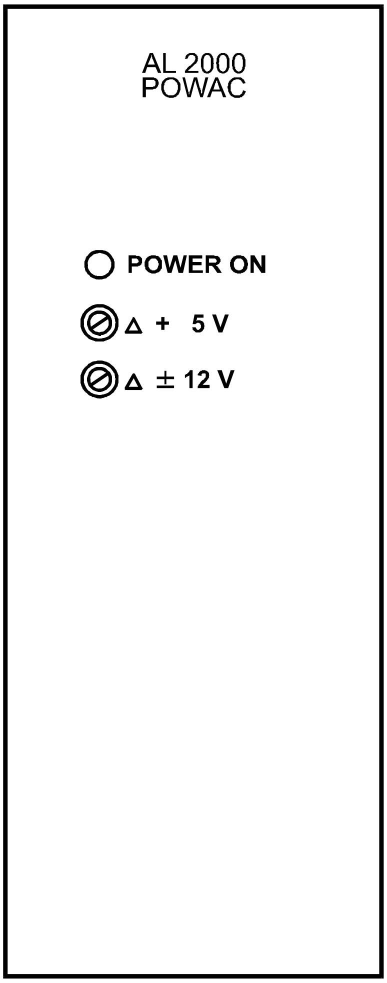
The POWDC power supply module isolates the supply to AL 2000 from the external supply voltage. The POWDC also provides electronic overload protection to limit the output power. The high frequency interference filter is designed to standards VDE 0806, VDE 0871 and FCC 15B.



5.1.4 POWAC Power Supply Module

The POWAC power supply module is designed for use with the PCON230 power connection module and supplies the following voltages to the AL2000:

+5V (10A)
+12V (0.6A)
-12V (0.6A)



There is one green LED indicator in the POWAC module:

POWER ON   Supply voltage ON

In addition the POWAC is furnished with two potentiometers for voltage adjustments:

Δ + 5V	Adjustment for the 5V supply
Δ ± 12 V	Adjustment for the 12V supplies

There is one glas tube fuse inside power unit,
(size 5 x 20 mm)
ratings 3.15 A 250 VAC slow blow.

5.1.5 PWR3 Power Supply Module

The PWR3 is a complete power supply module for use with the MR3 mounting rack. Input voltages can be either AC or DC. The input voltage ranges are as follows:

AC supply	180-265VAC
DC supply	20-32VDC

The PWR3 supplies the following voltages to the AL2000:

+5V (3A)

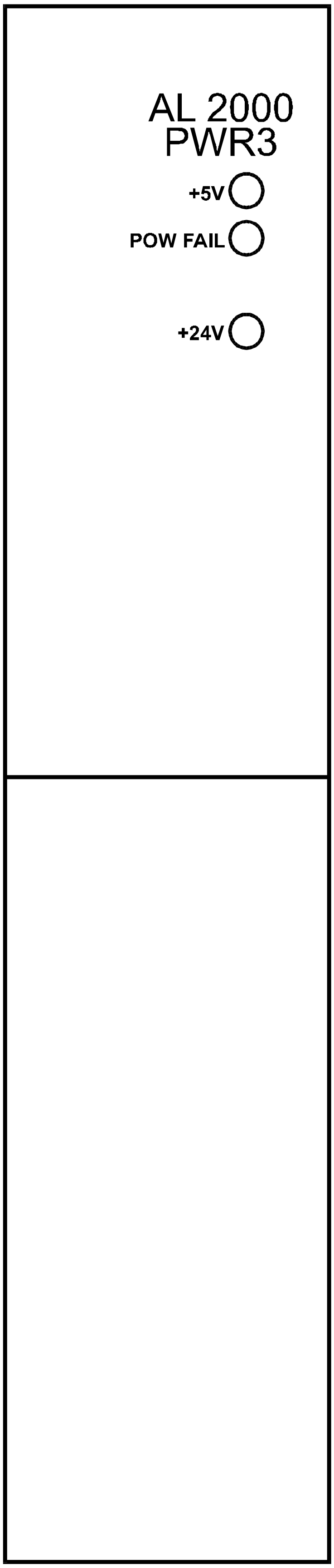
The following LED indicators are located on the front edge of the module:

+5V (green)	+5V supply ON
POW FAIL (red)	Supply voltage too low
+24V (green)	Auxiliary supply ON

There are three fuses provided on the PWR3 module:

Fin	T315mA slow-blow fuse for overload protection
Faux	T1A slow-blow fuse for auxiliary power supply
Fplc	T1A slow-blow fuse for supply to the AL2000

CAUTION: In order to prevent electric shock, disconnect mains supply before making any power connections.



5.2 Digital I/O Modules

All digital I/O modules are fitted with detachable screw terminals for I/O wiring. These terminals can be removed from the module without disconnecting the wiring. There are two labelling conventions for the screw terminals depending on whether the terminal is an input or an output:

Input	Black text on a white background
Output	White text on a black background

The numbers of the inputs or outputs are shown on both the screw terminals and the LED indicators, these being decimal numbers from 0-31. In the ALPro software inputs and outputs are defined as follows:

Inputs:	I <I/O slot number>.<input number on I/O module>
Outputs:	O <I/O slot number>.<output number on I/O module>
Example:	I 2.4 (digital input number 4 on the module in slot number 2)
	O 6.28 (digital output number 28 on the module in slot number 6)

All I/O modules are fitted with LED's to indicate the status of each input/output and the operational status of the module. These LED's are colour coded as follows:

Input	Yellow	The input is ON
Output	Red	The output is ON
ACT	green/yellow	The module is in operation
ERR	Red	The operating system has discovered one of the following faults: - Incorrect configuration, module has not been configured for this slot - Fault in the module
SP.F	green	The spare fuse is available and working (located on the card)

5.2.2 DIC32DC Input Module

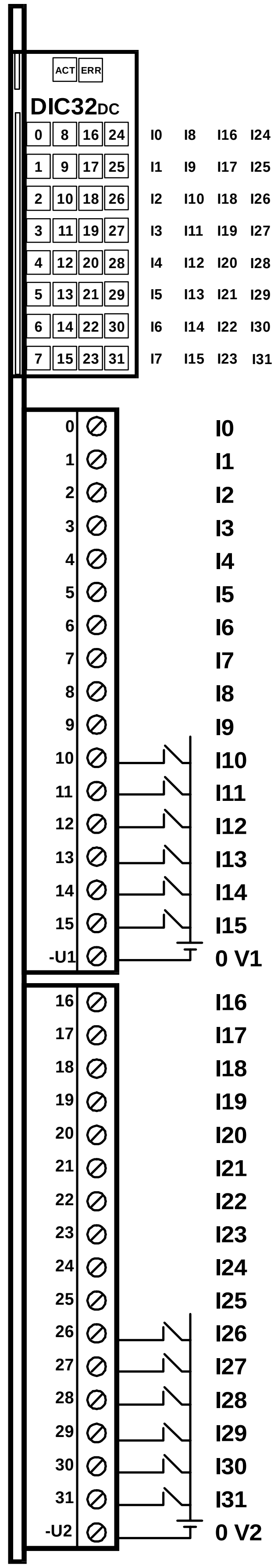
The DIC32DC is a digital input module with 32 isolated inputs in two groups of 16 inputs each. The current consumption of DIC32DC is 250 mA. Input wiring is connected to two detachable screw terminals on the front edge of the module. The common ground for each of the two groups is connected to the bottom screw in each of the two screw terminals. The logic range of the DIC32DC module is:

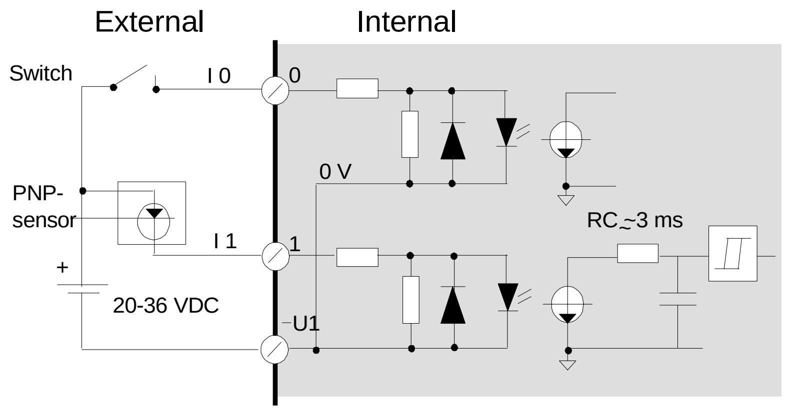
logic 0	-30 to +5VDC (0-1.1mA)
logic 1	+13 to +36VDC (4-11mA)

Sensor must be PNP output type or voltage supplying .

The following LED indicators are located on the front edge of the module:

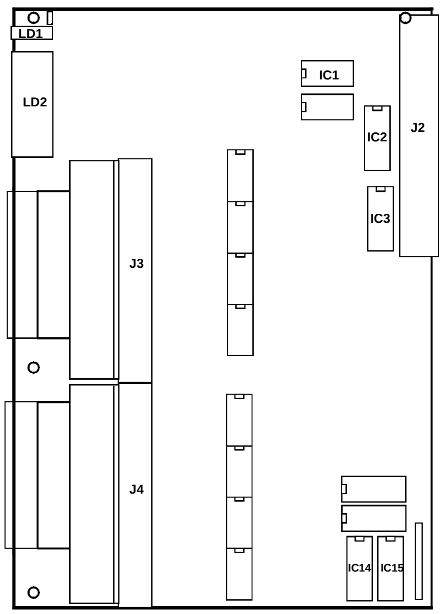
ACT (green)	Module in operation
ERR (red)	The operating system has discovered a fault in the module
0-31 (yellow)	Inputs I0-I31 are ON



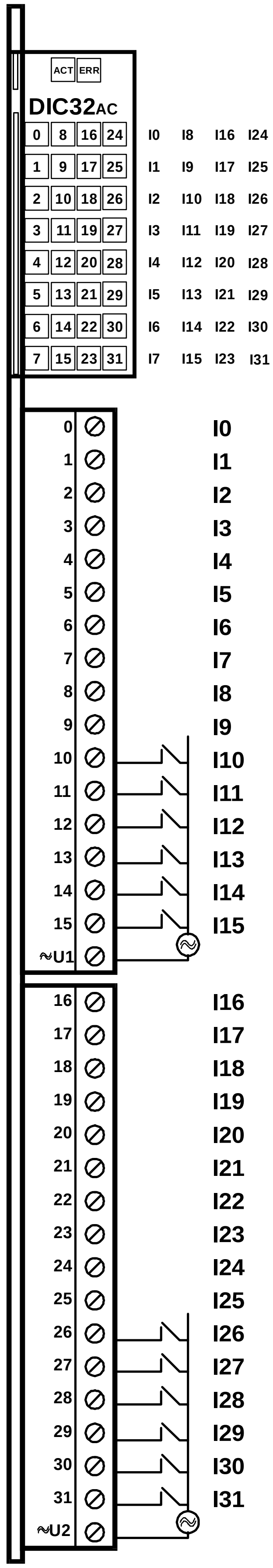


Layout of the DIC32DC

IC1	Interrupt register
IC2	Module address detection
IC3	Internal data bus buffer
IC14	Code register for module
IC15	Test register
LD1	LED-Unit (ACT/ERR indicator)
LD2	LED-Unit (input status I0-I31)
J2	Internal data bus connector
J3	Terminals for inputs I0-I15
J4	Terminals for inputs I16-I31



5.2.3 DIC32AC Input Module

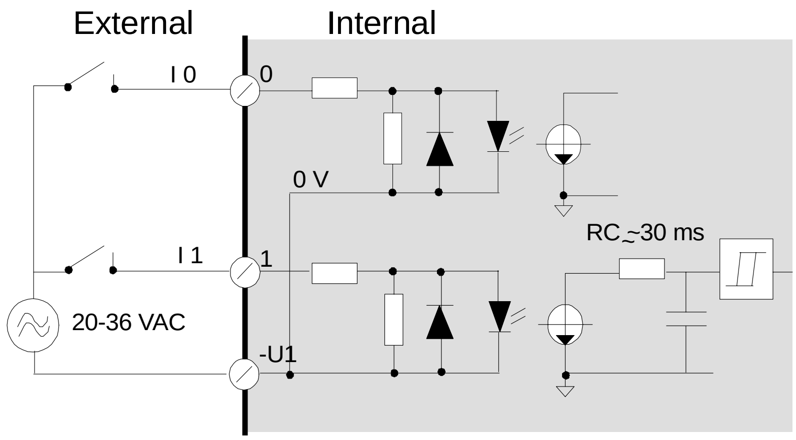


The DIC32AC is a digital input module with 32 isolated inputs in two groups of 16 inputs each. The current consumption of DIC32AC is 250 mA. Input wiring is connected to two detachable screw terminals on the front edge of the module. The common ground for each of the two groups is connected to the bottom screw in each of the two screw terminals. The logic range of the DIC32AC module is:

logic 0	0 to +5 VAC (0-1.1mA)
logic 1	19 to +36VAC (6-11mA)

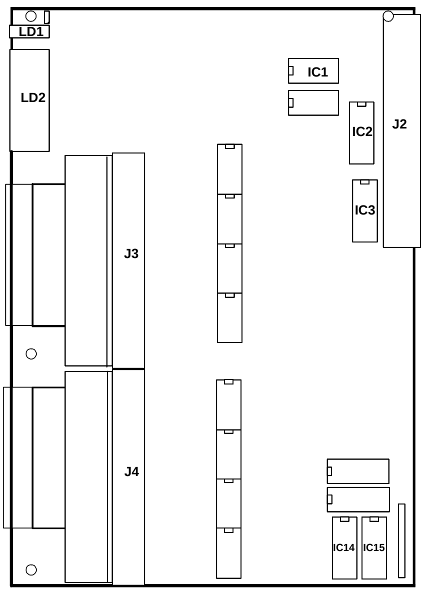
The following LED indicators are located on the front edge of the module:

ACT (green)	Module in operation
ERR (red)	The operating system has discovered a fault in the module
0-31 (yellow)	Inputs I0-I31 are ON



Layout of the DIC32AC

IC1	Interrupt register
IC2	Module address detection
IC3	Internal data bus buffer
IC14	Code register for module
IC15	Test register
LD1	LED-Unit (ACT/ERR indicator)
LD2	LED-Unit (input status I0-I31)
J2	Internal data bus connector
J3	Terminals for inputs I0-I15
J4	Terminals for inputs I16-I31



5.2.4 DIF16 Fault Checking Digital Input Module

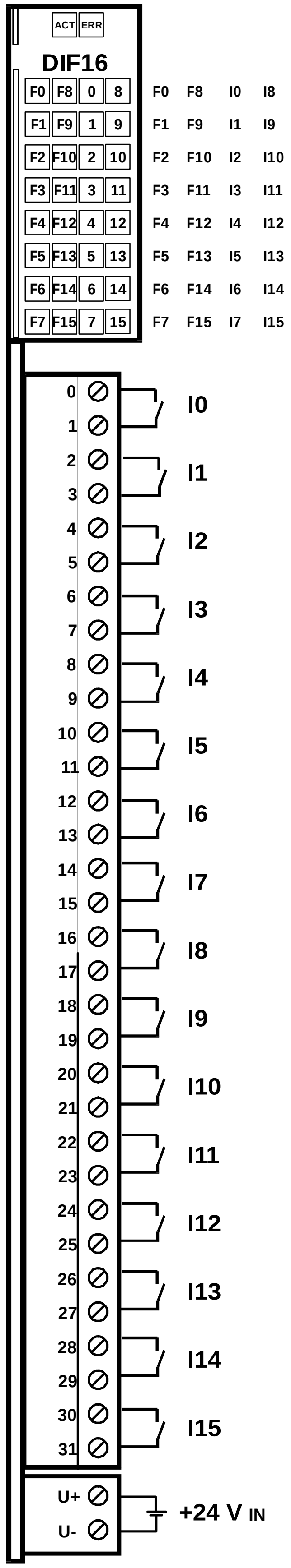
The DIF16 is a digital input module with 16 fault checking inputs isolated as a group.

The following LED indicators are located on the front edge of the module:

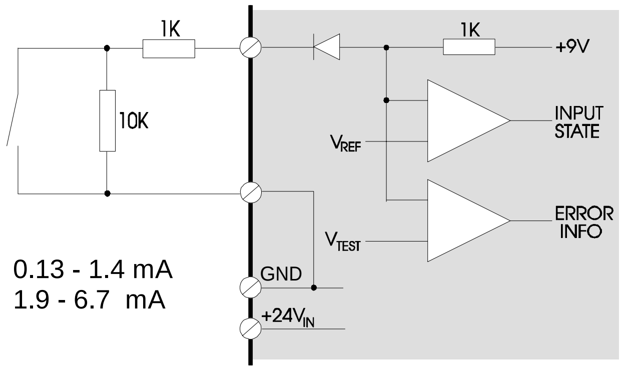
ACT (green) Module in operation
ERR (red) The operating system has discovered a fault in the module
0-15 (yellow) Inputs I0-I15 are ON
F0-15 (red) Fault in inputs I0-I15

The DIF16 can detect short and open-circuit faults in the cable. The status of each input can be determined from the LED indicators and can be used in the PLC program. Status indication is as follows:

Status 0-15	Fault F0-15	Condition
OFF (0)	OFF (0)	Input OFF
ON (1)	OFF (0)	Input ON
OFF (0)	ON (1)	Open circuit
ON (1)	ON (1)	Short circuit

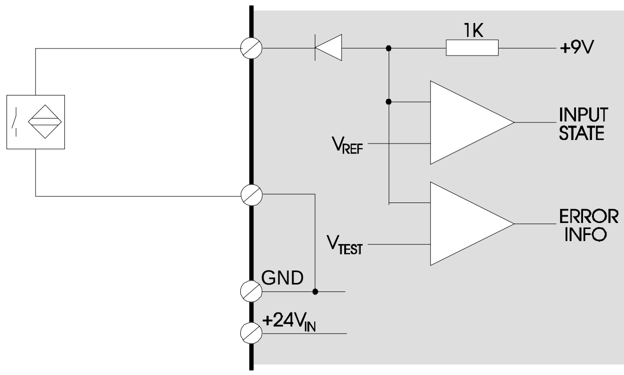


Connection diagram for a switch



logic 0 0.13 - 1.4 mA
logic 1 1.9 - 6.7 mA

Connections for DIN19234 or NAMUR transducer

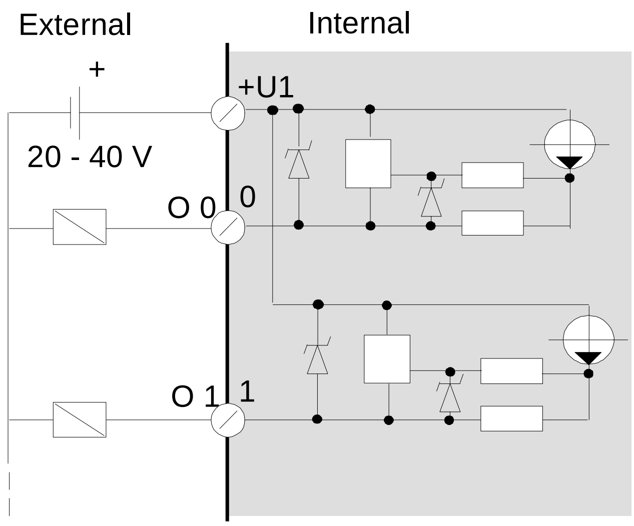


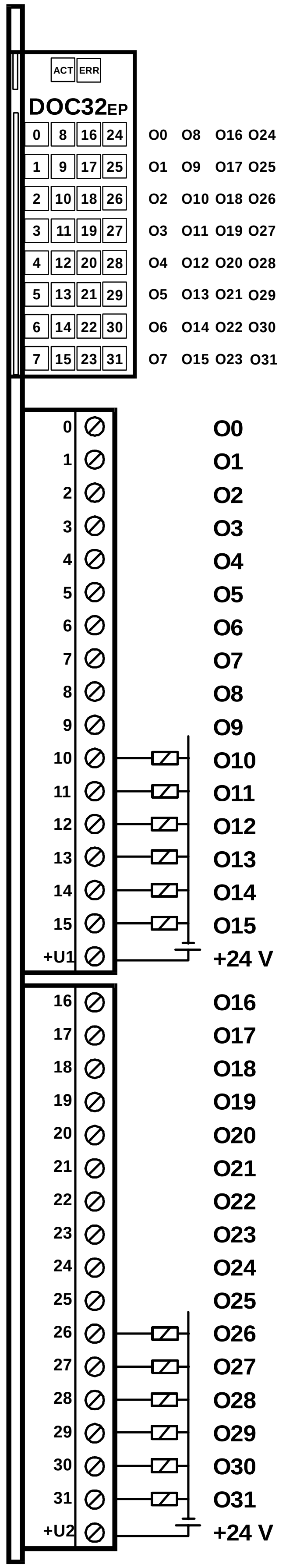
5.2.5 DOC32EP Digital Output Module

The DOC32EP is a digital output module with 32 x 24VDC PNP-type outputs in two groups of 16 outputs each. The current consumption of DOC32EP is 250 mA. The DOC32EP has electronic overload and short circuit protection. The outputs can withstand a load current of 1A each and can be connected in parallel. If connected in parallel the load on each output must not exceed 0.8A. The following LED indicators are located on the front edge of the module:

ACT (green) Module in operation
ERR (red) The operating system has discovered a fault in the module
0-31 (yellow) Inputs I0-I31 are ON

Output wires are connected to the screw terminals as shown below:



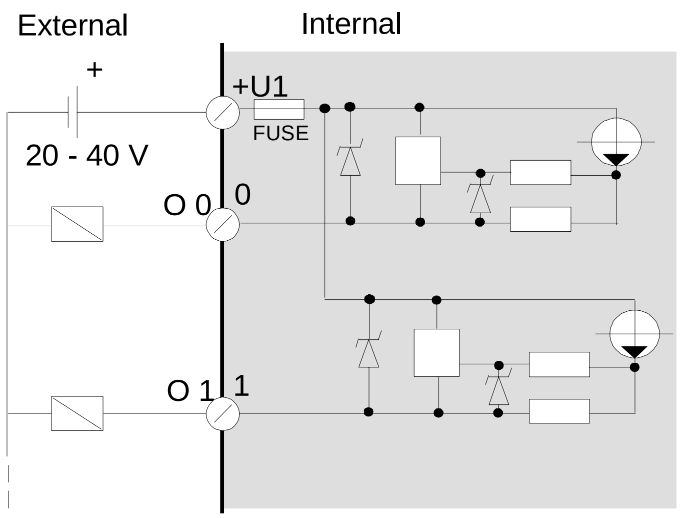


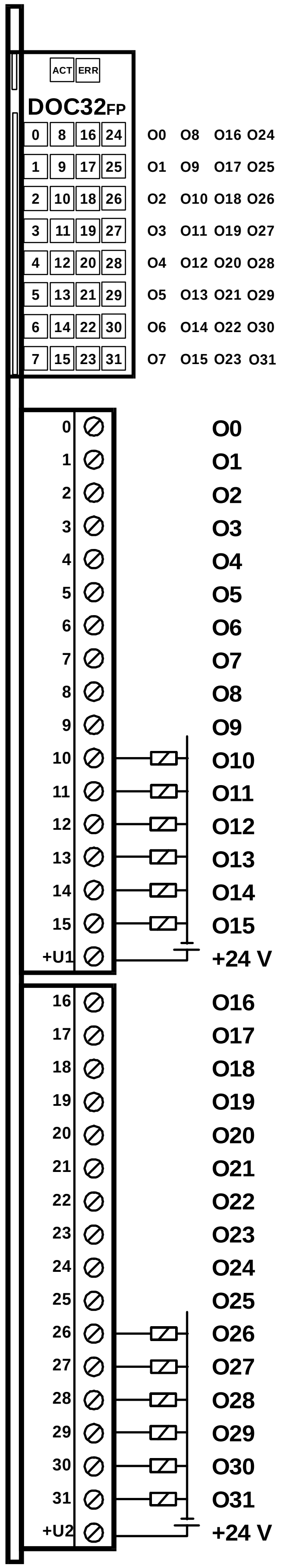
5.2.6 DOC32FP Digital Output Module

The DOC32FP is a digital output module with 32 x 24VDC PNP-type outputs in two groups of 16 outputs each. The current consumption of DOC32FP is 250 mA. The DOC32FP has group overload and short circuit protection (two groups) using 2 x F4A quick-blow fuses located behind the top edge of their respective connectors. The outputs can withstand a load current of 0.5A each, however, the maximum total current for each group must not exceed 4A. The following LED indicators are located on the front edge of the module:

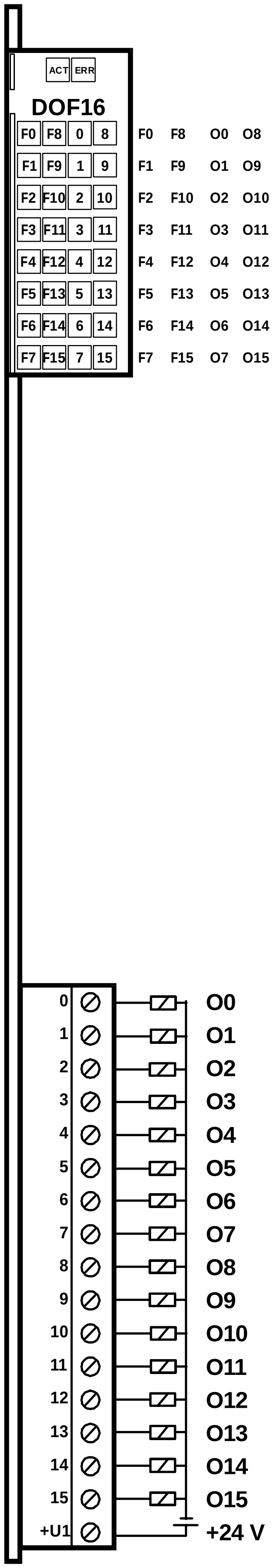
ACT (green) Module in operation
ERR (red) The operating system has discovered a fault in the module
0-31 (yellow) Inputs I0-I31 are ON

Output wires are connected to the screw terminals as shown below:





5.2.7 DOF16 Fault Checking Digital Output Module



The DOF16 is a digital output module with 16 fault checking FET-type outputs isolated as a group. The electronically protected outputs can withstand a load of 1A each and may be connected in parallel (load/output must not exceed 0.8A).

The following LED indicators are located on the front edge of the module:

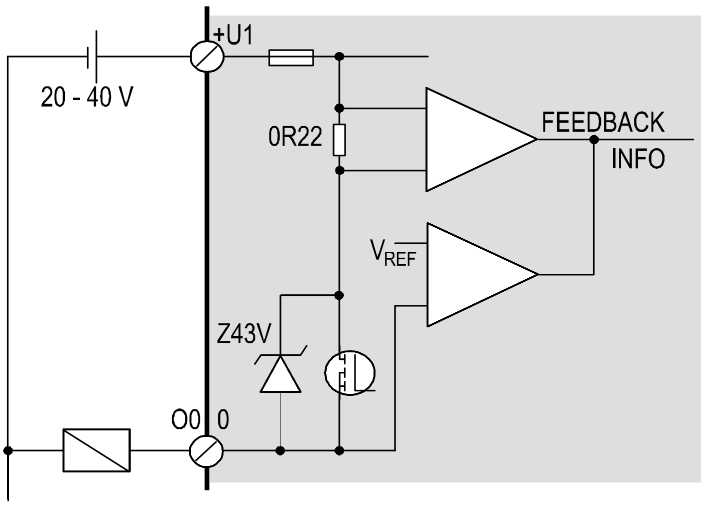
ACT (green)   Module in operation
ERR (red)   The operating system has discovered a fault in the module
0-15 (red)   Outputs O0-O15 are ON
F0-15 (red)   Fault in outputs O0-O15

The statuts of each output can be determined as follows:

Status 0-15	Fault F0-15	Condition
OFF (0)	OFF (0)	Input OFF
ON (1)	OFF (0)	Input ON
OFF (0)	ON (1)	Open / short circuit
ON (1)		
ON (1)	ON (1)	Short circuit to GND/Vcc

These status conditions can be used in the PLC program.

The outputs are connected to as shown in the diagram below:



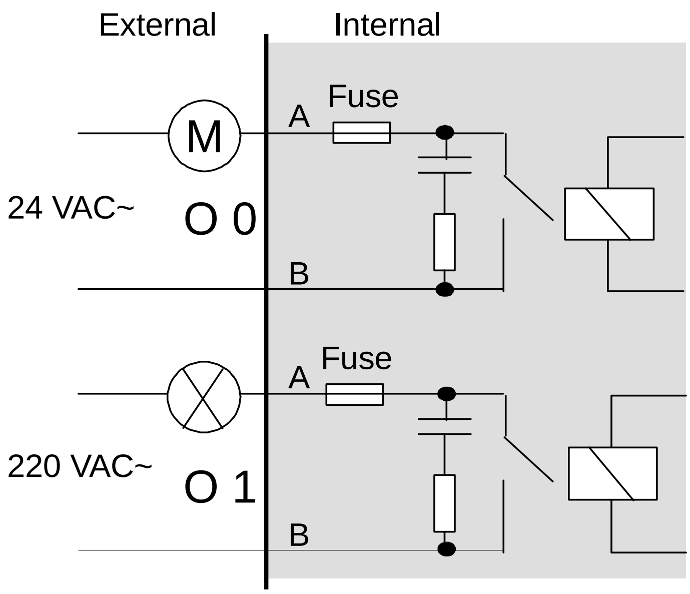
5.2.8 OOC16 Relay Output Module

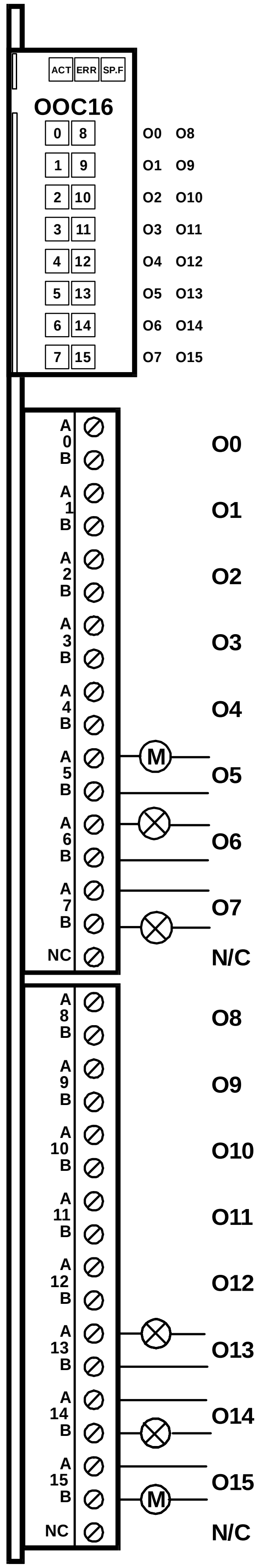
The OOC16 is a digital output module with 16 solid state relay outputs. The current consumption of OOC16 is 250 mA. The voltage range of the module is 20-280VAC with a maximum continuous load of 0.6A. The following LED indicators are located on the front edge of the module:

ACT (yellow/green)	Module is in operation
ERR (red)	The operating system has discovered a fault in the module
SP.F (yellow/green)	The spare fuse is available and working
0-15 (red)	Outputs O0-O15 are ON

Each output is protected by a separate F1.6A quick-blow fuse. The fuses are mounted in sockets located directly behind the output screw terminals and are easily replaceable. In the event of a fuse blowing, a spare fuse is located in a socket directly behind the LED indicators. The SP.F LED on the front edge of the module indicates if the spare fuse is working and mounted in its socket. This provides a simple way to check fuses. A spare fuse should always be installed in the module's spare fuse socket.

The output wires are connected to the screw terminals as shown below:





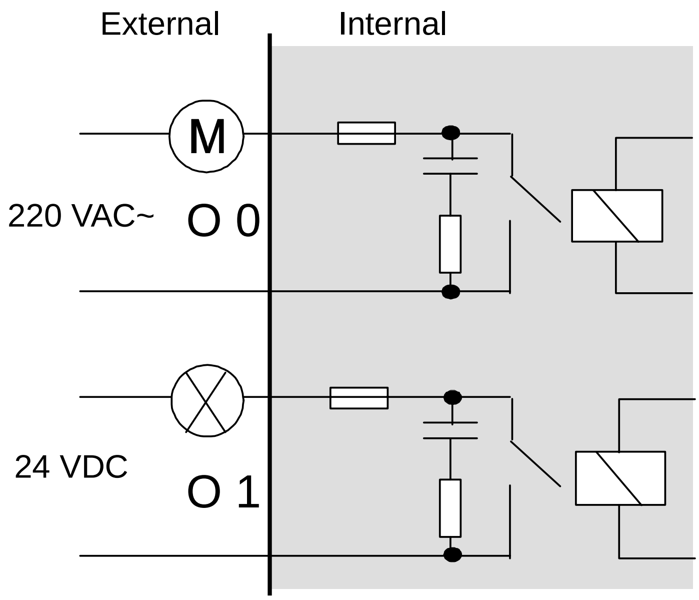
5.2.9 ROC16K Relay Output Module

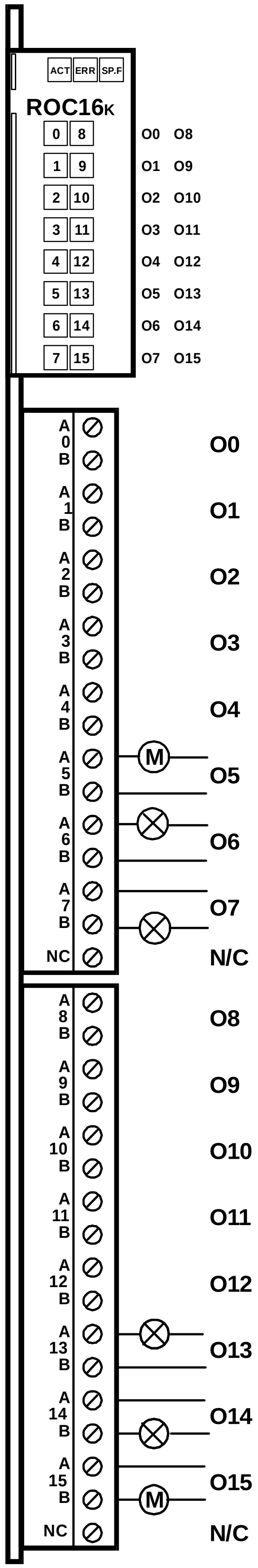
The ROC16K is a digital output module with 16 relay outputs. The current consumption of ROC16K is 500 mA.The outputs are potential free N/O gold-plated relay contacts with a maximum continuous load of 2A making this module ideal for low current signalling applications. The following LED indicators are located on the front edge of the module:

ACT (yellow/green)	Module is in operation
ERR (red)	The operating system has discovered a fault in the module
SP.F (green)	The spare fuse is available and working
0-15 (red)	Outputs O0-O15 are ON

Each output is protected by a separate T2.5A slow-blow fuse. The fuses are mounted in sockets located directly behind the output screw terminals and are easily replaceable. In the event of a fuse blowing, a spare fuse is located in a socket directly behind the LED indicators. The SP.F LED on the front edge of the module indicates if the spare fuse is working and mounted in its socket. This provides a simple way to check fuses. A spare fuse should always be installed in the module's spare fuse socket.

The output wires are connected to the screw terminals as shown below:





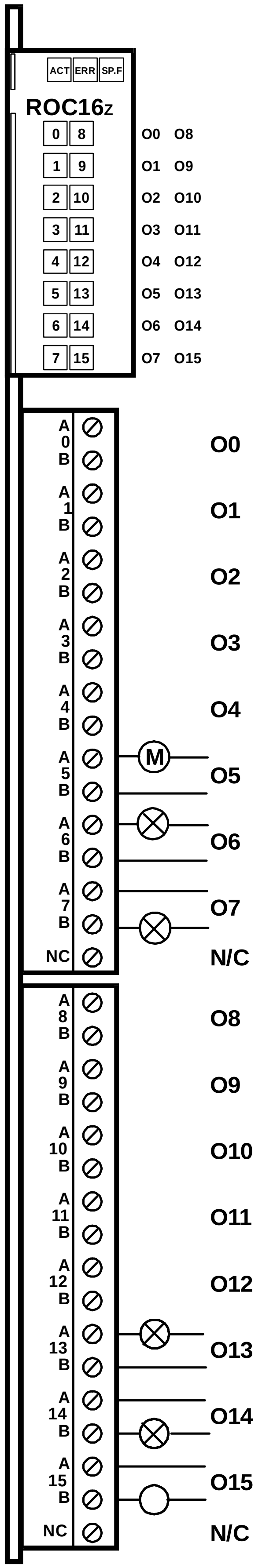
5.2.10 ROC16Z Relay Output Module

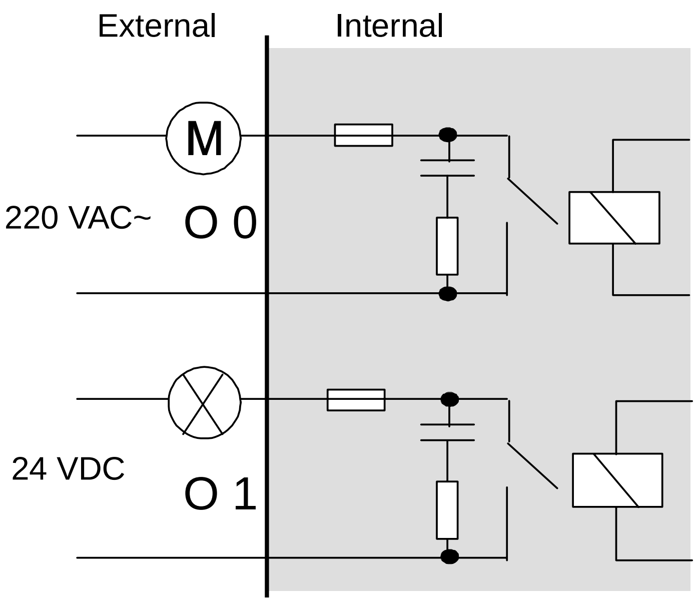
The ROC16Z is a digital output module with 16 relay outputs. These are potential free N/O relay contacts with a maximum continuous load of 4A. The following LED indicators are located on the front edge of the module:

ACT (yellow/green)	Module is in operation
ERR (red)	The operating system has discovered a fault in the module
SP.F (green)	The spare fuse is available and working
0-15 (red)	Outputs O0-O15 are ON

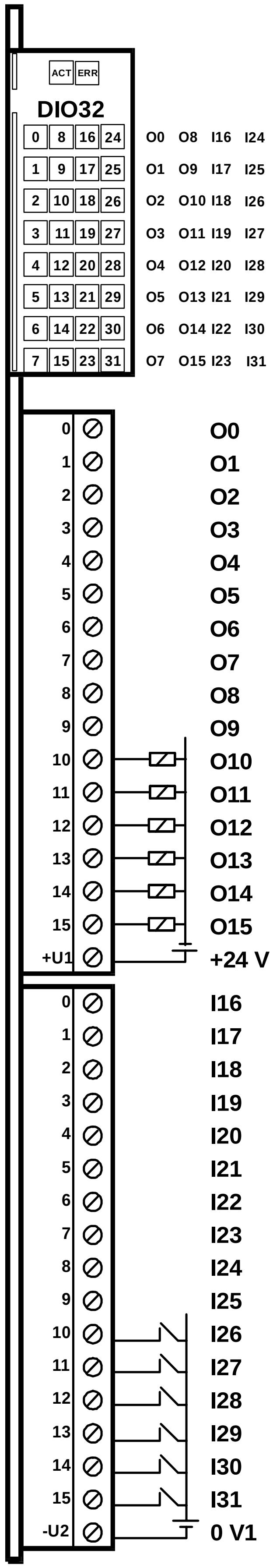
Each output is protected by a separate T5A slow-blow fuse. The fuses are mounted in sockets located directly behind the output screw terminals and are easily replaceable. In the event of a fuse blowing, a spare fuse is located in a socket directly behind the LED indicators. The SP.F LED on the front edge of the module indicates if the spare fuse is working and mounted in its socket. This provides a simple way to check fuses. A spare fuse should always be installed in the module's spare fuse socket.

