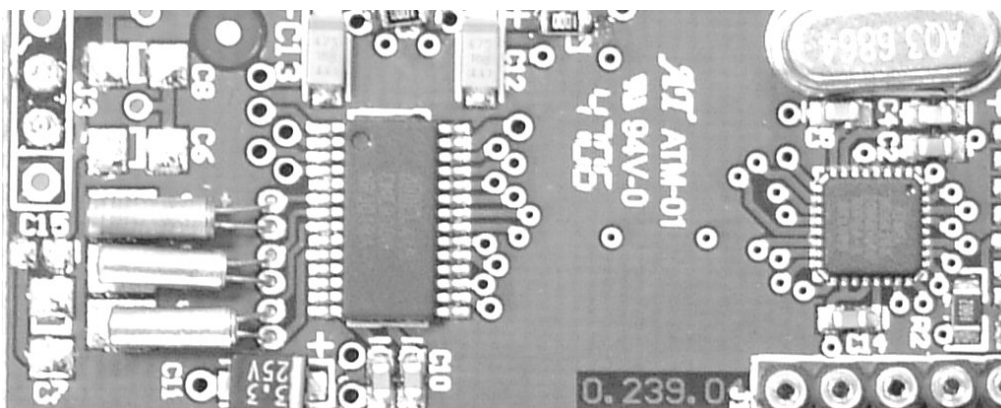


CMMC-8P-MF

Industrial module with CME8000 receiver IC

Table of contents:

1. Short description	2
2. Overview	3
3. Accessories	3
4. Ordering information	4
5. Programming guide	5



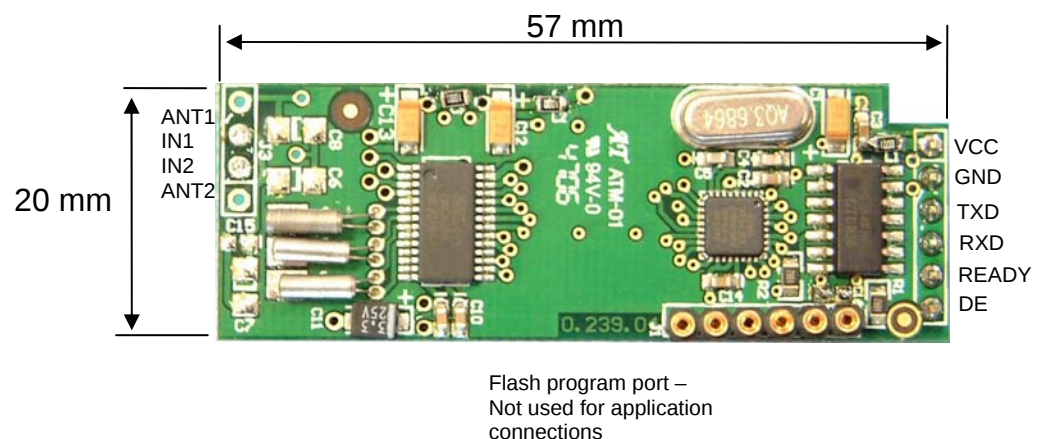
1. Short Description

The CME8000 is a BiCMOS integrated straight through receiver with build in very high sensitivity and a pre-decoding of the time signal transmitted from WWVB, DCF77, JJY40, JJY60, MSF and HBG. The receiver is prepared for multi-frequency and country reception by using an integrated logic. The CME8000 is connected to an in-built micro-controller in the CMMC-8P-MF module which is programmed with a pre-defined serial interface protocol.

Features

- o Automatic reception of long wave time signals world wide
- o Manual or automatic selection of radio control signal possible
- o Forced reception mode
- o Real time clock (local time and UTC time)
- o Time zone support
- o Real signal quality indicator during reception
- o 24-hour system
- o Adjustable reception settings (including time and duration of reception)
- o Low power consumption (< 2mA during reception active mode)
- o Open architecture for data exchange
- o Build in decoding for different signals
- o Automatic switch between dual band signals.
- o Wide operating range: 3 V to 5 V

Module Layout CMMC-8P-MF



Spec No.:	Revision	State	C-MAX printed	Version	Page
CMMC-8P-MF	179.05	07.01.2008	07.02.2008	English	2 of 22

2. Overview

2.1 Interface to application

The CMMC-8P-MF uses two wires (RXD, TXD) to communicate with your application.

2.2 Time piece functions

The module will respond to commands from the host and return required time information.

