KRAUSSMAFFEI
Kunststofftechnik

MC4

Central Computer Interface

Version 1.4

Status: September 1999

List of contents

1 General notes

The MC4-computer interface has been provided for inputting actual-values, switching-points, items of status information, alarms and setting-data for the MC4 control system. Setting data can also be entered there.

It provides an opportunity for optimising the setting data of several machines at a central point, where these data can also be administered; actual values and alarms can be monitored from here as well.

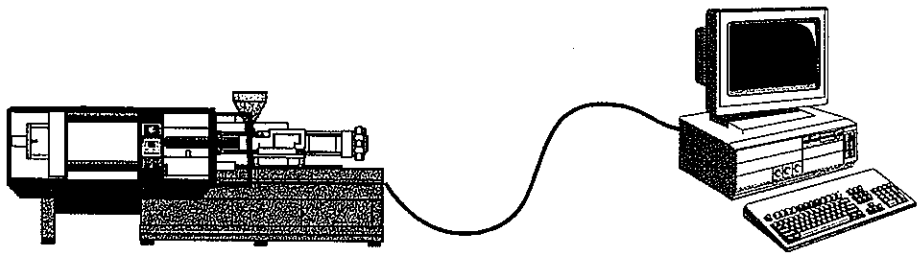
In this documentation, various abbreviations are employed, which have the following meaning:

MC4	MC4-Injection Moulding Machine control system
ZR	Main host PC
Host	Host computer
Netzwerk	10BaseT Ethernet
TP	Terminal Protocol
HP	Host Protocol
NTP	Network Transport Protocol

2 Interface

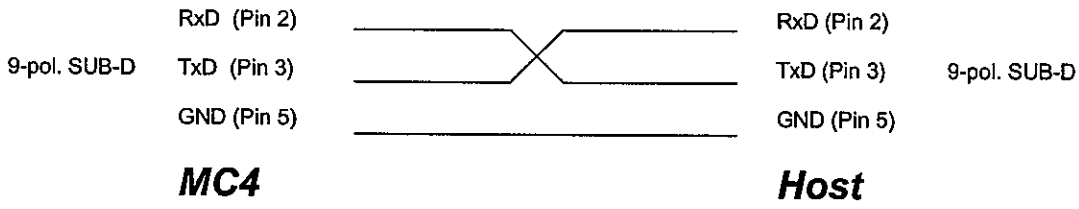
2.1 RS232 Connection

For the RS232 connection the host computer is connected directly to the MC4 machine by a serial cable.



In its physical manifestation, the computer interface has been designed as serial interface and subset of standard V.24 (CCITT), or RS232C. A 9-pin sub-miniature D-socket, whose contacts are assigned as follows, has been mounted on the control cabinet of the MC4:

Contact:	Abbreviation	Meaning
2	RxD	Reception data (for the MC4-control system)
3	TxD	Transmission data (from the MC4-control system)
5	GND	Signal earth



The standard settings of the interface are as follows in this case:

1 Startbit
8 Data bits
0 Parity bits
1 Stop bit

The Baud-rate is 19200.

There is **no** hand shake. It must therefore be assured that the connected host PC will always be able to accommodate a complete telegram in its reception buffer. The host PC's reception buffer must have a capacity of **at least 1024 Byte**.

If the connection is operated in "extended mode" (see INIT command), the receive buffer must have 8Kb.

The **cable length** should not exceed a **maximum** of **15 meters**. Otherwise an interface converter RS232/RS422 should be inserted on both sides, which permits cable lengths up to several hundred meters.

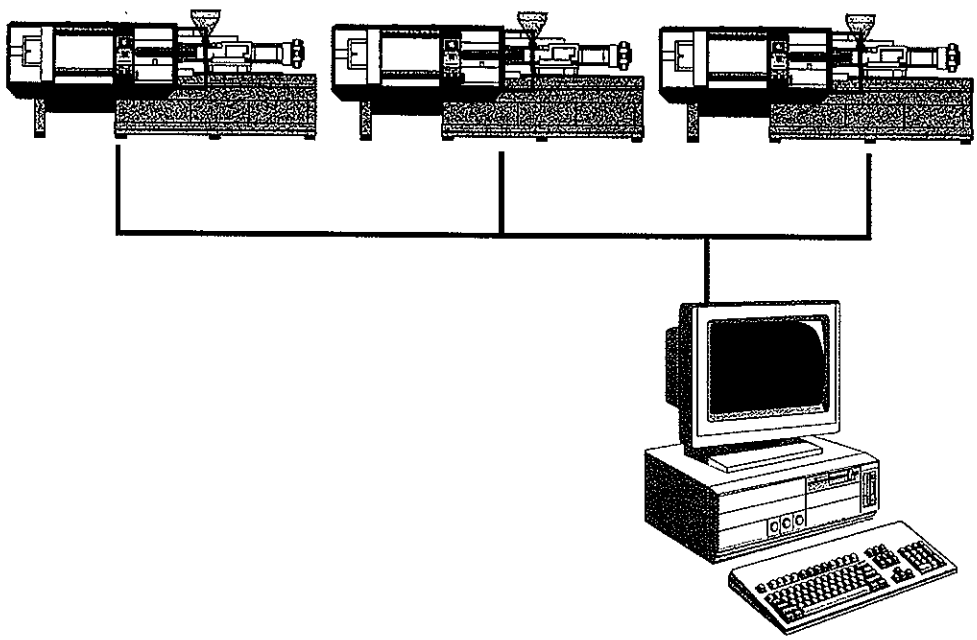
2.2 Network connection through 10BaseT Ethernet

As from version 1.0.04 the MC4 also supports an **Ethernet** network of type **10BaseT**. This is a 10 MBit Ethernet with Twisted Pair cables (UTP or STP) and with RJ45 connecting plugs (Western plugs) at both ends.

