

KMP

Kamstrup Heat Meter Protocol

MULTICAL[®] 601 and MULTICAL[®] 801

Protocol description

The Kamstrup Meter Protocol (KMP) description describes the internal protocols used in e.g. MULTICAL[®] 601 and 801, Heat and Cooling meters. The KMP description contains information necessary to connect MULTICAL[®] to a third-party system, by serial data communication.

While the KMP description is believed to be accurate at the time of issue, Kamstrup A/S does not guarantee the accuracy of the document, nor will it be held responsible for damages of any kind, direct or indirect, which may result from the use of the KMP description. Further, you understand that the protocols described in the KMP description are subject to change without notice.

The KMP Protocol description is provided free of charge from Kamstrup A/S. It is, however, copyrighted and must not be copied. Please feel free to order your personal copy from Kamstrup A/S.

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.:LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 1

1 Introduction

This protocol is designed to handle point-to-point communication in a master / slave bus system. The main application for this protocol is data readout of Kamstrup heat meters, MULTICAL® 601 and MULTICAL® 801.

The data reliability is improved by introducing 16 bit CRC (Cyclic Redundancy Check).

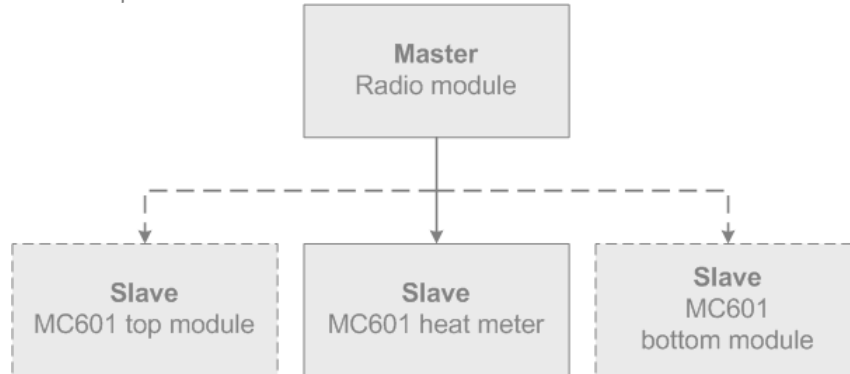
Data read out is made more flexible by introducing specific register read out. Registers will be represented by at register identification value. A register read out is associated with unit and exponent.

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.:LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 2

2 Communication format

The basic KMP communication is based on a master/slave bus communication (see 3.2 for addressing), where the meter (and most "feature" modules) is the slave, and any meter communication hardware is the master.

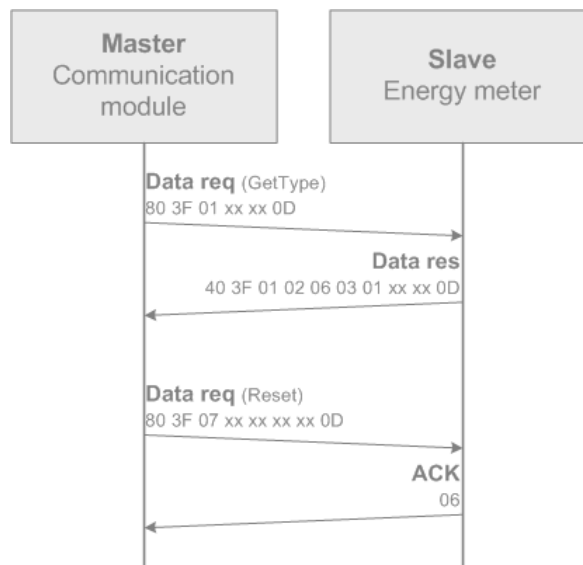
An example with a Kamstrup radio module as the master:



Basic KMP communication is always initiated from the master by a data request frame.

The meter can then respond either by a data response frame or a single application acknowledge character (ACK = 06h).

Example:



In addition to this, the meter can in some cases send a single character (= 00h) without being requested. This should be ignored.

2.1 MULTICAL timing sequence

The MULTICAL® (MC601 and MC801) works with a communication timeout of 1.6 second.

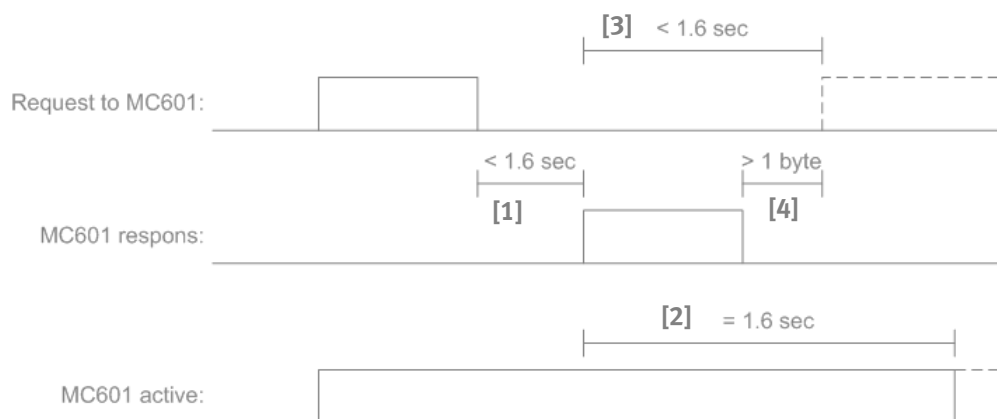
The meter gets active when it receives the first byte of a request frame. The timeout in between bytes in a request frame is 27 milliseconds.

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.:LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 3

The time, from the last byte in the request is send to the meter, to the first byte in the response is received from the meter, is less than 1.6 second (see [1] in the figure below).

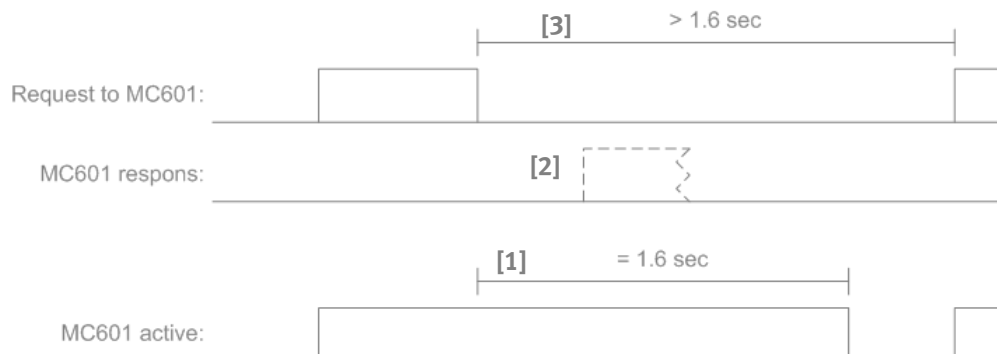
The meter then stays active in 1.6 second from the first byte of the response is send (independently of the response communication time) before going to sleep (see [2] in the figure below).

To optimize the communication time any following request has to be send less than 1.6 second after the first byte in the last response (see [3] in the figure below) to keep the meter from going “to sleep”. The time between the last byte in the last response is received to the first byte in the new request is send should be at least 1 ‘byte time’ (≈ 9 milliseconds at 1200 baud) though (see [4] in the figure below).



If the meter receives data, but the data is invalid or the meter is too busy, the meter stays active for 1.6 second after the last byte is received (see [1] in the figure below), in which any following request is discharge.

If the meter has not started the response frame after 1.6 second (or a response is received but the CRC is wrong – see [2] in the figure below). Then the transmission of the first byte of any following request frames should wait at least 1.6 second (see [3] in the figure below) for the meter to be ready for a new request.



2.2 Communication via 6705 top module

The 6705 top module gives a 2nd priority access to MULTICAL® 601.

It supports the same KMP communication and uses the same destination address as MULTICAL® 601.

However, the supported application commands are limited to 01h, 02h and 10h.

