

SMART 4.3“ TFT DISPLAY

480x272 Built-in Intelligence, Fonts and Memory



FEATURES

- * POWER SUPPLY +5V / 180mA(typ.)
- * TFT GRAPHIC DISPLAY WITH BULIT-IN GRAPHIC FUNCTIONS
- * 480x272 DOTS, 16-BIT (65,536) COLORS WITH LED BACKLIGHT
- * LANDSCAPE AND PORTRAIT MODE (272x480) BY COMMAND
- * 4MB ON BOARD FLASH FOR FONTS, PICTURES, ANIMATIONS AND MACROS
- * 8 PRE-DEFINED FONTS, CAN BE EXPANDED
- * FONT ZOOM FROM 2mm TO about 80mm, TURNABLE IN 90° STEPS
- * 3 DIFFERENT INTERFACES ON BOARD: RS-232, I²C-BUS OR SPI-BUS
- * POSITIONING ACCURATE TO THE PIXEL WITH ALL FUNCTIONS
- * DRAW LINE, PLACE A DOT, AREA, BARGRAPH...
- * PICTURES, ANIMATIONS; MIX TEXT AND GRAPHIC
- * MULTI-LINGUAL WITH MACRO PAGES
- * BACKLIGHT BRIGHTNESS BY SOFTWARE
- * ANALOGUE OR CAPACITIVE TOUCH PANEL: VARIABLE GRID
- * FREE DEFINABLE KEY AND SWITCH
- * 8 DIGITAL IN- AND 8 DIGITAL OUTPUT
- * 2 ANALOGUE INPUTS COMFORTABLE TO USE

ORDERING CODES

DISPLAYS

TFT 480x272 DOTS, WHITE LED BACKLIGHT

EA eDIPTFT43-A

AS ABOVE, BUT WITH RESISTIVE TOUCH PANEL

EA eDIPTFT43-ATP

AS ABOVE, BUT WITH CAPACITIVE TOUCH PANEL (SAME SIZE AS -ATP)

EA eDIPTFT43-ATS

AS ABOVE, BUT WITH CAPACITIVE MOUNTING PANEL

EA eDIPTFT43-ATC

STARTERKIT

INCLUDES EA eDIPTFT43-ATP AND EVALUATION BOARD WITH USB FOR DIRECT CONNECTION TO PC AND INTERFACE BOARDS FOR CONNECTION WITH YOUR HOST SYSTEM

EA EVALeDIPTFT43

AS ABOVE, BUT WITH EA eDIPTFT43-ATC

EA EVALeDIPTFT43C

ADDITIONAL PARTS

MOUNTING BEZEL (ALUMINIUM), BLACK ANODIZED

EA 0FP481-43SW

SOCKET 1x20, 4.5mm HIGH (1 piece)

EA B254-20

Documentation of revision				
Date	Software	Hardware	New	Reason / Description
July, 22nd. 2008	1.0	Rev.D		1st. Edition
March 2009	1.1	Rev.E	- additional command: ESC YD, ESC VM, ESC YX - bug fix	
June 2009	1.2		- additional command: ESC ZB - bug fix	
November 2009	1.3		- bug fix "bargraph" and "clear touch"	
May 2011	1.4		- additional functions: Instruments, extended Ports	
January 2012	1.5		- additional functions: X/Y-graph, String table	
July 2013	1.6	Rev.F Requires 1.6 or higher	- page 32: thickness corrected to 10.4/11.9mm	SMD pin header and new TFT panel
Oct 2014	1.7	Rev.G Requires 1.7 or higher	- new hardware option "capacitive Touchpanel" - new dimension drawing for EA eDIPTFT43-ATC	
July 2016	1.7	Rev.G Requires 1.7 or higher	- new hardware option "capacitive Touchpanel" as replacement for EA eDIPTFT43-ATP - new dimension drawing for EA eDIPTFT43-ATS	
August 2016	1.8		- supports Adesto FLASH - adjustable display timing	Component obsolescence
February 2017	1.9		- improved PCAP detection	
July 2020	1.9		- change TFT into IPS technology	Improve viewing angle, color stability and brightness

CONTENTS

GENERAL	3
RS-232, RS485, USB	4
SPI	6
I ² C	7
ANALOGUE / DIGITAL IN- AND OUTPUT	8
MATRIX KEYPAD	9
SOFTWARE PROTOCOL	10 - 11
TERMINAL MODE, COMMAND TRANSFER	12
COMMANDS / FUNCTIONS IN TABULAR FORMAT	13 - 17
TOUCH PANEL	18
RESPONSES OF THE CONTROL PANEL	19
CHARACTER SET	20-22
COLORS	22
FRAMES, KEY STYLE AND PATTERN	23-24
INSTRUMENTS	24-25
PROGRAMMING: FONTS, PICTURES, ANIMATIONS	26
BITMAPS AS BUTTONS	27
MACROS, MULTI-LINGUAL, MACRO PAGES	28-29
ELECTRICAL CHARACTERISTICS	30
DIMENSIONS, MOUNTING PANEL	31 - 32

GENERAL

The EA eDIP series of displays are the world's first displays with integrated intelligence. In addition to a variety of integrated fonts that can be used with pixel accuracy, they offer a whole range of sophisticated graphics functions.

The displays are ready for operation immediately with an operating voltage of 5V. They are controlled via one of the 3 integrated interfaces: RS-232, SPI or I²C. The displays are "programmed" by means of high-level language-type graphics commands. There is no longer any need for the time-consuming programming of character sets and graphics routines. The ease of use of this display with its touch panel dramatically reduces development times.

HARDWARE

The display is designed to work at an operating voltage of +5V. Data transfer is either serial and asynchronous in RS-232 format or synchronous via the SPI or I²C specification. To improve data security, a simple protocol is used for all types of transfer.

TOUCH PANEL

All versions are also available with an integrated touch panel: You can make entries and menu or bar graph settings by touching the display. The labeling of the "keys" is flexible and can also be changed during runtime (different languages, icons). The drawing of the individual "keys" and the labeling is handled by the integrated software. Depending on the field of application, you can use either the analogue resistive version, designed for the use with pens, thick gloves, or the capacitive version with its immune hard glass surface.

LED BACKLIGHT

All displays are equipped with modern, energy-saving LED illumination. Brightness can be varied 0~100% by command.

In 24-hour operation, the illumination should be dimmed or switched off as often as possible to increase their lifetime.

SOFTWARE

This display is programmed by means of commands, such as *Draw a rectangle from (0,0) to (479,271)*. No additional software or drivers are required. Strings and images can be placed with **pixel accuracy**. Text and graphics can be combined at any time. Different character sets can be used at same time. Each character set and the images can be zoomed from 2 to 8 times and rotated in 90° steps. With the largest character set, the words and numbers displayed will fill the screen.

ACCESSORIES

Evaluation-Board (Programmer) for internal data flash memory

The display is shipped fully programmed and with all fonts. The additional Evaluation-Board is thus generally not required.

However, if the internal character sets have to be changed or extended, or if images or macros have to be stored internally, the Evaluation-Board EA 9777-2USB, which is available as an accessory, will burn the data/images you have created into the on-board data flash memory (4 MB) permanently.

The Evaluation-Board runs under Windows and is connected to the PC's USB interface. It is shipped with an interface cable and the installation software. The Evaluation-Board is equipped with several LEDs, pushbuttons and potentiometer to test all peripheral modes of the eDIP.

Interface-Expansion for Evaluation-Board (included in the Starter-Kit):

With the expansion EA 9777-2PE for the Evaluation-Board all interfaces of the display are made available with the help from small adapter boards: RS-232, RS-485, SPI, I²C, RS-232 (CMOS level). Further information you will find in the datasheet of the Evaluation-Board.

RS-232 INTERFACE

If the display is wired as shown below, the RS-232 interface is selected. The pin assignment is specified in the table on the right.

The RxD and TxD lines lead 5V (CMOS level) to a microcontroller, for example, for direct connection.

If "genuine" RS-232 levels are required (e.g. for connection to a PC), an external level converter (e.g. MAX232) is required.

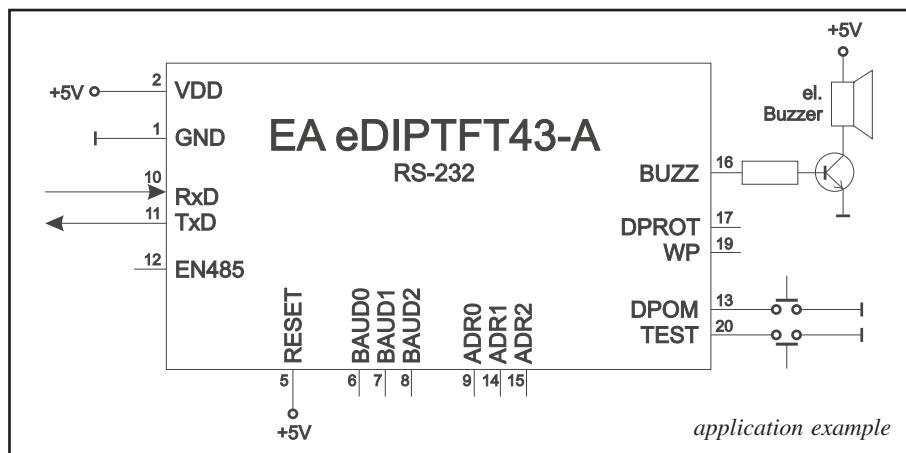
Pinout eDIPTFT43-A: RS-232/RS-485 mode			
Pin	Symbol	In/Out	Function
1	GND		Ground Potential for logic (0V)
2	VDD		Power supply for logic (+5V)
3	NC		do not connect
4	NC		do not connect
5	RESET	In	L: Reset
6	BAUD0	In	Baud Rate 0
7	BAUD1	In	Baud Rate 1
8	BAUD2	In	Baud Rate 2
9	ADR0	In	Address 0 for RS-485
10	RxD	In	Receive Data
11	TxD	Out	Transmit Data
12	EN485	Out	Transmit Enable for RS-485 driver
13	DPOM	In	L: disable PowerOnMacro do not connect for normal operation
14	ADR1	In	Address 1 for RS-485
15	ADR2	In	Address 2 for RS-485
16	BUZZ	Out	Buzzer output
17	DPROT	In	L: Disable Smallprotokoll do not connect for normal operation
18	DNC	Out	L: internal, do not connect
19	WP	In	L: Writeprotect for DataFlash
20	TEST SBUF	IN Out	open-drain with internal pullup 20..50k IN (Power-On) L: Testmode OUT L: data in sendbuffer
21	GND		Ground (0V)
22	VDD		Power supply (+5V)
23	AIN1	In	analogue input 0..5V DC impedance 1MOhm
24	AIN2		
25	OUT1 / MO8		
26	OUT2 / MO7		
27	OUT3 / MO6		
28	OUT4 / MO5		
29	OUT5 / MO4		
30	OUT6 / MO3		
31	OUT7 / MO2		
32	OUT8 / MO1		
33	IN1 / MI8		
34	IN2 / MI7		
35	IN3 / MI6		
36	IN4 / MI5		
37	IN5 / MI4		
38	IN6 / MI3		
39	IN7 / MI2		
40	IN8 / MI1		

BAUD RATES

The baud rate is set by means of pins 6, 7 and 8 (baud 0 to 2). The data format is set permanently to 8 data bits, 1 stop bit, no parity.

RTS/CTS handshake lines are not required. The required control is taken over by the integrated software protocol (see pages 10 and 11).

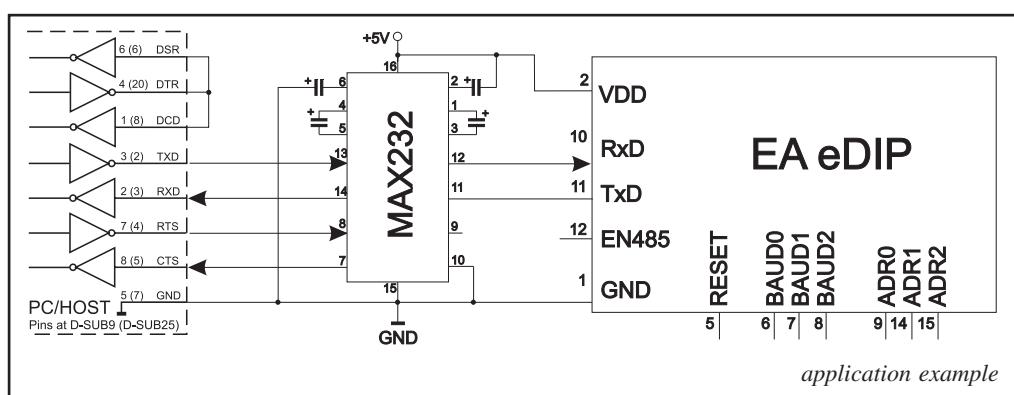
Baud Rates			
Baud0	Baud1	Baud2	data format 8,N,1
1	0	0	2400
0	1	0	4800
1	1	0	9600
0	0	1	19200
1	0	1	38400
0	1	1	57600
1	1	1	115200
0	0	0	230400



EA eDIPTFT43-A

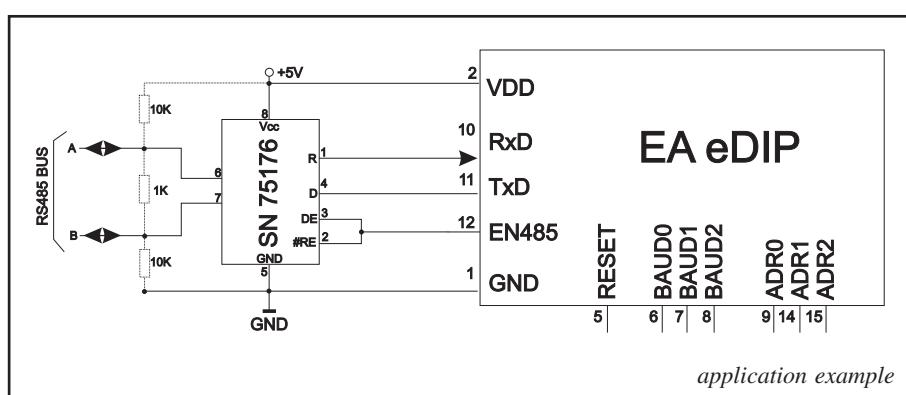
APPLICATION EXAMPLE „REAL“ RS-232 INTERFACE

The eDIP fits for direct connection to a RS-232 interface with CMOS level (VDD).
If you have an interface with $\pm 12V$ level, an external levelshifter is needed



APPLICATION EXAMPLE: RS-485 INTERFACE

With an external converter (e.g. SN75176), the EA eDIP can be connected to a 2-wire RS-485 bus. Large distances of up to 1200 m can thus be implemented (remote display). Several EA eDIP displays can be operated on a single RS-485 bus by setting addresses.