The output wires are connected to the screw terminals as shown below:





5.2.11 DIO32 Digital Input/Output Module



The DIO32 is hybrid module combining the half of the DIC32DC and half of the DOC32EP modules to give 16 isolated digital inputs and 16 digital outputs. The current consumption of DIO32 is 250 mA. The outputs have electronic overload and short circuit protection and can withstand a load current of 1A each. Outputs may be connected in parallel but the current on each output must not exceed 0.8A. The total sum current is thus for 4 parallel connected output

$\Sigma I = 4 * 0.8 \text{ A} = 3.2\text{A}$.

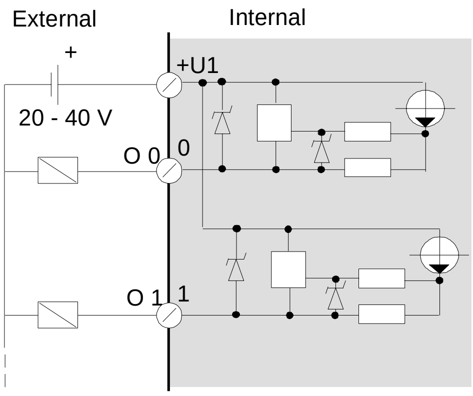
The input logic range of the DIO32 module is:

logic 0	-30 to +5VDC (0-1.1mA)
logic 1	+13 to +36VDC (4-11mA)

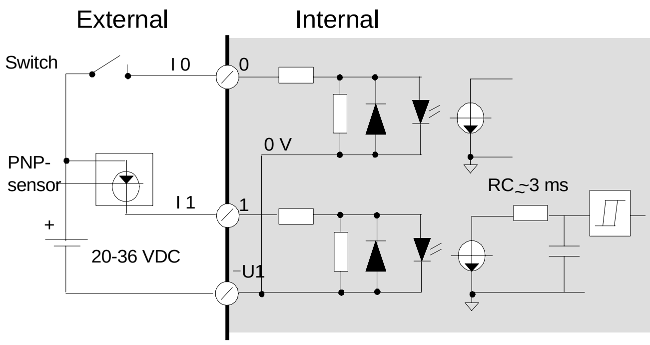
The sensor must be of PNP output type or voltage supplying.The following LED indicators are located on the front edge of the module:

ACT (green)	Module is in operation
ERR (red)	The operating system has discovered a fault in the module
0-15 (red)	Outputs O0-O15 are ON
0-15 (yellow)	Inputs I0-I15 are ON

Output Connections



Input Connections



5.3 Analog I/O Modules

All analog I/O modules have multiple-range input/outputs and are fitted with detachable screw terminals for I/O wiring. These terminals can be removed from the module without disconnecting the wiring. There are two labelling conventions for the screw terminals depending on whether the terminal is an input or an output:

Input	Black text on a white background
Output	White text on a black background

The numbers of the inputs or outputs are shown on both the screw terminals and any LED indicators provided, these being decimal numbers from 0-7. Analog I/O's must be configured using the ALPro software. Analog inputs and outputs are defined as follows:

Inputs:	W I <slot number>.<input number>
Outputs:	W O <slot number>.<output number>
Example:	W I 2.4 (analog input number 4 on the module in slot number 2)
	W O 6.3 (analog output number 3 on the module in slot number 6)

When an Analog Input or Output is defined the type of input or output has to be first defined in ALPro. After this definition, the approriate connections should be made preferably using twisted pairs (see diagrams on the following pages).

All modules are pre-calibrated at the factory and should require no further adjustment for normal use. However, if extremely accurate measurements are required, the user should re-calibrate using the ALPro software. ALPro also allows the user to perform calibration on-line.

5.3.2 AIO74 Analog Input/Output Module

The AIO74 is a combined analog input/output module with 4 x 12-bit analog outputs and 7 x 12-bit differential or 14 x 12-bit non-differential analog inputs. 500 VDC group isolation is provided by opto-isolators and DC/DC converters. The maximum common mode voltage for the AIO74 is 5V. The current consumption of AIO74 is 600 mA.
The ranges of the AIO74 are as follows:

Inputs:

Voltage		
0-480mV	(0-4000)	8.3 bits/mV
0-5V	(0-4000)	800 bits/V
0-10V	(0-4000)	400 bits/V
±10V	(0-4000)	200 bits/V
Current		
0(4)-20mA	(0-4000)	200 bits/mA
$Pt100_{385}$		
(-50 to 150ºC)	(0-4000)	20 bits/ºC
(-200 to 730ºC)	(0-3720)	4 bits/ºC
KTY10		
(-50 to 150ºC)	(0-2000)	10 bits/ºC
$Pt100_{391}$		
(-200 to 730ºC)	(0-3720)	4 bits/ºC
Cu50		
(-200 to 200ºC)	(100-900)	2 bits/ºC

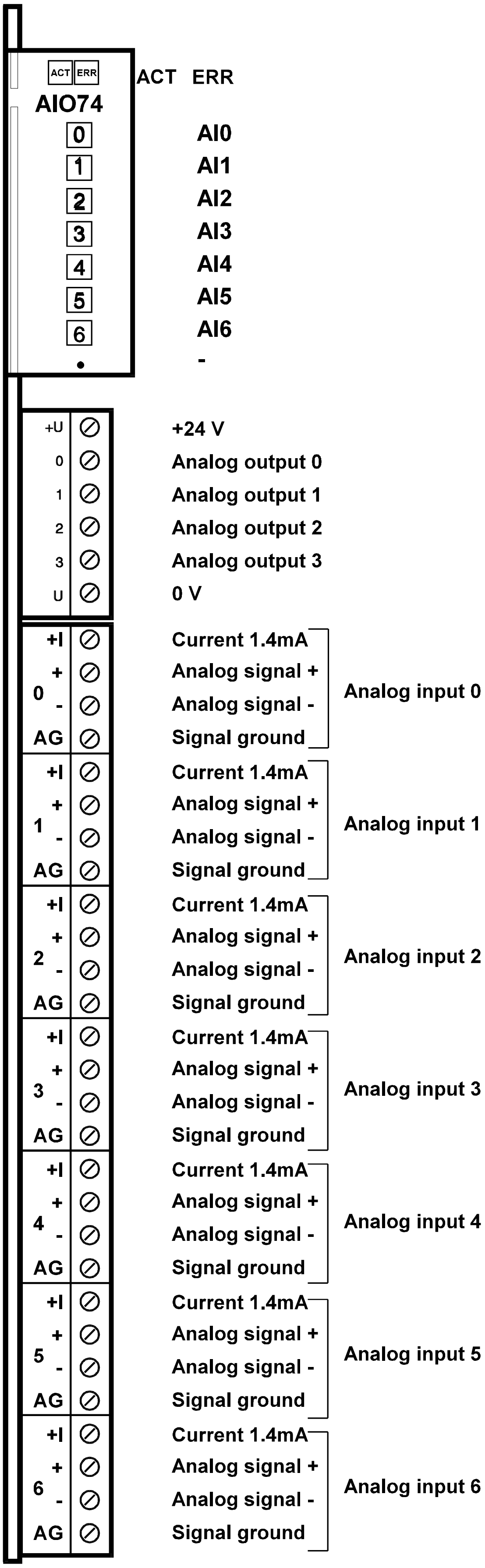
Outputs:

0(4) -20 mA	(0-4000)	200 bits/mA

A constant measuring current of 1.4mA is provided for each input.

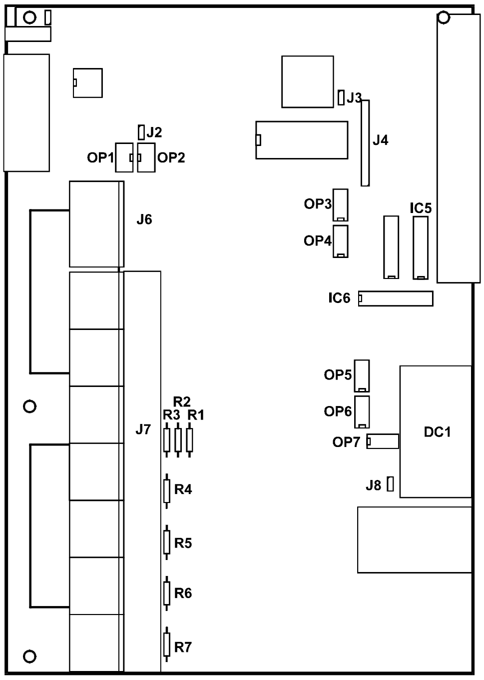
The following LED indicators are located on the front edge of the module:

ACT (green) Module is in operation
ERR (red) The operating system has discovered a fault in the module
0-6 (red) Status of analog inputs AI0-AI6
STEADY LIGHT: input not scaled
FLASHING LIGHT: input at upper/lower limit or not connected
NO LIGHT: input in use and OK

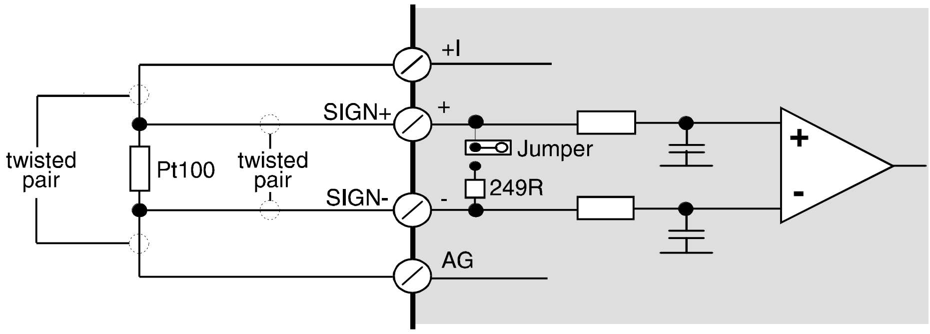


Layout of the AIO74

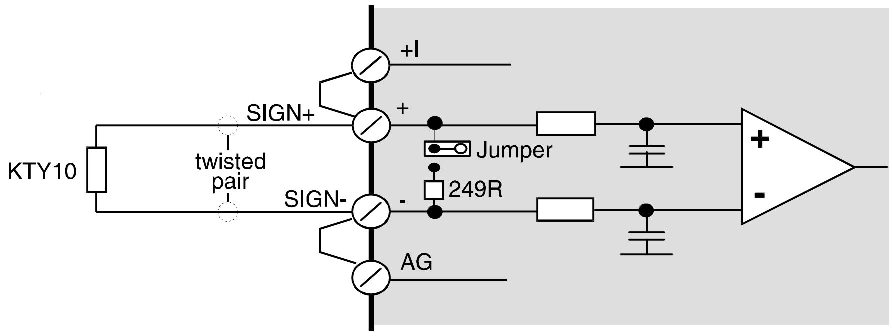
- IC5 Internal data bus buffer
- IC6 Module address detection circuit
- LD1 LED-unit (active/error indication)
- LD2 LED-unit (inputs status I0-I6)
- J5 Internal bus connector
- J6 Screw terminals for outputs
- J7 Screw terminals for inputs
- R1-7 Resistors for current to voltage conversion
- DC1 Voltage converter
- DC2 Voltage converter



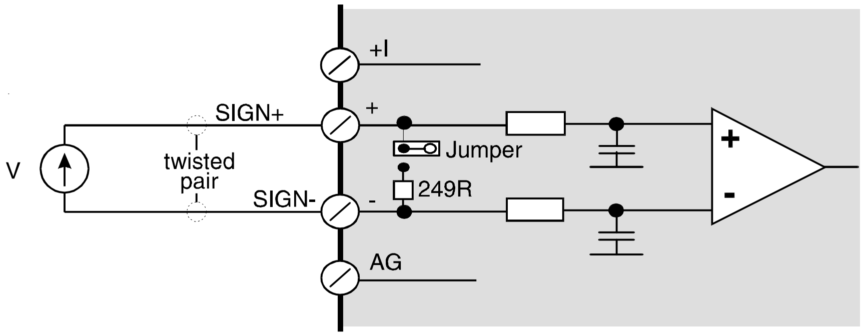
Temperature Measurement with Pt100



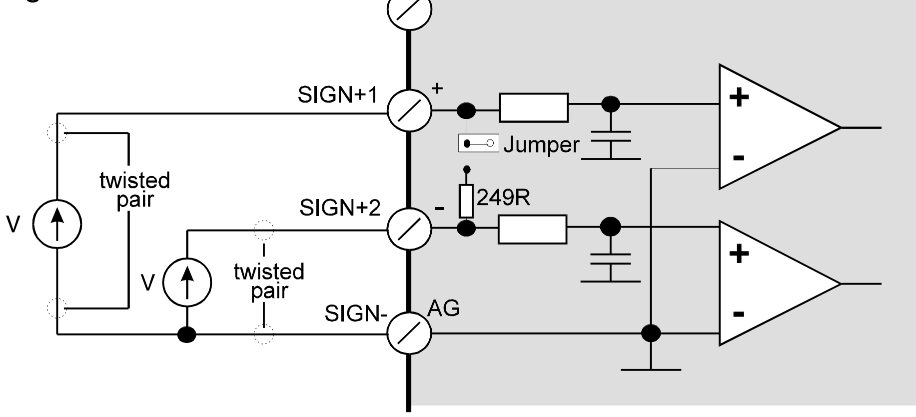
Temperature Measurement with KTY10



Differential Voltage Measurement

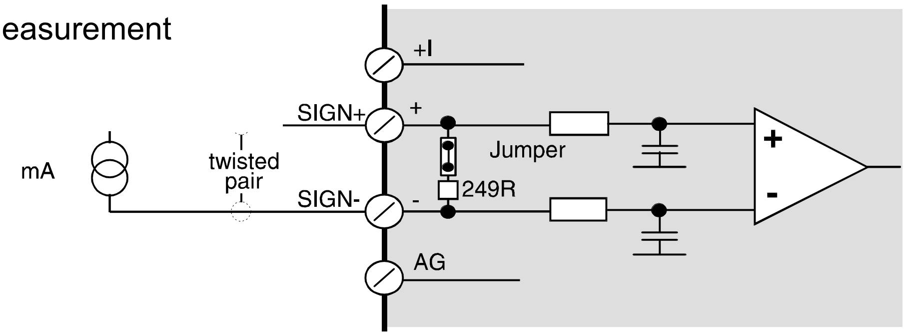


Non-Differential Voltage Measurement

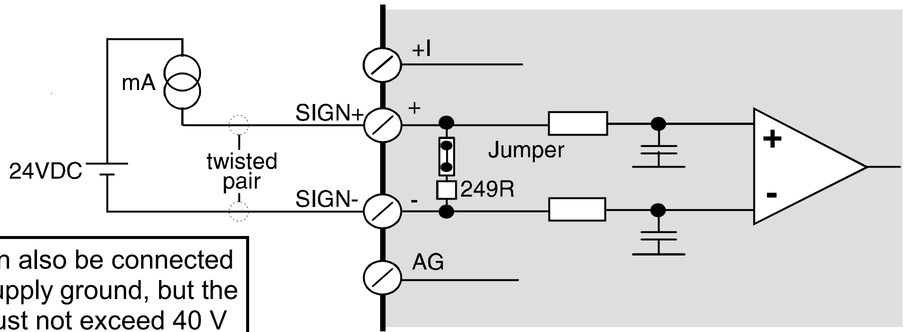


Differential Current Measurement

Active Transducer

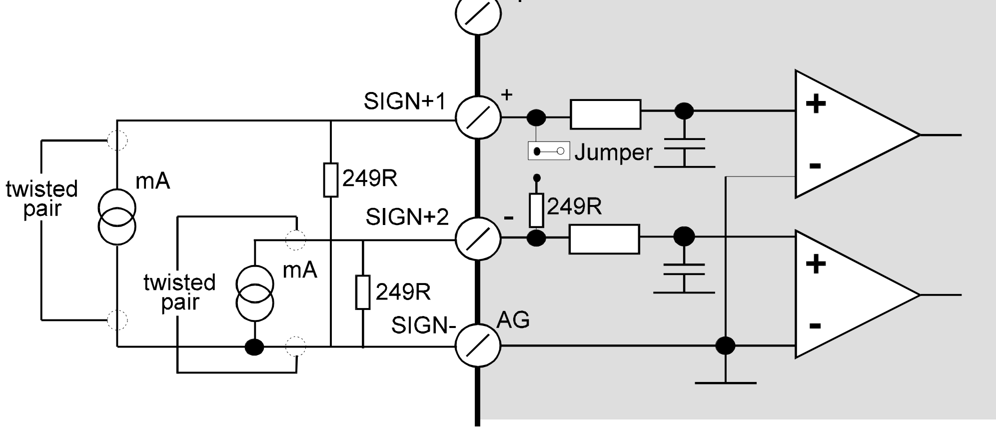


Passive Transducer

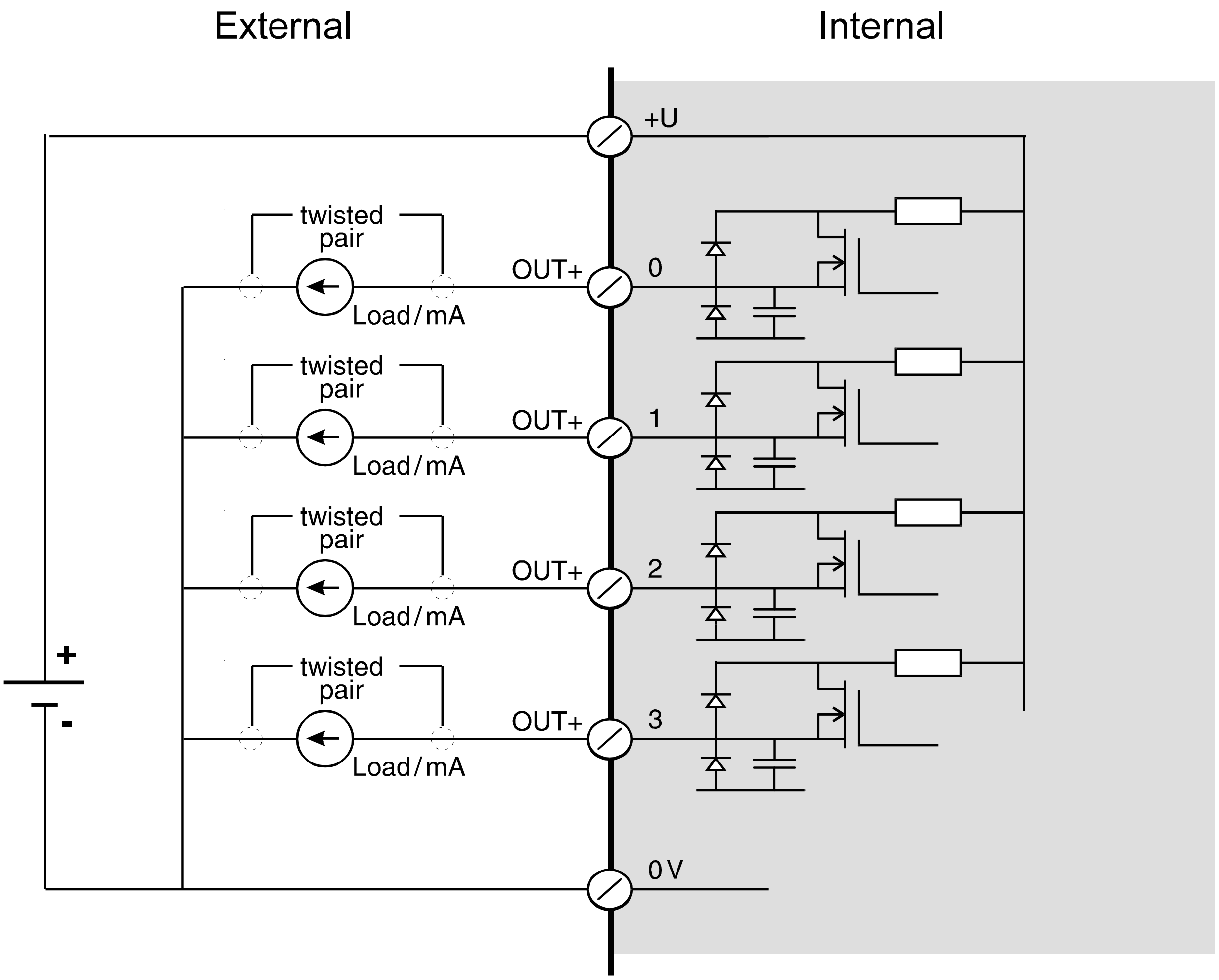


NOTE: The transducer can also be connected between SIGN- and the supply ground, but the common mode voltage must not exceed 40 V

Non-Differential Current Measurement



Connection Example of Analog Outputs



5.3.3 AIC8 Analog Input Module

The AIC8 is an analog input module with 8 x 12-bit differential or 16 x 12-bit non-differential analog inputs isolated as a group. The maximum common mode voltage for the AIC8 is 40V. The current consumption of AIC8 is 250 mA.
The ranges of the AIC8 are as follows:

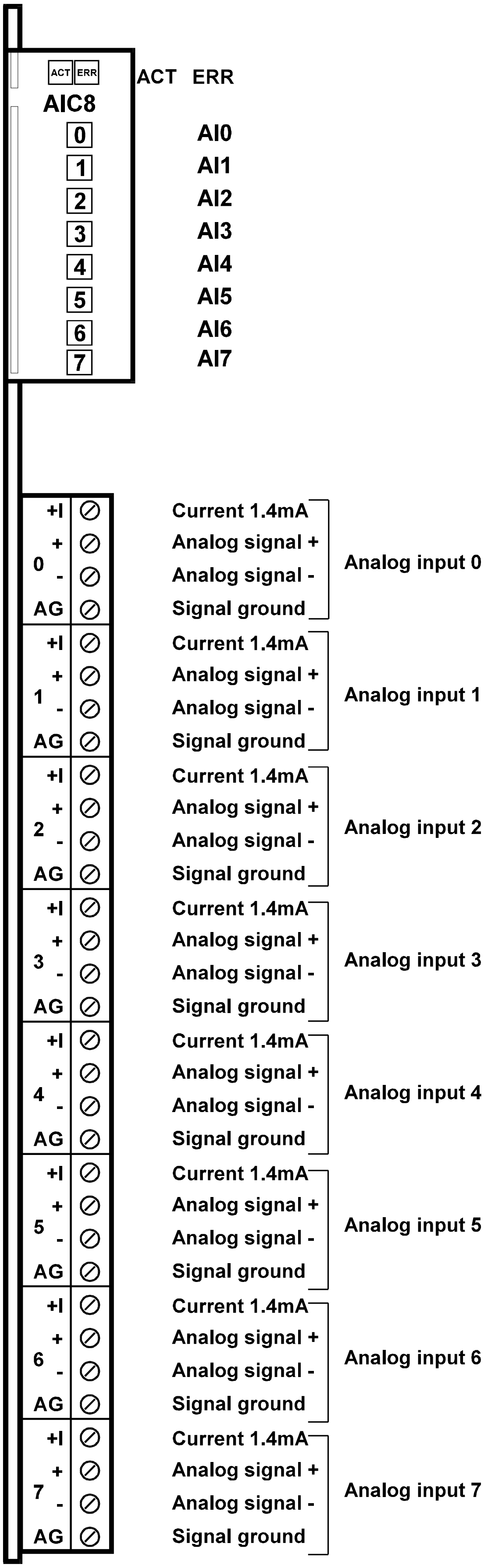
Inputs:

Voltage		
0-480mV	(0-4000)	8.3 bits/mV
0-5V	(0-4000)	800 bits/V
0-10V	(0-4000)	400 bits/V
±10V	(0-4000)	200 bits/V
Current		
0(4)-20mA	(0-4000)	200 bits/mA
$Pt100_{385}$		
(-50 to 150ºC)	(0-4000)	20 bits/ºC
(-200 to 730ºC)	(0-3720)	4 bits/ºC
KTY10		
(-50 to 150ºC)	(0-2000)	10 bits/ºC
$Pt100_{391}$		
(-200 to 730ºC)	(0-3720)	4 bits/ºC
Cu50		
(-200 to 200ºC)	(100-900)	2 bits/ºC

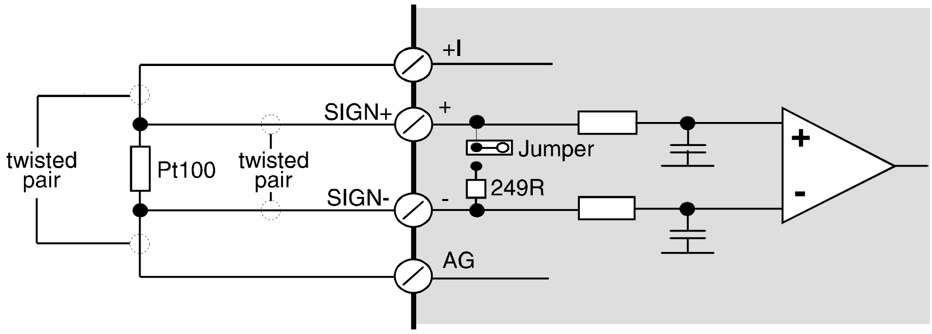
A constant measuring current of 1.4mA is provided for each input.

The following LED indicators are located on the front edge of the module:

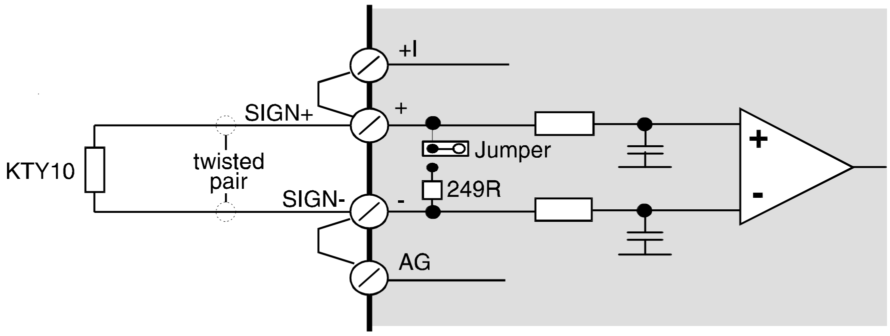
ACT (green) Module is in operation
ERR (red) The operating system has discovered a fault
in the module
0-7 (red) Status of analog inputs AI0-AI6
STEADY LIGHT: input not scaled
FLASHING LIGHT: input at upper/lower limit or not connected
NO LIGHT: input in use and OK



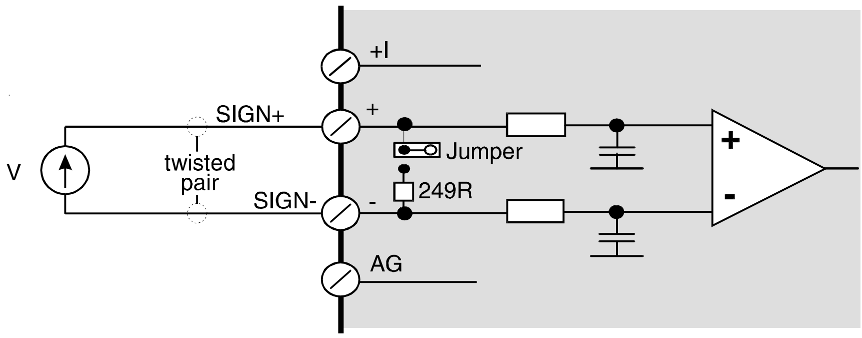
Temperature Measurement with Pt100



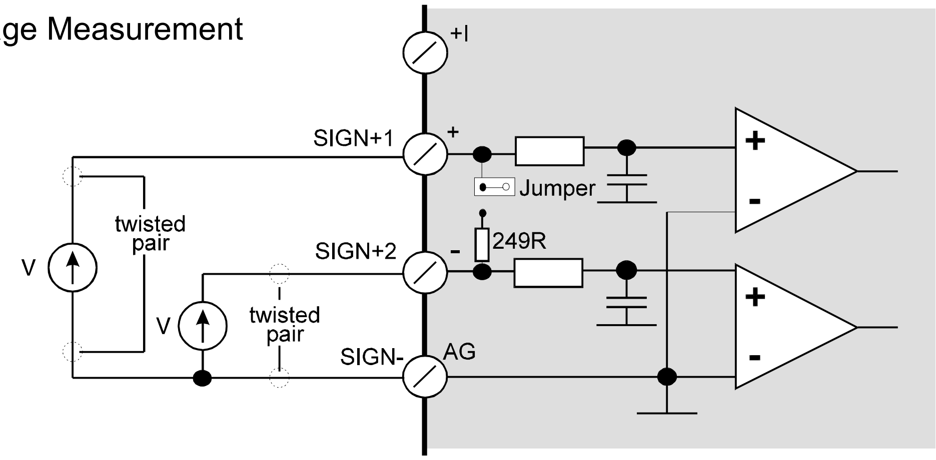
Temperature Measurement with KTY10



Differential Voltage Measurement

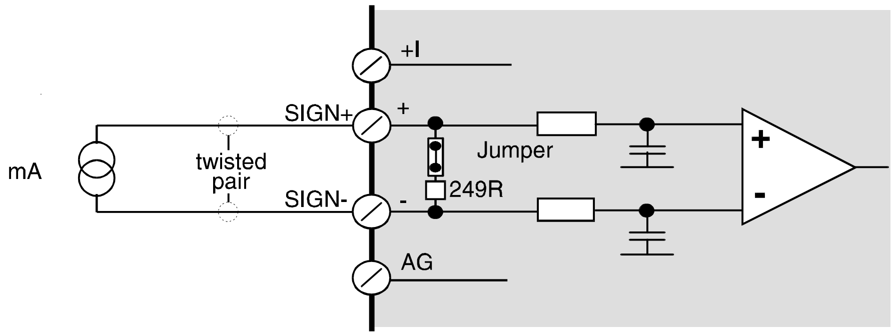


Non-Differential Voltage Measurement

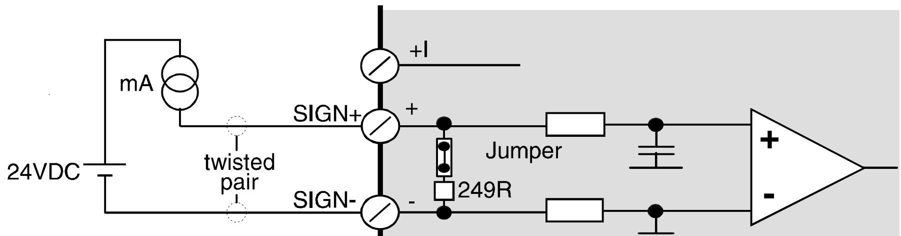


Differential Current Measurement

Active Transducer

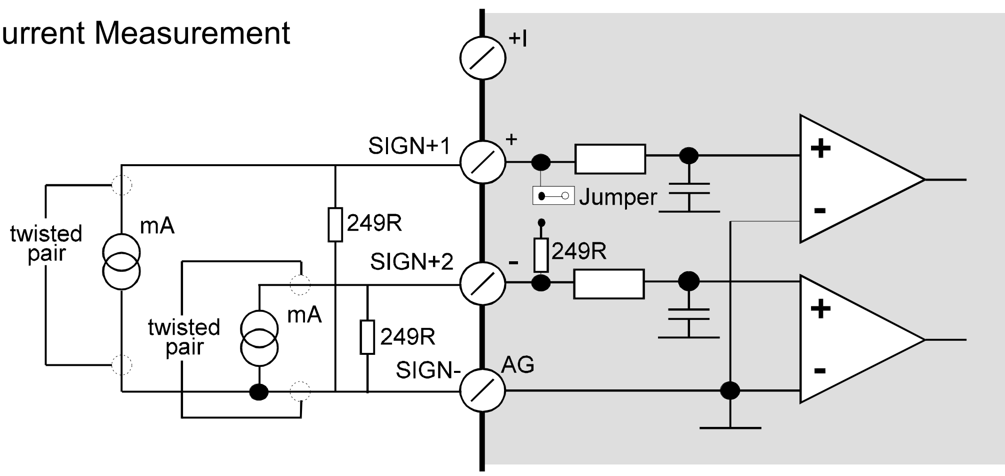


Passive Transducer



NOTE: The transducer can also be connected between SIGN- and the supply ground, but the common mode voltage must not exceed 40 V

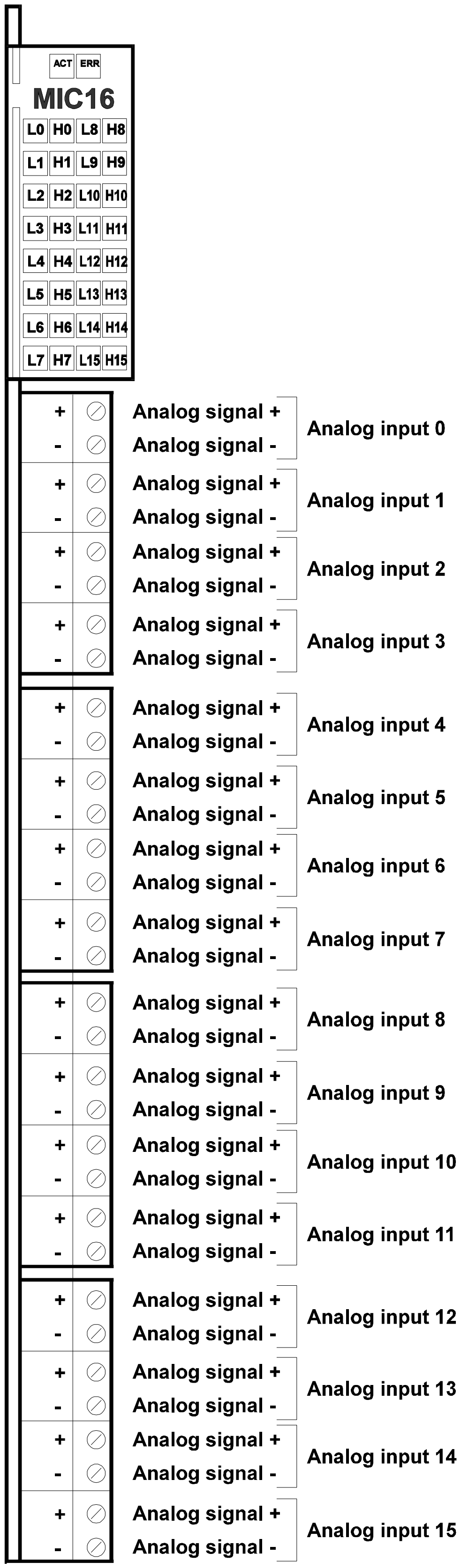
Non-Differential Current Measurement



5.3.4 MIC16 Analog Input Module

The MIC16 is an analog input module with 16 x 13-bit individually isolated analog inputs. The maximum voltage between input channels is 500VDC.
The current consumption of MIC16 is 600 mA. The input ranges of the MIC16 are 4 – 20 mA and the 13-bit raw values are presented in numbers 1600..8000. By deviding the raw value with 2 the raw value limits are equal with other AL2000 analog input boards (800 – 4000).

On the board the mA current is converted to frequency and led to the counter chip. The CPU on MIC16 board reads the frequencys every 0.5 second and calculates the value for analog input based on former value and the change during the last 0.5 second. The calculated value is after that corrected based on individual calibration values for that input.



The following LED indicators are located on the front edge of the module:

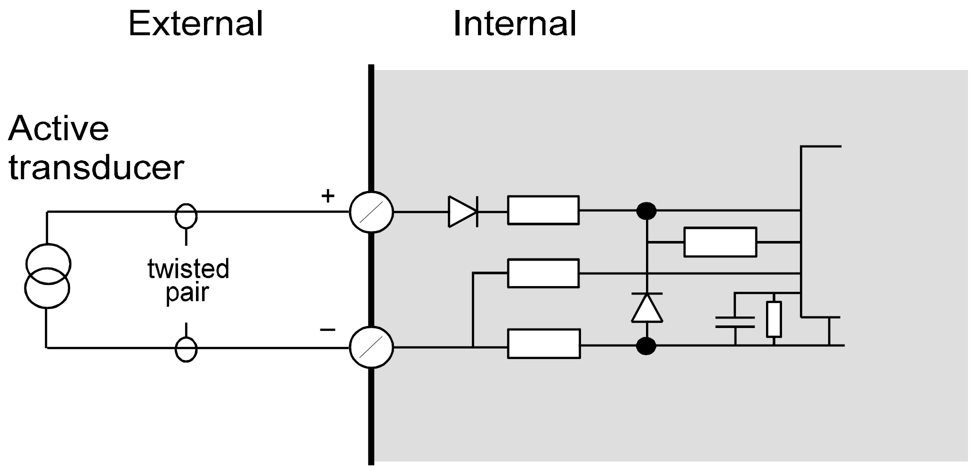
There are 32 indicator LEDs for inputs and
2 LEDs indicating the state of MIC16 board.