RJ45 pin assignment:

Pin:	Abbreviation	Meaning
1	TD+	Send line +
2	TD-	Send line -
3	RD+	Receive line +
6	RD-	Receive line -

The network has a **star-shaped** structure, and each MC4 machine must have a connection to the **hub** (star distributor). The hub forms the central exchange.



The **length of the cables** from and to the hub must not exceed **100 meters**. To obtain greater noise immunity, STP cables (Shielded Twisted Pair) should be used in preference to UTP cables (Unshielded Twisted Pair).

The average response time for the network connection is approx. 200ms.
The maximum response time is 1 sec.

3 Protocol

The MC4 control system supports various transmission protocols, depending on the type of connection. The data contents are always the same, regardless of the transmission protocol.

In **Terminal mode,** the interface can be operated with the aid of a simple terminal program (e.g. Windows Terminal). It is a line-by-line transmission. However, this mode has the disadvantage that the data contents cannot be checked for correctness.

At identical functionality, the **Host mode** on the other hand has the facility for checking the data for correctness against the transmitted check-sum.

The **network transport protocol (NTP)** is used for network connections. This subprotocol is based on TCP/IP and also ensures error-free transmission.

As a rule, the MC4 control system is always passive. It responds only to a command received from the host PC. The host determines the selection of the protocol-type, by the request being sent with or without check-sum. The MC4 always replies in the same manner in which the request took place.

ASCII-characters:

are set in inverted commas, or when several characters have been strung together to form a string, they are set in quotation marks.
e.g.: 'C' or "TEXT"

The binary value of an ASCII-character may also be shown in hexadecimal form. Thus for instance the expression 'C' corresponds to the hex.-value 0x43.
In telegrams, only portrayable ASCII-characters are transmitted (0x20 ... 0xFF)
All control-codes (smaller than 0x20) are employed for controlling the data transfer. They must not be used within a telegram. In the subsequent text, control-codes will be represented by a preceding ^-character (e.g. ^C for 0x03).

ASCII-control characters:

Abbreviation of the ASCII-character by capital letters. The following special ASCII-characters are employed:

STX	Start of Transmission (0x02, ^B)
ETX	End of Transmission (0x03, ^C)
BS	Backspace (0x08, ^H)
LF	Line Feed (0x0A, ^J)
CR	Carriage Return (0x0D, ^M)
DC2	Device Control 2 (0x12, ^R)
CAN	Cancel (0x18, ^X)
DEL	Delete (0x7F)

The transmission protocols are described on the following pages. The individual MC4 master computer commands are given in the chapter "Commands". The possible commands are identical for all transmission protocols.

All commas (,) in the command strings on the following pages are intended only to separate the different components.

3.1 Terminal mode

The terminal protocol can only be used with a **serial connection!**

A connection to the MC4 control system can easily be established with the aid of a simple terminal program (e.g. Windows Terminal or Telix), while in the Terminal Mode.

The Terminal Mode employs the following ASCII-control characters:

BS or DEL	delete last character
LF	is ignored by the MC4
CR	finish character, starts the command execution in the MC4 control system
CAN	Interface reset (employing the reception buffer)
DC2	Resend (send the last string once more)

Structure of an enquiry string:

The enquiry telegram consists of a command (an ASCII-letter) followed by a colon (0x3A). Subject to the command, this may be followed by transfer parameters, which are separated by a vertical line (ASCII-Code 0x7C). The telegram is concluded with a CR-signal.

Syntax: <Command+Data>,CR,[LF]

Reply-string structure:

The MC4 replies in different ways, subject to command. The reply string starts in each case with the requested command, however.

Syntax: <Command+Data>,CR,[LF]

Request for resend:

If the string is incorrect, it can be requested again at any time using the DC2 character.

Syntax: DC2

The actual data contents must contain only ASCII characters 0x20 to 0xFF.

A list of the commands and their functions is given in chapter 4.3.

3.2 Host mode

The host protocol can only be used with a **serial connection!**

The host mode has been designed for communication with a superposed master computer Basically, it is very similar to the Terminal Mode.
In the host mode, a telegram is additionally issued with a check-sum for fault identification.