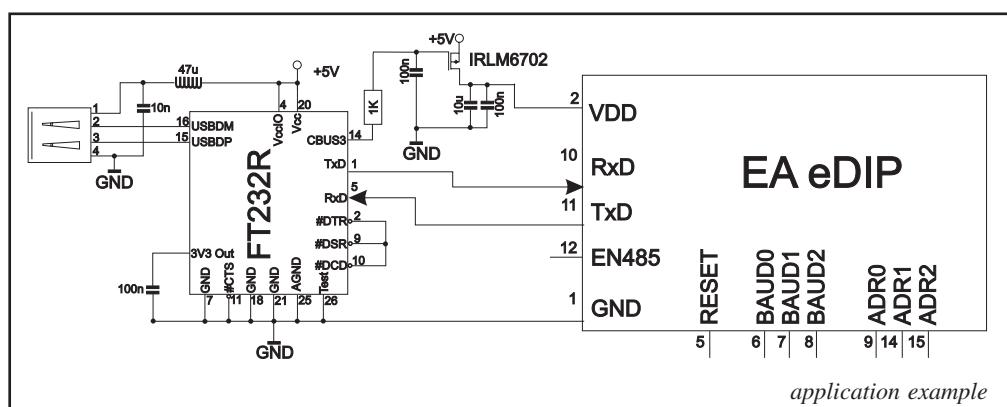


Addressing:

- Up to eight hardware addresses (0 to 7) can be set by means of Pins ADR0..ADR2
- The eDIP with the address 7 is selected and ready to receive after power-on.
- The eDIPS with the addresses 0 to 6 are deselected after power-on
- Up to 246 further software addresses can be set by means of the '#KA adr' command in the power-on macro (set eDIP externally to address 0)

APPLICATION EXAMPLE: USB INTERFACE

With an external converter (e.g. FT232R) from FTDI the eDIP can be connected to an USB-Bus. Virtual-COM-Port drivers are available for different Systems on the FTDI Homepage:
<http://www.ftdichip.com/drivers/vcp.htm>.



SPI INTERFACE

If the display is wired as shown below, SPI mode is activated. The data is then transferred via the serial, synchronous SPI interface.

The transfer parameter will be set via the pins DORD, CPOL and CPHA.

Pinout eDIPTFT43-A: SPI mode			
Pin	Symbol	In/Out	Function
1	GND		Ground Potential for logic (0V)
2	VDD		Power supply for logic (+5V)
3	NC		do not connect
4	NC		do not connect
5	RESET	In	L: Reset
6	SS	In	Slave Select
7	MOSI	In	Serial In
8	MISO	Out	Serial Out
9	CLK	In	Shift Clock
10	DORD	In	Data Order (0=MSB first; 1=LSB first)
11	SPIMO	In	connect to GND for SPI interface
12	NC		do not connect
13	DPOM	In	L: disable PowerOnMacro do not connect for normal operation
14	CPOL	In	Clock Polarity (0=LO 1=HI when idle)
15	CPHA	In	Clock Phase sample 0=1st;1=2nd edge
16	BUZZ	Out	Buzzer output
17	DPROT	In	L: Disable Smallprotokoll do not connect for normal operation
18	DNC	Out	L: internal, do not connect
19	WP	In	L: Writeprotect for DataFlash
20	TEST	IN	open-drain with internal pullup 20..50k
	SBUF	Out	IN (Power-On) L: Testmode OUT L: data in sendbuffer
21	GND		Ground (0V)
22	VDD		Power supply (+5V)
23	AIN1		In analogue input 0..5V DC impedance 1MOhm
24	AIN2		
25	OUT1 / MO8		8 digital outputs maximum current: IOL = IOH = 10mA
26	OUT2 / MO7		
27	OUT3 / MO6		
28	OUT4 / MO5		
29	OUT5 / MO4		Out alternativ up to 8 matrix keyboard output lines (reduces the digital output lines, see chapter external keyboard)
30	OUT6 / MO3		
31	OUT7 / MO2		
32	OUT8 / MO1		
33	IN1 / MI8		
34	IN2 / MI7		
35	IN3 / MI6		8 digital inputs open-drain with internal pullup 20..50k
36	IN4 / MI5		
37	IN5 / MI4		
38	IN6 / MI3		
39	IN7 / MI2		
40	IN8 / MI1		

Note:

The pins DORD, CPOL, CPHA, DPOM, DPROT and TEST/SBUF have an internal pullup, which is why only the LO level (0=GND) is to be actively applied. These pins must be left open for a Hi level.

On pin 20 (SBUF) the display indicates with a low level that data is ready to be retrieved from the internal send buffer.

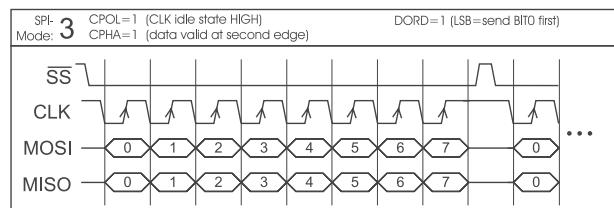
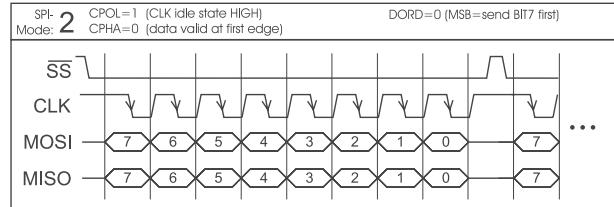
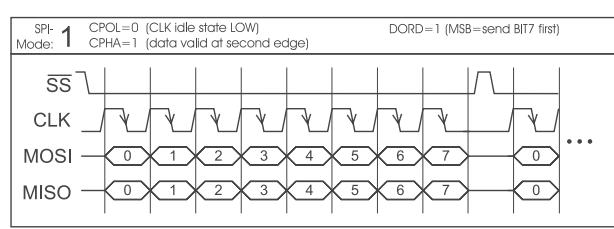
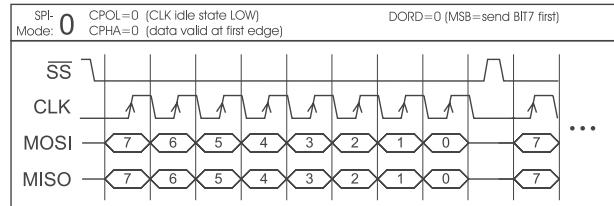
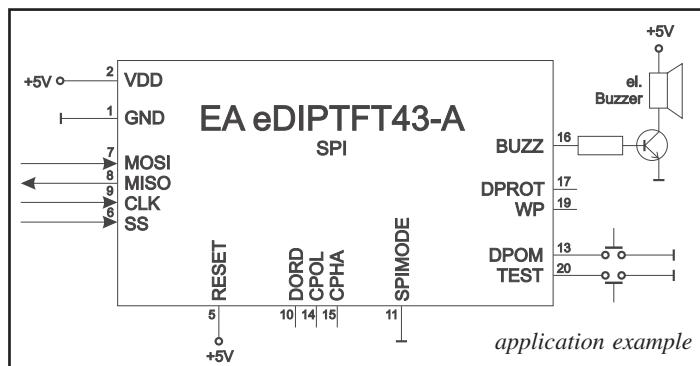
The line can be connected to an interrupt input of the host system, for example.

DATA TRANSFER SPI

Write operation: a clock rate up to 200 kHz is allowed without any stop. Together with a pause of 100 µs between every data byte a clock rate up to 3 MHz can be reached.

Read operation: to read data (e.g. the "ACK" byte) a dummy byte (e.g. . 0xFF) need to be sent.

Note that the EA eDIP for internal operation does need a short time before providing the data; therefore a short pause of min. 6µs (no activity of CLK line) is needed for each byte.



EA eDIPTFT43-A

I²C-BUS INTERFACE

If the display is wired as shown below, it can be operated directly on an I²C bus.

8 different base addresses and 8 slave addresses can be selected on the display.

Data transfer is possible at up to 100 kHz. However, if pauses of at least 100 µs are maintained between the individual bytes during transfer, a byte can be transferred at up to 400 kHz.

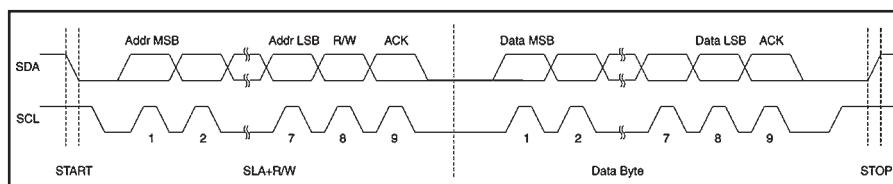
Pinout eDIPTFT43-A: I ² C mode			
Pin	Symbol	In/Out	Function
1	GND		Ground Potential for logic (0V)
2	VDD		Power supply for logic (+5V)
3	NC		do not connect
4	NC		do not connect
5	RESET	In	L: Reset
6	BA0	In	Basic Address 0
7	BA1	In	Basic Address 1
8	SA0	In	Slave Address 0
9	SA1	In	Slave Address 1
10	SA2	In	Slave Address 2
11	BA2	In	Basic Address 2
12	I ² CM0	In	connect to GND for I ² C interface
13	DPOM	In	L: disable PowerOnMacro do not connect for normal operation
14	SDA	Bidir.	Serial Data Line
15	SCL	In	Serial Clock Line
16	BUZZ	Out	Buzzer output
17	DPROT	In	L: Disable Smallprotokoll do not connect for normal operation
18	DNC	Out	L: internal, do not connect
19	WP	In	L: Writeprotect for DataFlash
20	TEST SBUF	IN Out	open-drain with internal pullup 20..50k IN (Power-On) L: Testmode OUT L: data in sendbuffer
21	GND		Ground (0V)
22	VDD		Power supply (+5V)
23	AIN1	In	analogue input 0..5V DC impedance 1MΩ
24	AIN2		
25	OUT1 / MO8		
26	OUT2 / MO7		
27	OUT3 / MO6		
28	OUT4 / MO5		
29	OUT5 / MO4		
30	OUT6 / MO3		
31	OUT7 / MO2		
32	OUT8 / MO1		
33	IN1 / MI8		
34	IN2 / MI7		
35	IN3 / MI6		
36	IN4 / MI5		
37	IN5 / MI4		
38	IN6 / MI3		
39	IN7 / MI2		
40	IN8 / MI1		

Note:

The pins BA0 to 2, SA0 to 2, DPOM, DPROT and TEST/SBUF have an internal pullup, which is why only the LO level (L=0=GND) is to be actively applied. These pins must be left open for a Hi level (H=1).

On pin 20 (SBUF) the display indicates with a low level that data is ready to be retrieved from the internal send buffer.

The line can be connected to an interrupt input of the host system, for example.



I ² C - Address				
Pin 11,7,6	Base	I ² C address		
BA2	BA1	BA0	address	
L	L	L	\$10	
L	L	H	\$20	
L	H	L	\$30	
L	H	H	\$40	
H	L	L	\$70	
H	L	H	\$90	
H	H	L	\$B0	
H	H	H	\$D0	
S A 2		S A 1	S A 0	
R W				

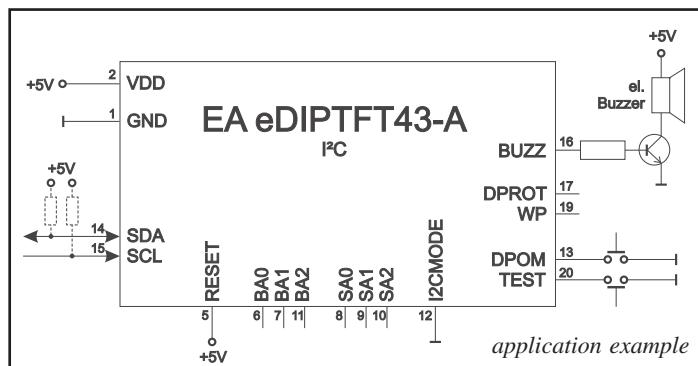
all pins open: Write \$DE
Read \$DF

DATA TRANSFER I²C-BUS

principle I²C-bus transfer:

- I²C-Start
- Master-Transmit: EA eDIP-I²C-address (e.g. \$DE), send smallprotocol package (data)
- I²C-Stop
- I²C-Start
- Master-Read: EA eDIP-I²C-Address (e.g. \$DF), read ACK-byte and opt. smallprotocoll package (data)
- I²C-Stop

Read operation: for internal operation the EA eDIP does need a short time before providing the data; therefore a short pause of min. 6µs is needed for each byte (no activity of SCL line).



ANALOGUE INPUT AIN1 AND AIN2 (PIN 23+24)

For analogue measurement 2 inputs with a range of 0..VDD are available. Each input is grounded (GND) and DC impedance is $1\text{M}\Omega$. Please make sure that only positive voltages will be supplied there. Internal resolution is 10 Bit, equal to a 3-digit DVM modul. Linearity (after adjustment) is around 0.5%.

Adjustment

Analogue inputs are not calibrated when shipped out. A procedure for adjustment may be like that:

- 1.) Put a well known voltage within a range of 2-VDD to analogue input (example: 3,0V, AIN1)
- 2.) Run command for calibration (see page 16). Example: „ESC V @ 1 3000“.

Each input query can be done via serial interface or directly shown on display (as digits or bargraph in various colors and sizes).

Best way for direct visualisation are Process-macros or one of Analogue-macros (e.g. starting at every voltage change, or above/below a limit).

Both input lines are scaleable from 0 to ± 9999.9 . Scaling will be done via definition at 2 voltages „value1=string1;value2=string2“ (see table on page 16).

DIGITAL INPUT AND OUTPUT

The EA eDIP is featured with 8 digital input and 8 digital output lines (CMOS level, grounded).

8 outputs (Pin 25-32)

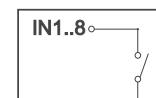
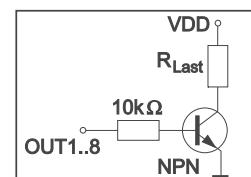
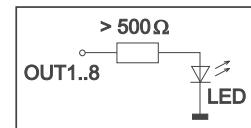
Each line can be controlled individually using the „ESC Y W“ command. A maximum current of 10mA can be switched per line. This give the opportunity to drive a low power LED in direct way. To source higher current please use an external transistor.

