Prof.Dr.H.Ziegler CEN/TC176-WG4-N107 Rev.4

Proposal for future M-Bus Application Layer

1. Introduction

The M-Bus application layer describes a standard especially for meter readout. It can be used with various physical layers and with link layers and network layers which support the transmission of variable length binary transparent telegrams. The first byte of an application layer telegram is the CI-field which distinguishes between various telegram types and application functions. It is also used to distinguish between true application layer communication and management commands for lower layers. The meaning of the remaining bytes of the telegram depends also on the value of the CI-field. This second revision is a compatible enhancement of the sections 6.4 to 6.6 of the original standard EN1434-part 3 (1997). Besides some clarifications and implementation hints it contains optional enhancements especially for complex meters. Due to technical progress some variants (Fixed format and mode 2=high byte first) are no longer recommended for new developments but are still included as a reference.

Note that this standard contains only directions how data should be coded. It is beyond the task of an application layer standard to define which data must be transmitted under what conditions by which types of slaves or which data transmitted to a slave must have which reactions. Therefore adherence to this standard guarantees the coexistance and common communication and readout capability of slaves via a universal master software (covering all optional features), but not yet functional or communication interchangeabilty of meters following this standard. For several meter types and meter classes the company „Fernwärme Wien“ and the „AGFW“-group of remote heating users have provided such application descriptions required for full interchangeability. They are accessible via the www-server of the m-bus users group (http://www.m-bus.com).

2. CI-Field

2.1 Overview~~Master to slave~~

The original EN1434-3 defined two possible data sequences in multibyte records. The bit two (value 4), which is called M bit or Mode bit, in the CI field gives an information about the used byte sequence in multibyte data structures. If the Mode bit is not set (Mode 1), the least significant byte of a multibyte record is transmitted first, otherwise (Mode 2) the most significant byte. The Mode 2 (M=1) is obsolete and should not be used. It is documented here only as a reference for universal master software with support of old meters.

Mode 1	(Mode 2)	Application
00h-4Fh		reserved for DLMS
50h		application reset
51h	(55h)	data send (master to slave)
52h	(56h)	selection of slaves
53h		reserved
54h-58h		reserved for DLMS
59h-5Bh		reserved
5Ch		synchronize action
60h-6Fh		reserved
70h~~-7Fh~~	(74h)	~~reserved for answer directions~~slave to master: report of application errors
71h		slave to master: report of alarms
72h	(76h)	slave to master: Variable format data respond
73h	(77h)	slave to master: Fixed format data respond
78~~80~~h-8Fh		reserved
90h-97h		manufacturer specific (obsolete)
A0h-AFh		manufacturer specific
B0-B7h		manufacturer specific (obsolete)
B8h		set baudrate to 300 baud
B9h		set baudrate to 600 baud
BAh		set baudrate to 1200 baud
BBh		set baudrate to 2400 baud
BCh		set baudrate to 4800 baud
BDh		set baudrate to 9600 baud
BEh		set baudrate to 19200 baud
BFh		set baudrate to 38400 baud
C0h-FFh		reserved

Table 1 *CI-Field codes used by the master*

„Reserved for DLMS“ refers to the DLMS PDU´s in A-XDR encoding of the FDIS version of the IEC-DLMS standard. Choosing CI-fields different from these code simplifies the construction of slaves which can have dual application layers for both standards and which then can automatically distinguish between communication according to both standards.

Note that the CI-codes $50, $52/$56, $5C, $70/$74, $71, $A0-$AF and $B8-$BF are optional compatible enhancements of the original standard. Note also that even if the functions of these optional CI-codes are not implemented in a slave the link layer protocol requires a proper link layer acknowledge of SND_UD telegrams containing any of these CI-codes.

2.1.2 Application reset (CI = $50), (optional)

With the CI-Code $50 the master can release a reset of the application layer in the slaves. Each slave himself decides which parameters to change - e.g. which data output is default - after it has received such an application reset. This application reset by a SND_UD with CI=$50 is the counterpart to the reset of the data link layer by a SND_NKE.

2.2.1.3 Application reset subcode (optional)

It is allowed to use optional parameters after CI = $50. If more bytes follow, the first byte is the application reset subcode. Further bytes are ignored. The application reset subcode defines which telegram function and which subtelegram is requested by the master. The datatype of this parameter is 8 bit binary. The upper 4 bits define the telegram type or telegram application and the lower 4 bits define the number of the subtelegram. The lower four bits may me ignored for slaves which provide only a single telegram for each application. The use of the value zero for the number of the subtelegram means that all telegrams are requested.
Slaves with only one type of telegram may ignore application reset and the added parameters but have to confirm it ($E5).

The following codes can be used for the upper 4 bits of the first parameter:

Coding	Description	Examples
0000b	All	
0001b	User data	consumption
0010b	Simple billing	actual and fixed date values+dates
0011b	Enhanced billing	historic values
0100b	Multi tariff billing	
0101b	Instaneous values	for regulation
0110b	Load management values for management	
0111b	Reserved	
1000b	Installation and startup	bus adress, fixed dates
1001b	Testing	high resolution values
1010b	Calibration	
1011b	Manufacturing	
1100b	Development	
1101b	Selftest	

1110b	Reserved	
1111b	Reserved	

Table 2 *Coding of the upper four bits of the first parameter after CI = $50*

Note that this table has been expanded with optional elements from the original standard.

2.3 Master to slave data send (51h/55h) (optional)

The CI-Field codes 51h(55h) are used to indicate the data send from master to slave:

Variable Data Blocks (Records)	MDH(opt)	Opt.Mfg.specific data
variable number	1 Byte	variable number

Fig. 1 *Variable Data Structure master to slave*

Note that this structure is identical to the slave to master direction (see chapter 4) with the exception of the fixed header which is omitted in this direction.

2.4 Slave select (52h/56h) (optional)

The CI-Field codes 52h(56h) are used for the management of an optional netwerk layer using secondary adressing (See chapter 9).

2.5~~1.4~~ Synchronize action (CI = $5C) (optional)

This CI-code can be used for synchronizing functions in slaves and masters (e.g. clock synchronization). Special actions or parameter loads may be prepared but their final execution is delayed until the reception of such a special CI-field command.

~~2.2 Slave to master (answer) codes~~

~~The following codes can be used for the direction slave to master:~~

~~CI M=0~~	~~CI M=1~~	~~Application~~
~~70h~~	~~(74h)~~	~~report of general application errors~~
~~71h~~		~~report of alarm status (See Appendix)~~
~~72h~~	~~(76h)~~	~~variable data respond~~
~~73h~~	~~(77h)~~	~~fixed data respond~~

~~**Table 3** *CI-Field codes used by the slave*~~

The use of these control information codes is described in the chapters 3 (fixed data respond), 4 (variable data respond), 6.6 (report of general application errors) and 8.1 (report of alarm status).

2.6 Report of application errors (slave to master) (CI = $70/74) (optional)

3 Fixed Data Structure

For details of the report of application errors see chapter 6.

2.7 Report of alarm status (slave to master) (CI = $71) (optional)

For details of the report of alarm status errors see appendix C.

2.8 Fixed and variable data respond (slave to master) (CI = $72/$76 and $73/$77)

In the reply direction (slave to master) with a long frame originally two different data structures were used. The fixed data structure, besides a fixed length, is limited to the transmission of only two counter states of a predetermined length, which have binary or BCD coding. In contrast the variable data structure allows the transmission of more counter states in various codes and further useful information about the data. The number of bytes of the transmitted counter states is also variable with this data structure. Contrary to the fixed structure, the variable structure can also be used in calling direction. For thesеis reasons the fixed data structure is not recommended for future developments. For information on this obsolete fixed data structure see appendix D. For new development therefore only the CI-field $72 shall be used. This is a restriction from the original standard.

To identify the fixed data structure, the numbers 73h/77h for the control information field are used. In this way a universal master software can see how it must interpret the data. For further details on the structure of telegrams starting with CI-values of $72/($76) see chapter 3.

2.9 Baudrate switch commands $B8-$BF (optional)

These optional commands can be used by a master to switch the baudrate of a slave. For details see chapter 9.1.

3 Data Header of variable data respond 4 Variable Data Structure

4.1 Slave to master (answer) direction

The CI-Field codes 72h/(76h) are used to indicate the variable data structure in long frames (RSP_UD). Figure 2 1 shows the way this data is represented:

Data Header(Req.)	Variable Data Blocks (Records)	MDH(opt)	Opt.Mfg.specific data
12 Byte	variable number	1 Byte	variable number

Fig. 12 *Variable Data Structure in Answer Direction*

3.14.1.1 Structure of Data Header

The first twelve bytes of the user data consist of a block with a fixed length and structure (see fig. 32).

Ident. Nr.	Manufr.	Version	Device	Access No.	Status	Signature
4 Byte	2 Byte	1 Byte	1 Byte	1 Byte	1 Byte	2 Byte

Fig. 23 *Fixed Data Header Block*

3.24.1.2 Identification number

The **Identification Number** is either a fixed fabrication number or a customer number changable by the customer, coded with 8 BCD packed digits (4 Byte), and which thus runs from 00000000 to 99999999. It can be preset at fabrication time with a unique number, but could be changeable afterwards, especially if in addition an unique and not changeable fabrication number (DIF = $0C, VIF = $78, see chapter 6.7.3) is provided.

34.1.3 Manufacturer identification

The field **manufacturer** is coded unsigned binary with 2 bytes. This manufacturer ID is calculated from the ASCII code of EN 61107 manufacturer ID (three uppercase letters) with the following formula:

$$\begin{aligned} \text{IEC 870 Man. ID} = {} & [\text{ASCII(1st letter)} - 64] \cdot 32 \cdot 32 \\ & + [\text{ASCII(2nd letter)} - 64] \cdot 32 \\ & + [\text{ASCII(3rd letter)} - 64] \end{aligned}$$

Note that currently the flag association administers these three letter manufacturers ID of EN61107. For details see appendix X.

34.1.4 Version identification

The field **version** specifies the generation or version of the meter and depends on the manufacturer. It can be used to make sure, that within each version number the identification # is unique.

34.1.5 Medium Device type identification

The medium device byte is coded as follows:

Device type (previously called medium)Medium	Code bin. Bit 7 .. 0	Code hex.
Other	0000 0000	00
Oil	0000 0001	01
Electricity	0000 0010	02
Gas	0000 0011	03
Heat (Volume measured at return temperature: outlet)	0000 0100	04
Steam	0000 0101	05
WarmHot Water (30°C-90°C)	0000 0110	06
Water	0000 0111	07
Heat Cost Allocator.	0000 1000	08
Compressed Air	0000 1001	09
Cooling load meter (Volume measured at return temperature: outlet)	0000 1010	0A
Cooling load meter (Volume measured at flow temperature: inlet)	0000 1011	0B
Heat (Volume measured at flow temperature: inlet)	0000 1100	0C
Heat / Cooling load meter	0000 1101	OD
Bus / System component	0000 1110	0E
Unknown Medium	0000 1111	0F
Reserved	.....	10 to 145
Hot water (>=90°C)	0001 0101	15
Cold Water	0001 0110	16
Dual Water	0001 0111	17
Pressure	0001 1000	18
A/D Converter	0001 1001	19
Reserved	.....	20 to FF

Table 3 Device type identification

Note that this table has been expanded with optional elements from the original standard.

3.4.1.6 Access number

The **Access Number** has unsigned binary coding, and is increased (modulo 256) by one before or after each RSP_UD from the slave. Since it can also be used to enable private end users to detect an unwanted overfrequently readout of its consumption meters, it should not be resettable by any bus communication.

34.1.7 Status byte

Bit	Meaning with Bit set	Significance with Bit not set
0,1	See table 5Fig.4	See table 5Fig.4
2	Power low	Not power low
3	Permanent error	No permanent error
4	Temporary error	No temporary error
5	Specific to manufacturer	Specific to manufacturer
6	Specific to manufacturer	Specific to manufacturer
7	Specific to manufacturer	Specific to manufacturer

Table 4 Fig. 3 *Coding of the Status Field*

Status bit 1 bit 0	Application status
0 0	No Error
0 1	Application Busy
1 0	Any Application Error
1 1	Reserved

Table 5 Fig. 4 *Application Errors coded with the Status-Field*

Note that more detailed error signalling can be provided by application telegrams starting with CI=$70 and/or using data records signalling even more detailed error information.

3.8 Signature field

The **Signature** is reserved for optional encryptation of the application data. If no encryptation is used its value shall be 00 00 h.