The following ASCII-control characters are employed in the Terminal Mode:

STX	Start character
ETX	Concluding character, starts command execution in MC4 control system
CAN	Interface reset (emptying the reception buffer)
DC2	Resend (send last string once more)

Enquiry-string structure:

The enquiry telegram always starts with an STX-code, followed by the command (an ASCII-letter and a colon (0x3A). Subject to command, this may subsequently be followed by transfer parameters, which are in each case separated by a vertical line.
The telegram is concluded with the check-sum (as 2-figure ASCII-Hex number) and an ETX-code.
The check-sum is calculated form the total of all characters modulo 256 (without STX, ETX and check-sum) and always consists of a 2-figure hex-number.

Syntax: STX,<command+data>,<check-sum>, ETX

Reply-string structure

The answer is similar to the Terminal Mode, but with the addition of start- and finishing codes, address and check-sum, in the manner of the enquiry string.

Syntax: STX,<Command+data>,<check-sum>,ETX

Request for resend:

STX,DC2,<check-sum>,ETX

A list of the commands and their functions is given in chapter 4.3.

3.3 Network protocol (NTP)

The network protocol NTP is used when the MC4 machines are connected via **Ethernet**.

The **TCP/IP protocol** serves as the basic protocol for the network connection. The MC4 control system expects a TCP/IP Network environment in a class C network. At present the **network number** is set at **196.136.153.xxx** (in hexadecimal: C4.88.99.xx). The host ID's of the individual machines must be entered in the MC4 control systems. The network numbers must be assigned in accordance with TCP/IP standards.

The packets are transmitted as streams (TCP protocol) at **Port 18901**. Since this method already ensures error-free transmission, no additional check-sum calculation is required at this point. The NTP protocol expands the data to be transmitted only by the addition of information concerning the length and type of data.

The NTP frame has the following structure:

Bit 7		Bit 0
0	Ctrl	Data type
Data length (MSB)		
Data length (LSB)		
Data (with length 0..65535)		

Details of fields:

Data type byte:

Bit 7 of the data type byte must always be 0. This bit is reserved for future expansion. The Ctrl bit (Bit 6) is only set on in the case of a control telegram for the transmission layer. In the case of the MC4 computer interface 0 should always be given here. The data type describes the type of telegram, in other words which kind of data have been transmitted. Here 0 must always be given.

Data length:

Two bytes which describe the length of the following data in the telegram. Theoretically up to 65535 bytes can be transmitted in one telegram. At present telegrams of **max. 8 Kb** are transmitted.

Data:

Within the data block any characters and data may be used.

3.3.1 Login telegram

When the host computer is connected to the MC4 you first have to login to the MC4 before you can exchange data packages through the net. This is done with the login telegram.

The Login telegram has the data type 1 and provides a user name and a password, both terminated with a NUL character. At the moment the user name and the password is not checked in the MC4, so the can be omitted.
It's sufficient to send the following three bytes as login telegram.

01H (Login)
00H Data length (MSB)
00H Data length (LSB)

3.3.2 Data telegram

Only if you are logged into the MC4 you can send data telegrams. When you send data packages you always have to use data type 0.
Thus a data telegram always starts with a 0 followed by two bytes specifying the data length and finally the real data.

00H (Data telegram)
Data length (MSB)
Data length (LSB)
Data (with max length 8k bytes)

A list of the commands and their functions is given in chapter 4.3.

4 The MC4 central computer protocol

4.1 Parameters

4.1.1 Parameter designation convention

Parameter designations are employed, in order to be able to identify parameters when reading or writing values. These parameter names are firmly defined for all MC4-extrusion lines.
Parameter designations always consist of 3 name parts, which are separated from each other by a full stop:

<parameter object>.<parameter element>.<parameter value>

All functionally related parameters are united in a <parameter object>. Thus parameter object SCRW1_H_BAR_Z01 contains all parameters for instance, that are connected with the barrel heating zone 1 of injection unit 1.

A single element of the parameter object is designated <parameter element>. The number of possible parameter elements within an object is subject to its type. In the case of a barrel heating zone, the following parameter elements are available, for example:

SET	Set-value
ACT	Actual-value
ED	ED-position (set-value programming)
EDACT	current heating-/cooling value
XPH	Proportional value for heating
TAH	Heating-controller scanning frequency
TNH	Integral action time for heating
...	

With the <parameter value>-detail it is determined, which information of the parameter element is to be read or written. Conventional parameter values are

VAL	current value
MIN	minimum value
MAX	maximum value
LTXT	parameter plain text
DIM	Dimension (= physical unit)
...	

If you want to read out the maximum value of the set value of barrel zone 1, the complete parameter name must read as

"SCRW1_H_BAR_Z01.SET.MAX".

4.1.2 Parameter handle

A parameter name can be up to 36 characters long. In order to reduce the number of transmission data, the MC4 computer interface has the facility for allocating an identification number (parameter handle) to each parameter designation. These parameter handles consist of **4-figure hexadecimal numbers**, which carry the '$'-sign as prefix (0x24). This means, that the parameter designation can be reduced from a maximum 36 characters to just 5 of them.

Example: "$0001" or "$001B"

The MC4 always assigns a parameter handle to a parameter name. The Q-command must be used, when asking for a parameter handle.