8 inputs (Pin 33-40)

Each input provides an internal 20..50 k Ω pull-up resistor, so it is possible to connect a key or switch directly between input and GND. The inputs can be queried and evaluated directly via the serial interface („ESC Y R“).

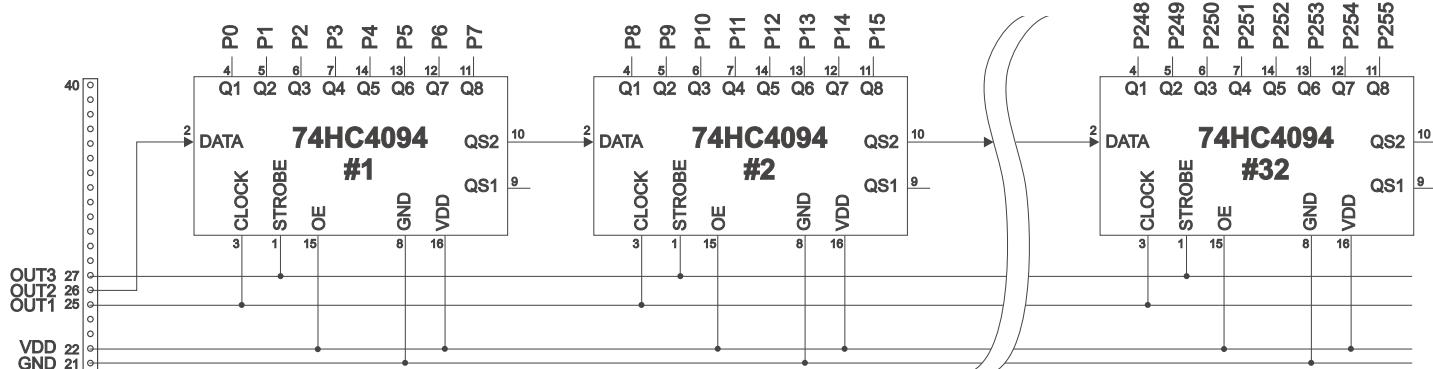
In addition to that every port change may start an individual port - or bit- macro (see p. 28). The command "ESC Y A 1" activates automatic port query. Every alteration of inputs firstly calls bit macros and afterwards port macros. If there is no defined macro, the new status is transferred into the send buffer (refer to p. 19).

Note: The logic circuitry is designed for slow operations; in other words, more than 3 changes per second cannot be easily executed.



Extended outputs (V1.4 and higher)

It is possible to connect 1 to 32 chips like 74HC4094 to the eDIP (OUT1...OUT3), this is why it is attainable to have 8 to 256 additional outputs. The command "ESCY E n1 n2 n3" (see p. 17) provides a comfortable way to control the outputs.



EXTERNAL KEYBOARD

A keyboard (anything from individual keys to a 8x8 matrix keyboard) can be connected to the I/O-ports. The command 'ESC Y M n1 n2 n3' defines the count of input lines ($n1=1..8$) and output lines ($n2=1..8$). $n3$ set debounce function with 10ms steps ($n3=0..15$). Please note that count of digital input and output lines will be reduced while connecting an external keyboard at the same port. Each key is connected with 1 output and 1 input. All inputs are terminated with a $20..50\text{k}\Omega$ pull-up resistor. For double-keystroke function decoupling of outputs is necessary. For that please use schottky diodes (e.g. BAT 46).

Transmitting the keystrokes

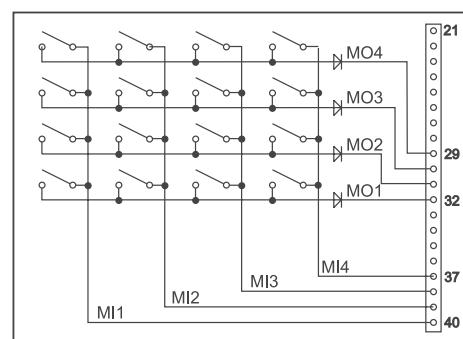
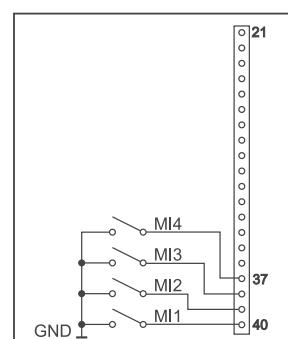
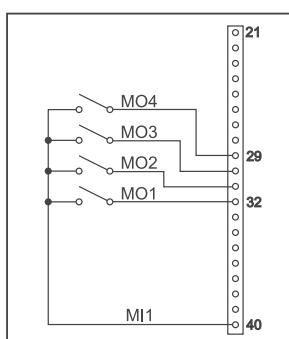
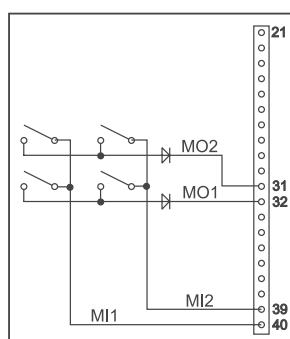
At each keystroke, the associated key number (1..64) is transmitted or - if a corresponding Matrix-Macro is defined, Matrix-Macro will be started. The release of the key is not transmitted. If the release of the key is to be transmitted as well, this can be done by defining Matrix-Macro no. 0. (see page 19: Responses of EA eDIPTFT43-A)

Calculation of key numbers:

Key_number = (output-1) * count_of_inputs + input (output = MO_x, input = MI_x).

Examples

- 2x2 matrix: Command 'ESC Y M 2 2 ..' defines the 2x2 matrix. Keypad will need input lines MI1, MI2 and output lines MO1, MO2. Output lines are decoupled by diodes; this is for double keystrokes necessary. 6 input and 6 output lines remain free for other requirements.
- 1x4 matrix: Command 'ESC Y M 1 4 ..' defines the 1x4 matrix. Keypad will need output lines MO1..MO4 and a single input line MI1. With that connection 7 input and 4 output lines remain free for other requirements.
- 4x0 matrix: Using one single output only (physically 4x1 Matrix), all keys can switch to GND. So no output line is necessary and command 'ESC Y M 4 0 ..' defines 4 input lines only. With that connection 4 input and 8 output lines remain free for other requirements.
- 4x4 matrix: Command 'ESC Y M 4 4 ..' defines the 4x4 matrix. Keypad will need input lines MI1..MI4 and output lines MO1..MO4. Output lines are decoupled by diodes; this is for double keystrokes necessary. 4 input and 4 output lines remain free for other requirements.



2x2 Matrix

1x4 Matrix

4x0 Matrix

4x4 Matrix

DATA TRANSFER PROTOCOL (SMALL PROTOCOL)

The protocol has an identical structure for all 3 interface types: RS-232, SPI and I²C. Each data transfer is embedded in a fixed frame with a checksum (protocol package). The EA eDIPTFT43-A acknowledges this package with the character <ACK> (=06) on successful receipt or <NAK> (=15) in the event of an incorrect checksum or receive buffer overflow. In the case of <NAK>, the entire package is rejected and must be sent again.

Receiving the <ACK> byte means only that the protocol package is ok, there is no syntax check for the command.

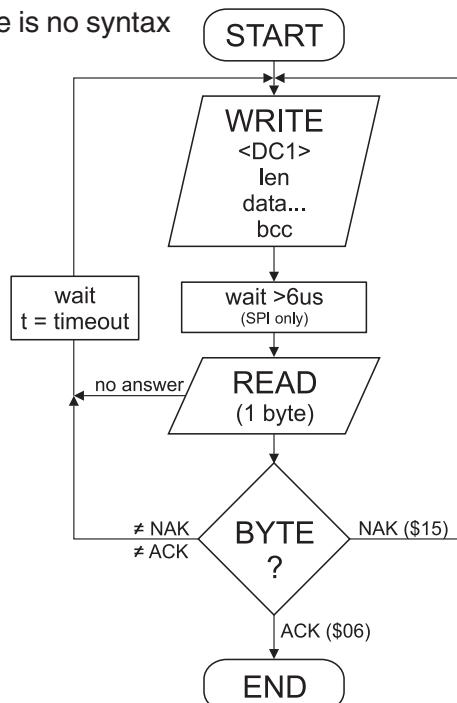
Note: it is necessary to read the <ACK> byte in any case.

If the host computer does not receive an acknowledgment, at least one byte is lost. In this case, the set timeout has to elapse before the package is sent again.

The raw data volume per package is limited to 255 bytes (len <= 255). Commands longer than 255 bytes (e.g. Load image ESC UL...) must be divided up between a number of packages. All data in the packages are compiled again after being correctly received by the EA eDIPTFT43-A.

Deactivating the small PROTOCOL

For tests the protocol can be switched off with an L level at pin 17 = DPROT. In normal operation, however, you are urgently advised to activate the protocol. If you do not, any overflow of the receive buffer will not be detected.



Building the small protocol packages

Command/Data to the display

>	<DC1>	len	data...	bcc
<	<ACK>			

<DC1> = 17(dec.) = \$11 <ACK> = 6(dec.) = \$06

len = count of user data (without <DC1>, without checksum bcc)

bcc = 1 byte = sum of all bytes incl. <DC1> and len, modulo 256

The user data is transferred framed by <DC1>, the number of bytes (len) and the checksum (bcc). The display responds with <ACK>.

Clear display and draw a line from 0,0 to 479,271															
>	<DC1>	len	ESC	D	L	ESC	G	D	0	0	479	271	bcc		
\$11	\$0E	\$1B	\$44	\$4C	\$1B	\$47	\$44	\$00	\$00	\$00	\$DF	\$01	\$0F	\$01	\$60

Example to a complete data package

```

void SendData(unsigned char *buf, unsigned char len)
{
    unsigned char i, bcc;
    SendByte(0x11);           // Send DC1
    bcc = 0x11;

    SendByte(len);           // Send data length
    bcc = bcc + len;

    for(i=0; i < len; i++)   // Send buf
    {
        SendByte(buf[i]);
        bcc = bcc + buf[i];
    }

    SendByte(bcc);           // Send checksum
}                           // C source code to transmit a data package

```

Request for content of send buffer

>	<DC2>	1	S	bcc	
<	<ACK>				
<	<DC1>	len	data...	bcc	

<DC2> = 18(dec.) = \$12 1 = 1(dec.) = \$01 S = 83(dec.) = \$53

<ACK> = 6(dec.) = \$06

len = count of user data (without <DC2>, without checksum bcc)

bcc = 1 byte = sum of all bytes incl. <DC2>, modulo 256

The command sequence <DC2>, 1, S, bcc empties the display's send buffer. The display replies with the acknowledgement <ACK> and begins to send all the collected data such as touch keystrokes.

EA eDIPTFT43-A

Request for buffer information

>	<DC2>	1	I	bcc	
<	<ACK>				
<	<DC2>	2	send buffer bytes ready	receive buffer bytes free	bcc

<DC2> = 18(dec.) = \$12 I = 1(dez.) = \$01 I = 73(dez.) = \$49

<ACK> = 6(dec.) = \$06

send buffer bytes ready = count of bytes stored in send buffer

receive buffer bytes free = count of bytes for free receive buffer

bcc = 1 byte = sum of all bytes incl. <DC2>, modulo 256

Protocol settings

>	<DC2>	3	D	packet size for send buffer	timeout	bcc
<	<ACK>					

<DC2> = 18(dec.) = \$12 3 = 3(dez.) = \$03 D = 68(dez.) = \$44

packet size for send buffer = 1..128 (standard: 128)

timeout = 1..255 in 1/100 seconds (standard: 200 = 2 seconds)

bcc = 1 byte = sum of all bytes incl. <DC2>, modulo 256

<ACK> = 6(dec.) = \$06

This command queries whether user data is ready to be picked up and how full the display's receive buffer is.

This is how the maximum package size that can be sent by the display can be limited. The default setting is a package size with up to 128 bytes of user data.

The timeout can be set in increments of 1/100 seconds. The timeout is activated when individual bytes get lost. The entire package then has to be sent again.

Request for protocol settings

>	<DC2>	1	P	bcc	
<	<ACK>				
<	<DC2>	3	max. packet size	akt. send packet size	akt. timeout

This command is used to query protocol settings.

<DC2> = 18(dec.) = \$12 I = 1(dez.) = \$01 P = 80(dez.) = \$50

<ACK> = 6(dec.) = \$06

max. packet size = count of maximum user data for 1 package (eDIPTFT43-A = 255)

akt. send packet size = current package size for send

akt. timeout = current timeout in 1/100 seconds

bcc = 1 byte = sum of all bytes incl. <DC2>, modulo 256

Repeat the last package

>	<DC2>	1	R	bcc	
<	<ACK>				
<	<DC1>	len	data...		bcc

If the most recently requested package contains an incorrect checksum, the entire package can be requested again. The reply can then be the contents of the send buffer (<DC1>) or the buffer/protocol information (<DC2>).

<DC2> = 18(dec.) = \$12 I = 1(dez.) = \$01 R = 82(dez.) = \$52

<ACK> = 6(dec.) = \$06

<DC1> = 17(dec.) = \$11

len = count of user data in byte (without checksum, without <DC1> or <DC2>)

bcc = 1 byte = sum of all bytes incl. <DC2> and len, modulo 256

Addressing (only for RS232/RS485)

>	<DC2>	3	A	select or deselect	adr	bcc
<	<ACK>					

This command can be used to select or deselect the eDIP with the address adr.

<DC2> = 18(dec.) = \$12 3 = 3(dez.) = \$03 A = 65(dez.) = \$41

select or deselect: 'S' = \$53 or 'D' = \$44

adr = 0..255

bcc = 1 byte = sum of all bytes incl. <DC2> and adr, modulo 256

<ACK> = 6(dec.) = \$06

TERMINAL MODE

When you switch the unit on, the cursor flashes in the first line, indicating that the display is ready for operation. All the incoming characters are displayed in ASCII format on the terminal (exception: CR,LF,FF,ESC,'#'). The prerequisite for this is a working protocol frame or a deactivated protocol (see pages 10 and 11).

Line breaks are automatic or can be executed by means of the 'LF' character. If the last line is full, the contents of the terminal scroll upward. The 'FF' character (page feed) deletes the terminal.

The character '#' is used as an escape character and thus cannot be displayed directly on the terminal. If the character '#' is to be output on the terminal, it must be transmitted twice: '##'.

The size of the terminal-window can be set by command 'ESC TW'.

Attention: Graphic commands are able to draw inside terminal window. For example 'ESC DL' will delete terminal window, too.