3.8.1 Functions

Data privacy for consumption meters values

Detecting simulated meter transmission
Preventing later playback of old meter values

3.8.2 Structure of encrypted telegrams

a) The first 12-byte block containing the ID-number, the manufacturer etc. is always unencrypted. The last word of this block is the signature word. If the following data are unencrypted, this signature word contains a zero.

b) If the transmission contains encrypted data, the high byte of this signature word contains a code for the encryptation method. The code 0 signals no encryptation. Currently only the encryptation codes 2 or 3 (see below) are defined. The other codes are reserved. The number of encrypted bytes is contained in the low byte of the signature word. The content of this signature word is currently defined as zero, corresponding consistently to no encrypted data.

c) The encrypted data follow directly after the signature word, thus forming the beginning of the DIF/VIF-structured part of the telegram.

3.8.3 Partial Encryption

a) If the number of encrypted bytes is less than the remaining data of the telegram, unencrypted data may follow after the encrypted data. They must start at a record boundary, i.e. the first byte after the encrypted data will be interpreted as a DIF.

b) If a partially encrypted telegram must contain encrypted manufacturer specific data a record with a suitable length DIF (possibly a variable length string DIF) and a VIF= $7F (manufacturer specific data record) must be used instead of the usual MDH-DIF=$0F. This is required to enable after decryptation standard DIF/VIF-decoding of a previously partially encrypted telegram containing encrypted manufacturer specific data .

3.8.4 Encryptation methods

a) Encryptation according to the DES (data encryptation standard) as described in ANSI X3.92-1981

b) Cipher Block Chaining (CBC)-method as described in ANSI X3.106-1983 with an initial initialization vector of zero: (Encryptation Method Code=2). In this case the data records should contain the current date before the meter reading.
Note that in this case the data after the date record, i.e.especially the encrypted meter reading data change once per day even if their data content itself is constant. This prevents an undetectable later playback of stored encrypted meter readings by a hacker.

c) The "Initialization Vector IV" with length 64 bits of this standard may alternatively be defined by the the first 6 bytes of the identification header in mode 1 sequence, i.e. identification number in in the lowest 4 bytes followed by the manufacturer ID in the two next higher bytes and finally by the current date coded as in record structure "G" for the two highest bytes.
In this case the encryptation method is coded as "3".
Note that in this case all encrypted data change once per day even if the data content itself is constant.
This prevents an undetectable later playback of any stored encrypted data by a hacker.

d) To simplify the verification of correct decoding and to prevent an undetected change in the identification of the not encrypted header, the encrypted part of the telegram must contain at least together with the appropriate application layer coding (DIF and VIF) again the same identification number as in the unencrypted header.

e) Due to the mathematical nature of the DES-algorithm the encrypted length contained in the low byte of the signature word must be an integer multiple of 8 if the high byte signals DES-encryptation. Unused bytes in the last 8-byte block must be filled with appropriatly structured dummy data records to achieve the required record boundary at the end of the encrypted data. One or several bytes containing the filler DIF=$2F are suggested to fill such gaps.

f) The application of certain encryptation methods might be prohibited by local laws.

4.1.8 Signature field

The **Signature** remains reserved for future encryptation applications, and until then is allocated the value 00 00 h.

Variable Data Blocks (Records)	MDH(opt)	Opt.Mfg.specific data
variable number	1 Byte	variable number

Fig. 5 *Variable Data Structure in Answer Direction*

Note that this structure is identical to the slave to master direction with the exception of the fixed header which is omitted in this direction.

4.3 Variable Data Blocks (Records)

The data, together with information regarding coding, length and the type of data is transmitted in data records in arbitrary sequence. As many records can be transferred as there is room for within the maximum total data length of 234255 Bytes, and taking account of the C, A, and CI fields, and the data header. This limits the total telegram length to 255 bytes. This restriction is required to enable gateways to other link- and application layers. e upper limit for characters in the variable data blocks is thus 240 byte. A maximum total telegram length of 255 bytes (234 bytes for variable data blocks) is recommended to simplify gateways and the integration in other link layers . The manufacturer data header (MDH) is made up by

the character 0Fh or 1Fh and indicates the beginning of the manufacturer specific part of the user data and should be omitted, if there are no manufacturer specific data.

DIF	DIFE	VIF	VIFE	Data
1 Byte	0-10 (1 Byte each)	1 Byte	0-10 (1 Byte each)	0-N Byte
Data Information Block DIB		Value Information Block VIB		
Data Record Header DRH				

Fig. 46 *Structure of a Data Record (transmitted from left to right)*

Each data record contains one value (data) with its description (DRH)~~as shown in figure 20, a data record, which consists of a data record header (DRH) and the actual data~~. The DRH in turn consists of the DIB (data information block) to describe the length, type and coding of the data, and the VIB (value information block) to give the value of the unit and the multiplier.

4.3.1 Data Information Block (DIB)

The DIB contains at least one byte (DIF, data information field), and can be extended by a maximum of ten DIFE's (data information field extensions).

4.3.2 Data Information Field (DIF)

The following information is contained in a DIF:

Bit 7	6	5	4	3	2	1	0
Extension Bit	LSB of storage number	Function Field		Data Field : Length and coding of data			

Fig. 57 *Coding of the Data Information Field (DIF)*

4.3.3 Data Field

The **data field** shows how the data from the master must be interpreted in respect of length and coding. The following table contains the possible coding of the data field:

Length in Bit	Code	Meaning	Code	Meaning
0	0000	No data	1000	Selection for Readout
8	0001	8 Bit Integer	1001	2 digit BCD
16	0010	16 Bit Integer	1010	4 digit BCD
24	0011	24 Bit Integer	1011	6 digit BCD

32	0100	32 Bit Integer	1100	8 digit BCD
32 / N	0101	32 Bit Real	1101	variable length
48	0110	48 Bit Integer	1110	12 digit BCD
64	0111	64 Bit Integer	1111	Special Functions

Table 46 *Coding of the data field*

Note that this table has been expanded with optional elements from the original standard.

For a detailed description of data types refer to appendix A8.2 "Coding of data records" (e.g. BCD = Type A, Integer = Type B, Real = Type H).

Variable Length:

With data field = `1101b` several data types with variable length can be used. The length of the data is given after the DRH with the first byte of real data, which is here called LVAR (e.g. LVAR = 02h: ASCII string with two characters follows).

LVAR = 00h .. BFh	:	Text string according to ISI/IEC 646 with LVAR characters
LVAR = C0h .. C9h	:	positive BCD number with (LVAR - C0h) • 2 digits
LVAR = D0h .. D9H	:	negative BCD number with (LVAR - D0h) • 2 digits
LVAR = F8h	:	floating point number according to IEEE 754
Others LVAR values	:	Reserved

Like all multibyte fields in mode 1 the last character and in mode 2 the first character is transmitted first.

Special Functions (data field = 1111b):

DIF	Function
0Fh	Start of manufacturer specific data structures to end of user data
1Fh	Same meaning as DIF = 0Fh + More records follow in next telegram
2Fh	Idle Filler (not to be interpreted), following byte = DIF of next record
3Fh..6Fh	Reserved
7Fh	Global readout request (all storage#, units, tariffs, function fields)

Table 7: DIF-coding for special functions

Note that this table has been expanded with optional elements from the original standard.

If data follows after DIF=$0F or $1F these are manufacturer specific unstructured data. The number of bytes in these manufacturer specific data can be calculated from the link layer information on the total length of the application layer telegram. The DIF 1Fh signals a request from the slave to the master to readout the slave once again. The master must readout the slave until there is no DIF=1Fh inside the respond telegram (multi telegram readout) or use an application reset.

4.3.4 Function field

The **function field** gives the type of data as follows:

Code	Description	Code	Description
00b	Instantaneous value	01b	Maximum value
10b	Minimum value	11b	Value during error state

Table 8: Function Field

4.3.5 Storage number

The Bit 6 of the DIF serves as the LSB of the **storage number** of the data concerned, and the slave can in this way indicate and transmit various stored metering values or historical values of metering data. This bit is the least significant bit of the storage number and allows therefore the storage numbers 0 and 1 to be coded. If storage numbers higher than „1“ are needed, following (optional) DIFE´s contain the higher bits. The storage number 0 signals an actual value. Note that a each storage number is associated with a given timepoint. So all data records with the same storage number refer to the value of the associated variable at this (common) timepoint for this storage number. It is recommended, that a time/date record with this storage number is included somewhere to signal this timepoint. Normally (but not necessarily) higher storage numbers indicate an older timepoint. A sequential block of storage numbers can be associated with a sequence of equidistantly spaced time points (profile). Such a block can be described by its starting time, by the time spacing, by the first storage number of such a block and by the length of such a block. The coding for such a block description in contrast to an individual time/date record for each individula storage number is given in the appendix.

4.3.6 Extension Bit

The extension bit (MSB) signals that more detailed or extended descriptions (data field extension=DIFE)-bytes follow.

4.3.7 Data field extension byte(s) (DIFE)

Each DIFE (maximum ten) contains again an extension bit to show whether a further DIFE is being sent. Besides giving the next most significant bits of the storage number, DIFE´s allow the transmission of informations about the **tariff** and the **subunit** of the device ~~from which the data come~~. In this way, exactly as with the storage number, the next most significant bit or bits will be transmitted. The figure 8 which follows shows the structure of a DIFE:

Bit 7	6	5	4	3	2	1	0
Extension Bit	(Device) Unit	Tariff		Storage Number			

Fig. 58 *Coding of the Data Information Field Extension (DIFE)*

With the maximum of ten DIFE´s which are provided, there are 41 bits for the storage number, 20 bits for the tariff, and 10 bits for the subunit of the meter. There is no application conceivable in which this immense number of bits could all be used.

4.3.8 Tariff information

For each (unique) value type designation given by the following value information block (VIB) at each unique time point (given by the storage number) of each unique function (given by the function field) there might exist still various different data, measured or accumulated under different conditions. Such conditions could be time of day, various value ranges of the variable (i.e. separate storage of positive accumulatad values and negative accumulated values) itself or of other signals or variables or various averaging durations. Such variables which could not be distinguished otherwise are made different by assigning them different values of the tariff variable in their data information block. Note that this includes but is not necessarily restricted to various tariffs in a monetary sense. It is at the distinction of the manufacturer to describe for each tariff (except 0) what is different for each tariff number. Again as with the storage numbers all variables with the same tariff information share the same tariff associating condition.

4.93.8 Subunit information

A slave component may consist of several functionally and logically independent subunits of the same or of different functionallity. Such a device may either use several different primary and/or secondary adresses. Such it is from a link layer and an application layer view just several independent devices which share a common physical layer interface. This is recommended for devices which represent a physical collection of several truely independent (often similar or idential) devices. For devices which share common information and values and have logical connections an approach with a common link layer (i.e.a single address) is reccomended. The various subunits can include their specific information into a common telegram and have them differentiated by the individual subunit number in the subunit-datafield of their records.

54.4 Value Information Block (VIB)

After a DIF (with the the exception of $xF) or a DIFE without a set extension bit there follows the VIB (value information block). This consists at least of the VIF (value information field) and can be expanded with a maximum of 10 extensions (VIFE). The VIF and also the VIFE's show with a set MSB that a VIFE will follow. In the value information field VIF the other seven bits give the unit and the multiplier of the transmitted value.

Bit 7	6	5	4	3	2	1	0
Extension Bit	Unit and multiplier (value)						

Fig. 69 *Coding of the Value Information Field (VIF)*

There are five types of coding depending on the VIF:

a)1. Primary VIF: E000 0000b .. E111 1011b

The unit and multiplier is taken from the table for primary VIF (Table 9chapter 8.4.3).

b)2. Plain-text VIF: E111 1100b

In case of VIF = 7Ch / FCh the true VIF is represented by the following ASCII string with the length given in the first byte. Please note that the byte order of the characters after the length byte depends on the used byte sequence. Since only the „LSB first mode“ (M=1) of multibyte data transmission is recommended, the rightmost character is transmitted first. This plain text VIF allows the user to code units that are not included in the VIF tables..

c)3. Linear VIF-Extension: FDh and FBh

In case of VIF = FDh and VIF = FBh the true VIF is given by the next byte (i.e. the first VIFE) and the coding is taken from the tables 11 respectively table 12 for secondary VIF (chapter 8.4.4). This extends the available VIF´s by another 256256 codes.

d)4. Any VIF: 7Eh / FEh

This VIF-Code can be used in direction master to slave for readout selection of all VIF´s. See chapter 6.4.3.

e)5. Manufacturer specific: 7Fh / FFh

In this case the remainder of this data record including VIFE´s has manufacturer specific coding.

5.1 Primary VIF´s (main table)

The first section of the main table contains integral values, the second typically averaged values, the third typically instantaneous values and the fourth block contains parameters (E: extension bit).

