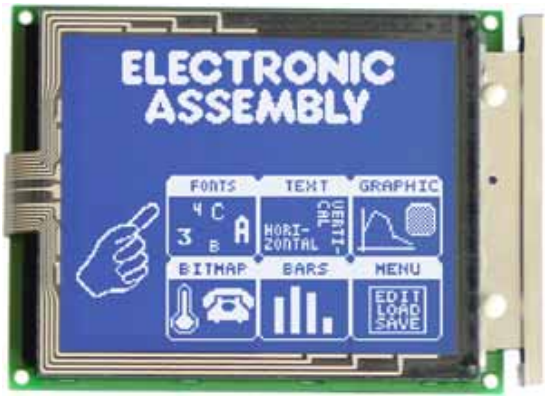
EA KIT160-7

CONTROL PANEL WITH FONTS, GRAPHICS COMMANDS AND MACROS

5,1" Touch Panel included



EA KIT160-7LWTP
Dimensions 140x102mm

TECHNICAL DATA

* 160x128 PIXELS WITH CFL ILLUMINATION, BLUE NEGATIVE
* ALSO WITH LOG-LIFE LED BACKLIGHT WHITE-BLUE
* INTEGRATED TOUCH PANEL WITH 8x7 FIELDS (ANTI-GLARE, SCRATCH-RESISTANT)
* FONT ZOOM OF approx. 3mm VIA approx. 5mm UP TO approx. 50mm
* SUPPLY VOLTAGE 5V/500mA (-C)/300mA (-LW) OR OPTIONALLY 9..35V
* RS-232 OR ALTERNATIVELY RS-422 WITH BAUD RATES 1200..115200
* POSITIONING **ACCURATE TO THE PIXEL** WITH ALL FUNCTIONS
* PROGRAMMING BY MEANS OF HIGH-LEVEL LANGUAGE-TYPE COMMANDS:
* STRAIGHT LINE, POINT, AREA, AND/OR/EXOR, BAR GRAPH...
* UP TO 256 MACROS PROGRAMMABLE
* COMBINATION OF TEXT AND GRAPHICS
* 4 CLIPBOARD FUNCTIONS, PULL-DOWN MENUS
* 8 DIGITAL INPUT AND 8 DIGITAL OUTPUT
* BACKLIGHT CAN BE SWITCHED ON/OFF BY SOFTWARE CONTROL

ACCESSORIES

SUPPLY VOLTAGE 9..35V INSTEAD OF 5V	**EA OPT-9/35V**
RS-422 INTERFACE INSTEAD OF RS-232	**EA OPT-RS4224**
OPTOCOUPLER FOR 8 INPUTS AND 8 OUTPUTS	**EA OPT-OPTO16**
ALUMINUM BEZEL: BLACK ANODIZED	**EA 0FP160-7SW**
ALUMINUM BEZEL: BLUE ANODIZED	**EA 0FP160-7BL**
CABLE (1.5m) FOR CONNECTION TO 9-PIN SUB-D (RS-232 FEMALE)	**EA KV24-9B**
FLOPPY DISK FOR MACRO PROGRAMMING (PC DOS/WIN)	**EA DISK240**

ORDERING INFORMATION

160x128 DOTS WITH CFL ILLUMINATION, BLUE NEGATIVE, TOUCH PANEL	**EA KIT160-7CTP**
DITO WITHOUT TOUCH PANEL	**EA KIT160-7C**
160x128 DOTS WITH **WHITE LED-B/L.**, BLUE NEGATIVE, TOUCH PANEL	**EA KIT160-7LWTP**
DITO WITHOUT TOUCH PANEL	**EA KIT160-7LW**

ELECTRONIC ASSEMBLY GMBH
ZEPPELINSTRASSE 19 · D-82205 GILCHING
PHONE +49-8105-778090 · FAX +49-8105-778099 · http://www.lcd-module.de

GENERAL

The EA KIT160-7 graphics kit is a fully assembled control and operating unit with a variety of integrated functions. The display has very compact dimensions and offers excellent super-twist contrast, which means the unit can be put into operation immediately. It is controlled via the standard RS-232 or RS-422 interface. In addition to complete graphics routines for display output, the graphics kit also contains a wide variety of fonts. Graphics command similar to those used in high-level programming languages are used for programming. There is thus no longer any need for the time-consuming programming of character sets and graphics routines. The ease of use offered by macros and input via touch panel make it a real power display.

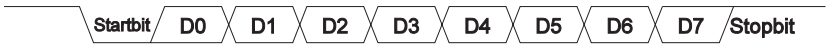
DISPLAY OPTIONS

CFL-Backlight	EA KIT160-7CTP: blue background with white characters. Extreme bright and contrastful. Life time of backlight 10,000~20,000 hours. Backlight unit is available as a spare part: EA CFL160-7. Power consumption: typ. 300mA@5V
LED-Backlight	EA KIT160-7LWTP: blue background with white characters. Great contrast, power consumption: typ. 250mA@5V

HARDWARE

The graphics kit is designed for an operating voltage of +5V. A supply voltage of 9..35V is also possible. Serial asynchronous data transfer is carried out in RS-232 or RS-422 format. The transmission format is set permanently to 8 data bits, 1 stop bits, and no parity. A transimssion rate of between 1200 and 115,200 baud can be selected by means of DIP switches. RTS and CTS handshake lines are available.

Data format:



TOUCH PANEL

The EA KIT160-7CTP and -7LWTP versions are equipped with an integrated touch panel. You can make entries and choose menu 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 or grouping of several fields is handled by the integrated software.

SOFTWARE

The graphic kits are programmed by means of commands such as *Draw a rectangle from (0,0) to (64,15)*. No additional software or drivers are required. Strings can be placed with **pixel accuracy**. Text and graphics can be combined at any time. Up to 16 different character sets can be used. Thus, when the 8-times zoom is used with the largest character set (16x8), the words and numbers fill the screen (128x64).

ACCESSORIES

Front panel for mounting

A front panel made of anodized aluminum is available as an accessory. This allows the graphics kit to be mounted without any screws visible. Installing it is child's play. The EA 0FP160-7 front panel is available in black (SW) and blue (BL).

Floppy disk for creating macros

A floppy disk (EA DISK240) is required for macro programming*). This converts the commands entered in a text file into a code that can be read by the graphics kit, and programs them into the EEPROM.

Cable for PC

To enable simple connection to PCs (macro programming), we provide a 1.5m cable and a 9-pin SUB-D female connector (EA KV24-9B). Simply insert it into COM 1 or COM 2 and get started. Note: The cable is not suitable for the RS-422 version (EA OPT-RS4224).

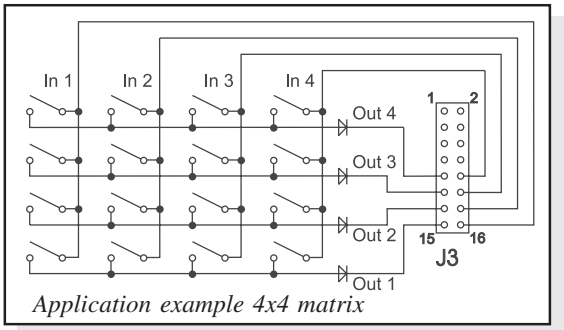
*) *Also on the Internet: http://www.lcd-module.de/deu/disk/disk240.zip*

EXTERNAL KEYBOARD

A keyboard (anything from individual keys to a 8x7 matrix keyboard, J8) can be connected at the plug-in connection. The connected keys are debounced by means of software. Please note that it is only possible to connect an external keyboard to versions without an integrated touch panel.
Each key is switched between an output and an input. Each input has a 100kΩ pullup. Up to 6 keys can be connected at each output.
To find out double-key-strokes all outputs must be decoupled by a Schottky-Diode (e.g. BAT 43).

Matrix - Keypad Connector J8

Pin	Symbol	Function	Pin	Symbol	Function
1	-	nc	2	IN 8	input column 8
3	OUT 7	output line 7	4	IN 7	input column 7
5	OUT 6	output line 6	6	IN 6	input column 6
7	OUT 5	output line 5	8	IN 5	input column 5
9	OUT 4	output line 4	10	IN 4	input column 4
11	OUT 3	output line 3	12	IN 3	input column 3
13	OUT 2	output line 2	14	IN 2	input column 2
15	OUT 1	output line 1	16	IN 1	input column 1



Application example 4x4 matrix

Transmitting the keystrokes

At each keystroke, the associated key number (1..56) is transmitted or an internal Touch Makro (only if defined) 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 touch macro no. 0. The automatic keyboard scan can be deactivated by means of the command "ESC T A 0". If the handshake line (e.g. CTS) does not permit transmission, keystrokes can be lost.
The key number can be determined as follows:
Key number = (output -1) * 8 +input
(output: a number between 1 and 7; input: a number between 1 and 8).