- Time information available includes hours, minutes, second, day of week, month, year (in last two digits), special information (e.g. DST status)
- The CMMC-8P-MF provides local time (considering time zone) + UTC time
- Manual time can be set, e.g. if no reception is possible

2.3 Time signal reception functions

- Selection of WWVB, DCF, JJY or MSF signal
- Adjustable reception options:
 - o max. duration of reception
 - o number of receptions
 - o Validation of receptions
 - o Decoding options, e.g. time only, time + date only
- Three selectable reception modes:
 - o manual
 - o continuous
 - o pre-set reception sequence

(The duration of reception, the number of auto receptions, interval between auto reception trials (if more than one is defined), start time of 1st auto reception of the day can all be defined by the application)

2.4 Available reception signal settings

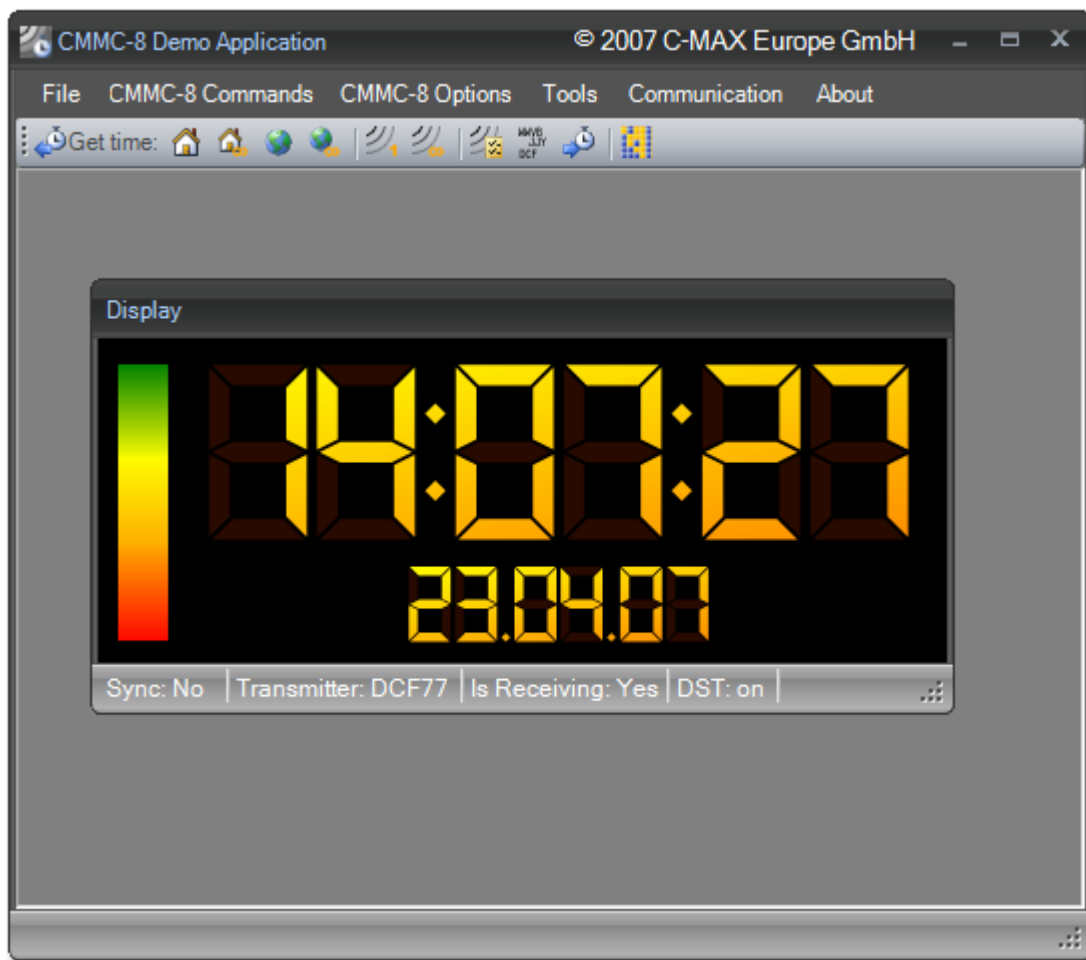
Radio control reception can be set by software command from an external host. Various reception modes can be defined:

<u>Single band :</u>	<u>Dual band:</u>	<u>Multi band</u>
DCF	JJY40 / JJY60	Scan all signals
JJY40	MSF / DCF	
JJY60		
MSF		
WWVB		