+ Lower Upper	\$0 (0)	\$1 (1)	\$2 (2)	\$3 (3)	\$4 (4)	\$5 (5)	\$6 (6)	\$7 (7)	\$8 (8)	\$9 (9)	\$A (10)	\$B (11)	\$C (12)	\$D (13)	\$E (14)	\$F (15)
\$00 (dez: 0)	ₙₗ	↑	↓	❖	❖	✓	♪	▬	+	↔	ₗₑ	↓	ₖₑ	ₖₑ	ₖₑ	ₖₑ
\$10 (dez: 16)	₀	₁	₂	₃	₄	₅	₆	₇	₈	₉	ₜ	ₚ	ₘ	₞	ₚ	ₘ
\$20 (dez: 32)	!	"	#	\$	%	&	'	()	*	+	,	-	.	/	
\$30 (dez: 48)	₀	₁	₂	₃	₄	₅	₆	₇	₈	₉	:	;	<	=	>	?
\$40 (dez: 64)	ₜ₏	₏ₐ	₏₊	₏₋	₏₃	₏₄	₏₅	₏₆	₏₇	₏₈	₏₉	₏ₑ	₏ₒ	₏ₓ	₏ₔ	₏ₕ
\$50 (dez: 80)	ₚ₏	₏ₐ	₏₊	₏₋	₏₃	₏₄	₏₅	₏₆	₏₇	₏₈	₏₉	₏ₑ	₏ₒ	₏ₓ	₏ₔ	₏ₕ
\$60 (dez: 96)	`	₏₏	₏ₐ	₏₊	₏₋	₏₃	₏₄	₏₅	₏₆	₏₇	₏₈	₏₉	₏ₑ	₏ₒ	₏ₓ	₏ₔ
\$70 (dez: 112)	₏₏	₏ₐ	₏₊	₏₋	₏₃	₏₄	₏₅	₏₆	₏₇	₏₈	₏₉	₏ₑ	₏ₒ	₏ₓ	₏ₔ	₏ₕ
\$80 (dez: 128)	ç	ü	é	â	ä	à	å	ç	ê	ë	è	í	î	ì	ñ	ñ
\$90 (dez: 144)	É	æ	Æ	ô	ö	ò	û	ý	ö	ü	¢	£	¥	®	ƒ	ƒ
\$A0 (dez: 160)	á	í	ó	ú	ñ	â	ô	é	ê	ë	è	í	î	ì	»	»
\$B0 (dez: 176)
\$C0 (dez: 192)	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ
\$D0 (dez: 208)	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ
\$E0 (dez: 224)	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ
\$F0 (dez: 240)	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ	ₗ

Terminal-Font 2: 8x16

USING THE SERIAL INTERFACE

The operating unit can be programmed by means of various integrated commands. Each command begins with ESCAPE followed by one or two command letters and then parameters.

There are two ways to transmit commands:

1. ASCII mode

- The ESC character corresponds to the character '#' (hex: \$23, dec: 35).
- The command letters come directly after the '#' character.
- The parameters are transmitted as plain text (several ASCII characters) followed by a separating character (such as a comma ',') - also after the last parameter e.g.: #GD0,0,479,271,
- Strings (text) are written directly without quotation marks and concluded with CR (hex: \$0D) or LF (hex: \$0A).

2. Binary mode

- The escape character corresponds to the character ESC (hex: \$1B, dec: 27).
- The command letters are transmitted directly.
- The coordinates xx and yy are transmitted as 16-bit binary values (first the LOW byte and then the HIGH byte).
- All the other parameters are transmitted as 8-bit binary values (1 byte).
- Strings (text) are concluded with CR (hex: \$0D) or LF (hex: \$0A) or NUL (hex: \$00).

No separating characters, such as spaces or commas, may be used in binary mode.

The commands require **no final byte**, such as a carriage return (apart from the string \$00).

ALL COMMANDS AT A GLANCE

The built-in intelligence allows an easy creation of your individual screen content. Below mentioned commands can be used either directly via the serial interface (see page 12) or together with the self-definable macro (see pages 26/28).

EA eDIPTFT43-A: Terminal commands											after reset		
Command	Codes					Remarks					8,1		
Set terminal color	ESC	F	T	fg	bg	Preset color for terminal mode: fg=foreground color; bg=background color					8,1		
Define window	ESC	T	W	n1	C	L	W	H	The terminal output is executed with font n1: 1=8x8; 2=8x16 only within the window from column and line L (=upper-left corner) with a width of W and a height of H (specifications in characters). Display organisation 480x272: C=1..60; L=1..34/17; 272x480: C=1..34; L=1..60/30		C8x16 1,1 60,17		
Form feed FF (dec:12)	^L					The contents of the screen are deleted and the cursor is placed at pos. (1,1)							
Carriage return CR (13)	^M					Cursor to the beginning of the line on the extreme left							
Line feed LF (dec:10)	^J					Cursor 1 line lower, if cursor in last line then scroll							
Position cursor	ESC	T	P	C	n1	L	C=column; L=line; origin upper-left corner (1,1) n1=0: Cursor is invisible; n1=1: Cursor flashes;				1,1 1		
Cursor on/off			C	S			The current cursor position is saved						
Save cursor position	ESC	T	R				The last saved cursor position is restored						
Restore cursor position			A				Terminal display is switched off; outputs are rejected						
Terminal off	ESC	T	E				Terminal display is switched on;				on		
Terminal on			V				The version no. is output in the terminal e.g. "EA eDIPTFT43-A V1.0 Rev.A"						
Output version	ESC	T	J				The macrofile-projectname is output in the terminal e.g. "init / delivery state"						
Output projectname			Q				The used interface is output in the terminal e.g. "RS232,115200 baud,ADR: \$07"						
Output interface	ESC	T	I				The terminal is initialized and cleared; the software version, hardware revision, macrofile-projectname and CRC-checksum are output in the terminal						

EA eDIPTFT43-A: Graphic commands											after reset		
Command	Codes					Remarks							
Display commands (effect on the entire display)													
Set display color	ESC	F	D	fg	bg	Defines color 1..32 (0=transparent) for display and areas: fg=foreground color; bg=background color					8,1		
Set display orientation	ESC	D	O	n1		n1=0: 0°; n1=1: 90°; n1=2: 180°; n1=3: 270° (0°+180°=480x272; 90°+270°=272x480)					0°		
Delete display			L			Delete display contents (all pixels to background color)							
Fill display	ESC	D	S			Fill display contents (all pixels to foreground color)							
Fill display with color			F	n1		Fill complete display content with color n1=1..32							
Invert display	ESC	Z	I			Invert display content							
Commands for outputting strings													
Set text color	ESC	F	Z	fg	bg	Color 1..32 (0=transparent) for string and character: fg=text color; bg=background color					8,0		
Set font	ESC	Z	F	n1		Set font with the number n1					3		
Font zoom factor			Z	n1	n2	n1 = X-zoom factor (1x to 8x); n2 = Y-zoom factor (1x to 8x)					1,1		
Additional width/height	ESC	Z	Y	n1	n2	n1=0..15: additional width left/right; n2=0..15: additional height top/bottom					0, 0		
Spacewidth			J	n1		n1=0: use spacewidth from font; n1=1: same width as a number; n1>2 width in dot					0		
Text angle	ESC	Z	W	n1		Text output angle: n1=0: 0°; n1=1: 90°; n1=2: 180°; n1=3: 270°					0		
Output string			L			A string (...) is output to xx1,yy1 end of string: 'NUL' (\$00), 'LF' (\$0A) or 'CR' (\$0D) several lines are separated by the character ' ' (\$7C, pipe) the character '\\' (\$5C, backslash) cancels the special function of ' ' and '\'							
L: left justified	ESC	Z	C	xx1	yy1	text	...	NUL					
C: centered			R										
R: right justified	ESC	Z	B	xx1	yy1	xx2	yy2	n1	text	...	NUL		
Output string in an area (since V1.2)						Output a string (...) inside area from xx1,yy1 to xx2,yy2 at position n1=1..9; the area will be filled with background color; n1=1: Top Left; n1=2: Top Center; n1=3: Top Right n1=4: Middle Left; n1=5: Middle Center; n1=6: Middle Right n1=7: Bottom Left; n1=8: Bottom Center; n1=9: Bottom Right							
String for terminal	ESC	Z	T			Command for outputting a string from a macro to the terminal							
Draw straight lines and points													
Set color for lines	ESC	F	G	fg	bg	Colors 1..32 (0=transparent): fg = color for line; bg = pattern background					8,1		
Draw rectangle	ESC	G	R	xx1	yy1	xx2	yy2						
Draw straight line			D	xx1	yy1	xx2	yy2						
Continue straight line	ESC	G	W	xx1	yy1								
Draw point			P	xx1	yy1								
Point size/line thickness	ESC	G	Z	n1	n2	n1 = X-point size (1 to 15); n2 = Y-point size (1 to 15); Set straight line/point pattern no. n1=1..255; 0=do not use pattern					1,1 0		
Pattern			M	n1									
Set start point	ESC	G	S	xx1	yy1								
Draw X-Graph			X	xs	ya	yy1							
Draw Y-Graph	ESC	G	Y	ys	xa	xx1							
Change/draw rectangular areas													
Delete area	ESC	R	L	xx1	yy1	xx2	yy2						
Fill area			S	xx1	yy1	xx2	yy2	n1					
Fill area with color	ESC	R	F	xx1	yy1	xx2	yy2	n1					
Invert area			I	xx1	yy1	xx2	yy2						
Copy area	ESC	R	C	xx1	yy1	xx2	yy2	xx3	yy3	Copy an area from xx1,yy1 to xx2,yy2 to new position xx3,yy3			
Patterncolor			F	M	fg	bg				Color 1..32 (0=transp.) for monochrome pattern: fg=foreground; bg=background color			
Area with fill pattern	ESC	R	M	xx1	yy1	xx2	yy2	n1					
Draw box			O	xx1	yy1	xx2	yy2	n1					
Set color for border	ESC	R	F	R	c1	c2	c3			Set color for border segments: c1=frame outside; c2=frame inside; c3=filling		8,1,1	
Set border type			E	n1	n2					Set border type n1=1..255; border angle: n2=0: 0°; n1=1: 90°; n1=2: 180°; n1=3: 270°			
Draw border box	ESC	R	R	xx1	yy1	xx2	yy2			Draw a border box from xx1,yy1 to xx2,yy2		1, 0	

EA eDIPTFT43-A: Bitmap / Animation commands								after reset	
Command	Codes				Remarks				
Bitmap image commands									
Set bitmap colors	ESC	F	U	fg	bg			painting color for monochrome bitmaps fg=foreground color; bg=background color	
Image zoom factor			Z	n1	n2			n1 = X-zoom factor (1x to 8x); n2 = Y-zoom factor (1x to 8x)	
Image angle			W	n1				output angle of the image: n1=0: 0°; n1=1: 90°; n1=2: 180°; n1=3: 270°	
Mirror Image			X	n1				n1=0: normal display; n1=1: the image is mirrored horizontally	
Transparency for color bitmaps			T	n1				n1=0: no transparency; show picture with all colors rectangular n1=1: color of the first dot at top left side will be defined as transparent (like a mask) n1=2: if defined - use transparent color from bitmap-file (.GIF .TGA .G16) n1=3: replace transparent color from bitmap-file with actually background color	
Load internal image	ESC	U	I	xx1	yy1	nr		Load internal image with the no (0 to 255) from the data flash memory to xx1,yy1	
Load image			L	xx1	yy1	G16 data ...		Load an image to xx1,yy1; see image structure (G16 format) for image data	
RLE compression	ESC	U	R					the next hardcopy ('ESC U H xx1,yy1,xx2,yy2) is send with RLE compression (since V1.5)	
Send hardcopy	ESC	U	H	xx1	yy1	xx2	yy2	After this command, the image extract is sent (to sendbuffer) in G16 format	
Animation image commands									
Set animation colors	ESC	F	W	fg	bg			color for monochrome animation images fg=foreground color; bg=background color	
Animation zoom factor			Z	n1	n2			n1 = X-zoom factor (1x to 8x); n2 = Y-zoom factor (1x to 8x)	
Animation angle			W	n1				output angle of the animation image n1=0: 0°; n1=1: 90°; n1=2: 180°; n1=3: 270°	
Mirror animation			X	n1				n1=0: normal display; n1=1: the animation image is mirrored horizontally	
Transparency for color animation			T	n1				n1=0: no transparency; show animation with all colors rectangular n1=1: color of the first dot at top left side will be defined as transparent (like a mask) n1=2: if defined - use transparent color from animation-file (.GIF .G16) n1=3: replace transparent color from animation-file with actually background color	
Load single image	ESC	W	I	xx1	yy1	n1	n2	Load from animation n1=0..255 the single image n2 to xx1,yy1	
Define animationprocess	ESC	W	D	no	xx1	yy1	n2	Define an animationprocess no=1..4 at position xx1,yy1 (=left top edge) with animation image n2=0..255. type: 1=run once; 2=cyclically; 3=pingpong; 4=once backwards; 5=cyclic backwards 6=pingpong backwards; 7=manually (use command ESC W N P F M) time: 0=stop; 1..254=time in 1/10 sec; 255=use time from animation-file	
Change animation type			Y	no	type			Assign a new type=1..7 to animationprocess no=1..4	
Change animation time			C	no	time			Assign a new time=0..255 to animationprocess no=1..4	
Next animation image			N	no				Show the next image from animationprocess no=1..4	
Previous animation image			P	no				Show the previous image from animationprocess no=1..4	
Show animation image			F	no	n2			Show image n2 from animationprocess no=1..4	
Run to animation image			M	no	n2			Run animationprocess no=1..4 from actually image to image n2	
Stop animationprocess			L	no				Stop animationprocess no=1..4 and clear last image with actually background color	