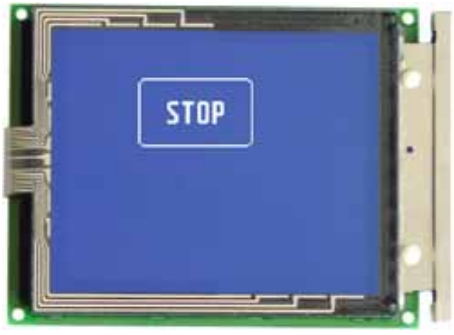
TOUCH PANEL (EA KIT160-7XXTP ONLY)

Versions EA KIT160-7CTP and -7LWTP are supplied with an integrated touch panel with 56 fields. The graphics kit offers convenient commands supporting this touch panel. It is possible, for example, to group a number of touch fields to form a single large key and then draw and label the key. You can also assign a record code (1..255) to the key you have defined. If a return code of 0 is assigned, the key is disabled and has no effect when it is pressed.
When the touch keys are touched, they can be automatically inverted and a tone can sound, indicating they have been touched. At the same time, the defined return code of the key is transmitted via the serial interface, or an internal touch macro with the number of the return code is started.

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56

Example:
Definition of a key from field 11 to 21 with the return code 65=´A´ and the text "STOP". Note: Before individual keys are defined, all fields should be disabled by means of "ESC T R".

Example	Codes to be output												Note
For compiler	#TH 11, 21, ´A´, 2, "STOP"												The end code 0 is not specified here
As ASCII	ESC	T	H	.	.	A	.	S	T	O	P	.	The dots ´.´ stand for ASCII characters that are not to be displayed
In hex	$1B	$54	$48	$0B	$15	$41	$02	$53	$54	$4F	$50	$00	
In decimal	27	84	72	11	21	65	2	83	84	79	80	0	
	Befehlskennung	Einleitung Touch-Befehl	horizontale Beschriftung	linke oberes Touchfeld	rechtes untere Touchfeld	Return Code	Taste zeichnen mit Rahmen					Text Ende Kennung	



BAUD RATES

The baud rate can be set by means of the 3 DIP switches on the left. 9,600 baud is set at the factory (DIP 3 ON). Please note that the internal data buffer only holds 24 bytes. It is therefore imperative that the RTS handshake line be queried (a level of +10V means data can be accepted; a level of -10V means the display is busy). The data format is fixed at 8 data bits, 1 stop bit and no parity.

Baud rates			
DIP switches			Data format 8,N,1
1	2	3	
ON	ON	ON	1200
OFF	ON	ON	2400
ON	OFF	ON	4800
OFF	OFF	ON	9600
ON	ON	OFF	19200
OFF	ON	OFF	38400
ON	OFF	OFF	57600
OFF	OFF	OFF	115200

WRITE PROTECTION FOR PROGRAMMED MACROS

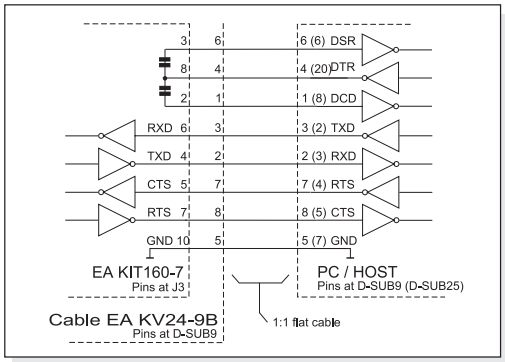
You can use DIP switch 6 to prevent the programmed macros, images and fonts from being inadvertently overwritten.

Write protection	
DIP 6	Write protection for EEPROM
ON	On No macro progr. possible
OFF	Off Macro progr. possible

RS-232/RS-422 CONNECTION

The graphics kit is shipped with an RS-232 interface as standard. The pin assignment of the plug connector (J3) is as shown in the table on the left. The J3 has a 2.54mm grid. If the graphics kit is ordered together with the EA OPT-RS4224 optional component, RS-422 drivers are fitted. In this case, the pin assignment is as shown in the table on the right.
The same serial data with 5V levels and TTL logic is available at the J5 eyelet strip. These levels are suitable for direct connection to a µC. However, if these signals are used, solder link LB5 and LB6 must be cut or opened.

RS-232 J3 connection			
Pin	Symbo	In/Out	Function
1	VDD	-	+ 5V supply
2	DCD	-	Strap to DTR
3	DSR	-	Strap to DTR
4	TxD	Out	Transmit data
5	CTS	In	Clear to send
6	RxD	In	Receive data
7	RTS	Out	Request to send
8	DTR	-	See pin 2, pin 3
9	-	-	NC
10	GND	-	0V ground

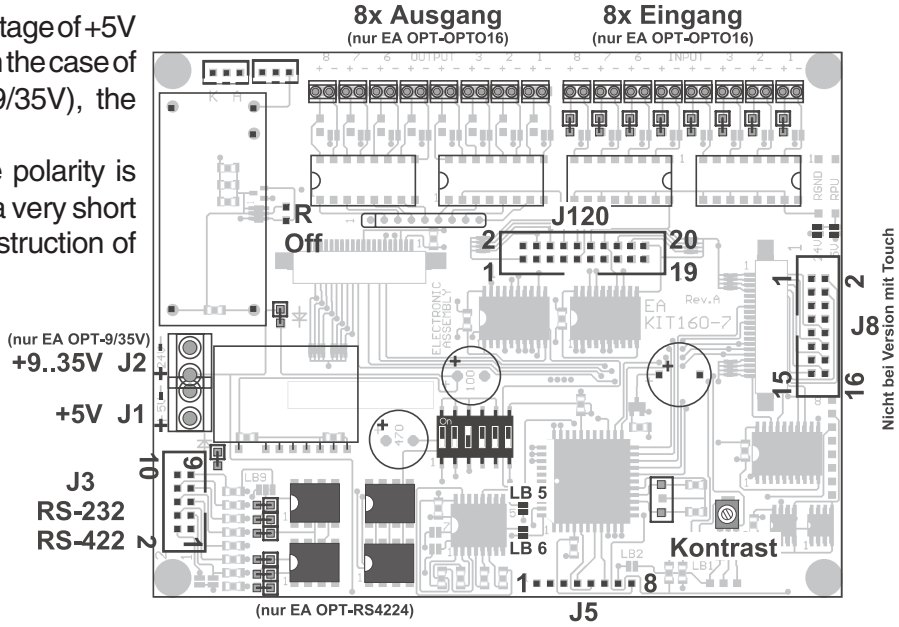


RS-422 J3 connection		
Pin	Symbol	Function
1	VDD	+ 5V supply
2	Data In-	Receive data
3	Data In+	Receive data
4	Data Out-	Transmit data
5	Data Out+	Transmit data
6	HS In-	Handshake
7	HS In+	Handshake
8	HS Out-	Handshake
9	HS Out+	Handshake
10	GND	0V ground

SUPPLY VOLTAGE / EA OPT-9/35V

In the standard model, the supply voltage of +5V is fed in via screw-type terminal J1. In the case of the version for 9..35V (EA OPT-9/35V), the power is supplied via J2.
Important: It is imperative that the polarity is correct. Polarity reversal, even for a very short time, can cause the immediate destruction of the entire display.

J5 ad-on			
Pin	Symbol	In/Out	Function
1	VU	-	9..35V Supply
2	VDD	-	+ 5V Supply
3	GND	-	0V, Ground
4	TxD5	Out	Transmit Data
5	RxD5	In	Receive Data
6	RTS5	Out	Request To Send
7	CTS5	In	Clear To Send
8	RESET	In	H: Reset



DIGITAL INPUTS AND OUTPUTS

All EA KIT160-7 series provide 8 digital In- and 8 outputs (5V CMOS level, grounded).

8 outputs

Each line can be controlled individually using the "ESC Y W" command. A maximum current of 10mA can be switched per line. For more power use an external transistor or MOSFET.

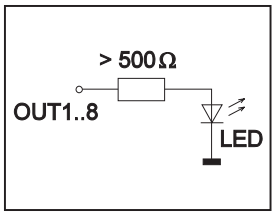
8 inputs

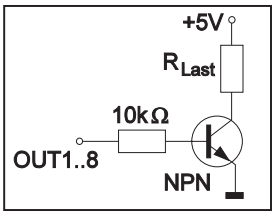
A voltage of >4V starts an internal port macro. However, the inputs can also be queried and evaluated directly via the serial interface ("ESC Y R"). When the 8 lines are combined, up to 256 port macros can thus be addressed. Each of these port macros can change the contents of the screen or switch an output, thus enabling a wide range of control functions. To create the port macros you need a PC and the EA DISK240 floppy disk. You will find a more detailed description on page 6. The automatic port query can be disabled by means of the "ESC Y A 0" command.