3. Accessories

CMMC-8P-MF Demo Application

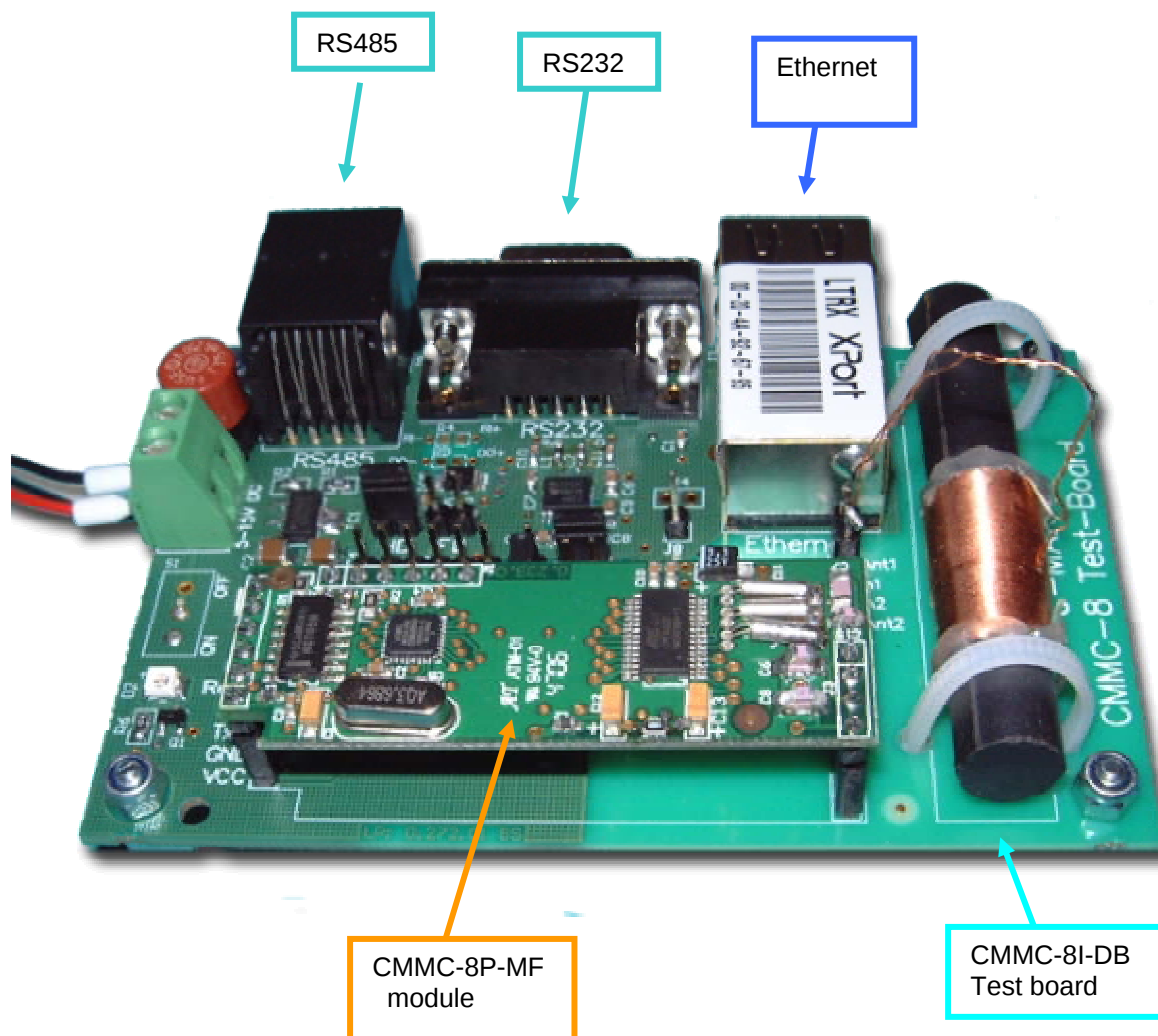
For first steps with our CMMC-8P-MF module and to demonstrate features and functions, a demo application software (including a startup wizard) is available for download from our homepage.



Requires an interface (CMMC-8I-DB).

CMMC-8P-MF Interface

For connection of the CMMC-8P-MF module to a PC or industrial application, adapters are available from C-MAX or can be built as shown in the circuit (download from C-MAX webpage).

**4. Ordering information**

Ordering no.	Description	Dimension
CMMC-8P-MF 	CMMC-8P-MF module – PCB antenna included	20 x 57mm (without antenna)
CMMC-8I-DB RS232/RS485  Ethernet 	Test board with RS232/485 and Ethernet interface for CMMC-8P-MF module	100 x 65 mm

General command structure

CMMC-8P-MF Layer (Request):

byte #0	Command
byte #1	Parameter length (no. of bytes)
byte #2	1st parameter byte
byte #3	2nd parameter byte
...	...
byte #n	last Parameter byte

CMMC-8P-MF Layer (Response):

byte #0	Command
byte #1	Result length (no. of bytes)
byte #2	1st result byte
byte #3	2nd result byte
...	...
byte #n	last result byte

RS-232 Layer:

byte #0	0x02 (STX)
byte #1	<i>bytes from the CMMC-8P-MF Layer (see above)</i>
...	
byte #n	
byte #n+1	CRC 16 (LSB)
byte #n+2	CRC 16 (MSB)
byte #n+3	0x03 (ETX)

RS-485 Layer:

byte #0	0x02 (STX)
byte #1	Receiver's Address
byte #2	Sender's Address
byte #3	<i>bytes from the CMMC-8P-MF Layer (see above)</i>
...	
byte #n	
byte #n+1	CRC 16 (LSB)
byte #n+2	CRC 16 (MSB)
byte #n+3	0x03 (ETX)

Please use the following serial settings when communicating to the module:

- **9600 baud**
- **No parity bit**
- **Eight data bits**
- **One stop bit**

Examples

Requesting the local time via the RS232 protocol:

byte #	value (dec)	value (hex)
0	2	0x02
1	2	0x02
2	0	0x00
3	98	0x62
4	102	0x66
5	3	0x03