EA eDIPTFT43-A: Bargraph commands													after reset
Command	Codes					Remarks							
Bargraph commands													
Set color for bargraph	ESC	F	B	fg	bg	fc							Colors: fg = foreground; bg = background; fc = color for frame
Bargraph pattern			M	n1	Pattern for bargraph n1=1..255; n1=0 no pattern/solid (valid for type=0..3)								8,1,8
Bargraph border	ESC	B	E	n1	Border for bargraph n1=1..255 (valid for type=4..7)								0
Bargraph linewidth			B	n1	Linewidth for bargraph n1=1..255; n1=0 automatic (valid for type=2,3,6,7)								1
			R		Define bargraph no=1..20 to L(left), R(ight), O(up), U(down) xx1,yy1,xx2,yy2 rectangle enclosing the bar graph.								0
Define bargraph	ESC	B	L	no	xx1	yy1	xx2	yy2	sv	ev	type		sv, ev are the values for 0% and 100%. type: 0=pattern bar; 1=pattern bar in rectangle; type: 2=pattern line; 3=pattern line in rectangle; type: 4=border bar; 5=border bar in rectangle; type: 6=border line; 7=border line in rectangle;
			O										no bar defined
Update bargraph			A	no	val								Set and draw the bargraph with the number no=1..20 to the new value val
Draw bargraph	ESC	B	N	no									Entirely redraw the bargraph with the number no=1..20
Send bargraph value			S	no									Send the current value of bargraph number no=1..20 to sendbuffer
Delete bargraph	ESC	B	D	no	n2								The definition of the bar graph with the number no=1..20 becomes invalid. If the bar graph was defined as input with touch, this touch field will also be deleted. n2=0: Bar graph remains visible; n2=1: Bar graph is deleted
User values - Format text output													
User value color	ESC	F	X	fg	bg								Set color for bargraph user value; fg=foreground, bg=background color
User value font			F	n1									Set font n1 for bargraph user value
User value zoom			Z	n1	n2								Set zoom factor for bargraph user value; n1=X-Zoom 1x..8x; n2=Y-Zoom 1x..8x
User value additional width/height	ESC	B	Y	n1	n2								n1=0..15: additional width left/right; n2=0..15: additional height top/bottom for bargraph user value;
User value angle			W	n1									Set writing angle for bargraph user value; n1=0: 0°; n1=1: 90°; n1=2: 180°; n1=3: 270°;
User values / scaling	ESC	B	X	no	xx1	yy1	For mat Str ing	NUL					Define user value for bargraph no=1..20. Output is always right justified to xx1,yy1; Format String: "bv1=uservalue1;bv2=uservalue2". NUL' (\$00) = termination; Assign two bar values (bv1,bv2 =0..254) to user defined values max. range: 4 1/2 digits 19999 + decimal point ('.' oder ',') + sign e.g. display "-123.4" for bar value bv1=0 and "567.8" for bar value bv2=100 Format String: "0=-123.4;100=567.8"

EA eDIPTFT43-A: Instrument commands (since V1.4)													after reset
Command	Codes					Remarks							
Define, use instruments													
Definine instrument	ESC	I	P	n1	xx1	yy1	n2	n3	aw	ew			Define instrument n1=1..4 an xx1,yy1 (=left top edge); Use intrument image n2=0..255 Output angle n3=0: 0°; n3=1: 90°; n3=2: 180°; n3=3: 270°; aw, ew (0..254) are start and endvalue (0% and 100%).
Update instrument			A	n1	val								nothing defined
Redraw instrument	ESC	I	N	n1									Update instrument with new value and redraw
Send instrument value			S	n1									Redraw entirely instument n1=1..4
Delete instrument	ESC	I	D	n1	n2								Send actual instrument value n1=1..4 to send buffer
													The definition of the instruments gets invalid. If the instrument was adjustable by touch, the touch area will be deleted, too.
													n2=0: Instrument stays visible; n2=1: Instrument is deleted completely
User values - formatted string output													
User value color	ESC	F	I	vf	hf								Set color 1..32 for instrument user value fg=foreground; bg=background color
User value font			F	n1									Set font nr for instrument user value
User value zoom			Z	n1	n2								Set zoom factor for instrument user value: n1=X-Zoom 1x..8x; n2=Y-Zoom 1x..8x
User value additional height/width	ESC	I	Y	n1	n2								n1=0..15: additional width left/right; n2=0..15: additional height top/bottom for instrument user value;
User value angle			W	n1									Set writing angle for instrument user value: n1=0: 0°; n1=1: 90°; n1=2: 180°; n1=3: 270°
User values / scaling	ESC	I	X	n1	xx1	yy1	For mat Str ing	NUL					Define user value for instrument no=1..4. Output is right justified to x1,y1 Format String: "iv1=uservalue1;iv2=uservalue2" Assign two instrument values (iv1,iv2 =0..254) to user defined values max. range: 4 1/2 digits 19999 + decimal point ('.' oder ',') + sign e.g. display "-123.4" for iv1=0 and "567.8" for iv2=100 Format String: "0=-123.4;100=567.8"

EA eDIPTFT43-A: Analogue input AIN1, AIN2 commands											after reset	
Command	Codes				Remarks							
Commands for analogue inputs												
Calibration	ESC	V	@	ch	xx1						not calib rated	
Enable/disable AIN scan			A	n1							0	
Send analog value			D	ch								
Limit for analog macro			K	ch	n1	n2	n3				0	
Redefine analogmacro (since V1.1)	ESC	V	M	n1	n2							
Bargraph for AIN1/AIN2			B	ch	no							
Instrument for AIN1/AIN2 (since V1.4)			+	ch	no							
Redraw bargraph			R	ch								
User values - Format text output												
User value color		F	V	ch	fg	bg					8,1	
User value Font			F	ch	n1						5	
User value zoom			Z	ch	n1	n2					1,1	
User value additional width/height			Y	ch	n1	n2					0, 0	
User value angle			W	ch	n1						0	
User values / scaling	ESC	V	E	ch	format string ...		NUL				0 =0.00 5000 =5.00	
Send user value		S	ch									
Display on terminal	ESC	V	T	ch	xx1	yy1						
Display user value		G	ch									

EA eDIPTFT43-A: Macro commands											after reset	
Command	Codes				Remarks							
Macro commands												
Run macro			N	no								
Run touch macros			T	no								
Run port macro			P	no								
Run bit macro			B	no								
Run matrix macro			X	no								
Run process macro			C	no								
Run analogue macro			V	no								
Disable macros	ESC	M	L	type	n1	n2						
Enable macros			U	type	n1	n2						
Select macro/image page	ESC	M	K	n1								
Save macro/image page			W									
Restore macro/imagepage			R									
Automatic (normal-) macro												
Macro with delay			G	n1	n2							
Autom. macros once only			E	n1	n2	n3						
Autom. macros cyclical			A	n1	n2	n3						
Autom. macros ping pong			J	n1	n2	n3						
Macro processes												
Define macro process			D	no	type	n3	n4	zs				
Macro process interval			Z	no	zs							
Stop macro processes			S	n1							1	

EA eDIPTFT43-A: Commands for backlight, I/O-port and misc										after reset
Command	Codes			Remarks						
Backlight commands										
Illumination brightness	ESC	Y	H	n1						Set brightness of the LED illumination n1=0 to 100%.
Increase brightness			N							Increase brightness of the LED illumination (one step=1%)
Decrease brightness			P							Decrease brightness of the LED illumination (one step=1%)
Brightness changetime			Z	n1						Time n1=0..31 in 1/10sec for changing brightness from 0 to 100%
Illumination on/off			L	n1						LED n1=0: OFF; n1=1: ON; n1=2 to 255: LED switched ON for n1/10sec
Assign bar with backlight			B	no						Assign bar no=1..20 for changing brightness of the backlight
Assign instrument with backlight (since V1.4)			+	n1						Brightness is connected to instrument 1..4.
Save parameter			@							Save the actual brightness and changetime for poweron to EEPROM
			Port commands							
Write output port	ESC	Y	W	n1	n2					n1=0: Set all 8 output ports in accordance with n2 (=8-bit binary value) n1=1..8: Reset output port n1 (n2=0); set (n2=1); invert (n2=2)
Read input port			R	n1						n1=0: Read all 8 input ports as 8-bit binary value (to sendbuffer) n1=1..8: Read input port <n1> (1=H-level=5V, 0=L-level=0V)
Port scan on/off			A	n1						The automatic scan of the input port is n1=0: deactivated; n1=1: activated
Invert input port			I	n1						The input port is n1=0: normal; n1=1: evaluated inverted
Matrix keyboard			M	n1	n2	n3				Specifies an external matrix keyboard at the inputs and outputs. n1=number of inputs (1..8); n2=number of outputs (0..8); n3=debouncing (0..7)
Redefine input bitmacro (since V1.1)	ESC	Y	D	n1	n2	n3				input port n1=1..8 is assigned by falling edge n2=0 to new BitMacro number n3=0..255 input port n1=1..8 is assigned by rising edge n2=1 to new BitMacro number n3=0..255
Redefine matrixmacro for keys (since V1.1)			X	n1	n2					Assign keynumber n1=1..65 with matrixmacro number n2=0..255 After release the key n1=0 run matrixmacro number n2=0..255
Extended outputs (since V1.4)	ESC	Y	E	n1	n2	n3				Set the outputs of the external 74HC4094 (refer to page 8) from port n1=0..255 to port n2=0..255; n3=0: low; n3=1: high; n3=2: invert;
Other commands										
Define color	ESC	F	P	no	R5	G6	B5			Set a new RGB value for color no. n1=1..32 (R5:Bit7..3; G6:Bit7..2; B5:Bit7..3)
Wait (pause)	ESC	X	n1	adr						Wait n1 tenths of a second before the next command is executed.
Set RS485 address	ESC	K	A							For RS232/RS485 operation only and only possible when Hardware address is 0. The eDIP is assigned a new address adr (in the Power-On macro).
String table code	ESC	S	T	n1						n1=0: no use of internal strings
Tone on/off	ESC	Y	S	n1						n1>0: after code n1 appears following codes are internal string numbers (since V1.5) The tone output (pin 16) becomes n1=0:OFF; n1=1:ON; n1=2 to 255:ON for n1/10s
Send bytes	ESC	S	B	num	data...					num (=1 to 255) bytes are sent to the sendbuffer data... = num Bytes. In the source text of the macro programming, the number num must not be specified. This is counted by the ediptftcompiler and entered.
Send version			V							The version is sent as a string to sendbuffer e.g. "EA eDIPTFT43-A V1.0 Rev.A TP+"
Send projectname			J							The macro-projectname is sent as a string to the sendbuffer e.g. "init / delivery state"
Send internal infos			I							Internal information about the edip is sent to the sendbuffer.

TOUCH PANEL

The Version EA eDIPTFT43-ATP is shipped with an analog, resistive touch panel, the version EA eDIPTFT43-ATC with a capacitive touch panel. Up to 60 touch areas (keys, switches, menus, bar graph inputs) can be defined simultaneously. The fields can be defined with pixel accuracy. The display supports user-friendly commands. When the touch "keys" are touched, they can be automatically inverted and an external tone can sound (pin 16), indicating they have been touched. The predefined return code of the "key" is transmitted via the interface, or an internal touch macro with the number of the return code is started instead.

EA eDIPTFT43-A: Commands for the touch panel													after reset		
Command	Codes				Remarks										
Touch presets															
Touch bordercolors	ESC	F	E	n1	n2	n3	s1	s2	s3	Set the colors (0..32) for touch borders (ESC AT AK). n=normal; s=selected; 1=frame outside; 2=frame inside; 3=filling				8,1,2 8,1,7	
Touch borderform		A	E	n1	n2					n1=1..255 border number; n1=0 no border; n2=angle 0=0°; 1=90°; 2=180°; 3=270°				1,0	
Touch button colors	ESC	F	C	nf	nb	sf	sb			Set the colors (0..32) for monochrome touch buttons (ESC AU AJ). n=normal; s=selected; f=foreground; b=background				8,1 8,1	
Touch button number		A	C	n1	n2	n3	n4			n1=0..255 button number; n2=button angle; n3=X-Zoom 1..8; n4=Y-Zoom 1..8				1,0,1,1	
Radio group for switches	ESC	A	R	n1						n1=0: newly defined switches do not belong to a group. n1=1 to 255: newly defined switches belong to the group with the number n1. Only 1 switch in a group is active at any one time; all the others are deactivated. In the case of a switch in a group, only the down code is applicable. the up code is ignored.				0	
Label font presets															
Font color	ESC	F	A	nf	sf					Color for touch labeling. nf=normal fontcolor; sf= fontcolor for selection				8,1	
Label font			F	n1						Set font with the number n1 for touch key label				5	
Label zoom factor			Z	n1	n2					n1 = X-zoom factor (1x to 8x); n2 = Y-zoom factor (1x to 8x)				1,1	
Additional width/height	ESC	A	Y	n1	n2					n1=0..15: additional width left/right; n2=0..15: additional height top/bottom				0,0	
Label angle			W	n1						Label output angle: n1=0: 0°; n1=1: 90°; n1=2: 180°; n1=3: 270°				0	
Offset for selected label			O	n1	n2					n1=X-offset; n2=Y-offset; n1,n2=0..7 (add +8 for negative direction)				0,0	
Define touch areas															
Define touch key	ESC	A	T	xx1	yy1	xx2	yy2	dow Cod	up Cod	text ...	NUL	'T': The area from xx1,yy1 to xx2,yy2 is defined as a key 'K': The area from xx1,yy1 to xx2,yy2 is defined as a switch 'U': The actual button is loaded to xx1,yy2 and defined as a key 'J': The actual button is loaded to xx1,yy2 and defined as a switch 'down code':(1-255) return/touchmacro when key pressed. 'up code': (1-255) return/touchmacro when key released. (down/up code = 0 press/release not reported). 'text': this is a string that is placed in the key with the current touch font. The first character determines the alignment of the text (C=centered, L=left justified, R=right justified). Multiline texts are separated with the character ' ' (\$7C, dec: 124); optional: after the character '~' (\$7E, dec: 126) you can write a 2nd text for a selected touch key/switch e.g. "LED on~LED off" 'nul': (\$00) = end of string			
Define touch switch (status of the switch toggles after each contact)	ESC	A	K	xx1	yy1	xx2	yy2	dow Cod	up Cod	text ...	NUL	(\$7C, dec: 124); optional: after the character '~' (\$7E, dec: 126) you can write a 2nd text for a selected touch key/switch e.g. "LED on~LED off"			
Define drawing area	ESC	A	D	xx1	yy1	xx2	yy2	n1	fg	A drawing area is defined. You can then draw with a line width of n1 and color fg within the corner coordinates xx1,yy1 and xx2,yy2.					
Define free touch area	ESC	A	H	xx1	yy1	xx2	yy2			A freely usable touch area is defined. Touch actions (down, up and drag) within the corner coordinates xx1,yy1 and xx2,yy2 are sent.					
Set bar by touch	ESC	A	B	n1						The bargraph with number n1 is defined for input by touch panel.					
Set instrument by touch (since V1.4)	ESC	A	+	n1						The instrument with number n1 is defined for input by touch panel.					
Global settings															
Touch query on/off	ESC	A	A	n1						Touch query is deactivated (n1=0) or activated (n1=1)				1	
Touch key response	ESC	A	I	n1						Automatic inversion when touch key touched: n1=0=OFF; n1=1=ON;				1	
			S	n1						Tone sounds briefly when a touch key is touched: n1=0=OFF; n1=1=ON				1	
Send bar value on/off	ESC	A	Q	n1						Automatic transmission of a new bar graph / instrument value by touch input is n1=0: deactivated; n1=1: is placed in the sendbuffer once at the end of input n1=2: changes are placed continuous in the sendbuffer during input				1	
Other functions															
Invert touch key			N	code						The touch key with the assigned return code is inverted manually					
Set touch switch	ESC	A	P	code	n1					The status of the switch is changed by means of a command (n1=0=off; n1=1=on)					
Query touch switch			X	code						The status of the switch with the return code (off=0; on=1) is placed in the sendbuffer					
Query radio group			G	n1						down code of the activated switch from the radio group n1 is placed in the sendbuffer					
Delete touch area	ESC	A	L	code	n1					The touch area with the return code (code=0: all touch areas) is removed from the touch query. n1=0 the area remains visible on the display; n1=1, the area is deleted.					
			V	xx1	yy1	n1				remove the Touch area that includes the coordinates xx1,yy1 from the touch query. n1=0: area remains visible; n1=1: Delete area					

EA eDIPTFT43-A

TOUCH ADJUSTMENT (ONLY VALID FOR RESESTIVE TOUCH EA EDIPTFT43-ATP)

The touch panel is perfectly adjusted and immediately ready for operation on delivery. As a result of aging and wear, it may become necessary to readjust the touch panel:

- 1a. Send Command 'ESC A@' or
 - 1b. Touch the touch panel at power-on and keep it depressed. After the message "touch adjustment ?" appears, release the touch panel. Touch the touch panel again within a second for at least a second.
 2. Follow the instructions for adjustment (press the 2 points upper left and lower right).