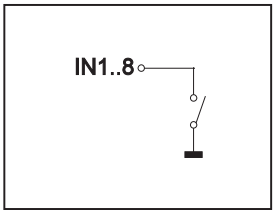
Note: The logic circuitry is designed for slow operations; in other words, more than 3 changes per second cannot be easily executed. If an input is left open, it is logical high (internaly pulled-up via 100 kOhm).

J120 Inputs and Outputs

Pin	Symbol	Function	Pin	Symbol	Function
1	VDD	+5V Supply	2	GND	0V, Ground
3	OUT 1	Output 1	4	IN 1	Input 1
5	OUT 2	Output 2	6	IN 2	Input 2
7	OUT 3	Output 3	8	IN 3	Input 3
9	OUT 4	Output 4	10	IN 4	Input 4
11	OUT 5	Output 5	12	IN 5	Input 5
13	OUT 6	Output 6	14	IN 6	Input 6
15	OUT 7	Output 7	16	IN 7	Input 7
17	OUT 8	Output 8	18	IN 8	Input 8
19	GND	0V, Ground	20	VDD	+5V Supply



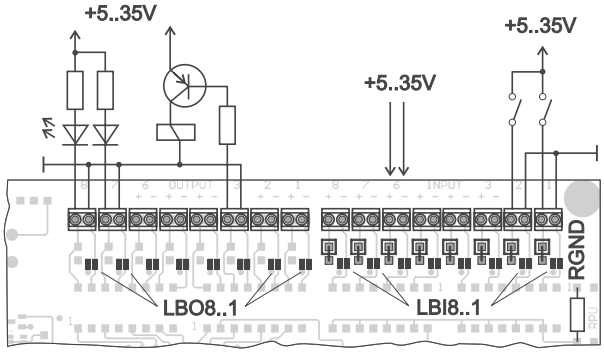




IN- AND OUTPUTSVIA OPTOCOUPLER (EA OPT-OPTO16)

Optionally all In- and Outputs are isolated via optocouppler circuit (EA OPT-OPTO16 only). The connection is made via 16 different screw-type terminals. Direct voltages of 5..35V can be applied at all 8 inputs. Voltages of over 4V are recognized as high level, while voltages of under 2V are low level. Voltages of between 2 and 4V are undefined. The polarity is insignificant.

Output is a "open-collector" circuit with collector (+) and emitter (-) of a NPN transistor. A maximum current of 10mA can be switched per line.



Note: The negative pole of each screw-type terminal can be interconnected by closing the solder straps LBI1..8 and LBO1..8. These solder straps can also be connected to system ground GND (solder 0Ω strap R_{GND}).

Note: Logic will be inverted by optocouppler circuit (all inputs left open: port-macro #255). The command "ESC Y I 1" inverts logic back again (all inputs left open: port-macro #0)

DEFAULT SETTINGS

After power-on or a manual reset, the registers shown here are set to a specific value.

Please note that all the settings can be overwritten by creating a power-on macro (normal macro no. 0).

Default settings

Register	Command	After power-on/reset
Text mode	ESC L	Set, black
Terminal font	ESC FT	Font 3, no zoom
Cursor	ESC QC	On
Flashing time	ESC QZ	0.6 secs
User-defined characters	ESC E	Undefined
Graphics mode	ESC V	Set
Graphics font	ESC F	Font 3, no zoom
Last xy	ESC W	(0;0)
Bar graph 1..16	ESC B	Undefined
Clipboard	ESC C	Empty
Select/deselect	ESC K	Selected
Outputs OUT1..8	ESC Y	Low level/open

MACRO PROGRAMMING

Single or multiple command sequences can be grouped together in macros and stored in the EEPROM. You can then start them by using the *Execute macro* commands. There are 3 different types of macros:

Touch macros (1..255)

These are started when you touch a touch field (in versions with a touch panel - TP) or when you operate an external key/matrix keyboard. Touch macro no. 0 is different: It is started when you release a key.

Port macros (0..255)

These are started when voltage is applied to IN 1..8.

Normal macros (1..255)

These are started by means of a 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).

Power-on macro

Normal macro no. 0 is different: It is executed automatically after power-on. It allows you to switch off the cursor and define an opening screen, for example.

Note: Programming a endless loop in Power-On-Macro makes the display non-accessible. In that case switch DIP 5 to ON position, power off, and then power on again and DIP 5 back to off position. Now all fonts and macro must be downloaded again.

STORING 256 IMAGES IN THE EEPROM

To reduce the transmission times of the serial interface or to save storage space in the processor system, up to 256 images can be stored in the internal EEPROM. They can be called using the "ESC U E" command via the serial interface or from within a touch/port/normal macro. All the images can be used in the Windows BMP format. They can be created and edit using widely available software such as Windows Paint or Photoshop.

CREATING INDIVIDUAL MACROS

To create your own macros, you need the following:

- The EA DISK240*) floppy disk, which contains a compiler, examples and fonts
- A PC with a COM1 or COM2 serial interface and approximately 500KB hard disk space
- A text editor such as WordPad or Norton Editor

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.

Once the macros are defined, you start the program C:>KITCOMP DEMO.KMC. This creates an EEPROM file called DEMO.EEP, which is then automatically stored in the display EEPROM with the baud rate entered. This only takes a few seconds, and you can then use your user-defined macros immediately. You will find a detailed description of how to program macros, together with a large number of examples, in the files DOKU.DOC (for WORD) and DOKU.TXT (DOS) on the EA DISK240*) floppy disk.

```
;Makro Demo
KIT160-7                    ; KIT festlegen
COM2: 115200                ; KIT ist an COM2 angeschlossen,
                            ; Übertragung mit 115.200 Baud
;-------------------------------------------------------------
;Konstanten definieren
AUS =  0
EIN =  1
FONT4x6 = 1
FONT5x6 = 2
FONT6x8 = 3
FONT8x8 = 4
FONT8x16= 5
;-------------------------------------------------------------
;Fonts einbinden
Font: FONT4x6,  32, 95  INTERN4x6
Font: FONT5x6,  32,158  INTERN5x6
Font: FONT6x8,  32,158  INTERN6x8
Font: FONT8x8,  32,158  INTERN8x8
Font: FONT8x16, 32,158  INTERN8x16
;-------------------------------------------------------------
Makro: 0                    ; Power-On/Reset Makro
     #QC EIN                ; Cursor sichtbar
     #FT FONT8x16           ; Terminalfont einstellen
     #UL 0,20,<EA2.BMP>     ; ELECTRONIC ASSEMBLY Logo
```

*) *Also on the Internet: http://www.lcd-module.de/deu/disk/disk240.zip*

INTEGRATED FONTS

5 character sets are integrated in each graphics unit as standard. Each character set can be used at its normal height or at up to 8 times this height. Independently of the height, the width can also be increased two to eight times.

Upper \ + Lower	$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	[	\	]	^	_

Font 1: 4x6

Upper \ + Lower	$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	{	|	}	~	⌂
$80 (dez: 128)	Ç	ü	é	â	ä	à	å	ç	ê	ë	è	ï	î	ì	Ä	Å
$90 (dez: 144)	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	¢	£	¥	ß	

Font 3: 6x8

Nr.	Zeichen höhe	Zeilen x Zeichen	Größe in Pixel	ASCII-Bereich	Frei def. ASCII-Codes	Bemerkung
1	2,2 mm	21 x 60	4 x 6	32 - 95	1..21	Microschrift
2	2,2 mm	21 x 48	5 x 6	32 - 158	1..21	Minischrift
3	3,1 mm	16 x 40	6 x 8	32 - 158	1..16	Normalschrift
4	3,1 mm	16 x 30	8 x 8	32 - 158	1..16	Fettschrift
5	6,3 mm	8 x 30	8 x 16	32 - 158	1..8	Großschrift

In addition, you can define up to 21 characters of your own, depending on the font. These characters are preserved until the supply voltage is switched off. (See the ESC E command.)

Each character can be positioned with **pixel accuracy**. Text and graphics can be combined as required. Several different font sizes can also be displayed together.

Each text can be output left justified, right justified or centered. 90° rotation (for vertical installation of the display) is also possible.

Macro programming permits the inclusion of up to 11 additional fonts and the complete redesign of the individual characters. A font editor on the EA DISKFONT6963 floppy disk allows you to create and program in any font you like with a size of up to 16x16 pixels.