The CMMC-8P-MF will respond:

byte #	value (dec)	value (hex)
0	2	0x02
1	2	0x02
2	11	0x0B
3	52	0x34
4	45	0x2D
5	11	0x0B
6	8	0x08
7	11	0x0B
8	6	0x06
9	3	0x03
10	0	0x00
11	0	0x00
12	1	0x01
13	1	0x01
14	170	0xAA
15	154	0x9A
16	3	0x11

Result: It is 11:47:49 AM on November 8th 06 (a Wednesday). The clock of the CMMC-8P-MF is not in sync with a time signal transmitter and the DST is not active. The time zone is set to UTC+1 and while calculating local time, the DST information has to be considered.

CRC-Calculation

The CRC used in the CMMC-8P-MF is based on a CRC16 (see <http://en.wikipedia.org/wiki/CRC16> for details). It is calculated for all bytes of the CMMC-8P-MF Layer (see *General command structure*). If you are using the RS485 protocol, both addresses are also included in the CRC calculation. For testing and evaluating reasons, the CMMC-8P-MF will also accept two 0x00 bytes as CRC. For production environments however, we strongly recommend using valid CRC bytes.

Implementation

A standard implementation in C:

```
unsigned int CalcCrc16(unsigned char *Buffer,
unsigned char Length)
{
    volatile unsigned char i,j;
    volatile unsigned int Crc16 = 0;
    volatile int c;
    char *pBuf;

    pBuf = (char*)(Buffer);
    for( i=0; i<Length; i++)
    {

        c = *pBuf++;
        c = c << 8;

        for( j=0; j<8; j++ )
        {
            if( (Crc16^c) & 0x8000 )
                Crc16 = ( Crc16<<1 ) ^ 0x1021;
            else
                Crc16 = Crc16 << 1;

            c = c << 1;
        }
    }

    return (Crc16);
}
```

Implementation of CRC Calculation using a lookup table in C# (.NET Framework) and C:

```

public class CRCCalc
{
    static ushort[] Crc16Table = new ushort[] {
        0x0000, 0x1021, 0x2042, 0x3063, 0x4084, 0x50A5,
0x60C6, 0x70E7,
        0x8108, 0x9129, 0xA14A, 0xB16B, 0xC18C, 0xD1AD,
0xE1CE, 0xF1EF,
        0x1231, 0x0210, 0x3273, 0x2252, 0x52B5, 0x4294,
0x72F7, 0x62D6,
        0x9339, 0x8318, 0xB37B, 0xA35A, 0xD3BD, 0xC39C,
0xF3FF, 0xE3DE,
        0x2462, 0x3443, 0x0420, 0x1401, 0x64E6, 0x74C7,
0x44A4, 0x5485,
        0xA56A, 0xB54B, 0x8528, 0x9509, 0xE5EE, 0xF5CF,
0xC5AC, 0xD58D,
        0x3653, 0x2672, 0x1611, 0x0630, 0x76D7, 0x66F6,
0x5695, 0x46B4,
        0xB75B, 0xA77A, 0x9719, 0x8738, 0xF7DF, 0xE7FE,
0xD79D, 0xC7BC,
        0x48C4, 0x58E5, 0x6886, 0x78A7, 0x0840, 0x1861,
0x2802, 0x3823,
        0xC9CC, 0xD9ED, 0xE98E, 0xF9AF, 0x8948, 0x9969,
0xA90A, 0xB92B,
        0x5AF5, 0x4AD4, 0x7AB7, 0x6A96, 0x1A71, 0x0A50,
0x3A33, 0x2A12,
        0xDBFD, 0xCBDC, 0xFBBF, 0xEB9E, 0x9B79, 0x8B58,
0xBB3B, 0xAB1A,
        0x6CA6, 0x7C87, 0x4CE4, 0x5CC5, 0x2C22, 0x3C03,
0x0C60, 0x1C41,
        0xEDAE, 0xFD8F, 0xCDEC, 0xDDCD, 0xAD2A, 0xBD0B,
0x8D68, 0x9D49,
        0x7E97, 0x6EB6, 0x5ED5, 0x4EF4, 0x3E13, 0x2E32,
0x1E51, 0x0E70,
        0xFF9F, 0xEFBE, 0xDFDD, 0xCFFC, 0xBF1B, 0xAF3A,
0x9F59, 0x8F78,
        0x9188, 0x81A9, 0xB1CA, 0xA1EB, 0xD10C, 0xC12D,
0xF14E, 0xE16F,
        0x1080, 0x00A1, 0x30C2, 0x20E3, 0x5004, 0x4025,
0x7046, 0x6067,
        0x83B9, 0x9398, 0xA3FB, 0xB3DA, 0xC33D, 0xD31C,
0xE37F, 0xF35E,
        0x02B1, 0x1290, 0x22F3, 0x32D2, 0x4235, 0x5214,
0x6277, 0x7256,
        0xB5EA, 0xA5CB, 0x95A8, 0x8589, 0xF56E, 0xE54F,
0xD52C, 0xC50D,
        0x34E2, 0x24C3, 0x14A0, 0x0481, 0x7466, 0x6447,
0x5424, 0x4405,
        0xA7DB, 0xB7FA, 0x8799, 0x97B8, 0xE75F, 0xF77E,
0xC71D, 0xD73C,
        0x26D3, 0x36F2, 0x0691, 0x16B0, 0x6657, 0x7676,
0x4615, 0x5634,
        0xD94C, 0xC96D, 0xF90E, 0xE92F, 0x99C8, 0x89E9,
0xB98A, 0xA9AB,
    }
}