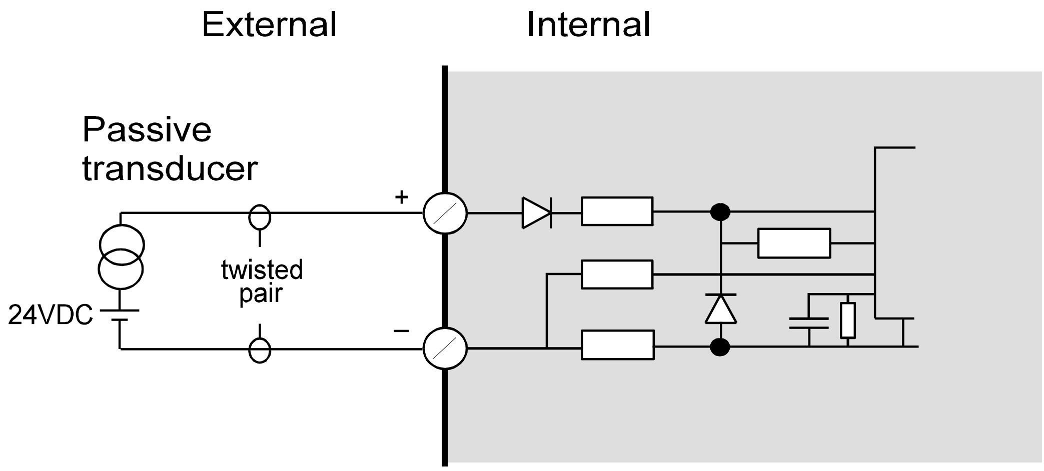
The LEDs for MIC16 state are
ACT (green) Steady light = Module is in normal operation
Blink light = communication to AL2000 CPU doesn't operate correctly
ERR (red) Steady light = AL2000 CPU doesn't get connection to MIC16 board or the Albus is in reset state.

For every input there are two leds L and H, and they indicate the following states:

	LED L	LED H
Steady light	Input is not calibrated	Input is not connected
Blinks	Input value is near low limit	Input value is near high limit
Dark	Input is calibrated	Input signal is in the range

Examples for connecting mA transducers to MIC16 board.





5.3.5 TIC8 Thermocouple Input Module

The TIC8 is a thermocouple input module with 8 x 12-bit analog inputs (isolated as a group). 500 VDC isolation is provided by opto-isolators and DC/DC converters. The current consumption of TIC8 is 600 mA. The input ranges of the TIC8 are as follows:

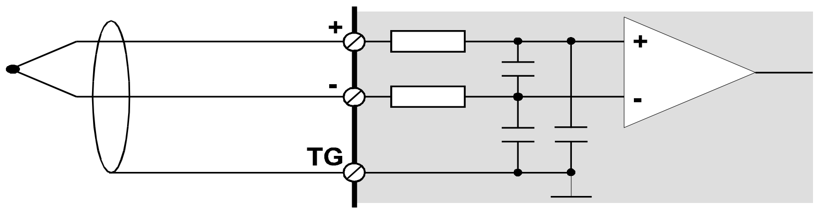
K	-50 to 150°C	(0-4000)	20 bits/°C
K	-250 to 1372°C	(0-3244)	2 bits/°C
T	-50 to 150°C	(0-4000)	20 bits/°C
L	-200 to 730°C	(0-3720)	4 bits/°C

The following LED indicators are located on the front edge of the module:

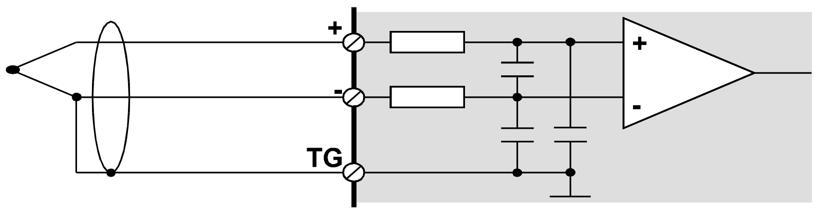
ACT (green)	Module is in operation
ERR (red)	The operating system has discovered a fault in the module
0-7 (red)	Status of analog inputs TI0-TI7 STEADY LIGHT: input not scaled FLASHING LIGHT: input at upper/lower limit or not connected NO LIGHT: input in use and OK

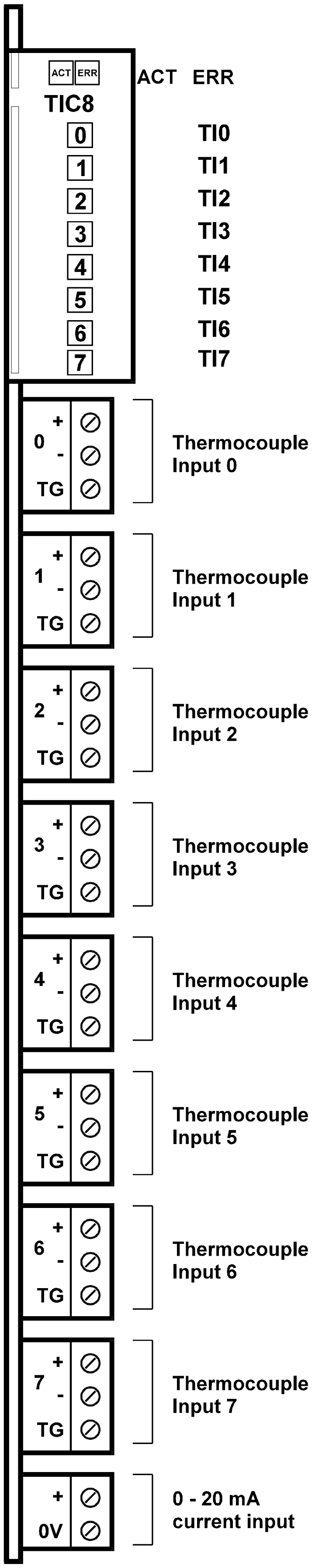
Thermocouples can be connected using normal cable but for longer lengths screened cable is recom-mended.Both earthed and floating thermocouples may be connected to the TIC8 as shown below.

Connections for floating thermocouple



Connections for earthed thermocouple





5.3.6 MOC16 Analog Output Module

The MOC16 is an analog output module with 16 x 12-bit individually isolated analog outputs. The current consumption of MOC16 is 700 mA.

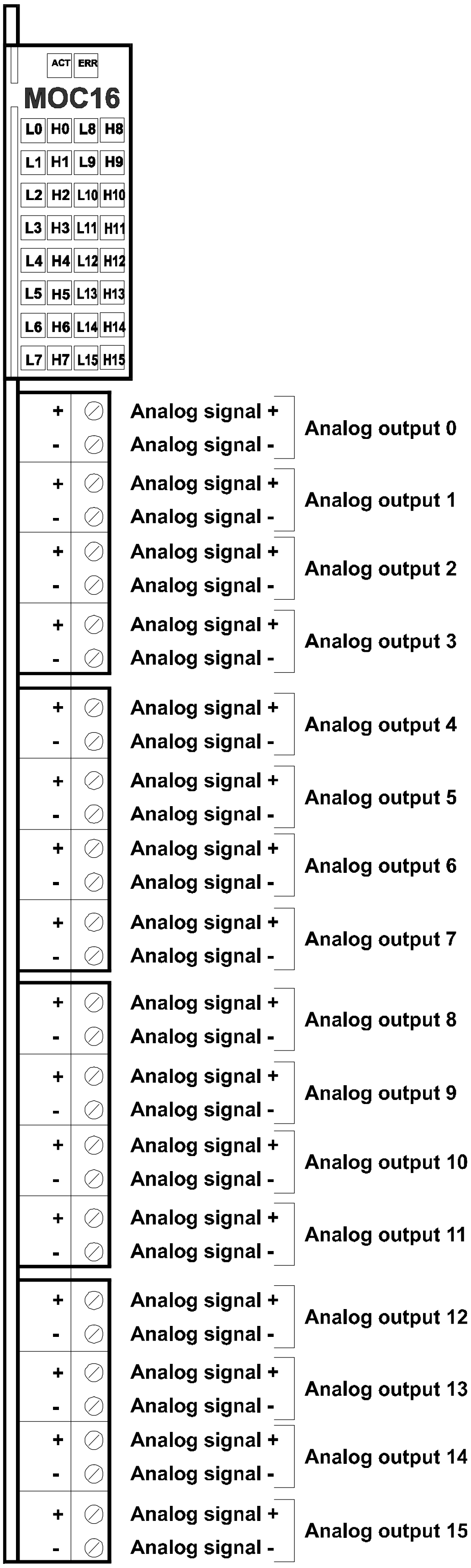
Outputs:

4 -20 mA (0-4000) 200 bits/mA

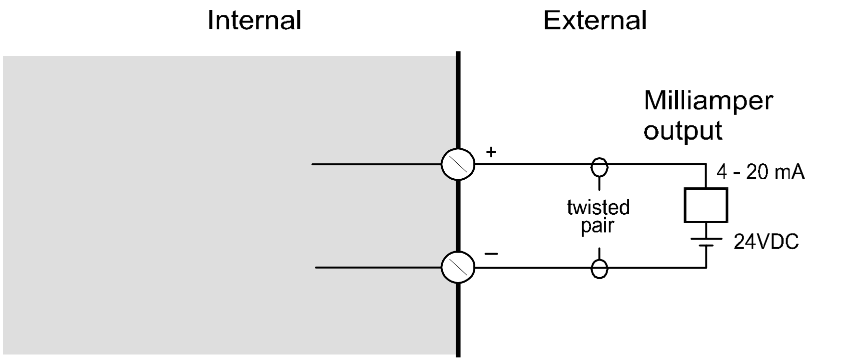
There are 32 indicator LEDs for inputs and 2 LEDs indicating the state of MOC16 board.

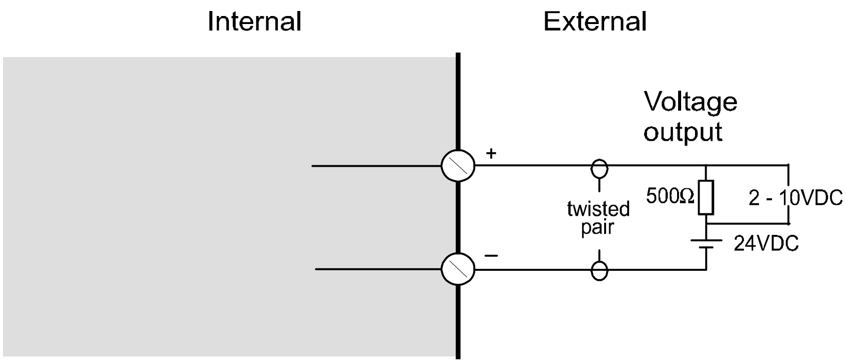
The LEDs for MOC16 state are
ACT (green) Steady light = Module is in normal operation
Blink light = communication to AL2000 CPU doesn't operate correctly
ERR (red) Steady light = AL2000 CPU doesn't get connection to MOC16 board or the Albus is in reset state.

For every input there are two leds L and H, and they indicate the following states:



Examples for connecting mA outputs from MOC16 board.





5.4 Central Processor Units

There are several types of CPU module available for the AL2000. These are the CPU2000S, CPU2000P and CPU2000L. Module CPU2000S, has three serial interfaces, two RS-232C and one RS-485. An I^2C interface for a display/keypad unit and the following LED indicators, located on the front edge of all CPU2000S module:

RUN (green)	see table below
RES (red)	Fault condition
CTS1 (red)	SER1 serial interface CTS handshake
RTS1 (red)	SER1 serial Interface RTS handshake
TX1 (yellow)	SER1 serial interface transmitting
RX1 (yellow)	SER1 serial Interface receiving
CTS2 (red)	SER2 serial interface CTS handshake
RTS2 (red)	SER2 serial Interface RTS handshake
TX2 (yellow)	SER2 serial interface transmitting
RX2 (yellow)	SER2 serial Interface receiving
TX3 (yellow)	SER3 serial interface transmitting
RX3 (yellow)	SER3 serial Interface receiving
BAT (red)	Battery low indicator
VS3 (red)	Fault in SER 3 interface voltage

Run LED	**Program state**	**Reason**	**Measure to be taken**
Steady light or no light.	Execution of the program has stopped. The status of the outputs remain.	The program was stopped using the programming tool.	Start execution of the program with the START (!) command.
Slow blinking 0,5 Hz Pulse width 50%	The program is running	Normal operation	
Combined slow/quick blinking	The program is running	Disturbances in supply voltage. After 4 min the blinking returns to normal.	Check the supply voltage.
Quick blinking Pulse width 90%	The execution of the program has stopped. The status of all outputs is 0.	Too low supply voltage. It has dropped below 17 VDC (180 VAC) and has not exceeded 20 VDC (195 VAC).	Check the supply voltage.
Quick blinking 5 Hz Pulse width 50%	The execution of the program has stopped. The status of all outputs is 0.	There are faults in the program.	Correct the program and re-start execution.
Very quick blinking Pulse width 20%	The execution of the program has stopped. The status of all outputs is 0.	Hardware fault or the STOP command has been erased or the END command was moved during the execution of the program.	Switch off the supply voltage and switch on again. Correct the program and re-start execution.

All CPU2000S series modules are equipped with DIP switches located directly beneath the LED indicators on the front panel of the module. The position of these DIP switches determines the manner in which the module operates. The possible settings for these switches are shown in the table below.

DIP	**ON**	**OFF**
1	Enable writing to FLASH memory	Disable writing to FLASH memory
2	Serial line: (Mode selected by R O 214) - terminal printing - modbus slave - programming device	Serial line: Used for programming

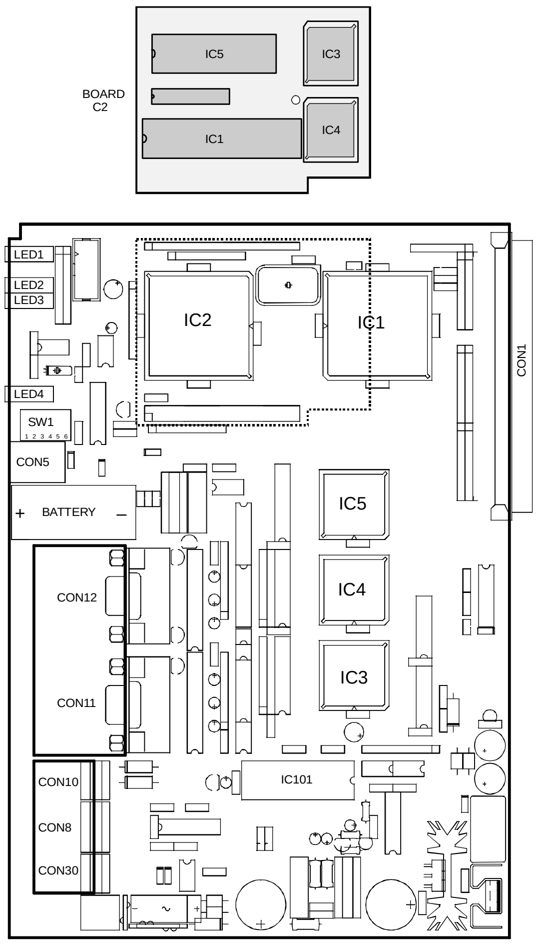
DIP	**300 bd**	**1200 bd**	**9600 bd**	**Rate determined by R O 213**
3 4	OFF OFF	ON OFF	OFF ON	ON ON

DIP	**ON**	**OFF**
5	Contents of data memory erased on power up	Contents of data memory saved in the case of a power supply failure

CPU2000S series module layout

C2_IC1	Processor
C2_IC3	System program FLASH
C2_IC4	Application program FLASH
C2_IC5	Data RAM

IC1	ALBus FPGA
IC2	Memory decoder FPGA
IC3	SER3 Uart
IC4	SER2 Uart
IC5	SER1 Uart
SW1	DIP-switch
BATTERY	Lithium Battery
LD1	LED-unit (operational information)
LD2	LED-unit (SER1)
LD3	LED-unit (SER2)
LD4	LED-unit (SER3)
CON12	SER1 RS-232C serial interface 1
CON11	SER2 RS-232C serial interface 2
CON10	SER3 RS-485 serial Interface 3
CON1	Internal bus connector
CON5	Display unit connector
CON8	
CON30	



Precautions Against Data Loss

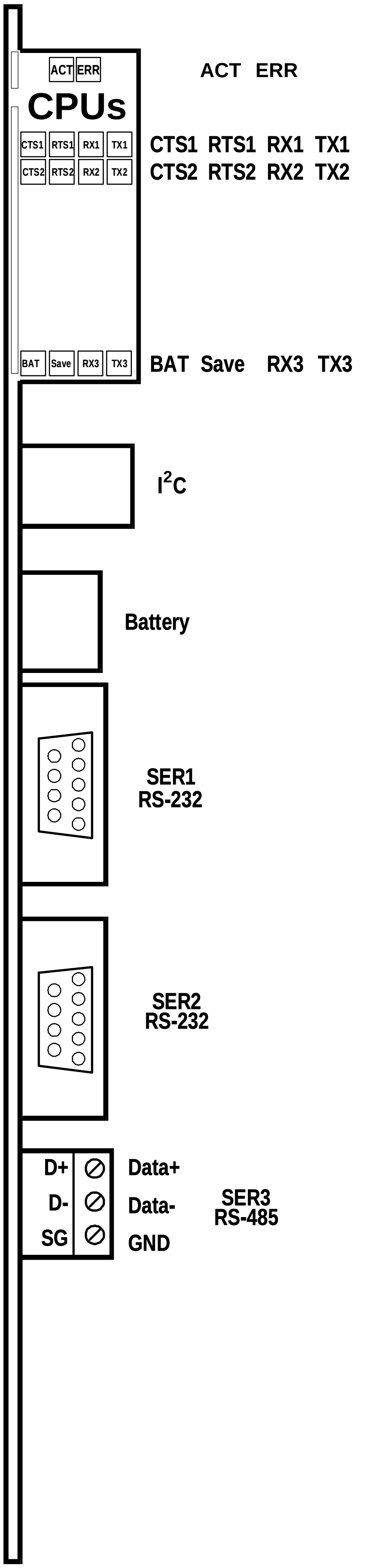
When handling CPU modules, care should be taken not to touch the metallic parts of the PCL rack, this could cause damage to data stored in the RAM.

If the AL2000's working environment is especially noisy or susceptible to violent electrical storms, DIP1 should be set to the OFF position after programming, this protects the contents of the FLASH memory.

5.4.1 CPU2000S Module

The CPU2000S is the basic CPU module. The RS-485 serial interface uses detachable screw terminals for ease of connection. The RS-232 and RS-485 serial ports are isolated in two groups to 500VDC.

The current consumption of CPU2000S is 700 mA at 5 VDC voltage level.



5.4.2 CPU2000P Module

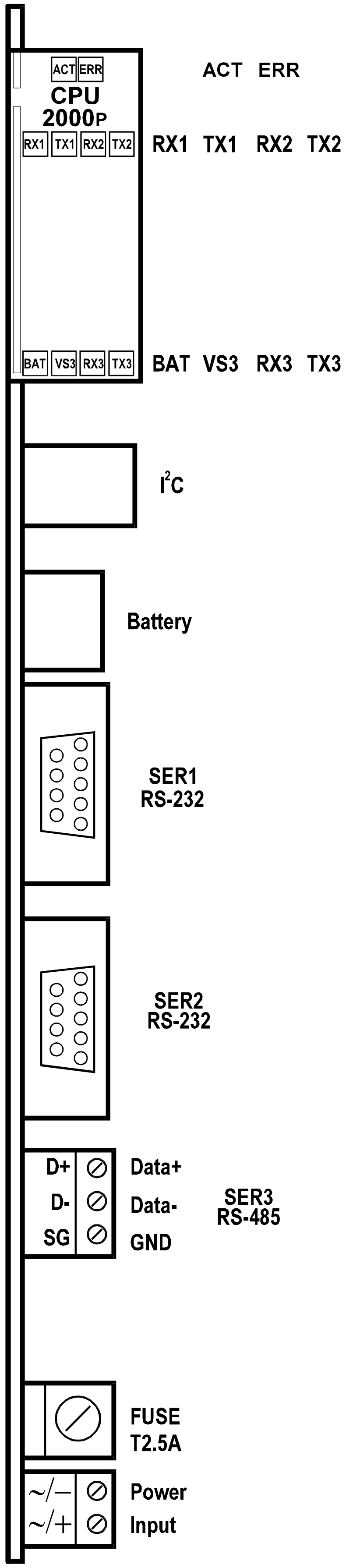
The CPU2000P is a CPU module with two fast counter inputs and a built in power supply. The counter inputs and the RS-485 serial interface use detachable screw terminals for ease of connection. The RS-232 and RS-485 serial ports are isolated in two groups to 500VDC. They are also isolated as a group to 500VDC.

The current consumption of CPU2000P is 600 mA.

This CPU is designed for use with the MR3-mini mounting rack and its built in power supply is capable of driving the I/O modules fitted in this rack. The CPU2000P cannot be used with any other type of mounting rack.

The CPU2000P requires an input voltage in the range:

20-32 VDC or
20-26 VAC



5.4.3 CPU2000L Modules

486 DX4/DX5 version

The CPU2000L is the most powerful CPU module in the AL2000 range. The CPU2000L is based on the 486 DX4/DX5, running up to 133MHz. The module is PC compatible running QNX. The CPU2000L is equipped with connectors for an IBM-AT keyboard, RS232C port, RS232/422/485 port and Ethernet LAN (AUI/ 10BaseT). The board can be equipped with PC104 extension board to connect additional PC104 devices.

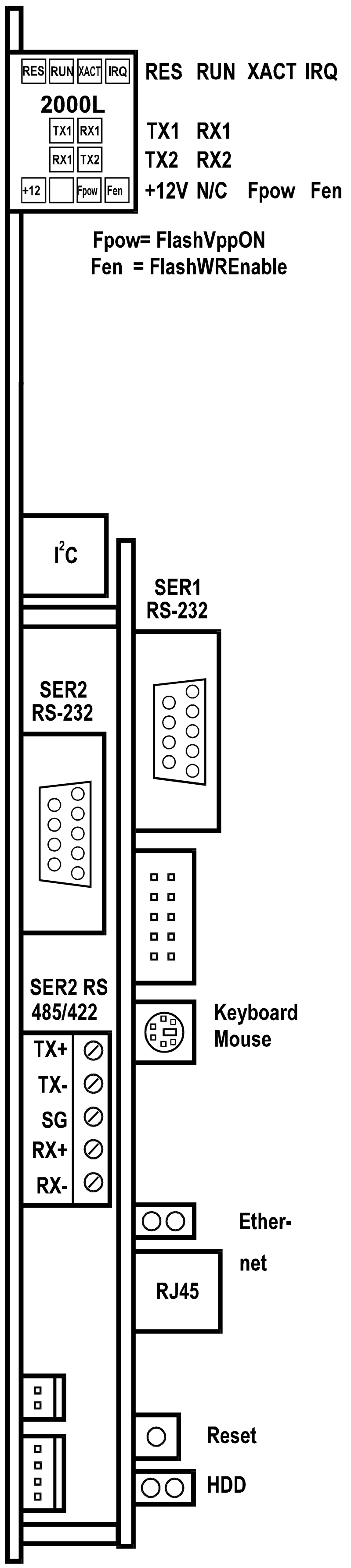
The FLASH disks are available in 16 MB, 24 MB and 48 MB sizes and are ideal for use in poor environmental conditions where standard disk systems would be unreliable.

Pentium 586 version

The CPU2000L is the most powerful CPU module in the AL2000 range. The CPU2000L is based on the 586, running on 233MHz. The module is PC compatible running QNX. The CPU2000L is equipped with connectors for an IBM-AT keyboard, RS232C port, RS232/422/485 port and Ethernet LAN 10/100 (AUI/10BaseT). The board can be equipped with PC104 extension board to connect additional PC104 devices.

The FLASH disks are available in 16 MB, 24 MB and 48 MB sizes and are ideal for use in poor environmental conditions where standard disk systems would be unreliable.

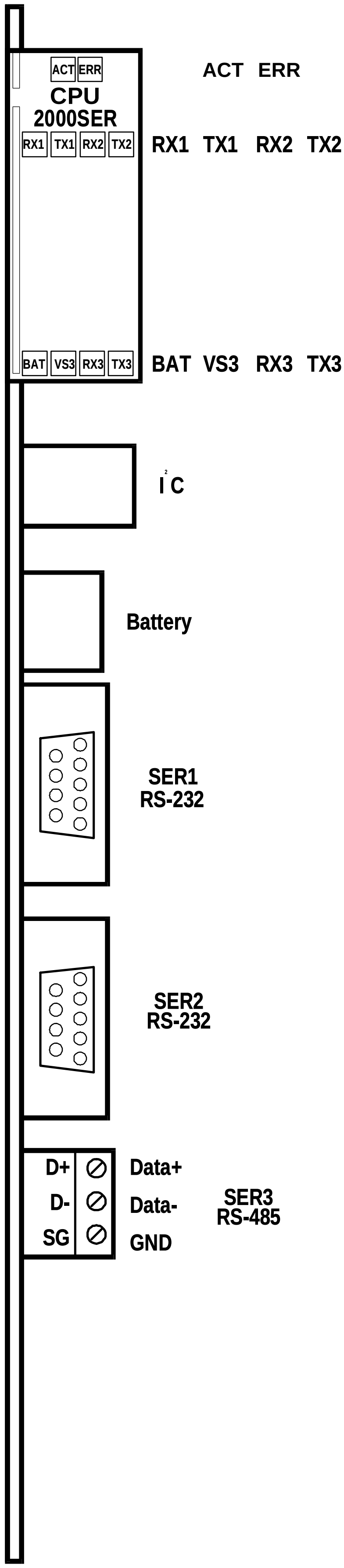
A separate disk module is also available for connection to the CPU2000L. This has space for two standard IDE floppy or hard disk drives and can be installed in the mounting rack alongside the CPU2000L.



5.4.4 CPU2000SER Module

The CPU2000SER is serial communication expansion CPU module. The module is similar with the CPU2000S except that it is not able to control any I/O points of the rack. The module can be installed into any card place in the rack. When AL2000SER module is used, it doubles the number of variables in the rack.
The RS-422/485 serial interface uses detachable screw terminals for ease of connection. The RS-232 and RS-485 serial ports are isolated in two groups to 500VDC.

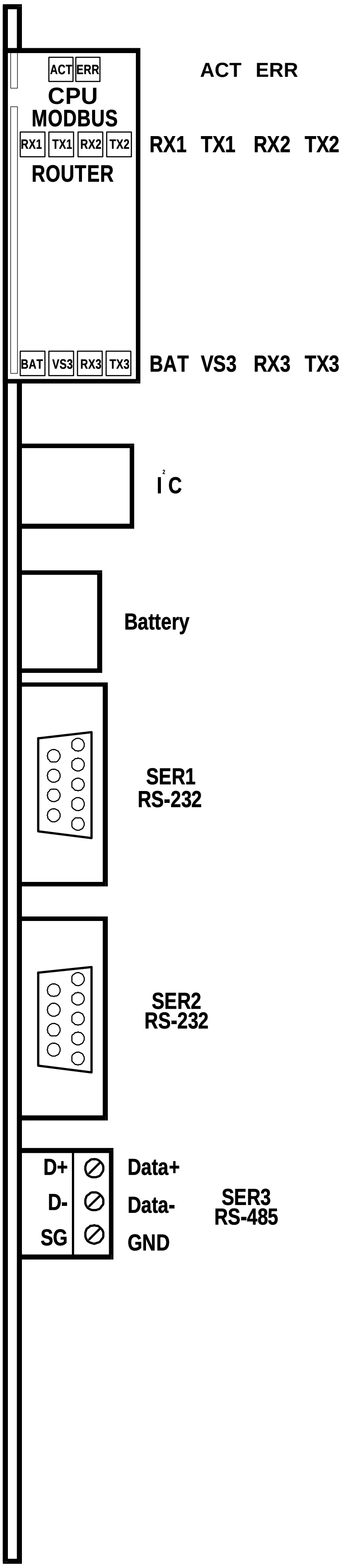
The current consumption of CPU2000SER is 700 mA at 5 VDC voltage level.



5.4.5 CPU Modbus Router Module

The CPU Modbus Router is serial modbus communication CPU module. The module is not able to control any I/O points of the rack and can be installed into any card place in the rack.
The RS-422/485 serial interface uses detachable screw terminals for ease of connection. The RS-232 and RS-485 serial ports are isolated in two groups to 500VDC.

The current consumption of CPU Modbus Router is 700 mA at 5 VDC voltage level.



TECHNICAL FEATURES	
Mode	Modbus Master, Slave, Not used
Serial connections	3, isolated in 2 blocks (500 VDC)
Ser1 (9 pin D type)	RS-232, 300...19200 Bd, Modbus master/slave
Ser2 (9 pin D type)	RS-232, 300....19200 Bd, ModBus master/slave
Ser3 (3 screw terminal)	RS-485, 300...28800 Bd, ModBus master/slave
Addresses	Address / Address block
Retransmissions	0 - 7
On error	Enabled, Disabled
Send delay	0 - 250 ms
Responce timeout	100 - 1500 ms
Serial channel diagnostics	Acknowledged messages Warnings Rejected messages Error code and slave address

6. CONTROLLERS

6.1 8-Bit Controllers

The AL 2000 PLC system software includes 8 x 8-bit direct digital controllers (DDC) with PID characteristics. The controller parameters are held in R GM's:

	Controller 0	Controller 1	Controller 2		Controller 7
Mode	0	8	16	-	56
Actual value	1	9	17	-	57
Set point	2	10	18	-	58
D term	3	11	19	-	59
I term	4	12	20	-	60
Gain term	5	13	21	-	61
Output	6	14	22	-	62
Aux.	7	15	23	-	63

Mode (R GM 0,8,16 etc.) gives the control program the following information:

0 Controller not in use
1 Controller in automatic mode
3 Controller in manual mode

Writing 1 to register memory R GM 8 (mode) would activate controller 1 in automatic mode. The contents of register memory R GM 14 (output) will then be generated in accordance with the control algorithm and parameters. Three-point control outputs are also written to register outputs R O 192 and R O 196. The required pulse interval can be written to register output R O 200.

Writing 3 to register memory R GM 8 (mode) would activate controller 1 in manual mode. In this mode the controller program computes the control algorithm once and then leaves the controller in manual mode. This allows the user to use greater control intervals. The controller program continuously monitors the output, so returning to automatic mode does not cause steps or spikes in the controller output.

If a controller is deactivated by writing 0 to its mode register memory, the remaining register memories for that controller become available for other purposes. For example, deactivating controller 2 by writing 0 to R GM 16 would leave register memories R GM 17-23 available for other purposes.

6.1.1 Control Algorithm

The following control algorithm is used:

DY=P*{e(ti)-e(ti-1)+D*[e(ti)-2e(ti-1)+e(ti-2)]+e(ti)/I} Where P = gain (0-1)
D = differentiation time constant
e = error
I = integration time constant

6.1.2 Effect of the gain, P

P is a value 0-255 corresponding to gains of 0-1 (128 corresponds to a gain of 0.5).

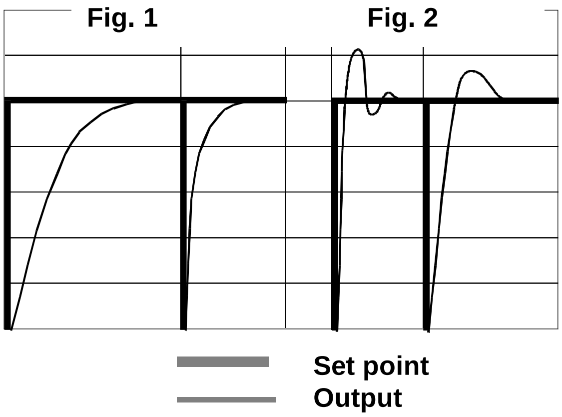


Figure 1

P = 100	P = 150
I = 20	I = 20
D = 2	D = 2

Figure 2

P = 100	P = 20
I = 2	I = 2
D = 2	D = 2

Figure 1 shows how increasing the gain yields faster control oscillation caused by short

Figure 2 shows how reducing the gain stabilises the action. integration time.

The sum of the proportional, derivative and integral terms is multiplied by the gain, P. Thus the gain, P, strengthens or weakens the effects of these coefficients.

6.1.3 Effect of the integration time constant, I

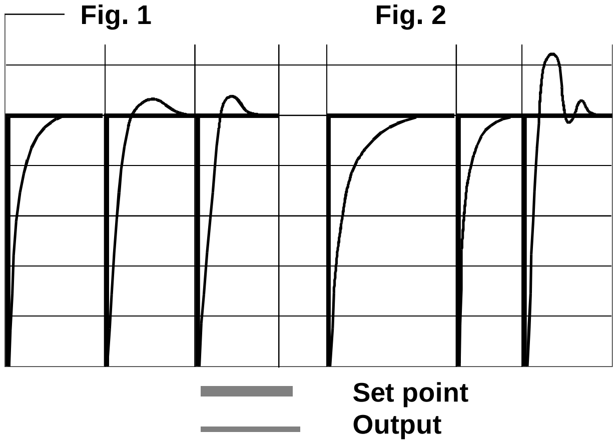


Figure 1

P = 150	P = 150	P = 150
I = 120	I = 12	I = 2
D = 2	D = 2	D = 2

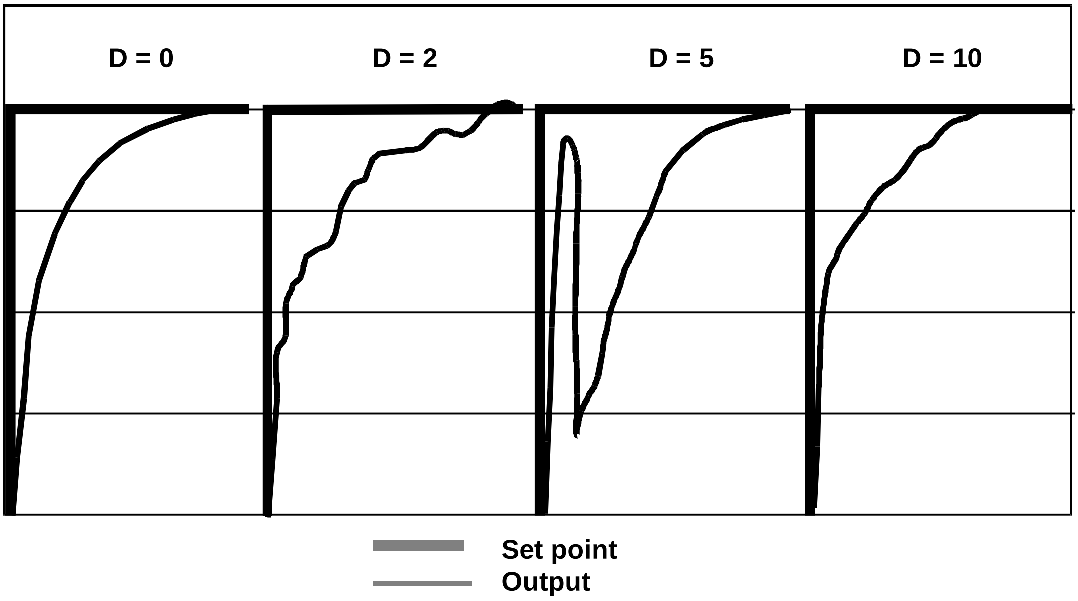
Figure 2

P = 100	P = 100	P = 100
I = 120	I = 12	I = 2
D = 2	D = 2	D = 2

The figures show the behaviour of the same process for two different values of gain, P and varying integration time, I. It can be seen that using high gain will result in overshoot even with longer integration times.
If the integration time, I is too short, an oscillation of the type shown in the right-hand figure will occur.

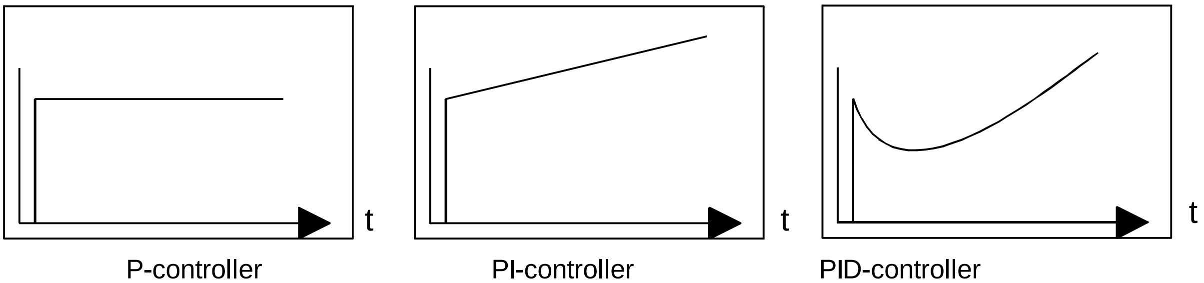
6.1.4 Effect of the differentiation time constant, D

The effect of the differentiation time is critical, as illustrated in the figures below. Derivative control is unsuitable for many types of systems, and is often not necessary at all.



6.1.5 Controller types

The figure below shows the step response of the three basic controller types.



- The P controller has constant gain, and the result is unsatisfactory, especially if the gain
  is low.
- The PI controller changes the apparent gain and correscts the error in time.
- The PID controller exaggerates variations of error in order to abtain rapid error correction. The step responce becomes faster.

6.2 12-Bit Controllers

The AL2000 has 32 x 12-bit controllers with PID characteristics. The shortest possible update interval for one controller is 100ms. The controller parameters are held in W GM memories. The controllers are divided into four groups of eight controllers. Within these groups the user can select different update intervals.

The number of controllers to be used in a particular group is given in the register outputs (R O 128 to R O 131). This number also determines the group update interval depending on the number of controllers used. If one controller in a group is updated, the update time is 100ms. Similarly, if three controllers in a group are updated, the update time is 300ms etc. The group update interval can be reduced by reducing the number of controllers used within that particular group. If the number of controllers is given as 0, no controllers will be used.

The system software runs the group so that there are 20 PID controllers with 500ms update intervals (R O 128 to R O 131 = 5) automatically upon power-up. This configuration can later be changed.

	Controller								Number
Group 1	0	1	2	3	4	5	6	7	R O 128
Group 2	8	9	10	11	12	13	14	15	R O 129
Group 3	16	17	18	19	20	21	22	23	R O 130
Group 4	24	25	26	27	28	29	30	31	R O 131

For Example, if we require three controllers with 100ms update intervals and two controllers with 500ms update intervals:

Group 1 - required interval 100ms = 1 controller (No 0)
Group 2 - required interval 100ms = 1 controller (No 8)
Group 3 - required interval 100ms = 1 controller (No 16)
Group 4 - required interval 500ms = 5 controllers (No 24-28)

Controllers 0, 8 and 16 can be used for the three 100ms update interval controllers and two of the five controllers 24-28 can be used as the 500ms update interval controllers.