Upper \ + Lower	$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	{	|	}	~	⌂
$80 (dez: 128)	Ç	ü	é	â	ä	à	å	ç	ê	ë	è	ï	î	ì	Ä	Å
$90 (dez: 144)	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	¢	£	¥	ß	

Font 5: 8x16

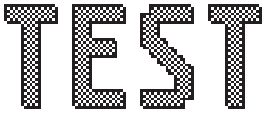
TIP: FONT EFFECTS

With large fonts, you can use the command ESC L TEXT mode (link, pattern) to produce interesting effects through overlaying (writing and offsetting a word several times).

*Original font 8x16 with ZOOM 3 at **position 0,0** with black pattern*

*"Outline font" produced by overlaying **(EXOR)** at **pos. 1,1***

*Overlaying **(EXOR)** of the "outline font" at **pos. 2,2**. results in an "outline font with fill"*



*Overlaying (**OR**) with 50% gray pattern of the "outline font" at **pos. 0,0**. results in a "font with pattern fill"*

ALL COMMANDS AT A GLANCE

Command table for the EA KIT160-7									
Command	**Codes**								**Note**
Commands for terminal operation									
Form feed FF (dec:12)	^L								Deletes the screen and sets the cursor at position (1,1)
Carriage return CR(13)	^M								Positions the cursor on the left at the beginning of the line
Line feed LF (dec:10)	^J								Positions the cursor in the line below the current one. If the cursor is in the last line, positions it in the 1st line
Cursor on/off	ESC	Q	C	n1					n1=0: cursor is not visible; n1=1: cursor flashes (inverse 6/10s)
Position cursor	ESC	O	n1	n2					n1=column; n2=line; upper left origin is (1,1)
Set terminal font	ESC	F	T	n1					n1=1: sets font no. n1 (1..16) for terminal operation
Text output commands									
Text mode	ESC	L	n1	pat					Mode n1: 1=set; 2=delete; 3=inverse 4=replace; 5=inverse replace; pat: pattern no. 0..7
Set font	ESC	F	n1	n2	n3				Sets font with the number n1 (1..16); n2=X- n3=Y-zoom factor (1x..8x)
Output string horizontally	ESC	Z	L Z R	x1	y1	Text ...	NUL		Outputs a string (...) at x1,y1. ´NUL´ ($00)=end of string; lines are separated by the character '|' ($7C, dec:124); 'L':= left justified at x1; 'Z':= centered at x1; 'R':= right justified at x1; y1 is always the upper edge of the string
Output string rotated by 90° (vertically)	ESC	Z	O M U	x1	y1	Text ...	NUL		Outputs a string (...) rotated by 90° at x1,y1; ´NUL´ ($00)=end; lines are separated by the character '|' ($7C, dec: 124); 'O':= top justified at y1; 'M':= vertically centered at y1; 'U':= bottom justified at y1 x1 is always the right edge of the string
Define character	ESC	E	n1	data ...					n1=character no.; data=number of bytes depending on current font
Drawing commands									
Graphics mode	ESC	V	n1						Sets the drawing mode for the commands 'Set point', 'Draw straight line', 'Rectangle', 'Rouded rectangle' and 'Fill area with pattern' n1: 1=set; 2=delete; 3=inverse; 4=replace; 5=inverse replace
Set point	ESC	P	x1	y1					Sets a pixel at position x1, y1
Draw straight line	ESC	G	x1	y1	x2	y2			Draws a straight line from x1,y1 to x2,y2
Continue straight line	ESC	W	x1	y1					Draws a straight line from the last end point to x1, y1
Rectangle commands									
Draw rectangle	ESC	R	R	x1	y1	x2	y2		Draws a rectangle (frame) from x1,y1 to x2,y2
Draw rounded rectangle	ESC	R	N	x1	y1	x2	y2		Draws a rectangle with rounded corners from x1,y1 to x2,y2
Delete area	ESC	R	L	x1	y1	x2	y2		Deletes an area from x1,y1 to x2,y2 (all pixels off)
Invert area	ESC	R	I	x1	y1	x2	y2		Inverts an area from x1,y1 to x2,y2 (inverts all pixels)
Fill area	ESC	R	S	x1	y1	x2	y2		Fills an area from x1,y1 to x2,y2 (all pixels on)
Fill area with pattern	ESC	R	M	x1	y1	x2	y2	pat	Fills an area from x1,y1 to x2,y2 with the pattern pat (0..7)
Draw box	ESC	R	O	x1	y1	x2	y2	pat	Draws a rectangle with the fill pattern pat (0..7); (always replace)
Draw rounded box	ESC	R	J	x1	y1	x2	y2	pat	Draws a rectangle with the fill pattern pat (0..7); (always replace)
Bitmap image commands									
Image from EEPROM	ESC	U	E	x1	y1	no			Loads an internal image with the number (0..255) from the EEPROM to x1,y1
Load image	ESC	U	L	x1	y1	data ...			Loads an image to x1,y1; see image structure for the data of the image
Send hard copy	ESC	H	x1	y1	x2	y2			Requests an image. Sends the width and height in pixels followed by the actual image data via RS232
Display commands (which apply to the whole display)									
Delete display	ESC	D	L						Deletes the contents of the display (all pixels off)
Invert display	ESC	D	I						Inverts the contents of the display (inverts all pixels)
Fill display	ESC	D	S						Fills the contents of the display (all pixels on)
Switch display off	ESC	D	A						Makes the contents of the display invisible, but they remain there and further commands are possible
Switch display on	ESC	D	E						Makes the contents of the display visible again
Clipboard display	ESC	D	C						Displays the contents of the clipboard. The display contents are no longer visible
Normal display	ESC	D	N						Displays the current image (normal mode). All outputs are visible again
Reset display	ESC	D	R						Resets and re-initializes the display controller
Macro commands									
Execute macro	ESC	M	N	n1					Calls the (normal) macro with the number n1 (max. 7 levels)
Execute touch macro	ESC	M	T	n1					Calls the touch macro with the number n1 (max. 7 levels)
Execute port macro	ESC	M	P	n1					Calls the port macro with the number n1 (max. 7 levels)
Macros autom. cyclical	ESC	M	A	n1	n2	n3			Processes macros n1..n2 automatically cyclically; n3=pause in 1/10s
Macros autom. ping-pong	ESC	M	J	n1	n2	n3			Processes macros n1..n2..n1 automatically (ping-pong); n3=pause in 1/10s

