

The information contained in this documentation is the property of MAZeT. Photocopying or otherwise reproducing any part of the catalog, whether electronically or mechanically, is prohibited, except where the express permission of MAZeT GmbH has been obtained. In general, all company and brand names, as well as the names of individual products, are protected by brand, patent or product law.	VERSION AMENDMENTS		
	NO.	VERSION	APPROVED
	1	V 1.9	2008-08-04

Software Description

MTCS-C2-DLL MTCS-ME1-DLL

Library Description (MTCSApi.dll) API programming interface DLL From Revision 2.43

Table of Contents

1 Introduction	3
2 MTCS – FunCtionS	4
2.1 MTCS – Functions which are applicable to all MTCS-ME1 versions	4
2.1.1 <i>MTCSInitSystem</i>	4
2.1.2 <i>MTCSCloseDevice</i>	4
2.1.3 <i>MTCSDllGetVersion</i>	5
2.1.4 <i>MTCSReadVersion</i>	5
2.1.5 <i>MTCSReadMemory</i>	5
2.1.6 <i>MTCSWriteMemory</i>	6
2.1.7 <i>MTCSSetShift</i>	6
2.1.8 <i>MTCSSetUpdateMode</i>	7
2.2 MTCS – Functions which are applicable to the TOP, FRONT, DARK TIA versions	7
2.2.1 <i>MTCSGetADCxx</i>	7
2.2.2 <i>MTCSGetADCBL</i>	7
2.2.3 <i>MTCSGetADCAVR</i>	8
2.2.4 <i>MTCSGetADCBuf</i>	8
2.2.5 <i>MTCSGetADC</i>	9
2.2.6 <i>MTCSSetSwitch</i>	9
2.2.7 <i>MTCSSetEPoti</i>	9
2.3 MTCS – Functions which are applicable to DARK CCC versions	10
2.3.1 <i>MTCSStartRGB, MTCSStopRGB, MTCSGetRGB</i>	10
2.3.2 <i>MTCSStartRGB</i>	10
2.3.3 <i>MTCSStopRGB</i>	11
2.3.4 <i>MTCSGetRGB</i>	11

MAZeT GmbH Sales Department Göschwitzer Strasse 32 07745 Jena, Germany Tel.: +49 3641 2809-0 Fax: +49 3641 2809-12 Email: sales@MAZeT.de Url: http://www.MAZeT.de	Acknowledgement	Date	MAZeT GmbH	
	Created:	2008-05-26	Status: valid	
	Checked:	2008-08-04		
	Released:	2008-08-04	DOC. NO.: DB-05-171e	Page 1 of 18

Software Description
 MTCS-C2-DLL / MTCS-ME1-DLL (MTCSApi.dll)

VERSION		
NO.	VERSION	APPROVED
1	V 1.9	2008-08-04

2.4 MTCS – Functions which are applicable to the MTCS-C2	12
2.4.1 <i>MTCSGetADCAVR2</i>	12
2.4.2 <i>MTCSGetADCSummen</i>	13
2.4.3 <i>MTCSSetParameter</i>	14
2.4.4 <i>MTCSSearchAmplification</i>	14
2.4.5 <i>MTCSWriteMemToAdr</i>	15
2.4.6 <i>MTCSReadMemFromAdr</i>	16
2.4.7 <i>MTCSGetSerienNummer</i>	16
3 EEPROM.....	17

Software Description
MTCS-C2-DLL / MTCS-ME1-DLL (MTCSApi.dll)

VERSION		
NO.	VERSION	APPROVED
1	V 1.9	2008-08-04

1 Introduction

The document describes the MTCSApi.dll as a gateway between the firmware on the Evaluation Kit MTCS-ME1 (Mod EVA) or the firmware on the board MTCS-C2 (Colorimeter2) and the user interface (host).

A USB 2.0 port is required for the software to be executable. The software was written for WindowsXP© and tested with LAB Windows© und Microsoft C 6.0. Please pay attention to the firmware version for Mod EVA (>V2.22) and / or for Colorimeter2 (V0.13) and the version numbers of the DLL (> V2.43).