RESPONSE OF THE EA eDIPTFT43-A VIA SERIAL INTERFACE

The table below contains all response codes. Some response data will come automatically some others on request. In addition to that with command 'ESC SB ...' user is able to transmit individual data packages. All responses are placed into the sendbuffer. With the smallprotocol command 'Request for content of send buffer' (see page10) the host can read out the sendbuffer. This can be done per polling, alternatively pin 20 'SBUF' shows with LO-signal that data is ready to transmit.

PRELOADED FONTS

As standard, there are 3 monospaced, 3 proportional character sets and 2 large digit fonts integrated. The proportional character sets (which have a narrow "I" and a wide "W", for example) look better and take up less space on the screen. Each character can be placed with **pixel accuracy**, and its height and width can be increased by a factor of 1 to 8. A text can be output left justified, right justified or centered. Rotation in 90° steps is possible. Macro programming permits further fonts to be integrated. All kinds of fonts can be converted from True-Type Fonts by using the LCD toolkit/eDIPTFTcompiler (the USB Evaluation Board EA 9777-2USB is required).

¹ refer to web: <http://www.lcd-module.com/products/edip.html>

+ Lower Upper \	\$0 (0)	\$1 (1)	\$2 (2)	\$3 (3)	\$4 (4)	\$5 (5)	\$6 (6)	\$7 (7)	\$8 (8)	\$9 (9)	\$A (10)	\$B (11)	\$C (12)	\$D (13)	\$E (14)	\$F (15)
\$20 (dez: 32)	!	"	*	5	*	*	*	*	*	*	*	*	*	*	*	/
\$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
\$40 (dez: 64)	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	0
\$50 (dez: 80)	P	Q	R	S	T	U	V	W	X	Y	Z	[\	^	-	
\$60 (dez: 96)	~	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
\$70 (dez: 112)	p	q	r	s	t	u	v	w	x	y	z	{	}	^	-	
\$80 (dez: 128)	€	ü		ä												
\$90 (dez: 144)				ä				ü								

Font 1: 4x6 monospaced

+ Lower Upper \	\$0 (0)	\$1 (1)	\$2 (2)	\$3 (3)	\$4 (4)	\$5 (5)	\$6 (6)	\$7 (7)	\$8 (8)	\$9 (9)	\$A (10)	\$B (11)	\$C (12)	\$D (13)	\$E (14)	\$F (15)
\$20 (dez: 32)	!	"	*	\$	%	8	*	()	*	+	,	-	.	/	
\$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
\$40 (dez: 64)	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	0
\$50 (dez: 80)	P	Q	R	S	T	U	V	W	X	Y	Z	[\	^	-	
\$60 (dez: 96)	~	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
\$70 (dez: 112)	p	q	r	s	t	u	v	w	x	y	z	{	}	^	-	
\$80 (dez: 128)	€	ü	é	â	ä	à	å	ç	è	ë	ì	í	í	ñ	á	
\$90 (dez: 144)	É	æ	Æ	ô	ö	ò	û	ÿ	ö	ü	þ	£	¥	ß	ƒ	
\$A0 (dez: 160)	á	í	ó	ú	ñ	ä	ë	ξ	ó	í	í	í	í	í	í	
\$B0 (dez: 176)																
\$C0 (dez: 192)																
\$D0 (dez: 208)																
\$E0 (dez: 224)	α	β	Γ	π	Σ	σ	μ	τ	ξ	θ	Ω	δ	∅	ε	η	
\$F0 (dez: 240)	≡	±	≥	≤	ƒ	J	÷	≈	°	*	*	•	∫	n	z	ʒ

Font 3: 7x12 monospaced

+ Lower Upper \	\$0 (0)	\$1 (1)	\$2 (2)	\$3 (3)	\$4 (4)	\$5 (5)	\$6 (6)	\$7 (7)	\$8 (8)	\$9 (9)	\$A (10)	\$B (11)	\$C (12)	\$D (13)	\$E (14)	\$F (15)
\$20 (dez: 32)	!	"	*	\$	%	8	*	()	*	+	,	-	.	/	
\$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
\$40 (dez: 64)	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	0
\$50 (dez: 80)	P	Q	R	S	T	U	V	W	X	Y	Z	[\	^	-	
\$60 (dez: 96)	~	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
\$70 (dez: 112)	p	q	r	s	t	u	v	w	x	y	z	{	}	^	-	
\$80 (dez: 128)	€	ü	é	â	ä	à	å	ç	è	ë	ì	í	í	í	í	
\$90 (dez: 144)	É	æ	Æ	ô	ö	ò	û	ÿ	ö	ü	þ	£	¥	ß	ƒ	
\$A0 (dez: 160)	á	í	ó	ú	ñ	ä	ë	ξ	ó	í	í	í	í	í	í	
\$B0 (dez: 176)																
\$C0 (dez: 192)																
\$D0 (dez: 208)																
\$E0 (dez: 224)	α	β	Γ	π	Σ	σ	μ	τ	ξ	θ	Ω	δ	∅	ε	η	
\$F0 (dez: 240)	≡	±	≥	≤	ƒ	J	÷	≈	°	*	*	•	∫	n	z	ʒ

Font 2: 6x8 monospaced

+ Lower Upper \	\$0 (0)	\$1 (1)	\$2 (2)	\$3 (3)	\$4 (4)	\$5 (5)	\$6 (6)	\$7 (7)	\$8 (8)	\$9 (9)	\$A (10)	\$B (11)	\$C (12)	\$D (13)	\$E (14)	\$F (15)
\$20 (dez: 32)	!	"	*	\$	%	8	*	()	*	+	,	-	.	/	
\$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
\$40 (dez: 64)	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	0
\$50 (dez: 80)	P	Q	R	S	T	U	V	W	X	Y	Z	[\	^	-	
\$60 (dez: 96)	~	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
\$70 (dez: 112)	p	q	r	s	t	u	v	w	x	y	z	{	}	^	-	
\$80 (dez: 128)	€	ü	é	â	ä	à	å	ç	è	ë	ì	í	í	í	í	
\$90 (dez: 144)	É	æ	Æ	ô	ö	ò	û	ÿ	ö	ü	þ	£	¥	ß	ƒ	
\$A0 (dez: 160)	á	í	ó	ú	ñ	ä	ë	ξ	ó	í	í	í	í	í	í	
\$B0 (dez: 176)																
\$C0 (dez: 192)																
\$D0 (dez: 208)																
\$E0 (dez: 224)	β															
\$F0 (dez: 240)	ø															

Font 4: GENEVA10 proportional

EA eDIPTFT43-A

+ Lower Upper \	\$0 (0)	\$1 (1)	\$2 (2)	\$3 (3)	\$4 (4)	\$5 (5)	\$6 (6)	\$7 (7)	\$8 (8)	\$9 (9)	\$A (10)	\$B (11)	\$C (12)	\$D (13)	\$E (14)	\$F (15)
\$20 (dez: 32)	!	"	#	\$	%	฿	'	()	*	+	,	-	.	/		
\$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
\$40 (dez: 64)	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
\$50 (dez: 80)	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	-		
\$60 (dez: 96)	~	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
\$70 (dez: 112)	p	q	r	s	t	u	v	w	x	y	z	{ }	~	Δ		
\$80 (dez: 128)	€	ü	é	â	ä	à	å	ç	ê	ë	è	í	î	ï	â	å
\$90 (dez: 144)	É	æ	Œ	Ô	Ö	Ò	Û	Ù	Ý	Ó	Ü					
\$A0 (dez: 160)	á	í	ó	ú	ñ	Ñ	ä	ö								
\$B0 (dez: 176)																
\$C0 (dez: 192)																
\$D0 (dez: 208)																
\$E0 (dez: 224)		ß														
\$F0 (dez: 240)									°							

Font 5: CHICAGO14 proportional

+ Lower Upper \	\$0 (0)	\$1 (1)	\$2 (2)	\$3 (3)	\$4 (4)	\$5 (5)	\$6 (6)	\$7 (7)	\$8 (8)	\$9 (9)	\$A (10)	\$B (11)	\$C (12)	\$D (13)	\$E (14)	\$F (15)
\$20 (dez: 32)	!	"	#	\$	%	฿	'	()	*	+	,	-	.	/		
\$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
\$40 (dez: 64)	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
\$50 (dez: 80)	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_		
\$60 (dez: 96)	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
\$70 (dez: 112)	p	q	r	s	t	u	v	w	x	y	z	{ }	~	Δ		
\$80 (dez: 128)	€	ü	é	â	ä	à	å	ç	ê	ë	è	í	î	ï	â	å
\$90 (dez: 144)	É	æ	Œ	Ô	Ö	Ò	Û	Ù	Ý	Ó	Ü					
\$A0 (dez: 160)	á	í	ó	ú	ñ	Ñ	ä	ö								
\$B0 (dez: 176)																
\$C0 (dez: 192)																
\$D0 (dez: 208)																
\$E0 (dez: 224)		ß														
\$F0 (dez: 240)									°							

Font 6: Swiss30 Bold proportional

+ Lower Upper \	\$0 (0)	\$1 (1)	\$2 (2)	\$3 (3)	\$4 (4)	\$5 (5)	\$6 (6)	\$7 (7)	\$8 (8)	\$9 (9)	\$A (10)	\$B (11)	\$C (12)	\$D (13)	\$E (14)	\$F (15)
\$20 (dez: 32)											+	-	.			
\$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:					

Font 7: big numbers BigZif50

+ Lower Upper \	\$0 (0)	\$1 (1)	\$2 (2)	\$3 (3)	\$4 (4)	\$5 (5)	\$6 (6)	\$7 (7)	\$8 (8)	\$9 (9)	\$A (10)	\$B (11)	\$C (12)	\$D (13)	\$E (14)	\$F (15)
\$20 (dez: 32)											+	-	.			
\$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:					

Font 8: big numbers BigZif100



This hard copy shows all the fonts with which the product is shipped

Additional Fonts

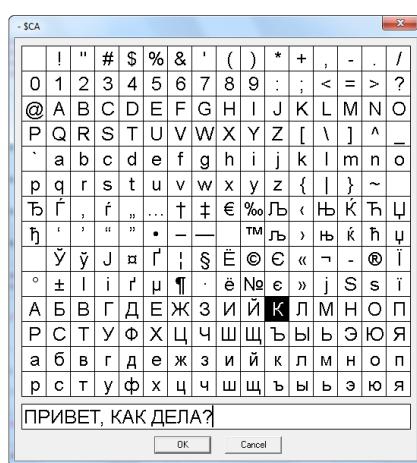
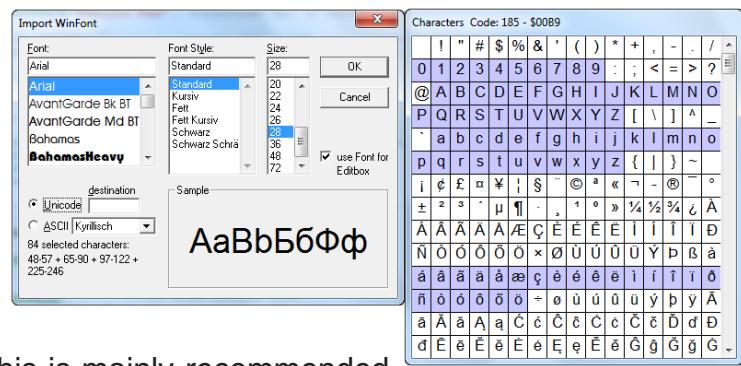
Up to 256 fonts á 16 pages can be loaded into the internal DataFlash.

COMPILER OPTION "WinFont:"

It is possible to raster TrueType-Fonts in different sizes which can be used. A double-click to the fontname within the KitEditor opens the font selection box.

To simplify the use of fonts, there is the possibility of a edit box. If you output a string with KitEditor (e.g. #ZL 5,5, "Hello"), you can perform a double click on the string to open it.

Now you can select the desired characters. This is mainly recommended



65,536 COLORS

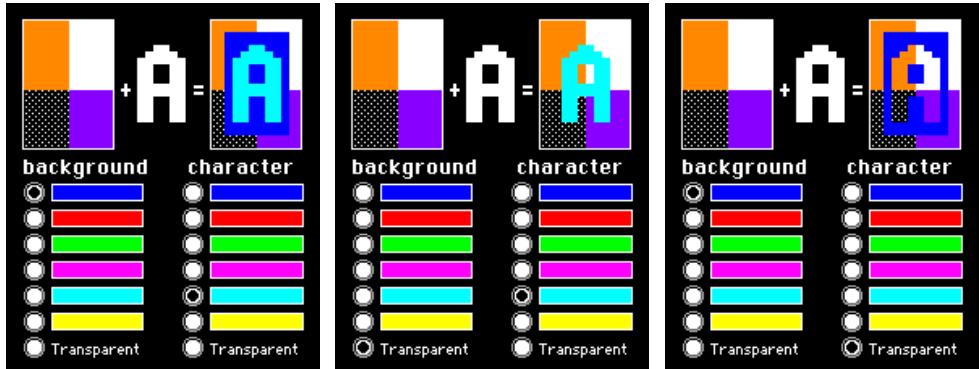
EA eDIPTFT32-A is able to work with 65,536 colors for true-color pictures/icons and animations. For an easy use there exists a color palette with 32 entries (16 colors are predefined after PowerOn). This color palette can be redefined at any time without changing the content of the display (command: ESC FP no R G B). To use a color for text and graphic functions you set only a number between 1..32.