Bar graph commands												
Define bar graph	ESC	B	R L O U	no	x1	y1	x2	y2	sv	ev	pat	Defines a bar graph to the left (L), right (R), top (O) or bottom (U with the number no (1..16). x1,y1,x2,y2 define the rectangle enclosing the bar graph. sv,ev are the values for 0% and 100%. pat=pattern (0..7)
Draw bar graph			no	value								Sets the bar graph with the number no (1..16) to the new user 'value'
Clipboard commands (clipboard for image areas)												
Save display contents	ESC	C	B						Copies the entire contents of the display to the clipboard as an image area			
Save image			S	x1	y1	x2	y2		Copies the image area from x1, y1 to x2, y2 to the clipboard			
Restore display			R						Copies the image area on the clipboard back to the display			
Copy area			K	x1	y1				Copies the image area on the clipboard to position x1, y1 in the display			
Keyboard/touch panel commands												
Define touch key with horizontal label	ESC	T	H	f1	f2	Ret code	Form	Text ...	NUL	Groups touch fields f1 to f2 (diametrically opposite corner fields) together to form a touch key with the return value ´Ret. code´ (=1..255) (Ret. code=0 means the touch key is inactive). ´Form´: Draws touch key (=0 nothing; =1 delete; =2 with frame) ´Text´: Positions a string on the touch key (centered) using the current font; lines are separated by the character '|' ($7C, dec: 124); NUL character ($00) = end of string			
Define touch key with vertical label (rotated by 90°)			V									
(P)reset touch keys			P						Activates all touch keys in ascending order (fields with code 1..60)			
			R						Deactivates all touch keys (all fields with code 0)			
Touch key response			I	n1					n1=0: Touch key is not inverted when touched n1=1: Touch key is automatically inverted when touched			
			S	n1					n1=0: No tone sounds when (touch) key is touched n1=1: Tone sounds briefly when (touch) key is touched			
Invert touch key			M	n1					The touch key assigned the return code n1 is inverted manually			
Query key manually			W						Sends the currently depressed (touch) key at the RS-232/RS-422 interface			
Key query on/off			A	n1					The keyboard query is n1=0:deactivated; n1=1:activated, keystrokes are sent automatically; n1=2:activated, keystrokes are not sent (query with ESC T W)			
Menu/pop-up commands												
Define menu with horizontal items	ESC	N	H	x1	y1	no	Text ...	NUL	Draws a menu from the corner x1,y1 (horizontal menu = upper left corner; vertical menu = upper right corner) using the current font. no:= currently inverted item (e.g.: 1 = 1st item) Text:= string with the menu items. The items are separated by the character '|' ($7C,dec:124), e.g. "Item1|Item2|Item3" The background of the menu is automatically saved to the clipboard. If a menu is already defined, it is automatically canceled and removed			
Define menu with vertical items (rotated by 90°)			V									
Invert menu box			I						Inverts the entire menu box. Useful for negative display			
Next item			N						Inverts the next item or remains at the end			
Previous item			P						Inverts the previous item or remains at the beginning			
Menu end/send			S						Removes the menu from the display and replaces it with the clipboard contents. The current item is sent as a number (1..n) (0=no menu displayed)			
Menu end/macro			M	no					Removes the menu from the display and replaces it with the contents of the clipboard. Macro 'no' is called for item 1; macro no+1 for item 2, and so on			
Menu end/cancel			A						Removes the menu from the display and replaces it with the contents of the clipboard			
Control/definition commands												
Automatic flashing area (cursor function)	ESC	Q	D	x1	y1	x2	y2		Defines a flashing area from x1,y1 to x2,y2; activates the flashing function			
			Z	n1					Sets the flashing time n1= 1..15 in 1/10s; 0=deactivates the flashing function			
			M	I					Inverse mode (flashing area is inverted); activates the flashing function			
				pat					Clipboard mode pat=pattern (0..7) of the block cursor; activates flashing			
			C	n1					Automatically flashing area as cursor for terminal operation n1=0: deactivates flashing function; n1=1: activates flashing function (inverse, 6/10s)			
Select/deselect	ESC	K	S	add					Activates the kit with the address n1 (n1=255: all)			
			D	add					Deactivates the kit with the address n1 (n1=255: all)			
			A	add					Assigns a new address (add) (in the power-on macro, for example)			
Wait (pause)	ESC	X	n1						Wait n1 tenths of a second before the next command is executed			
Buzzer on/off	ESC	J	n1						n1=0:tone off; n1=1:tone on; n1=2..255:for n1 1/10s long on			
Send bytes	ESC	S	num	data ...					Sends num (1..255; 0=256) bytes at the RS-232/RS-422 interface; data ... = num bytes (e.g. control of an external serial printer)			
Port commands												
Write output port	ESC	Y	W	n1	n2				n1=0: Sets all 8 output ports in accordance with n2 (=8-bit binary value) n1=1..8: Resets (n2=0), sets (n2=1) or inverts (n2=2) output port n1			
Read input port			R	n1					n1=0: Reads in all 8 input ports as 8-bit binary value n1=1..8: Reads in input port <n1> (1=high level=5V, 0=low level=0V)			
Port scan on/off			A	n1					Deactivates (n1=0) or activates (n1=1) automatic scanning of the input port			
Input port inverse			I	n1					Evaluates the input port (n1=0: normal; n1=1: inverted)			
Switch backlight on/off			L	n1					CFL/LED backlight n1=0: off; n1=1: on; n1=2: invert; n1=3..255: backlight for n1/10 sec. switched on and then automat. turn off			

PARAMETERS

The graphics kit can be programmed by means of various integrated commands. Each command begins with ESC followed by one or two command letters and then parameters. All the commands and their parameters, such as coordinates and other transfer values, are always expected as bytes. No separating characters, such spaces or commas, must be used between them. The commands require **no final byte** such as a carriage return (except for the string $00).

A..Z, L/R/O/U ..... All commands are transferred as ASCII characters.
Example: G= 71 (dec.) = $47 initiates the straight-line command.

x1, x2, y1, y2 ..... Coordinates are transferred with 1 byte.
Example: x1= 10 (dec.) = $0A

ESC ..... 1 byte: 27(dec.) = $1B

n1,n2,no,sv,ev,value,pat,ret, frm,data ..... Numerical values are transferred with 1 byte.
Example: n1=15(dec.) = $0F

PROGRAMMING EXAMPLE

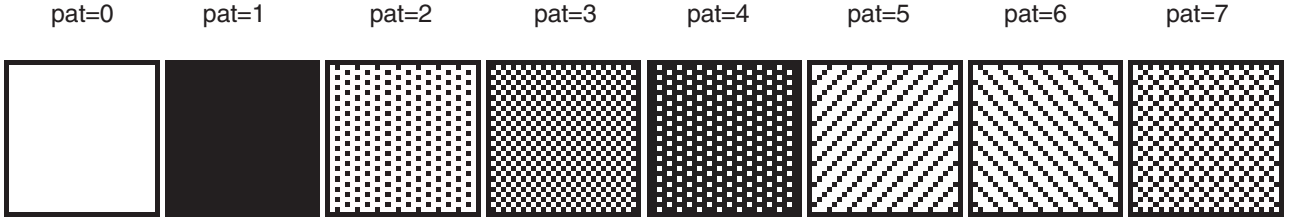
The following table shows an example in which the string "Test" is output left justified at coordinate 7,3.

Example	Codes to be output									
In ASCII	ESC	Z	L	BEL	ETX	T	e	s	t	NUL
In hex	$1B	$5A	$4C	$07	$03	$54	$65	$73	$74	$00
In decimal	27	90	76	7	3	84	101	115	116	0
For Turbo Pascal	write(aux, chr(27), 'Z', 'L', chr(7), chr(3), 'Test', chr(0));									
For C	fprintf(stdaux, "\x1BZL%c%c%s\x00", 7, 3, "Test");									
For Q Basic	OPEN "COM1:9600,N,8,1,BIN" FOR RANDOM AS #1 PRINT #1,CHR$(27)+"ZL"+CHR$(7)+CHR$(3)+"Test"+CHR$(0)									

PATTERN

A pattern type (mst = 0..7) can be set as a parameter with some commands. In this way, rectangular areas, bar graphs and even texts can be linked to different patterns and displayed.

The following fill patterns are available:

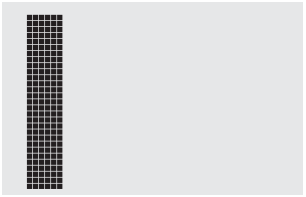


DESCRIPTIONS OF THE VARIOUS GRAPHICS FUNCTIONS

On the following pages you will find detailed descriptions of all of the functions in alphabetical order. In each case, an enlarged section of the image, 50x32 pixels in size, is shown as a hard copy example, indicating the contents of the display after the command is executed. The bytes to be transferred are shown as hex values in the examples.

ESC B L/R/O/U no x1 y1 x2 y2 sv ev pat — Define bar graph

Up to 16 bar graphs (**no**=1..16) can be defined. These can extend to the left (**L**), right (**R**), up (**O**) or down (**U**). At its full extent, the bar graph occupies an area from **x1**,**y1** to **x2**,**y2**. It is scaled with the start value (no extension) **sv** (=0..254) and the end value (full extension) **ev** (=0..254). The bar graph is always drawn in inverse mode with the pattern (**pat**): The background is thus always retained. (Note: When this command is executed, it defines the bar graph but does not display it).



Example: $1B $42 $4F $01 $04 $02 $09 $1E $04 $14 $01

Bar graph no. 1, which extends upwards, is defined. When it is fully extended, it takes up an area from 4,2 to 9,30. The start and end values correspond to a 4..20 mA display. (The diagram shows the bar graph fully extended, as represented with $42 $01 $14.)

ESC B no value — Draw bar graph

The bar graph with the number **n1** (1..16) is set to the new value (sv <= **value** <= ev). If **value** > ev, the end value (ev) is displayed. The bar graph must be defined first (see above).

Example: $1B $42 $01 $0A

Bar graph no. 1 defined in the above example is set to a value of 10.

ESC C B — Save contents of display to clipboard

Copies the entire contents of the display to the clipboard.

Example: $1B $43 $42

Saves the entire contents of the display to the clipboard so that the screen can subsequently be restored. The contents of the display do not change.

ESC C S x1 y1 x2 y2 — Save area to clipboard

Copies an area from the upper left corner (**x1,y1)** to the lower right corner (**x2,y2**) to the clipboard.

Example: $1B $43 $53 $00 $00 $17 $1B

Saves the area from 0,0 to 23,27 so that the screen can subsequently be restored. The contents of the display do not change.

ESC C R — Restore area

Copies the the area last saved from the clipboard back to the display. Destination: the original coordinates.

Example: $1B $43 $52

Restores the area last saved.

ESC C K x1 y1 — Copy area from clipboard

Copies the area last saved on the clipboard to a new position (**x1,y1**) on the display.

Example: $1B $43 $4B $0A $20

Copies the area last saved to the point 10,32.

ESC D L/I/S — Change contents of display

The entire contents of the display are deleted (**L** - white), inverted (**I**) or filled (**S** - black).

Example: $1B $44 $49

Inverts the entire contents of the display.

ESC D A/E — Switch display on/off

Switches the contents of the display off (**A** - not visible) or on (**E** - visible). Outputs are still possible when it is switched off.