Once the MC4 has been asked for a parameter handle, each access to this parameter can be controlled through the handle from then on.
All asked-for parameter handles are consecutively numbered from 0 onwards, so that the parameter handle can be used directly as an index of an administration list.

4.2 Extended telegram mode

There are two different telegram modes for the MC4 central computer interface, the standard mode and the extended mode.

In **standard mode** up to four (4) parameters can be treated within one telegram.

In **extended mode** up to 16 parameters can be transmitted in one telegram. In the extended mode it is important to ensure that the interface reception buffer is large enough for the higher number of parameters. A buffer capacity of 8 KByte is therefore required.
The extended mode is especially recommended for network connections, since the reduced number of telegrams results in a higher throughput rate than shortening the length of telegrams.

The telegram mode can be selected using the INIT command.

4.3 Commands

For explaining the commands, only the data-section of the program will be displayed in the text following. In the Terminal Mode, CR must be set in conclusion, whereas in host mode, the complete transmission frame-work, with STX, ETX and check-sum has to be put.

4.3.1 Command I (Init): Interface reset

Function: All parameter requests, which may possibly have been set previously, are deleted.
All allocated parameter handles are erased and are no longer valid.
This command ought to be carried out once before the start of every interface communication.

A distinction is made between "standard" and "extended" operating modes during initialization.
In "standard mode" up to 4 parameters per telegram can be transmitted.
In "extended mode" up to 16 parameters per telegram can be sent. In "extended mode" the reception buffer must have a maximum capacity 8 Kb.

ZR → MC4:
1) "I:" Initialize and activate "standard mode"
2) "I:X" Initialize and activate "extended mode"

MC4 → ZR: "I:*"

Faults: None.

Example:
ZR → MC4: "I:"
MC4 → ZR: "I:*"

4.3.2 Command V (Version): Read MC4 program version number

Function: With this command, the version number of the MC4 VIS-PROG (program) can be read. The version number allways is shown in this format x.x.xx-xxx.

ZR → MC4: "V:"

MC4 → ZR: "V:x.x.xx-xxx"

Errors: none.

Example: ZR → MC4: "V:"
MC4 → ZR: "V:1.0.04-000"

4.3.3 Command D (database): Read MC4 database version number

Function: With this command, the version number of the MC4 VIS-PARAM (database) can be read. The version number allways is shown in this format x.x.xx-xxx.

ZR → MC4: "D:"

MC4 → ZR: "D:x.x.xx-xxx"

Errors: none.

Example: ZR → MC4: "D:"
MC4 → ZR: "D:1.0.02-000"

This command is only available with version VIS-PROG 1.0.04-xxx or higher.

4.3.4 Command Q (Query Handle): Request parameter handle

Function: With this command, identification numbers (so-called parameter handles) can be allocated to each MC4 parameter name. These handles are displayed as 4-figure hex numbers, with a '$'-sign as prefix.

ZR → MC4: "Q:p1| p2 | ... |pn"

p1,p2,..,pn Parameter name
Up to 4 parameter handles (up to 16 in extended mode) can be requested simultaneously.

MC4 → ZR: "Q:h1| h2 | ... |hn"

h1,h2,..,hn Parameter handles (see Chapter 3.3.2)
The parameter handles are shown as 4-figure hexadecimals in the shape of an ASCII-string with a '$'-sign prefix.
Should the parameter name prove to be wrong, or the parameter is non-existent, a question mark is returned instead of the parameter handle.

Example: ZR → MC4: "Q:MOLD_S_CLOS_SAFE1.SET.VAL|XYZ"
MC4 → ZR: "Q:$0000|?"

Handle $0000 is returned for the first parameter. The second one is invalid, however and is therefore acknowledged with a question mark.

4.3.5 Command L (Block): Get all available Parameters

Function: The command gives the possibility to get all Parameters, which are available für central computer (ZR). The transfer is in Blockmode with 4 parameters at once.

ZR → MC4: "L:"

Up to 4 parameters (in extended mode till 16) can be requested.

MC4 → ZR: "L:p1| p2 | ... |pn"

p1,p2,..,pn Parameter names
The Parameter names are ASCII-Strings.

Beispiel:
ZR → MC4: "L:"
MC4 → ZR: "L:EJEC_C_HYDR_REPEAT.ACT|
EJEC_C_HYDR_REPEAT.SET|...| EJEC_C_HYDR_REPEAT.CYCACT"
...
ZR → MC4: "L:"
MC4 → ZR: "L:*"

After each finished datablock the ZR must call for the next block. If no more parameters are available, the MC4 respondes with "L:*".
With "l:" (small L) the block transfer can be aborted. The MC4 answers with "l:*".

This command is available since VIS-PROG-Version 1.4.00-xxx.

4.3.6 Commando B (Block): All Parameter value with Attribut "Disk" read

Function: This command requests all parameter values, which are saved on diskette, no parameters (like actual values). The transfer uses the block mode.

ZR → MC4: "B:"

Up to 4 parameters (in extended mode till 16) can be read.