	Controller								Number
Group 1	0	1	2	3	4	5	6	7	R O 128
Group 2	8	9	10	11	12	13	14	15	R O 129
Group 3	16	17	18	19	20	21	22	23	R O 130
Group 4	24	25	26	27	28	29	30	31	R O 131

Mode (W GM 0,8,16 etc.) gives the control program the following information:

0 Controller not in use
1 Controller in automatic mode
2 Controller in automatic inverted mode
3 Controller in manual mode

6.3 Three Point Controllers

There are 32 three point controllers available

Controllers	0-7	8-15	16-23	24-31
Control interval	R O 128	R O 129	R O 130	R O 131
Pulse interval	R O 132	R O 133	R O 134	R O 135
Valve closing bit	R O 136	R O 137	R O 138	R O 139
Valve opening bit	R O 140	R O 141	R O 142	R O 143

Controller parameters are stored in W GM 0-255.

	Controller 0	Controller 1	Controller 2	-	Controller 31
Mode	0	8	16	-	248
Actual value	1	9	17	-	249
Set point	2	10	18	-	250
D time/100ms	3	11	19	-	251
I time/100ms	4	12	20	-	252
Gain term	5	13	21	-	253
Output	6	14	22	-	254
Aux.	7	15	23	-	255

6.3.1 Control Algorithm

The control algorithm is as follows:

$DY=(100/gain)*(e(t_l)-(e(t_l-1))$ + ; P term
$e(t_l)$/integration time constant+ ; I term
diff. time constant$*(e(t_l)-2e(t_l-1)+e(t_l-2))$; D term

The D term controls how strongly the control process reacts to rapid changes in the input signal. The differentiation time constant is an 8-bit parameter and is given in hundreds of milliseconds (0.1-25.5s). The larger the time the greater the changes to the output signal. If the differentiation time constant is given as 0, the D term is not used.

The I term controls how quickly the process reacts to stabilise offset error. The integration time constant is a 16-bit parameter and is given in hundreds of milliseconds (0.1-6553.5s). The larger the integration time constant the slower the process reacts to offset errors. If the integration time constant is given as 0, the I term is not used.

The P term controls the gain such that: P=100/gain, where gain can be from 0.01-100

Therefore:	if gain = 0	P-TERM NOT USED
	if gain = 5	P = 20
	if gain = 0.5	P = 200
	if gain = 0.02	P = 5000
	if gain = 0.01	P = 10000

Example 1

If P term = 100
D term = 0
I term = 0
Then a 1% change in input signal would produce a 1% change in output signal

Example 2

If P term = 50
D term = 0
I term = 0
Then a 1% change in input signal would produce a 2% change in output signal

The P term can also be used where the input and output signals are not in the same range (e.g. If the input signal range is 0-1000 and the output signal range is 0-100 we need a value P=1000. This would result in a gain of 1).
All controllers provide three-point control outputs in the form of bit data in register outputs R O 136-143.

Controller No.:		0	1	2	3	4	5	6	7
Close output:	R O 136 bit	0	1	2	3	4	5	6	7
Open output:	R O 140 bit	0	1	2	3	4	5	6	7
Controller No.:		8	9	10	11	12	13	14	15
Close output:	R O 137 bit	0	1	2	3	4	5	6	7
Open output:	R O 141 bit	0	1	2	3	4	5	6	7
Controller No.:		16	17	18	19	20	21	22	23
Close output:	R O 138 bit	0	1	2	3	4	5	6	7
Open output:	R O 142 bit	0	1	2	3	4	5	6	7
Controller No.:		24	25	26	27	28	29	30	31
Close output:	R O 139 bit	0	1	2	3	4	5	6	7
Open output:	R O 143 bit	0	1	2	3	4	5	6	7

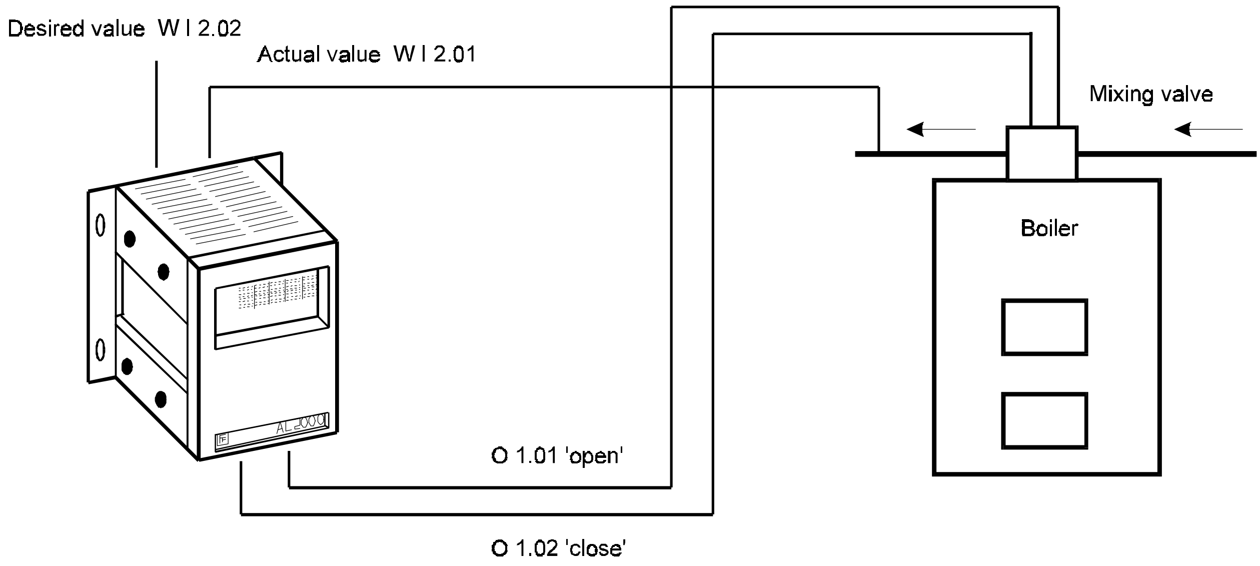
The pulse interval, i.e. the dead time that the controller will wait for the process response to settle, is programmed into register outputs R O 132-135. The time range is from 0.1s to 25.5s, corresponding to the numbers 1-255.

R O 132	Controllers 0-7
R O 133	Controllers 8-15
R O 134	Controllers 16-23
R O 135	Controllers 24-31

A typical application, controlling the temperature of the circulating water in a central heating system, is shown over the page.

The mixing valve is controlled by a pulse driven actuator motor. The dead time in an application of this kind can be several seconds, as changes in the water temperature will not immediately be detected by the sensor. It is, therefore, futile to give further drive pulses to the valve before the true effect of the previous action is known. This dead time can be accounted for when tuning the controller using the step response method.

Example: Temperature control of circulating water by 3-point control.



Logic program for water heating:

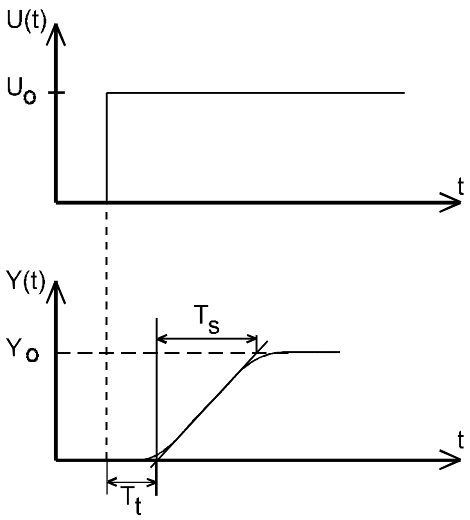
```
STR   W  C    050   ;Integration time (50 * 0.1s) ( 0 - 65535)
EQ    W  GM   012   ; greater value gives slower action
STR   W  C    002   ;Differentiation time (2 * 0.1s) ( 0 - 255)
EQ    W  GM   011   ; greater value gives bigger steps
STR   W  C    110   ;Gain P = (100/110)
EQ    W  GM   013   ; gretaer value gives slower action
STR   W  I    2.2   ;Set point, e.g. from potentiometer
EQ    W  GM   010
STR   W  I    2.1   ;Actual value from temperature sensor
EQ    W  GM   009
STR   W  C    001   ;Controller in automatic mode
EQ    W  GM   008
STR   R  C    010   ;Pulse interval 1s
EQ    R  O    132
STR   R  O    140   ;Read 'open' bits
BIT      M    020   ;Convert to bits (controllers 0 - 7)
STR      M    021   ;Use bit for controller 1
EQ       O    1.2   ;to open valve
STR   R  O    136   ;Read 'close' bits
BIT      M    020   ;Convert to bits
STR      M    021   ;Use bit for controller 1
EQ       O    1.3   ;to close valve
STOP
```

6.4 Controller Tuning

In a control system, suitable values must be found for the control parameters P, I and D. Suitable parameters can be determined using mathematics or a Bode diagram, however, these methods can be time consuming. Controllers are usually tuned on the basis of experimental data from the control system. Two simple and effective methods for this are described below.

6.4.1 Step Response Method

The delay and rise time of the step response of the process are determined, and the controller settings are determined on that basis.



P controller: $UP = \frac{TtUO}{TSYO} = c$

PI controller: UP = 1.25c
TI= 3Tt

PID controller: Up = 0.85c
TI = 2Tt
TD = 0.42Tt

6.4.2 Oscillation Method

The gain and oscillation cycle time at the point of oscillation are determined, and the controller settings are determined on that basis.

P controller: KP = 0.5KPcr

PI controller: KP = 0.455KPcr
TI = 0.85Tcr

PID controller: KP = 0.6KPcr
TI = 0.5Tcr
TD = 0.12Tcr

KP = 1 / UP = gain

Where:

TI	Integration time constant
TD	Differentiation time constant KPcr
	Critical gain at which the process oscillates
Tcr	Cycle time of process oscillation

The above two methods yield reasonable starting points for the controller parameters which can be further refined during operation. The ALPro software can be used for monitoring controller variables, building trend data and producing documentation.

7. DISPLAY/KEYPAD UNITS

There are several display/keypad units available for the AL2000, most of which can be connected to the CPU board via I^2C interface..

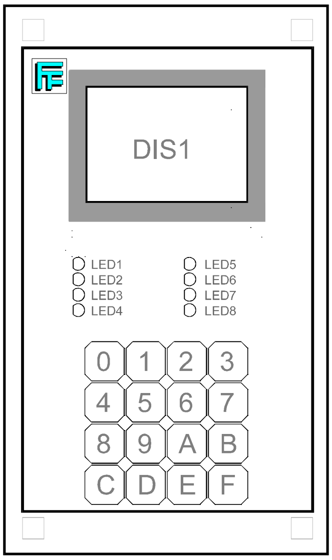
7.1 AL1096 Display/Keypad Units

AL 1096PS/PE
- front panel mounted enclosure, RS232 connection
- 320x240 pixel STN graphic LCD display, backlit
- touch screen max. 40 x 30 touch keys
- Clock and calendar, battery backup

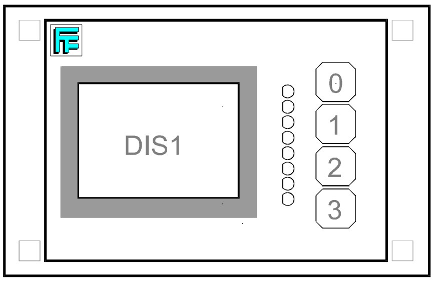
AL 1096S
- front panel mounted enclosure, RS232 connection
- 5 function keys
- 240 x 128 pixel graphic LCD display, backlit

AL1096T
- front panel mounted enclosure, RS232 connection
- touch screen max. 10 x 8 touch keys
- 240 x 128 pixel graphic LCD display, backlit

7.2 AL1095A/B Display/Keypad Units



AL1095A



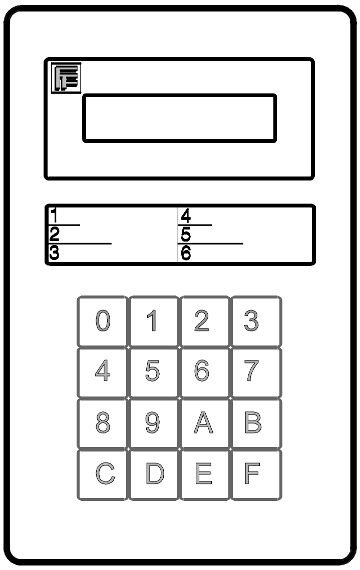
AL1095B

- Connection to PLC's I^2C bus by system cable, length 1 m.
- 8x21 character alphanumeric LCD display/
  128x64 pixel graphic display, backlit
- 8 LED indicator lights
- 16 keys 0 ... F (model A), 4 keys 0 ... 3 (model B)
- skandinavian/cyrillic characters, selectable by jumper
- supply voltage through system cable from PLC's supply unit
- clock/calendar, battery backup
- the unit can be fitted with a clien designed face plate
- encapsulated, front panel mount, front panel seal IP54

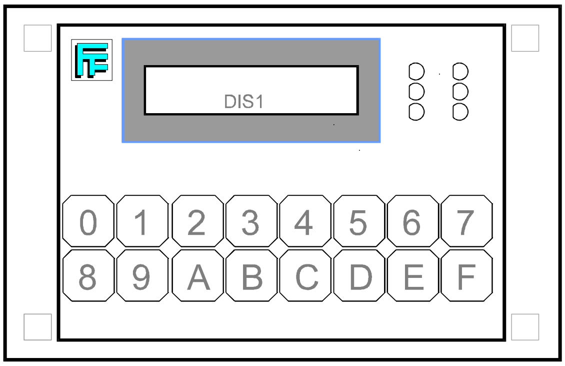
7.3 AL1093 Display/keypad Units

There are three versions of the AL1093 unit. These are AL1093C, AL1093D and AL1093F. All AL1093 units have 2 x 16 character alphanumeric display, 16-key hexa-decimal keypad and 6 LED indicator lights. AL1093 unit can be connected to any Autolog PLC. In addition to displaying the values and time/date information, the AL1093 unit can also display text (using the PRT instruction). the LCD display is backlighted. The clock and calendar functions are battery backed-up.
All AL1093 models are front panel moun type and models D and F can be fitted with client designed face plate.

AL1093C / D

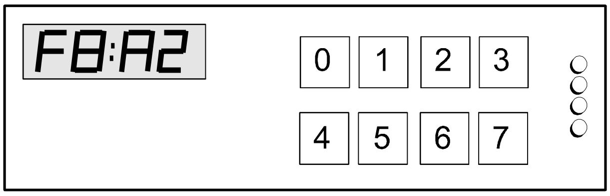


AL1093F

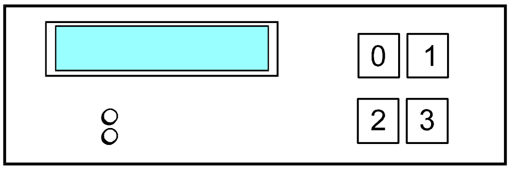


7.4 AL1094 / R / AF Display/keypad Units

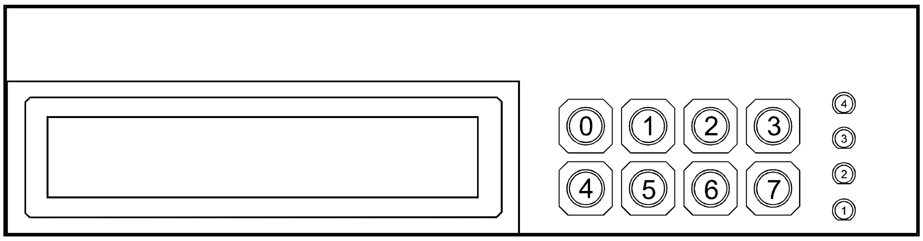
The AL1094 display/keypad unit without clock and calendar functions. It has a 4-digit 7-segment display, four LED indicators and a 4-key keypad. The unit can be fitted with client designed face plate.



The AL1094R display/keypad unit without clock and calendar functions. It has 2 x 16 character alphanumeric display, two LED indicators and a 4-key keypad. In addition to displaying the values and time/date information, the AL1094R unit can also display text (using the PRT instruction). The unit can be fitted with client designed face plate.



The AL1094AF display/keypad unit has 2x20 character alphanumeric display, clock/calendar, 8 keys, 4 indicating leds, summer and one 24VDC 300 mA output for external indicating ligth. The AL1904AF can display text (using the PRT instruction). The unit can be fitted with client designed front plate. Output O 4 = buzzer, output O 5 = control output for external indicating ligth .



AL1094AF

7.5 Reading The Keypad

R O 209: The ASCII code of the last character from keypad is stored in the register output R O 209, when the keypad is connected to PLC's I^2C connector .

Example: PLC receives a character from keypad.

```
STR   R    O    209    ; Read the character from keypad
LES   R    C    000    ; If value <> 0
EQ    R    RO   209    ; reset the buffer
PRT   R    T           ; and output the character
STOP
```

The keypad status is stored in the bits of R O 207 and R O 208. When no key is being pressed, all bits are set to '0'. When a key is pressed, the bit corresponding to that key is set to '1'. The keys and their corresponding bit are shown below.

R O 207	bit	7	6	5	4	3	2	1	0
	key	7	6	5	4	3	2	1	0
R O 208	bit	7	6	5	4	3	2	1	0
	key	F	E	D	C	B	A	9	8

7.6 Control of LED Indicators

The LED indicators / other outputs from display are controlled by the bits of R O 204. An LED / output can be set on by setting the approriate bit to '1', and subsequently turned off by setting the bit to '0'. The LEDs and their corresponding bits are shown below.

R O 204	bit	0	1	2	3	4	5
	LED	L0	L1	L2	L3	L4	L5

7.7 Display Control Characters

Mnemonic	Hex	Function
BS (Back space)	08	Moves the cursor one character to the left and clears that position. If the cursor is at the beginning of a line, it moves to the end of the previous line.
HT (Tabulator)	09	Moves the cursor one character to the right without clearing that position. If the cursor is at the end of a line, it goes to the beginning of the next line.
LF (Line Feed)	0A	Moves the cursor to the next line.
FF (Form Feed)	0C	Clears the display and moves the cursor to the beginning of the first line.
CR (Return/Enter)	0D	Moves the cursor to the beginning of the present line.
NAK	15	Moves the cursor one character to the left without clearing that position.If the cursor is at the beginning of a line, it goes to the end of the previous line.
<ESC>,"Y", <line>,<col>	1B 59 01 08	Moves the cursor to line 1, column 8. Line 1...4, column 1...40, maximum numbers along displays.

7.8 Display Modes

All display/keypad units have several display modes. Some modes are not available for every unit (e.g. those without clock and calendar functions and reduced keypads). The display modes are outlined below and defined by register output R O 240.

R O 240:

Content	Function
0	Mode "0" (normal Mode)
2	Mode "1"
4	Mode "2"
8	Mode "3"
16	Mode "4" the entire display is reserved for PRT outputs
32	Mode "5" as mode "0", but the B-key disabled (changes)
128	Mode "6" display locked (doesn’t allow any changes in display)

7.8.1 Normal display mode - Mode '0' (R O 240 = 0)

Key	**Function**
0-9	Numeric keys
A	Set clock/calendar
B	Enter parameters into register variables
C	Display register variables
D	Next display/entry
E	Display address (in register variable display mode)
F	Return to clock display

'A' Set Clock/Calendar

The clock/calendar is set as follows:

Key	display		continue
'A'	1-xx	year (two digits)	D or F
	2-xx	month (two digits, 01-12)	D or F
	3-xx	day (two digits, 01-31)	D or F
	4-xx	hour (two digits, 01-24)	D or F
	5-xx	minute (two digits, 01-59)	D or F
	6-x	day of week (one digit, 01-07)	D or F
		(1 = mon, 2 = tue ,..., 7 = sun)	

'B' Enter Parameters into Register variable

Key	display		continue
'B'	RM000	new address (octal number)	D or B or F
	WM000	continue with 'B' browsing variables	D or B or F
	RO000	address of variable continue with 'D'	D or F
	xxx	enter parameters value	D or F
	XX001	address 001 or new address	D or F
	xxx	enter parameters value	D or F
		and/or quit by ‘F’-key	

'C' Display register memory/output or word memory

The value of register variable is updated continuously into the display. Update rate is one second.

Key	display		continue
'C'	RM000	new address (octal number)	C or D or F
	WM000	continue with 'C' browsing variables	C or D or F
	RL000	address of variable 'D' -> update	D or E or F
	yyy	‘D’ -> next variable	D or F
	RL000	‘E’ display varaible’s address	C or D or F
		escape by ‘F’-key	

'D' Next Display Mode

In the normal display mode ‘0’, the display information may changed by pressing the ‘D’-key. This is performed as follows:

Key	display		continue
‘D’	15:30	hours:minutes	D or F
	17.05	date.month	D or F
	00:01	date of week	D or F
		(models AL1093, AL1094AF, AL1095)	
	FF:FF	R O 245 and R O 246 in hex	D or F
	15:30	hours:minutes.... etc.	

7.8.2 Data Display Mode '1' [R O 240 = 2]

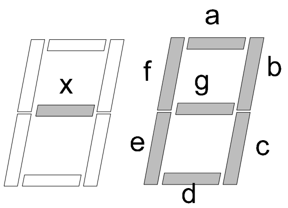
This mode is available for all models. In this mode the contents of R O 245 and R O 246 are sent to the display. In this mode it is not neccessary use the ‘D’ key to change the display mode (as with mode ‘0’) each time after power-up.

Example: I000 connects the display control program and the contents of R O 245 and R O 246 are immediately displayed. When I001 is truned off, it disconnects the control program, and the display will again show the time.

```
STR   R   C    000      ; Normal display mode
EQ    R   O    240      ; chosen for display
STR       I    000      ; set input on (=ON)
STR   R   C    002      ; the control code for mode '1' is
EQ    R   SO   240      ; transferred to register output R O 240.
STOP                    ; When writing the value to R O 240 ends,
                        ; returns the to normal mode (two first instruction lines)
```

7.8.3 7-Segment mode - mode '2' (R O 240 = 4)

This display mode is available only for the AL1094 with 7-segment display. In this mode R O 245 controls the right hand side of the 7-segment display, so that seven of the eight bits controls a segment each, and the eighth bit controls the minus sign as shown in the diagram below.



R O 245:	bit	7	6	5	4	3	2	1	0
	segment	x	g	f	e	d	c	b	a

Segments are illuminated by writing the sum of the corresponding values into R O 245

Segment	Number	Segment	Number
a	1	e	16
b	2	f	32
c	4	g	64
d	8	x	128

For example, to display segments e, c and d, the sum of corresponding numbers (16+8+4=28) should be written into register output R O 245.

```
STR   R   C    028
EQ    R   O    245      ; Display segments e, c and d
```

7.8.4 Text Display Mode ‘4’ (R O 240 = 16)

This mode is available only for the AL1094R, AL1093, AL1094AF and AL1095 with alphanumeric displays. The entire display will be reserved for text produced by the PRT instruction.

7.8.5 Normal Mode with Key 'B' Function Disabled. Mode ‘5’ (R O 240 = 32)

This mode is available only for the AL1094R, AL1093C/D/F and AL1095A models with alphanumeric displays. The unit functions otherwise normally but the changes of content of variables are disabled.

7.8.6 Display Lock Mode ‘6’ (R O 240 = 128)

This mode is available only for the AL1094F/R, AL1093C/D/F and AL 1095A/B with alphanumeric displays. Current display information will be retained and no changes will be allowed.

7.9 Program examples

7.9.1 Printing Date Information

```
STR        I     0.01
EQ         M      000
STR        DP     000        ; print
PRT   R    O      248        ; date
PRT        T       .@        ; point
PRT   R    O      247        ; month
PRT        T     .19@        ; point and hundreds of year
PRT   R    O      255        ; year ( 0 - 99)
```

7.9.2 Real Time Control with R O 253 (6 minutes from midnight)

```
STR   R    O      253        ; read number of six minute periods from beginning of day
LES   R    C      174        ; every day starting from 17:30
EQ         M      000        ; (17.5 * 60/6 = 175)
GRT   R    C      210        ; until 21:00
AND        M      000
EQ         O     0.01        ; output on
STOP
```

7.9.3 Display Step Registers 0 and 1 (0 ... 99)

```
STR   R    C      002        ; set register outputs R O 245 and 246
EQ    R    O      240        ; display mode '1' value
STR   R    S      000        ; read step register’s 0 step into register accumulator
BCD                          ; convert to BCD form
EQ    R    O      245        ; save the value into display variable
STR   R    S      001        ; read step register’s 1 step into register accumulator
BCD                          ; convert to BCD form
EQ    R    O      246        ; save the value into display variable
STOP
```

7.9.4 Display Control Characters

```
STR   W    I   0.01            ; read word input 001 into word accumulator
BDC   W    T                   ; convert to BCD
EQ    W    M   000             ; save into word memory
STR        P   001             ; print on 1 second period
PRT   (<1Bh>,"Y",<01>,<04>)    ; to line 1 column 4
PRT   R    M   000             ; the value of W T 1, first high byte
PRT   R    M   001             ; and then low byte
STOP
```

7.9.5 Setting the Time and Date Using Register variables

```
STR   R    S    000     ; Step register 0 in use
STR        I    0.00    ; Clock synchronising input
EQ         M    065
STR        DP   065
AND        S    000
STEP       S    001     ; activate clock time setting
STR   R    S    000     ; step registers step to register accumulator
LES   R    C    000     ; program is executed only if current
IF         T            ; step is greater than 0
STR   R    S    000
LES   R    C    019     ; 2 seconds delay, made by STEP register 0
STEP       S    000
STR   R    C    005
EQ    R    SO   242     ; transfer time to real time clock
STR   S         001     ; if in step
STR   R    C    001
EQ    R    SO   242     ; disable reading of real time clock
STR   R    C    001     ; check if reading of real time clock
EQU   R    O    242     ; is disabled
AND        P    000     ; pulse, interval 0.1 second
STEP       T            ; go to next step
STR   R    C     96     ; set year
BCD   R    T
EQ    R    O    255
STR   R    C      3     ; set month
BCD   R    T
EQ    R    O    247
STR   R    C     10     ; set day
BCD   R    T            ; of month
EQ    R    O    248
STR   R    C      9     ; set hour
BCD   R    T            ; part of time
EQ    R    O    250
STR   R    C     25     ; set minute
BCD   R    T            ; part of time
EQ    R    O    251
CONT
STOP
```

Sequencies in setting clock time:

1. Stop system program RTC clock read function by inserting to R O 242 value 1.
2. Set new values to clock variables from R O 247 to R O 255 in BCD format.
3. Start function "insert new values" by inserting to R O 242 value 5. Now system program updates the RTC and resets R O 242 to zero which starts normal clock reading to R O variables.

8. TECHNICAL DESCRIPTIONS

8.1 CPU2000S Series Modules

Memories	FLASH - Operating system FLASH - PLC-program & configuration RAM (battery backed-up) - PLC-program variables RAM - buffers, operating system variables, PLC-program when executed
PLC-program space	8192 instruction lines
Commands	Approximately 260
Auxiliary memories	50 kBytes of memory reserved for auxiliary memories and variables
Timers	4 x 0.01-2.55s, addressable from PLC program 68 x 0.1-25.5s, addressable from PLC program 8 x 1-255s, addressable from PLC program
Counters	16 x 0-255, addressable from PLC program
Step registers	32 x 0-255 step. The first 8 are reset during power failure. All others can maintain their position during power failure by setting a DIP switch.
Controllers	32 x 16-bit PID controller functions with programmable parameters 8 x 8-bit PID controller functions with programmable parameters
Loop time	5ms+20µs/instruction line (average)
Serial Interfaces	3 isolated (500 VDC) SER1 RS-232C (300-115200 bit/s, 9-pin D-type connector). Used for programming, operator's terminal, bar-code reader, MODBUS slave/master, communication with supervising PC, modem handshakes. SER2 RS-232C (300-115200 bit/s, 9-pin D-type connector). Used for operator's terminal, bar-code reader, MODBUS slave/master, communication with supervising PC, modem handshakes. SER3 RS-485 (300-115200 bit/s, screw terminals). Used for operator's terminal, bar-code reader, MODBUS slave/master, communication with supervising PC
I^2C interface	For display/keyboard unit
Isolation	Serial interfaces SER1 and SER2 are isolated as a group. Serial interface SER3 is isolated separately Isolation is between groups, 500VDC, metal frame and the internal data bus.

LED indicators	RES (fault, red) RUN (PLC functioning, green) TX1, TX2, TX3 (serial interfaces transmitting, yellow) RX1, RX2, RX3 (serial interfaces receiving, yellow) CTS1, CTS2 (serial interfaces CTS handshaking, red) RTS1, RTS2 (serial interfaces RTS handshaking, red) BAT (battery low, red) VS3 (fault in SER3 interface, red)
Watchdog	Separate watchdog circuit with excess voltage protection and low voltage detection
Clock	Year, month, week, day, hour, minute, second and number of six-minute periods from the beginning of the day. Battery backed-up
Power consumption	600mA, 5V for CPU2000S 0.2-1.2A, 24V for CPU2000P (depending on number of I/O modules)
Battery life	With no external supply, 2 years with max. ambient temperature +60°C Normal use, up to 10 years.
Cooling	Natural
Dimensions	6HE, 4TE, = 233.3 mm x 160 mm x 25 mm
Weight	0.35 kg
Product Codes	CPU2000S 902200 CPU2000P 902205

8.2 CPU2000L Module

CPU	486DX4/DX5, 5x86 AMD
Ethernet	10 Base2/5 AUI / 10 Base T RJ-45
or	
CPU	Pentium 233 MHz AMD
Ethernet	10/100 Base2/5 AUI / 10/100 Base T RJ-45
Flash memory	16 - 48 MB FLASH
RAM memory	16 - 32 MB
Bus	Optional PC-104 extensions (HDD, FDD, etc.)
Operating system	QNX
Software	FCS control system software
Serial Interfaces	1 isolated SER1 RS-232C (300 - 115200 bits/s, 9-pin D-type connector). SER2 RS-422/485/232 (300 - 115200 bits/s).
LED Indicators	11 LED indicators
Power Consumption	4 A, 5VDC
Cooling	Natural
Operating temperature	0 - 50 °C. Available also for higher ambient temperatures

For testing PC in installation phase following devices can be connected to CPU2000L

Option Display driver	PC-104 VGA/SVGA, 256 colour/grey scale, maximum resolution 1024 x 768
Option Display unit	VGA/SVGA video monitor with 15-pin D-type connector
Option Keyboard	Connector for Mouse & Standard IBM-AT keyboard

8.3 AL2000S/P Variables

8.3.1. Single Bit Variables

I	Input (0-255).
NI	One's complement of input. When In=1, NIn=0.
M	Auxiliary memory (0-255).
NM	One's complement of auxiliary memory (M).
GM	Additional auxiliary memory (0-255).
NG	One's complement of additional auxiliary memory (GM).
BM	Additional auxiliary memory (0-255).
NB	One's complement of additional auxiliary memory (BM).
O	Output (0-255).
NO	One's complement of output.
SM, SG, SB, SO	Conditional setting of memory (M/GM/BM)/output (0-255). Used with EQ instruction.
RM, RG, RB, RO	Conditional resetting of memory (M/GM/BM)/output (0-255). Used with EQ instruction.
DP	Change auxiliary memory from 0 to 1, numbered from 0-127. Compares memory state to that at beginning of cycle.
DN	Change auxiliary memory from 1 to 0, numbered from 0-127. Compares memory state to that at beginning of cycle.
P	Pulse variable. The variable is 1 for the duration of one program cycle, P000 ten times per second, P001 once per second, and P002 once per minute.
S	Sequence register or a step of a sequence register. There are 64 sequence registers (number configurable from 8-64, default 32). These registers have 256 steps each.
NS	One's complement of sequence register (S).
T	Timer (0-79). Timer resolutions are as follows:

Timer	Resolution	Time Span
T0-3	10 ms	0.02-2.55s
T4-7	100 ms	0.2-25.5s
T8-15	1 s	2-255s
T16-79	100ms	0.2-25.5s

In the IF and STEP instructions T refers to the bit accumulator.
In the PRT instruction T refers to TEXT.

C	Counter (0-15), count down from 255 to 0. In the PRT instruction C refers to a numerical value.

8.3.2. Register (8-bit) Variables

R M	Register memory (0-255). These are 8-bit registers.
R NM	One's complement of register memory.
R GM	Register general memory (0-255). These are 8-bit registers, some of which are reserved for 8-bit PID controllers.
R NG	One's complement of register general memory.
R O	Register output (0-255), all of which are reserved for special purposes. Register outputs defines the operational parameters of CPU.
R NO	One's complement of register output.
R SM, R SG, R SO	Conditional setting of register memory/general memory/output. Used with EQ instruction.
R RM, R RG, R RO	Conditional resetting of register memory/general memory/output. Used with EQ instruction.
R T	Register timer. This is identical to the timers listed above; T or R T is used as the variable according to the instruction. Also refers to the register accumulator in some instructions.
R C	Register constant (0-255), except register counter with the READ and LOAD instructions. C or R C is used as the variable according to the instruction.
Q	Queue (0-3). Each queue has 256 x 8-bit locations.
F	First In First Out store (0-7). Each FIFO has 256 x 8-bit locations.

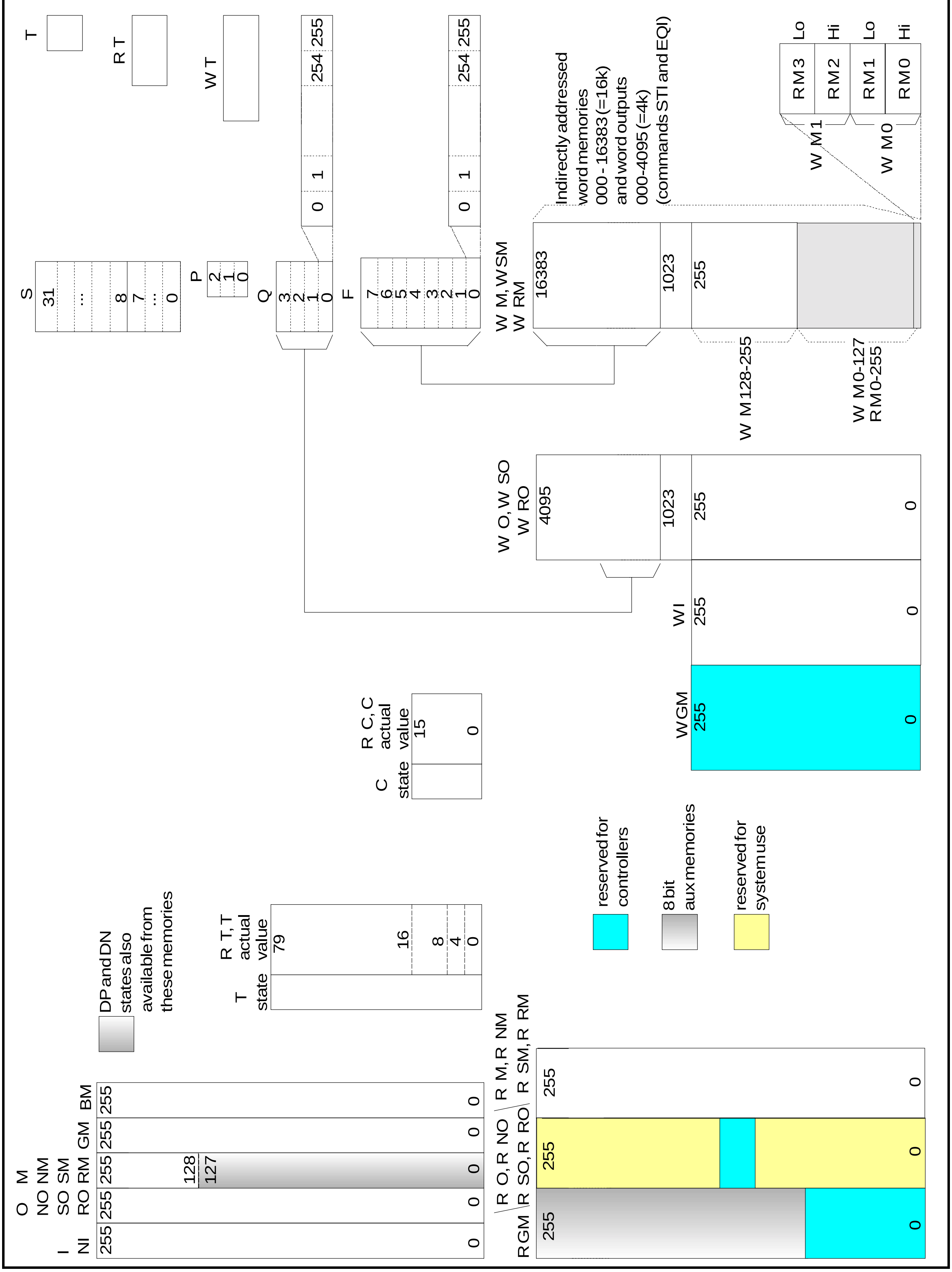
8.3.3. Word Variables or 16 bit variables

W I	Word input (0-255).
W M	Word memory (0-255, indirectly addressable up to 16363). These are 16-bit memories . Word memories 0-127 overlap register memories 0-255.
W GM	Additional word memory (0-255). These are 16-bit memories, some of which are reserved for 16-bit PID controllers.
W O	Word output (0-255, indirectly addressable up to 4096). These can be used as auxiliary memories.
W SM, W SO	Conditional setting of word memory/output.
W RM, W RO	Conditional resetting of word memory/output.
W T	Word accumulator.
W C	Word constant (0-65535,bin / 0 - 9999 bcd).