Example: $1B $44 $41

The contents of the display are no longer visible after this command.

ESC D N/C — Display normal/clipboard contents

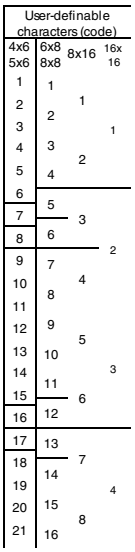
The normal contents (**N**) or the clipboard contents (**C**) appear on the display. Concealed drawing is possible with this command. Example: The current contents of the display are saved to the clipboard with **ESC C B**, and the contents of the clipboard are then displayed with **ESC D C**. All subsequent outputs to the display will be invisible until the command **ESC D N** is entered, at which point the current contents will become visible again.

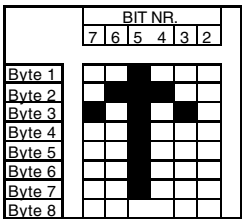
Example: $1B $44 $49

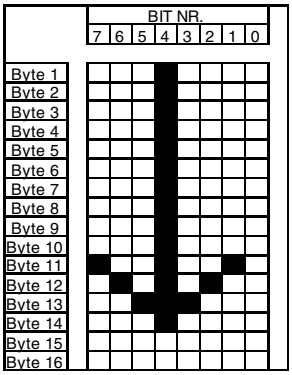
The display now shows the contents of the clipboard (only complete images are recognizable).

ESC E n1 data — Define character

You can define up to 21 characters yourself (depending on the font size). These characters then have the ASCII codes 1 to max. 21 and remain in an invisible screen RAM 128 bytes in size until the supply voltage is switched off. In the case of a 4x6 font, up to 21 characters can be defined, whereas only 8 characters can be defined for an 8x16 font. Please note that if you want to define several characters in different fonts, you must bear in mind that a character with code 1 of the 8x16 font, for example, requires the same amount of RAM as the characters with the codes 1 to 3 in the 4x6 font (see the adjacent table).







Example 1:
$1B $45 $01
$20 $70 $A8 $20 $20 $20 $20 $00
Defines an arrow pointing upward for ASCII no. 1 using the 6x8 character set.

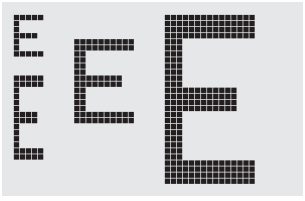
Example 2:
$1B $45 $02
$10 $10 $10 $10 $10 $10 $10 $10 $10 $10 $92 $54 $38 $10 $00 $00
Defines an arrow pointing downward for ASCII no. 2 using the 8x16 character set.

ESC F n1 n2 n3 — Set font

Sets the font with the number **n1** (1=4x6 uppercase letters only; 2=6x8; 3=8x16). In addition, an enlargement factor (1..8 times) is set for the width (**n2**) and height (**n3**) separately.

Example: $1B $46 $02 $03 $04

The 6x8 with 3 times the width and 4 times the height is set with immediate effect. In the adjacent figure, the character 'E' is shown in the 6x8 font and with various enlargement factors.



ESC F T n1 — Set terminal font

Sets the font with the number **n1** for terminal operation. The font for the terminal is always used without zoom and in REPLACE mode.

Example: $1B $46 $54 $03

The 6x8 font is set as the terminal font with immediate effect.

ESC G x1 y1 x2 y2 — Draw straight line

A straight line is drawn from **x1,y1** to **x2,y2** taking into account the graphics mode set 'V' (set/delete/inverse).

Example: $1B $47 $03 $14 $28 $06

A straight line is drawn from 3,20 to 50,6.

ESC H x1 y1 x2 y2 — Create hard copy of display contents

Requests the area from the upper left corner (**x1,y1**) to the lower right corner (**x2,y2**). The graphics chip then immediately sends the width and height of the image section followed by the image data. See the upload image command ('U') for the structure of the image data.

Example: $1B $48 $00 $00 $1F $0F

The upper left part of the screen (32 x 16 pixels) is sent via RS-232.

ESC J n1 — Switch tone on/off manually

Switches the tone off (**n1**=0), on for an undefined period (**n1**=1) or on for n1/10 seconds (**n1**=2..255). (This only applies to versions with EA KIT240-7CTP and EA KIT240-7LEDTP touch panels.)

Example: $1B $4A $0A

The tone sounds for 1 second after this command.

ESC K A add — Assign address

Assigns an address to the KIT240 (**add**=0..254). The best place for this command is in the power-on macro.

Example: $1B $4B $41 $01

The KIT240 is assigned the address $01 with immediate effect.

ESC K S/Dadd — (De)select KIT160

Selects (**S**) or deselects (**D**) the KIT240 with the address **add** (0..254); the address 255=$FF is a master address for all KIT160 units.

Example: $1B $4B $44 $01

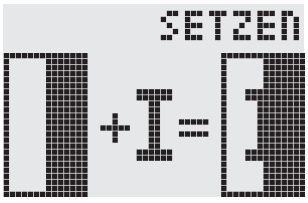
All commands for the KIT160 with the address $01 are ignored with immediate effect.

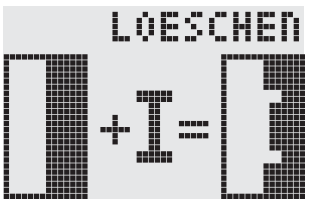
ESC L n1 pat — Set text mode

Sets the link mode (**n1**) and pattern (**pat)** for the string output text function (**ESC Z**).

Example: $1B $4C $03 $03

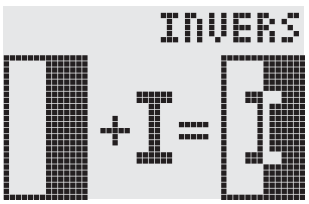
Sets the link mode for all subsequent text functions to gray characters (pattern 3 = 50% gray) inverted with the background.

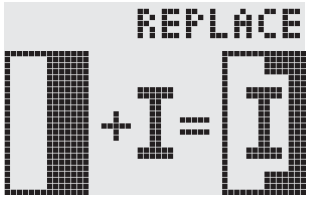


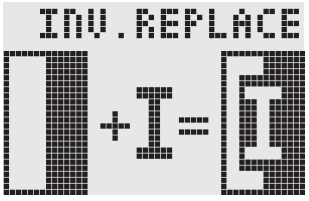


Link mode n1:

1 = set: black pixels irrespective of the previous value (OR)
2 = delete: white pixels irrespective of the previous value
3 = inverse: changes black pixels to white pixels and vice versa (EXOR)
4 = replace: deletes the background and sets black pixels
5 = inverse replace: fills the background and sets white pixels







ESC M N/T/P n1 — Call macro

Calls the normal macro (**N**), touch macro (**T**) or port macro (**P**) with the number **n1** (0..255).

Example: $1B $4D $4E $0F
The (normal) macro with the number 15 is executed.

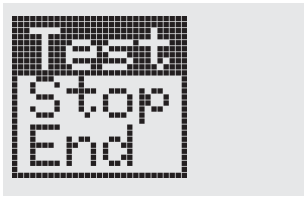
ESC M A/J n1 n2 n3 — Execute macros automatically

Calls the normal macros with the numbers **n1** to **n2** automatically every **n3**/10 seconds. **A**=cyclical call (e.g. 1,2,3,4,1,2,3,4, etc.); **J**=ping-pong call (e.g. 1,2,3,4,3,2,1,2,3,4, etc.).
Automatic execution is terminated:
- When a character is received from the RS-232 interface
- When a touch automatically executes a touch macro
- When an input change executes a port macro

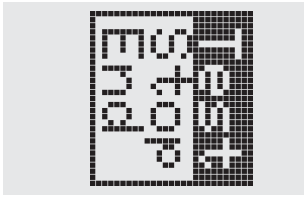
Example: $1B $4D $41 $01 $03 $05
The macros with the numbers 1, 2 and 3 are executed automatically with a break of 1/2 second.

ESC N H/V x1 y1 no Text... NUL — Display menu

Defines and displays a menu with the current font. The background of the menu box is automatically saved on the clipboard **(the previous contents of the clipboard are lost)**: **H**=horizontal menu at **x1,y1** (upper left corner) or **V**=vertical menu (rotated 90°) at **x1,y1** (upper right corner). **n1**=currently inverted item; **Text...**=string containing the items. The individual items are separated by the character '|' (=$7C). The string must be terminated with **NUL**= $00.



Example 1 - Horizontal menu:
$1B $4E $48 $02 $02 $01
$54 $65 $73 $74 $7C $53 $74 $6F $70 $7C $45 $6E $64 $00
Defines a horizontal menu containing the items "Test", "Stop" and "End" at position 2,2. The 1st item is inverted.