To distinguish a connection through 6705 top module from one directly at a MULTICAL® 601, the meter type returned by application command “GetType” is different (see chapter 6.2.1).

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 4

3 Data frame format

The data frame is based on the OSI model. In this protocol, only three layers of the OSI model are used. These are the physical layer, data link and the application layer. The figure below show how each of the layers (shown as shaded) is related to the data frame.

Field name	Start byte	Destination address	CID	Data	CRC	Stop byte
Number of bytes in each field	1	1	1	0-?	2	1
OSI – layer				Application layer		
				Data link layer		
				Physical layer		

In some communication examples only the APL is shown (CID and Data).

3.1 Physical layer

The protocol is based on half duplex serial asynchrony communication with the set-up: 8 data bit, no parity bit and 2 stop bits. Standard data bit rate is 1200 baud for both transmitted and received data.

Data is transmitted byte wise in a binary data format. 8 data bit represent one byte of data.

The physical layer in the data frame is used to synchronize and terminate data transmission. For this purpose the physical layer uses a unique start and stop byte.

Different start bytes values are used in frames to and from the meter. This is done to make an easy identification of the frame start and frame direction in communication where TXD is 'echoed' to the RXD line – e.g. at communication via the IR readout head.

Start byte in a data frame to the meter = 80h
Start byte in a data frame from the meter = 40h

Stop byte = 0Dh

⚠ Note that the KMP protocol also supports an ACK byte (0x06h) used some commands as a reply for successful execution. The ACK is an APL level acknowledge, but send as a single byte – Without start, CRC or stop bytes.

The physical layer uses 'Byte stuffing' to compensate for byte values reserved as start, stop and acknowledge. The method is to substitute the reserved bytes values with a pair of byte values. In this protocol the substitution shown below is used:

80h	→	1Bh, 7Fh	=	1Bh, NOT(80h)
40h	→	1Bh, BFh	=	1Bh, NOT(40h)
0Dh	→	1Bh, F2h	=	1Bh, NOT(0Dh)
06h	→	1Bh, F9h	=	1Bh, NOT(06h)
1Bh	→	1Bh, E4h	=	1Bh, NOT(1Bh)

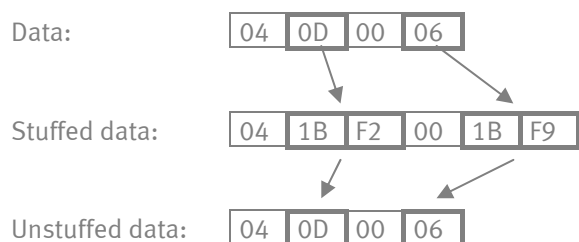
Note that byte stuffing can increase the length of the physical layer by 100% compared to the data link layer.

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 5

Unstuffing data is done by removing the byte '1Bh' and the following byte and make a substitution according to the table below:

1Bh, 7Fh	→	80h	=	NOT(7Fh)
1Bh, BFh	→	40h	=	NOT(BFh)
1Bh, F2h	→	0Dh	=	NOT(F2h)
1Bh, F9h	→	06h	=	NOT(F9h)
1Bh, E4h	→	1Bh	=	NOT(E4h)

Example:



3.2 Data link layer

The destination address is included in order to prepare a future enhanced version of the protocol. For heat meters the destination address is 3Fh. The logger top module use 7Fh and the logger base module use BFh.

Included in the data link layer is a CRC with reference to the CCITT-standard using the polynomial 1021h. Only deviation from the standard is the initial value, which is 0000h instead of FFFFh.

The CRC result is calculated for destination address, CID and data. CRC is transmitted with MSByte first and LSByte last.

3.3 Application layer

The first byte of the application layer is the Command ID (CID). This also applies for a response data frame from the meter. The response CID will then equal the request CID. The number of bytes in the data field (if any) depends on the CID. The interpretation of the bytes in the data field also depends on the CID.

See chapter 6 for more details on CIDs.

Most data in the meter is handled in a KMP register format (see chapter 4.4).

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.:LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 6

4 Data formats

4.1 Basis formats

[] Specifies one byte data.

() Specifies a set of bytes – e.g. an integer.

The value specified by () is of the length 2, 3, 4 ... bytes.

The order of most and least significant bytes is independent of the number of bytes. Most significant byte is transmitted first.

Examples:

(data) = [data 1][data 2][data 3]

[data 1] is the most significant byte, [data 3] is the least significant byte.

4.2 Floating point data format

This data format applies to transmission of data registers.

One value consists of the following bytes:

[number of bytes][sign+exponent] (integer)

(integer) is the register data value. The length of the integer value is specified by [number of bytes].

[sign+exponent] is an 8-bit value that specifies the sign of the data value and sign and value of the exponent. The meaning of the individual bits in the [sign+exponent] byte is shown below:

Sign + exponent							
Bit 7	6	5	4	3	2	1	0
SI	SE	E5	E4	E3	E2	E1	E0

$$\text{Floating point value} = -1^{SI} \cdot (\text{integer}) \cdot 10^{-1^{SE} \cdot \text{exponent}}$$

Examples:

-123.45 = 04h, C2h, 0h, 0h, 30h, 39h

87654321*10³ = 04h, 03h, 05h, 39h, 7Fh, B1h

255*10³ = 01h, 03h, FFh

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 7

4.3 Logger timestamp data format

(Time) is a 64bits/8bytes integer. It contains a timestamp in the following format:

$$(Time) = ([YY][MM][DD][hh][mm][ss][WK][Info])$$

With the following definitions:

Byte	Parameter	Range(dec)	Definition
0	[YY]	0-99	Year –Year 2000: [YY] = 0d, Year 2099: [YY] = 99d
1	[MM]	1-12	Month
2	[DD]	1-(28)31	Day
3	[hh]	0-23	Hours
4	[mm]	0-59	Minutes
5	[ss]	0-59	Seconds
6	[WK]	0	Weekday – not used
7	[Info]	0	Info byte – not used

Examples:

2007-05-17 10:30:00 ⇔ 07|05|11|0A|1E|00|00|00
2091-12-31 23:59:59 ⇔ 5B|0C|1F|17|3B|3B|00|00

4.4 KMP register data format

Most of the data in the meter is implemented as registers. A register description consists of three parts shown below and explained in the following subchapters.

Register ID		Register format			Register value			
0	1	2	3	4	5	6	7	8
xxxx		xxxxxx			xxxxxxxx			

4.4.1 Register ID (RID)

Meter registers are associated with unique register identification numbers (RID). The RID is a 16-bit value. It is used in commands for reading actual and historical data.

Examples:

Register ID		Register
0001h	= 1d	Energy A14
0048h	= 72d	Mass1
03E9h	= 1001d	Serial number

For a list of RIDs supported in MC601 see chapter 5.2.

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.:LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 8

4.4.2 Register format

The register format consists of three bytes; Unit, NumberOfBytes(NoB) and SignExp(SiEx):

Unit	NoB	SiEx
0	1	2
xx	xx	xx

These parameters are defined as the following:

Parameter	Description
Unit	8bit – Register unit Specifies the unit of the register data value. Example: 3d = MWh 40d = m ³ 47d = clock (hh:mm:ss) See chapter 5.1 for more units.
NoB	8bit – Number Of Bytes (n) Specifies the byte size of the register data value. This parameter is always 4 in MC601 and MC801.
SiEx	8bit – Sign and Exponent See ‘sign+exponent’ in chapter 4.2.

4.4.3 Register value

The register value is defined as a 4 byte integer as described in chapter 4.1.