The commands for Colorimeter2 only function with DLL Version 2.40 or later.

Please refer to data sheet MTCS-ME1 and / or MTCS-C2 for a better understanding of the hardware functionality.

The basic principle is that the host sends a request as a command to the Mod EVA. After processing the command, the Mod EVA sends its response to the host. The initiative always comes from the host and a response, which is forwarded after the command is sent, is always anticipated.

The MTCSApi.dll uses the USB port for the communication.

The dll is located in the directory of the executable program. The lib necessary for program development is in the directory of the corresponding project.

The used of *USB_DLL_API* in the functions is defined as
#define USB_DLL_API __declspec(dllimport)

VERSION		
NO.	VERSION	APPROVED
1	V 1.9	2008-08-04

2 MTCS – FunCtionS

2.1 MTCS – Functions which are applicable to all MTCS-ME1 versions

2.1.1 MTCSInitSystem

Initialising the system

Definition

```
int USB_DLL_API MTCSInitSystem(char cTyp, int iVendorID, int iProductID );
```

Parameter

char cTyp	Type of connection between the board and the user interface
cTyp = 0	USB 0xff must appear in address 0x3fe in the EEPROM
cTyp = 1	Serial port 0 must appear in address 0x3fe in the EEPROM
int iVendorID = 0x400	Vendor- und Product-ID for Mod EVA
int iProductID = 0xc35d	
int iVendorID = 0x152a	Vendor- und Product-ID for Colorimeter2
int iProductID = 0x8220	

Return value

≥ 0	No errors exist, Number of devices
= -1	Internal initialisation error, reset Mod EVA
= -2	Faulty device notification
= -4	This Mod EVA has already been configured
= -5	Device not found

2.1.2 MTCSCloseDevice

The device will be closed.

Definition

```
int USB_DLL_API MTCSCloseDevice( void);
```

Return value

= 0	No errors exist
!=0	Can´t close the device

Via USB no communication is possible.

VERSION		
NO.	VERSION	APPROVED
1	V 1.9	2008-08-04

2.1.3 MTCSDIIGetVersion

The DLL version is read.

Definition

void USB_DLL_API MTCSDIIGetVersion(char* cBuf);

Parameter

char* cBuf Pointer on the buffer, which contains the DLL version,
 Buffer size for the response from Mod EVA : 5 bytes

2.1.4 MTCSTReadVersion

The current version number of the firmware is retrieved.

Definition

int USB_DLL_API MTCSTReadVersion(char* cBuf);

Parameter

char* cBuf Pointer on the buffer, which contains the firmware version,
 e.g. 0x32, 0x2e, 0x34, 0x30 for version number 2.40
 Buffer size for the response from Mod EVA : 5 bytes, the last
 byte is 0.

Return value

- = 0 No errors exist
- = 1 Parameter error 1
- = 2 Firmware detected an illegal command
- = 7 Transmission error

Comments

With the error code, the status of the firmware initialisation is restored after connection via USB. This command can be used by the application on the host site to test the communication via USB.

2.1.5 MTCSTReadMemory

iAnz integrity values for the contents of the EEPROM are read in bytes successively from address 0.

Definition

int USB_DLL_API MTCSTReadMemory(unsigned char* cBuf, int iAnz);

Parameter

unsigned char* cBuf Pointer on the buffer, which contains the contents of the
 EEPROM,
 Buffer size for the response from Mod EVA : iAnz*2 bytes
 int iAnz The number of integrity values which are to be read.
 (Param1 in the example)

Software Description
 MTCS-C2-DLL / MTCS-ME1-DLL (MTCSApi.dll)

VERSION		
NO.	VERSION	APPROVED
1	V 1.9	2008-08-04

Return value

- = 0 No errors exist
- = 1 Parameter error
- = 2 Firmware detected an illegal command
- = 6 The number of integrity values exceeds the total number for EEPROM
- = 7 Transmission error

Comments

The EEPROM contains, for example, all the correction value data for the sensor calibration. More detailed information can be found under 3. EEPROM.