Example 2 - Vertical menu:
$1B $4E $56 $28 $01 $01
$54 $65 $73 $74 $7C $53 $74 $6F $70 $7C $45 $6E $64 $00
Defines a vertical menu containing the items "Test", "Stop" and "End" at position 40,1. The 1st item is inverted.

ESC N N/P — Next/previous menu item

Inverts the next (**N**) or previous (**P**) menu item. If the last/first item is already inverted, the command is ignored.
Example: $1B $4E $4E
The next menu item is inverted.

ESC N I — Invert menu box

Inverts the entire menu box.
Example: $1B $4E $49

ESC N S — Terminate and send menu

Removes the menu from the display and replaces it with the background from the clipboard. The currently selected item is sent as a number (1..max. item) via the RS 232 interface.
Example: $1B $4E $53

ESC N M n1 — Terminate menu and call macro

Removes the menu from the display and replaces it with the background from the clipboard. If item 1 is selected, the (normal) macro with the number **n1** is called, for item 2 the macro **n1**+1 etc.
Example: $1B $4E $4D $0A

ESC N A — Cancel menu

Removes the menu from the display and replaces it with the background from the clipboard.
Example: $1B $4E $41

ESC O n1 n2 — Position cursor

Sets the cursor to column **n1** and row **n2** for terminal operation. The origin in the upper left corner is 1,1.

Example: $1B $4F $03 $05

Sets the cursor to the 3rd column in row 5.

ESC P x1 y1 — Set dot

Sets a pixel at **x1,y1** taking into account the graphics set mode 'ESC V' (set/delete/invert).

Example: $50 $11 $0D

Sets the pixel at 17,13.

ESC Q C n1 — Cursor on/off

n1=1: Switches the cursor on; it flashes at the current position on the terminal.
n1=0: Switches the cursor off.

Example: $1B $51 $43 $01

Switches the cursor off.

ESC Q D x1 y1 x2 y2 — Define flashing area

Defines the area from the upper left corner (**x1,y1**) to the lower right corner (**x2,y2**) as an automatically flashing area. The flashing function is started at the same time. This deactivates the terminal cursor.

Example: $1B $51 $44 $00 $0F $07 $10

Defines the flashing area from 0,15 to 7,16.

ESC Q Z n1 — Set flashing time

Sets the flashing time to **n1** (=1..15) tenths of a second. When **n1**= 0, the flashing function is deactivated and the original screen restored.

Example: $1B $51 $5A $03

Sets the flashing time to 0.3 seconds.

ESC Q M I — Inverse flashing mode

Automatically inverts the defined flashing area cyclically with the set flashing time. The flashing function is started at the same time.

Example: $1B $51 $49

Sets the inverse flashing mode.

ESC Q M pat — Block cursor flashing mode

Saves the defined flashing area on the clipboard **(the previous contents of the clipboard are lost)**. There is a cyclical changeover between the original area and the pattern **pat** (=0..7) on the basis of the set flashing time. In this way, for example, a block cursor can be simulated (pat=1 black) or a flashing word displayed (pat=0 white). The flashing function is started at the same time.

Example: $1B $51 $43 $00

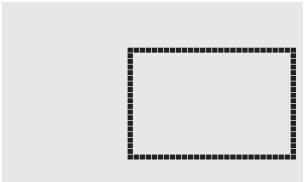
The block cursor flashing mode is set with the pattern white. As a result, the set area flashes on a white background.

ESC R R x1 y1 x2 y2 — Draw rectangle

Draws a rectangle from the upper left corner (**x1,y1**) to the lower right corner (**x2,y2**) taking into account the set graphics mode 'V' (set/delete/inverse). The contents of the rectangle are not changed. See 'ESC R O' (Draw box).

Example: $1B $52 $52 $15 $08 $30 $25

Draws a rectangle from 21,8 to 48,37.

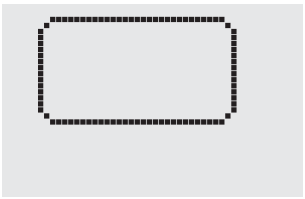


ESC R N x1 y1 x2 y2 — Draw rounded rectangle

Draws a rectangle with rounded corners from the upper left corner (**x1,y1**) to the lower right corner (**x2,y2**) taking into account the set graphics mode 'V' (set/delete/inverse). The contents of the rounded rectangle are not changed. See 'ESC R J' (Draw rounded box).

Example: $1B $52 $4E $06 $02 $26 $13

Draws a rounded rectangle from 6,2 to 38,19.

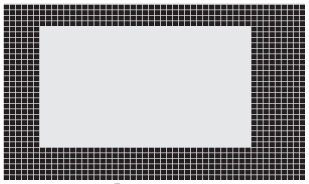


ESC R L x1 y1 x2 y2 — Delete area

Deletes the area from the upper left corner (**x1,y1**) to the lower right corner (**x2,y2**).

Example: $1B $44 $53 $1B $52 $4C $06 $04 $28 $19

The display is filled with **ESC D S** and then deleted from 6,4 to 40,25.

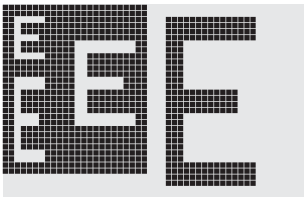


ESC R I x1 y1 x2 y2 — Invert area

Inverts the area from the upper left corner (**x1,y1**) to the lower right corner (**x2,y2**) (black pixels turn white and vice versa).

Example: $1B $52 $49 $00 $00 $17 $1B

Inverts the area from 0,0 to 23,27 with the display contents from the "Set font" example.

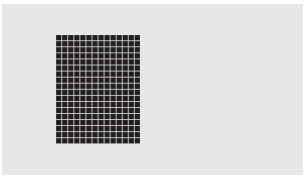


ESC R S x1 y1 x2 y2 — Fill area

Fills the area from the upper left corner (**x1,y1**) to the lower right corner (**x2,y2**) (sets the pixels to black).

Example: $1B $52 $53 $09 $05 $16 $16

Sets the area from 9,5 to 22,22 black.

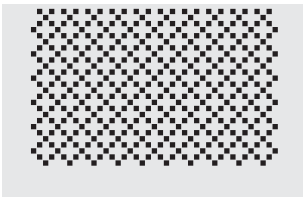


ESC R M x1 y1 x2 y2 pat — Fill area with pattern

Fills a rectangular area from the upper left corner (**x1,y1**) to the lower right corner (**x2,y2**) with the pattern **pat** taking into account the set graphics mode "ESC V" (set/delete/invert/replace/inverse replace).

Example: $1B $52 $4D $05 $01 $2D $1A $07

Fills the area with the pattern 7=45°cross from 5,1 to 45,26.

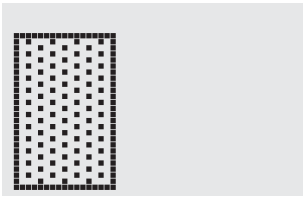


ESC R O x1 y1 x2 y2 pat — Draw box

Draws a rectangle from the upper left corner (**x1,y1**) to the lower right corner (**x2,y2**) with the pattern **pat**. The background of the box is deleted. See 'ESC R R' (Draw rectangle).

Example: $1B $52 $4F $02 $05 $12 $1E $02

Draws a box from 2,5 to 18,30 with the pattern 2=25%gray.

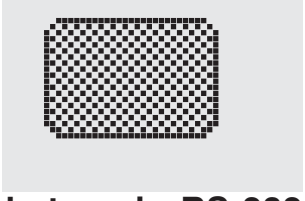


ESC R J x1 y1 x2 y2 pat — Draw rounded box

Draws a rectangle with rounded corners from the upper left corner (**x1,y1**) to the lower right corner (**x2,y2**) with the pattern **pat**. The background is deleted. See 'ESC R N' (Draw rounded rectangle).

Example: $1B $52 $4A $07 $03 $23 $16 $03

Draws a rounded box from 7,3 to 35,22 with the pattern 3=50%gray.



ESC S num data... — Send bytes via RS-232

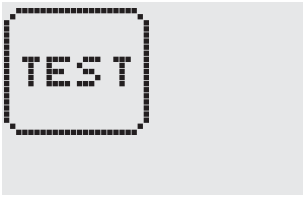
Outputs the next **num** (1..255, 0=256) bytes at the serial interface.

Example: $1B $53 $04 $54 $45 $53 $54

Transmits the word 'TEST' via the RS-232C interface.