```

```

        0x5844, 0x4865, 0x7806, 0x6827, 0x18C0, 0x08E1,
0x3882, 0x28A3,
        0xCB7D, 0xDB5C, 0xEB3F, 0xFB1E, 0x8BF9, 0x9BD8,
0xABBB, 0xBB9A,
        0x4A75, 0x5A54, 0x6A37, 0x7A16, 0x0AF1, 0x1AD0,
0x2AB3, 0x3A92,
        0xFD2E, 0xED0F, 0xDD6C, 0xCD4D, 0xBDAA, 0xAD8B,
0x9DE8, 0x8DC9,
        0x7C26, 0x6C07, 0x5C64, 0x4C45, 0x3CA2, 0x2C83,
0x1CE0, 0x0CC1,
        0xEF1F, 0xFF3E, 0xCF5D, 0xDF7C, 0xAF9B, 0xBFBA,
0x8FD9, 0x9FF8,
        0x6E17, 0x7E36, 0x4E55, 0x5E74, 0x2E93, 0x3EB2,
0x0ED1, 0x1EF0,
    };

```

```

public static ushort CalcCRC(byte[] buffer)
{
    ushort crc = 0;
    ushort bufferPointer = 0;
    for (int i = 0; i < buffer.Length; i++)
    {
        ushort crcRightShift = (ushort)(crc >> 8);
        ushort crcLeftShift = (ushort)(crc << 8);
        ushort byteFromBuffer =
buffer[bufferPointer++];

        crc =
        (ushort)(Crc16Table[(ushort)((ushort)(crcRightShift
        ^ byteFromBuffer)) & 0xFF]) ^
        crcLeftShift);
    }
    return crc;
}
}

```

```

const unsigned int code Crc16Table[256] =
{
    0X0000, 0X1021, 0X2042, 0X3063, 0X4084, 0X50A5,
0X60C6, 0X70E7,
    0X8108, 0X9129, 0XA14A, 0XB16B, 0XC18C, 0XD1AD,
0XE1CE, 0XF1EF,
    0X1231, 0X0210, 0X3273, 0X2252, 0X52B5, 0X4294,
0X72F7, 0X62D6,
    0X9339, 0X8318, 0XB37B, 0XA35A, 0XD3BD, 0XC39C,
0XF3FF, 0XE3DE,
    0X2462, 0X3443, 0X0420, 0X1401, 0X64E6, 0X74C7,
0X44A4, 0X5485,
    0XA56A, 0XB54B, 0X8528, 0X9509, 0XE5EE, 0XF5CF,
0XC5AC, 0XD58D,