4.4.4 Examples

-123.45 kCal =>

Unit	NoB	SiEx	Value			
0	1	2	3	4	5	6
0A	04	C2	00003039			

87654321*10³ l =>

Unit	NoB	SiEx	Value			
0	1	2	3	4	5	6
27	04	03	05397FB1			

255*10³ m³ =>

Unit	NoB	SiEx	Value
0	1	2	3
28	01	03	FF

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 9

5 Units and ID's

5.1 RegisterUnit

Byte, which describes the measuring units:

Unit ID		Unit name	Description
hex	dec		
01	1	Wh	
02	2	kWh	
03	3	MWh	
08	8	Gj	
0C	12	Gcal	
16	22	kW	
17	23	MW	
25	37	C	
26	38	K	
27	39	l	
28	40	m ³	
29	41	l/h	
2A	42	m ³ /h	
2B	43	m ³ xC	
2C	44	ton	
2D	45	ton/h	
2E	46	h	
2F	47	clock	hh:mm:ss (dec) Examples: 10:30:00 ⇔ 00 01 92 58 23:59:59 ⇔ 00 03 99 B7
30	48	date1	yy:mm:dd (dec) Examples: 06:08:24 ⇔ 00 00 ED 98 99:12:31 ⇔ 00 0F 1F FF
32	50	date3	mm:dd (dec) Examples: 08:24 ⇔ 00 00 03 38 12:31 ⇔ 00 00 04 CF
33	51	number	
34	52	bar	

5.2 MULTICAL® 601 Register ID's

Register ID		Register name	Description
hex	dec		
03EB	1003	DATE	Current date (YYMMDD)
003C	60	E1	Energy register 1: Heat energy
005E	94	E2	Energy register 2: Control energy
003F	63	E3	Energy register 3: Cooling energy
003D	61	E4	Energy register 4: Flow energy
003E	62	E5	Energy register 5: Return flow energy
005F	95	E6	Energy register 6: Tap water energy
0060	96	E7	Energy register 7: Heat energy Y
0061	97	E8	Energy register 8: [m ³ • T1]
006E	110	E9	Energy register 9: [m ³ • T2]
0040	64	TA2	Tariff register 2
0041	65	TA3	Tariff register 3
0044	68	V1	Volume register V1

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 10

Register ID		Register name	Description
hex	dec		
0045	69	V2	Volume register V2
0054	84	VA	Input register VA
0055	85	VB	Input register VB
0048	72	M1	Mass register V1
0049	73	M2	Mass register V2
03EC	1004	HR	Operational hour counter
0071	113	INFOEVENT	Info-event counter
03EA	1002	CLOCK	Current time (hhmmss)
0063	99	INFO	Infocode register, current
0056	86	T1	Current flow temperature
0057	87	T2	Current return flow temperature
0058	88	T3	Current temperature T3
007A	122	T4	Current temperature T4
0059	89	T1-T2	Current temperature difference
005B	91	P1	Pressure in flow
005C	92	P2	Pressure in return flow
004A	74	FLOW1	Current flow in flow
004B	75	FLOW2	Current flow in return flow
0050	80	EFFEKT1	Current power calculated on the basis of V1-T1-T2
007B	123	MAX FLOW1DATE/ÅR	Date for max. this year
007C	124	MAX FLOW1/ÅR	Max. value this year
007D	125	MIN FLOW1DATE/ÅR	Date for min. this year
007E	126	MIN FLOW1/ÅR	Min. value this year
007F	127	MAX EFFEKT1DATE/ÅR	Date for max. this year
0080	128	MAX EFFEKT1/ÅR	Max. value this year
0081	129	MIN EFFEKT1DATE/ÅR	Date for min. this year
0082	130	MIN EFFEKT1/ÅR	Min. value this year
008A	138	MAX FLOW1DATE/MÅNED	Date for max. this year
008B	139	MAX FLOW1/MÅNED	Max. value this year
008C	140	MIN FLOW1DATE/MÅNED	Date for min. this month
008D	141	MIN FLOW1/MÅNED	Min. value this month
008E	142	MAX EFFEKT1DATE/MÅNED	Date for max. this month
008F	143	MAX EFFEKT1/MÅNED	Max. value this month
0090	144	MIN EFFEKT1DATE/MÅNED	Date for min. this month
0091	145	MIN EFFEKT1/MÅNED	Min. value this month
0092	146	AVR T1/ÅR	Year-to-date average for T1
0093	147	AVR T2/ÅR	Year-to-date average for T2
0095	149	AVR T1/MÅNED	Month-to-date average for T1
0096	150	AVR T2/MÅNED	Month-to-date average for T2
0042	66	TL2	Tariff limit 2
0043	67	TL3	Tariff limit 3
0062	98	XDAY	Target date (reading date)
0098	152	PROG NO	Program no. ABCCCCC
0099	153	CONFIG NO 1	Config no. DDDEE
00A8	168	CONFIG NO 2	Config no. FFGGMN
03E9	1001	SERIE NO	Serial no. (unique number for each meter)
0070	112	METER NO 2	Customer number (8 most important digits)
03F2	1010	METER NO 1	Customer number (8 less important digits)
0072	114	METER NO VA	Meter no. for VA
0068	104	METER NO VB	Meter no. for VB
03ED	1005	METER TYPE	Software edition
009A	154	CHECK SUM 1	Software check sum
009B	155	HIGH RES	High-resolution energy register for testing purposes
009D	157	TOPMODUL ID	ID number for top module (only mc 601)
009E	158	BOTMODUL ID	ID number for base module
00AF	175	ERRORHOURCOUNTER	Error hour counter

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.:LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 11

Register ID		Register name	Description
hex	dec		
00EA	234	INA LITERIMP	Liter/imp value for input A
00EB	235	INB LITERIMP	Liter/imp value for input B

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.:LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 12

5.3 MULTICAL® 801 Register ID's

Additional to the register Id's listed above, MULTICAL® 801 include the following registers.

hex	dec		
	171	BOTMODULE ID2	ID number for base module 2
	172	EXTMODULE ID	ID number for external base module
	184	MbusPriAdrMod1	Primary Mbus adress for module 1
	185	MbusSekAdrMod1	Sekundary Mbus adress for module 1
	218	MbusPriAdrMod2	Primary Mbus adress for module 2
	219	MbusSekAdrMod2	Sekundary Mbus adress for module 2
	220	MbusPriAdrExtMod	Primary Mbus adress for external module
	221	MbusSekAdrExtMod	Sekundary Mbus adress for external module
	157	FEACPU ID	Feature cpu ID (dummy topmodule id)

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.:LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 13

6 General application commands

6.1 Command overview

The following general commands are specified for the MULTICAL:

CID	Command
01h	GetType
02h	GetSerialNo
09h *	SetClock
10h	GetRegister

* Not accessible through 67-05 top module.

The above and the following commands are specified for the MC601 logger modules (67-0B top module and 67-00-22 base module) and for MC801:

CID	Command
A0h	GetLogTimePresent
A1h	GetLogLastPresent
A2h	GetLogIDPresent
A3h	GetLogTimePast
9Bh	GetEventStatus
9Ch	ClearEventStatus
11h	PutRegister

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 14

6.2 Description of general application commands

6.2.1 GetType - CID = 01h

Function: This command returns identification of the type of meter and software revision.

The document 'KPM Product Identification' lists meter types and the corresponding identification values.

Request: [80h] [destination address] [01h] (CRC) [0Dh]

Response: [40h] [destination address] [01h] (MT) (SW revision) (CRC) [0Dh]

The meter type (MT) is a 16-bit value.

Meter type:

- 0001h = Multical 601 – pt500
- 0004h = Multical 601 – pt100
- 0005h = Multical 601 – pt500, 4-wire
- 1501h = Multical 601 – pt500, through 67-05 top module
- 1504h = Multical 601 – pt100, through 67-05 top module
- 1505h = Multical 601 – pt500, 4-wire, through 67-05 top module
- 09xx = A general top module (use dest addr 0x7F)
- 16xx = A general base module (use dest addr 0xBF)

The software revision is a 16 bit value: (SW revision) = [data1] [data 2]

[data 1] identify revision letter:

- 01h = A
- 02h = B
- 03h = C
- 04h = D
- 05h = E
- ...

[data 2] identify revision number:

- 00h = 0
- 01h = 1
- 02h = 2
- 03h = 3
- 04h = 4
- ...