4.4.1 Primary VIF´s (main table)

Coding	Description	Range Coding	Range
E000 0nnn	Energy	$10^{(nnn-3)}$ Wh	0.001Wh to 10000Wh
E000 1nnn	Energy	$10^{(nnn)}$ J	0.001kJ to 10000kJ
E001 0nnn	Volume	$10^{(nnn-6)}$ m^3	0.001l to 10000l
E001 1nnn	Mass	$10^{(nnn-3)}$ kg	0.001kg to 10000kg
E010 00nn	On Time	nn = 00 seconds nn = 01 minutes nn = 10 hours nn = 11 days	
E010 01nn	Operating Time	coded like OnTime	
E010 1nnn	Power	$10^{(nnn-3)}$ W	0.001W to 10000W
E011 0nnn	Power	$10^{(nnn)}$ J/h	0.001kJ/h to 10000kJ/h
E011 1nnn	Volume Flow	$10^{(nnn-6)}$ m^3/h	0.001l/h to 10000l/h
E100 0nnn	Volume Flow ext.	$10^{(nnn-7)}$ m^3/min	0.0001l/min to 1000l/min
E100 1nnn	Volume Flow ext.	$10^{(nnn-9)}$ m^3/s	0.001ml/s to 10000ml/s
E101 0nnn	Mass flow	$10^{(nnn-3)}$ kg/h	0.001kg/h to 10000kg/h
E101 10nn	Flow Temperature	$10^{(nn-3)}$ °C	0.001°C to 1°C
E101 11nn	Return Temperature	$10^{(nn-3)}$ °C	0.001°C to 1°C
E110 00nn	Temperature Difference	$10^{(nn-3)}$ K	1mK to 1000mK
E110 01nn	External Temperature	$10^{(nn-3)}$ °C	0.001°C to 1°C
E110 10nn	Pressure	$10^{(nn-3)}$ bar	1mbar to 1000mbar
E110 110n	Time Point	n = 0 date n = 1 time & date	data type G data type F
E110 1110	Units for H.C.A.		dimensionless
E110 1111	Reserved		
E111 00nn	Averaging Duration	coded like OnTime	
E111 01nn	Actuality Duration	coded like OnTime	
E111 1000	Fabrication No		
E111 1001	(Enhanced)		see appendix E2~~chapter~~
E111 1010	Bus Address		data type C (x=8)

Table 9: Primary VIF-codes

Note that this table has been expanded with optional elements from the original standard.

54.4.2 VIF-Codes for special purposes:

Coding	Description	Purpose
1111 1011	Extension of VIF-codes	true VIF is given in the first VIFE and is coded using table 108.4.4 b) (128 new VIF-Codes)
E111 1100	VIF in following string (length in first byte)	allows user definable VIF´s (in plain ASCII-String) *
1111 1101	Extension of VIF-codes	true VIF is given in the first VIFE and is coded using table 118.4.4 a) (128 new VIF-Codes)
E111 1110	Any VIF	used for readout selection of all VIF´s (see chapter 9.26.4.3)
E111 1111	Manufacturer Specific	VIFE´s and data of this block are manufacturer specific

Table 10: Special VIF-Codes

Note that this table has been expanded with optional elements from the original standard.

Note:

* Coding the VIF in an ASCII-String in combination with the data in an ASCII-String (datafield in DIF = 1101 b) allows the representation of data in a free user defined form.

54.4.3 MainFirst VIFE-Code Extension table (following VIF=$FD for primary VIF)

Coding	Description	Group
E000 00nn	Credit of 10^{nn-3} of the nominal local legal currency units	Currency Units
E000 01nn	Debit of 10^{nn-3} of the nominal local legal currency units	
E000 1000	Access Number (transmission count)	Enhanced Identification
E000 1001	Device typeMedium (as in fixed header)	
E000 1010	Manufacturer (as in fixed header)	
E000 1011	Parameter set identification	
E000 1100	Model / Version	
E000 1101	Hardware version #	

E000 1110	Firmware version #	
E000 1111	Software version #	
E001 0000	Customer location	Implementation of all TC294 WG1 requirements (improved selection ..)
E001 0001	Customer	
E001 0010	Access Code User	
E001 0011	Access Code Operator	
E001 0100	Access Code System Operator	
E001 0101	Access Code Developer	
E001 0110	Password	
E001 0111	Error flags (binary) (Device type specific)	
E001 1000	Error mask	
E001 1001	Reserved	
E001 1010	Digital Output (binary)	
E001 1011	Digital Input (binary)	
E001 1100	Baudrate [Baud]	
E001 1101	response delay time [bittimes]	
E001 1110	Retry	
E001 1111	Reserved	

E010 0000	First storage # for cyclic storage	Enhanced storage management
E010 0001	Last storage # for cyclic storage	
E010 0010	Size of storage block	
E010 0011	Reserved	
E010 01nn	Storage interval [sec(s)..day(s)]	
E010 1000	Storage interval month(s)	
E010 1001	Storage interval year(s)	
E010 1010	Reserved	
E010 1011	Reserved	
E010 11nn	Duration since last readout [sec(s)..day(s)]	
E011 0000	Start (date/time) of tariff	Enhanced tariff management
E011 00nn	Duration of tariff (nn=01 ..11: min to days)	
E011 01nn	Period of tariff [sec(s) to day(s)]	
E011 1000	Period of tariff months(s)	
E011 1001	Period of tariff year(s)	
E011 1010	dimensionless / no VIF	
E011 1011	Reserved	

E011 11xx	Reserved	
E100 nnnn	10^{nnnn-9} Volts	electrical units
E101 nnnn	$10^{nnnn-12}$ A	

E110 0000	Reset counter	
E110 0001	Cumulation counter	
E110 0010	Control signal	
E110 0011	Day of week	
E110 0100	Week number	
E110 0101	Time point of day change	
E110 0110	State of parameter activation	
E110 0111	Special supplier information	
E110 10pp	Duration since last cumulation [hour(s)..years(s)]	
E110 11pp	Operating time battery [hour(s)..years(s)]	
E111 0000	Date and time of battery change	
E111 0001 to E111 1111	Reserved	

Table 11: Main VIFE-code extension table

Note that this optional table has been added to the original standard.

Notes:

nn = 00 second(s)
01 minute(s)
10 hour(s)
11 day(s)

The information about usage of data type F (date and time) or data type G (date) can be derived from the datafield (0010b: type G / 0100: type F).

pp = 00 hour(s)
01 day(s)
10 month(s)
11 year(s)

5~~4~~.4 Alternate VIFE-Code Extension table (following VIF=$FB for primary VIF)

Coding	Description	Range Coding	Range
E000 000n	Energy	$10^{(n-1)}$ MWh	0.1MWh to 1MWh
E000 001n	Reserved		
E000 01nn	Reserved		
E000 100n	Energy	$10^{(n-1)}$ GJ	0.1GJ to 1GJ
E000 101n	Reserved		
E000 11nn	Reserved		
E001 000n	Volume	$10^{(n+2)}$ m^3	
E001 001n	Reserved		
E001 01nn	Reserved		
E001 100n	Mass	$10^{(n+2)}$ t	100t to 1000t
E001 1010-E010 0000	Reserved		
E010 0001	Volume	0,1 feet^3	
E010 0010	Volume	0,1 am.gallon	
E010 0011	Volume	1 am.gallon	
E010 0100	Volume flow	0,001 am.gallon/min	
E010 0101	Volume flow	1 am.gallon/min	
E010 0110	Volume flow	1 am.gallon/h	
E010 0111	Reserved		
E010 100n	Power	$10^{(n-1)}$ MW	0.1MW to 1MW
E010 101n	Reserved		
E010 11nn	Reserved		
E011 000n	Power	$10^{(n-1)}$ GJ/h	0.1GJ/hto 1GJ/h
E011 0010-E101 0111	Reserved		
E101 10nn	Flow Temperature	$10^{(nn-3)}$ °F	0.001°F to 1°F
E101 11nn	Return Temperature	$10^{(nn-3)}$ °F	0.001°F to 1°F
E110 00nn	Temperature Differ.	$10^{(nn-3)}$ °F	0.001°F to 1°F
E110 01nn	Flow Temperature	$10^{(nn-3)}$ °F	0.001°F to 1°F
E110 1nnn	Reserved		
E111 00nn	Cold/Warm Temp. Lim.	$10^{(nn-3)}$ °F	0.001°F to 1°F
E111 01nn	Cold/Warm Temp. Lim.	$10^{(nn-3)}$ °C	0.001°C to 1°C
E111 1nnn	Cum.Count Max.power	$10^{(nnn-3)}$ W	0.001W to 10000W

Table 12: Alternate extended VIF-code Table

Note that this optional table has been added to the original standard.

54.4.5 Combinable (Orthogonal) VIFE-Code Extension table (Following primary VIF)

VIFE-Code	Description
E00x xxxx	Reserved for object actions (master to slave): see chapter 6.3 and table 16table on page 75 or for error codes (slave to master): see chapter 7 and table 17table on page 74
E010 0000	per second
E010 0001	per minute
E010 0010	per hour
E010 0011	per day
E010 0100	per week
E010 0101	per month
E010 0110	per year
E010 0111	per revolution / measurement
E010 100p	increment per input pulse on input channel #p
E010 101p	increment per output pulse on output channel #p
E010 1100	per liter
E010 1101	per m^3
E010 1110	per kg
E010 1111	per K (Kelvin)
E011 0000	per kWh
E011 0001	per GJ
E011 0010	per kW
E011 0011	per (K*l) (Kelvin*liter)
E011 0100	per V (Volt)
E011 0101	per A (Ampere)
E011 0110	multiplied by sek
E011 0111	multiplied by sek / V
E011 1000	multiplied by sek / A
E011 1001	start date(/time) of
E011 1010	VIF contains uncorrected unit instead of corrected unit
E011 1011	Accumulation only if positive contributions
E011 1100	Accumulation of abs value only if negative contributions
E011 1101 to E011 1111	Reserved

VIFE-Code	Description
E100 u000	u=1: upper, u=0: lower limit value
E100 u001	# of exceeds of lower u=0) / upper (U=1) limit
E100 uf1b	Date (/time) of: b=0: begin, b=1: end of, f=0: first, f=1: last, u=0: lower, u=1: upper limit exceed
E101 ufnn	Duration of limit exceed (u,f: as above, nn=duration)
E110 0fnn	Duration of (f: as above, nn=duration)
E110 1x0x	Reserved
E110 1f1b	Date (/time) of (f,b: as above)
E111 0nnn	Multiplicative correction factor: 10^{nnn-6}
E111 10nn	Additive correction constant: 10^{nn-3} • unit of VIF (offset)
E111 1100	Reserved
E111 1101	Multiplicative correction factor for value (not unit): 10^3
E111 1110	future value
E111 1111	next VIFE's and data of this block are maufacturer specific

Table 13: Combinable (orthogonal) VIFE-Table

Note that this optional table has been added to the original standard.

Notes:

"Date(/time) of" or "Duration of" relates to the information which the whole data record header contains.

The information about usage of data type F (date and time) or data type G (date) can be derived from the datafield (0010b: type G / 0100: type F).

65 Application Layer Status and error reporting

The data link layer reports only communication errors by means of omittingleaving out the acknowledgement $E5 or a via a negative acknowledgement. It is not allowed to report errors of the application layer (which can occur for example in data writing) via the link layer. The slave can transmit an $E5 after a SND_UD to indicate that it has received the telegram, but can´t respond with data. There are three different techniques for reporting application errors:

65.1 Status Field

One possible solution is to use the reserved 2 lowest bits of the Status field in the variable data structure for the application layer status (see Table 6).:

Fig. 5 *Application Errors coded with the Status-Field*

65.2 General Application Layer Errors

For reporting general application errors a slave can use a RSP_UD telegram with CI=$70 and zero, one or several data bytes, which then describes the type of error:

68h	04h	04h	68h	08h	PAdr	70h	DATA	CS	16h

Fig. 1 *Telegram for reporting general application errors*

The following values for DATA are defined:

0	Unspecified error: also if data field is missing
1	Unimplemented CI-Field
2	Buffer too long, truncated
3	Too many records
4	Premature end of record
5	More than 10 DIFE´s
6	More than 10 VIFE´s
7	Reserved
8	Application too busy for handling readout request
9	Too many readouts (for slaves with limited readouts per time)
10..255	Reserved

Table 5 15 *Codes for general application errors*

Note that this optional table has been added to the original standard.

65.3 Record Errors

To report errors belonging to a special record the slave can use this data record header with a VIFE containing one of the following values to code the type of application error, which has been occured.