```

```

    0X3653, 0X2672, 0X1611, 0X0630, 0X76D7, 0X66F6,
    0X5695, 0X46B4,
    0XB75B, 0XA77A, 0X9719, 0X8738, 0XF7DF, 0XE7FE,
    0XD79D, 0XC7BC,
    0X48C4, 0X58E5, 0X6886, 0X78A7, 0X0840, 0X1861,
    0X2802, 0X3823,
    0XC9CC, 0XD9ED, 0XE98E, 0XF9AF, 0X8948, 0X9969,
    0XA90A, 0XB92B,
    0X5AF5, 0X4AD4, 0X7AB7, 0X6A96, 0X1A71, 0X0A50,
    0X3A33, 0X2A12,
    0XDBFD, 0XCDBC, 0XFBBF, 0XEB9E, 0X9B79, 0X8B58,
    0XBB3B, 0XAB1A,
    0X6CA6, 0X7C87, 0X4CE4, 0X5CC5, 0X2C22, 0X3C03,
    0X0C60, 0X1C41,
    0XEDAE, 0XFD8F, 0XCDEC, 0XDDCD, 0XAD2A, 0XBD0B,
    0X8D68, 0X9D49,
    0X7E97, 0X6EB6, 0X5ED5, 0X4EF4, 0X3E13, 0X2E32,
    0X1E51, 0X0E70,
    0XFF9F, 0XEFBE, 0XDFDD, 0XCFFC, 0XBF1B, 0XAF3A,
    0X9F59, 0X8F78,
    0X9188, 0X81A9, 0XB1CA, 0XA1EB, 0XD10C, 0XC12D,
    0XF14E, 0XE16F,
    0X1080, 0X00A1, 0X30C2, 0X20E3, 0X5004, 0X4025,
    0X7046, 0X6067,
    0X83B9, 0X9398, 0XA3FB, 0XB3DA, 0XC33D, 0XD31C,
    0XE37F, 0XF35E,
    0X02B1, 0X1290, 0X22F3, 0X32D2, 0X4235, 0X5214,
    0X6277, 0X7256,
    0XB5EA, 0XA5CB, 0X95A8, 0X8589, 0XF56E, 0XE54F,
    0XD52C, 0XC50D,
    0X34E2, 0X24C3, 0X14A0, 0X0481, 0X7466, 0X6447,
    0X5424, 0X4405,
    0XA7DB, 0XB7FA, 0X8799, 0X97B8, 0XE75F, 0XF77E,
    0XC71D, 0XD73C,
    0X26D3, 0X36F2, 0X0691, 0X16B0, 0X6657, 0X7676,
    0X4615, 0X5634,
    0XD94C, 0XC96D, 0XF90E, 0XE92F, 0X99C8, 0X89E9,
    0XB98A, 0XA9AB,
    0X5844, 0X4865, 0X7806, 0X6827, 0X18C0, 0X08E1,
    0X3882, 0X28A3,
    0XCB7D, 0XDB5C, 0XEB3F, 0XFB1E, 0X8BF9, 0X9BD8,
    0XABBB, 0XBB9A,
    0X4A75, 0X5A54, 0X6A37, 0X7A16, 0X0AF1, 0X1AD0,
    0X2AB3, 0X3A92,
    0XFD2E, 0XED0F, 0XDD6C, 0XCD4D, 0XBDAA, 0XAD8B,
    0X9DE8, 0X8DC9,
    0X7C26, 0X6C07, 0X5C64, 0X4C45, 0X3CA2, 0X2C83,
    0X1CE0, 0X0CC1,
    0XEF1F, 0XFF3E, 0XCF5D, 0XDF7C, 0XAF9B, 0XBFBA,
    0X8FD9, 0X9FF8,
    0X6E17, 0X7E36, 0X4E55, 0X5E74, 0X2E93, 0X3EB2,
    0X0ED1, 0X1EF0,
};

```

```

unsigned int CalcCrc16(char *Buffer, unsigned char Length)

```

```
{  
    unsigned char i;  
    unsigned int Crc16 = 0;  
  
    for ( i=0; i<Length; i++)  
    {  
        Crc16 =  
Crc16Table[ ((Crc16>>8) ^ *Buffer++) & 0xFF] ^ (Crc16<<8);  
    }  
  
    return (Crc16);  
}
```

CMMC-8P-MF – List of commands

Command	GetUTCTime (0x01)	
Parameter(s)	-	
Result	<i>byte</i> second	00-59
	<i>byte</i> minute	00-59
	<i>byte</i> hour	00-23
	<i>byte</i> day	01-31
	<i>byte</i> month	01-12
	<i>byte</i> year	00-99
	<i>byte</i> day of week	1 ... Monday 2 ... Tuesday [...] 6 ... Saturday 7 ... Sunday
	<i>byte</i> Sync State	0 ... not synchronized 1 ... synchronized
Description	Reads the UTC time from the CMMC-8P-MF.	

Command	GetLocalTime (0x02)	
Parameter(s)	-	
Result	<i>byte</i> second	00-59
	<i>byte</i> minute	00-59
	<i>byte</i> hour	00-23
	<i>byte</i> day	01-31
	<i>byte</i> month	01-12
	<i>byte</i> year	00-99
	<i>byte</i> day of week	1 ... Monday 2 ... Tuesday [...] 6 ... Saturday 7 ... Sunday
	<i>byte</i> sync state	0 ... not synchronized 1 ... synchronized
	<i>byte</i> DST-Flag	0 ... Standard time 1 ... DST
	<i>byte</i> timeZone	See SetLocalTimeOptions
	<i>byte</i> DSTSupport	See SetLocalTimeOptions
Description	Reads the UTC time from the CMMC-8P-MF.	

Command	SetUTCTime (0x03)	
Parameter(s)	<i>byte</i> second	00-59
	<i>byte</i> minute	00-59
	<i>byte</i> hour	00-23
	<i>byte</i> day	01-31
	<i>byte</i> month	01-12
	<i>byte</i> year	00-99
Result	<i>byte</i> commandResult	see below
Description	Sets the internal clock of the CMMC-8P-MF. Side Effects: After the time is set manually, the GetLastSuccessfulSync command will always return 000000 and the sync state of the GetUTCTime / GetLocalTime commands will always be 0.	

Command	SetLocalTime (0x04)	
Parameter(s)	<i>byte</i> second	00-59
	<i>byte</i> minute	00-59
	<i>byte</i> hour	00-23
	<i>byte</i> day	01-31
	<i>byte</i> month	01-12
	<i>byte</i> year	00-99
Result	<i>byte</i> commandResult	see below
Description	Sets the internal clock of the CMMC-8P-MF. If the time/date information was set successfully, the CMMC-8P-MF returns 1. If the CMMC-8P-MF is not able to set the time-information (e.g. hour > 23), it returns 0. Side Effects: After the time is set manually, the GetLastSuccessfulSync command will always return 000000 and the sync state of the GetUTCTime / GetLocalTime commands will always be 0.	