Example

Request:

Application Layer:

01

CID

Data Link Layer:

3F 01

Destination address added

3F 01 05 8A

CRC added (No byte stuffing needed)

Physical Layer:

80 3F 01 05 8A 0D

Start and stop byte added

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 15

Response:

Physical Layer:

40	3F	01	00	04	1B	F9	01	26	99	0D
----	----	----	----	----	----	----	----	----	----	----

Data Link Layer:

3F	01	00	04	1B	F9	01	26	99
----	----	----	----	----	----	----	----	----

Start and stop byte removed

3F	01	00	04	06	01	26	99
----	----	----	----	----	----	----	----

Destuffing

3F	01	00	04	06	01	26	99	00	00
----	----	----	----	----	----	----	----	----	----

CRC calculated

3F	01	00	04	06	01
----	----	----	----	----	----

New and old CRC removed

Application Layer:

01	00	04	06	01
----	----	----	----	----

Destination address removed

		0601h = SW revision F1
		0004h = Multical 601 – pt100
		01h = GetType

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.:LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 16

6.2.2 GetSerialNo - CID = 02h

Function: This command returns the serial number of the meter.
The serial number is a 32-bit number. Every heat meter has a unique serial number and these meters will not have identical serial numbers.
This number is used as an unambiguous identification of the meter.

Request: [80h] [destination address] **[02h]** (CRC) [0Dh]

Response: [40h] [destination address] **[02h] (SN)** (CRC) [0Dh]

The serial number (S/N) is a 32-bit value.

Example

Request:

Application Layer:

02

CID

Data Link Layer:

3F 02

Destination address added

3F 02 35 E9

CRC added (No byte stuffing needed)

Physical Layer:

80 3F 02 35 E9 0D

Start and stop byte added

Response:

Physical Layer:

40 3F 02 01 23 45 67 E9 56 0D

Data Link Layer:

3F 02 01 23 45 67 E9 56

Start and stop byte removed

3F 02 01 23 45 67 E9 56 00 00

CRC calculated (No destuffing needed)

3F 02 01 23 45 67

New and old CRC removed

Application Layer:

02 01 23 45 67

Destination address removed

01234567h = S/N: 19088743d
01h = GetSerialNo

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 17

6.2.3 SetClock - CID = 09h

Function: This command sets the meter clock.
(Not accessible through 67-05 top module.)

Request: [80h] [destination address] [09h] (date) (time) (CRC) [0Dh]

The (date) is a 32 bit binary value formatted as YY-MM-DD.

Example: The date 25th august 2004 results in (date) = 40825d

The (time) is a 32 bit binary value formatted as HH:MM:SS.

Example: The time 23:59:59 results in (time) = 235959d

Example: The time 00:00:00 results in (time) = 0d

Validation demands:

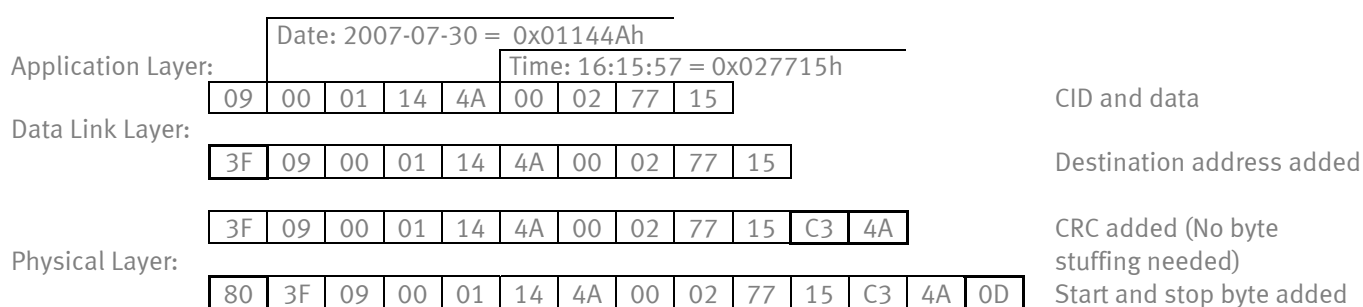
YY: 0 – 99d
MM: 1 – 12d
DD: 1 – X X depends on month and leap year

HH: 0 – 23d
MM: 0 – 59d
SS: 0 – 59d

Response: An application acknowledge is returned if (date) and (time) are valid.

Example

Request:



Response:

Physical Layer: [06] Acknowledge (No start/stop, CRC, destination address or CID)

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 18

6.2.4 GetRegister - CID = 10h

Function: This command returns a variable set of registers.

Request: [80h] [destination address] ...
[10h] [number of registers] (register 1 ID) (register 2 ID) ... (register n ID) ...
 (CRC) [0Dh]

[number of registers] specifies the number of registers to be returned in the response and thus the number of following register codes.

A maximal number of 8 registers can be read with one request.
 Thus [number of registers] = 1 to 8.

(register 1 ID) ... (register n ID) are 16 bit register identification codes and specifies registers to be read out.

Response: [40h] [destination address] ...
[10h] ...
(register 1 ID) [register 1 unit] (register 1 format) (register 1 value) ...
(register 2 ID) [register 2 unit] (register 2 format) (register 2 value) ...
(register n ID) [register n unit] (register n format) (register n value) ...
 (CRC) [0Dh]

[register unit] is a unit identification code and specifies units.

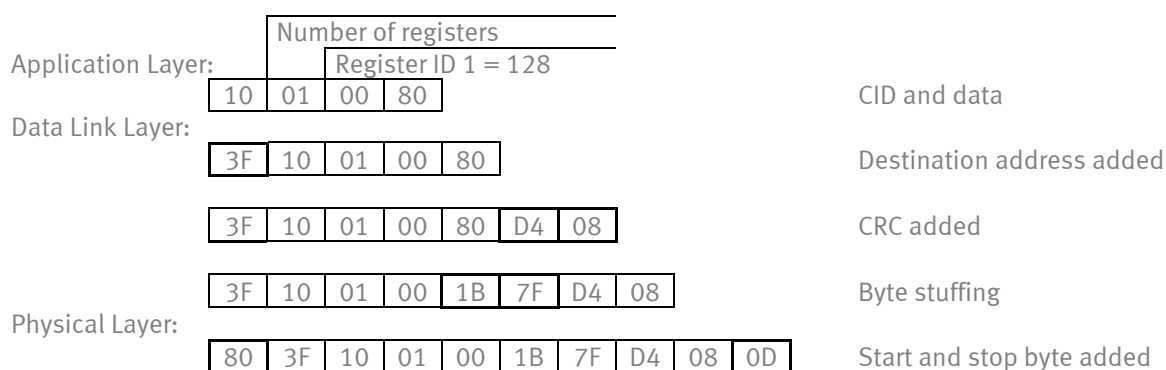
(register format) described in detail in section 4.2, page 7, is the floating-point data format consisting of:

[number of bytes] [sign+exponent]

An occurrence of an unsupported register in the request is handled by omitting the register in the response. This means that a request for ex. 5 registers may result in a response with less than 5 registers – in principle no registers at all.

Example

Request:



Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 19

Response:

Physical Layer:

40	3F	10	00	1B	7F	16	04	11	01	2A	F0	24	F3	8A	0D
----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

Data Link Layer:

3F	10	00	1B	7F	16	04	11	01	2A	F0	24	F3	8A
----	----	----	----	----	----	----	----	----	----	----	----	----	----

Start and stop byte removed

3F	10	00	80	16	04	11	01	2A	F0	24	F3	8A
----	----	----	----	----	----	----	----	----	----	----	----	----

Destuffing

3F	10	00	80	16	04	11	01	2A	F0	24	63	03	00	00
----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

CRC calculated

3F	10	00	80	16	04	11	01	2A	F0	24
----	----	----	----	----	----	----	----	----	----	----

New and old CRC removed

Application Layer:

10	00	80	16	04	11	01	2A	F0	24
----	----	----	----	----	----	----	----	----	----

Destination address removed

					012AF024h => int = 19.591.204				
					11h => SI = 0 ; SE = 0 ; exp = 17 => $-1^0 \cdot \text{int} \cdot 10^{-1^0 \cdot 17}$				
					04h = 4 bytes				
					16h = kW				
					0080h = MAX EFFEKT1/ÅR				
					10h = GetRegister				

MAX EFFEKT1/ÅR (Register ID 128) =
 $-1^0 \cdot 19.591.204 \cdot 10^{-1^0 \cdot 17} \text{ kW} = 19.591.204 \cdot 10^{17} \text{ kW}$

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 20

6.3 Description of logger application commands

The description in this section covers the loggermodules 670B000000 and 6700220000 for MULTICAL®601 and the internal logger in MULTICAL®801.