Color	R	G	B
1	0	0	0
2	0	0	255
3	255	0	0
4	0	255	0
5	255	0	255

Color	R	G	B
1	0	0	0
2	0	0	255
3	255	0	0
4	0	255	0
5	255	0	255
6	0	255	255
7	255	255	0
8	255	255	255
9	111	111	111
10	255	143	0
11	143	0	255
12	255	0	143
13	0	255	143
14	143	255	0
15	0	143	255
16	175	175	175



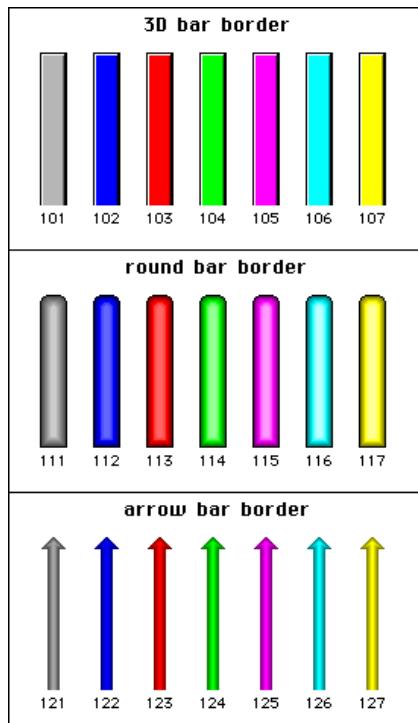
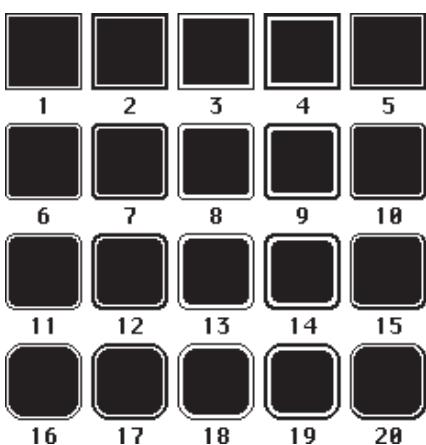
is used to invert all dots (=complementary). Two times inverted will end same as action was started (original drawing is restored).



some examples to show the letter 'A' onto a fixed background

BORDERS, KEY STYLES AND BARGRAPH

The eDIP is shipped with 20 predefined border (no:1..20) for the commands draw box frame and draw touchkeys. There are also three special borders in various colors for using with bargraph commands (no:101..107, 111..117 and 121..127). All of them can be used in various sizes via coordinates. The frames 1..20 are split into 3 segments: the outer frame, inner frame and filling. Each segment will get an individual colour for normal and for selected state. This will give the user the opportunity, when touching a field, the individual part of the key will be inverted only.

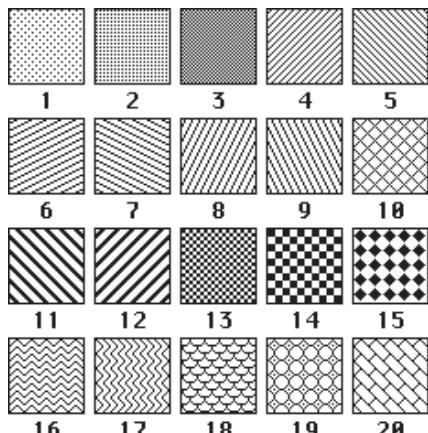
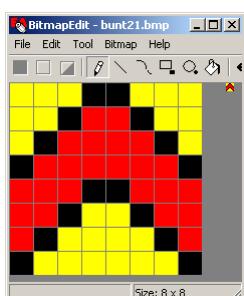


FILL PATTERNS

A pattern type can be set as a parameter with various commands. In this way, for example, rectangular areas and bar graphs can be filled with different patterns. The eDIP is shipped with 20 predefined fill patterns.

Define own pattern:

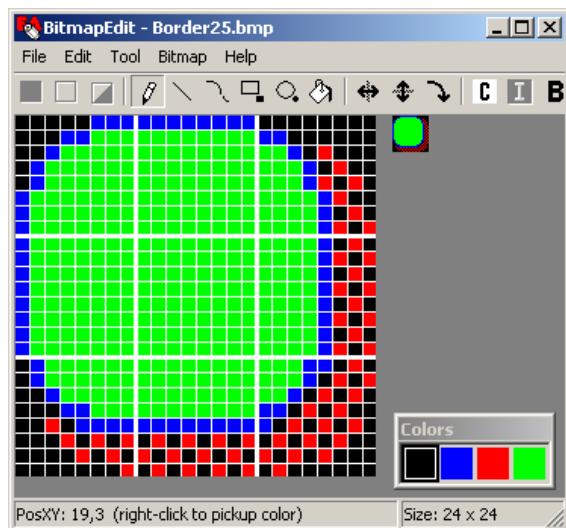
With the LCD-tools it is possible to define new pattern (=bitmaps with exactly 8x8 dots).(Compileroption "Pattern:"). The foreground- and background color can be set for monochrome pattern (as the 20 preloaded pattern) incl. transparency. It is also possible to define full colored pattern. With the LCD-tools some sample pattern has been installed (see folder 'Pattern').



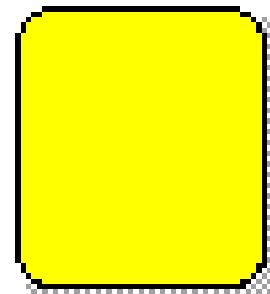
^{*) full version is free available on web at <http://www.lcd-module.de/eng/dip/edip.htm>}

DEFINE OWN BORDER

With the LCD-tools it is possible to define new borders (Compileroption "Border:"). Each of these new border is a bitmap with exactly 24x24 dots (9 segments with 8x8 dots: 4x edge, 4x middle part, 1x filling).



Scaling for bigger touchkeys/frames will be done by repetition of these 8x8 dot segments. This makes it necessary to keep the 8x8 size in every case. If 4-color bitmaps are used (as the preloaded border 1..20) the color can be set individually (the first color is always transparent and is not used by the eDIP). It is also possible to define full colored border (as the preloaded border for bargraphs 101...127). With the LCD-tools some sample border has been installed (see folder 'Bitmaps\Color\Border').

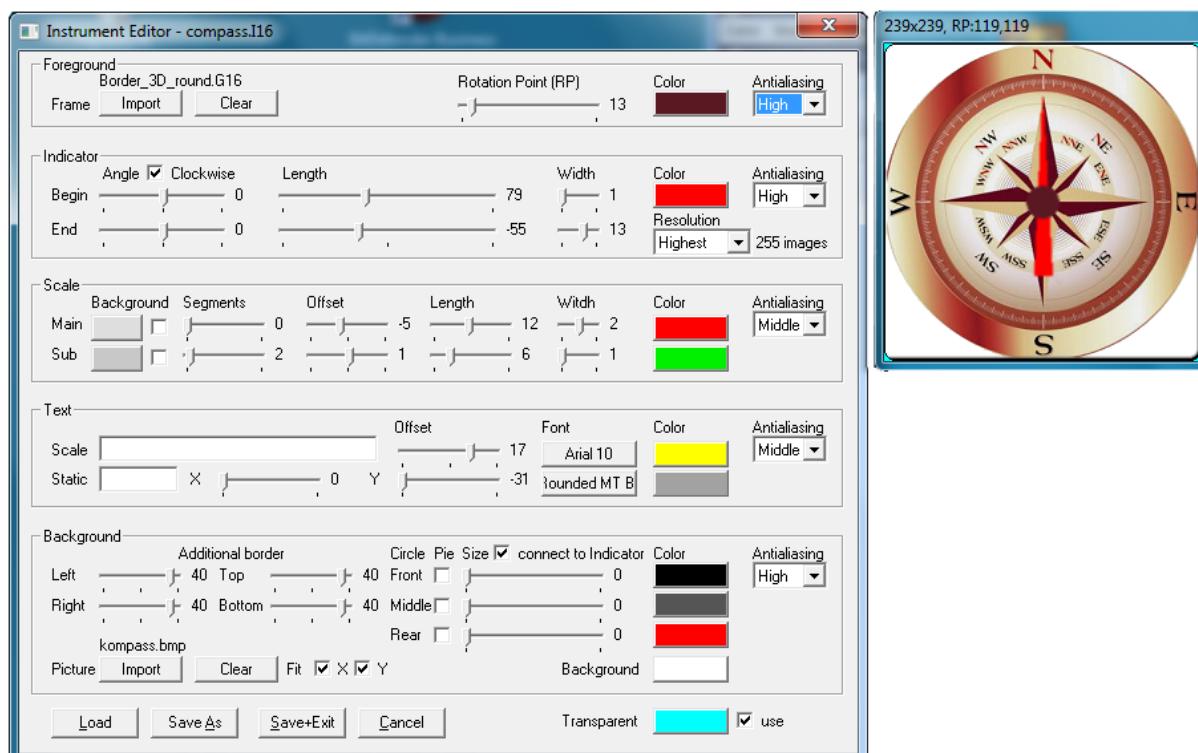


*border25:
50x56 Dot size*

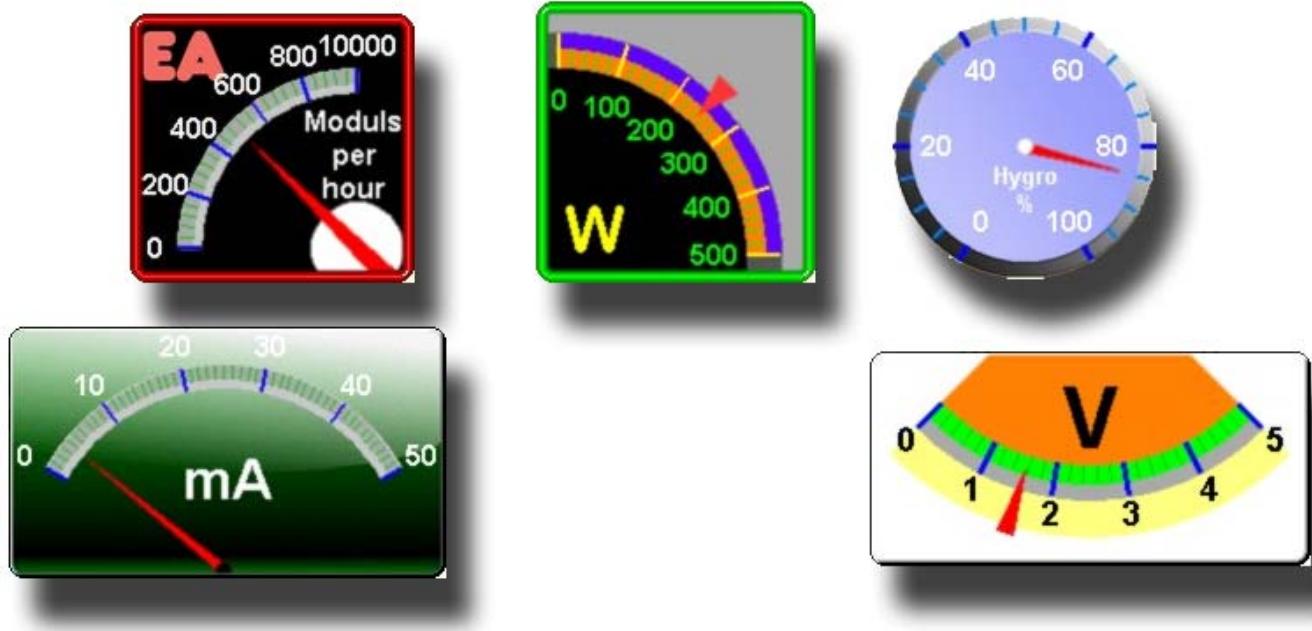
ROTARY AND POINTER INSTRUMENTS (SINCE V. 1.4)

With the help of the LCD-Tools it is possible to include instruments (Compiler command: `Instrument: 4, <instrument.i16>`). Performing a double click in the KitEditor on the instrument file opens the instrument editor.

The instruments are supported by convinient commands ('ESC I..'). For example the instruments are connectable to an analog input. In addition they are configurable by touch.



Some instrument examples

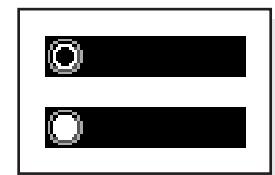


BUTTONS AS KEYS

Apart from the border types, which are infinitely scalable, it is also possible to use bitmaps as touch keys or touch switches (Compiler option "Button:").

A button always consists of two Bitmaps of equal size (one bitmap to display the touch key in its normal state and another for when it is pressed).

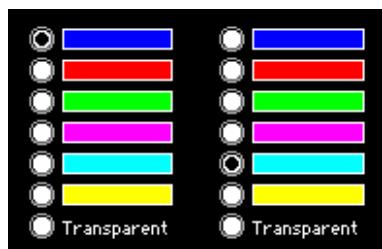
The active area of the touch key automatically results from the size of the button bitmaps.



RadioBlack75x15_0.bmp
RadioBlack75x15_1.bmp

SWITCHES IN GROUPS (RADIO GROUPS)

Touch switches change their status from *ON* to *OFF* or vice versa each time they are touched. Several touch switches can be included in a group ('ESC A R n1' command). If a touch switch in the group 'n1' is switched on, all the other touch switches in this group are automatically switched off. Only one switch is ever on (see table on page 16).



two radio groups with touch switches

CREATING INDIVIDUAL MACROS AND IMAGES

To create your own fonts, images, animations and macros you need the following:

- To connect the display to the PC, you need the EA 9777-2USB USB programmer, which is available as an accessory, or a self-built adapter with a MAX232 level converter (see the application example on p.5).
- ELECTRONIC ASSEMBLY LCD-Tools^{*}, which contains a kiteditor, bitmappeditor, ediptftcompiler, fonts, images, border, pattern and examples (for Windows PCs)
- A PC with an USB or serial COM interface

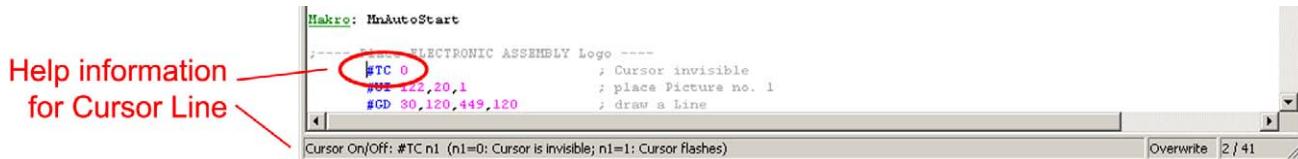
To define a sequence of commands as a macro, all the commands are written to a file on the PC (e.g. DEMO.KMC). You specify which character sets are to be integrated and which command sequences are to be in which macros.