MC4 → ZR: "B:p1=v1|p2=v2| ... |pn=vn"

p1,p2,..,pn	Parameter names
=	separation sign
v1,v2,..,vn	Parameter values

The parameter names are ASCII-Strings.
The parameter values are decimal numbers as ASCII-Strings.

Beispiel:
ZR → MC4: "B:"
MC4 → ZR: "B:MOLD_H_TEMP1_Z101.SET=250,6|
MOLD_H_TEMP1_Z101.SFTMOD=1|...
MOLD_H_TEMP1_Z101.PARSET=4"
...
ZR → MC4: "B:"
MC4 → ZR: "B:*"

After each finished datablock the ZR must call for the next block. If no more parameters are available, the MC4 respondes with "B:*".
With "b:" (small L) the block transfer can be aborted. The MC4 answers with "b:*".

This command is available since VIS-PROG-Version 1.4.00-xxx.

4.3.7 Command R (Read): Parameter values reading

Function: With this command set-values, actual-values, limiting values, as well as texts of parameters can be read-out.

ZR → MC4: "R:p1| p2 | ... |pn"

p1,p2,..,p4 Parameter names or parameter handles
Up to 4 parameter values (up to 16 in extended mode) can be read simultaneously.

MC4 → ZR: "R:v1| v2 | ... |vn"

v1,v2,..,vn Parameter values
Parameter values are displayed as decimal figures in the shape of an ASCII-string.
Should the parameter name/the parameter handle be wrong, or the parameter be non-existent, a question mark is returned instead of the parameter value.

Example: ZR → MC4: "R:$0000|EJEC_S_AIR1_START.SET.VAL|XYZ"
MC4 → ZR: "R:16,1|88,7|?"

The value is returned for the first two parameters, the third one is invalid, however and is therefore acknowledged with a question mark. A handle had already been requested for the first parameter and is therefore shown here, instead of the name.

4.3.8 Command W (Write): Parameter writing

Function: Parameter set-values can be written to the MC4 with this command.

ZR → MC4: "W:p1=v1| p2=v2 | ... |pn=vn"

p1,p2,..,pn Parameter names or parameter handles
Up to 4 parameter (up to 16 in extended mode) values can be written simultaneously.

v1,v2,..,vn Parameter values: the parameter values are displayed as decimal figures in the shape of an ASCII-string.

MC4 → ZR: "W:q1| q2 | ... |qn"

q1,q2,..,qn Acknowledgement. The following characters can be returned as acknowledgement:

* (0x2A):	The parameter value has been written successfully.
+ (0x2B):	The parameter value is too large.
- (0x2D):	The parameter value is too small
? (0x3F):	The parameter name/the parameter handle is wrong or the parameter is non-existent or the parameter cannot be written.

Example: ZR → MC4:
"W:INJU1_V_BACK_STEP1.SET.VAL=20,1|INJU1V_BACK_STEP2.SET.VAL=500"
MC4 → ZR: "W:*|+"

The first parameter has been written successfully, the second parameter value is too large.

Parameters that can be written become immediately effective in the MC4 and can affect the extrusion process directly. Therefore, writing access via the computer interface must be treated with the greatest care and should be avoided during production runs.

4.3.9 Command S (Set Request): Set-request for parameters

Function: This command allows frequently read parameters to be included in a request list. By selecting the N-command, it is then very easy to read-out all parameters, that are contained within this request list and which have changed since the last reading procedure. This function dispenses with the elaborate polling of a multitude of parameters.

ZR → MC4: "S:p1| p2 | ... |pn"

p1,p2,..,pn Parameter names or parameter handles
Up to 4 parameters (up to 16 in extended mode) can be requested simultaneously.

MC4 → ZR: "S:q1| q2 | ... |qn"

q1,q2,..,qn Acknowledgement. The following characters can be returned as acknowledgement:
* (0x2A): The parameter value has been requested successfully.
? (0x3F): The parameter name/the parameter handle is wrong or the parameter is non-existent.

Example: ZR → MC4: "S:$0000|MOLD_V_CLOS_BEGIN.SET.VAL|XYZ"
MC4 → ZR: "S:*|*|?"

The first two parameters have been requested successfully, the third parameter is invalid, however, and is therefore acknowledged with a question mark only. A handle had already been requested for the first parameter and has here been used instead of the name.

4.3.10 Command C (Clear Request): Cancel request for parameter

Function: With this command, requested parameters can be cancelled again from the request list. These parameters are then no longer taken into consideration with the N-command.

ZR → MC4: "C:p1| p2 | ... |pn"

p1,p2,..,pn Parameter names or parameter handles
Up to 4 parameter requests (up to 16 in extended mode) can be cancelled simultaneously.

MC4 → ZR: "C:q1| q2 | ... |qn"

q1,q2,..,qn Acknowledgement. The following characters can be returned as acknowledgement:
* (0x2A): The parameter value has been enabled successfully.
? (0x3F): The parameter name/parameter handle is wrong or. The parameter is non-existent.

Example: ZR → MC4: "C:SCRW1_P_HLDP_STEP02.SET.VAL"
MC4 → ZR: "C:*"

The parameters have been cancelled successfully from the request list.