- ⚠ Remember to use the right destination address when communicating with the logger (See Data link layer, section 3.2, page 6).
- 670B is a MC601 top module and replies on 0x7F.
- 670022 is a MC601 base module and replies on 0xBF.
- MC801 is a meter and therefore uses 0x3F however the logger commands in this section is accessed using 0x7F (Logger functionality in MC801 is seen as a virtual top module).

The following terms and expressions are used for documentation of a logger (module or meter):

Term	Explanation
Device	A device is the module/meter in the field.
Log	A specific log contains data logged in records.
Record	A specific log record contains a copy of each of the registers covered by the log.


Following log example is referenced in the four readout commands in subchapter 6.3.1 to 6.3.4:

Log ID: 01 (interval logger)			
Log record ID	Log record timestamp	Logged register values	
		003C	0044
...
10F2h	2007-05-24 10:00:00	1D8B83A0	9DBAD708
10F3h	2007-05-24 11:00:00	FF068662	E993DC05
10F4h	2007-05-24 12:00:00	B50A5C5E	93E05047
10F5h	2007-05-24 13:00:00	CF065396	F2584C96
10F6h	2007-05-24 14:00:00	A38FB479	2E44778E
10F7h	2007-05-24 15:00:00	2F4CED5F	931D90A5
10F8h	2007-05-24 16:00:00	04AC8C55	B5700C78
10F9h	2007-05-24 17:00:00	6067E035	7CE685E5
10FAh	2007-05-24 18:00:00	7BA2B68E	F3947871
10FBh	2007-05-24 19:00:00	717F72A0	8C7D6E62
10FCh	2007-05-24 20:00:00	99F948C3	EA02904C
10FDh	2007-05-24 21:00:00	331CD991	5D43BB58
10FEh	2007-05-24 22:00:00	A835284A	F4B07BD8
10FFh	2007-05-24 23:00:00	96F3547C	B3A8F8B2
1100h	2007-05-25 00:00:00	DA24B37C	12DD6D6B
1101h	2007-05-25 01:00:00	0EF8D8A0	4388026C
1102h	2007-05-25 02:00:00	0CD83E6A	54F6D02E
...

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 21

Each parameter in the examples is referenced with a certain abbreviation.

The abbreviations of the most common command parameters in this document are described in the following table:

Parameter abbreviation	Parameter description
LID	8bit – Log ID. The ID of a specific log. A logger device contains one or more logs. Each log in a device has a unique consecutively number; the log ID.  The log ID is not unique between devices!
NRgs	8bit – Number of registers Number of registers requested – also denoted ‘n’. n is limited to the range 1-8 to keep the response frame in a manageable size.
Reg ID 1-n	16bit – Register ID 1-n Register ID for each of the n registers requested (see paragraph 4.4.1). For a list of registers supported see paragraph 5.2.
NRcs	8bit – Number of records Number of records requested – also denoted ‘m’. If the requested response is larger than MaxL or communication limits in the device, then the number of records is reduced in the response (see response example in paragraph 6.3.1).
MaxL	8bit – Max APL Length The maximum APL length in the response. This is used to limit the response size if for instance the receiver of the response has a limited communication buffer. If the requested response exceeds this limit the NRcs is reduced before creating the response (see response example in paragraph 6.3.1).
Time	64 bit – Timestamp The timestamp used as search criteria in time based readout commands. For a description of the KMP timestamp format see paragraph 4.3.
NRtR	8bit – Number of returned registers Number of registers included in the response – also denoted ‘N’. If all registers requested exist in the log this number equals NRgs. Any register ID not included in the entity list will be left out of the response.
RecID 1-M	16bit – Record ID 1-M Record ID for each of the records in the response. M denotes the amount of records in the response. If all records requested exist in the log, and the response size is within the limits (see the description of NRcs), this number equals NRcs. The records is read out and placed in the response frame in a consecutive order (either descending or ascending depending on the command) with the record ID found in the search placed first.
Reg ID 1-N format	24bit – Reg ID 1-N format The register format of each register included in the frame. For a description of the KMP register format see paragraph 4.4.2.
Reg ID 1-N value	8bit – Reg ID n value The register value of each register included in the frame (see paragraph 4.4.3).
LRORecID	16bit – LastReadOutRecord ID The ID of the record included in the response which was readout last/most recent from the log. It will always correspond to the ID of the last record in a specific response, because the last record in the response is the last record readout from the log. Depending on the command (whether it read out towards present or towards past) this will correspond to either the newest/highest or the oldest/lowest record ID from the log included in the response.

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.:LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 22

Parameter abbreviation	Parameter description																		
NewRec ID	16bit – NewestRecord ID The ID of the newest record contained in the log. This is (only) changed when the log creates a new record – and then set to the ID of this new record.																		
INFO	8bit – Info An info byte indicating errors concerning log readout. The byte is interpreted as a collection of 8 single bit flags. 0 = No error ; 1 = Error occurred <table border="1"> <thead> <tr> <th>Bit</th><th>Error</th></tr> </thead> <tbody> <tr> <td>0</td><td>Log is empty.</td></tr> <tr> <td>1</td><td><i>Not used</i></td></tr> <tr> <td>2</td><td>Requested record ID or timestamp is out of the current range of records in the log.</td></tr> <tr> <td>3</td><td>MaxL was reached. One or more records were excluded in the response.</td></tr> <tr> <td>4</td><td>The end of the log was reached. One or more records were excluded in the response. When reading towards present this will occur when the newest record in the log is reached. When reading towards past this will occur when the oldest record in the log is reached.</td></tr> <tr> <td>5</td><td><i>Not used</i></td></tr> <tr> <td>6</td><td><i>Not used</i></td></tr> <tr> <td>7</td><td><i>Not used</i></td></tr> </tbody> </table>	Bit	Error	0	Log is empty.	1	<i>Not used</i>	2	Requested record ID or timestamp is out of the current range of records in the log.	3	MaxL was reached. One or more records were excluded in the response.	4	The end of the log was reached. One or more records were excluded in the response. When reading towards present this will occur when the newest record in the log is reached. When reading towards past this will occur when the oldest record in the log is reached.	5	<i>Not used</i>	6	<i>Not used</i>	7	<i>Not used</i>
Bit	Error																		
0	Log is empty.																		
1	<i>Not used</i>																		
2	Requested record ID or timestamp is out of the current range of records in the log.																		
3	MaxL was reached. One or more records were excluded in the response.																		
4	The end of the log was reached. One or more records were excluded in the response. When reading towards present this will occur when the newest record in the log is reached. When reading towards past this will occur when the oldest record in the log is reached.																		
5	<i>Not used</i>																		
6	<i>Not used</i>																		
7	<i>Not used</i>																		

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.:LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 23

6.3.1 A0h GetLogTimePresent (GetLogFromTimeTowardsPresent)

Function: Log readout from specified 'timestamp' towards 'now'.

The log is searched for the oldest record with a timestamp equal to or newer than the requested timestamp.

This record and the following records (with higher/ascending record ID's) is read out.

⚠ If the clock in the device is changed, an overlap of timestamps can occur.

⚠ If the response contains a log error (if any bit in the INFO byte is set), the response package length and application content can vary. The CID and INFO byte is always included and placed as respectively the first and the last byte in the application layer.