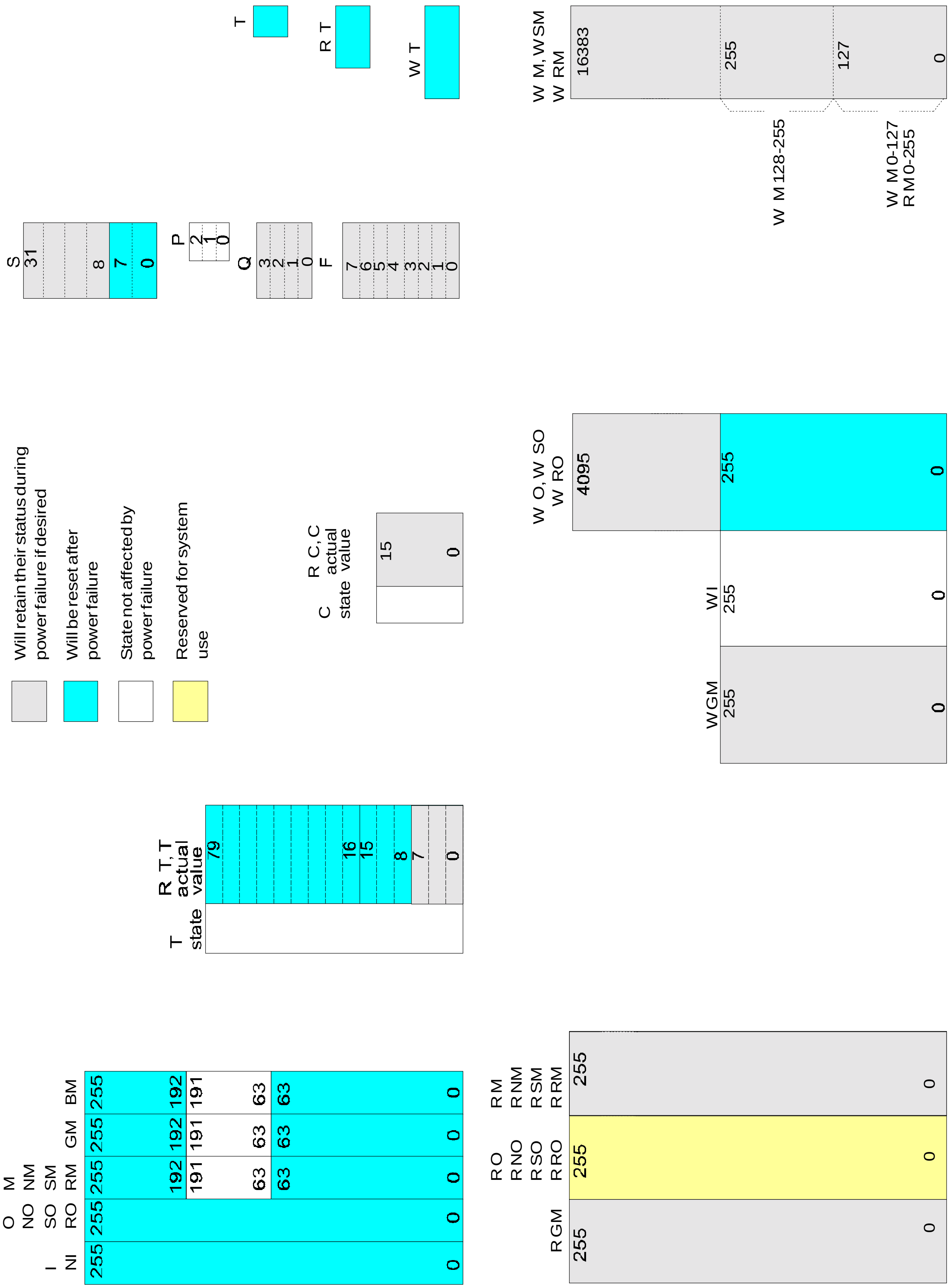
8.3.4. Special Variables

TX	String constant (0-255). These are text strings with a maximum length of 16 characters each. Used in conjunction with the PRT instruction.

Memory Map of AL2000S CPU



The behaviour of the AL2000S Memories on power failure



9. REGISTER MEMORIES, OUTPUTS & VARIABLES

9.1 Reserved Memories and Outputs

R O 000-031	Status information for I/O modules in slots 0-31
R O 032	Watchdog status information
R O 033	Keypad/Display unit type
R O 034	System program version
R O 035	DIP switch (SW1) status
R O 036	Battery and SER3 fault status
R O 037	File transfer error counter
R O 038	Processor Type (0=80C32, 1=80C320)
R O 039	CPU type (0=S, 1=SCP)
R O 040	Error code on program save
R O 043	FPGA IC1 version
R O 044	FPGA IC2 version
R O 045	Flash manufacturer code
R O 046	Flash device code
R O 048	Number of non answered Modbus slave address for SER1
R O 049	Type number of detected Modbus error for SER1
R O 050	MODBUS error information for SER1
R O 051	MODBUS counter : abandoned messages on SER1
R O 052	MODBUS counter : accepted messages on SER1
R O 054	MODBUS message transmission delay for SER1
R O 055	MODBUS response timeout for SER1
R O 056	MODBUS response timeout for SER2
R O 057	CTS / RTS control memory forSER1 (CTS=bit 7, RTS=bit 1)
R O 058	CTS / RTS control memory forSER2 (CTS=bit 7, RTS=bit 1)
R O 059	MODBUS error information for SER2
R O 060	MODBUS response timeout for SER3
R O 063	MODBUS error information for SER3
R O 080	MODBUS counter : abandoned messages on SER2
R O 081	MODBUS counter : accepted messages on SER2
R O 082	MODBUS message list completed for SER2
R O 083	MODBUS message transmission delay for SER2
R O 084	MODBUS counter : abandoned messages on SER3
R O 085	MODBUS counter : accepted messages on SER3
R O 086	MODBUS message list completed for SER3
R O 087	MODBUS message transmission delay for SER3
R O 088	Number of non answered Modbus slave address for SER2
R O 089	Type number of detected Modbus error for SER2
R O 090	Number of non answered Modbus slave address for SER3
R O 091	Type number of detected Modbus error for SER3

R O 128-131	Update interval for controller groups 1-4 (default 5=500ms)
R O 132-135	Pulse interval for 12-bit PID-controller groups 1-4
R O 136-139	Closing bits for 12-bit PID-controller groups 1-4
R O 140-143	Opening bits for 12-bit PID-controller groups 1-4
R O 160 - 183	Reserved for Counters on old AL2000SAC/SCP boards
R O 192	Closing control bits for 8-bit PID-controllers 0-7
R O 196	Opening control bits for 8-bit PID-controllers 0-7
R O 200	Pulse interval for 8-bit PID-controllers 0-7
R O 204	Control for LEDs on display unit
R O 207	Bit information from keys 0-7
R O 208	Bit information from keys 8-F
R O 209	Last character received from the keyboard
R O 210	SER1 data length and parity in terminal mode
R O 211	Reserved for test information
R O 212	Selected language
R O 213	SER1 communication speed
R O 214	SER1 mode; programming/ terminal /MODBUS SLAVE/MASTER
R O 215	SER2 mode; terminal/EVS/MODBUS SLAVE/MODBUS MASTER
R O 216	SER3 mode; terminal/EVS/MODBUS SLAVE/MODBUS MASTER
R O 217	SER3 communication speed
R O 218	SER3 data length and parity in terminal mode
R O 219	SER2 data length and parity in terminal mode
R O 220	Calibration of analog input
R O 221	Slot number of module to be calibrated
R O 222	Input number on module to be calibrated
R O 223	Input type to be calibrated
R O 224	Lower calibration value (4 MSB)
R O 225	Lower calibration value (8 LSB)
R O 226	Higher calibration value (4 MSB)
R O 227	Higher calibration value (8 LSB)
R O 228	Number of step registers (configurable from 8 to 64, default=32)
R O 229	SER2 communication speed
R O 230	Word variable: multiplication/division (MSB)
R O 231	Word variable: multiplication/division (LSB)
R O 232	SER1 last character received from terminal
R O 233	SER2 last character received from terminal
R O 234	SER3 last character received from terminal
R O 235	CPU counter input status (AL2000SCP)
R O 236	Analog input update rate (1/2/4/8 inputs per program cycle)
R O 237	Value of CPU analog output 0 (AL2000SAA/SAC)
R O 238	Value of CPU analog output 1 (AL2000SAA/SAC)
R O 239	Value of CPU analog output 2 (AL2000SAA/SAC)

R O 240	Selection the display information
R O 241	Number of power fail interruptions
R O 242	Setting the time
R O 243	Slave address of AL2000
R O 244	High byte of multiplication/division remainder
R O 245	Left side of numerical display (for AL1092)
R O 246	Right side of numerical display (for AL2092)
R O 247	Date and time information: month
R O 248	Date and time information: date
R O 249	Date and time information: day of the week
R O 250	Date and time information: hour
R O 251	Date and time information: minute
R O 252	Date and time information: second
R O 253	Clock & Calendar: number of six minutes from the beginning of the day
R O 255	Date and time information: year
R GM 0-7	Controller 0
R GM 8-15	Controller 1
R GM 16-23	Controller 2
R GM 23-31	Controller 3
R GM 32-39	Controller 4
R GM 40-47	Controller 5
R GM 48-55	Controller 6
R GM 56-63	Controller 7
R GM 64-80	Counter 1 control registers (SAC)
R GM 96-112	Counter 2 control registers (SAC)
R GM 128-191	AL1095 Graphical objects (see 9.3)

9.2 Modbus Error information Register Outputs

	SER1	SER2	SER3
Slave ID number	R O 48	R O 88	R O 90
Type number of detected error	R O 49	R O 89	R O 91

Error codes:

First cycle	second cycle	
17d	33d	warning, error in building master request
18d	34d	warning, error on sending master request
19d	35d	warning, no answer
20d	36d	warning, receive timeout (command)
21d	37d	warning, receive timeout (data)
22d	38d	warning, CRC error
23d	39d	warning, received address/command different than request address/command

9.3 Control variables for AL1095 graphical symbols

R GM 128	200	Graphic object 1:mode	R GM 160	240	Graphic object 5:mode
R GM 129	201	Value	R GM 161	241	Value
R GM 130	202	X start point	R GM 162	242	X start point
R GM 131	203	X length	R GM 163	243	X length
R GM 132	204	Y start point	R GM 164	244	Y start point
R GM 133	205	Y length	R GM 165	245	Y length
R GM 134	206	uppdate interval	R GM 166	246	uppdate interval
R GM 135	207		R GM 167	247	
R GM 136	210	Graphic object 2: mode	R GM 168	250	Graphic object 6: mode
R GM 137	211	Value	R GM 169	251	Value
R GM 138	212	X start point	R GM 170	252	X start point
R GM 139	213	X length	R GM 171	253	X length
R GM 140	214	Y start point	R GM 172	254	Y start point
R GM 141	215	Y length	R GM 173	255	Y length
R GM 142	216	uppdate interval	R GM 174	256	uppdate interval
R GM 143	217	Graphic object 3: mode	R GM 175	257	Graphic object 7: mode
R GM 144	220	Graphic object 3: mode	R GM 176	260	Graphic object 7: mode
R GM 145	221	Value	R GM 177	261	Value
R GM 146	222	X start point	R GM 178	262	X start point
R GM 147	223	X length	R GM 179	263	X length
R GM 148	224	Y start point	R GM 180	264	Y start point
R GM 149	225	Y length	R GM 181	265	Y length
R GM 150	226	uppdate interval	R GM 182	266	uppdate interval
R GM 151	227	X start point	R GM 183	267	X start point
R GM 152	230	Graphic object 3: mode	R GM 184	270	Graphic object 8: mode
R GM 153	231	Value	R GM 185	271	Value
R GM 154	232	X start point	R GM 186	272	X start point
R GM 155	233	X length	R GM 187	273	X length
R GM 156	234	Y start point	R GM 188	274	Y start point
R GM 157	235	Y length	R GM 189	275	Y length
R GM 158	236	uppdate interval	R GM 190	276	uppdate interval
R GM 159	237		R GM 191	277	

Graphic object update interval: 1 = 1 sec., 2 = 2 sec., ..., 0 = 256 sec.

9.4 Register Outputs (which must be set)

R O 212 Selected language

0	English
1	Finnish

R O 210 Terminal mode SER1: data configuration

0	8 bit parity NONE
1	7 bit parity EVEN
2	7 bit parity ODD
3	8 bit parity EVEN
4	8 bit parity ODD

R O 213 Serial interface SER1: communication speed

0	300 bit/s
1	1200 bit/s
2	2400 bit/s
3	4800 bit/s
4	9600 bit/s
5	19200 bit/s
6	28800 bit/s
7	38400 bit/s
8	57600 bit/s
9	115 kbit/s

R O 214 Serial interface SER1: mode

0	Programming
1	EVS 112 bar code camera
2	Terminal / printing
3	-
4	Serial interface to computer (MODBUS slave)/modem
5	MODBUS master
	Note! DIP switch number 2 to ON position

R O 219 Terminal mode SER2: data configuration

0	8 bit parity NONE
1	7 bit parity EVEN
2	7 bit parity ODD
3	8 bit parity EVEN
4	8 bit parity ODD

R O 229 Serial interface SER2: communication speed

0	300 bit/s
1	1200 bit/s
2	2400 bit/s
3	4800 bit/s
4	9600 bit/s
5	19200 bit/s
6	28800 bit/s
7	38400 bit/s
8	57600 bit/s
9	115 kbit/s

R O 215 Serial interface SER2: mode

0	Terminal / printouts / modem
1	EVS112 bar code camera
2	-
3	-
4	Serial interface to computer (MODBUS slave)/modem
5	MODBUS master

R O 218 Terminal mode SER3: data configuration

0	8 bit parity NONE
1	7 bit parity EVEN
2	7 bit parity ODD
3	8 bit parity EVEN
4	8 bit parity ODD

R O 217 Serial interface SER3: communication speed

0	300 bit/s
1	1200 bit/s
2	2400 bit/s
3	4800 bit/s
4	9600 bit/s
5	19200 bit/s
6	28800 bit/s
7	38400 bit/s
8	57600 bit/s
9	115 kbit/s

R O 216 Serial interface SER3: mode

0	Terminal / printouts / modem
1	EVS112 bar code camera
2	-
3	-
4	Serial interface to computer (MODBUS slave)/modem
5	MODBUS master

R O 236 Analog input update rate

0	1 analog input per program cycle (default)/ each analog board
1	2 analog inputs per program cycle
2	4 analog inputs per program cycle
3	8 analog inputs per program cycle
4	2 analog inputs per program cycle of which one (input 0) will be updated every cycle
?	1 analog input per program cycle (default)

10. TROUBLE SHOOTING AND SERVICE

In the design of the AL2000 special attention has been paid to ease of service, rapid fault finding and correction. Special procedures have been built into the operation system which continuously supervise the AL2000's operation.

A good selection of spare modules (especially power supplies and CPUs), as well as cables and fuses is highly recommended. This will reduce expensive down-time in the event of a fault occurring.

10.1 Checking the Hardware Configuration

On power-up the AL2000 checks that the hardware configuration is the same as the one defined with the ALPro software. The hardware configuration table defines the following items:

Slot number of the module
The address and number of the I/O's in the module
Type code of the module
Action to be taken in the event that the hardware configuration is not the same as that defined in the ALPro Software:

After a 'warm start' with the START instruction the following operations are performed:

Status of memories and outputs remain unchanged
PLC program is moved from RAM memory into the EEPROM
PLC program is checked and executed if no errors found
After the second program cycle, parameters are read from the register memories

If the configuration for a particular I/O module is correct, the green ACT LED on that module should be on, and the red ERR LED should be off. In the event of an incorrect hardware configuration for a particular module, the red ERR LED on that module will be on. Information about the configuration error is simultaneously sent to a supervising PC via serial interface (provided that this function is enabled). After a hardware configuration error has been detected, the CPU will react in the manner specified during configuration with the ALPro software. The following alternatives are available:

STOP and RESET	Suspends execution of the PLC program and sends a reset signal to all the I/O modules, returning all outputs to 0.
STOP	Suspends execution of the PLC program, maintaining all outputs in their current state.
IGNORE	Continues execution of the PLC program. If the faulty module is replaced with a correct and working one, the CPU will resume using of that module.

The above checking procedure is performed on all defined I/O modules. Any modules not defined using the ALPro software are completely ignored by the CPU. This checking procedure supervises the address and data busses in the motherboard and I/O modules.

10.2 Tracing Faults and Solving Problems

AL2000 informs the user of any faults and malfunctions by the following means:

Operating system
LED indicators on all modules
Error messages sent via serial interface to a supervising PC
Error messages during PLC programming with the ALPro software

10.2.1 Hardware Faults

Trouble shooting table

Fault signal	Probable Cause	Remedy
RES signal on the CPU module	disturbances in the supply voltage	Check the power supply cabling and the PCON** and POW** modules
All LED signals are missing on the I/O module	disturbances in the supply voltage	Check the Power Supply cabling and the PCON and POW** modules
ERR signal on the I/O	The I/O module is not in the right slot The I/O module is broken	Check the hardware configuration module Replace the broken I/O module
The I/O modules do not operate in the expected manner	The PLC program uses wrong I/Os The definition of the analog I/O module is incorrect Incorrect resistors on the inputs/outputs of the analog I/O module Fuse in the relay output module is blown	Correct the program Check and correct the definitions Check the resistors and replace them with correct resistors Replace the fuse
The content of R O 241	Disturbances in the supply voltage is not equal to 0	Check the power supply

10.2.2 The ALPro Diagnostic Display

The ALPro programming software can be used to display diagnostic information to assist in the correction of hardware problems. This display can be produced by selecting **Diagnostics** from the **Configure** menu.

10.2.3 The I/O Module Status Table

Each I/O Module has one byte reserved for status information. The status bytes for the Modules in each slot are held in register outputs R O 0-31 (representing slot numbers 0-31 respectively). The status bytes contain possible error information as follows:

For All Modules:

0	No error!
101	Module does not match configuration table
102	Data transfer error between CPU and module
255	Module is not configured

For Analog Modules Only:

103-109	Error in write operation (due to module being in incorrect state)
110	e2prom not responding. This indicates an error in either the e2prom or the I2C serial link. This is a serious error condition because without calibration information from the e2prom all input values from that module are incorrect
111	Error in module's data bus
112	Data transfer error between CPU and module. Error occurred during module initialisation
113	Error writing to module's RAM memory
114-116	Error in write operation (due to module being in incorrect state)

10.2.4 The CPU Status Table

Register outputs R O 32-36 contain the following information about the CPU status:

R O 32	Watchdog test. If value is 255, then Watchdog is not present		
R O 33	Display/keypad version:	0	No display/keypad connected
		11	AL1094R
		15	AL1093
		19	AL1094
		21	AL1092
		34	AL1095
		67	AL1094AF
R O 34	Operating system version number. The version number is obtained by adding 200 to the contents of this register output		
R O 35	Status of DIP switches		
R O 36	Status of battery:	0(4)	Battery good
		1(5)	Battery low
		2(6)	Battery flat, faulty or missing
		4	Serial connection 3 power supply ok.
R O 37	Reserved		
R O 38-63	Reserved for possible future use.		

10.2.5 The Remote Unit Status Table

Each remote unit has one byte reserved for status information. This is held in register outputs R O 64-79 (representing remote unit numbers 32-47 respectively). The status bytes contain possible error information as follows:

0	No error!
33/161	Excessive bus traffic
34/162	Remote station not responding
35/163	Remote response framing error
36/164	Remote response break-up
37/165	CRC checksum error
144-159	Error in the module in remote unit slot 0-15 respectively

10.2.6 Software Faults

If the ALPro software gives the message PLC not connected, check the programming cable and communication speed/serial port selection.

10.3 The CPU Battery

The CPU2000S series modules require a 3-3.6V 1/2AA lithium battery (e.g. TADIRAN TL-5101, VARTA CR 1/2AA etc.) . This battery is used for the RAM and the clock and calendar IC. The life expectancy of the battery is over 2 years without the CPU being connected to a power supply (if the operating environment temperature does not exceed +60 C). However, under normal use the battery has a life expectancy of over 10 years.

Information about the condition of the battery is stored in R O 36. There are three possible battery states:

R O 36	0(4) - OK, battery in good condition
	1(5) - WARNING, battery weak and should be replaced
	2(6) - FAIL, battery exhausted, faulty or missing
	4 - Serial connection 3 power supply ok.

The AL2000 checks the battery on power-up. If the voltage is sufficient, the battery condition indicated in R O 36 is OK. If the voltage is close to minimum, the condition indicated is WARNING. The system program continues to check the battery condition and if it does not improve shortly after power-up, the condition indicated will be FAIL.

The system program continuously monitors the battery condition during operation. If the battery becomes exhausted, develops a fault or is removed, this information is immediately sent to R O 36.

10.3.1 Battery Replacement

The battery is mounted in a compartment on the CPU module immediately above the SER1 interface port. The battery can be replaced in the following way:

(1. Turn off power to the AL2000)
(2. Remove the CPU module from its slot)
3. Remove battery compartment cover
4. Remove old battery
5. Insert new battery with positive pole facing the edge of the PCB
6. Replace battery compartment cover
(7. Replace the CPU module in its slot)
(8. Turn on power to the AL2000)

Note: If there is data saved in the RAM memory of the PLC, ignore items 1, 2 ,7 and 8.

The CPU module must have an uninterrupted power supply in order to maintain the contents of the RAM and the correct time and date. An electrolytic capacitor preserves the memory contents for approximately one minute after disconnecting the power and removing the CPU module from its slot, during this time the battery can be changed without losing the memory contents. However, it is recommended that the battery be changed while the module is powered-up, thus avoiding any risk of losing the memory contents.

NOTE: After a battery change during CPU operation, R O 36 should be manually reset to OK. However, if the battery is changed during power-down time, R O 36 is automatically updated by the system program next time the CPU is powered-up.

10.4 Fuse Replacement

The power connector modules and the relay output modules are equipped with fuses. All fuses are located in separate sockets and are thus easily replaceable. Fuses are replaced in the following way:

10.4.1 PCON24

There are three fuse holders on the front panel of the module which house T1.6A, T1A and T2.5A, 5 x 20mm slow-blow fuses (to IEC 127/III standard).

The LED indicators on the front panel of the unit can be used to determine which of the three fuses is blown as shown below:

AUX	PLC	Blown Fuse
ON	OFF	FPLC / F1 (T1.6A)
OFF	ON	FAUX / F2 (T1A)
OFF	OFF	FIN / F3 (T2.5A)

1. Turn the fuse holder cap anti-clockwise about a quarter of a turn
2. Remove the cap and replace the blown fuse with a new one
3. Replace the cap. Push and turn the cap clockwise until locked.

10.4.2 PCON230

There is one fuse holder on the front panel of the module housing a T2.5A, 5 x 20mm slow-blow fuse (to IEC 127/III standard).

1. Turn the fuse holder cap anti-clockwise about a quarter of a turn
2. Remove the cap and replace the blown fuse with a new one
3. Replace the cap. Push and turn the cap clockwise until locked.

10.4.3 PWR3

There are three fuse holders on the front panel of the module which house a T315mA and 2 x T1A, 5 x 20mm slow-blow fuses (to IEC 127/III standard).

1. Turn the fuse holder cap anti-clockwise about a quarter of a turn
2. Remove the cap and replace the blown fuse with a new one
3. Replace the cap. Push and turn the cap clockwise until locked.

10.4.4 DOC32FP

This module has two T4A, 5 x 20mm slow-blow fuses (to IEC 127/III standard), one for each group. These are located behind the top edge of their respective connectors.

1. Remove the blown fuse by unplugging it
2. Replace the blown fuse with a new one

10.4. 5 ROC16K

In this module there are 16 x T2.5A* slow-blow fuses located behind their respective output terminals, and one spare fuse located behind the LED indicators. When the spare fuse is on place and working the SP.F LED on the front edge of the module is on. Thus the spare fuse holder provides a simple way to check fuses.

1. Remove the blown fuse by unplugging it
2. Replace the blown fuse with the spare fuse
3. Remember to order more spare fuses

10.4.6 ROC16Z

The procedure is exactly the same as for the ROC16K module except that ROC16Z uses 16 x T5A* slow-blow fuses.

10.4.7 OOC16

The procedure is exactly the same as for the ROC16K module except that the OOC16 uses 16 x F1.6A* quick-blow fuses.

* BUSS 'PC-Tron'
WICKMANN-WERKE 'TR5'
SCHURTER 'MSF Microfuse'

10.5 Adjusting of Analog Inputs

AL2000’s analog inputs are pre-calibrated prior to delivery and should require no further adjustment. The non-linearity of temperature measuring elements are compensated on analog boards system program. The resolution of the analog input is 12 bits.

However, for extremely accurate results, inputs can be re-calibrated in the following way.

On the table below the calibration variables and their functions are showed:

Register output	Meaning of variable	Min / max.values 8 bit 16 bit	Calculation of the value for adjustment variables
R O 220	Adjustment step	0 ... 5	
R O 221	Slot number of analog input	0 ... 31	
R O 222	Input number on analog board	0 ... 15	
R O 224	Lower adjustment value high byte	0 ... 15 (0 ... 4000)	500 = 1 * 256 +
R O 225	Lower adjustment value low byte	0 ... 255	244
R O 226	Upper adjustment value high byte	0 .... 15 (0 ... 4000)	3500 = 13 * 256 +
R O 227	Upper adjustment value low byte	0 ... 255	172

The initialization of adjustment step is controlled through register ouput R O 220.
The permitted adjustment steps are:

Before step 1 Slot number, Input number on analog board and both calibration variable values (into R O’s 224, 225, 226, 227) should be inserted

Step 1: Start adjustment
Step 2: Set the lower adjustment value for the input
Before going to step 3 the lower calibration value must be inserted
Step 3: Read the lower adjustment value
Step 4: Set the upper adjustment value for the input
Before going to step 5 the upper calibration value must be inserted
Step 5: Read the upper adjustment value

AS the result of succesful/unsuccesful adjustment of input PLC’s system program writes into register output R O 220 the result value:

0 adjustment was succesful
n > 5 error code

NOTE: When calibrating at the upper and lower calibration points, be sure to allow enough time for the sensor to stabilise. (e.g. allow sensor enough time to heat up of the boiling water).

The result of adjustment is that the zero point value from the PLC's analog input corresponds the minimum value of input and high value in PLC corresponds the high value of input. The value can be seen in the PLC as a number from 0 to 4095 (as a raw value).

10.5.1 What is needed for adjustment

You need the ALPro or AlproWin software, a PC, an accurate mA-meter, an accurate V-meter, an adjustable resistance reference and when calibrating current inputs millamper source.
For making the readings of the analog inputs you have to use the PC and the ALPro software. For the adjustment of the analog inputs of AL2000 CPU only one instruction has to be downloaded to the PLC; the STOP instruction. Through the AlproWin watch table you can see the values of all AL2000's analog inputs.

How to read the input values using the PC and AlproWin

1. Connect the AL2000 to the PC and start the AlproWin software.
2. If you want to read inputs in values from 0 to 1000, write the following program, otherwise you see the values from 0 to 4000.

```
STR   W   I   000
DIV   W   C   004
EQ    W   M   000
STOP
```

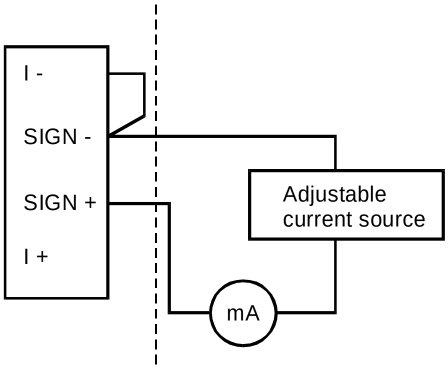
This program was made for the analog input 0. If you want to include all analogue inputs at once, you can add similar instructions for all the rest of the analog inputs

3. Compile program by choosing **Compile** from the main menu.
4. Transfer the program to the PLC by choosing **Transfer project** from the **Transfer** submenu and start PLC program execution.
5. Choose **Watch table** from the **View** submenu, insert desired variables to watch table and start updating by choosing **Start loop** from **Loop** submenu.
6. Now you can see the values of the variables on watch table.
   Check that the PLC's Run led is blinking indicating normal operation.
7. Begin calibration procedure.

10.5.2 Current input

You need an adjustable current source and a mA meter with enough accuracy (0.1%). Connect the adjustable current source to the analogue input number 1 (terminals SIGN+, SING- and I-).

Analog input 1.0

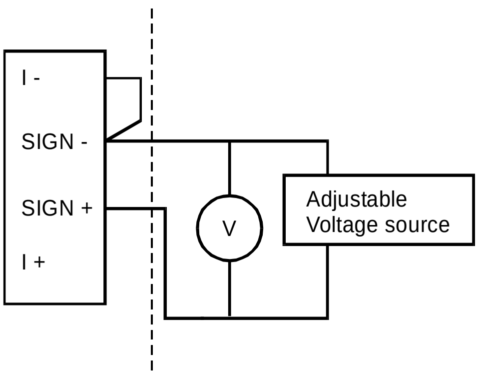


Note! When calibrating analog input 1.0 insert number 1 into R O 221 = 1 and number 0 into R O 222 = 0. Other inputs must be adjusted with corresponding numbers in R O 221 and in R O 222. (Input 0.4, R O 221 = 0 and RO 222 = 4 .)

10.5.3 Voltage input

You need an adjustable voltage source and a V meter with enough accuracy (0.1%). Connect the adjustable voltage source to the analogue input (terminals SIGN+, SING- and I-).

Analog input 2.1

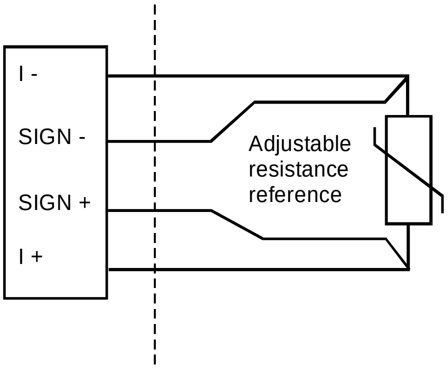


Note! When calibrating analog input 2.1 insert number 2 into R O 221 = 2 and number 1 into R O 222 = 1. Other inputs must be adjusted with corresponding numbers in R O 221 and in R O 222. (Input 3.5, R O 221 = 3 and RO 222 = 5 .)

10.5.4 Temperature input with PT100 sensor

You need an adjustable resistance reference 0.1 ... 250 Ω with an accuracy of 0.1%. Connect the adjustable resistance in the following way to terminals I+, SIGN+, SIGN- and I-.

Analog input 2.1



Note! When calibrating analog input 2.1 insert number 2 into R O 221 = 2 and number 1 into R O 222 = 1.
Other inputs must be adjusted with corresponding numbers in R O 221 and in R O 222. (Input 3.5, R O 221 = 3 and RO 222 = 5 .)

The calibration of an analog input is made as follows:

1. Start the adjustment by setting number one (1) to R O 220. The calibration program moves to step two (2) and waits you to insert the lower calibration values into R O 224 and R O 225.

2. Adjust the level of the analog signal to lower calibration value. In order to make the calibration of the Pt100 input as accurate as bossible over the measurement range, it is advisable to take the lower value about +10% higher than minimum measure value.

3. When the signal value corresponding to the lower calibration value can be read on AlproWin's watch table, set into R O 220 a value of three (3). Now the calibration program reads the signal value and moves to step four (4) to insertion of the upper calibration value into register outputs R O 226 and R O 227.

4. When the signal value corresponding to the upper calibration value can be read on ALPro's debug screen, set into R O 220 a value of five (5). Now the calibration program reads the signal value, calculates the values of the calibration parameters and saves the values to the FLASH memory. After that the calibration program informs the user if the calibration was succesful by writing into R O 220 the value zero (0), or if the result was unsuccesfull the error code. From the error codes the user can check what the problem was.

The error codes for analogue input calibration are as follows:

027 FLASH verification error
028 FLASH write error
128 input signal low > input signal high
129 adjustment value low > adjustment value high
130 input signal high > 4000
140 input address too high

The Pt100 sensor is nonlinear. The following temperatures corresponds to the resistance values given in the table (DIN 43 760) below:

°C	Ω	Diff. Ω	°C	Ω	Diff. Ω	°C	Ω	Diff. Ω
-50	80.31	0.40	55	121.32	0.39	160	161.04	0.37
-45	82.29	0.40	60	123.24	0.38	165	162.90	0.37
-40	84.27	0.39	65	125.16	0.39	170	164.76	0.37
-35	86.25	0.39	70	127.07	0.38	175	166.61	0.37
-30	88.22	0.39	75	128.98	0.38	180	168.46	0.37
-25	90.19	0.39	80	130.89	0.38	185	170.31	0.37
-20	92.16	0.39	85	132.80	0.38	190	172.16	0.37
-15	94.12	0.39	90	134.79	0.38	195	174.00	0.37
-10	96.06	0.39	95	136.60	0.38	200	175.84	0.37
-5	98.04	0.39	100	138.50	0.38	205	177.68	0.37
0	100	0.39	105	140.39	0.37	210	179.51	0.37
5	101.95	0.39	110	142.29	0.39	215	181.34	0.37
10	103.90	0.39	115	144.17	0.37	220	183.17	0.37
15	105.85	0.39	120	146.06	0.38	225	184.99	0.36
20	107.79	0.39	125	147.94	0.37	230	186.82	0.37
25	109.73	0.38	130	149.82	0.37	235	188.63	0.36
30	111.67	0.39	135	151.70	0.37	240	190.45	0.36
35	113.61	0.39	140	153.58	0.38	245	192.26	0.36
40	115.54	0.39	145	155.45	0.38	250	194.07	0.36
45	117.47	0.39	150	157.31	0.37	255	195.88	0.36
50	119.40	0.39	155	159.04	0.37	260	197.69	0.36

In the following table the calibration points are calculated for the two different, standard Pt100 measurement ranges available for AL2000.

PT100:

Temp. range °C	-50 ... +150	-200 ... +730
Resolution °C	0.05	0.25
OFFSET point 10% °C OFFSET point 20% °C	-30.0	 -5
Resistor value Ω Calibration constant (low)	88.22 400	98.04 800
GAIN point 80% °C GAIN point 90% °C	 +130.0	+544.0
Resistor value Ω Calibration constant (high)	149.82 3600	295.43 3200

On the PC you should see the following values when the adjustment is correct:

Input range	OFFSET 10%	GAIN 90%
-50 ... 150	400	3600

11. PROGRAMMING

(**Paragraph 11.1** discusses the available alternative programming methods. Since separate documentation is provided for the programming programs, they are only briefly outlined below.)

11.1 Programming methods

The programming device can be any IBM compatible PC computer. The program is transferred through the programming cable to the PLC. When programming with a PC, the AlPro programming software is used. With the programming software, the PLC program can be written either in OFF LINE mode (without a PLC, for subsequent transfer) or in ON LINE mode (the program is immediately transferred to the PLC).

The AL 2000 series Programmable Logic Controller incorporates a programming port Ser1, (information on the operation of instructions and commands) and a RS-232C interface. Accordingly, the following alternative programming methods are available:

- the **ALProWin** programming software
- the **ALPro** programming software
- a PROM programmer

A complete program for the PLC can be produced using the programming software ALProWin or ALPro, which run on a personal computer. The programming software also allows for off-line programming and versatile program documentation etc.

Memories of the PLC and programs:

When a program is transferred from a programming device to the PLC, the program first goes to RAM and, after completion of the START command, to FLASH memory. When a logic program is transferred from the PLC to a programming device, the program is first stored from RAM to FLASH, and then transferred to the programming device.
The logic program is stored in a FLASH memory. When power is applied to the Programmable Logic Controller, the program is read into RAM for execution.

11.1.1 Programming with a PROM programmer

When a number of PLCs with identical functions are required, programming is easiest to do by duplication. First program one FLASH using one of the programming programs or a terminal and then copy the finished program into the program memories of the remaining PLCs with a FLASH programmer.

11.1.2 Programming with ALProWin/ALPro

The program, with comments, is first written in plain text form using the text editor of ALProWin. (Any other text editor producing ASCII text will do just as well.) The program is then compiled with ALProWin compiler into a form that the Programmable Logic Controller understands, stored on a diskette, and transferred into the memory of the PLC when desired.

Further features of ALProWin:

- **Macros:** Emphasizing the program structure. (Page division, writing similar program sections, etc.)
  Windowing: Error messages easy to include in the listing file.
- **ON-LINE features** including display of variables, history display, on-line editor, etc. For example, the basic displays can be stored on disk, which facilitates the clearing of fault conditions.
- ALProWin contains HELP screens describing the functions.
- Available in Finnish and English language versions.

A free version of the program, ALProWin-demo or ALPro-demo, is also available (similar to ALProWin / ALPro proper, but the length of the program that can be written is limited) as well as a manual (ALProWin / ALPro Programming manual).

11.1.3 Fast program loop

A fast program loop can be written into the Programmable Logic Controller for execution at timer interrupts, at intervals of 5 ms. The executable program of the loop starts as program line 0000 and ends at an END instruction.

When the fast program loop is in use, the PLC reads inputs 0 - 7 and writes to outputs 0 - 7 every time before executing the fast loop.

The differentiation of auxiliary memories 0 - 7 is timed by the fast loop.

The maximum permissible length of the fast loop is 62 instructions, but it is advisable to keep it as short as possible in order to avoid slowing down the main program more than necessary.

No PRT instructions are allowed in the fast loop.

CAUTION! The END instruction must not be written while program execution is in progress. The program must always be halted when instructions preceding the END instruction are deleted, added, or changed.

11.2 AL2000 variables

11.2.1 One bit variables

I	Input, 0 - 7 off. Numbered 000...007.
NI	One's complement of input. When I n=1 , NI n=0.
M	Auxiliary memory, 256 off. Numbered 000...255.
BM, GM	Auxiliary memory, 256 off. Numbered 000...255.
NM, NB, NG	One's complement of auxiliary memory.
O	Output, 0 - 7 off. Numbered 000...007. The unused ouputs and ouputs 008...255 can also be used as one-bit auxiliary memory.
NO	One's complement of output.
SM, SB, SG, SL	Conditional set of memory or output, 256 off. Used in conjunction with EQ instruction.
RM, RB, RG, RL	Conditional reset of memory or output, 256 off. Used in conjunction with EQ instruction.
DP	Change of auxiliary memory from 0 to 1, 128 off. Numbered 000...127. Compares the state of the memory to its state at the beginning of the program cycle.
DN	Change of auxiliary memory from 1 to 0, 128 off. Numbered 000...127. Compares the state of the memory to its state at the beginning of the program cycle.
P	Pulse variable. The variable is one for the duration of one program cycle, P000 ten times per second, P001 once per second and P002 once per minute.
S	Sequence register or a step of a sequence register. There are 32 sequence registers, with 256 step each.
TX	String constant (0 - 255). These are text strings with a maximum length of 16 ASCII characters each. Used in conjunction with the PRT or EQU R TX instructions
T	Timer, 80 off. Numbered 000...79. The resolution and range of the timers varies as follows:

	resolution	range
T 000...003	10 ms	0,02...2,55 s
T 004...007	100 ms	0,2...25,5 s
T 008...015	1 s	2...255 s
T 016...079	100 ms	0.2...25.5s

In the IF and STEP instructions T refers to the bit accumulator.
In the PRT instruction T refers to TEXT.

C	Counter, 16 off. Numbered from 000 to 015. The counters count down; their range is 255 . . 0. In the PRT instruction C refers to to a numerical value.