Command **SetLocalTimeOptions (0x05)**

Parameter(s)	<i>byte</i> timeZone	0 ... +- 0
		1 ... +1
		[...]
		14 ... +14
		255 ... -1
		[...]
		244 ... -12
	<i>byte</i> DSTSupport	0 ... false
		1 ... true

Result	<i>byte</i> commandResult	see below
---------------	---------------------------	-----------

Description Sets the options to enable the CMMC-8P-MF to calculate the local time from the UTC time.

- **timeZone:** defines the deviation of your local timezone from the UTC time.
- **hasDST:** if false, the one hour DST offset is not taken into account when calculating the local time, even if the transmitter sends the DST flag.

Command **SetDST (0x06)**

Parameter(s)	<i>byte</i> dst	0 ... DST off
		1 ... DST on

Result	<i>byte</i> commandResult	see below
---------------	---------------------------	-----------

Description Enables / disables the one hour DST offset. At the next successful reception, the manual set DST flag will be overwritten by the transmitted value.

Command	SetTransmitter (0x07)	
Parameter(s)	<i>byte</i> transmitterCode	1 ... DCF77 2 ... WWVB 3 ... MSF 4 ... JJY60 5 ... JJY40 6 ... DCF77 / MSF 7 ... JJY 40/60 8 ... Global Scan
Result	<i>byte</i> commandResult	see below
Description	Sets the transmitter / mode of the CMMC-8P-MF Side Effects: - The timezone information is set according to the location of the chosen transmitter. If you are using one of the scan modes, you should set the time zone manually by using SetLocalTimeOptions to avoid wrong local time calculations. - A manual reception is started	

Command	GetReceiverStatus (0x08)	
Parameter(s)	-	
Result	<i>byte</i> transmitterCode <i>byte</i> isReceiving <i>byte</i> bsi	See SetTransmitter 1 ... true 0 ... false 0-3
Description	- transmitterCode: code of the transmitter, the CMMC-8P-MF is using (has used the last time) - isReceiving: indicates, whether the CMMC-8P-MF is receiving or not - bsi: value of the BS-Indicator of the CMMC-8P-MF. Always 0 if CME8000 is not in receiving mode.	

Command	SetReceptionOptions (0x09)	
Parameter(s)	<i>byte</i> maxDuration	
	<i>byte</i> numberOfValidReceptions	
	<i>byte</i> consecutiveReceptions	0 ... false 1 ... true
	<i>byte</i> autoRecEnabled	0 ... false 1 ... true
	<i>byte</i> autoRecHour	0-23
	<i>byte</i> autoRecMinute	0-59
	<i>byte</i> autoRecDelta	0-255
	<i>byte</i> autoRecCount	0-10
	<i>byte</i> checkParity	0 ... true 1 ... false
	<i>byte</i> protocolMask	0 ... FullDecode 2 ... TimeOnly 1 ... TimeDateOnly
Result	<i>byte</i> commandResult	see below

Description Sets the reception options of the CMMC-8P-MF.

- maxDuration: duration of a reception try
- numberOfValidReceptions: defines how many decodings must fit logical together, to allow an update of the internal clock.
- consecutiveReceptions: if true, the decodings defined by numberOfValidReceptions must be received in a consecutive order.
- autoRecEnabled: defines whether the automatic reception is enabled or not.
- autoRecHour / autoRecMinute: defines the starting point of an automatic reception
- autoRecDelta: timespan (in minutes) between two autoreception tries, if the prior reception has failed.
- autoRecCount: number of autoreception tries. If the last try was also unsuccessful, the next try will start at the next the starting point defined by autoRecHour / autoRecMinute
- checkParity: if false, the parity-bits sent by the transmitters are not taken into account.
- protocolMask: defines which part of the transmission is to be decoded.

Command **StartReception (0x0A)**

Parameter(s) -

Result *byte* commandResult see below

Description Starts a manual reception.

Command **StartContinuousReception (0x0B)**

Parameter(s) -

Result *byte* commandResult see below

Description Starts a continuous reception. Immediately after an invalid or valid reception, the next reception is started again.

Command **GetLastSuccessfulSync (0x0C)**

Parameter(s) -

Result *byte* second 00-59
 byte minute 00-59
 byte hour 00-23
 byte day 01-31
 byte month 01-12
 byte year 00-99

Description Returns the time (UTC) and date of the last successful reception. This information will be lost, if the module loses power. In this case it will return six bytes valued 0.