Request: [A0h] [Log ID] [Number of registers] (Reg ID 1) (Reg ID 2) ...
... (Reg ID n) [Number of records] [Max APL length] (Time)

Parameter	Description
LID	8bit – Log ID* The ID of a specific log.
NRgs	8bit – Number of registers* Number of registers requested – also denoted 'n'.
Reg ID 1-n	16bit – Register ID 1-n* Register ID for each of the n registers requested.
NRcs	8bit – Number of records* Number of records requested – also denoted 'm'.
MaxL	8bit – Max APL Length* The maximum APL length in the response.
Time	64 bit – Timestamp* The timestamp used as search criteria in time based readout commands.

* See abbreviation list at paragraph 6.3

Example:

Two registers (E1 and V1) from 2007-05-24 10:30:00 and 3 records forward

↓

CID	LID	NRgs	Reg ID 1	Reg ID 2
0	1	2	3	4
A0	01	02	003C	0044

NRcs	MaxL	Time
7	8	9
03	80	xxxx001E0A180507

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 24

Response: [A0h] [Log type ID] [Number of returned registers]
 (RecordID 1) (Reg ID 1) (Reg ID 1 format: [Unit] [Number of bytes] [SignExp]) (Reg ID 1 value)
 (Reg ID 2) (Reg ID 2 format: [Unit] [Number of bytes] [SignExp]) (Reg ID 2 value)
 ...
 (Reg ID *n*) (Reg ID *n* format: [Unit] [Number of bytes] [SignExp]) (Reg ID *n* value)
 (RecordID 2) (Reg ID 1 value) (Reg ID 2 value) ... (Reg ID *n* value)
 ...
 (RecordID *M*) (Reg ID 1 value) (Reg ID 2 value) ... (Reg ID *n* value)
 (LastReadOutRecord ID) (NewestRecord ID) [Info]

Parameter	Description
NRtR	8bit – Number of returned registers* Number of registers included in the response – also denoted ‘N’.
RecID 1- <i>M</i>	16bit – Record ID 1- <i>M</i> * Record ID for each of the records in the response.
Reg ID 1- <i>N</i> format	24bit – Reg ID 1- <i>N</i> format* The register format of each register included in the response. See section 4.2, page 7, for details on the floating-point data format.
Reg ID 1- <i>N</i> value	xxbit ¹⁾ – Reg ID <i>n</i> value* The register value of each register included in the response for each log record.
LRORecID	16bit – LastReadOutRecord ID* The ID of the record included in the response which was readout last/most recent from the log.
NewRec ID	16bit – NewestRecord ID* The ID of the newest record contained in the log.
INFO	8bit – Info* An info byte indicating errors concerning log readout.

* See abbreviation list at paragraph 6.3

Example (see log record examples at paragraph 6.3):

CID	LTID	NRtR	RecID 1	Reg ID 1	Reg ID 1 format	Reg ID 1 value
0	1	2	3 4	5 6	7 8 9	10 11 12 13
A0	01	02	10F3	003C	080400	FF068662

Reg ID 2	Reg ID 2 format	Reg ID 2 value
14 15	16 17 18	19 20 21 22
0044	270400	E993DC05

RecID 2	Reg ID 1 value	Reg ID 2 value
23 24	25 26 27 28	29 30 31 32
10F4	B50A5C5E	93E05047

RecID 3	Reg ID 1 value	Reg ID 2 value
33 34	35 36 37 38	39 40 41 42
10F5	CF065396	F2584C96

LRORecID	NewRec ID	INFO
43 44	45 46	47
10F5	1102	00

↓

Last page: 36	Date for last revision: 2009-12-07	Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL
	EDB: GHA	Doc. No.: 5512-447
		Page: 25

Register ID	Record ID		
	10F3	10F4	10F5
	003C	FF068662	B50A5C5E
	0044	E993DC05	93E05047
			F2584C96

Expected size of the response, with NRcs and NRgs in consideration (SOF, DestAddr, CRC and EOF incl. – byte stuffing not incl.):

$$\begin{aligned}
 \text{APL length} &= (\text{CID} \leftrightarrow \text{NRtR}) + (\text{RecID } 1\text{-}M) + (\text{Reg ID } 1\text{-}N + \text{Reg ID } 1\text{-}N \text{ format}) + \\
 &\quad (M * \text{Reg ID } 1\text{-}N \text{ value}) + (\text{LRORecID} \leftrightarrow \text{INFO}) \\
 &\Rightarrow (3) + (2 * m) + (5 * n) + (4 * m * n) + (5) \\
 \text{Response length} &= (\text{SOF \& DestAddr}) + \text{APL length} + (\text{CRC \& EOF}) \\
 &\Rightarrow (2) + (3) + (2 * m) + (5 * n) + (4 * m * n) + (5) + (3)
 \end{aligned}$$

This result in the following table:

		<i>m</i> (Number of records - NRcs)																			
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
<i>n</i> (Number of registers pr. record - NRgs)	0	13	15	17	19	21	23	25	27	29	31	33	35	37	39	41	43	45	47	49	51
	1	18	24	30	36	42	48	54	60	66	72	78	84	90	96	102	108	114	120	126	132
	2	23	33	43	53	63	73	83	93	103	113	123	133	143	153	163	173	183	193	203	213
	3	28	42	56	70	84	98	112	126	140	154	168	182	196	210	224	238	252	266	280	294
	4	33	51	69	87	105	123	141	159	177	195	213	231	249	267	285	303	321	339	357	375
	5	38	60	82	104	126	148	170	192	214	236	258	280	302	324	346	368	390	412	434	456
	6	43	69	95	121	147	173	199	225	251	277	303	329	355	381	407	433	459	485	511	537
	7	48	78	108	138	168	198	228	258	288	318	348	378	408	438	468	498	528	558	588	618
	8	53	87	121	155	189	223	257	291	325	359	393	427	461	495	529	563	597	631	665	699

MaxL will overrule this and reduce the NRcs if the size is larger than MaxL after excluding non existing registers. The broken line indicates a limitation of 128 bytes in the response.

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 26

6.3.2 A1h GetLogLastPresent (GetLogFromLastReadTowardsPresent)

Function: Log readout from 'LRORecID' towards 'now'.

This command returns the records from the newest read record ('LRORecID') and forward – 'LRORecID' not included (see 6.3, 6.3.5 and 6.3.6 for examples).

On request, the record and the following/newer records, after the newest record already read out, are returned in the response.

If the amount of new records exceeds the limitations of the device response size, the oldest records in the interval is returned.

If the log contains no new records, a response without any records is returned.

⚠ This command is not intended for use with any other log readout commands. Other log readout commands can change the registration of which record, of the already readout records, is the newest. This can cause records to be missed by this command. The situation is handled product specific.

⚠ If the response contains a log error (if any bit in the INFO byte is set), the response package length and application content can vary. The CID and INFO bytes are always included and placed as respectively the first and the last byte in the application layer.

Request: [A1h] [Log type ID] [Number of registers] (Reg ID 1) (Reg ID 2) ...
... (Reg ID n) [Number of records] [Max APL length]

Parameter	Description
LID	8bit – Log ID* The ID of a specific log.
NRgs	8bit – Number of registers* Number of registers requested – also denoted 'n'.
Reg ID 1-n	16bit – Register ID 1-n* Register ID for each of the n registers requested.
NRcs	8bit – Number of records* Number of records requested – also denoted 'm'.
MaxL	8bit – Max APL Length* The maximum APL length in the response.

* See abbreviation list at paragraph 6.3

Example:

Two registers (E1 and V1) from LRORecID and 3 records forward



CID	LID	NRgs	Reg ID 1		Reg ID 2		NRcs	MaxL
0	1	2	3	4	5	6	7	8
A1	01	02	003C		0044		03	80

Response: The response frame is identical with the response frame of GetLogTimePresent (see 6.3.1 – response) when LRORecID is 10F2h – except for the CID, which is A1h.

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 27

6.3.3 A2h GetLogIDPresent (GetLogFromIDTowardsPresent)

Function: Log readout from specified 'Record ID Requested' towards 'now'.