VIFE-Code	Type of Record Error	Error Group
E000 0000	None	DIF Errors
E000 0001	Too many DIFE´s	
E000 0010	Storage number not implemented	
E000 0011	Unit number not implemented	
E000 0100	Tariff number not implemented	
E000 0101	Function not implemented	
E000 0110	Data class not implemented	
E000 0111	Data size not implemented	
E000 1000 to E000 1010	Reserved	
E000 1011	Too many VIFE´s	VIF Errors
E000 1100	Illegal VIF-Group	
E000 1101	Illegal VIF-Exponent	
E000 1110	VIF/DIF mismatch	
E000 1111	Unimplemented action	
E001 0000 to E001 0100	Reserved	
E001 0101	No data available (undefined value)	Data Errors
E001 0110	Data overflow	
E001 0111	Data underflow	
E001 1000	Data error	
E001 1001 to E001 1011	Reserved	
E001 1100	Premature end of record	Other Errors
E001 1101 to E001 1111	Reserved	

Table 6—16: *Codes for record errors (E = extension bit)*

Note that this optional table has been added to the original standard.

In case of record errors the data maybe invalid. The slave has some options to transmit the data:

- datafield = 0000b: no data
- datafield = 0000b: no data and idle filler (DIF=$2F): fill telegram record up to the normal length
- other datafield: dummy data of correct length
- other datafield: unsafe or estimated data

76 Generalized Object Layer

The fundamental idea of an object is the encapsulation of data and methods or actions for the data. In case of writing data to a slave the master software can pack data and information about the action, which the slave shall do with this data, in one data record. This variable data record with actions is now called an object. Following any VIF including a VIF=$FD or VIF=$FB with the true value information in the first VIFE another (usually the last) VIFE can be added which contains a code signalling object actions according to the following table.
Action: (E: extension bit)

VIFE-Code binary	Action	Explanation
E000 0000	Write (Replace)	replace old with new data
E000 0001	Add Value	add data to old data
E000 0010	Subtract Value	subtract data from old data
E000 0011	OR (Set Bits)	data OR old data
E000 0100	AND	data AND old data
E000 0101	XOR (Toggle Bits)	data XOR old data
E000 0110	AND NOT (Clear Bits)	NOT data AND old data
E000 0111	Clear	set data to zero
E000 1000	Add Entry	create a new data record
E000 1001	Delete Entry	delete an existing data record
E000 1010	Reserved	
E000 1011	Freeze Data	freeze data to storage no.
E000 1100	Add to Readout-List	add data record to RSP_UD
E000 1101	Delete from Readout-List	delete data record from RSP_UD
E000 111x	Reserved	
E001 xxxx	Reserved	

~~*Fig. 7*~~ ***Table 17:*** Action Codes for the Generalized Object layer (Master to Slave)
Note that this optional table has been added to the original standard.

Note:
The object action "write / replace" (VIFE = E000 0000) is the default and is assumed if there is no VIFE with an object action for this record.

8~~7~~ Manufacturer Specific unstructured Data Block

The MDH consists of the character 0Fh or 1Fh (DIF = 0Fh or 1Fh) and indicates that all following data are manufacturer specific. When the total number of bytes given from the link/network layers and the number of record-structured bytes and the length of the fixed header is known, the number of remeining unstructured manufacturer specific bytes can be calculated.
Note that stuctured manufacturer specific data (i.e. those with a known data structure including variable length binary or ASCII but with a manufacturer specific meaning or unit) can be described using normal data records with a value information field of VIF=E1111111b.
In case of MDH = 1Fh the slave signals to the master that it wants to be readout once again (multitelegram readouts). The master must readout the data until there is no MDH = 1Fh in the respond telegram.

9~~8~~ Management of lower layers

Because changing of parameters like baudrate and address by higher layers is not allowed in the ISO-OSI-Model, a **Management Layer** beside and above the seven OSI-Layers is defined:

MANAGEMENT LAYER	
Application Layer	
Presentation Layer	
Session Layer	
Transport Layer	
Network Layer (if address = 253)	Address 253 / Enable Disable CI=$52/$56
Data Link Layer	
Physical Layer	Address 254 (255)/251

Fig. 2 *Management-Layer of the M-Bus*

So the address 254 and perhaps 255 can be used also for managing the physical layer of the bus and the adress 251 is reserved for managing the (primary) M-Bus level converter/bridge and the address 253 (selection) for network layer (see chapter 7), which is only used in certain cases. With such a managment addresses and or CI-fields we can directly manage each OSI-layer to implement features, which are beyond the elementary OSI-Model.

98.1 Switching Baudrate

All slaves must be able to communicate with the master using the minimum transmission speed of 300 baud. , after reception of a break signal and after each bus power fail.Split baudrates between transmit and receive are not allowed, but there can be devices with different baudrates on the bus.
In point to point connections the slave is set to another baudrate by a Control Frame (SND_UD with L-Field = 3) with address FEh and one of the following CI-Field codes: Note that for safety reasons a baudrate switch command to the (unacknowledged) broadcast adress 255 is not recommended.
A complete bus can be switched to another baudrate via the broadcast adress $FF.

CI-Field	B8h	B9h	BAh	BBh	BCh	BDh	Beh	BFh
Baud	300	600	1200	2400	4800	9600	19200	38400
Note		1	1		1		1	1

Fig. 3 *CI-Field-Codes for Baudrate Switching*

Note that this optional figure has been added to the original standard.

Notes:
1) These baudrates are not recommended only by agreement with operator.

The slave always confirms the correctly received telegram by transmitting an E5h with the old baudrate and uses the new baudrate from now on, if he is capable of this. Otherwise the slave stays at ists previous baudrate after the $E5 acknowledge. To make sure that a slave without autospeed detect has properly switched to the new baudrate and that it can communicate properly at the new baudrate in its segment it is required that after a baudrate switch to a baudrate other than 300 Baud the master attempts imediately (<2min) after the baudrate

switch command a communication. If (even after the appropriate number of retries) this is not acknowledged by the slave, the master shall issue a baudrate set command (at the attempted new baudrate) back to the previous baudrate. If a slave without autospeed detect does not receive a valid communication at the new baudrate within 2-10 minutes of the baudrate switch command the slave must fall back to its previous baudrate. This is required individually and sequentially for each adressable slave.

The master must know the highest available baudrate on the bus to forbid the user switching to a transmission speed, which is not available on the bus. Otherwise the slave would never answer again. In this case intelligent primary level converters or bridges (with special address = 251) can generate a break signal on the bus if they receive a SND_UD-telegram with a CI-field of $B0.

For compatibility with older slaves with fallback to 300 baud the master should also attempt a communication at 300 baud if the slave does not answer at its last baudrate.

98.2 Selection and Secondary Addressing (optional network layer)

The network layer takes care of choosing the best transmission route between the communication parties in a network. We define that the network layer in the M-Bus protocol "connects" a slave with a certain secondary address to the bus and and associates it with the primary address of 253 ($FD). So the maximum number of 250 addresses (primary) is extended by the network layer associating a selected slave to this address 253. The network layer is only enabled by a SND_UD with CI_Field $52 / $56 to address 253.

When addressing in the data link layer with the help of the A-Field, the problem of the address allocation could arise. The addresses are normally set to a value of 0 by the manufacturer of the meters, in order to designate them as unconfigured slaves. A very laborious method of address allocation consists of setting the addresses when installing the slaves, for example with DIP switches. A further method of address allocation is to determine the bus addresses when connecting the equipments to the bus with the master software. This sends a command for address allocation (see 6.4.2Appendix E2) to the address 0. In this case the slaves must however all be successively connected to the bus, which very much gets in the way of a simple installation procedure.

When however addressing in the network layer these disadvantages are avoided and the address region is essentially extended beyond the number of 250 with primary addressing (A-Field). The addressing of the slaves takes place with secondary addressing with the help of the following so-called selection:

68h	0Bh	0Bh	68h	53h	FDh	52h	ID1-4	Man 1-2	Gen	Med	CS	16h

Fig. 4 *Structure of a telegram for selecting a slave (mode 1)*

The master sends a SND_UD with the control information 52h (Mode 1) or 56h (Mode 2) to the address 253 (FDh) and fills the specific meter secondary address (identification number, manufacturer, version and device typemedium) with the values of the slave which is to be addressed. After the reception of the address FDh the selection mode is entered. If then the proper CI-selection code (normally CI=52h, i.e. mode 1) is received the internal selection bit is set otherwise it is reset. If further data bytes follow they are compared with the corresponding internal addresses respective values of the meter. If they disagree, the selection bit is cleared otherwise it is left unchanged. Thus „selecting“ a meter with only a proper CI-field and no further data will select all meters on the bus capable of secondary addressing. A set selection bit means that this slave can be addressed (e.g. REQ_UD) with the bus address FDh and in this example will reply with RSP_UD. In other words the network layer has associated this slave with the address FDh.

During selection individual positions of the secondary addresses can be occupied with wildcards (Fh). Such a Wildcard means that this position will not be taken account of during selection, and that the selection will be limited to specific positions, in order to address complete groups of slaves (Multicasting). In the identification number each individual digit can be wildcarded by a wildcard nibble Fh while the fields for manufacturer, version and device typemedium can be wildcarded by a wildcard byte FFh.

The state of the selection remains unchanged until the slave is deselected with a selection command (as described above) with non-matching secondary addresses, or a SND_NKE to address 253. The slave, which uses mode 1 for multibyte records, will be selected by a telegram with the CI-Field 52h and the right secondary address, but it will be deselected by a telegram with the CI-Field 56h and any secondary address.

A sSlave with implemented primary and secondary addressing should also answer telegrams to his primary address. A sSlave with only secondary addressing (i.e. internal primary adress=253) should occupy the address field in the RSP_UD telegram with $FD to signal that it will not participate in primary addressing.

98.3 Generalized Selection Procedure

For including new or restructed identification parameters into a selection procedure an enhanced definition of the selection telegram (CI=$52/$56) can be used:

After the 8 byte of the fixed selection header may also follow standard records with data. In this case only those meters will be selected, where in addition to the fixed header all record data agree. In most but not all cases this means that the DIF and parts of the VIF (not exponent) must match. Again wildcard rules apply to the record data (digit wildcard for BCD-coded data and byte wildcard for binary or string data).

With this generalized selection it will be possible to select slaves using e.g. additional fabrication number, longer identification numbers, customer, customer location and more information. Two useful examples from the primary table for VIF´s are the "Fabrication No." and "Customer name". For inclusion of the fabrication number in the selection process after the field „device type" the 8-digit BCD-fabrication number follow. Parts of the fabrication number (Fab1..Fab4) can be occupied with wildcards (Fh).

If a fabrication number exists the slave should add this data to the variable data blocks in every RSP-UD telegram. If the fabrication number and enhanced selection is not implemented in a slave this device will not confirm the enhanced selection telegram and will be deselected. Enhanced selection should be used only if the normal kind of selection is not successful.

98.4 Searching for Installed Slaves

98.4.1 Primary Addresses

To read out all installed slaves the master software must know all the slaves, which are connected to the bus. Therefore the software searches for slaves with primary adressing by sending a REQ_UD2 to all allowed adresses (1..250) with all available baudrates. The master notes used primary addresses with the respective baudrates.

98.4.21 Secondary Addresses

The secondary addressing described in the preceding section draws attention to the problem of determining the secondary addresses of slaves connected to the bus. The master can after this read out the slaves making use of secondary addresses with previous selection. Testing all possible identification numbers with the master software would take years, since the identification number offers millions of combinations. For this reason, a procedure was developed for the rapid and automatic determination of already installed slaves:

98.4.32 Wildcard searching procedure

The following wildcard searching procedure uses the occupation of individual parts of the secondary address with wildcards (Fh) for selection:

In this case with the identification number (BCD) each individual position, and by manufacturer, version and medium (binary coding), only one complete byte, can be occupied

with wildcards. The master begins the selection using a SND_UD with the control information 52h (Mode 1), and occupies all positions in the identification number, except the top one, with wildcards. The top position is run through in ten selections from 0 to 9 (0FFFFFFF to 9FFFFFFF).

If after such a selection the master receives no acknowledgement, it then goes to the next selection. If the master receives an E5h, it then sends a REQ_UD2 and learns the secondary address of the slaves from the reply telegram, as long as no collision occurs. If there is a collision after the selection or the REQ_UD2, the master varies the next positions and holds the existing one. If there is a collision, for example at 5FFFFFFF, the selection is run through from 50FFFFFF to 59FFFFFF. If in this case collisions again occur, then a change is made to a variation of the next position. After running through a complete position, the next higher position is processed up to 9.

With this Wildcard searching procedure, it will be seen that at least the top position must be run through in order to reach all slaves. Running through further positions may be necessary, depending on the number of the slaves and the distribution of the identification numbers. This procedure allows a statement of the maximum number of selections in relation to the number of slaves, but as disadvantage frequent collisions, which occur, should be mentioned. The wildcard searching procedure must be performed for all used baudrates and both byte sequences (mode 1 and 2).