11.1.2 Register variables or 8 bit variables

R M, GM	Register memory, 256 off. Numbered from 000 to 255.
R NM, NG	One's complement of register memory.
R O	Register output. Register outputs are reserved for PLC's system program. Register outputs 0 and 1 are the analogue outputs from PLC.
R NO	One's complement of register output.
R SM, SG, SL	Conditional set of register memory or output. Used in conjunction with EQ instruction.
R RM, RG, RL	Conditional reset of register memory or output. Used in conjunction with EQ instruction.
R T	Register timer. Identical with the timers listed above; T or R T is used as the variable according to the instruction. Also refers to register accumulator in some instructions.
R C	Register constant 000...255 (decimal), except register counter with the READ and LOAD instructions. C or R C is used as the variable according the instruction.
Q	Queue, 4 off. Numbered from 0 to 3. The queue length can be 1 - 256.
F	FIFO store, 8 off. Numbered from 0 to 7. The FIFO store has 256 locations.

11.2.3 Word variables or 16 bit variables

W I	Word input, 0 - 255 off. Numbering depends on configuration.
W M	Word memory, 16384 off. Directly addressable numbered from 000 to 256. Indirectly addressable from 000 to 16384 (EQI and STI instructions). Word memories 0 to 127 overlaps with register memories 0 - 255.
W O	Word output, 4096 off. Directly addressable numbered from 000 to 256. Indirectly addressable from 000 to 4096 (EQI and STI instructions).
W SM, SO	Conditional set of directly addressable word memory or output.
W RM, RO	Conditional reset of directly addressable word memory or output.
W T	Word accumulator.
W C	Word constant 0...9999 (decimal).

NOTE! Refer to chapter 15 at the end part of this manual:
- AL2000 series PLC's memory and I/O maps
- AL2000 series PLC's data retention during power failure

11.3 Error messages

The programming software issues error messages when it encounters error conditions during programming:

ERROR 010	No AutoLog instruction on program line
ERROR 020	Instruction write to memory failed
ERROR 021	Instruction write to memory failed
ERROR 022	Instruction write to memory failed
ERROR 025	The instruction entered is not valid
ERROR 030	Variable number too large
ERROR 031	Decimal number too large
ERROR 032	Decimal number setting too high
ERROR 033	Invalid octal number
ERROR 040	Instruction not found in program by FIND
ERROR 044	Not an AutoLog 16 instruction
ERROR 050	Transfer exceeds program space
ERROR 060	Hex file read error
ERROR 080	Invalid variable (DISP)
ERROR 085	Insertion pushes program beyond 2047/4095
ERROR 090	Unidentified instruction in program at start
ERROR 091	Address of END instruction is 62 = 76 oct
ERROR 092	A second END instruction in the program
ERROR 093	No STOP instruction in the program

11.4 AL2000S Instructions

BA	Bit Accumulator	RA	Register Accumulator
WA	Word Accumulator		
n	variable number	d	constant

- **AND I/M/O/NI/NM/NO/BM/GM/NB/NG/DP/DN/P n**

Operation: Sets the bit accumulator equal to the logical product of its old value and the variable.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR   I    0.1    ; If input I 1 from card 0 is ON and
AND   NI   0.2    ; input I 2 from card 0 is OFF
```

- **AND S d**

Operation: If the bit accumulator is 1 before the instruction and the current sequence register is at step d, the bit accumulator remains at 1; otherwise it is reset to 0.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR    I    0.0    ;If input 0 is 1 and the
AND    S    019    ;sequence register is at step 19,
EQ     SM   201    ;set memory 201 to 1.
```

- **AND R C d**

Operation: Sets the register accumulator equal to the logical product of its old value and the constant d (0 - 255).

Affected

BA	RA	WA	Variable
No	Yes	No	No

Example

```
STR   R   M   002    ;R M  2   = 10010001 B
AND   R   C   015    ;         = 00001111 B
EQ    R   M   006    ;R M 6    = 00000001 B
```

- **AND R M/O/NM/NO n**

Operation: Sets the register accumulator equal to the bit-by-bit logical product of its old value and the variable.

Affected

BA	RA	WA	Variable
No	Yes	No	No

Example

```
STR   R   M   002    ;R M 2   = 10010001 B
AND   R   M   005    ;R M 5   = 00110001 B
EQ    R   M   006    ;R M 6   = 00010001 B
```

- **AND W C d**

Operation Set the word accumulator equal to the logical product of its old value and the constant d.

Affected

BA	RA	WA	Variable
No	No	Yes	No

Example

```
STR  W  I   2.3      ;W I 2.3   = 0000 0011 0000 1101 B
AND  W  C   02047    ;          = 0000 0111 1111 1111 B
EQ   W  M   015      ;W M 15    = 0000 0011 0000 1101 B
```

- **AND W I/M/O n**

Operation Set the word accumulator equal to the logical product of its old value and variable n.

Affected

BA	RA	WA	Variable
No	No	Yes	No

Example

```
STR  W  I   2.3      ;W I 2.3   = 0000 0011 0000 1101 B
AND  W  M   014      ;W M 14    = 0000 0000 0010 0110 B
EQ   W  M   015      ;W M 15    = 0000 0000 0000 0100 B
```

- **BCD R T**

Operation Converts the number in the register accumulator into a two-digit BCD number 00 - 99; if the number in the accumulator is greater than 99, the result is indeterminate.

Affected

BA	RA	WA	Variable
No	Yes	No	No

Example

```
STR  R  M   000      ;Acc.  = 0101 0000B   (= 80 DES)
BCD  R  T            ;now   = 1000 0000B   (= 80 BCD)
```

- **BCD W T**

Operation Converts the number in the word accumulator into a four-digit BCD number; if the number in the accumulator is greater than 9999, the result is indeterminate.

Affected

BA	RA	WA	Variable
No	No	Yes	No

Example

```
STR  W  C   04396    ;W A before = 0001 0001 0001 0001 B
BCD  W  T            ;W A after = 0100 0011 1001 0110 B
                     ; before ( 4396 DEC) after ( 4396 BCD)
EQ   W  M   000      ;W M 0 = 17302 DEC
```

- **BIN R T**

Operation: Converts the contents of the register accumulator into binary format, assuming it contained a two-digit BCD number before the instruction; if the accumulator does not contain a BCD number, the result is indeterminate.

Affected:

BA	RA	WA	Variable
No	Yes	No	No

Example:

```
STR  R  C   148     ;Acc.  = 1001 0100B   (= 94 BCD)
BIN  R  T           ;now   = 0101 1110B   (= 94 DES)
```

- **BIN W T**

Operation: Converts the contents of the word accumulator into binary format, assuming it contained a four-digit BCD number before the instruction; if the accumulator does not contain a BCD number, the result is indeterminate.

Affected:

BA	RA	WA	Variable
No	No	Yes	No

Example:

```
STR  W  C  00512    ;W A before=0000 0010 0000 0000 B
BIN  W  T           ;W A after  = 0000 0000 1100 1000 B
                    ; before W A = 200 BCD, after 200 DEC
```

- **BIT M/O/BM/GM n**

Operation: Moves the content of the register accumulator into 8 successive bit variables such that the least significant bit goes to address n, the next to address n + 1, etc., and the most significant bit to address n + 7.

Register accumulator	Variable
bit 0	n
bit 1	n + 1
bit 2	n + 2
bit 3	n + 3
bit 4	n + 4
bit 5	n + 5
bit 6	n + 6
bit 7	n + 7

Affected:

BA	RA	WA	Variable
No	No	No	Yes

Example:

```
STR  R  C   130     ;= 10000010B
BIT     O  2.08     ;Turn on outputs 2.15 and 2.09
```

- **BYT I/M/O/BM/GM n**

Operation Converts 8 successive variables to a byte in the register accumulator. Variable n becomes the least significant bit and variable n + 7 the most significant bit.

Variable	Register accumulator
n	bit 0
n + 1	bit 1
n + 2	bit 2
n + 3	bit 3
n + 4	bit 4
n + 5	bit 5
n + 6	bit 6
n + 7	bit 7

Affected

BA	RA	WA	Variable
No	Yes	No	No

Example

```
BYT         I    0.00     ;The time setting for
BIN                       ;timer 6 is read from
STR         NI   0.10     ;inputs, for example
LOAD   R    T     006     ;a thumbwheel switch.
```

- **CLO R M/O n**

Operation Compares the 16-bit value in register memories RO 250 and RO 251 (RO 250 = high byte) with the variable given as the instruction parameter. The bit accumulator is reset to 0 when (RO 250, R0 251) = (R M/O n, R M/O n + 1).
CAUTION: The 16-bit values compared must be in the same format, i.e. both binary or both BCD.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
CLO    R    M     192     ;Output 012 is turned on
EQ          O    0.12     ;when the clock time reaches
CLO    R    M     194     ;the value written into RM 192
INV                       ;and RM 193, and off when the
AND         O    0.12     ;clock time reaches the value
EQ          O    0.12     ;written into RM 194 and RM 195.
```

- **CONT**

Operation Terminates the skipping of instructions. (The STOP instruction also has this effect.) The first CONT instruction encountered stops the effect of all nested IF instructions.

Affected

BA	RA	WA	Variable
No	No	No	No

Example

```
IF          I    0.00     ;If Input 000 = 1,
STR         I    0.01     ;then ...
EQ          O    1.01
IF          I    0.02     ;if Input 000 = 1 and Input 002 = 1,
STR         O    1.03     ;then ...
EQ          O    1.04
CONT                      ;Continue here in any case.
```

- **CSR n**

Operation: The PLC executes the subprogram n if the bit accumulator is true

	BA	RA	WA	Variable
Affected	No	Yes	No	No

The states of the accumulators remain unchanged, so subroutine can use the accumulator states.
See page 11-40.

- **DCD R M/O n**

Operation: Decodes the value of the variable n into the register accumulator. If the variable is between 0 and 7, the corresponding bit is set to 1 and the remaining bits are reset to 0. If the variable is greater than 7, the register accumulator is reset to 0.
The value 0 corresponds to the least significant bit and 7 to the most significant bit.

	BA	RA	WA	Variable
Affected	No	Yes	No	No

Example

```
STR    R    C     006
EQ     R    M     000
DCD    R    M     000          ;Acc. = 01000000B
```

Value of variable	Register accumulator after DCD instruction
0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128
>7	0

- **DCR C n**

Operation: If the bit accumulator is 1, decrement counter n by 1. This instruction does not perform differentiation - for example, when counting pulses, the program must ensure that the counter is decremented by one for each pulse.

	BA	RA	WA	Variable
Affected	No	No	No	Yes

Example

```
STR      NI   1.00
LOAD      C    000   100    ;Count 100 pulses
EQ        O   0.00
STR       I   1.01
EQ        M    001
STR      DP    001          ;Differentiate input 1
DCR       C    001
```

- **DEC R M/O n**

Operation: If the bit accumulator is 1, decrement the variable by 1 and load the value of the variable into the register accumulator. Decrementing a variable whose value is 0 gives 255. If the bit accumulator is 0, the variable is not decremented but only loaded into the register accumulator. If the variable was decremented (the bit accu mulator was 1) and the result was 255 (zero minus 1), the bit accumulator is set to 1; otherwise the bit accumulator remains at 0.

Affected

BA	RA	WA	Variable
Yes	Yes	No	Yes

Example

```
STR      P   001     ;Decrement memory 112
DEC   R  M   112     ;once per second. Every time
XOR      O   001     ;the memory goes from
EQ       O   001     ;0 to 255, invert output 1.
```

- **DEC W M/O n**

Operation: If the bit accumulator is 1, decrement variable n by 1 and load the new value of the variable into the word accumulator. If the bit accumulator is 0, the variable is not decremented but only loaded into the word accumulator. Decrementing a variable whose value is 0 gives 65535. If the variable is decremented (the bit accumulator was 1)and the result is 65535, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0.

Affected

BA	RA	WA	Variable
Yes	No	Yes	Yes

Example

```
STR      P   000     ;Decrement memory 130
DEC   W  M   130     ;10 times per second. Every time
XOR      O   000     ;the memory goes from
EQ       O   000     ;0 to 65535, invert output 0.
```

- **DIV R C d**

Operation: Divide the register accumulator by the constant d. The quotient remains in the register accumulator and the remainder is stored in register output RM 244.

Affected

BA	RA	WA	Variable
No	Yes	No	No

Example

```
STR   R  M   010     ;RM 10 = 15
DIV   R  C   006     ;Acc. = 2
                     ;RM 244 = 3
```

- **DIV R M/O n**

Operation: Divide the register accumulator by the variable n. The quotient remains in the register accumulator and the remainder is stored in register output RM 244.

Affected

BA	RA	WA	Variable
No	Yes	No	No

Example

```
STR  R  M  100
DIV  R  O  000
```

- **DIV W I/M/O n**

Operation: Divide the word accumulator by the variable n. The quotient remains in the word accumulator and the remainder is stored in register outputs RO 230,231 (MSB/LSB).

Affected

BA	RA	WA	Variable
No	No	Yes	No

Example

```
STR  W  M  025
DIV  W  I  010      ;result is in word accumulator
EQ   W  M  010
STR  R  O  230
EQ   R  M  230
STR  R  O  231
EQ   R  M  231
STR  W  M  115
EQ   W  M  011      ; the remainder is in  W M 011
```

- **DIV W C d**

Operation: Divide the word accumulator by the constant d. The quotient remains in the word accumulator and the remainder is stored in register outputs RO 230,231 (MSB/LSB).

Affected

BA	RA	WA	Variable
No	No	Yes	No

Example

```
STR  W  M    025
DIV  W  C  01040
```

- **END**

Operation: Last instruction of the fast program loop. CAUTION: The END instruction must not be written while the program is being executed. The program execution must always be suspended if any of the instructions preceding the END instruction are removed, added, or altered.

Affected

BA	RA	WA	Variable
No	No	No	No

Example

```
STR   NO  000   ;A 100 Hz oscillator.
EQ    O   000   ;This loop runs every 5 ms.
END
STR   I ...     ;The regular program starts here.
```

- **EQ M/O/BM/GM n**

Operation: Sets the variable equal to the contents of the bit accumulator.

Affected

BA	RA	WA	Variable
No	No	No	Yes

Example

```
STR     I   1.00        ;Output 0
EQ      O   0.00        ;follows input 0.
```

- **EQ SM/SO/SB/SG n**

Operation: If the bit accumulator is 1, set the variable to 1;
if the bit accumulator is 0, do nothing.

Affected

BA	RA	WA	Variable
No	No	No	Yes

Example

```
STR     I   1.00        ;Input 0 sets
EQ     SO   0.00        ;output 10 to 1.
```

- **EQ RM/RO/RB/RG n**

Operation: If the bit accumulator is 1, reset the variable to 0;
if the bit accumulator is 0, do nothing.

Affected

BA	RA	WA	Variable
No	No	No	Yes

Example

```
STR     I   1.00        ;Input I 0.00 resets
EQ     RO   0.00        ;output O 1.00.
```

- **EQ R M/O n**

Operation: Sets the variable equal to the contents of the register accumulator.

Affected

BA	RA	WA	Variable
No	No	No	Yes

Example

```
STR  R  C   085         ;Constant 85 loaded into
EQ   R  M   211         ;register memory 211
```

- **EQ R RM/RO n**

Operation: If the bit accumulator is 1, reset the variable to 0;
if the bit accumulator is 0, do nothing.

Affected

BA	RA	WA	Variable
No	No	No	Yes

Example

```
STR  R  C   123         ;If input 10 = 1,
EQ   R  M   100         ;register memory 100
STR     I   010         ;is set to 0,
EQ   R  RM  100         ;otherwise 123.
```

- **EQ R SM/SO n**

Operation: If the bit accumulator is 1, set the variable equal to the contents of the register accumulator; if the bit accumulator is 0, do nothing.

Affected:

BA	RA	WA	Variable
No	No	No	Yes

Example:

```
STR  R  C    123
EQ   R  M    100     ;If input I1 = 1,
STR  R  C    200     ;register memory 100 = 200,
STR     I    0.01    ;otherwise register
EQ   R  SM   100     ;memory 100 = 123.
```

- **EQ W M/O n**

Operation: Set the variable equal to the contents of the word accumulator.

Affected:

BA	RA	WA	Variable
No	No	No	Yes

Example:

```
STR  W  I    2.00    ;Word memory 3
EQ   W  M    003     ;follows analog input 0.
```

- **EQ W RM/RO n**

Operation: If the bit accumulator is 1, reset the variable to 0; if the bit accumulator is 0, do nothing.

Affected:

BA	RA	WA	Variable
No	No	No	Yes

Example:

```
STR  W  C    03000   ;If input I004 = 1,
EQ   W  M    130     ;word memory 130
STR     I    0.04    ;is set to 0,
EQ   W  RM   130     ;otherwise 3000.
```

- **EQ W SM/SO n**

Operation: If the bit accumulator is 1, set variable n equal to the contents of the word accumulator; if the bit accumulator is 0, do nothing.

Affected:

BA	RA	WA	Variable
No	No	No	Yes

Example:

```
STR  W  C    04500
EQ   W  M    130     ;If input I004 = 1,
STR  W  C    05000   ;word memory 130 = 5000,
STR     I    0.04    ;otherwise word
EQ   W  SM   130     ;memory 130 = 4500.
```

- **EQI M/O/BM/GM**

Operation: Writes the state of bit accumulator into bit output/memory. Register accumulator is pointing to the bit variable to be written.

	BA	RA	WA	Variable
Affected	No	No	No	Yes

Example

```
STR   R   C    008      ;program sets the state of bit accumulator
EQI       O             ;into output 8.
```

- **EQI R M/O n**

Operation: Loads the content of the register accumulator into the variable whose address is the value of the variable given as the instruction parameter

	BA	RA	WA	Variable
Affected	No	No	No	Yes

Example

```
STR   R   C    001      ;Initialize pointer
EQ    R   M    000      ;R M 0=1
STR       I   0.01      ;take posite derivation
EQ        M      1      ;from input 0
STR      DP      1      ;every time input goes active
INC   R   M    000      ;Increment pointer &
STR   R   O    252      ;save
EQI   R   M    000      ;Seconds into memory 2
```

- **EQI W M/O n**

Operation: Loads the content of the word accumulator into the variable whose address is the value of the variable given as the instruction. Accessible addresses are W M variables 0 to 16383, W O variables 0 to 4095 (check from memory tables)

	BA	RA	WA	Variable
Affected	No	No	No	Yes

Example

```
STR   W   C     11      ; Initialize pointer
EQ    W   M    000
STR   W   M    125      ;Load hour & minutes into WA
EQI   W   M    000      ;value of word accumulator into memory 11
```

- **EQU R C d**

Operation: If the constant d is equal to the register accumulator, the bit accumulator is set to 1; if unequal, the bit accumulator is reset to 0.

	BA	RA	WA	Variable
Affected	Yes	No	No	No

Example

```
STR   R   M    001
EQU   R   C    010
EQ        O   1.01      ;There may be several
EQU   R   C    020      ;successive comparisons
EQ        O   1.02
```

- **EQU R M/O n**

Operation: If variable n is equal to the number in the register accumulator, the bit accumulator is set to 1; if unequal, the bit accumulator is reset to 0.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR   R  M    001      ;If memories 1 and 2
EQU   R  M    002      ;are equal,
EQ       O    1.00     ;output 0 is 1.
```

- **EQU R TX n**

Operation: Compares if the content of text string (TX n) n is found in given FIFO n. If an identical text string is found, the bit accumulator is set to 1. Into register accumulator is transferred BCD number which is formed from the next two characters after the string found. If the characters don't represent a BCD number, the register accumulator becomes zero. Into the word accumulator is transferred the ASCII codes of the former characters. The FIFO number to be compared is given in register accumulator.

Affected

BA	RA	WA	Variable
Yes	Yes	Yes	No

Example

```
STR      C    001      ; Bit accumulator to 1
EQ    R  RM   024      ; reset W M 12
EQ    W  RM   025      ; reset W M 12
STR   R  C    005      ; Compare if FIFO 5 (ser 1)
EQU   R  TX   024      ; includes the text srting (TX 24)
EQ       SM   024      ; Save the result to bit memory, the BCD
EQ    R  SM   024      ; number into R M 24 and characters
EQ    W  SM   025      ; ASCII codes into W M 25 (R M 50, 51)
```

- **EQU W I/M/O n**

Operation: If variable n is equal to the number in the word accumulator, the bit accumulator is set to 1; if unequal, the bit accumulator is reset to 0.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR   W  C    03000    ;If word memory 37
EQU   W  M    037      ;equals 3000,
EQ       O    0.08     ;output 8 is on.
```

- **EQU W C d**

Operation: If the constant d is equal to the number in the word accumulator, the bit accumulator is set to 1; if unequal, the bit accumulator is reset to 0.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR   W  M    001
EQU   W  C    01000
EQ       O    0.01     ;There may be several
EQU   W  C    02000    ;successive comparisons
EQ       SO   0.01
```

- **FCN n**

Operation

If the bit accumulator is true, the FCN instruction executes one of the following functions. After successful execution of the instruction the bit accumulator is true. An undefined function call or illegal calculation parameter(s) may cause bit accumulator to reset.
CAUTION! The FCN instruction must not be used in the fast program loop!

	BA	RA	WA	Variable
Affected	Yes	No	Yes	No

n selects the function for execution calculationtime
0 = logarithm a. 13 ms
1 = natural logarithm a. 12 ms
2 = sqrt (square) a. 7 ms
8 = percent
10 = scaling of word variable a. 7 ms
11 = scaling of word variable a. 8 ms
output: sign, integer and decimal part
24 = Send modbus message
30 = write to I^2C channel
31 = read from I^2C channel
32 = read serial number from Ibutton

Before calling the FCN instruction the variables content, which is to be scaled, must be written to word accumulator. After execution of FCN instruction the result of calculation is in word accumulator. The resolution of the result depends on the used function.
0 log 65535= 4.8164, in word acc. is 48164
1 ln 65535= 11.090, in word acc. is 11090
2 sqrt 65535= 255.99, in word acc. is 25599
8 %, value in WA, number in RA = percent value, after execution the result is in word accumulator (WA * RA)/100

- **FCN 10 Scaling**

In WA is the number to be scaled, in RA is the address of word memory pointer, that contains the scaling parameters. The range of scale word memories is freely selectable. The order of scaling parameters is:
W M 100 = low limit of measured input (0 ... 4095)
W M 101 = high limit of measured input (0 ... 4095)
W M 102 = low range scaled output (eg. 10 (°C))
W M 103 = high range scaled output (eg. 115 (°C))

Example

```
STR   W   C     0800   ; low limit of input (4 - 20 mA)
EQ    W   M      100   ; wanted output 0 - 1000 (tenths of percent)
STR   W   C     4095   ; high limit of input
EQ    W   M      101
STR   W   C     0000   ; low limit of output
EQ    W   M      102
STR   W   C     1000   ; high limit of output
EQ    W   M      103
STR   R   C      100   ; parameters start address to RA
STR   W   I     0.01   ; word input to be scaled into WA
STR       C      001   ; set bit accumulator to 1
FCN              010   ; call scaling function
EQ    W   M      200   ; save the result to WM 200
STOP
```

- **FCN 11 Scaling, returns sign,integer- and decimat part**

In WA is the number to be scaled, in RA is the address of word memory pointer, that points to the scaling parameters. Pointer offset 0 points to W M area, offset 32768 points to W O area.The range of scale word memories is freely selectable.
The order of scaling parameters is:

W	M	100	= low limit of measured input (0 ... 4095)	
W	M	101	= high limit of measured input (0 ... 4095)	
W	M	102	= low range scaled output (eg. 10 (°C))	
W	M	103	= high range scaled output (eg. 115 (°C))	
W	M	104	= offset to 0-value	(input data)
W	M	105	= decimal count 0..4	(input data)
W	M	106	= Sign 0=plus, 1=negative	(output data)
W	M	107	= interger part	(output data)
W	M	108	= decimal part	(output data)

Example

```
STR   W   C    00000   ; low limit of input (Pt100 -50-150°C)
EQ    W   M      100   ; wanted output -50.0 - 150.0 (tenths of percent)
STR   W   C    04000   ; high limit of input
EQ    W   M      101
STR   W   C    00000   ; low limit of output
EQ    W   M      102
STR   W   C    00200   ; high limit of output
EQ    W   M      103
STR   W   C    01000   ;offset (1000=0°C)
EQ    W   M      104
STR   W   C    00001   ;decimals
EQ    W   M      105
STR   W   C    00100   ; Intialize pointer 2
EQ    W   M      099   ; points to W M 100
STR   R   C      099   ; Initial pointer 1 points to W M 099
STR   W   I     0.01   ; word input to be scaled into WA
STR       C      001   ; set bit accumulator to 1
FCN              011   ; call scaling function
STOP
```

- **FCN 24 Launch modbus message to serial channel**

Command is executed only if Bit accu equals 1 before instruction.
Function parameters are defined in word memories. Register accu must hold the address of the first parameter before FCN instruction

Parameters: 1st parameter:Serial Channel (1= Ser1, 2=Ser2, 3 = Ser3)
2nd parameter:Message number (0..511)

Execution time is about 0.03 ms

FNC 24 checks the parameter values and if parameters are OK it will add the message number to the first place in message queue and returns with bit accu set to 1.
If either parameter is not good, FCN will return with bit accu set to 0.
If message already exists in message queue, the previous message request will be re moved from message list.
If current "transfer in progress" is the same message as requested message, request is ignored. Function will however return 1.

Example 1 Send message every time input goes actice

```
Parameters:          W  M    010   :3    ;Serial channel 3
                     W  M    011   :10   ;message number
Program:     STR          I    0.0         ;
             EQ           M    000
             STR         DP    000         ; On positive derivation
             STR      R   C    010         ; Get parameters from W M 10->
             FCN               024
             INV
             EQ          SM  ERROR         ;
             STOP
```

- **FCN 32 Read Ibutton serialcode**

This command is used to read serial code from IButton device connected into Autolog PLC's I2C channel. Command is executed only if Bit accu equals 1 before instruction.
Function parameter is defined in register accu. Register accu must hold the address of the 1st byte save location
Execution time is 1.4 ms
If there is no ibutton connected to PLC, command retuns bit accu = 0 and register accu = 0.
If there is iButton connected but CRC does not match, bit accu = 0 and register accu <> 0.
If there is Ibutton connected and it was successfully read, bit accu = 1 and register accu = 0.

Example 1
Check for iButton every 1 second

```
Program:  STR          P   001
          STR      R   C   000      ; serial code saved into R M 0 ->
          FCN              032      ; check for ibutton
          IF           T            ; if ibutton was attached to PLC and
                                    ; data was succesfully read
          STR      R   M   000      ; make comparison
          EQU      R   M   100      ; device code
          EQ           M   000      ; result to memory 0
          STR      R   M   001      ; 1st data byte
          EQ       R   M   101      ;
          AND          M   000
          EQ           M   000
          Etc.
          CONT
```

Example 2
Read iButton once in a second
Program:

```
STR   R  C   000        ;parameters from R M 000 ->
STR      P   001        ;every second…
IF       T              ;parametrit alkaen R M 10 ->
FCN          032        ;Read possible Ibutton
EQ       O   005        ;set output 5 active
CONT
```

- **FCN 30** and **FCN 31** **I^2C Write (FCN30) & read (FCN31) functions**

These commands are used to communicate with external I^2C devices connected into Autolog PLC's I^2C channel.
Command is executed only if Bit accu equals 1 before instruction.
Function parameters are defined in word memories. Register accu must hold the address of the first parameter before FCN instruction

Parameters: 1st parameter: Slave address
2nd parameter: Byte count
3rd parameter: Data 0
nth parameter: Data n

Execution time is about 0.1 ms /byte

For read function (FCN 31), last byte is always read without ACK.
Read bytes are stored right after parameter "byte count "

When you wish to connect I2C devices to PLC, refer always to specification provided by the device manufacturer.(device address, READ/WRITE operations etc.)
Also make shore that there are no duplicated device addresses in I^2C-channel.

Example 1 Write Outputs every 1 second

```
Parameters:         R  M   010  : 4Ch   ; I2C I/O.board address
                    R  M   011  : 1     ; byte count
                    R  M   012  : 0Fh   ; output data.
Program:      STR      P   001
              STR   R  C   010          ; parameters R M 10 ->
              FCN          030          ; Write
```

Example 2 Read keypad

```
Parameters:         R  M   020  : 40h   ; AL1093 keypad address
                    R  M   021  : 1
                    R  M   022          ; keypad data
Program:      STR   R  C   020          ; parameters R M 020 ->
              FCN          031          ; read
```

Example 3 Read Clock 8583

```
Parameters:       R   M   010   : A0h   ; Base address
                  R   M   011   : 1     ;byte count
                  R   M   012   : 1     ;data address
                  R   M   013   : A0h   ;Base address
                  R   M   014   : 6     ;byte count
                  R   M   015           : clock data: hundredth of a second
                  R   M   016           : clock data: seconds
                  R   M   017           : clock data: minutes
                  R   M   018           : clock data: hours,
                  R   M   019           : clock data: year/date
                  R   M   020           : clock data: weekday/month
  Program:  STR       P   001
            STR   R   C   010           ;parameters from R M 010 ->
            FCN           030           ;initialize read address
            STR   R   C   013           ;parameters for read fcn
            FCN           031           ;read only if FCN 30 was successful.
```

- **FCN 32 Read Ibutton serialcode**

This command is used to read serial code from IButton device connected into Autolog PLC's I^2C channel.

Command is executed only if Bit accu equals 1 before instruction.
Function parameter is defined in register accu. Register accu must hold the address of the 1st byte save location

Execution time is 1.4 ms

If iButton is not connected to PLC, command retuns **BA** = 0 and **RA** = 0.
If iButton is connected but CRC does not match, **BA** = 0 and **RA** <> 0.
If iButton is connected and it was successfully read, **BA** = 1 and **RA** = 0.

Example 1 **Read iButton once in a second**

```
Program:  STR       P   001
          STR   R   C   000   ; serial code saved into R M 0 ->
          FCN           032   ; check for ibutton
          IF        T         ; if ibutton was attached to PLC and data
                              ; was succesfully read
          STR   R   M   000   ; make comparison
          EQU   R   M   100   ; device code
          EQ        M   000   ; result to memory 0
          STR   R   M   001   ; 1st data byte
          EQ    R   M   101   ;
          AND       M   000
          EQ        M   000
          Etc.
          CONT
```

Example 2Check for iButton every 1 second

```
Program:  STR   R   C   000   ; parameters from R M 000 ->
          STR       P   001   ; every second…
          IF        T         ; parametrit alkaen R M 10 ->
          FCN           032   ; Read possible Ibutton
          EQ        O   005   ; set output 5 active
          CONT
```

- **FIN F n**

Operation If the bit accumulator is 1, move the contents of the register accumulator into FIFO store n (0 - 7). If there was room in the FIFO, set the bit accumulator to 1, if not, clear the bit accumulator to 0.

Affected

BA	RA	WA	Variable
Yes	No	No	Yes

Example

```
STR     I   0.00   ;At the rising edge of input 0
EQ      M   000
STR     DP  000
STR  R  C   012    ;enter the number 12
FIN     F   000    ;into FIFO 0.
```

- **FOU F n**

Operation If the bit accumulator is 1, move the "oldest" number in FIFO store n (0 - 7) into the register accumulator. If a number was available in the FIFO, set the bit accumulator to 1, if not, clear the bit accumulator to 0.

Affected

BA	RA	WA	Variable
Yes	Yes	No	Yes

Example

```
STR     I   0.00   ;At the rising edge of input 0
EQ      M   000
STR     DP  000
STR  R  C   024    ;enter the number 24
FIN     F   000    ;into FIFO 0,
STR  R  C   034    ;enter the number 34
FIN     F   000    ;into FIFO 0, move the number
FOU     F   000    ;24 from FIFO 0 into the register acc.
```

- **GRT R C d**

Operation If the constant d is greater than the register accumulator, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR  R  O   001    ;When register output 1
GRT  R  C   100    ;is less than 100
EQ      O   0.00   ;output 1 is on.
GRT  R  C   200    ;When RI 1 is less than 200
EQ      O   1.02   ;output 2 is on.
```

- **GRT R M/O n**

Operation If variable n is greater than the register accumulator, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR  R  C   100    ;When register memory 0
GRT  R  M   000    ;is greater than 100,
EQ      O   0.00   ;output 0 is 1.
```

- **GRT W C d**

Operation: If the constant d is greater than the number in the word accumulator, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR   W   M    001         ;When word memory 1
GRT   W   C    04009       ;is less than 4009,
EQ        O    1.02        ;output 002 is on;
GRT   W   C    01050       ;when WM 1 is less than 1050,
EQ        O    1.03        ;output 003 is on.
```

- **GRT W I/M/O n**

Operation: If variable n is greater than the number in the word accumulator, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR   W   C    00500       ;When analog input 002
GRT   W   I    0.02        ;is greater than 500,
EQ        O    1.33        ;output 33 is on.
```

- **IF T**

Operation: If the bit accumulator is 0, subsequent instructions up to the CONT instruction are skipped; if the bit accumulator is 1, the instructions are executed. Further IF instructions can be placed between the IF and CONT instructions. The effect of all IF instructions ceases at the first CONT instruction encountered.

Affected

BA	RA	WA	Variable
No	No	No	No

Example

```
STR       I    0.10
IF        T                ;If input 10 = 0,
STR       I    0.01        ;these instructions will
EQ        O    1.01        ;not be executed
CONT
```

- **IF I/M/O/NI/NM/NO/DP/DN/BM/GM/NB/NG/P n**

Operation: If the variable is 0, subsequent instructions up to the CONT instruction are skipped; if the variable is 1, the instructions are executed. Further IF instructions can be placed between the IF and CONT instructions. The effect of all IF instructions ceases at the first CONT instruction encountered.

Affected

BA	RA	WA	Variable
No	No	No	No

Example

```
IF        DP   077         ;When memory 77 changes
STR       I    0.00        ;from 0 to 1, these instructions
EQ        SM   1.01        ;are executed
CONT
```

- **IF S d**

Operation: If the current sequence register is at step d, subsequent instructions are executed; otherwise execution continues from the next CONT instruction. Further IF instructions can be placed between the IF and CONT instructions. The effect of all IF instructions ceases at the first CONT instruction encountered.

Affected

BA	RA	WA	Variable
No	No	No	No

Example

```
STR    R   S    003      ;When the sequence register
IF         S    098      ;is at step 98,
STR        NO   1.31     ;invert output 71.
EQ         O    1.31
CONT
```

- **INC R M/O n**

Operation: If the bit accumulator is 1, add 1 to register variable n. If the variable was 255, the new value is 0 and the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0. If the bit accumulator was 0, the variable is not incremented. The value of the variable is loaded into the register accumulator.

Affected

BA	RA	WA	Variable
Yes	Yes	No	Yes

Example

```
STR        P    001      ;Increment register memory 0 once per second.
INC    R   M    000
```

- **INC W M/O n**

Operation: If the bit accumulator is 1, add 1 to variable n and load the new value of the variable into the word accumulator. If the variable was 65535, the new value is 0 and the bit accumulator is set to 1; otherwise the bit accumulator is 0. If the bit accumulator was 0, the variable is not incremented but only loaded into the word accumulator.