2.1.6 MTCSWriteMemory

iAnz integrity values for the contents of the EEPROM are always written in bytes successively from address 0.

Definition

int USB_DLL_API MTCSWriteMemory(unsigned char* cBuf, int iAnz);

Parameter

- unsigned char* cBuf Pointer on the buffer, which contains the contents for the EEPROM,
(Param2 in the example)
- int iAnz The number of integrity values which are to be written.
(Param1 in the example)

Return value

- = 0 No errors exist
- = 2 Firmware detected an illegal command
- = 4..x Parameter error Parameter 1, or .. Parameter x
- = 6 The number of integrity values exceeds the total number for EEPROM.
- = 7 Transmission error

2.1.7 MTCSSetShift

This function defers the ADC values to left with the value iShift.

Definition

int USB_DLL_API MTCSSetShift (int ilIndex, unsigned int iShift);

- int ilIndex = 0 reserved for device number
- unsigned int iShift = 0...6

Return value

- = 0 No errors exist
- !=0 Connection error

Comments

The result of the following formula $ADC * 2^x$ is a ADC value with shift. Notice that the noise will be amplified by the bit shift too.

Software Description
MTCS-C2-DLL / MTCS-ME1-DLL (MTCSEApi.dll)

VERSION		
NO.	VERSION	APPROVED
1	V 1.9	2008-08-04

2.1.8 MTCSSetUpdateMode

Shifts the firmware into update mode.

Definition

```
int USB_DLL_API MTCSSetUpdateMode(void);
```

Return value

= 2 Firmware detected an illegal command
= 4..x Parameter error Parameter 1, or .. Parameter x
Other values are not possible as the USB port is no longer in use.

Comments

After PowerUp Reset, the firmware can be updated using the "flip software(ATMEL)" via the USB port. PowerUp Reset resets the Setup mode.

2.2 MTCS – Functions which are applicable to the TOP, FRONT, DARK TIA versions

2.2.1 MTCSSetADCxx

These commands are used for measuring reflective samples or sources with a constant brightness.

The brightness of the LED lighting (only with MTCS-ME1 FRONT and TOP) must be adjusted using MTCSSetEPoti.

“iCounts“ determines the number of measurement values, which are used to take an average.

2.2.2 MTCSSetADCBL

Measurement with stray light compensation

Definition

```
int USB_DLL_API MTCSSetADCBL( unsigned short * usBuf, int iCounts);
```

Parameter

int iCounts The number of measurement values for averaging (Param1 in the example)
unsigned short * usBuf Pointer on the buffer, which contains the ADC values
 Buffer size for the response from Mod EVA : 8 bytes

Return value

= 0 No errors exist
= 2 Firmware detected an illegal command
= 4..x Parameter error Parameter1, or .. Parameter x
= 7 Transmission error

Software Description
 MTCS-C2-DLL / MTCS-ME1-DLL (MTCSApi.dll)

VERSION		
NO.	VERSION	APPROVED
1	V 1.9	2008-08-04

Comments

Stray light compensation is the process whereby measurements are taken in the illuminated and unilluminated condition and the difference is calculated.
 UTD (AD value of the voltage of the isolating diode), URT (AD value Red), UGR (AD value Green), UBL (AD value Blue) are directly read out using iCounts measurements.

2.2.3 MTCSGetADCAVR

Measurement with ADC mean values

Definition

int USB_DLL_API MTCSGetADCAVR(unsigned short * usBuf, int iCounts);

Parameter

int iCounts The number of measurement values for averaging (Param1 in the example)
 unsigned short * usBuf Pointer on the buffer, which contains the ADC values
 Buffer size for the response from Mod EVA : 8 bytes

Return value

- = 0 No errors exist
- = 2 Firmware detected an illegal command
- = 4..x Parameter error Parameter 1, or .. Parameter x
- = 7 Transmission error

Comments

UTD (AD value of the voltage of the isolating diode), URT (AD value Red), UGR (AD value Green), UBL (AD value Blue) are directly read out using iCounts measurements.