The search procedure can be extended with searching for manufacturer, generation and finally device types~~media~~ to find slaves, which have the same identification number. It is also possible to search for all slaves of a certain manufacturer or all slaves of a certain device type~~media~~ by setting the corresponding value. With extended selection meters which differ only in their manufacturer specific fixed fabrication number can be distinguished.

Appendix A: Coding of Data Records (Normative)

The standard IEC 870-5-4 defines the following data types for usage inside the application layer:

Type A = Unsigned Integer BCD := XUI4 [1 to 4] <0 to 9 BCD>

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
digit 10				digit 1			
8	4	2	1	8	4	2	1
...	...	...	...	...	...	...	...
8	4	2	1	8	4	2	1

1UI4 [1 to 4] <0 to 9 BCD> := digit 10^0
2UI4 [5 to 8] <0 to 9 BCD> := digit 10^1
...
XUI4 [5 to 8] <0 to 9 BCD> := digit 10^{X-1}

Digits values of $A-$E in any digit position signals invalid.
A hex code $F in the MSD position signals a negative BCD number in the remaining X-1 digits. For details of this coding see appendix B.

Type B = Binary Integer := I[1..X] <(-2^{X-1} -1) to +(2^{X-1}-1)>

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
...							...
S	2^{X-2}						2^{X-8}

1B1 [X] := S=Sign: S<0> := positive
S<1> := negative
negative values in two´s complement

The coding „10000000b“ signals „invalid“

Type C = Unsigned Integer := UI[1 to X] <0 to 2^X-1>

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
...							...
2^{X-1}							2^{X-8}

UI8 [1 to 8] <0 to 255>

Type D = Boolean (1 bit binary information) := XB1 B1[i] <0 to 1>

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
...							...

XB1: B1[i] <0 to 1>
B1[i] <0> := false

2^{X-1}	2^{X-8}

B1[i] <1> := true

Type E = Compound CP16 (types and units information)

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8

1UI6[1 to 6] <0 to 63> := physical unit 1
1UI6[9 to 14] <0 to 63> := physical unit 2
1UI4[7,8,15,16] <0 to 15> := measured media

Note that this is special coding used only in fixed format data respond. It shall therefore not be used in new developments.

The following data types can only be used with the variable data structure:

Type F = Compound CP32: Date and Time

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8
2^{23}	2^{22}	2^{21}	2^{20}	2^{19}	2^{18}	2^{17}	2^{16}
2^{31}	2^{30}	2^{29}	2^{28}	2^{27}	2^{26}	2^{25}	2^{24}

min: UI6 [1 to 6] <0 to 59>
„63“: every minute
hour: UI5 [9 to13] <0 to 23>
„31“: every hour
day: UI5 [17 to 21] <1 to 31>
„0“: every day
month: UI4 [25 to 28] <1 to 12>
„15“: every month
year: UI7[22 to 24,29 to 32] <0 to 99> 99>
„127“: every year
hundred year: UI2 [14 to 15] <0 to 3>
The year is 1900+100*hundred year+year
IV: B1[8] {time invalid}: IV<0> := valid,
IV>1> := invalid
SU: B1[16] {summer time}:
SU<0> := standard time,
SU<1> := summer time
RES1: B1[7] {reserved}: <0>
RES2: B1[14] {reserved}: <0>
RES3: B1[15] {reserved}: <0>

Type G: Compound CP16: Date

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8

day: UI5 [1 to 5] <1 to 31>
„0“: every day
month: UI4 [9 to 12] <1 to 12>
„15“: every month

year: UI7[6 to 8,13 to 16] <0 to 9999>
127: every year

For compatibility with old meters with a circular two digit date it is recommended to consider in any master software the years „00“ to „80“ as the years 2000 to 2080.

TType H: Floating point according to IEEE-standard

"Short floating Point Number IEEE STD 754" = R32IEEESTD754
R32IEEESTD754 := R32.23 {Fraction, Exponent, Sign}

Fraction	=	F := UI23 [1to 23] <0 to $1-2^{-23}$>
Exponent	=	E := UI8 [24 to 31] <0 to 255>
Sign	=	S := BS1 [32] S<0> = positive
		S <1> = negative

F <0> and E <0>	:= (-1) S ∗ 0	= ± zero
F <≠0> and E <0>	:= (-1) S ∗ 2E-126(0.F)	= denormalized numbers
E <1 to 254>	:= (-1) S ∗ 2E-127(1.F)	= normalized numbers
F <0> and E <255>	:= (-1) S ∗ ∞	= ± infinite
F <≠0> and E <255>	:= NaN	= not a number, regardless of S

bits	8	7	6	5	4	3	2	1
octet 1	F = Fraction							
	2^{-16}	2^{-17}	2^{-18}	2^{-19}	2^{-20}	2^{-21}	2^{-22}	2^{-23}
octet 2	F = Fraction							
	2^{-8}	2^{-9}	2^{-10}	2^{-11}	2^{-12}	2^{-13}	2^{-14}	2^{-15}
octet 3	E (LSB)	F = Fraction						
	2^{-0}	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}
octet 4	Sign	E = Exponent						
	S	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}

The following ranges are specified by IEE Std 754-1985 for floating point arithmetics:

Range: $(-2^{128} + 2^{104})$ to $(+2^{128} - 2^{104})$, that is $-3.4*10^{38}$ to $+3.4*10^{38}$
smallest negative number: -2^{-149}, that is: $-1.4*10^{-45}$
smallest positive number: $+2^{-149}$, that is: $+ 1.4*10^{-45}$

Appendix B: Interpretation of Hex-Codes $A-$F in BCD-data fields

General description

1.) Standard Reference

This standard allows multi-digit BCD-coded datafields. It does however not contain information about what happens if a non-BCD hex code ($A-$F) is detected by the master software.

2.) Purpose of this proposal

a) Define the treatment of non BCD-digits in slave to master RSP_UD-telegrams

To fully define a master software including error treatment such a definition would be desirable.

b) Utilize these codes for simplified error treatment by slave

The current user group proposal contains various techniques for signalling errors or abnormal situations. Most of them are hard to implement on weak mikro processors. Utilizing these "illegal" codes $A to $F for signalling these states to the master would simplify the software design of the slaves.

c) Anormal states of variables

- Value not available

  This happens for example, if a fixed date value is not yet available, because the first fixed date is in the future. The display at the meter or the remote PC should read "---".

- Device error

  This could happen for a temperature variable, if the sensor is malfunctioning. The display at the meter or a remote PC should signal some error code. Multiple error codes should be supported.

- Soft overflow

  Exceeding the upper count limit on integral values or the upper value limit on momentary values should be signallable. For a wrap around carry of integral variables the display should be consistent with old mechanical wrap around counters. In addition a wrap around flag should be extractable.

- Soft underflow

Underflowing the lower count limit of 00 on integral values or a negative value on momentary values should be signallable. For a wrap around carry of integral variables the display should be consistent with old mechanical wrap around counters. In addition a wrap around flag should be extractable.

- Simple visible error signalling

  To simplify the design of slaves with integrated displays, the above mentioned non-BCD states of the variables should be both transmittable in the form of suitable (Hex) codes but also be displayable directly from the value codes of a 7-segment (usually LCD) display by extending the normal ten entry BCD to 7-segment decoding a table to either a dual 16-entry or a single 32-entry decoding table where 16 entries are used for decoding the MSD (Most Significant Digit) and the other 16 entries are used for the decoding for all other for all other digits. For very weak mikroprocessors with a maximum of a single decoding table with only 16 entries a compatible solution with decreased functionality is also presented.

Recommendation

1.) Definition of hex code meanings

a) $A-$E

Such a code in the MSD (Most significant digit) position signals a one digit value overflow either of a number or due to an addition or increment carry. The display at the meter or a remote PC should display a "0" at the appropriate display position. This makes the display compatible with conventional counter rollover. In addition the leading digit can be treated by the slave software simply as a hexadecimal digit instead of the BCD-coded other digits to realize this function. Processing software in the master could convert this data digit to a value of 10 in an extended length data field. In addition an appropriate application error code could be generated if desired. If this hex code appears in any other digit position than the MSD, it signals a value error of the complete data field. If an alternative display decoding table for the digits other than the MSD is possible, this hex code should be displayed in these other digit positions as the symbol "A". This would allow more flexible displayable error codes.

Example: A 4-digit BCD code of "A321" should be interpreted by the master software as "10321" with an optional overrange VIFE-error code and displayed as 0321 on a 4-digit only display.

b) $B

Such a code in the MSD digit position signals a two digit value overflow either of a number or due to an addtion or increment carry. The display at the meter or a remote PC should display a

"1" at the appropriate display position. This makes the display compatible with conventional counter rollover. In addition the leading digit can be treated by the slave software simply as a hexadecimal digit instead of the BCD-coded other digits to realize this function. Processing software could convert this data digit to a value of 11 in an extended length data field. In addition an appropriate application error code could be generated if desired. If this hex code appears in any other digit position than the MSD, it signals a value or availability error of the complete data field. It should be displayed as the symbol "-".

Example: A 4-digit BCD code of "B321" should be interpreted by the master software as "11321" with an optional overrange VIFE-error code and displayed as 1321 on a 4-digit only display with digit selective decoding.

c) $C

Such a code in the MSD digit position signals a three digit value overflow either of a number or due to an addition or increment carry. The display at the meter or a remote PC should display a "2" at the appropriate display position. This makes the display compatible with conventional counter rollover. In addition the leading digit can be treated by the slave software simply as a hexadecimal digit instead of the BCD-coded other digits to realize this function. Processing software could convert this data digit to a value of 12 in an extended length data field. In addition an appropriate application error code could be generated if desired. If this hex code appears in any other digit position than the MSD, it signals a value error of the complete data field. It should be displayed as the symbol "C". Note that the suggested interpretation of $A to $C in the MSD effectivly supports a 30% overrange guard band against an undetected rollover and flexible error codes including the letters "A", "C", "E" and "F".

Example: A 4-digit BCD code of "C321" should be interpreted by the master software as "12321" with an optional overrange VIFE-error code and displayed as 2321 on a 4-digit only display.

d) $D

Such a code in any digit position signals a general error of the complete data field. The display at the meter or a remote PC should display an appropriate symbol blank at the appropriate display position. Since both an overflow from $C and an underflow from $E end in this out of range type error the function of an out-of-range over/underflow can be implemented by simple hex arithmetic. It is however recommended that the slave arithmetic checks this $D-code in the MSD before incrementing or decrementing the value for integral variables to make such an error irreversible if the slave does not expect such an over- or underflow.

e) $E

Such a code in the MSD digit position signals a two digit value underflow either of a number or due to a subtraction or decrement borrow. The display at the meter or a remote PC should display an "8" at the appropriate display position. This makes the display compatible with conventional counter rollunder. In addition the leading digit can be treated by the slave software simply as a hexadecimal digit instead of the BCD-coded other digits to realize this function. Processing software could convert the data in the field to a negative value using 16´s complement on the leading digit and tens complement at the other digits. In addition an appropriate application error code could be generated if desired. If this hex code appears in any other digit position than the MSD, it signals a value error of the complete data field. It should be displayed as the symbol "E".

Example: A 4-digit BCD code of "E321" should be interpreted by the master software as "- 1679" (F999-E321+1) with an optional underrange VIFE-error code and displayed as 8321 on a 4-digit only display.

bf) $F

Such a code in the MSD digit position signals a „minus-sign“ in front of the remaining (N-1) digit number. In any other digit position it signals an error.n one-digit value underflow either of a number or due to a subtraction or decrement borrow. The display at the meter or a remote PC should display an "9" at the appropriate display position. This makes the display compatible with conventional counter rollunder. In addition the leading digit can be treated by the slave software simply as a hexadecimal digit instead of the BCD-coded other digits to realize this function. Processing software could convert the data in the field to a negative value using 16´s complement on the leading digit and tens complement at the other digits. In addition an appropriate application error code could be generated if desired. If an alternative display decoding table for the digits other than the MSD is possible, this hex code should be displayed in these other digit positions as the symbol "F". Note that the suggested interpretation of $E and $F in the MSD effectivly supports a 20% underrange guard band against an undetected rollunder for displays and flexible error displays if dual decoding tables are available. In addition it allows a simplified coding for small negative values as often required for values like temperature or flow rate.