Affected

BA	RA	WA	Variable
Yes	No	Yes	Yes

Example

```
STR        P    001      ;Increment word memory 3
INC    W   M    003      ;once per second.
```

- **INV**

Operation: Changes the bit accumulator to its one's complement.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR        O    1.00     ;Bit accumulator
AND        NO   1.00     ;is zero after this
INV                      ;and one after this
```

- **LES R C d**

Operation If the constant d is less than the register accumulator, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR  R  O   001    ;When register output 1
LES  R  C   145    ;is between 146 and 154,
EQ      M   000
GRT  R  C   155
AND     M   000    ;output 2 is 1.
EQ      O   1.02
```

- **LES R M/O n**

Operation If variable n is less than the register accumulator, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR  R  O   001    ;When register memory 2
LES  R  M   002    ;is less than register output 1,
EQ      O   1.00   ;output 0 is 1.
```

- **LES W C d**

Operation If the constant d is less than the number in the word accumulator, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR  W  I   0.001  ;When analog input 01
LES  W  C   06000  ;is less than 6000,
EQ      O   1.30   ;output 30 is 1.
```

- **LES W I/M/O n**

Operation If variable n is less than the number in the word accumulator, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR  W  I   2.01   ;When word memory 4
LES  W  M   004    ;is less than analog input 01,
EQ      O   1.30   ;output 30 is 1.
```

- **LOAD C n d**

Operation

If the bit accumulator is 1, counter n is loaded with the initial setting d. After the instruction, the bit accumulator contains the status of the counter (1 = counted out, 0 = running). A setting can be defined for the counter with the programming device. If it is non-zero, it will be used; if it is zero, the value set in the program will be used. When the PLC is switched on, all counters are at zero.

Affected

BA	RA	WA	Variable
Yes	No	No	Yes

Example

```
STR    NI   0.00         ;Delay of 100 seconds
LOAD   C    000   100    ;made with
EQ     O    1.00         ;a counter 0
STR    P    001
DCR    C    001
```

- **LOAD T n d**

Operation

If the bit accumulator is 1, timer n is loaded with the initial setting d. After the instruction, the bit accumulator contains the status of the timer (1 = timed out, 0 = running). A setting can be defined for the timer with the programming device. If it is non-zero, it will be used; if it is zero, the time set in the program will be used. When the PLC is switched on, all timers are at zero.

Affected

BA	RA	WA	Variable
Yes	Yes	No	Yes

Example

```
STR    NI   0.00
LOAD   T    008   100    ;A 100-second delay
EQ     O    1.00         ;from input 0 to output 0
```

- **LOAD R T/C n**

Operation

If the bit accumulator is 1, timer/counter n is loaded with an initial setting equal to the contents of the register accumulator. After the instruction, the bit accumulator contains the status of the timer/counter. A setting made with the programming device or computer has no effect when this instruction is used.

Affected

BA	RA	WA	Variable
Yes	No	No	Yes

Example

```
STR       NI   0.00      ;Delay from input to output,
STR    R  M    000       ;time set via
LOAD   R  T    000       ;register memory
EQ        O    1.00
```

- **LOAD Q n ddd**

Operation: If the bit accumulator is 1, move the contents of the register accumulator into element d (0 - 255) of shift register n (0 - 3).

Affected

BA	RA	WA	Variable
No	No	No	Yes

Example

```
STR   R  C   000        ;Clear the register accumulator.
STR      I   0.00       ;At the rising edge of input 0
EQ       M   000
STR      DP  000
LOAD     Q   000  002   ;clear element 2 of shift register 0.
```

- **MID R C d**

Operation: Subtracts the constant d and the bit accumulator from the register accumulator, assuming that both are two-digit BCD numbers. If the result is less than 0, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0. If the numbers are not BCD numbers, the result is indeterminate.

Affected

BA	RA	WA	Variable
Yes	Yes	No	No

Example

```
IF       DP  000
STR      NM  000        ;M0 = 1 - NM0 = 0
STR   R  M   015        ; Subtract 16 = 0001 0000B =
MID   R  C   016        ;10 BCD (=16 DES) from
EQ    R  M   015        ;the 16-bit number in
STR   R  M   014        ;register memories 14,15.
MID   R  C   000
EQ    R  M   014
CONT
```

- **MID R M/O n**

Operation: Subtracts the variable n and the bit accumulator from the register accumulator, assuming that they are two-digit BCD numbers. If the result is less than 0, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0. If the numbers are not BCD numbers, the result is indeterminate.

Affected

BA	RA	WA	Variable
Yes	Yes	No	No

Example

```
IF       DP  000
INV                     ;Reset bit accumulator
STR   R  M   123
MID   R  O   001
EQ    R  M   123
CONT
```

- **MID W C d**

Operation

Convert the constant d (0 - 9999) to BCD format and subtract it and the bit acc. from the word accumulator, assuming that the word accumulator contains a BCD number. If the result is less than 0, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0. If the number in the word accumulator is not a BCD number, the result is indeterminate.

Affected

BA	RA	WA	Variable
Yes	No	Yes	No

Example

```
IF        DP    000      ;Reset bit accumulator
INV
STR   W   M     030      ;W M 30  = 0000 0011 0110 0011 B (363 BCD)
MID   W   C   00054      ;const.  = 0000 0000 0011 0110 B (36 BCD)
EQ    W   M     030      ;W M 30  = 0000 0011 0010 0111 B (327 BCD)
CONT
```

- **MID W M/O n**

Operation

Subtract the variable n and the bit accumulator from the word accumulator, assuming that they are BCD numbers. If the result is less than 0, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0. If the numbers are not BCD numbers, the result is indeterminate.

Affected

BA	RA	WA	Variable
Yes	No	Yes	No

Example

```
IF        DP    000      ;Reset bit accumulator
INV
STR   W   M     030      ;Subtract contents of word memory 45
MID   W   M     045      ;(as a BCD number) from word memory 30.
EQ    W   M     030
CONT
```

- **MIN R C d**

Operation

Subtracts the constant d and the bit accumulator from the register accumulator. If the result is less than 0, the bit accumulator is set to 1; otherwise the bit acc. is reset to 0.

Affected

BA	RA	WA	Variable
Yes	Yes	No	No

Example

```
STR       C     000      ;Subtract the constant 100
STR   R   M     101      ;from the 16-bit number in
MIN   R   C     100      ;register memories RM 100,101
EQ    R   M     100
STR   R   M     100
MIN   R   C     000      ;If underflow occurs
EQ    R   M     100      ; subtract 1 from R M 12
```

- **MIN R M/O n**

Operation: Subtracts the variable n and the bit accumulator from the register accumulator. If the result is less than 0, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0.

Affected

BA	RA	WA	Variable
Yes	Yes	No	No

Example

```
STR      C   000    ;Reset bit accumulator.
STR   R  M   013    ;R M 12,13 (16-bit number) =
MIN   R  O   001    ;R M 12,13 – R O 1
EQ    R  M   013    ;If underflow occurs
STR   R  M   012    ; subtract 1 from R M 12
MIN   R  C   000
EQ    R  M   012
```

- **MIN W C d**

Operation: Subtract the constant d and the bit accumulator from the word accumulator. If the result is less than 0, the bit accumulator is set to 1;
otherwise the bit accumulator is reset to 0.

Affected

BA	RA	WA	Variable
Yes	No	Yes	No

Example

```
STR      C    000   ;Reset bit accumulator.
STR   W  M     60   ;Subtract the constant 3500
MIN   W  C  03500   ;from word memory 60.
```

- **MIN W I/M/O n**

Operation: Subtract variable n and the bit accumulator from the word accumulator. If the result is less than 0, the bit accumulator is set to 1;
otherwise the bit accumulator is reset to 0.

Affected

BA	RA	WA	Variable
Yes	No	Yes	No

Example

```
STR      C   000    ;Reset bit accumulator.
STR   W  M    60    ;Subtract value of word memory 106
MIN   W  M   106    ;from value of word memory 60.
```

- **MUL R C d**

Operation: Multiply the register acc. by the constant d. The least significant byte remains in the register acc. and the most significant byte is stored in register output RM 244.

Affected

BA	RA	WA	Variable
No	Yes	No	No

Example

```
STR   R  M   000
MUL   R  C   002
```

- **MUL R M/O n**

Operation: Multiply the register accumulator by the variable n. The least significant byte remains in the register accumulator and the most significant byte is stored in register output R O 244.

Affected

BA	RA	WA	Variable
No	No	No	No

Example

```
STR   R  O   001          ;R O 1 * R M 3 = R M 5,4
MUL   R  M   003
EQ    R  M   005
STR   R  O   244
EQ    R  M   004
```

- **MUL W C d**

Operation: Multiply the word accumulator by the constant d. The least significant word of the product remains in the word accumulator and the MSB/LSB eight bit variables are stored in register outputs R O 230, 231.

Affected

BA	RA	WA	Variable
No	No	Yes	No

Example

```
STR   W  M   030
MUL   W  C 00010          ;Contents of word memory 30 * 10
EQ    W  M   010          ; WM 30 * 10 = WM10,11
STR   R  O   230
EQ    R  M   230
STR   R  O   231
EQ    R  M   231
STR   W  M   115
EQ    W  M   011
```

- **MUL W I/M/O n**

Operation: Multiply the word accumulator by the variable n. The least significant word of the product remains in the word accumulator and the MSB/LSB eight bit variables are stored in register outputs R O 230, 231.

Affected

BA	RA	WA	Variable
No	No	Yes	No

Example

```
STR   W  M   001          ;WM 1 * WM 2 = WM 10,11
MUL   W  M   002
EQ    W  M   010
STR   R  O   230
EQ    R  M   230
STR   R  O   231
EQ    R  M   231
STR   W  M   115
EQ    W  M   011
```

- **NEXT S d e**

Operation: If the current sequence register has been at step d for e seconds, it moves on to the next step.

Affected

BA	RA	WA	Variable
No	No	No	No

Example

```
NEXT   S   000   010   ;The sequence register
NEXT   S   001   010   ;moves around steps 0, 1, 2
NEXT   S   002   010   ;at intervals of 10 seconds.
STR    S   003         ;Back to start from step 3
STEP   S   000
```

- **OR I/M/O/NI/NM/NO/DP/DN/BM/GM/NB/NG/P n**

Operation: Sets the bit accumulator equal to the logical sum of its old value and the variable.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR    M    000
OR     NM   000         ;Set bit accumulator to 1.
```

- **OR S d**

Operation: If the bit accumulator is 1 before the instruction or if the current sequence register is at step d, the bit accumulator is set to 1; otherwise it is reset to 0.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR       S   027      ;If the sequence register
OR        S   028      ;is at step 27 or 28,
EQ    R   M   005      ;reset memory 5.
```

- **OR R C d**

Operation: Sets the register acc. equal to the logical sum of its old value and the constant d.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR   R   M   001      ;R M 1   = 00100011 B
OR    R   C   128      ;        = 10000000 B
EQ    R   M   002      ;R M 2   = 10100011 B
```

- **OR R M/O/NM/NO n**

Operation: Sets the register acc. equal to the logical sum of its old value and the variable.

Affected:

BA	RA	WA	Variable
No	Yes	No	No

Example:

```
STR  R  M  001    ;R M 1  = 00001111 B
OR   R  M  002    ;R M 2  = 11110110 B
EQ   R  M  003    ;R M 3  = 11111111 B
```

- **OR W C d**

Operation: Set the word acc. equal to the logical sum of its old value and the constant d.

Affected:

BA	RA	WA	Variable
No	No	Yes	No

Example:

```
STR  W  M    045    ;W M 45 = 0001 1101 1111 0000 B
OR   W  C  09006    ;       = 0010 0011 0010 1110 B
EQ   W  M    021    ;W M 21 = 0011 1111 1111 1110 B
```

- **OR W I/M/O n**

Operation: Set the word accumulator equal to the logical sum of its old value and variable n.

Affected:

BA	RA	WA	Variable
No	No	Yes	No

Example:

```
STR  W  M  033    ;W M 33 = 1011 0110 0110 1100 B
OR   W  M  024    ;W M 24 = 0000 0010 1111 0100 B
EQ   W  M  012    ;W M 12 = 1011 0110 1111 1100 B
```

- **PLD R C d**

Operation: Adds the constant d and the bit accumulator to the register accumulator, assuming that both are BCD numbers. If the result is greater than 99, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0. If the numbers are not BCD numbers, the result is indeterminate. NOTE The constant d which is a BCD number is written in the program as a decimal number.
Example: constant 16 = 0001 0000 (bit format)
1 0 (BCD format)

Affected:

BA	RA	WA	Variable
Yes	Yes	No	No

Example:

```
STR  R  C  017    ;      = 0001 0001B = 11 BCD
PLD  R  C  025    ;      = 0001 1001B = 19 BCD
                  ;Acc.  = 0011 0000B = 30 BCD
```

- **PLD R M/O n**

Operation: Adds the variable n and the bit accumulator to the register accumulator, assuming that each contains a BCD number. If the result is greater than 99, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0. If the addend and augend are not BCD numbers, the result is indeterminate.

Affected:

BA	RA	WA	Variable
Yes	Yes	No	No

Example:

```
STR      I   0.00      ;Register memory 1 = 9,
OR       NI  0.00      ;bit accumulator = 1
STR   R  C   016       ;acc. = 0001 0000B (10 BCD)
PLD   R  M   001       ;+      0000 1001B (9 BCD)
                       ;+              1B (bit acc.)
                       ;acc. = 0010 0000B (20 BCD)
```

- **PLD W C d**

Operation: Convert the constant d (0 - 9999) to BCD format and add it and the bit accumulator to the word accumulator, assuming that both are BCD numbers. If the result is greater than 9999, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0. If the numbers are not BCD numbers, the result is indeterminate.

Affected:

BA	RA	WA	Variable
Yes	No	Yes	No

Example:

```
STR   W  C   00313     ;W A     = 0000 0001 0011 1001 B (139 BCD)
PLD   W  C   00400     ;W const.= 0000 0001 1001 0000 B (190 BCD)
                       ;W A     = 0000 0011 0010 1001 B (329 BCD)
```

- **PLD W M/O n**

Operation: Add variable n and the bit accumulator to the word accumulator, assuming that each contains a BCD number. If the result is greater than 9999, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0. If the addend and augend are not BCD numbers, the result is indeterminate.

Affected:

BA	RA	WA	Variable
Yes	No	Yes	No

Example:

```
STR      I   0.00      ;Word memory 003 = 4900 at the beginning,
OR       NI  0.00      ;set bit accumulator to 1
STR   W  C   02450     ;W A     =0000 1001 1001 0010 B (992 BCD)
PLD   W  M   003       ;+ WM 3  =0001 0011 0010 0100 B (1324 BCD)
                       ;+ BA    =0000 0000 0000 0001 B (1 BCD)
                       ;W A     =0010 0011 0001 0111 B (2317 BCD)
```

- **PLU R C d**

Operation: Adds the constant d and the bit accumulator to the register accumulator. If the result is greater than 255, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0.

	BA	RA	WA	Variable
Affected	Yes	Yes	No	No

Example

```
STR      P   001     ;Add 1 to register memory 123
STR  R   M   123     ;once per second.
PLU  R   C   000
EQ   R   M   123
```

- **PLU R M/O n**

Operation: Adds the variable n and the bit accumulator to the register accumulator. If the result is greater than 255, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0. Using the bit accumulator in this way as a carry bit allows calculation with numbers greater than 255.

	BA	RA	WA	Variable
Affected	Yes	Yes	No	No

Example

```
STR      C   000     ;Reset bit accumulator.
STR  R   M   002     ;RM 22 = RM 12 + RM 2
PLU  R   M   012
EQ   R   M   022
```

- **PLU W C d**

Operation: Add the word constant d and the bit accumulator to the word accumulator. If the result is greater than 65535, the bit accumulator is set to 1; otherwise it is reset to 0.

	BA	RA	WA	Variable
Affected	Yes	No	Yes	No

Example

```
STR      P     002   ;Add 1 to word memory 50
STR  W   M     050   ;once per minute.
PLU  W   C   00000
EQ   W   M     050
```

- **PLU W I/M/O n**

Operation: Add variable n and the bit accumulator to the word accumulator. If the result is greater than 65535, the bit accumulator is set to 1; otherwise the bit accumulator is reset to 0.

	BA	RA	WA	Variable
Affected	Yes	No	Yes	No

Example

```
STR      I    0.00   ;When I000 = 1,
STR  W   M     034   ;WM 003 = WM 34 + WM 25 + 1
PLU  W   M     025
EQ   W   M     003
```

- **PRI TX**

Operation: If the bit accumulator is 1 the text string, whos number is given in the register accumulator, is printed to the active output port. The bit accumulator is 1 after the instruction if the characters were output (there was room in the output buffer).

Affected:

BA	RA	WA	Variable
Yes	No	No	No

Example:

```
STR      P   001     ; Set bit accumulator to 1:
STR  R   C   022     ; Set 22 to register accumulator
PRI     TX           ; Print with indirectly addressing
                     ; the text string number 22.
```

- **PRT C d**

Operation: The number of vacant character positions in the print buffer is returned in the register accumulator, and selects the output port for printing (0=I^{2}C, 1= Ser1, 2= Ser2, 3=Ser3).
If there is room for more than 80 characters, the bit accumulator is set to 1; if not, the bit accumulator is reset to 0.

Affected:

BA	RA	WA	Variable
Yes	Yes	No	No

Example:

```
PRT      C    002    ;Repeatedly prints the text:
PRT      T  Still    ;Still here
PRT      T  here@
```

In WinAlpro yu can write this:

```
PRT      C    000
PRT(“Still here”)    ;
```

- **PRT T ccccc**

Operation: If the bit accumulator is 1, the five characters given in the instruction are output.
If it is not desired to output as many as five characters, the output can be terminated with the @ character, in which case a carrier return and line feed will not be added at the end, or with the # character, in which case a carrier return and line feed will be added. Any characters that can be produced with the keyboard are valid as parameters except that CTRL-B (12 hex) is not acceptable as the second character. The bit accumulator is 1 after the instruction if the characters were output (there was room in the output buffer).

Affected:

BA	RA	WA	Variable
Yes	No	No	No

Example:

```
STR      I   0.00
EQ       M    000
STR     DP    000
PRT      T  input
PRT      T   0 is
PRT      T    on#
```

or

```
STR     DP    000
PRT (“Input 0 is on”)
```

- **PRT TX n/name**

Operation: If the bit accumulator is 1 the text string number n or name is printed to the active output port. The text strings can be edited with ALPro / Symbol/Edit Text selection The bit accumulator is 1 after the instruction if the characters were output (there was room in the output buffer).

Affected:

BA	RA	WA	Variable
Yes	No	No	No

Example:

```
STR     P    001        ; Set bit accumulator to 1:
PRT     TX  STR003      ; Print the text string name STR003 .
                        ; to active output port
```

- **PRT R C d**

Operation: If the bit accumulator is 1, the constant d given as the instruction parameter is output with two digits. The constant d is assumed to be between 00 and 99. If the constant d is greater than 99, indeterminate characters will be output. If there was room for the characters in the print buffer, the bit accumulator is 1 after the instruction.
The number (0 - 99) must be coverted to BCD format before printing.

Affected:

BA	RA	WA	Variable
Yes	No	No	No

Example:

```
STR   R  C   020        ; century ( R  C   d)
BCD   R  T              ; convert to BCD format
EQ    R  M   000        ; save to auxuliary memory
STR   R  M   255        ; year
BCD   R  T              ; convert to BCD format
EQ    R  M   001        ; save to auxuliary memory
PRT      C   000        ; Put output to display unit
STR      DP  001        ; Print once per second
PRT   (<ESC>,"Y",<00>,<00>)  ; place the cursor up left
PRT   R  M   000        ; century: 20
PRT   R  M   001        ; & tens & ones
```

- **PRT R T**

Operation: If the bit accumulator is 1, the character in the register accumulator is output. If there was room for the character in the output buffer, the bit accumulator is 1 after the instruction.

Affected:

BA	RA	WA	Variable
Yes	No	No	No

Example:

```
PRT      C   000        ; Put output to display unit
STR   R  M   232        ;If a character has been
LES   R  C   000        ;received from the serial line,
PRT   R  T              ;echo it and
EQ    R  RM  232        ;clear the character buffer.
```

- **PRT R M/O n**

Operation: If the bit accumulator is 1, the value of the variable n is output with two digits, assuming it to be a BCD number. If the bit accumulator is 0, it will be 0 after the instruction. If the variable is not a BCD number, indeterminate characters will be output.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR      P   001      ;Outputs the content of register
PRT   R  M   001      ;memory 1 once per second.
```

- **READ F n**

Operation: Read the current number of elements in FIFO store n (0-7) into the register accumulator. If there are elements in the FIFO, set the bit accumulator to 1, if not, clear the bit accumulator to 0.

Affected

BA	RA	WA	Variable
Yes	Yes	No	No

Example

```
STR      I   0.00     ;At the rising edge of input 0
EQ       M   000
STR      DP  000
STR   R  C   012      ;enter the number 12
FIN      F   000      ;into FIFO 0,
STR   R  C   004      ;the number 4
FIN      F   000      ;into FIFO 0,
STR   R  C   024      ;the number 24
FIN      F   000      ;into FIFO 0,
READ     F   000      ;number 3 into register acc.
```

- **READ R T/C n**

Operation: The register accumulator is loaded with the remaining count in timer/counter n. If the timer/counter has counted out, the register accumulator will be 0.

Affected

BA	RA	WA	Variable
No	Yes	No	No

Example

```
STR      NI  0.00
LOAD     T   020  100
EQ       O   1.00     ;Register output 0
READ  R  T   020      ;will show the
EQ    R  M   000      ;remaining time
```

- **READ S n**

Operation: Selects sequence register n as the current sequence register.

Affected

BA	RA	WA	Variable
No	No	No	No

Example

```
READ     S   002      ;Subsequent sequence register
                      ;instructions apply to sequence register 2.
```

- **READ T/C n**

Operation: The status of the timer/counter (1=counted out, 0=running) is loaded into the bit accumulator.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
READ    T    010         ;Output 0 changes state
EQ      M    000         ;for a moment
LOAD    T    010   005   ;at intervals of
STR    DP    000         ;5 seconds.
EQ      O    1.00
```

- **RES F n**

Fifo operates in data areas F0 WM1024-1151, F1 WM1152-1279,...,F7 WM1920-2047.

Operation: If the bit accumulator is 1, clear FIFO store n (0 - 7) (delete contents).

Affected

BA	RA	WA	Variable
No	No	No	Yes

Example

```
STR     I    0.00        ;At the rising edge of input 0
EQ      M    000
STR    DP    000
RES     F    000         ;clear FIFO 0.
```

- **RES Q n**

Operation: If the bit accumulator is 1, reset all elements of shift register n (0 - 3) to zero.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR     I    0.00        ;When input 0 is 1, reset all
RES     Q    002         ;elements of shift register 2.
```

- **RET**

Operation: The subroutine ends here, execution is then returned to main program.

Affected

BA	RA	WA	Variable
Yes	Yes	Yes	Yes

The states of the accumulators remain unchanged, so main program can use the accumulator states after executing the subroutine.

- **SBR n**

Operation: This is the starting instrucion of a subroutine. **See page 12-44.**

- **SHL Q n ddd**

Operation: If the bit accumulator is 1, shift the elements of shift register n (0 - 3) one step to the left. The number in the register accumulator is shifted into the vacated position (right end) of the shift register. The element shifted out of the shift register goes into the register accumulator. d (0 - 255) is the length of the shift register - 1.

Affected:

BA	RA	WA	Variable
No	Yes	No	Yes

Example:

```
STR        NI   0.00          ;The falling edge of input 0
EQ         M    000
STR        DP   000
READ  R    C    003
SHL        Q    000   250     ;moves a 560-step conveyor
SHL        Q    001   250     ;one step to the left.
SHL        Q    002   060
```

- **SHR Q n ddd**

SHR instruction uses word outputs (Q0 uses WO 512- 639, Q1 uses WO 640- 767 etc.) and these word outputs can be used for other purposes only if SHR instruction is not used.

Operation: If the bit accumulator is 1, shift the byte-sized elements of shift register n (0 - 3) one step to the right. The number in the register accumulator is shifted into the vacated position (left end) of the shift register. The element shifted out of the shift register goes into the register accumulator. d (0 - 255) is the length of the shift register - 1.

Affected:

BA	RA	WA	Variable
No	Yes	No	Yes

Example:

```
STR        I    0.00          ;The rising edge of input 0
EQ         M    000
STR        DP   000
STR   R    M    030
SHR        Q    000   200     ;moves a 300-step conveyor
SHR        Q    001   100     ;one step to the right.
```

- **STEP R T**

Operation: If the bit accumulator is 1, move the current sequence register to the step indicated by the contents of the register accumulator. If the bit accumulator is 0, do nothing.

Affected:

BA	RA	WA	Variable
No	No	No	No

Example:

```
STR        C    001           ;The sequence register
STR   R    M    003           ;moves to the step indicated
STEP  R    T                  ;by register memory 3.
```

- **STEP S d**

Operation: If the bit accumulator is 1, the current sequence register moves to step d. If the bit accumulator is 0, do nothing.

Affected

BA	RA	WA	Variable
No	No	No	No

Example

```
STR      S   139     ;If at step 139,
STEP     S   000     ;move to step 0
```

- **STEP T**

Operation: If the bit accumulator is 1, the current sequence register moves on to the next step. If the previous step is 255, the new step is 0.

Affected

BA	RA	WA	Variable
No	No	No	No

Example

```
STR   R  S   000     ;Select sequence register 0
STR      P   000     ;(valid until the next selection).
STEP     T           ;Sequence register moves to
                     ;next step 10 times per second.
```

- **STI I/M/O/BM/GM n**

Operation: Reads the state of binary input/output/memory to bit accumulator. Register accumulator is pointing to the bit variable to be read.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR   R  C   012     ;The state of input I 12 is the bit accumulators
STI      I           ;state (read address is 12 decimal).
```

- **STI R M/O n**

Operation: Loads the register accumulator with the variable whose address is the value of the variable given as the instruction parameter.

Affected

BA	RA	WA	Variable
No	Yes	No	No

Example

```
STR   R  C   010     ;= 00 001 010B = 10 dec
EQ    R  M   020     ;Read contents of RM 10
STI   R  M   020     ;into the register accumulator.
```

- **STI W M/O/I n**

Operation: Load the word accumulator with the variable whose address is the value of the variable given as the instruction parameter. For accessible addresses see the tables of the variables.

Affected

BA	RA	WA	Variable
No	No	Yes	No

Example

```
STR   W  C   010     ;= 0 000 000 000 001 010 B = 10
EQ    W  M   020     ;Read contents of WM 10
STI   W  M   020     ;into the word accumulator.
```

- **STOP**

Operation: Last instruction of the program; the first program line will be executed next. Terminates the range of the IF instruction.

	BA	RA	WA	Variable
Affected	No	No	No	No

- **STP**

Operation: The PLC's main program ends with this command and all subroutine programs must be written after the STP command.

- **STR I/M/O/NI/NM/NO/BM/GM/NB/NG/P n**

Operation: Reads the state of variable n into the bit accumulator. The variable is not affected.

	BA	RA	WA	Variable
Affected	Yes	No	No	No

Example:

```
STR    I   0.00    ;Read state of input 0 into acc.
```

- **STR DP n**

Operation: If auxiliary memory n is 1 and was 0 at the start of the program cycle, the bit accumulator is set to 1. If the auxiliary memory is 0, or was already 1 at the start of the program cycle, the bit accumulator is reset to 0.

	BA	RA	WA	Variable
Affected	Yes	No	No	No

Example:

```
STR    I   0.00    ;Read state of input 0 into memory
EQ     M   010
STR    DP  010     ;Rising edge
XOR    O   1.05    ;inverts output 5.
EQ     O   1.05
```

- **STR DN n**

Operation: If auxiliary memory n is 0 and was 1 at the start of the program cycle, the bit accumulator is set to 1. If the auxiliary memory is 1, or was already 0 at the start of the program cycle, the bit accumulator is reset to 0.

	BA	RA	WA	Variable
Affected	Yes	No	No	No

Example:

```
STR    I   0.01    ;At falling edge of input 1
EQ     M   001     ;decrement counter 3 by 1.
STR    DN  001
DCR    C   003
```

- **STR S/(NS) d**

Operation: If the current sequence register is/(not) at step d, the bit accumulator is set to 1; otherwise it is reset to 0.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR     S   024     ;When sequence register is at
EQ      O   125     ;step 24, O 125 is on.
```

- **STR R M/O/NM/NO n**

Operation: Reads the value of a register variable into the register accumulator.

Affected

BA	RA	WA	Variable
No	Yes	No	No

Example

```
STR  R  M   210     ;Read contents of register memory 210 into
                    register accumulator.
```

- **STR R C d**

Operation: Reads the constant d (0 - 255) into the register accumulator.

Affected

BA	RA	WA	Variable
No	Yes	No	No

Example

```
STR  R  C   019     ;Store 19 (decimal) in register accumulator.
```

- **STR R S n**

Operation: Reads the number of the current step of sequence register n (0 - 31) into the register accumulator. Register n is made the current sequence register, i.e. subsequent sequence register instructions will affect it.

Affected

BA	RA	WA	Variable
No	Yes	No	No

Example

```
STR  R  S   002     ;When sequence register 2 is at
STR     S   029     ;step 29, output 5 is on.
EQ      O   1.05
```

- **STR C d**

Operation: Reads the state of variable n into the bit accumulator. The accumulator is reset to 0, if d = 0; for other values of d, it is set to 1.

Affected

BA	RA	WA	Variable
Yes	No	No	No

Example

```
STR     C   000     ;Initialize bit accumulator.
STR  R  M   010
PLU  R  M   011
EQ   R  M   020
```

- **STR Q n**

Operation: Read the element whose position is given by the register accumulator, of shift register n (0 - 3), into the register accumulator.

	BA	RA	WA	Variable
Affected	No	Yes	No	No

Example

```
STR  R  C  005    ;Move the value of element 5
STR     Q  000    ;of shift register 0 into the register accumulator.
```

- **STR W I/M/O n**

Operation: Read the value of variable n into the word accumulator.

	BA	RA	WA	Variable
Affected	No	No	Yes	No

Example

```
STR  W  I  2.01   ;Read value of input 1 (0 - 4095)into word acc.
```

- **STR W C d**

Operation: Read the constant d (0 - 65535) into the word accumulator.

	BA	RA	WA	Variable
Affected	No	No	Yes	No

Example

```
STR  W  C  00455  ;Initialize word accumulator to 455.
```

- **XOR I/M/O/NI/NM/NO/DP/DN/BM/GM/NB/NG/P n**

Operation: Sets the bit accumulator equal to the exclusive-or function of its old value and the variable.

	BA	RA	WA	Variable
Affected	Yes	No	No	No

Example

```
STR     P  001
XOR     O  1.00   ;Output 0 is inverted
EQ      O  1.00   ;once per second.
```

- **XOR S d**

Operation: If the current sequence register is at step d and the bit accumulator is 0, or if the sequence register is not at step d and the bit accumulator is 1, the bit accumulator is set to 1; otherwise it is reset to 0.

	BA	RA	WA	Variable
Affected	Yes	No	No	No

Example

```
STR  R  S  000    ;If only one of sequence registers
STR     S  010    ;0 and 1 is at step 10,
STR  R  S  001    ;memory 10 is set to 10.
XOR     S  010
EQ      M  010
```

- **XOR R M/O/NM/NO n**

Operation: Sets the register accumulator equal to the exclusive-or function of its old value and the variable.

Affected:

BA	RA	WA	Variable
No	Yes	No	No

Example:

```
STR  R  M  000    ;R M 0 = 01111011 B
XOR  R  M  001    ;R M 1 = 10000011 B
EQ   R  M  002    ;R M 2 = 11111000 B
```

- **XOR R C d**

Operation: Sets the register accumulator equal to the exclusive-or function of its old value and the constant d.

Affected:

BA	RA	WA	Variable
No	Yes	No	No

Example:

```
STR  R  M  000    ;R M 0  = 01010101 B
XOR  R  C  255    ;       = 11111111 B
EQ   R  M  001    ;R M 1  = 10101010 B
```

- **XOR W I/M/O n**

Operation: Set the word accumulator equal to the exclusive-or function of its old value and variable n.

Affected:

BA	RA	WA	Variable
No	No	Yes	No

Example:

```
STR  W  M  033    ;W M 33 = 1011 0110 0110 1100 B
XOR  W  M  024    ;W M 24 = 0000 0010 1111 0100 B
EQ   W  M  012    ;W M 12 = 1011 0100 1001 1000 B
```

- **XOR W C d**

Operation: Set the word accumulator equal to the exclusive-or function of its old value and the constant d.

Affected:

BA	RA	WA	Variable
No	No	Yes	No

Example:

```
STR  W  M    045    ;W M 45 = 0001 1101 1111 0000 B
XOR  W  C  09006    ;       = 0010 0011 0010 1110 B
EQ   W  M    021    ;W M 21 = 0011 1110 1101 1110 B
```

Subroutines

The PLC's main program ends with an STP instruction. It is possible to write subroutine programs which start with the SBR n command (n is subroutine number), and ends with the RET command. The maximum number of subroutines is 32 (numbers 0 - 31). The STOP command must be appear after the subroutines, this ends the PLC program. The PLC system program ignores all SBR and RET instructions which appear before STP instruction. If the RET instruction is missing from a subroutine the PLC executes the next subroutine(s) until the a RET command is found. If no RET command is found, program executes until the STOP command is reached, and the PLC system program does not return to main program. The PLC now performs I/O updates and the main program starts again from line 1. Therefore the main program is executed only up to the first CSR instruction, the rest of the main program is not executed at all. If there is a CSR call for a subroutine that does not exist, the PLC interpretes the CSR instruction as a NOP instructionand continues normally.

- **STP**

Operation: The PLC's main program ends with this command and all subroutine programs must be written after the STP command

- **CSR n**

Operation: The PLC executes the subprogram n if the bit accumulator is true
Affected: Bit accumulator Register accumulator Word accumulator Variable
The states of the accumulators remain unchanged, so subroutine can use the accumulator states.

- **SBR n**

Operation: This is the starting instruction of a subroutine

- **RET**

Operation: The subroutine ends here, execution is then returned to main program.
Affected: Bit accumulator Register accumulator Word accumulator Variable
The states of the accumulators remain unchanged, so main program can use the accumulator states after executing the subroutine.

Example: Scaling subroutine for analogue variables. The scaling parameters are transferred in main program with the EQI instruction. The result after the subroutine call is retrieved with the STI instruction from word accumulator or from WM 101.