2.2.4 MTCSGetADCBuf

Readout the Buffer of the Measurement values

Definition

int USB_DLL_API MTCSGetADCBuf(unsigned short * usBuf);

Parameter

unsigned short * usBuf Pointer on the buffer, which contains the ADC values
 Buffer size for the response from Mod EVA : 8 bytes

Return value

- = 0 No errors exist
- = 2 Firmware detected an illegal command
- = 4..x Parameter error Parameter 1, or .. Parameter x
- = 7 Transmission error

Comments

The ADC values UTD, URT, UGR, UBL are read from the buffer and the red LED is reset. This occurs by means of a keystroke for the measurement.

VERSION		
NO.	VERSION	APPROVED
1	V 1.9	2008-08-04

2.2.5 MTCSGetADC

Simple measurement

Definition

int USB_DLL_API MTCSGetADC(unsigned short * usBuf);

Parameter

unsigned short * usBuf Pointer on the buffer, which contains the ADC values
 Buffer size for the response from Mod EVA : 8 bytes

Return value

- = 0 No errors exist
- = 2 Firmware detected an illegal command
- = 4..x Parameter error Parameter 1, or .. Parameter x
- = 7 Transmission error

Comments

The ADC values UTD (AD value of the voltage of the isolating diode), URT (AD value Red), UGR (AD value Green), UBL (AD value Blue) are directly read out.

2.2.6 MTCSSetSwitch

The switch for transimpedance amplification of the isolating diode is set.

Definition

int USB_DLL_API MTCSSetSwitch(int iCounts);

Parameter

int iCounts Value, at which the switch is to be set
 0(off) oder 4(on)
 (Param1 in the example)

Return value

- = 0 No errors exist
- = 2 Firmware detected an illegal command
- = 4..x Parameter error Parameter 1, or .. Parameter x
- = 7 Transmission error

2.2.7 MTCSSetEPoti

The brightness of the LED lighting (only with MTCS-ME1 FRONT and TOP) is adjusted.

Definition

int USB_DLL_API MTCSSetEPoti(int iCounts);

Parameter

int iCounts Value, at which the Epoti is to be set
 (Param1 in the example)

Software Description
MTCS-C2-DLL / MTCS-ME1-DLL (MTCSApi.dll)

VERSION		
NO.	VERSION	APPROVED
1	V 1.9	2008-08-04

Return value

- = 0 No errors exist
- = 2 Firmware detected an illegal command
- = 4..x Parameter error Parameter 1, or .. Parameter x
- = 7 Transmission error

Comments

The E-Poti is adjusted to the assigned value (0x00...0xFF).

2.3 MTCS – Functions which are applicable to DARK CCC versions

2.3.1 MTCSSstartRGB, MTCSSstopRGB, MTCSSgetRGB

These commands are used for measuring light sources e.g. with illumination LEDs and monitors.

The iTime parameter is entered into the integration time. It is variable between 5000µs and 25000µs.

It corresponds to the synchronisation of source frequencies (e.g. CRT monitors) 200Hz to 40Hz.

A measurement is carried out within a multiple of the given integration time. The measurement value (“ADC / number of iterations“) represents the integral deviation of the signal level divided by the number of integration intervals (“number of iterations“). With the “iZyklen“ parameter, the stop criterion, the maximum number of recorded integration intervals, is defined for a measurement cycle.

2.3.2 MTCSSstartRGB

The interval time and the maximum number of integration cycles is defined. RGB integration is started.

Definition

int USB_DLL_API MTCSSstartRGB(unsigned short* usBuf, int iTime, int iZyklen);

Parameter

- unsigned short* usBuf Pointer on the buffer, which contains the outcome message
Buffer size for the response from Mod EVA : 2 bytes
- int iTime Interval time
(Param1 in the example)
- int iZyklen max. integration cycles
(Param2 in the example)

Return value

- = 0 No errors exist
- = 2 Firmware detected an illegal command
- = 4..x Parameter error Parameter 1, or .. Parameter x
- = 7 Transmission error

VERSION		
NO.	VERSION	APPROVED
1	V 1.9	2008-08-04

2.3.3 MTCSStopRGB

The command stops a measurement and reads out the current RGB values and the number of intervals, which have occurred.