Example: A 4-digit BCD code of "F321" should be interpreted by the master software as "-0321679" (F999-F321+1) with an optional underrange VIFE-error code and displayed as -9321 on a 4-digit only display.

g) Combinations

If with the exception of the MSD all other digits are true BCD-digits ($0-$9) the value is either considered as "Overflow" for the MSD hex codes $A to $C or as "Underflow" for the MSD hex codes $E and $F or a general error for the MSD hex code $D.
The code $DBB.. in the data field is always considered as "not available". This is displayed as " ---" (with a blank in the MSD). Any other non BCD hex codes in one or several digits other than the MSD is interpreted as an error for the complete data field. The error type is formed from the characters "A", "C", "E", "F" (all corresponding to their hex code), "-", blank and the digits 0-9. The display may show an identical error code if displaying the variable, but the MSD digit on the display can contain only blank or the digits 0..9.

h) Decoding table for 2*16 entries (or 32 entries)

	0	1	2	3	4	5	6	7	8	9	$A	$B	$C	$D	$E	$F
MSD	"0"	"1"	"2"	"3"	"4"	"5"	"6"	"7"	"8"	"9"	"0"	"1"	"2"	" "	"8"	"9"
other digits	"0"	"1"	"2"	"3"	"4"	"5"	"6"	"7"	"8"	"9"	"A"	" "	"C"	" "	"E"	"F"

i) Subset functions

A slave may utilize either non, a single, several or all suggested special functions and their associated hex codes. A slave might utilize also a different number of hex code functions for different data fields. A slave could also use different display implementations for the various special functions and error displays but the suggested solution would simplify the operation of the system, since the master display will be identical to the slave display for the value associated with the appropriate data field.

2.) Recommended LCD-Decoding tableSubset for single 16-entry display decoding

a) Decoding table

0	1	2	3	4	5	6	7	8	9	$A	$B	$C	$D	$E	$F
"0"	"1"	"2"	"3"	"4"	"5"	"6"	"7"	"8"	"9"	"A0"	"b-"	"C"	" "	"E"	"-9"

b) Overrange

The overrange feature should be limited to a 10% overrange ($A in MSD). A further increment should lead directly to the MSD-error hex code $D, which should stop further increment and decrement and generate a suitable error code in the other digits

c) Underrange

The underrange feature should be limited to a 10% underrange ($F in MSD). A further decrement should lead directly to the MSD-error hex code $D, which should stop further increment and decrement and generate a suitable error code in the other digits.

d) Error codes

Error codes may contain only the letters "C" and "E", blank, "-" and the digits 0-9.

e) Compatiblity

At the master side this subset realisation is completely compatible and transparent to the full implementation.

Appendix C: Alarm Protocol (Recommendation)

The formerly described method for an alarm protocol (see diploma work of Andreas Steffens "Eigenschaften und Anwendungen des M-Bus") was based on time slices for each of the maximum 64 alarm devices. This alarm protocol has not been standardized.
We now suggest to return to standard alarm protocol which conforms to the standard IEC 870-2:
The master software polls the maximum 250 alarm devices by requesting time critical data (REQ_UD1 to adresses 1 .. 250). A slave can transmit either a single character acknowledgement E5h signalling no alarm or a RSP_UD with the CI-Field 71h to report an alarm state.

68h	04h	04h	68h	08h	Adr	71h	Alarm State	CS	16h

Fig. 5 *Telegram for an Alarm-Respond*

The alarm state is coded with data type D (boolean, in this case 8 bit). Set bits signal alarm bits or alarm codes. The meaning of these bits is manufacturer specific.

The timeout for time critical communication must be set to 11..33 bit periods to ensure a fast poll of all alarm devices. With a baudrate of 9600 Bd and all 250 slaves reporting an alarm just in time before a timeout occurs each slave will be polled in periods of maximum 5.5 seconds. This seems to be fast enough for alarms in building control systems and other applications. For faster alarm systems the number of alarm sensors could be limited to 63 (reducing the worst case overall signal delay to less than 1.5 sec or increase the transmission speed to 38400 Bd and achieve the same speed for up to 250 devices.
The functionality of the FCB- and FCV-Bit should be fully implemented in this alarm protocol to ensure that one-time alarms are safely transmitted to the master. If the slave has reported an one-time alarm and the next REQ_UD1 has a toggled FCB (with FCV=1) the slave will answer with an E5h signalling no alarm. Otherwise it will repeat the last alarm frame to avoid that the alarm message gets lost.

Appendix D: Fixed data Structure (Informative, Obsolete)

This alternate data structure has been defined in the first edition of this standard. Due to ist limited flexibility it is not recommend for new developments of meters or other slaves. It is described here for implementing universal master software for the transition period where devices using this protocol are still on the market or in use.

Identification No.	Access No.	Status	Medium/Unit	Counter 1	Counter 2
4 Byte	1 Byte	1 Byte	2 Byte	4 Byte	4 Byte

Fig. 6 *Fixed Data Structure in Reply Direction (transmit sequence from left to right)*

The **Identification Number** is a serial number allocated during manufacture, coded with 8 BCD packed digits (4 Byte), and which thus runs from 00000000 to 99999999.
The **Access Number** has unsigned binary coding, and is increased by one after each RSP_UD from the slave. With the field **Status** various information about the status of counters, and faults which have occurred, can be communicated - see Figure 16:

Bit	Meaning with Bit set	Significance with Bit not set
0	Counter 1 and 2 coded signed binary	Counter 1 and 2 coded BCD
1	Counter 1 and 2 are stored at fixed date	Counter 1 and 2 are actual values
2	Power low	Not power low
3	Permanent error	No permanent error
4	Temporary error	No temporary error
5	Specific to manufacturer	Specific to manufacturer
6	Specific to manufacturer	Specific to manufacturer
7	Specific to manufacturer	Specific to manufacturer

Fig. 7 *Coding of the Status Field*

The field **Medium/Unit** is **always** transmitted with least significant byte first and gives the medium measured for both counter states, and the units for each of the two counter states. The units of counter 1 are coded with the first 6 bits of the first byte, and the units of counter 2 with the first 6 bits of the second byte. The coding of the medium is made up of the two highest bits of these bytes, and can therefore have 16 different values (4 bits). Tables to represent the physical units and the coding of the medium are in the appendix.

Byte	Byte No. 8 (byte 2 of medium/unit)								Byte No. 7 (byte 1 of medium/unit)							
Bit	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
	Medium		physical unit of counter 2						Medium		physical unit of counter 1					
	MSB		MSB					LSB		LSB	MSB					LSB

Fig. 8 *Coding of physical unit and medium in fixed data structure (data type E)*

To allow transmission of one historic value with one of the two counters the special unit (111110b or hex code share of 3Eh) has been defined. This unit declares that this historic counter has the same unit as the other actual counter.

D.1 Measured Medium Fixed Structure

Value hexadecimal	Field Medium/Unit				Medium
	Bit 16	Bit 15	Bit 8	Bit 7	
0	0	0	0	0	Other
1	0	0	0	1	Oil
2	0	0	1	0	Electricity
3	0	0	1	1	Gas
4	0	1	0	0	Heat
5	0	1	0	1	Steam
6	0	1	1	0	Hot Water
7	0	1	1	1	Water
8	1	0	0	0	H.C.A.
9	1	0	0	1	Reserved
A	1	0	1	0	Gas Mode 2
B	1	0	1	1	Heat Mode 2
C	1	1	0	0	Hot Water Mode 2
D	1	1	0	1	Water Mode 2
E	1	1	1	0	H.C.A. Mode 2
F	1	1	1	1	Reserved

Notes:

1. Record Medium/Unit is always least significant byte first.
2. H.C.A. = Heat Cost Allocator
3. Media from "Gas Mode2" to "H.C.A. Mode2" are defined additionally to EN1434-3 for some existing meters with CI-Field 73h (intentionally mode1), which transmit the multibyte records with high byte first in contrast to the CI-Field. The master must know that these media codes mean mode 2 or high byte first. Further use of these codes for "pseudo media" is not allowed for new developments.

D.2 Table of Physical Units

Unit	MSB..LSB	Hex code share Byte 7/8	Unit	MSB..LSB	Hex code share Byte 7/8
h,m,s	000000	00	MJ/h	100000	20
D,M,Y	000001	01	MJ/h * 10	100001	21
Wh	000010	02	MJ/h * 100	100010	22
Wh * 10	000011	03	GJ/h	100011	23
Wh * 100	000100	04	GJ/h * 10	100100	24
kWh	000101	05	GJ/h * 100	100101	25
kWh * 10	000110	06	ml	100110	26
kWh * 100	000111	07	ml * 10	100111	27
MWh	001000	08	ml * 100	101000	28
MWh * 10	001001	09	l	101001	29
MWh * 100	001010	0A	l * 10	101010	2A
kJ	001011	0B	l * 100	101011	2B
kJ * 10	001100	0C	m^3	101100	2C
kJ * 100	001101	0D	m^3 * 10	101101	2D
MJ	001110	0E	m^3 * 100	101110	2E
MJ * 10	001111	0F	ml/h	101111	2F
MJ * 100	010000	10	ml/h * 10	110000	30
GJ	010001	11	ml/h * 100	110001	31
GJ * 10	010010	12	l/h	110010	32
GJ * 100	010011	13	l/h * 10	110011	33
W	010100	14	l/h * 100	110100	34
W * 10	010101	15	m^3/h	110101	35
W * 100	010110	16	m^3/h * 10	110110	36

kW	010111	17	m^3/h * 100	110111	37
kW * 10	011000	18	°C * 10^{-3}	111000	38
kW * 100	011001	19	units for HCA	111001	39
MW	011010	1A	reserved	111010	3A
MW * 10	011011	1B	reserved	111011	3B
MW * 100	011100	1C	reserved	111100	3C
kJ/h	011101	1D	reserved	111101	3D
kJ/h * 10	011110	1E	same but historic	111110	3E
kJ/h * 100	011111	1F	without units	111111	3F

Example for a RSP_UD with fixed data structure (mode 1):

The slave with address 5 and identification number 12345678 responds with the following data (all values hex.):

68 13 13 68	header of RSP_UD telegram (L-Field = 13h = 19d)
08 05 73	C field = 08h (RSP_UD), address 5, CI field = 73h (fixed, LSByte first)
78 56 34 12	identification number = 12345678
0A	transmission counter = 0Ah = 10d
00	status 00h: counters coded BCD, actual values, no errors
E9 7E	Type&Unit: medium water, unit1 = 1l, unit2 = 1l (same, but historic)
01 00 00 00	counter 1 = 1l (actual value)
35 01 00 00	counter 2 = 135 l (historic value)
3C 16	checksum and stop sign

Appendix E: Examples

Example for a RSP_UD with variable data structure answer (mode 1):

(all values are hex.)

68 1F 1F 68	header of RSP_UD telegram (length 1Fh=31d bytes)
08 02 72	C field = 08 (RSP), address 2, CI field 72H (var.,LSByte first)
78 56 34 12	identification number = 12345678
24 40 01 07	manufacturer ID = 4024h (PAD in EN 61107), generation 1, water
55 00 00 00	TC = 55h = 85d, Status = 00h, Signature = 0000h
03 13 15 31 00	**Data block 1:** unit 0, storage No 0, no tariff, instantaneous volume, 12565 l (24 bit integer)
DA 02 3B 13 01	**Data block 2:** unit 0, storage No 5, no tariff, maximum volume flow, 113 l/h (4 digit BCD)
8B 60 04 37 18 02	**Data block 3:** unit 1, storage No 0, tariff 2, instantaneous energy, 218,37 kWh (6 digit BCD)
18 16	checksum and stopsign

~~The **VIFE** can be used for actions which shall be done with the data (master to slave, chapter 6.5), for reports of application errors (slave to master, chapter 6.6) and for an enhancement of the VIF (orthogonal VIF, chapter 8.4.5). The last feature allows setting VIF´s into relation to the base physical units (e.g. VIF=10 liter, VIFE= per hour) or coding indirect units, pulse increments and change speeds.~~

~~In case of **VIFE** = FFh the next **VIFE's** and the data of this block are manufacturer specific, but the VIF is coded as normal.~~

~~After a VIF or VIFE with an extension bit of "0", the value information block is closed, and therefore also the data record header, and the actual data follow in the previously given length and coding.~~

Example baud rate switch:

The master switches the slave (in point to point connection) from now 2400 baud to 9600 baud.

Master to slave:	68 03 03 68 | 53 FE BD | 0E 16	with 2400 baud
Slave to master:	E5	with 2400 baud

From that time on the slave communicates with the transmission speed 9600 baud, if the slave can handle 9600 baud, otherwise it remains at 2400 baud.

In busmode this must be followed within < 2min by an acknowledged communication (i.e. SND_NKE) at 9600 baud:

Master to slave: 10 40 FE 3E 16
Slave to master: E5

Example Reset with subcode:
The master releases an enhanced application reset to all slaves. All telegrams of the user data type are requested.