4.3.11 Command N (New Values): Read altered parameters

Function: All parameters requested with the 'S' command are checked for any alteration in their value. Parameters, whose value has changed, are returned automatically, accompanied by the new current parameter value. A maximum of 4/16 parameters can be returned simultaneously in a telegram. By cyclic repeat performance of the N-command it is thus possible, to update changes in values continuously, without elaborate polling of each individual parameter.

ZR → MC4: "N:"

MC4 → ZR: "N:p1=v1| p2=v2 | ... |pn=vn"

p1,p2,..,pn	Parameter names or parameter handles If the parameter's handle is known, then that handle is used every time, rather than the parameter name. Up to 4 parameter values (up to 16 in extended mode) can be returned simultaneously.
v1,v2,..,vn	Parameter values: the parameter values are shown as decimal figures in the shape of an ASCII-string.

If no parameter and no value is returned, then there is no change in value of the parameters requested.

Examples: ZR → MC4: "N:"
MC4 → ZR: "N:$0000=15,7|MOLD_H_HEAT1_Z501.ACT.VAL=100,1"

There are two parameter alterations. Parameter name/handle and parameter value will be returned in each case.

ZR → MC4: "N:"
MC4 → ZR: "N:"

There are no parameter changes.

If the reply telegram is faulty, it is possible to request a repeat of the last telegram with the control code DC2 (0x12). (Repeating the N-command would not produce the same reply-string).

4.3.12 Command A (Alarm): Read and reset alarms

Function: With the 'A' command you have four ways to operate with alarm information.

ZR → MC4:	1)	"A:"	read all active alarms
	2)	"A:COUNT"	read current number of active alarms
	3)	"A:RESET"	reset alarms
	4)	"A:PRIO"	read alarms und their priority

MC4 → ZR:	1)	"A:a1|a2|..|an"
	2)	"A:c"
	3)	"A:*"
	4)	"A:a1,p1|a2,p2|...|an,pn"

a1,a2,..,an alarm numbers of all currently active alarms
The number of currently active can be read with the "A:COUNT" command.

p1,p2,...,pn alarm priorities. The smaller the value the higher the priority. The alarm type can be determined by the priority .

priority 0 .. 31	system alarm
priority 32 .. 63	switch-off alarm
priority 64 .. 95	shut-down alarm
priority 96 .. 127	display alarm

c Number of alarms:

Examples:

ZR → MC4: "A:"
MC4 → ZR: "A:315|212|501"
Currently alarms 315, 212 and 501 active.

ZR → MC4: "A:"
MC4 → ZR: "A:"
No alarm active.

ZR → MC4: "A:COUNT"
MC4 → ZR: "A:3"
Currently 3 alarms active.

ZR → MC4: "A:RESET"
MC4 → ZR: "A:*"
alarms reset.

4.3.13 Command M (Machine info): Read machine info

Funktion: With the 'M' command you can read the machine name, the order number and the machine number.

ZR → MC4: "M:"

MC4 → ZR: "M:<machine name><order no.><machine no.>

The machine name is displayed with 16 characters (with leading blanks). Order and machine number are displayed with 8 characters (with leading blanks).

Example: ZR → MC4: "M:"
MC4 → ZR: "M: □□□□□□KM□110-520□□□□C2+"
(□ = blank)

5 The MC4-Parameters

The designation convention of the MC4 parameter names has been described in Chapter 4.1.1.

As described there, each parameter name is composed of the following three name sections:

<object name>.<element name>.<value name>

The number of possible elements within an object and the names of these elements are subject to the object type.
The possible values of a parameter element on the other hand are a function of the element type (also called parameter type in subsequent text).

5.1 Formats of the Parameter values

The MC4 differentiates between various parameter types. Subject to parameter type, different values of the parameter can be requested. These values may be available in different formats.
However, all formats are ASCII-strings on principle, with differentiation being made between:

Decimal	represents decimal figures, with a possible decimal point being represented by a comma (0x2C). (e.g. "23,99")
Text	Any given texts, that must not contain the vertical separating line (0x7C) as a character however, (e.g. "Injection Moulding Machine").
Date	Quoting the date, with day, month and year are divided by full stops '.' (e.g. "07.12.96")
Time	Stating the time, with hours, minutes and seconds being divided by a colon ':' (e.g. "12:20:00"). The time can be shown with or without seconds.
Duration	Duration is given in hours and minutes, which are separated by a colon (e.g. "188:12")

5.2 Parameter elements

We differentiate between the following parameter element types:

1. Set analog value parameter <Objectname>.**SET.**<xxx>

The following values are available with an analog value parameter.

Name	Description	Format
VAL	current value of the parameters	Decimal
MIN	lower limiting value of the parameters	Decimal
MAX	upper limiting value of the parameter	Decimal
DIM	physical unit in the language of the actual country	Text
LTXT	Parameter text in the current country's language	Text

2. Actual analog value parameter <Objectname>.**ACT.**<xxx>

The following values are available with an analog value parameter.