The log is searched for a record with an ID equal to the requested record ID. This record and the following records (with higher/ascending record ID's) is read out.

⚠ If the requested ID doesn't exist, the readout starts at the following ID. Whether an info bit is set in this situation is product specific.

⚠ If the response contains a log error (if any bit in the INFO byte is set), the response package length and application content can vary. The CID and INFO bytes are always included and placed as respectively the first and the last byte in the application layer.

Request: [A2h] [Log type ID] [Number of registers] (Reg ID 1) (Reg ID 2) ...
... (Reg ID n) [Number of records] [Max APL length] (Record ID Requested)

Parameter	Description
LID	8bit – Log ID* The ID of a specific log.
NRgs	8bit – Number of registers* Number of registers requested – also denoted 'n'.
Reg ID 1-n	16bit – Register ID 1-n* Register ID for each of the n registers requested.
NRcs	8bit – Number of records* Number of records requested – also denoted 'm'.
MaxL	8bit – Max APL Length* The maximum APL length in the response.
RecIDReq	16bit – Record ID Requested The record ID requested as the first and oldest record included in the response.

* See abbreviation list at paragraph 6.3

Example:

Two registers (E1 and V1) from record 10F3h and 3 records forward



CID	LID	NRgs	Reg ID 1		Reg ID 2		NRcs	MaxL	RecIDReg	
0	1	2	3	4	5	6	7	8	9	10
A2	01	02	003C		0044		03	80	10F3	

Response: The response frame is identical with the response frame of GetLogTimePresent (see 6.3.1 – response) – except for the CID, which is A2h.

Last page: 36			Date for last revision: 2009-12-07				Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 28		


6.3.4 A3h GetLogTimePast (GetLogFromTimeTowardsPast)

Function: Log readout from specified 'timestamp' towards the 'past'.

The log is searched for the oldest record with a timestamp equal to or older than the requested timestamp.

The starting point in the log for the response is then the record ID found in the search minus the value of 'Record ID Offset'. This record and the following records (with lower/descending record ID's) are read out.

 If the clock in the device is changed, an overlap of timestamps can occur.

 If the response contains a log error (if any bit in the INFO byte is set), the response package length and application content can vary. The CID and INFO bytes are always included and placed as respectively the first and the last byte in the application layer.

Request: [A3h] [Log type ID] [Number of registers] (Reg ID 1) (Reg ID 2) ...
... (Reg ID n) [Number of records] [Max APL length] (Time) (Record ID Offset)

Parameter	Description
LID	8bit – Log ID* The ID of a specific log.
NRgs	8bit – Number of registers* Number of registers requested – also denoted 'n'.
Reg ID 1-n	16bit – Register ID 1-n* Register ID for each of the n registers requested.
NRcs	8bit – Number of records* Number of records requested – also denoted 'm'.
MaxL	8bit – Max APL Length* The maximum APL length in the response.
Time	64 bit – Timestamp* The timestamp used as search criteria in time based readout commands.
RecIDOff	16bit – Record ID Offset A record ID offset in the log, in relation to the record ID found in the search, defining the readout starting point in the log for the response – also denoted 'k'. If for instance a request for 20 records from 12 o'clock a.m. is send, but only 10 records is contained in the answer because of APL overflow (see MaxL). Then another request for 10 records at the same timestamp, but with 'Record ID Offset' set to 10, would return the missing 10 records.

* See abbreviation list at paragraph 6.3

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 29

Example (see log record examples at paragraph 6.3):

Two registers (E1 and V1) from 2007-05-24 19:45:00 and 3 records backwards with an offset of 6 records.



CID	LID	NRgs	Reg ID 1		Reg ID 2	
0	1	2	3	4	5	6
A3	01	02	003C		0044	

NRcs	MaxL	Time										RecIDOff	
7	8	9	10	11	12	13	14	15	16	16	16		
03	80	xxxx002D13180507										0006	

Response: [A0h] [Log type ID] [Number of returned registers]
 (RecordID *M*) (Reg ID 1) (Reg ID 1 format: [Unit] [Number of bytes] [SignExp]) (Reg ID 1 value)
 (Reg ID 2) (Reg ID 2 format: [Unit] [Number of bytes] [SignExp]) (Reg ID 2 value)
 ...
 (Reg ID *n*) (Reg ID *n* format: [Unit] [Number of bytes] [SignExp]) (Reg ID *n* value)
 (RecordID *M-1*) (Reg ID 1 value) (Reg ID 2 value) ... (Reg ID *n* value)
 ...
 (RecordID 1) (Reg ID 1 value) (Reg ID 2 value) ... (Reg ID *n* value)
 (LastReadOutRecord ID) (NewestRecord ID) [Info]

Parameter	Description
NRtR	8bit – Number of returned registers* Number of registers included in the response – also denoted ‘N’.
RecID <i>M-1</i>	16bit – Record ID <i>M-1</i> * Record ID for each of the records in the response.
Reg ID 1- <i>N</i> format	24bit – Reg ID 1- <i>N</i> format* The register format of each register included in the response. See section 4.2, page 7, for details on the floating-point data format.
Reg ID 1- <i>N</i> value	xxbit ¹⁾ – Reg ID <i>n</i> value* The register value of each register included in the response for each log record.
LRORecID	16bit – LastReadOutRecord ID* The ID of the record included in the response which was readout last/most recent from the log.
NewRec ID	16bit – NewestRecord ID* The ID of the newest record contained in the log.
INFO	8bit – Info* An info byte indicating errors concerning log readout.

* See abbreviation list at paragraph 6.3

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.:LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 30

Example:

CID	LTID	NRtR	RecID 1		Reg ID 1		Reg ID 1 format			Reg ID 1 value			
0	1	2	3	4	5	6	7	8	9	10	11	12	13
A0	01	02	10F5		003C		080400			CF065396			

Reg ID 2		Reg ID 2 format			Reg ID 2 value			
14	15	16	17	18	19	20	21	22
0044		270400			F2584C96			

RecID 2		Reg ID 1 value						Reg ID 2 value			
23	24	25	26	27	28	29	30	31	32		
10F4		B50A5C5E						93E05047			

RecID 3		Reg ID 1 value						Reg ID 2 value			
33	34	35	36	37	38	39	40	41	42		
10F3		FF068662						E993DC05			

LRORecID		NewRec ID		INFO
43	44	45	46	47
10F3		1102		00

⇓

		Record ID		
		10F5	10F4	10F3
Register ID	003C	CF065396	B50A5C5E	FF068662
	0044	F2584C96	93E05047	E993DC05

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 31

6.3.5 9Bh GetEventStatus

Function: Returns the four event status bytes.

Request: [9Bh]

Example:

CID
0
9B

Response: [9Bh][EventStatusByte0] [EventStatusByte1] [EventStatusByte2] [EventStatusByte3]

Parameter	Description								
ES0	8bit – EventStatusByte0 Event status byte 0. <i>Not used.</i>								
ES1	8bit – EventStatusByte1 Event status byte 1. <i>Not used.</i>								
ES2	8bit – EventStatusByte2 Event status byte 2. Each bit in this byte is used as flags to indicate if any of the logs in a device contains new unread log records. 0 = No unread data in log ; 1 = New unread data in log <table border="1"> <thead> <tr> <th>Bit</th><th>Log</th></tr> </thead> <tbody> <tr> <td>0</td><td>Interval log contain new unread log record(s)</td></tr> <tr> <td>1</td><td>RTC log contain new unread log record(s)</td></tr> <tr> <td>2-7</td><td><i>Not used</i></td></tr> </tbody> </table>	Bit	Log	0	Interval log contain new unread log record(s)	1	RTC log contain new unread log record(s)	2-7	<i>Not used</i>
Bit	Log								
0	Interval log contain new unread log record(s)								
1	RTC log contain new unread log record(s)								
2-7	<i>Not used</i>								
ES3	8bit – EventStatusByte3 Event status byte 3. <i>Not used.</i>								

* See abbreviation list at paragraph 6.3

Example:

CID	ES0	ES1	ES2	ES3
0	2	3	4	5
9B	xx	xx	xx	xx

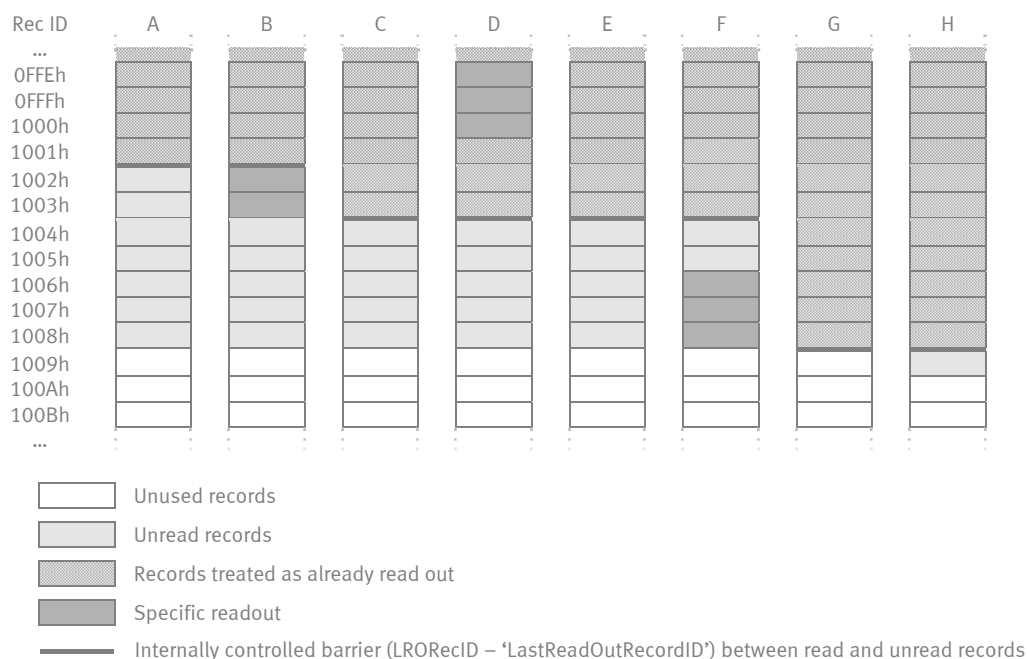
Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.:LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 32

6.3.6 9Ch ClearEventStatus

Function: Clear the event status bytes.

The function is prepared for future products, and for coexistence with the electrical meters. Only “conditional clear” (CC) is supported, in the MC601 logger modules and in MC801.

An example of a log and some read out scenarios are described here:



- The log contains data up to record ID 1008h. The last record ID read (followed by a ‘ClearEventStatus’) previous to this state is record ID 1001h. The ‘EventStatusByte’ flag will be 1 for the shown log because of the unread records in the log.
- Record 1002h and 1003h are read out. The last record ID read is still record ID 1001h, while LRORecID is only updated when ‘ClearEventStatus’ is called at the log in question.
- ‘ClearEventStatus’ is called. The last record ID read is changed to record ID 1003h. The ‘EventStatusByte’ flag is still 1 because of the unread records in the log.
- Record 00FEh to 1000h are read out. A read out will not affect the last record ID read when the record ID(s) read out is older than the newest record ever read.
- ‘ClearEventStatus’ is called. The last record ID read is unchanged (see D).
- Record 1006h to the newest record in the log (record ID 1008h) are read out. As described at B and D the read out action itself doesn’t change the LRORecID.
- ‘ClearEventStatus’ is called. The last record ID read is now changed to record ID 1008h. The ‘EventStatusByte’ flag will be 0 now because the newest read record equals the newest log in the log. All data from record ID 1008h and back in time is perceived as read, even though log record 1004h and 1005h hasn’t been read in this example.
- A new record is created in the log. This changes the ‘EventStatusByte’ flag to 1.

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 33

Request: [9Ch] [ClearEventStatusByte0-CC] [ClearEventStatusByte1-CC] [ClearEventStatusByte2-CC]
 [ClearEventStatusByte3-CC] [ClearEventStatusByte0-UC] [ClearEventStatusByte1-UC]
 [ClearEventStatusByte2-UC] [ClearEventStatusByte3-UC]

Parameter	Description								
CS0c	8bit – ClearEventStatusByte0-CC <i>Not used – default 0.</i>								
CS1c	8bit – ClearEventStatusByte1-CC <i>Not used – default 0.</i>								
CS2c	8bit – ClearEventStatusByte2-CC Clear mask for event status byte 2. This byte is used as a mask for clearing the event status byte 2. Each bit in ClearEventStatusByte2 control the same bit in EventStatusByte2. The flag is only cleared if the newest record in the corresponding log is read out. 0 = Event flag is not cleared 1 = Event flag is cleared if the condition(s) is true <table border="1"> <thead> <tr> <th>Bit</th><th>Log</th></tr> </thead> <tbody> <tr> <td>0</td><td>Conditional clear of Interval log flag</td></tr> <tr> <td>1</td><td>Conditional clear of RTC log flag</td></tr> <tr> <td>2-7</td><td>Not used</td></tr> </tbody> </table>	Bit	Log	0	Conditional clear of Interval log flag	1	Conditional clear of RTC log flag	2-7	Not used
Bit	Log								
0	Conditional clear of Interval log flag								
1	Conditional clear of RTC log flag								
2-7	Not used								
CS3c	8bit – ClearEventStatusByte3-CC <i>Not used – default 0.</i>								
CS0u	8bit – ClearEventStatusByte0-UC <i>Not used – default 0.</i>								
CS1u	8bit – ClearEventStatusByte1-UC <i>Not used – default 0.</i>								
CS2u	8bit – ClearEventStatusByte2-UC <i>Not used – default 0.</i>								
CS3u	8bit – ClearEventStatusByte3-UC <i>Not used – default 0.</i>								

* See abbreviation list at paragraph 6.3

Example:

CID	CS0c	CS1c	CS2c	CS3c	CS0u	CS1u	CS2u	CS3u
0	2	3	4	5	6	7	8	9
9C	00	00	01	00	00	00	00	00

Response: [9Bh][EventStatusByte0] [EventStatusByte1] [EventStatusByte2] [EventStatusByte3]

Example:

CID	ES0	ES1	ES2	ES3
0	2	3	4	5
9B	xx	xx	xx	xx

As CID 9B GetEventStatus.

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.: LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 34

6.3.7 PutRegister - CID = 11h

Function: This command is used for changing the value of a given register.

Request: [80h] [destination address][**11h**] (**Password**) (**register ID**)(**Unit**)(**Format**)(**Value**)(CRC) [0Dh]

(Password) is a 16 bit value. If the password is invalid, the register value is not changed.

(register ID) is a 16 bit register identification codes and specifies registers to be read out.

[register unit] is a unit identification code and specifies units.

(register format) is the floating-point data format consisting of:

[number of bytes] [sign+exponent]

(Value) is the new value to use.

Response: [06h]

ACK (0x06) if the value was written, otherwise no reply is given.

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.:LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 35

Index

F

Formatting

Floating point.....	7
Registers	9

K

KMP Commands

By CID

0x01.....	15
0x02.....	17
0x09.....	18
0x10.....	19
0x11.....	35
0x9B.....	32
0x9C.....	33
0xA0.....	24
0xA1.....	27
0xA2.....	28
0xA3.....	29

By Name

ClearEventStatus	33
GetEventStatus.....	32
GetLogIDPresent	28
GetLogLastPresent.....	27
GetLogTimePast	29
GetLogTimePresent	24
GetRegister	19
GetSerialNo.....	17
GetType.....	15
PutRegister.....	35
SetClock.....	18

P

Protocol.....	5
---------------	---

T

Timing

Communication	3
---------------------	---

Last page: 36		Date for last revision: 2009-12-07			Rev.: M1
Opr.: GHA	Aut.:LTS/AJ/MAJ	QA: SL	EDB: GHA	Doc. No.: 5512-447	Page: 36