Master to Slave: 68 04 04 68 | 53 FE 50 | 10 | B1 16
Slave to Master: E5

E.2 Writing Data to a Slave

The master can send data to a slave using a SND_UD with CI-Field 51h for mode 1 (or 55h for old mode 2-meters). Note that the data structure in such a write telegram has been changed in contrast to previous definitions by means of leaving out the fixed data header of 12 byte. The following figure shows the data structure for a write telegram. The order of the first three blocks in the following figure can be turned round, but the write only data record must be at the end of the telegram. All records are optional.

Primary Address Record	Enhanced Identification Record	Normal Data Records	Write Only Data Records

Fig. 9 *Data Structure for Writing Data*

- Primary Address Record:
  The primary address record is optional and consists of three bytes:

DIF = 01h	VIF = 7Ah	Data = Address (1 byte binary)

  With this data record a primary address can be assigned to a slave in point to point connections. The master must know all the used addresses on the bus and forbid setting the address of a slave to an already used address. Otherwise both slaves with the same address couldn´t be read out anymore.

- Enhanced Identification Record:
  With this optional data record the identification (secondary address) can be changed. There are two cases to be distinguished:

  1) Data is only the identification number

DIF = 0Ch	VIF = 79h	Data = Identification No. (8 digit BCD)

2) Data is the complete identification

DIF = 07h	VIF = 79h	Data = complete ID (64 bit integer)

The data is packed exactly as in the readout header of a $72/$76 variable protocol with low byte first for mode 1 and high byte first for mode 2:

Identification No.	Manufacturer ID	Generation	Medium
4 byte	2 byte	1 byte	1 byte

- Normal Data Records:
  The data records, which can be read out with a REQ_UD2, are sent back to the slave with the received DIF and VIF and the new data contents. Additional features can be implemented using the generalized object layer (see chapter 6.5).
- Write-Only Data:
  Data, which cannot be read out of the slave with a normal data block, can be transmitted using the VIF = 7Fh for manufacturer specific coding. The DIF must have a value corresponding to the type and length of data.

After receiving the SND_UD correctly without any error in data link layer the slave must answer with an acknowledgement (E5h). The slave decides whether to change variables or not after a data write from the master. In case of errors in executing parts of or whole write instructions the slave can decide whether to change no variables or single correct variables. The slave can report the this errors to the master in the next RSP_UD telegram using some of the methods which are described in chapter 6.6.
There are some methods for implementing write protect, for example allowing only one write after a hardware reset of the processor or enabling write if a protect disable jumper is set.

Examples:

1. Set the slave to primary address 8 without changing anything else:
   68 06 06 68 | 53 FE 51 | 01 7A 08 | 25 16

2. Set the complete identification of the slave (ID=01020304, Man=4024h (PAD), Gen=1, Med=4 (Heat):
   68 0D 0D 68 | 53 FE 51 | 07 79 04 03 02 01 24 40 01 04 | 95 16 ♣

3. Set identification number of the slave to "12345678" and the 8 digit BCD-Counter (unit 1 kWh) to 107 kWh.
   68 0F 0F 68 | 53 FE 51| 0C 79 78 56 34 12 | 0C 06 07 01 00 00 | 55 16

E.3 Configuring Data Output

For default the slave transmits all his data with a RSP_UD. It could be useful for some applications to read only selected data records out of one or more devices. There are two ways to select data records:

Selection without specified data field

The selection of the wanted data records can be performed with a SND_UD (CI-Field = 51h/55h) and data records containing the data field 1000b, which means "selection for readout request". The following VIF defines the selected data as listed in EN1434-3 and no data are transmitted. The answer data field is determined by the slave. The master can select several variables by sending more data blocks with this data field in the same telegram.

Special multiple values can be selected with the following methods:

- Any VIF:
  The VIF-Code $7E (any VIF) is especially for readout request of "all VIF" from the slave and can be interpreted as a selection wildcard for the value information field.
- Global readout request:
  The DIF-Code $7F is defined as "selection of all data for readout request", i.e. all storage numbers, units, tariffs and functions. If this DIF is the last byte of user data or the VIF=$7E follows, then all data is requested. So the selection of all data of one slave can de done with a SND_UD and the character $7F as the user data. If there follows a DIF unequal to $7E, then all subfields of this VIF are selected for readout.
- All Tariffs:
  The highest tariff number in the selection record is defined as selection of "all tariffs". For example the tariff 1111b (15) means selection of all tariffs in a record with two DIFE´s.
- All Storage Numbers:
  A selection of all storage numbers can be done with the maximum storage number if there is a minimum of one DIFE. For example the highest storage number is $1F (31) with one DIFE and $1FF (511) with two DIFE´s.

- All Units:
  "All units" can be selected by using a data record header with minimum two DIFE´s and the highest unit number.
- High Resolution Readout:
  The master can select the slave to answer with the maximum resolution to a given value / unit by a VIF with "nnn" = 000 (minimum exponent for range coding). The meter may then answer with a resolution of e.g. 1mWh (VIF=0000000b) or some higher decimal value if required. The unit values have been chosen so that their minimum provides sufficient resolution even for calibration. A readout request for a VIF with "nnn"=max (maximum exponent for range coding) signals a request for the standard resolution of the meter.

After the next REQ_UD2 the slave answers with the selected data in his own format, if the requested data are available. Otherwise the slave transmits his normal data and the master has to find out that the data are not the requested one. If there are more than one variables with the selected VIF, the device should send all these data records.

Selection with specified data field

The master is able to perform a readout request with a specified data field by using the object action “add to readout list“ (VIFE = E000 1100b) from VIFE-table for object actions (see chapter 7, table 176.5). The master transmits a SND_UD (CI-Field = 51h/55h) with a data record which consists of the desired DIF (data field), VIF and the VIFE = 0Ch / 8Ch. No data follows this VIFE and the slave should ignore the data field on reception. The slave should transmit this data record with the requested data field from now on, if he is capable of this. If the slave doesn´t support this data field (data coding), it can report a record error using one of the VIFE = E000 011x (data class not implemented or data size not implemented).

Deselection of data records

The master can release a reset of the application layer and especially a fallback to the slaves standard RSP_UD-Telegram by transmitting a SND_UD with the CI-Field $50.
Single data records can be deselected by transmitting a data record with DIF, VIF and the VIFE for the object action “Delete from Readout-List“ (VIFE = E000 1101b).

If the selected data is supported by the slave but too long for one RSP_UD telegram (especially for readout of all historic values), the slave transmits an additional data record consisting only of the DIF=$1F, which means that more data records follow in the next

respond telegram. In this case the master must readout the slave again until the respond telegram is only an $E5 (no data) or there is no DIF=$1F in the RSP_UD.
To avoid lost of data respond telegrams the slave should in this case support the Frame Count Bit (FCB). If the master wants to premature end such a multitelegram sequential readout of the selected data, it may send an application reset with CI=$50 instead of further REQ_UD2´s.

Examples:

1. A slave with address 7 is to be configured to respond with the data records containing volume (VIF=13h: volume, unit 1l) and flow temperature (VIF=5Ah: flow temp., unit 0.1 °C).
   68 07 07 68 | 53 07 51 | 08 13 08 5A | 28 16

2. A slave with address 1 is to be configured to respond with all storage numbers, all tariffs, and all VIF´s from unit 0.
   68 06 06 68 | 53 01 51 | C8 3F 7E | 2A 16

3. A slave with address 3 is to be configured to respond with all data for a complete readout of all available. After that the master can poll the slave to get the data.
   68 04 04 68 | 53 03 51 | 7F | 26 16

With these actions the master can alter the data of the slaves or configure the output data of the slaves (actions 12 and 13). The actions 0 to 6 alter the data of the slave by replacing the old data (action 0, equals to data write without VIFE) or do arithmetical or logical operations with the old and the transmitted data.
Note that this method of configuring the readout list (action 12 and 13) allows not only the adding but also the removal of elements in contrast to the method of using the DIF=1000b-type of readout request (described before~~in chapter 6.4.3~~).
All these actions can be used for normal slaves and for intelligent master which are manipulated by a higher order master.
The functions "Add entry" and "Delete entry" are useful to tell an intelligent master to add e.g. a new data record like maximum or minimum values of any slave.

With the action "freeze data to storage #" the master can tell the slave to freeze the actual value corresponding to the transmitted VIF, unit, tariff and function to a certain storage number given in the DIF/DIFE´s. In this case the data field inside the VIF has got the value 0000b (no data). This action allows freeze of selected values or multiple freeze with VIF=$7E (all VIF). The date / time should also be freezed to the same storage number.

Examples:

1) Set the 8 digit BCD-Counter (instantaneous, actual value, no tariff, unit 0) with VIF=06 (1kWh) of the slave with address 1 to 107 kWh.
   68 0A 0A 68 | 53 01 51 | 0C 86 00 07 01 00 00 | 3F 16

2) Same as in example 1) but add 10 kWh to the old data.
   68 0A 0A 68 | 53 01 51 | 0C 86 01 10 00 00 00 | 48 16

3) Add an entry with an 8 digit BCD-Counter (instantaneous, actual value, no tariff, unit 0, 1kWh) with the start value of 511 kWh to the data records of the slave with address 5.
   68 0A 0A 68 | 53 05 51 | 0C 86 08 11 05 00 00 | 59 16

4) Freeze actual flow temperature (0.1 °C: VIF = 5Ah) of the slave with address 1 into the storage number 1.
68 06 06 68 | 53 01 51 | 40 DA 0B | CA 16

E.4 Slave Collision Detect

Collisions between transmitting slaves can occur during slave search activities by the master. Very light collisions of (22..33) mA, which are equivalent to 2 or 3 transmitting slaves, are electrically undetectable by master and slave. New master hardware with double current detect can detect light collisons of (20..200) mA and then transmit a break (50 ms space) on the bus. The slave can detect medium collisions of (70..500) mA, if this is a collision between a mark and a space and if the slave supports this feature. Heavy collisions of (90..5000mA) will have the effect of a break down of the bus voltage (power fail in the slave) and possibly a shortcircuit in the master.
To avoid these consequences of (heavy) collisions new master have the feature of double current detect with break signaling and switching off the bus in overcurrent states. There are some means for the slaves to detect collisions and then stop transmitting:

1. Software based UART´s can test at the end of each Mark-Send-Bit whether the input is really a mark. This guarantees a very fast detection of collisions, is simple to implement and is strongly recommended for pure software UART.
2. A variation of the preceeding method is to test whether the bus voltage is mark ~~after each stop bit~~directly before the transmission of each start bit. This is simple for a software UART, but very tricky for a hardware UART and requires a master sending a break on collision detect.
3. A simple method for unbuffered hardware UART, but tricky for buffered hardware UART, is to compare the transmitted with the received byte.
4. Another method, which requires a master with break collision detect, is a hardware UART with break detect.

~~5. The baudrate of the communication process after a detected break will be 300 Baud ♣.~~

E.5 FCB-Bit and Selection

FCB-Implementation slave

A slave with implemented secondary addressing and with implemented FCB-administration must have an additional set of 0, 1 or 2 separate "Last Received FCB"-memory Bit(s) for all communication via the pseudo primary address 253 ($FD). If it can communicate also alternatively over some other primary address (exept the special addresses 254 and 255) an additional set of 0, 1 or 2 "Last received FCB"-memory bit(s) for each of these primary addresses is required. A valid selection telegram will not only set the internal selection bit but will also clear all 0, 1 or 2 internal "Last received FCB"-memory bit(s) associated with secondary addressing via the pseudo primary address 253 ($FD). The master will start the communication (REQ_UD2 or SND_UD) after any selection telegram (CI=$52 or $56) with the FCV-Bit set and the FCB-Bit set. If a slave has more than one alternative secondary identification, only a single set of 0, 1 or 2 "Last received FCB"-memory bit(s) for all secondary addresses is required.

FCB-Implementation master

The master must implement a separate pair of "Next FCB image"-Bits for pseudo primary address 253 ($FD) as for each other primary address. Although these "Next FCB image"-bits might be used for many slaves, no confusion exists, since for accessing another slave a selection telegram is required which will define the future FCB sequence both for slave and master.

E.6 Special Slave Features

Some optional or recommended features of the slaves will be described in this section.

E.6.1 Use of the fabrication Number

The fabrication number is a serial number allocated during manufacture. It is part of the variable data block (DIF = $0C and VIF = $78) and coded with 8 BCD packed digits (4 Byte).
Example:

68 15 15 68	header of RSP_UD telegram (length 1Fh=31d bytes)
08 02 72	C field = 08 (RSP), address 2, CI field 72H (var.,LSByte first)
78 56 34 12	identification number = 12345678
24 40 01 07	manufacturer ID = 4024h (PAD in EN 61107), generation 1, water
13 00 00 00	TC = 13h = 19d, Status = 00h, Signature = 0000h
0C 78 04 03 02 01	fabrication number = 01020304
9D 16	checksum and stopsign

The use of this number is recommended if the identification number is changeable. In this case two or more slaves can get the same secondary adress and can not be uniquely selected.