Name	Description	Format
VAL	current value of the parameters	Decimal
MIN	lower limiting value of the parameters	Decimal
MAX	upper limiting value of the parameter	Decimal
DIM	physical unit in the language of the actual country	Text
LTXT	Parameter text in the current country's language	Text

3. Switching function / status <Objectname>.**SEL.**<xxx>

The following values are available with a switching function:

Name	Description	Format
VAL	current value of the parameter	Decimal
MAX	Maximum selection function value	Decimal
LTXT	Parameter text in the current country's language	Text

4. String parameter <Objectname>.**STR.**<xxx>

The following values are available with a string parameter:

Name	Description	Format
VAL	current string of the parameters	Text
LEN	Maximum permissible length of the string	Decimal
LTXT	Parameter text in the current country's language	Text

5. Date/time parameter <Objectname>.**TIM.**<xxx>

The following values are available with a date parameter:

Name	Description	Format
VAL	Value as no. of seconds since 01.01.1980	Decimal
DATE	Date in the format "tt.mm.jj"	Date
CLOCK	Time in the "hh:mm" format	Time
CLKSEC	Time as seconds in the "hh:mm:ss" format	Time
LTXT	Parameter text in the current country's language	Text

6. Duration parameter <Objectname>.**TIM.**<xxx>

The following values are available with a time parameter:

Name	Description	Format
VAL	Value as number of seconds	Decimal
TIME	Duration as "hhh:mm" format	Duration
TIMSEC	Duration in seconds as "hhh:mm:ss" format	Duration
LTXT	Parameter text in the current country's language	Text

7. Cycle actual parameter <Objectname>.**CYCACT.**<xxx>

The following values are available with the parameters:

Name	Description	Format
CYCVAL	Cycle value of the parameter of cycle CYCNUM	Decimal
CYCNUM	Cycle number of the cycle parameter	Decimal
CYCFLG	Flags: Bit 0 - Quality control activ Bit 1 - +Tolerance fault Bit 2 - - Tolernace fault Bit 3 – Startup scrap cycle Bit 8 – value not guilty	Decimal
DIM	Pysical dimension in current country's language	Text
LTXT	Parameter text in the current country's language	Text

All parameters except VAL can only be read! Also the parameter values of the elements ACT and CYCACT cannot be written.

5.3 Parameter description

You find the list of the parameters in the separate document MC4PL-x.DOC.

Many of the parameters listed on the subsequent pages are linked to purchasing options, which means, that they are therefore only available, when the respective functions have been isolated.

The "**Element**" column indicates the parameter element types, that are described in Chapter 4.3. The type identifies the values, that can be requested from this parameter.

The "**Dim**" column indicates the physical unit, in which a parameter value can be read or written.

Parameters , that can be written, become effective immediately in the MC4, so that they can influence the extrusion process at once. Therefore, any access by writing via the computer interface must be treated with the greatest care and should be avoided, when production is running.

6 Programming a host interface

6.1 General

The interface for linking up a MC4 machine can be programmed on all systems or platforms which provide and support the interfaces described in chapter 2 (RS232 and TCP/IP via 10BaseT Ethernet) using suitable drivers.

The MC4 computer interface uses a master-slave protocol, i.e. the MC4 (slave) responds only when the host computer (master) sends a telegram. The host computer must wait until it receives a response from the MC4. The MC4 responds to every telegram, even if only with an error message. If no answer can be received, the connection to the MC4 must be checked (correct baud rate, correct cable connection, correct network address, etc.).

6.2 Connection setup

Before the first connection setup, an I command should always be performed. This returns the MC4 computer interface to its initial state and deletes all parameter definitions.

As long as initialization of the MC4 control system has not been completed, no connection can be set up via the MC4 computer interface. This is shown by the error message "?" received in response to an I command.
The I command should therefore be repeated until a positive acknowledgement ("I:*") is received from the MC4.

After the I command, it is advisable to request a handle (Q command) for all parameters which will be read or written frequently via the interface. The use of handles reduces the quantity of data on the interface and simplifies the management of the parameters. The handle is a 4-digit unambiguous number which can be used instead of the (relatively long) parameter name as long as the connection is maintained. In addition the handles are assigned with ascending numbers, so that they can be used at the same time as indexes in a management list for the parameters.

Example:

Inquiry from the host
Q:SCRW1_N_PLAS_STEP01.SET.VAL|SCRW1_N_PLAS_STEP02.SET.VAL|SCRW1_N_PLAS_STEP03.SET.VAL|SCRW1_N_PLAS_STEP04.SET.VAL
Response from the MC4
Q:$0000|$0001|$0002|$0003
Inquiry from the host
Q:MOLD_H_HEAT1_Z501.ACT.VAL|MOLD_H_HEAT1_Z502.ACT.VAL|MOLD_H_HEAT1_Z503.ACT.VAL
Response from the MC4
Q: $0004|$0005|$0006
etc.

Thus it is possible to draw up a table containing all the information required concerning each parameter.