Definition

```
int USB_DLL_API MTCSStopRGB( unsigned short* usBuf);
```

Parameter

unsigned short* usBuf Pointer on the buffer, which contains the outcome message (red, green, blue and number of cycles for red, number of cycles for green, number of cycles for blue.
Buffer size for the response from Mod EVA : 12 bytes

Return value

- = 0 No errors exist
- = 2 Firmware detected an illegal command
- = 4..x Parameter error Parameter 1, or .. Parameter x
- = 7 Transmission error

Comments

The MTCSStopRGB command is not required during normal operation because the measurement is terminated after the number of iterations.

2.3.4 MTCGetRGB

The command reads out the current RGB values and the number of iterations, which have occurred.

Definition

```
int USB_DLL_API MTCGetRGB( unsigned short* usBuf);
```

Parameter

unsigned short* usBuf Pointer on the buffer, which contains the outcome message (red, green, blue and number of iterations for red, number of iterations for green, number of iterations for blue.
Buffer size for the response from Mod EVA : 12 bytes

Return value

- = 0 No errors exist
- = 2 Firmware detected an illegal command
- = 4..x Parameter error Parameter 1, or .. Parameter x
- = 7 Transmission error

VERSION		
NO.	VERSION	APPROVED
1	V 1.9	2008-08-04

2.4 MTCS – Functions which are applicable to the MTCS-C2

2.4.1 MTCSSGetADCAVR2

The amplification and ADC mean values URT (AD value Red), UGR (AD value Green), UBL (AD value Blue) are directly read out using iCounts measurements.

Definition

```
int USB_DLL_API MTCSSGetADCAVR2( int iIndex, unsigned short* usBuf, int iCounts);
```

Parameter

int iIndex	= 0	reserved for device number
int iCounts		The number of measurement values for averaging (Param1 in the example)
unsigned short* usBuf		Pointer on the buffer, which contains the amplification factor and the ADC values Buffer size for the response from Col2 : 8 bytes

Return value

= 0	No errors exist
= 2	Firmware detected an illegal command
= 3	Difference in measurement values is too large
= 4..x	Parameter error Parameter 1, or .. Parameter x
= 7	Transmission error

Comments

The amplification can be set using MTCSSsetParameter or determined automatically using MTCSSsearchAmplification.

If the amplification factor is 0, the optimum amplification factor for the series of measurements is sought first. In addition, AD conversions on the red, green and blue channels are carried out until all channels provide values within a defined ADC operating range.

If the amplification factor is in the range (1, 8), the “automatic search of the optimum amplification factors” stage does not occur.

N series of measurements are carried out and the average value is generated.

If the subsequent measurement values differ from the first measurement value by more than 10%, then these are logged as error code 0xE003. (This serves as a reference to major sources of interference or pulsed light sources for which the measurement procedure is not suitable.)

These 3 mean values and the adjusted amplification factor are sent in the response to the host.

VERSION		
NO.	VERSION	APPROVED
1	V 1.9	2008-08-04

2.4.2 MTCSGetADCSummen

The amplification and ADC mean values URT (AD value Red), UGR (AD value Green), UBL (AD value Blue) are added together and read out using iCounts milliseconds.