The fabrication number together with manufacturer, version and medium field build an unique number instaed. Suitable masters use this number for an enhanced selection method if two or more slaves have the same identification numbersecondary adress (see chapter 9.37.4).

Appendix F: Secondary Search

The company Aquametro AG has simulated such a search to find the minimum, the average and the maximum number of selections as a function of the number of slaves. For the minimum number of attempts the optimum distribution of the identification numbers was chosen, for the maximum number the most unfavourable, and for the average number of attempts a random distribution. The following diagram shows the result of these calculations:

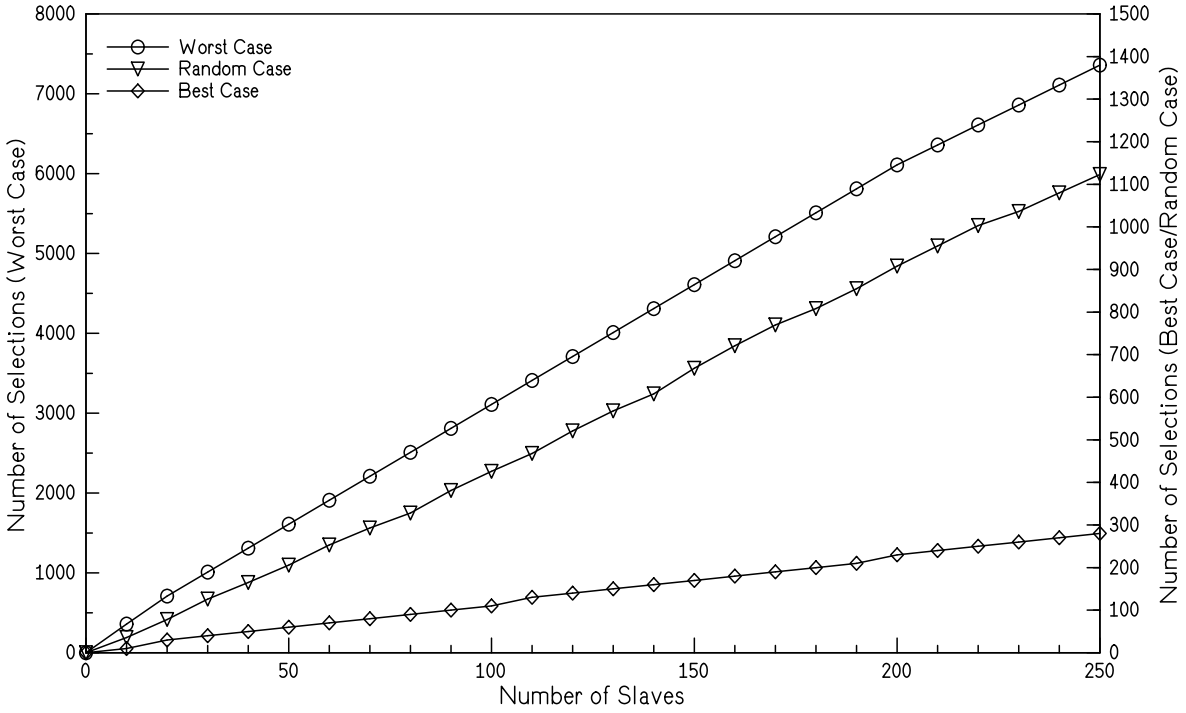

Fig. 10 *Number of Selections with Wildcard Searching Procedure*

Instructions for implementation of Wildcard Search

The following program flow diagram shows the realization of the Wildcard searching procedure, whereby the search is made only with the identification number. The codes for manufacturer, version and medium are in general specified with wildcards, but can be changed by the user in order (for example) to locate all meters from a particular manufacturer. In order to avoid the categorisation by a factor of eight of the "For-To" loops for the eight positions, the array "Value" is defined with 8 byte numbers, which are intended to define the contents of the positions. The digit number of the identification number which is presently running is noted in the variable "Pos" of type byte.

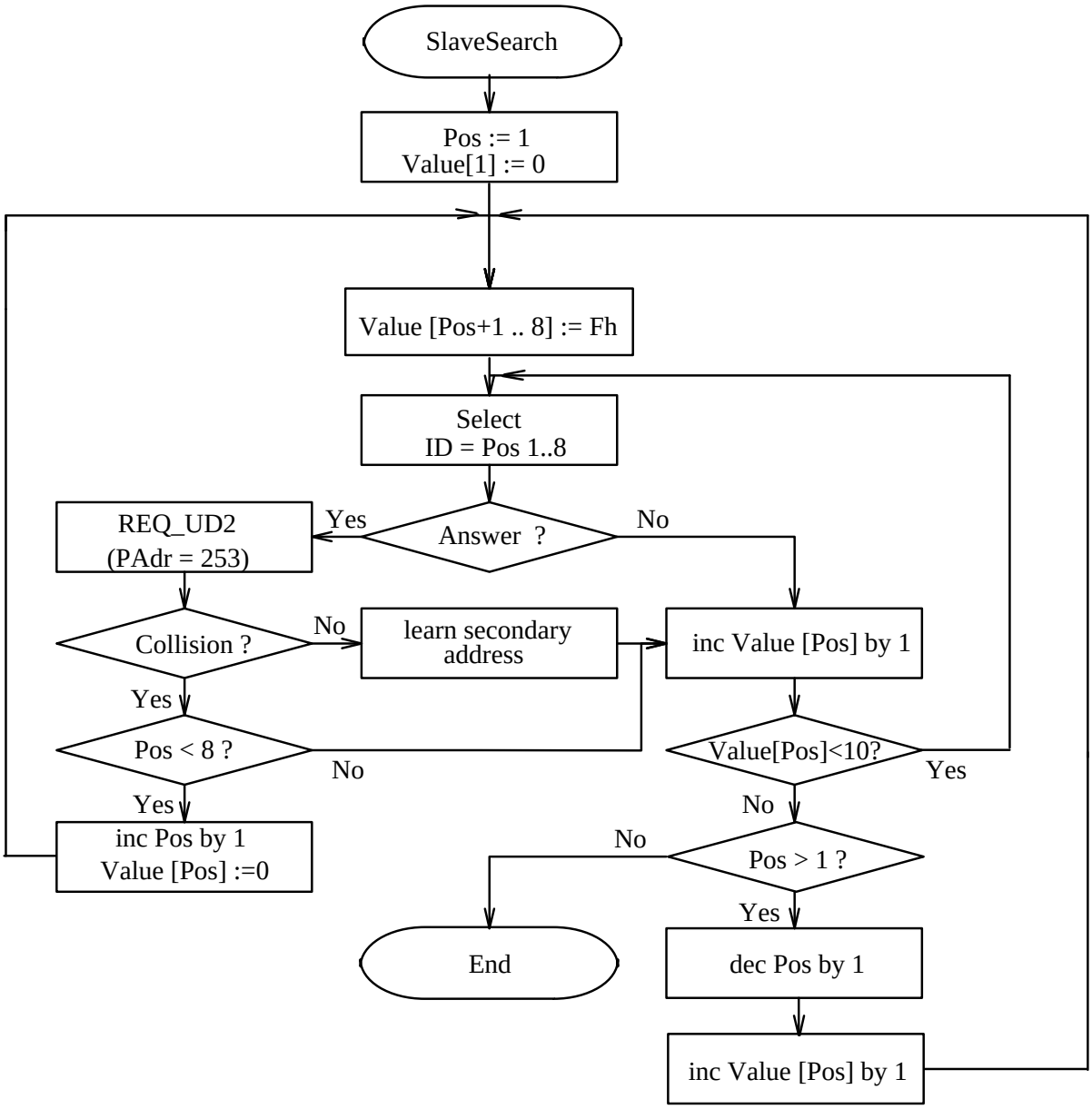

Fig. 11 *Flow Diagram for Slave Search with Wildcards*

The routine begins at the first position, and implements the following actions for the value of this position from 0 to 9:

- Selection with the ID-Nr. Pos 1, Pos 2, ..., Pos 8
- if no reply, Value [Pos] is raised by 1
- if there is a reply, a REQ_UD2 is sent to address 253, and if the telegram is correctly received the secondary address is learnt and the Value [Pos] raised by 1
- if there is a collision a jump is made to the next position (Pos increased by 1), as long as the last position has not yet been reached
- after going through a complete position from 0 to 9 the subroutine proceeds to the next lower position, or ends the search if the position Nr. 1 has already been processed

Example

The next figure shows an example for secondary addresses in order from top to bottom, as they will be found by the master software:

No.	Identification-Nr.	Manufacturer. (hex.)	Version (hex.)	Device type Media (hex.)
1	14491001	1057	01	06
2	14491008	4567	01	06
3	32104833	2010	01	02
4	76543210	2010	01	03

Fig. 12 *Secondary Addresses found with a Wildcard Search of Four Slaves*

Search Process:

1. Start with ID = 0FFFFFFF : no reply
2. ID = 1FFFFFFF : collision between Nr.1 and Nr.2
3. ID = 10FFFFFF, 11FFFFFF, 12FFFFFF, 13FFFFFF : no reply
4. ID = 14FFFFFF : collision between Nr.1 and Nr.2
5. Repeated steps 3 to 4 up to the ID = 1449100F
6. Learn ID = 14491001 and 14491008
7. Go backwards to 19999999
8. ID = 2FFFFFFF : no reply
9. ID = 3FFFFFFF : learn ID = 32104833
10. ID = 4FFFFFFF, 5FFFFFFF, 6FFFFFFF : no reply
11. ID = 7FFFFFFF : learn ID = 76543210
12. ID = 8FFFFFFF, 9FFFFFFF : no reply
13. End of the Search

Enhanced selection with fabrication number ♣

The identification number can be used as is a customer number and then can be changed by the operatormaster. Therefore it can be possible that two slaves have the same secondary adress. For this reason the selection telegram can be extended by a **fabrication number** to make sure that in any case all slaves are distinguishabledevide such slaves. This number is a serial number allocated during manufacture, coded with 8 BCD packed digits (4 Byte) like the identification number, and thus runs from 00000000 to 99999999.
The following figure shows the structure of an enhanced selection telegram released by the master.

68h	11h	11h	68h	53h	FDh	52h	ID1-4	Man1-2	Gen	Med	0Ch	78h	Fab1-4	CS	16h

Fig. 13 *Structure of a telegram for enhanced selection (mode 1)*

After the field medium the new data is given in form of a structured datarecord with DIF=0Ch and VIF=78h. Parts of the fabrication number (Fab1..Fab4) can be occupied with wildcards (Fh).
If a fabrication number exists the slave should add this data to the variable data blocks in every RSP-UD telegram. If the fabrication number and enhanced selection is not implemented in a slave this device will not confirm the enhanced selection telegram and will be deselected.
Enhanced selection should be used only if the normal kind of selection is not successful.

Appendix G: References

[1] Färber, G. : Bussysteme, R.Oldenbourg Verlag München Wien, 1987

[2] Gabele, E., Kroll, M., Kreft, W. : Kommunikation in Rechnernetzen, Springer Verlag Heidelberg, 1991

[3] Steffens, Andreas : Diplomarbeit " Der M-Bus - Eigenschaften und Anwendungen", University of Paderborn, Department of Physics, 1992

[4] Texas Instruments Deutschland GmbH : Data Sheet TSS 721, 1993

~~[5] Texas Instruments Deutschland GmbH : Seminar Material, M-Bus Workshop, 1992~~

~~[6] Ziegler, Horst : Seminar Material, M-Bus Workshop, 1992~~

[57] IEC 870-5-1 : Telecontrol Equipment and Systems, Part 5 Transmission Protocols, Section One - Transmission Frame Formats, 1990

[68] IEC 870-5-2 : Telecontrol Equipment and Systems, Part 5 Transmission Protocols, Section Two - Link Transmission Procedures, 1992

~~[9] EN1434-3: Heat Meters, Part 3 Data Exchange and Interface, 1997~~ ♣

~~[10] Aquametro AG Therwil : M-Bus Automatic Slave Recognition with Wildcard Algorithm, 1992~~

~~[11] Papenheim, Andreas: Diplomarbeit " Anwendungsbeispiele für den M-Bus", University of Paderborn, Department of Physics, 1993~~

[712] Texas Instruments Deutschland GmbH: Applications Report "Designing Applications for the Meter-Bus", 1994 (translation of reference [11])

[813] Ziegler, Horst; Froschermeier Günther: "M-Bus: Die Meßbus-Alternative", Elektronik 16/1993

Note that a WWW-server operated by the m-bus-user group at „http://www.m-bus.com“ provides a forum for up to date information on the M-bus.