```
STR       C    001      ; set bit accumulator to 1
CSR            001      ; call subroutine 1
STP                     ; main program end
...
SBR             01      ; start of subroutine 1
EQ    W   M    102
STR   R   M    100      ; scaling parameters start address
STR       C    001      ; set bit accumulator to 1
STR   W   M    100      ; variable to be scaled
FCN            010      ; call scaling function
EQ    W   M    101      ; save the result to WM 101
RET
...
STOP
```

12. TABLES

12.1 ASCII-codes

character ASCII code	Hex code	decimal code	character ASCII code	Hex code	decimal code	character ASCII code	Hex code	decimal code
NUL	00	00	+	2B	43	V	56	86
SOH	01	01	,	2C	44	W	57	87
STX	02	02	-	2D	45	X	58	88
ETX	03	03	.	2E	46	Y	59	89
EOT	04	04	/	2F	47	Z	5A	90
ENQ	05	05	0	30	48	[(Ä)	5B	91
ACK	06	06	1	31	49	\ (Ö)	5C	92
BEL	07	07	2	32	50	](Å)	5D	93
BS	08	08	3	33	51	^	5E	94
HT	09	09	4	34	52	_	5F	95
LF	0A	10	5	35	53	`	60	96
VT	0B	11	6	36	54	a	61	97
FF	0C	12	7	37	55	b	62	98
CR	0D	13	8	38	56	c	63	99
SO	0E	14	9	39	57	d	64	100
SI	0F	15	:	3A	58	e	65	101
DLE	10	16	;	3B	59	f	66	102
DC1	11	17	<	3C	60	g	67	103
DC2	12	18	=	3D	61	h	68	104
DC3	13	19	>	3E	62	i	69	105
DC4	14	20	?	3F	63	j	6A	106
NAK	15	21	@	40	64	k	6B	107
SYN	16	22	A	41	65	l	6C	108
ETB	17	23	B	42	66	m	6D	109
CAN	18	24	C	43	67	n	6E	110
EM	19	25	D	44	68	o	6F	111
RSUB	1A	26	E	45	69	p	70	112
ESC	1B	27	F	46	70	q	71	113
FS	1C	28	G	47	71	r	72	114
GS	1D	29	H	48	72	s	73	115
RS	1E	30	I	49	73	t	74	116
US	1F	31	J	4A	74	u	75	117
SP	20	32	K	4B	75	v	76	118
!	21	33	L	4C	76	w	77	119
"	22	34	M	4D	77	x	78	120
#	23	35	N	4E	78	y	79	121
$	24	36	O	4F	79	z	7A	122
%	25	37	P	50	80	{ (ä)	7B	123
&	26	38	Q	51	81	| (ö)	7C	124
‘	27	39	R	52	82	} (å)	7D	125
(	28	40	S	53	83	~	7E	126
)	29	41	T	54	84	DEL	7F	127
*	2A	42	U	55	85			

12.2 Decimal/Octal conversions

dec. code	octal code	dec. code	octal code	dec. code	octal code	dec. code	octal code	dec. code	octal code	dec. code	octal code
00	00	46	56	92	134	138	212	184	270	230	346
01	01	47	57	93	135	139	213	185	271	231	347
02	02	48	60	94	136	140	214	186	272	232	350
03	03	49	61	95	137	141	215	187	273	233	351
04	04	50	62	96	140	142	216	188	274	234	352
05	05	51	63	97	141	143	217	189	275	235	353
06	06	52	64	98	142	144	220	190	276	236	354
07	07	53	65	99	143	145	221	191	277	237	355
08	10	54	66	100	144	146	222	192	300	238	356
09	11	55	67	101	145	147	223	193	301	239	357
10	12	56	70	102	146	148	224	194	302	240	360
11	13	57	71	103	147	149	225	195	303	241	361
12	14	58	72	104	150	150	226	196	304	242	362
13	15	59	73	105	151	151	227	197	305	243	363
14	16	60	74	106	152	152	230	198	306	244	364
15	17	61	75	107	153	153	231	199	307	245	365
16	20	62	76	108	154	154	232	200	310	246	366
17	21	63	77	109	155	155	233	201	311	247	367
18	22	64	100	110	156	156	234	202	312	248	370
19	23	65	101	111	157	157	235	203	313	249	371
20	24	66	102	112	160	158	236	204	314	250	372
21	25	67	103	113	161	159	237	205	315	251	373
22	26	68	104	114	162	160	240	206	316	252	374
23	27	69	105	115	163	161	241	207	317	253	375
24	30	70	106	116	164	162	242	208	320	254	376
25	31	71	107	117	165	163	243	209	321	255	377
26	32	72	110	118	166	164	244	210	322		
27	33	73	111	119	167	165	245	211	323		
28	34	74	112	120	170	166	246	212	324		
29	35	75	113	121	171	167	247	213	325		
30	36	76	114	122	172	168	250	214	326		
31	37	77	115	123	173	169	251	215	327		
32	40	78	116	124	174	170	252	216	330		
33	41	79	117	125	175	171	253	217	331		
34	42	80	120	126	176	172	254	218	332		
35	43	81	121	127	177	173	255	219	333		
36	44	82	122	128	200	174	256	220	334		
37	45	83	123	129	201	175	257	221	335		
38	46	84	124	130	202	176	260	222	336		
39	47	85	125	131	203	177	261	223	337		
40	50	86	126	132	204	178	262	224	340		
41	51	87	127	133	205	179	263	225	341		
42	52	88	130	134	206	180	264	226	342		
43	53	89	131	135	207	181	265	227	343		
44	54	90	132	136	210	182	266	228	344		
45	55	91	133	137	211	183	267	229	345		

13. AL2000 PRODUCT LIST

13.1 Mounting Racks

Code	Model No.	Description
902158	MR3 mini	Mounting Rack for 3 I/O modules (no PSU)
902159	MR3 mini	Mounting Rack for 3 I/O modules (no PSU), with covers
902151	MR3	Mounting Rack for 3 I/O-modules (115 VAC)
902152	MR3	Mounting Rack for 3 I/O-modules (220 VAC)
902153	MR3	Mounting Rack for 3 I/O-modules (24 VDC)
902150	MR5	Mounting Rack for 5 I/O-modules
902154	MR11	Mounting Rack for 11 I/O-modules
902156	MR16	Mounting Rack 16 I/O-modules
902100	FC50	Flat cable 0,5 m
902110	FC100	Flat cable 1,0 m

13.2 Central Processing Units

Code	Model No.	Description
902200	CPU2000S	Central Processing Unit
902205	CPU2000P	Central Processing Unit
902204	CPU2000L	Central Processing Unit
902207		PC104 adapter (with CPU2000L, reserves 2 I/O Board Places)
902132		Mother Board for CPU2000L

13.3 Serial Communication and Router Boards

Code	Model No.	Description
902195	AL2000SER	Serial Communication Unit for use with AL2000
902197	AL-Router	Modbus Router

13.4 Input Modules

Code	Model No.	Description
902220	DIC32DC	Digital Input Module, 32 DI, 24 V DC
902224	DIC32AC	Digital Input Module, 32 DI, 24 V AC/DC
902228	DIF16	Digital Input Module, 16 Error-checking DI, 24 VDC

13.5 Output Modules

Code	Model No.	Description
902240	DOC32EP	Digital Output Module, 32 DO, 20-40 V DC, 1 A
902244	DOC32FP	Digital Output Module, 32 DO, 20-40 V DC, 0,5 A
902248	DOF16	Digital Output Module, 16 Error-checking DO, 24 VDC, 1 A
* 902260	ROC16K	Relay Output Module, 20-280 V AC, 2 A RO
902264	OOC16	Solid State Relay Output Module, 20-280 V AC, 0,6 A

* Delivery Time and Technical Data by Request

13.6 Input/Output Modules

Code	Model No.	Description
902266	DIO32	Digital Input/Output Module, 16 DI 24 V DC, 16 DO 20-40 V DC, 1 A

13.7 Analog Modules

Code	Model No.	Description
902273	AIC8	Analog Input Module, 8-16 AI
902274	TIC8	Thermocouple Input Module, 8 TI
902270	AIO74	Analog Input/Output Module, 7 - 14 AI, 4 AO
902272	MIC16	Analog Input Module, 16 AI galvanic isolated
* 902277	MOC16	Analog Output Module, 16 AO 4 - 20 mA
* 902310	FIC16	Frequency Input Module, maximum 10 kHz, 16 Channels

13.8 Power Units

Code	Model No.	Description
902209	POWDC-100W	Power Supply Module, Supply Voltage 24 VDC
902210	POWDC-25W	Power Supply Module, Supply Voltage 24 VDC
902211	PCON24	Power Connection Module (with POWDC-module)
902214	POWAC115	Power Supply Module, Supply Voltage 90-132 VAC
902215	PCON115	Power Connection Module (with POWAC115-module)
902212	POWAC230	Power Supply Module, Supply Voltage 180-265 VAC
902213	PCON230	Power Connection Module (with POWAC230 module)
902219	PCON24/100	Power connection module (with POWDC-100W module)

13.9 Special Units

Code	Model No.	Description
902400	DSIM 32	Digital Inputs/Output Simulator

13.10 Accessories

Code	Model No.	Description
900860	AL1093F	Clock and Calendar/Display/Keypad 2x16 Characters, 16 Function Keys
901017	AL1093D	Clock and Calendar/Display/Keypad (With Surface Mounting Plate) 2x16 Characters, 16 Function Keys
901017	AL1093DC	Clock and Calendar/Display/Keypad (for Customers front Plate) 2x16 Characters, 16 Function Keys
901023	AL1094	Display/Keypad
901019	AL1094R	Display/Keypad
900870	AL1094F	Clock and Calendar/Display/Keypad (for Customers front Plate)
900850	AL1095A	Clock and Calendar/Display/Keypad, 16 keys
900855	AL1095B	Clock and Calendar/Display/Keypad, 4 keys
902172	AL1096/S	Graphic Display (240 x 128), 5 function keys
902174	AL1096/T	Graphic Display (240 x 128), touch screen (10 x 8)
902178	AL1096PS	STN Graphic Display/Touch Panel (320 x 240), 40x30 Switches
902180	AL1096PE	STN Graphic Display/Touch Panel (320 x 240), 40x30 Switches, Centronic printer port

* Delivery Time and Technical Data by Request

13.11 Converters for Serial Communication

Code	Model No.	Description
901225	AL6442	CNV-1 full duplex Serial Interface Conversion Unit (RS232 - 422/485)
901462	AL 6405	Power Supply (CNV-1)
903190		CNV-2 Half-duplex Signal Converter RS232 - RS485

13.12 Power Supplies

Code	Model No.	Description
901383	AL9624/3.5	Power Supply Unit 24VDC/3,5A
901380	AL9624/8	Power Supply Unit 24VDC/8A
902218	AL-UPS	AL UPS Module for Use with Accumulator
901374	AL9624/2/1.5	Power Supply Unit 24VDC/2A, 20VAC/1,5A

13.13 Programming Cables

Code	Model No.	Description
901172	ALC2	Programming Cable PC - AL2000, Length 2 m
901173	ALC5	Programming Cable PC - AL2000, Length 5 m
901175		Cable AL1096 - PC, Length 5 m
901176		Cable AL1096 - Autolog, Length 5 m
901510	AL9042	Cable AutoLog(D9P) - CNV1, 2.5m
901177		Cable Modem(D9P) - AutoLog(D9P), Length 2.5m
901178		Cable PC(D9S) - Modem(D9P), Length 2.5m
901179		Cable AutoLog(D9P) - Radiomodem(15P), Length 2m
901180		Cable AutoLog(D9P) - Radiomodem(15P), Length 8m
901181		Cable PC/FCS(D9S) - CNV1, Length 8m
901187		Cable GSM modem - AL2000S
901193		Cable PC - CNV2, Length 2.5 m
901194		Cable CNV2 - AL (RJ45), Length 2.5 m

13.14 Programming Software

Code	Model No.	Description
941006	ALPro3.x	ALPro3.x Programming Software DOS version
941010		Update from AL/AX,ALGT,ALEDIT,ALPro(dos) to ALProwin 1.xx
941011		Update from ALPro1(2).x dos) to ALPro3.x(dos)
941020		ALProwin 1.xx

13.15 Other Programs

Code	Model No.	Description
941060		MODBUS RTU Development Kit for PC
906611		Modbus Analyser
906603		Modbus Test Program for PC, Dos

13.16 AutoLog FCS Control Software

Code	Model No.	Description
941024		Autolog 2000 FCS Development Licence
941025		Autolog 2000 FCS Run Time Interface Licence
941027		Autolog 2000 FCS Run Time Licence for AL2000 CPU
941026		AutoLog 2000 FCS Communication Programs Modbus RTU TCP/IP Siemens Allen Bradley etc.

14. CABLES AND COMMUNICATION TIME SHEET

14.1 Cables

Cables required for connecting a PC to the AL2000 and for SER1 and SER2 interfaces:

Pin	Signal	Pin	Signal
1	+4V max. 20 mA / DCD (selectable)	6	DSR
2	RXD data to the AL2000	7	RTS
3	TXD data from the AL2000	8	CTS
4	DTR +12V from AL2000	9	RI
5	GND		

IBM PC/XT Cannon DB25S	AL 2000 Cannon DE9P
TxD 2	2 RxD
RxD 3	3 TxD
CTS 5	4 DTR
GND 7	5 GND

IBM AT/ TOSHIBA 1000 Cannon DE9S	AL 2000 Cannon DE9P
TxD 3	2 RxD
RxD 2	3 TxD
CTS 8	4 DTR
GND 5	5 GND

AL1091Terminal Cannon DB25S	AL 2000 Cannon DE9P
TD 2	2 RxD
RD 3	3 TxD
GND 7	5 GND
CTS 5	4 DTR
RTS 3	8 CTS
+24V 12	
or+5V 9	
GND 8	

NOTE! Supply voltage shall be connected to either of these two not to both

MODEM Cannon DB25P	AL 2000 Cannon DE9P	
RxD 3	2 RxD	
TxD 2	3 TxD	
GND 7	5 GND	
DTR 20	7 RTS	This way when used on dialled phone lines.
DCD 8	8 CTS	
OR		
RTS 4	7 RTS	This way when used on fixed lines.
CTS 5	8 CTS	

14.2 Data Communication Time Calculation Sheet

Communication speeds (cmmt) 1200 bit/s = 9,2 ms, 9600 bit/s = 1,2 ms,
19200 bit/s = 0,57 ms, 28800 bit/s = 0,38 ms

number of bytes
message 1: (__________+ 8) x cmmt + 1 ms + (slave loop time) =__________________ms
message 2: (__________+ 8) x cmmt + 1 ms + (slave loop time) =__________________ms
message 3: (__________+ 8) x cmmt + 1 ms + (slave loop time) =__________________ms
message 4: (__________+ 8) x cmmt + 1 ms + (slave loop time) =__________________ms
message 5: (__________+ 8) x cmmt + 1 ms + (slave loop time) =__________________ms
message 6: (__________+ 8) x cmmt + 1 ms + (slave loop time) =__________________ms
message 7: (__________+ 8) x cmmt + 1 ms + (slave loop time) =__________________ms
message 8: (__________+ 8) x cmmt + 1 ms + (slave loop time) =__________________ms
message 9: (__________+ 8) x cmmt + 1 ms + (slave loop time) =__________________ms
message 10: (_________+ 8) x cmmt + 1 ms + (slave loop time) =__________________ms
message 11: (_________+ 8) x cmmt + 1 ms + (slave loop time) =__________________ms
message 12: (_________+ 8) x cmmt + 1 ms + (slave loop time) =__________________ms
message 13: (_________+ 8) x cmmt + 1 ms + (slave loop time) =__________________ms
message 14: (_________+ 8) x cmmt + 1 ms + (slave loop time) =__________________ms
message 15: (_________+ 8) x cmmt + 1 ms + (slave loop time) =__________________ms
message 16: (_________+ 8) x cmmt + 1 ms + (slave loop time) =__________________ms
message 17: (_________+ 8) x cmmt + 1 ms + (slave loop time) =__________________ms
message 18: (_________+ 8) x cmmt + 1 ms + (slave loop time) =__________________ms
total) =__________________ms
+ output set message 18 ms + 1 ms + (slave loop time) =__________________ms

total =__________________ms

The calculated time is THE GREATEST POSSIBLE TIME that the modbus message loop can take. In practice this time is shorter, because the slave does not always run its entire whole program loop before the slave can examine the received message.

The Total Responce Time

- When RIO channel is used
the number of Modbus master messages x RIO master program loop time

____x _________________ =_____________ms

- When Modbus channel/protocol is used
the number of Modbus master messages x (slave program loop time + responce message time)

____x _________________ =_____________ms

Additionally, in both cases the PC control software loop time must be included in the time calculations (how soon the control software is able to send the new message)

number of messages	9600	19200	28800	RIO 178500

14.3 The Commands of AutoLog

English	Finnish
AND	JA
BCD	BCD
BIN	BIN
BIT	BIT
BYT	BYT
CLO	CLO
CONT	JTK
CSR	CSR
DCD	DCD
DCR	VH
DEC	DEC
DIV	JAK
END	END
EQ	ON
EQI	ONI
EQU	EQU
ERROR	VIRHE
FCN	FCN
FIN	FIN
FOU	FOU
GRT	GRT
IF	JOS
INC	INC
INV	INV
LES	LES
LOAD	AS
MID	MID
MIN	MIN
MUL	KER
NEXT	SEUR
OR	TAI
PLD	PLD
PLU	PLU
PRI	PRI
PRT	PRT
READ	LUE
RES	RES
RET	RET
SBR	SBR
SHL	SHL
SHR	SHR
STEP	STEP
STI	STI
STOP	STOP
STP	STP
STR	STR
XOR	ETAI

Variables

English	Finnish
I	T
NI	ET
O	L
NO	EL
SO	L
RO	RL
M	M
NM	EM
SM	SM
RM	RM
DP	DP
DN	DN
BM	BM
NB	EB
SB	SB
RB	RB
GM	GM
NG	EG
SG	SG
RG	RG
P	P
T	A
TX	TX
C	N
S	S
F	F
Q	J

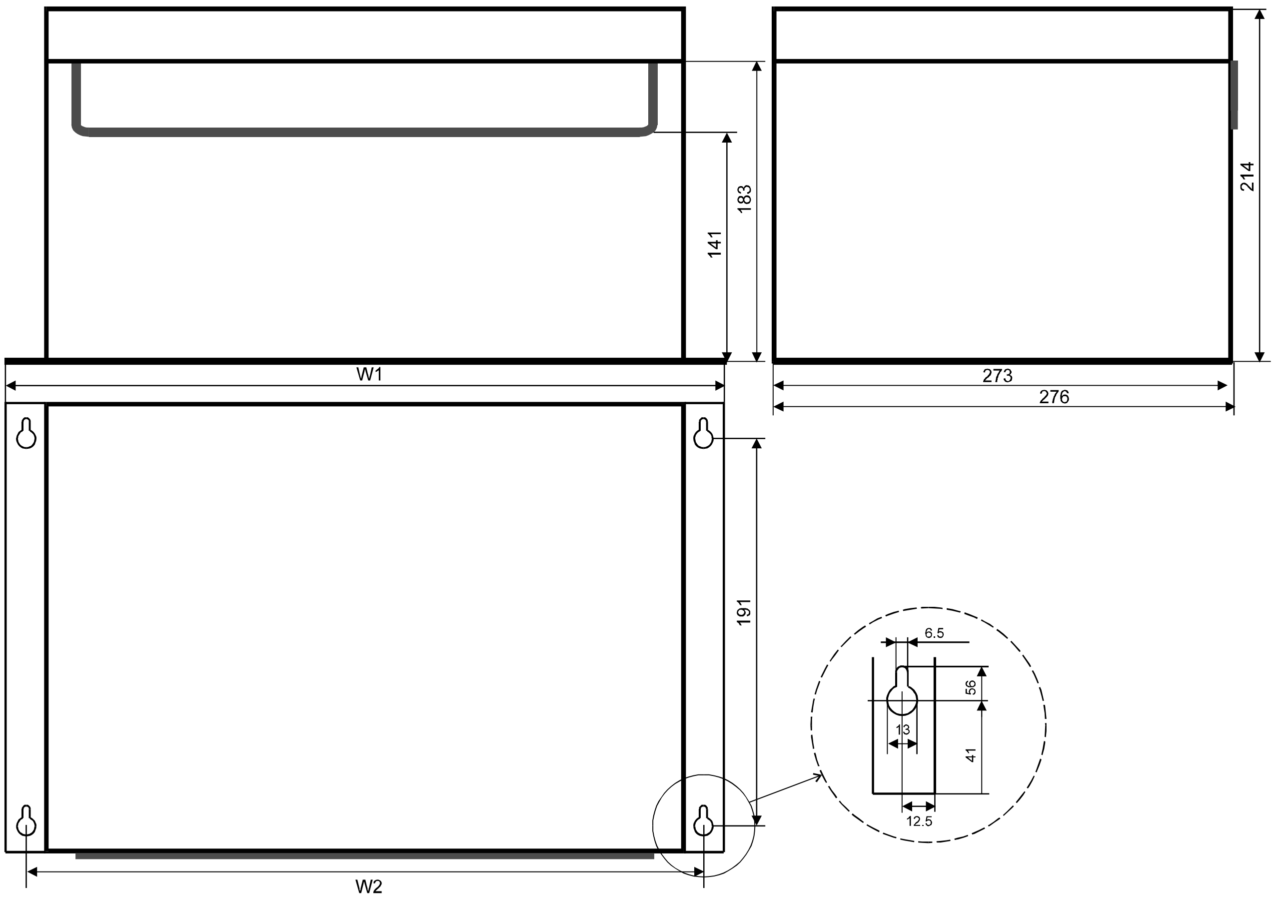
Variable types

English	Finnish
R	R
W	W

Appendix A. PHYSICAL INSTALLATION

Mounting the AL2000 in an enclosure is a rapid process due to key hole mounting. The Mounting Racks are fastened with 4 **M6** screws.

A.1 Mounting dimensions



RACK	MOUNTING SPACE REQUIREMENTS			DRILLING DISTANCES	
	Width W1 mm	Height mm	Depth mm	Width W2 mm	Height mm
MR3 mini	138	276	214	128	191
MR3	205	276	214	180	191
MR5	270	276	214	245	191
MR11	390	276	214	365	191
MR16	490	276	214	465	191

NOTE: At least 36 mm space should be left below the AL2000 for cabling.

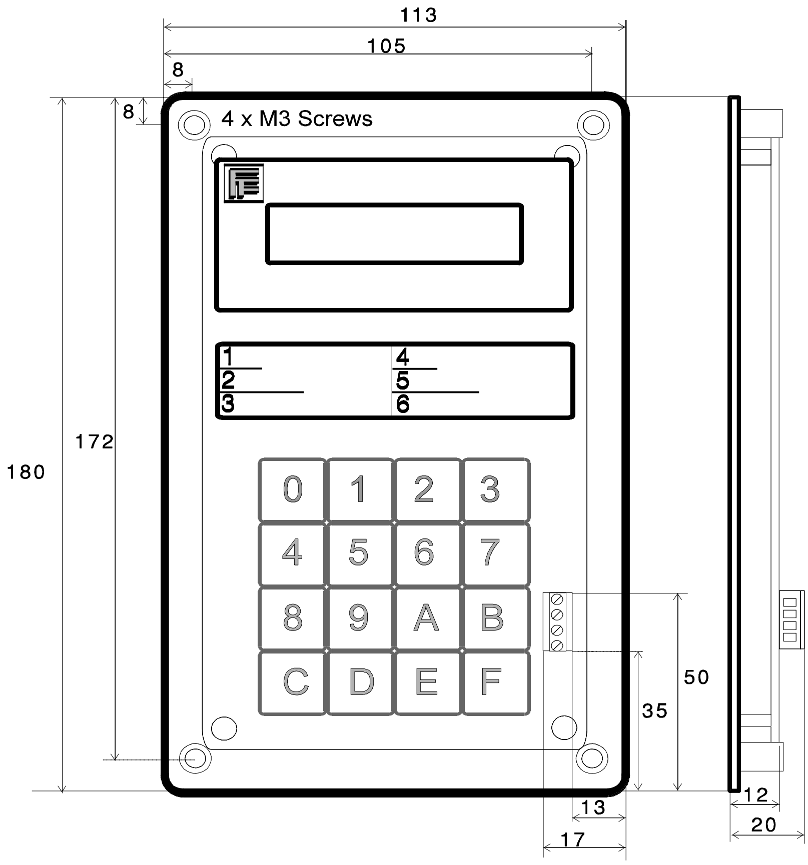


Figure A.2 Dimensions of display/keypad unit AL1093D

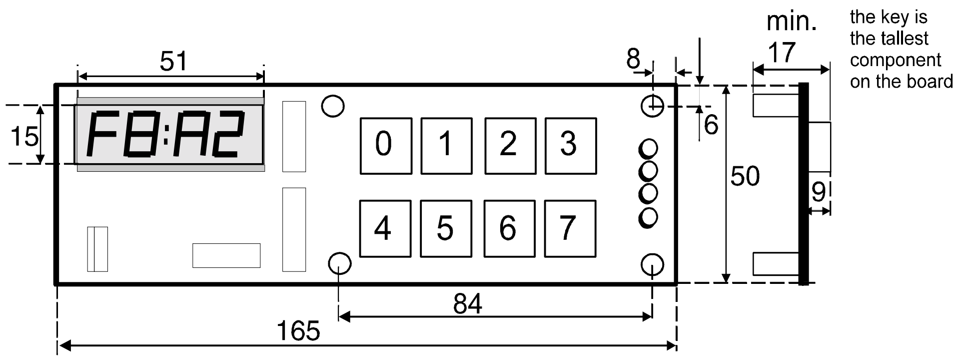


Figure A.3 Dimensions of display/keypad unit AL1094

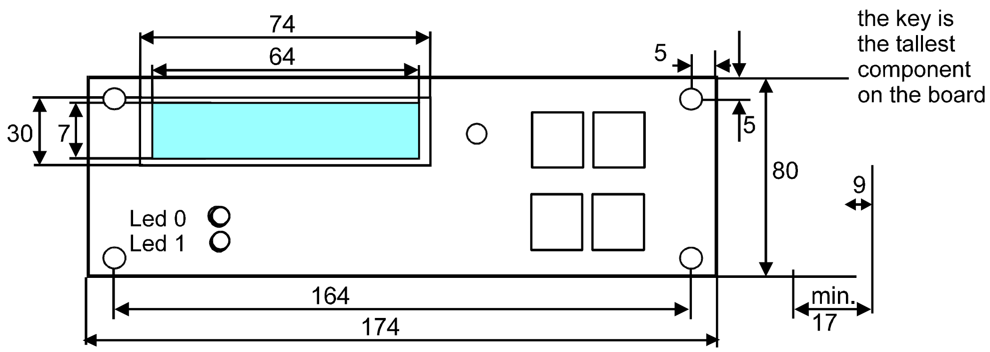


Figure A.4 Dimensions of display/keypad unit 1094R

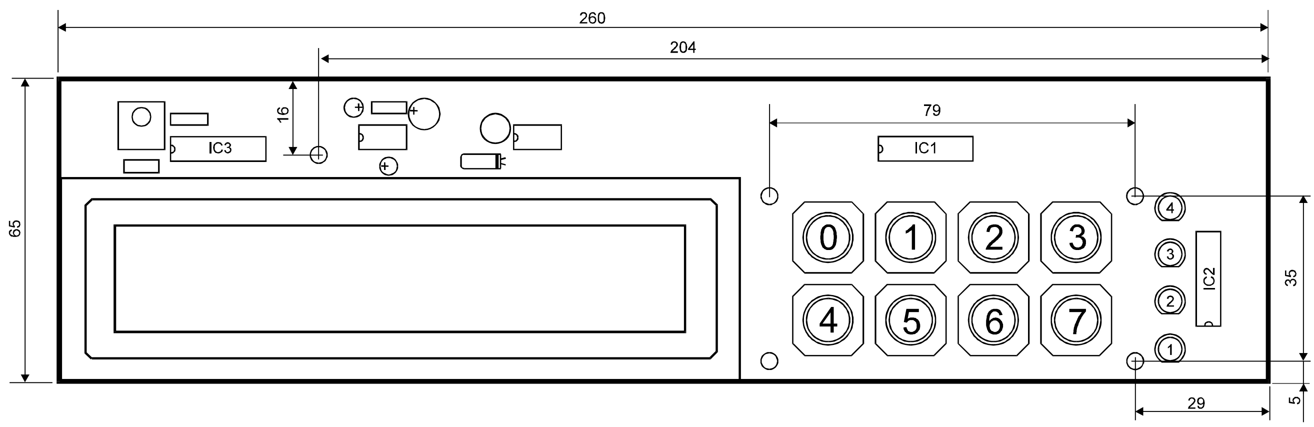
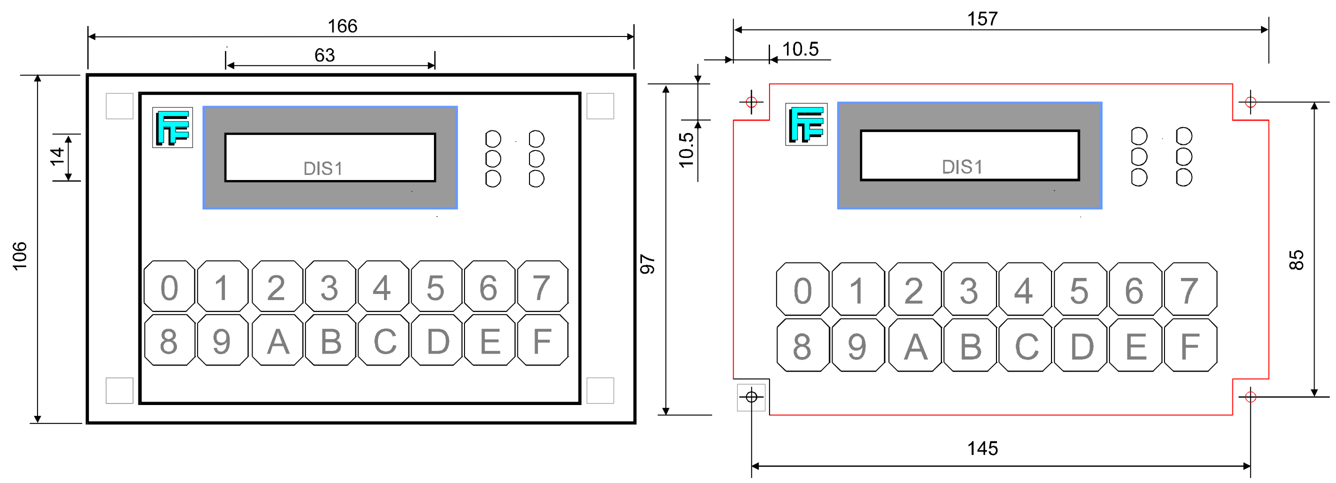


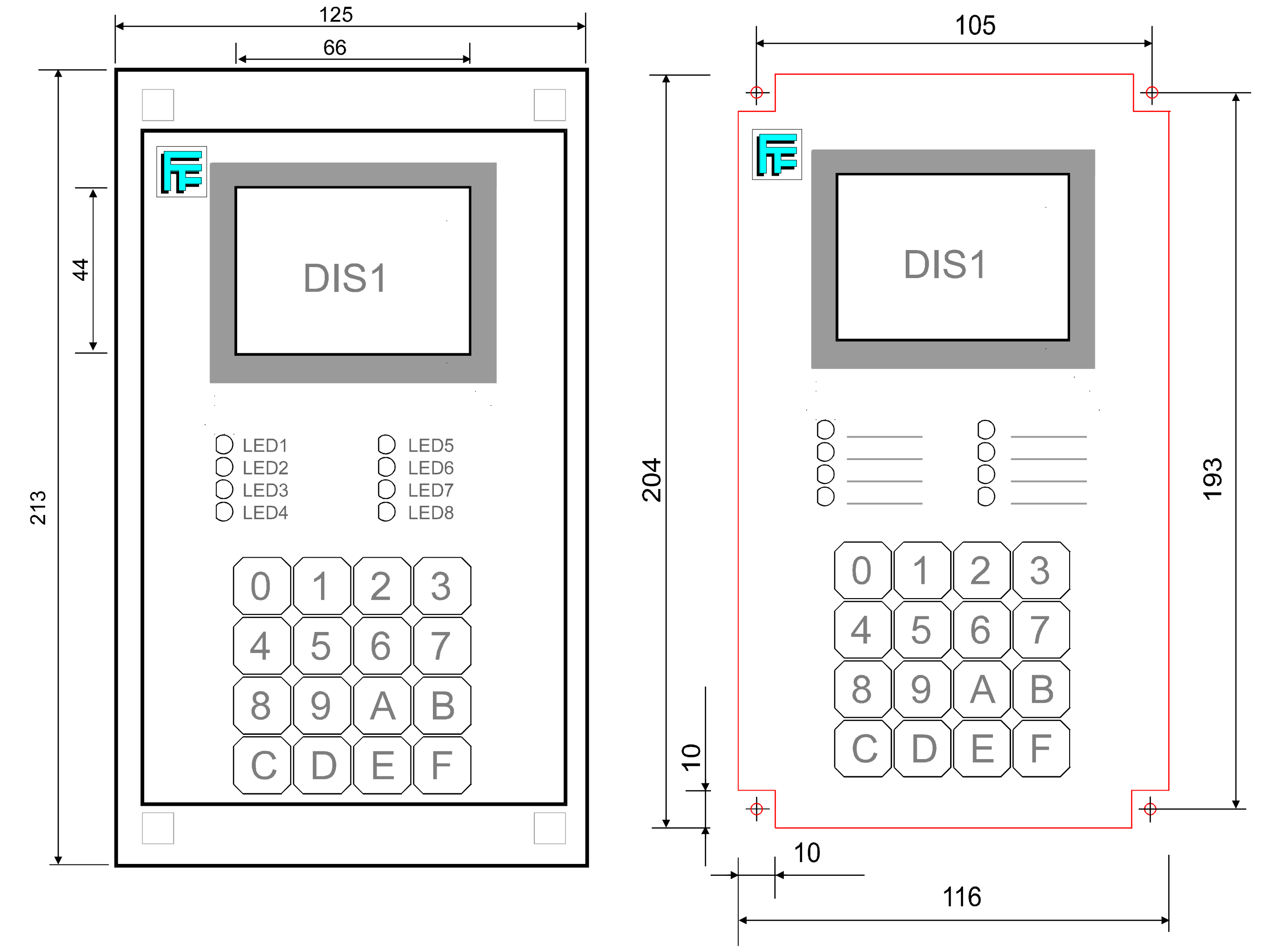
Figure A.5 Dimensions of display/keypad unit AL 1094AF



Display

Cutout

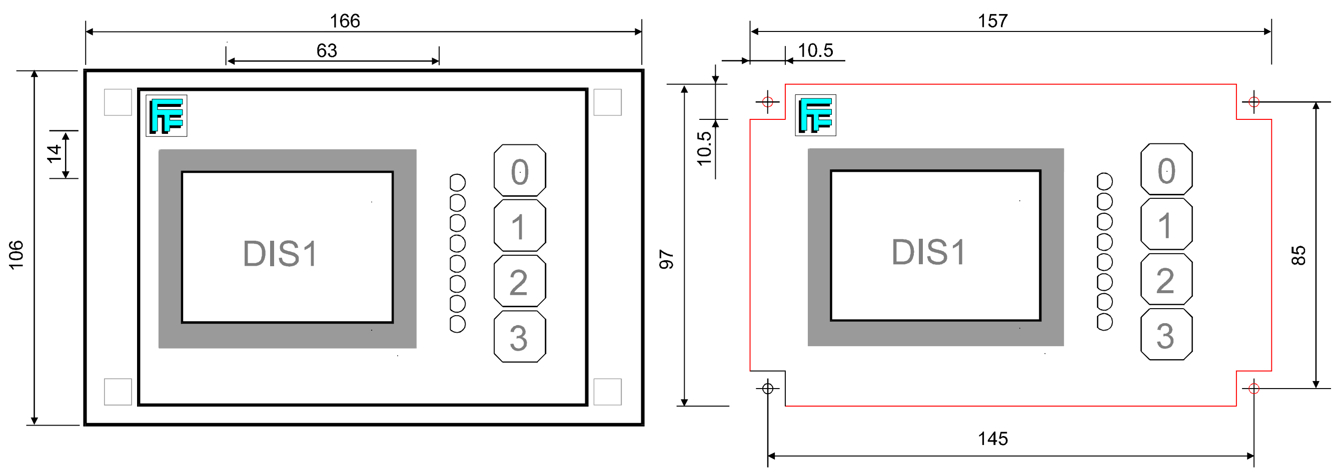
Figure A.6 Dimensions of display/keypad unit AL1093F



Display

Cutout

Figure A.7 Dimensions of display/keypad unit AL1095A



Display

Cutout

Figure A.8 Dimensions of display/keypad unit AL1095B

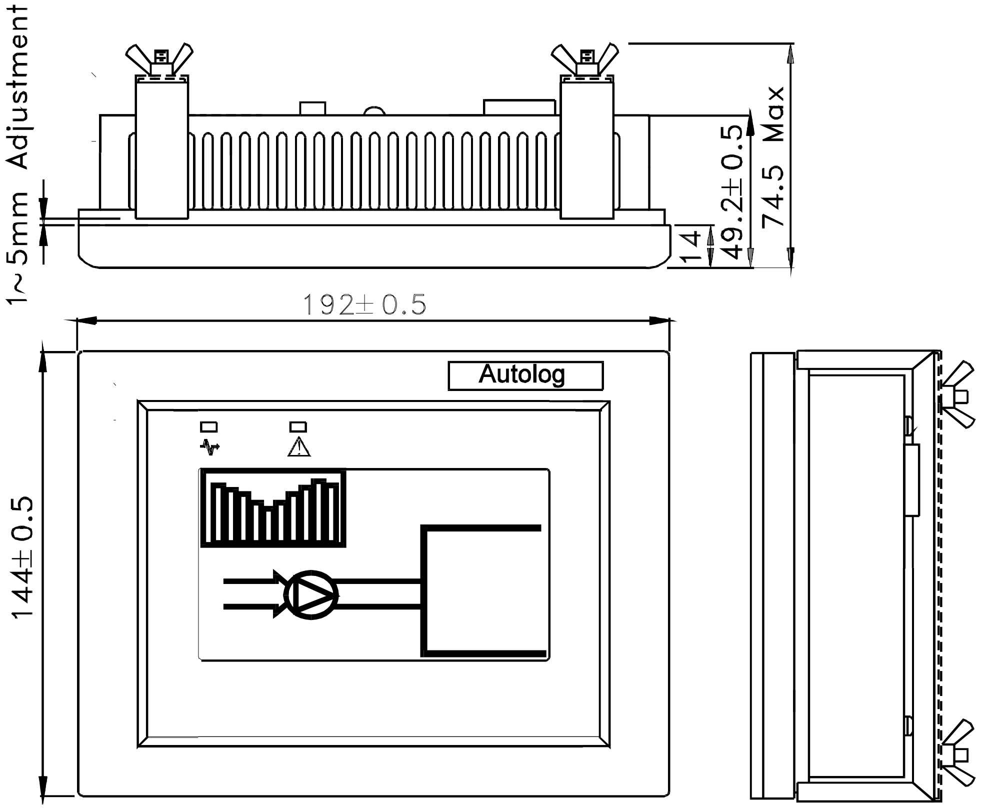


Figure A.9 Dimensions of graphic display - keypad/touchscreen unit AL1096S/T

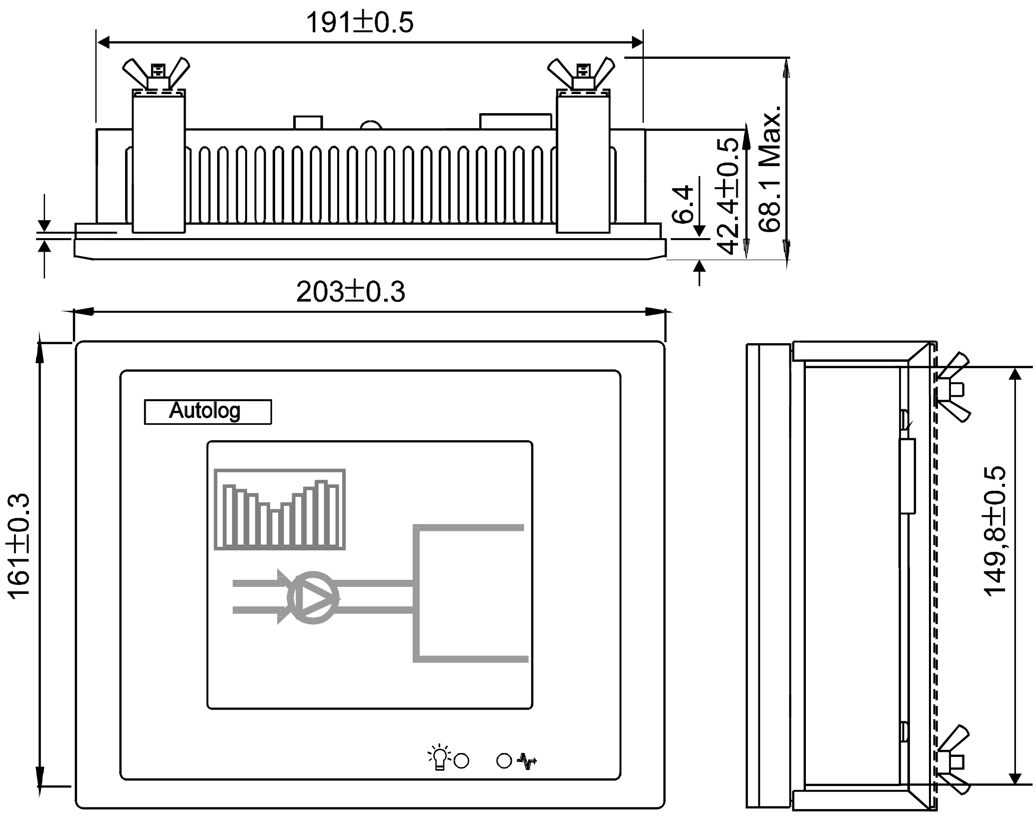


Figure A.10 Dimensions of graphic display - touchscreen unit AL1096PS/PE

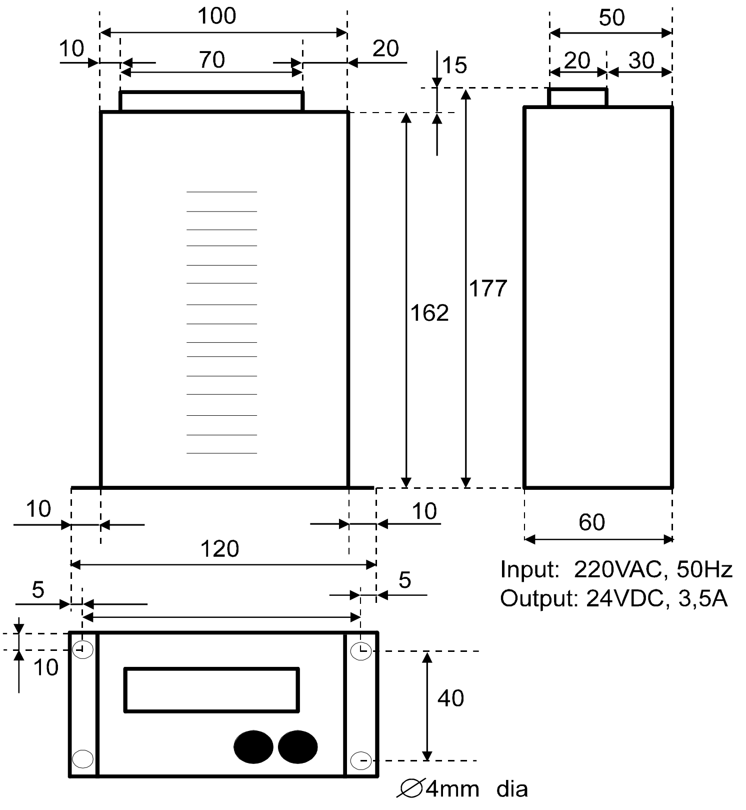


Figure A.12 Dimensions of AL9624/3.5 power supply

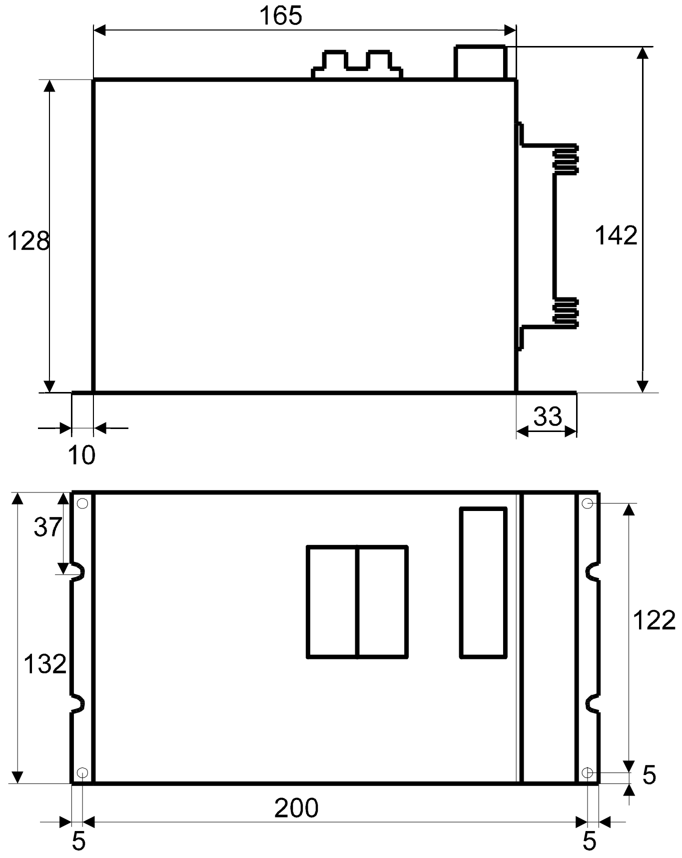


Figure A.12 Dimensions of AL9624/8 power supply