Definition

int USB_DLL_API MTCSGetADCSummen(int iIndex, unsigned long ulBuf, int iCounts);*

Parameter

int iIndex = 0 reserved for device number
 int iCount The number of milliseconds which are to be measured for the totals formation.
 (Param1 in the example)
 unsigned long * ulBuf Pointer on the buffer, which contains the amplification factor and the ADC values and the total number of measurements
 Buffer size for the response from Col2 : 20 bytes

Return value

= 0 No errors exist
 = 2 Firmware detected an illegal command
 = 3 Difference in measurement values is too large
 = 4..x Parameter error Parameter 1, or .. Parameter x
 = 7 Transmission error

Comments

The amplification can be set using MTCSSetParameter or defined automatically using MTCSSearchAmplification.
 If the amplification factor is 0, the optimum amplification factor for the series of measurements is sought first. In addition, AD conversions on the red, green and blue channels are carried out until all channels provide values within a defined ADC operating range.
 If the amplification factor is in the range (1, 8), the “automatic search of the optimum amplification factors” stage does not occur.
 N series of measurements are carried out using iCounts milliseconds and the values are added together.
 The adjusted amplification factor, the sum of the 3 AD values and the total number of measurements are sent in the response to the host.
 All values are unsigned long values.

VERSION		
NO.	VERSION	APPROVED
1	V 1.9	2008-08-04

2.4.3 MTCSSetParameter

The amplification factor and tolerance parameters are set.

Definition

int USB_DLL_API MTCSSetParameter (int iIndex, int iAmpl, int iTol);

Parameter

int iIndex = 0 reserved for device number
 int iAmpl Amplification factor
 int iTol Tolerance
 (Parm1 and Parm2 the example)

Return value

= 0 No errors exist
 = 2 Firmware detected an illegal command
 = 7 Transmission error

Comments

The default values are: Amplification factor = 8 Tolerance = 90(percent)
 Both parameters are included in the “automatic search of the optimum amplification factors“ stage.

2.4.4 MTCSSearchAmplification

Search for the optimum amplification factor

Definition

int USB_DLL_API MTCSSearchAmplification(int iIndex, unsigned short* usBuf, int iLimit);

Parameter

int iIndex = 0 reserved for device number
 int iLimit Limit values (in %), 1 byte
 unsigned short * usBuf Pointer on the buffer, which contains the amplification
 factor and the ADC values
 Buffer size for the response: 8 bytes

Return value

= 0 No errors exist
 = 2 Firmware detected an illegal command
 = 7 Transmission error

Comments

The optimum amplification factor for the series of measurements is sought. In addition, AD conversions on the red, green and blue channels are carried out with a decreasing amplification factor until the strongest channel provides a signal which is below the limit value (value range 1..100%).
 The measurement values are expanded according to the bit shifting parameter.

Software Description
 MTCS-C2-DLL / MTCS-ME1-DLL (MTCSEApi.dll)

VERSION		
NO.	VERSION	APPROVED
1	V 1.9	2008-08-04

The located amplification factor and the last 3 measurement values from the algorithm are sent in the response to the host.
 The located amplification factor is set for the following measurements.

This algorithm can also be used to bring the AD conversions into an optimum operating range. This is essential if, due to very bright light sources, the color sensor is overloaded when the amplification factor is at its highest. Therefore, the AD conversion would always provide the maximum AD values.

2.4.5 MTCSWriteMemToAdr

iAnz integrity values for the contents of the EEPROM are always written in bytes successively from address iAdr.

Definition

int USB_DLL_API MTCSWriteMemToAdr(int iIndex, unsigned char* cBuf, int iAdr, int iAnz)

Parameter

int iIndex = 0 reserved for device number
 unsigned char* cBuf Pointer on the buffer, which contains the contents for the EEPROM, (Param1 in the example)
 int iAdr Address, from which the EEPROM is to be rewritten. (Param2 in the example)
 int iAnz The number of integrity values which are to be written. (=1 in the example)

Return value

= 0 No errors exist
 = 2 Firmware detected an illegal command
 = 4..x Parameter error Parameter 1, or .. Parameter x
 = 6 The number of integrity values exceeds the total number for EEPROM
 = 7 Transmission error

Comments

iAnz items are written as a data stream in the external EEPROM from a defined address iAdr.
 iAnz = (1, 57)

The size of the EEPROM is 128 bytes. It can also be addressed in the range (0x00, 0x6F). If address + 2*number data items to be written is larger than the above address, error code 6 is restored.