Command **SetSyncTimespan (0x0D)**

Parameter(s) *byte* hour 0-255

Result *byte* commandResult see below

Description Defines the timespan between the last successful reception and the moment after that the sync state of the GetLocalTime / GetUTCtime result will switch back to 0

Command	SetSecondPolling (0x0E)	
Parameter(s)	<i>byte</i> polling	0 ... off 1 ... on
Result	<i>byte</i> commandResult	see below
Description	Enables / disables the second polling. If enabled, the CMMC-8P-MF will send a single byte exactly at the change between two seconds. The sent bytes represents the new second. If the CMMC-8P-MF is in polling mode, no other commands than SetSecondPolling will be accepted.	

Command	GetRAWData (0x0F)	
Parameter(s)	-	
Result	<i>10 x byte[8]</i> rawData	
Description	This commands returns 10 packages of an eight-byte-array (64 bits). The Protocol Identifier Bits added by the CME8000 are stored in the first 4 bits. The remaining 60 bits represent the signal received by the CME8000. Altogether the raw data of the 10 last receptions are transmitted. The latest received raw data is sent first. All raw data information will be erased, if - the CMMC-8P-MF is disconnected from the power supply - a new reception is started	

Command	GetCMMC8Version (0x10)	
Parameter(s)	-	
Result	<i>char[]</i> version	
Description	Returns the firmware version of the CMMC-8P-MF.	

Command	UseReferenceTime (0x11)		
Parameter(s)	<i>byte</i> useReferenceTime	0 ... false	1 ... true
Result	<i>byte</i> commandResult	see below	
Description	If the reference time is enabled, a reception will end, if a time-code is received, that fits to the internal time of the CMMC-8P-MF.		

Command	SetFastCode (0x12)		
Parameter(s)	<i>byte</i> FastCode-Factor		
Result	<i>byte</i> commandResult	see below	
Description	Sets the FastCode option of the CMMC-8P-MF. Setting this option will reset the internal CME8000 and aborts a running reception.		
	Note: If FastCode is enabled, you will not be able to receive a signal sent by standard transmitters. FastCode only works in combination with C-MAX engineering equipment.		

Command	SetRS485Address (0x13)		
Parameter(s)	byte newAddress		
Result	byte commandResult	see below	
Description	Sets the RS485 address of the CMMC-8P-MF.		

Command Result:

Each command that sets an option of the CMMC-8P-MF returns a single data byte.
If the CMMC-8P-MF has accepted the command, it will return 255 (or 0xFF). In case of an error, the result byte gives you some information about the cause of the error:

RD_START 0x00
RD_BOARDADDRESS 0x01
RD_REQUESTOR_ADDRESS 0x02
RD_RX_CMD 0x03
RD_DATA_LENGTH 0x04
RD_DATA 0x05
RD_CRC 0x06
RD_ETX 0x07
CRC_CHECK 0x08
RD_END 0x09
RD_ERROR 0x0A

SET_SECOND_ERROR 0x20
SET_MINUTE_ERROR 0x21
SET_HOUR_ERROR 0x22
SET_DATE_ERROR 0x23
SET_MONTH_ERROR 0x24
SET_YEAR_ERROR 0x25

SET_LOCAL_TIME_OPTION_ERROR 0x30
SET_HAS_DST_OPTION_ERROR 0x31

SET_DST_ERROR 0x40

TRANSMITTER_CODE_ERROR 0x50

MAX_DURATION_ERROR 0x60
NMB_VALID_RECEPTIONS_ERROR 0x61
CONSECUTIVE_RECEPTIONS_ERROR 0x62
AUTO_REC_ON_OFF_ERROR 0x63
AUTO_REC_HOUR_ERROR 0x64
AUTO_REC_MINUTE_ERROR 0x65
AUTO_REC_DELTA_ERROR 0x66
AUTO_REC_COUNT_ERROR 0x67
CHECK_PARITY_ERROR 0x68
PROTOCOL_MASK_ERROR 0x69

START_RECEPTION_ERROR 0x70

SET_SECOND_POLLING_ERROR 0x80

REFERENCE_TIME_ERROR 0x90

NO_ERROR 0xFF

Disclaimer of Warranty

Information furnished is believed to be accurate and reliable. However C-MAX assumes no responsibility, neither for the consequences of use of such information nor for any infringement of patents or other rights of third parties, which may

result from its use. No license is granted by implication or otherwise under any patent or patent rights of C-Max. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. C-MAX products are not authorized for use as critical components in life support devices without express written approval of C-MAX.

Note

It is not given warranty that the declared circuits, devices, facilities, components, assembly groups or treatments included herein are free from legal claims of third parties.

The declared data are serving only to description of product. They are not guaranteed properties as defined by law. The examples are given without obligation and cannot given rise to any liability.

Reprinting this data sheet - or parts of it - is only allowed with a license of the publisher.

C-MAX reserves the right to make changes on this specification without notice at any time.

C-MAX Time Solutions GmbH

Carl-Zeiss-Str. 13
74078 Heilbronn

Tel.: +49-7066-900400

Fax: +49-7066-9004029

e-mail: contact@c-max-time.com

Data sheets can also be retrieved from our Internet homepage: www.c-max-time.com