Name (string[16])	Value (string[20])	Type of parameter (WORD)	...
SCRW1_N_PLAS_STEP01.SET.VAL	50,5	decimal	
SCRW1_N_PLAS_STEP02.SET.VAL	55,3	decimal	
SCRW1_N_PLAS_STEP03.SET.VAL	62,4	decimal	
SCRW1_N_PLAS_STEP04.SET.VAL	65,7	decimal	
MOLD_H_HEAT1_Z501.ACT.VAL	220	decimal	
MOLD_H_HEAT1_Z502.ACT.VAL	210	decimal	
MOLD_H_HEAT1_Z503.ACT.VAL	220	decimal	
etc.			

By means of the handle (NOTE: the handle is a hexadecimal number) it is possible to directly consult the table and have access to all the information concerning a parameter.

After drawing up the table, all current parameter values must be read out (R command) and entered in the table (in the value field).

Finally all parameters which can change their value during the interface connection should be included in a request list (S command). The listing of a parameter causes the MC4 control system to register changes in the value of this parameter and to transmit them to the host upon request (using the N command).

Static information concerning the machine (e.g. version or machine name) does not need to be included in the list. It is sufficient if these data are read once after the connection has been set up.

6.3 Data exchange

In general parameter values should only be sent to the MC4 when they have changed, in order to avoid unnecessary loading of the interface.

The parameter values of the MC4 must be read out by polling (cyclic readout), since the MC4 computer interface only reacts to requests from the host system and thus cannot send information concerning changes in parameter values of its own accord. However, the polling procedure can be reduced to a minimum by using a parameter request list (see 'S' command). Listing parameters amounts to telling the MC4 that information will be required concerning any changes in certain parameters. A single command (see 'N' command) suffices to obtain a response telegram showing all changed parameters together with the new value. If several parameters have changed, they must be read out by repeating the N command.

After the connection setup has been completed (see chapter 5.3) it suffices to poll parameter changes using the N command.

If information is also required concerning the alarm status of the machine, it is possible to read out the current alarms of the MC4 by cyclic repeat performance of the A command. First the number of alarms should be requested using the command A:COUNT. If the number has changed since the last inquiry, details of each alarm can be read out using a further A command.

6.4 Severed connection

If no response is received after sending a telegram, this is a sign that the connection has been severed. In all cases the maximum response time should be allowed to pass before taking further measures.

With TP or HP, one or two repeat attempts should be made following a waiting period, before declaring the connection severed.
With NTP, error-free transmission is ensured by TCP/IP. Therefore no repetition of the telegram is necessary for network connections.

In general, following a severed connection, the receive and send buffers and all variables or tables relating to the control of the connection should be deleted or reset. Then a new connection setup should be started, beginning with the I command.

6.5 Programming with TCP/IP

```
/*****************************************************************/
/*                                                               */
/*   OS/2 example file for MC4 network connection                */
/*                                                               */
/*****************************************************************/

#include <stdio.h>
#include <stdlib.h>
#include <types.h>
#include <netinet\in.h>
#include <sys\socket.h>
#include <netdb.h>

main(int argc, char *argv[])
{
    unsigned short port;        /* port client will connect to
*/
    char buf[1024];             /* data buffer for sending and
receiving    */
    char address[32];           /* server address
*/
    struct sockaddr_in server; /* server address
*/
    int s;                      /* client socket
*/

    /*
     * Initialize with sockets.
     */
    sock_init();

    /*
     * Get the MC4 address.
     */
    strcpy(address, argv[1]);
    if (strlen(address) == 0)
    {
        fprintf(stderr, "Invalid MC4 address\n");
        exit(2);
    }

    /*
     * port number for MC4 central computer interface
     */
    port = 18901;

    /*
     * Put the server information into the server structure.
     * The port must be put into network byte order.
     */
    server.sin_family      = AF_INET;
    server.sin_port        = htons(port);
    server.sin_addr.s_addr = inet_addr(address);

    /*
     * Get a stream socket.
     */
    if ((s = socket(PF_INET, SOCK_STREAM, 0)) < 0)
    {
        psock_errno("Socket()");
```

```
        exit(3);
    }

    /*
     * Connect to the server.
     */
    if (connect(s, (struct sockaddr *)&server, sizeof(server)) < 0)
    {
        psock_errno("Connect()");
        exit(4);
    }

    /*
     * LOGIN to MC4
     */
    buf[0] = 1;             ; data type
    buf[1] = 0;             ; high order byte of data length
    buf[2] = 0;             ; low order byte of data length

    if (send(s, buf, 3, 0) < 0)
    {
        psock_errno("Send(LOGIN)");
        exit(5);
    }

    /*
     * Put the INIT command message into the buffer.
     */
    buf[0] = 0;             ; data type
    buf[1] = 0;             ; high order byte of data length
    buf[2] = 2;             ; low order byte of data length
    buf[3] = 'I';               ; INIT command
    buf[4] = ':';

    if (send(s, buf, 5, 0) < 0)
    {
        psock_errno("Send(INIT)");
        exit(5);
    }

    /*
     * The MC4 sends back the answer. Receive it into the buffer.
     */
    if (recv(s, buf, sizeof(buf), 0) < 0)
    {
        psock_errno("Recv()");
        exit(6);
    }

    /*
     * Close the socket.
     */
    soclose(s);

    exit(0);

}
```

Example:

TEST 196.136.153.10