If the macros are defined using the kit editor, you start the eDIPTFT compiler using F5. This creates a file called DEMO.DF. If an EA 9777-2USB programmer is also connected or the display is connected to the PC via a MAX232, this file is automatically burned in the display's data flash memory.

You can send the created macrofile *.DF with any other system to the EA eDIPTFT43-A. All programming commands are inside this file, so you only need to send the content of the *.df file (via RS232, SPI or I2C with smallprotocol in packets) to the EA eDIPTFT43-A.

KIT-EDITOR HELP (ELECTRONIC ASSEMBLY LCD TOOLS)

At bottom from the KitEditor window in the statusline you can see a short description for the current command and the parameters. For more information press F1.



Images

To save transfer time via serial interface, it is possible to store up to 256 bitmaps á 16 pages into internal dataflash (Compileroption "Picture:").

Following image file-formats can be used:

- BMP: Windows Bitmap with 1-, 4-, 8-, 16-, 24-, 32-BIT colordepth incl. RLE.
- GIF: Graphics Interchange Format incl. optionally transparency
- JPG: JPEG Compressed Images
- TGA: TARGA Images with 8-, 16-, 24-, 32-BIT colordepth incl. RLE and transparency.
- PNG: Portable Network Graphics incl. colour map, gray scale and transparency
- G16: internal eDIPTFT format, incl. RLE and transparency

All pictures are converted into internal G16 format with RLE encoding (saves memory).

Too big pictures are resized proportional (Compileroption "MaxSize:").

It is also possible to reduce the colordepth (Compileroption "MaxColorDepth:").

One color can be defined as transparent (Compileroption "MakeTransparent:“)

The internal pictures can be used with the command "ESC U I" via serial interface or from a macro.

The foreground- and background color can be set for monochrome pictures incl. transparency.

Animation

It is possible to store up to 256 animations á 16 pages into internal dataflash.

(Compileroption "Animation:").

Following image file-formats can be used:

- GIF: animated GIF (only identically transparent areas, transparency can be switched off).
- G16: internal animated eDIPTFT format
- two or more single bitmaps (BMP, GIF, JPG, TGA, PNG, G16) e.g. two bitmaps for blinking

Note that max. 4 animations (animationprocesses) can run at the same time . The animations are self-running pictures, but you can use the animations manually too.

The foreground- and background color can be set for monochrome animations.

Pattern

Patterns are used to fill a box, a bargraph or to draw a line. It is possible to store up to 256 pattern á 16 pages into internal dataflash (Compileroption "Pattern:").

Each bitmap (BMP, GIF, JPG, TGA, PNG, G16) with a size of 8x8 dots can be imported as a pattern.

The foreground- and background color can be set for monochrome pattern incl. transparency.

Border / Bargraph

A border can be scaled and is used for rectangles, bargraphs and touch keys/switches. It is possible to store up to 256 border á 16 pages into internal dataflash (Compileroption "Border:").

Each bitmap (BMP, GIF, JPG, TGA, PNG, G16) with a size of 24x24 dots can be imported as a border.

The transparency of GIF,TGA and G16 bitmaps is used by the EA eDIP.

It is possible to change the colors for 4-color borders, the first color ist always transparent and is not used by the EA eDIP. When used for a touch key/switch a second border can be loaded which will be used if the touch key/ switch is pressed.

Images for Touch keys (Buttons)

It is possible to store up to 256 touchkeys/buttons á 16 pages into internal dataflash.

(Compileroption "Button:").

A button consists of one or two images with the same size (BMP, GIF, JPG, TGA, PNG, G16).

The transparency of GIFTGA, PNG and G16 bitmaps is used by the EA eDIP and should be identical.

The first bitmap is used when the touch key/switch is released and the second bitmap is used if the touch key/ switch is pressed.

MACROS

Single or multiple command sequences can be grouped together in macros and stored in the data flash memory. You can then start them by using the *Run macro* commands. There are different types of macro (compiler directive marked in green letters):

Normal macros Macro:

These are started by means of an 'ESC MN xx' command via the serial interface or from another macro. A series of macros occurring one after the other can be called cyclically (movie, hourglass, multi-page help text). These automatic macros continue to be processed until either a command is received via the interface or a touch macro with a corresponding return code is activated.

Touch macros (1 to 255) TouchMacro:

Started when you touch/release a touch field (only in versions with a touch panel - TP / TC) or issue an 'ESC MT xx' command.

Bit macros BitMacro:

will be started by a single line IN 1..8 (bit) will change or by command 'ESC MB xx'.

Bit-Macro 1..8 are good for falling edge and Bit Macro 9..16 are good for rising edge at input 1..8. It is possible to change the assignment between Bitmacro and intput with command 'ESC YD n1 n2 n3' (since firmware V1.1, see page 17).

Port macros PortMacro:

These are started when voltage (binary) is applied to IN 1..8 or by command 'ESC MP xx'.

Matrix macros MatrixMacro:

Matrix Macro 1..64: start when keypressed or by command 'ESC MX xx'.

Matrix Macro 0: start after release of key or by command.

It is possible to change the assignment between keynumber and Matrixmacro with command 'ESC YX n1 n2 n3' (since firmware V1.1, see page 17)

Analogue Macros AnalogMacro:

will start whenever voltage changes or limit exceeds or by command 'ESC MV xx'. See table at the right:

It is possible to change the assignment between analoguemacrofunction and Analoguemacronumber with command 'ESC VM n1 n2' (since firmware V1.1, see page 15)

Process macros ProcessMacro:

automatic start at fixed periode (0.1s up to 25s) or by command 'ESC MC xx'. Up to 4 individual process may be defined by command 'ESC MD ...'. These Process Makro will never be stopped by other commands or activities.

Analogue Macro		
Macro No.		Macro starts at
AIN1	AIN2	
0	10	every change of input voltage
1	11	falling input voltage
2	12	rising input voltage
3	13	below lower limit
4	14	above lower limit
5	15	below upper limit
6	16	above upper limit
7	17	outside of both limits
8	18	inside of both limits
9	19	lower than other channel

Power-on macro PowerOnMacro:

Started after power-on. You can switch off the cursor and define an opening screen, for example.

Reset macro ResetMacro:

Started after an external reset (low level at pin 5).

Watchdog macro WatchdogMacro:

Started after a fault/error (e.g. failure).

Brown-out macro BrownoutMacro:

Started after a voltage drop under 4.6V.

Important: If a continuous loop is programmed in a power-on, reset, watchdog or brown-out macro, the display can no longer be addressed. In this case, the execution of the power-on macro must be suppressed. You do this by wiring DPOM:

- PowerOff - connect pin 13 (DPOM) to GND
- PowerOn - open pin 13 (DPOM) again.

MACRO PAGES (MULTILINGUAL CAPABILITY)

There are 16 complete macro sets available as well as the internal images and fonts. By simply switching the active macro page (ESC M K n1), for example, up to 16 different languages can thus be supported.

If a macro/picture is defined in the kit editor, a page number can be specified in square brackets after the macro/picture number.

If a macro/image is not defined in the currently set page [1] to [15], this macro/picture is automatically taken from page [0]. Thus, not all macros and images have to be stored separately for each language when they are identical in each language.

```
PICTURE: 100[0] <BIER.BMP>
PICTURE: 100[1] <BEER.BMP>
PICTURE: 100[2] <BIRRA.BMP>

MACRO: 2[0] ; SAME AS "MACRO: 2"
#FZ 3,1
#ZL 25,0 "DEUTSCH "
#UI 0,20, 100

MACRO: 2[1] ; ENGLISH
#FZ 3,1
#ZL 25,0 "ENGLISH "
#UI 0,20, 100

MACRO: 2[2] ; ITALIAN
#FZ 3,1
#ZL 25,0 "ITALIAN "
#UI 0,20, 100
```

If there is only the need of handling strings differently, stringtables might be used. Depending on the active macropage (ESC M K n1) the desired string is called. Please check the following example:

```
STRINGCODE=$01

STRING: 100[0] "HALLO WELT "
STRING: 100[1] "HELLO WORLD "
STRING: 100[2] "CIAO A TUTTI "

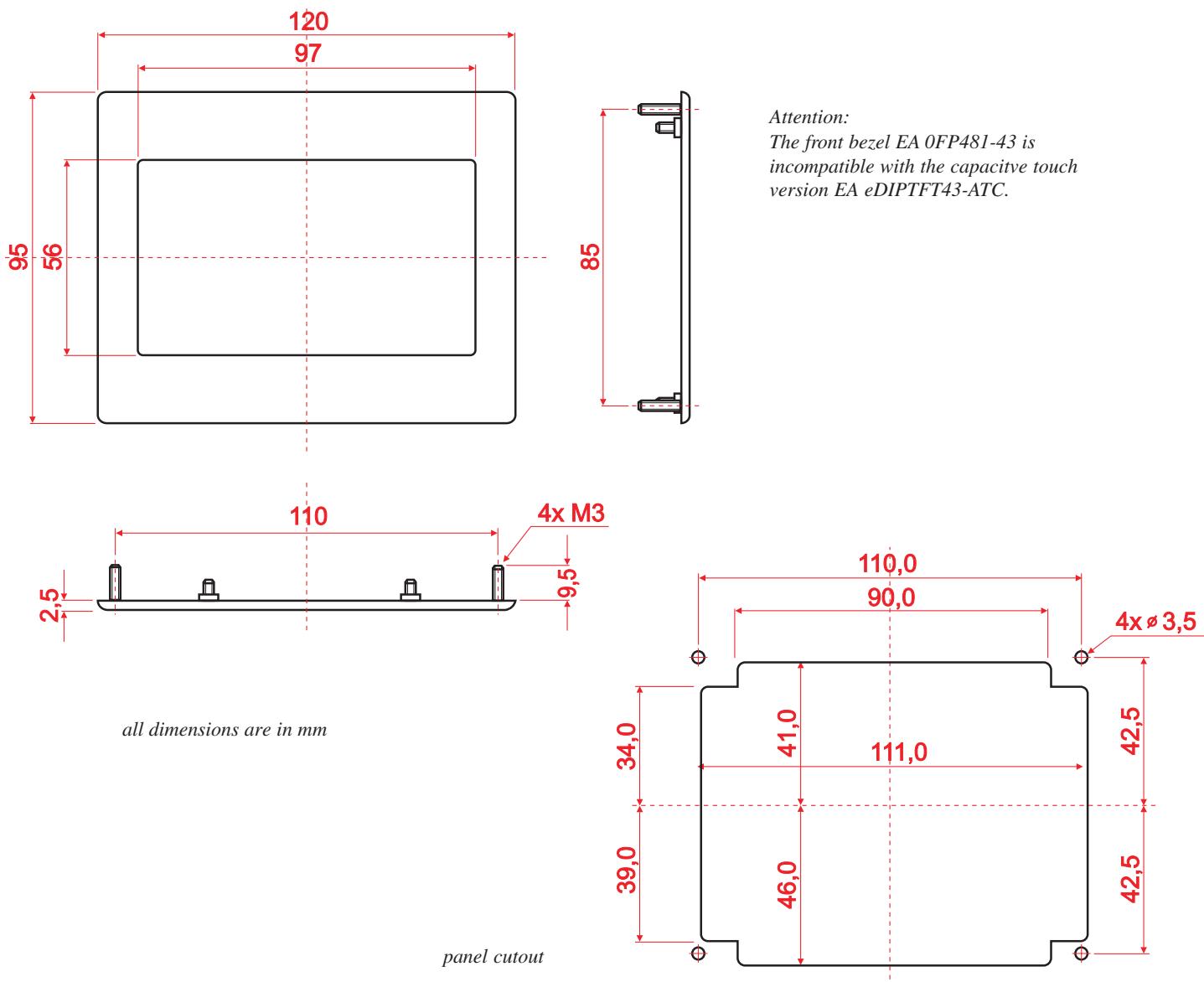
MACRO: 1
#ST StringCode
#ZL 10,5, StringCode, 100
```

WRITE PROTECTION FOR MACRO PROGRAMMING AND FONTS

A LO level at pin 19 (WP) prevents the macros, images and fonts in the data flash memory from being overwritten inadvertently (so it is highly recommended!).

SPECIFICATION AND CHARACTERISTICS

Characteristics					
Value	Condition	min.	typ.	max.	Unit
Operating Temperature		-20		+70	°C
Storage Temperature		-30		+80	°C
Storage Humidity	< 40°C			90	%RH
Operating Voltage		4.9	5.0	5.1	V
Input Low Voltage		-0.5		0.3*VDD	V
Input High Voltage	Pin Reset only	0.9*VDD		VDD+0.5	V
Input High Voltage	except Reset	0.6*VDD		VDD+0.5	V
Input Leakage Current	Pin MOSI only			1	uA
Input Pull-up Resistor		20		50	kOhms
Output Low Voltage				0.7	V
Output High Voltage		4.2			V
Brightness (white)	w./o. Touch		500		cd/m ²
	with Touch		410		cd/m ²
Output Current				20	mA
Power Supply	Backlight 100%		180		mA
	Backlight off		80		mA

EA eDIPTFT43-A**MOUNTING BEZEL EA 0FP481-43SW****NOTES ON HANDLING AND OPERATION**

- The module can be destroyed by polarity reversal or overvoltage of the power supply; overvoltage, reverse polarity or static discharge at the inputs; or short-circuiting of the outputs.
- It is essential that the power supply is switched off before the module is disconnected. All inputs must also be deenergized.
- The display and touch screen are made of plastic and must not come into contact with hard objects. The surfaces can be cleaned using a soft cloth without solvents.
- The module is designed exclusively for use in buildings. Additional measures have to be taken if it is to be used outdoors. The maximum temperature range of -20 to +70°C must not be exceeded. If used in a damp environment, the module may malfunction or fail. The display must be protected from direct sunshine.
- TFT displays may show residual or "ghost" image of the previous image when screen image is changed after a long period. To prevent such effect it is necessary to alternate the display content.
- Dot errors are allowed following ISO13406-2 class II.



EA eDIPTFT43-A

DIMENSIONS