VERSION		
NO.	VERSION	APPROVED
1	V 1.9	2008-08-04

2.4.6 MTCSReadMemFromAdr

iAnz integrity values for the contents of the EEPROM are always read in bytes successively from address iAdr.

Definition

```
int USB_DLL_API MTCSReadMemFromAdr(int iIndex, unsigned char* cBuf, int iAdr, int iAnz);
```

Parameter

int iIndex	= 0	reserved for device number
unsigned char* cBuf		Pointer on the buffer, which contains the contents of the EEPROM, Buffer size for the response from Colorimeter2 : iAnz*2 bytes
int iAdr		Address, from which the EEPROM is to be rewritten. (Param1 in the example)
int iAnz		The number of integrity values which are to be read. (Param2 in the example)

Return value

= 0	No errors exist
= 1	Parameter error
= 2	Firmware detected an illegal command
= 6	The number of integrity values exceeds the total number for EEPROM
= 7	Transmission error

Comments

iAnz values are read as a data stream by the external EEPROM from a defined address iAdr.

The size of the EEPROM is 128 bytes. It can also be addressed in the range (0x00, 0x6F). If address + 2* iAnz of data items to be read is larger than the above address, error code 6 is restored.

2.4.7 MTCSGetSerienNummer

Read the serial numbers from the board's EEPROM

Definition

```
void USB_DLL_API MTCSGetSerienNummer (int iIndex, char* cBuf);
```

Parameter

int iIndex	= 0	reserved for device number
char* cBuf		Pointer on the buffer, which contains the serial numbers which are in the USB descriptor. Buffer size for the response from Colorimeter2 : 16 bytes

Return value

= 0	No errors exist
= 2	Firmware detected an illegal command
= 7	Transmission error

VERSION		
NO.	VERSION	APPROVED
1	V 1.9	2008-08-04

3 EEPROM

The contents of the EEPROM is structured.
 System parameters are data which are saved permanently.
 Access to the data is gained using the ReadMemory, WriteMemory, WriteMemToAdr and ReadMemFromAdr commands. These commands treat the memory as a data stream.

The maximum size of the EEPROM of the Mod EVA is 1024 bytes. The last 10 bytes are reserved for internal characteristics. Only essential data should be written to the EEPROM because writing to the EEPROM is a slow process.

No.	Contents	Length in WORD
1.	Application code	1
2.	E-Poti	1
3.	Dark value	3
4.	White value	3
5.	BL (stray light compensation 0=off, 1=on)	1
6.	Number of average value	1
7.	Integration time	1
8.	Correction matrix sensor	9
9.	Correction matrix display	9
10.	Free memory	

Table 1: Overview relating to the system parameters in the EEPROM of the Mod EVA

The maximum size of the EEPROM of the Colorimeter 2 is 128 bytes.
 The top 16 bytes are reserved for the serial numbers.

No	Contents	Length in WORD	Comments
1	Application code	1	The application code for the Colorimeter 2 is 0x0011
2	E-Poti	1	No relevance
3	Dark value	3	Is the parameter for the PC application
4	White value	3	Is the parameter for the PC application
5	Stray light compensation BL	1	No relevance
6	Number of average values	1	For GetADCAvr2
7	Amplification factor	1	For GetADC2, GetADCAvr2
8	Correction matrix sensor	9	is the parameter for the PC application
9	Correction matrix display	9	is the parameter for the PC application
10			

Table 2: Overview relating to the system parameters in the EEPROM of the Colorimeter2

The system is initialised during startup using MTCSEInitSystem().

Software Description
MTCS-C2-DLL / MTCS-ME1-DLL (MTCSApi.dll)

VERSION		
NO.	VERSION	APPROVED
1	V 1.9	2008-08-04

For further information please contact:

MAZeT GmbH
Sales Department:
Göschwitzer Strasse 32
07745 Jena, Germany
Tel: +49 3641 2809-0
Fax: +49 3641 2809-12
Email: sales@MAZeT.de
Website: <http://www.MAZeT.de>