ESC T H/Vf1 f2 ret frm text... NUL — Define touch key

Defines a touch key and labels it with the current font. **H**=horizontal or **V**=vertical labeling (rotated 90°). Several touch fields can be grouped together to form a single touch key (**f1**=upper left touch field; **f2**=lower right touch field of the new touch key). This touch key is assigned a return code with **ret** (1..255). When the touch key is touched, the touch macro with the number **ret** is called or, if no touch macro is defined, this return code is sent via the RS232. You use **frm** to define the format of the touch key (frm=0: don't draw anything; frm=1: delete touch key; frm=2: delete touch key and draw with frame). **text...**=string with the label (which is always centered on the touch key). The label can also have more than one line; in this case, the lines are separated by the character '|' (=$7C). The string must be concluded with **NUL**= $00. See example on page 3.



Example 1: Horizontal touch key:
$1B $54 $48 $01 $01 $41 $02 $54 $45 $53 $54 $00
Defines a horizontal touch key (field no. 1 only) with the return code 65='A'. The touch key is drawn with a frame and labeled with the word 'TEST'.

Example 2: Vertical touch key:
$1B $54 $56 $02 $02 $42 $02 $54 $45 $53 $54 $00
Defines a vertical touch key (touch field no. 2 only) with the return code 66='B'. The touch key is drawn with a frame and labeled with the word 'TEST'.

ESC T P/R — Preset/reset touch fields

Assigns **P** (=ascending return code: 1..60) or **R** (=reset all touch fields) to all 60 touch fields. In the latter case all touch fields receive the return code 0 (i.e. they are deactivated).

Example: $1B $54 $52

All touch fields are deactivated by this command and no longer recognized.

ESC T I/S n1 — Touch key response

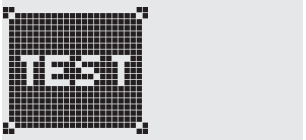
These commands set the automatic response of the touch panel to touching. Both responses can be activated simultaneously.
I=automatic inversion when the touch key is touched (**n1**=0: off or **n1**=1: on)
S=automatic signal tone when the touch key is touched (**n1**=0: off or **n1**=1: on)

Example: $1B $54 $49 $01

After this command the tone sounds when a touch key is touched.

ESC T M ret — Invert touch key manually

This command manually inverts the touch key with the return code **ret**.



Example: $1B $54 $4D $41

Inverts the touch key from the above example with the return code 65='A'.

ESC T A n1 — (Touch) key query on/off

This command sets the (touch) key query:
n1=0: Switches the key query off - no touch macros or manual key query possible.
n1=1: Activates the key query - keystrokes trigger touch macros or are sent via RS232.
n1=2: Activates the key query - keystrokes trigger touch macros; must be queried manually.

Example: $1B $54 $41 $02

Activates the (touch) key query. The keystrokes are not sent automatically via RS232; they have to be requested manually by means of the command **ESC T W**.

ESC T W — Query touch key manually

Sends the return code of the currently depressed touch key at the RS232.

Example: $1B $54 $57

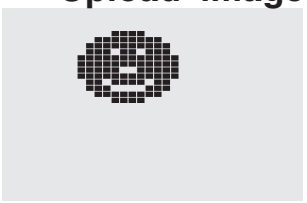
ESC U E x1 y1 n1 — Load image from EEPROM

Displays the image saved in the EEPROM with the number **n1** (0..255) at position **x1,y1**.

Example: $1B $55 $45 $02 $03 $0E

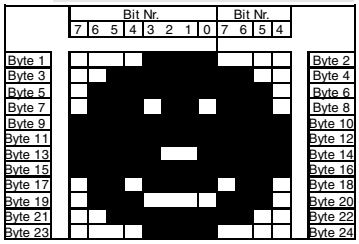
Displays image number 14 from the EEPROM at position 2,3.

ESC U L x1 y1 data... — Upload image



Displays an image at position **x1,y1**.

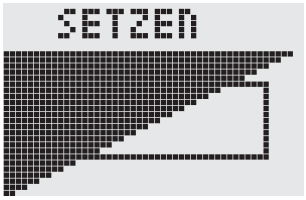
data..: - 1 byte for the image width in pixels
- 1 byte for the image height in pixels
- Image data: number = ((width+7) / 8) * height bytes.
1 byte stands for 8 horizontal pixels on the screen; 0=white, 1=black;
MSB: left, LSB: right; the image is stored from the top down.
The BMP2BLH.EXE program on the EA DISK240 floppy disk available as an accessory creates the image data, including the width and height, from monochrome Windows bitmap graphics (*.BMP).

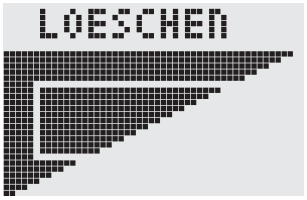


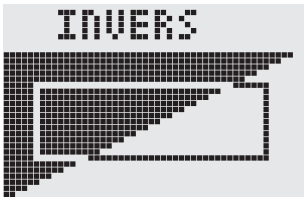
Example:
$1B $55 $4C $09 $04 $0C $0C
$0F $00 $3F $C0 $7F $E0 $76 $E0 $FF $F0 $FF $F0
$F1 $F0 $FF $F0 $6F $60 $70 $E0 $3F $C0 $0F $00

Loads the adjacent image at position 9,4.

ESC V n1 — Set graphics mode







Sets the link mode **n1** for the following graphics functions: ESC P (Set point), ESC G (Draw straight line), ESC W (Continue straight line), ESC R R (Draw rectangle), ESC R N (Draw rounded rectangle), ESC R M (Fill area with pattern).

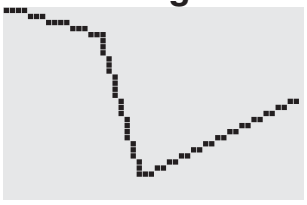
Example: $1B $56 $03
Sets the link mode to inverse.

By way of example, a rectangle is drawn alongside with the link modes set, delete and inverse on an existing background.

Link mode n1:

1 = set: black pixels irrespective of the previous value (OR)
2 = delete: white pixels irrespective of the previous value
3 = inverse: changes black pixels to white pixels and vice versa (EXOR)
4 = replace: deletes the background and sets black pixels; only area with fill pattern 'pat'
5 = inverse replace: fills the background and sets white pixels; only area with fill pattern 'pat'

ESC W x1 y1 — Continue straight line



Continues a straight line from the last end or point drawn to **x1,y1** taking into account the set graphics mode 'V'.

Example:
$1B $47 $00 $00 $10 $04
$1B $57 $16 $1B
$1B $57 $30 $0F
A straight line is drawn from 0,0 to 16,4.It is then continued to 22,27 and to 48,15.

ESC X n1 — Wait/pause

This command suspends the KIT240 for **n1**/10 seconds.

Example: $1B $58 $0A
After this command the KIT240 waits for a second before the next command is processed.

ESC Y R n1 — Read input port

Reads in the input port (**n1**=1..8 = IN1..IN8). When **n1**=0, all the inputs are read in as 8-bit binary values (MSB:IN8...In1:LSB); see application on page 5. Important: The optocouplers invert the input logic (input open: 1). The command "ESC Y I 1" puts this right (input open: 0).

Example: $1B $59 $52 $03
Reads in port IN3. The result is sent via RS232.

ESC Y W n1 n2 — Write output port

Changes the output port (**n1**=1..8 = OUT1..OUT8) to the value **n2** (0=low level; 1=high level; 2=invert port). When **n1**=0, all the outputs are output as a binary value **n2** (MSB:OUT8...OUT1:LSB); see application on page 5.

Example: $1B $59 $57 $02 $01
Switches the output port OUT2 to high level.

ESC Y A n1 — Automatic port query on/off

Each change at the input port (8-bit binary value IN8..IN1) can call a port macro (0..255). This command activates (**n1**=1) or deactivates (**n1**=0) the automatic port query. After power-on, the current port status is read and the associated port macro executed immediately.

Example: $1B $59 $41 $01
Activates the automatic port query and executes the associated port macro.

ESC Y I n1 — Invert input port

This command allows the logic of the input port to be inverted (**n1**=0 for normal or **n1**=1 for inverse). This is useful with the optocoupler inputs, for example.

Example: $1B $59 $49 $01
Inverts the input port logic.

ESC Z L/Z/R x1 y1 text... NUL — Horizontal string

Writes the string **text...** left justified (**L**), centered (**Z**) or right justified (**R**) at position **x1** taking into account the set text mode (**ESC L**). Multi-line text can also be output, with the lines separated by the character '|' (=$7C). The string must be concluded with **NUL**= $00. Position **y1** is the upper edge of the 1st line.

Example 1: Writes the text "Left|Ok" left justified at 0,0.
$1B $5A $4C $00 $00 $4C $65 $66 $74 $7C $4F $6B $00

Example 2: Writes the text "Center|Ok" centered at 25,0.
$1B $5A $5A $19 $00 $43 $65 $6E $74 $65 $72 $7C $4F $6B $00

Example 3: Writes the text "Right|Ok" right justified at 49,0.
$1B $5A $52 $31 $00 $52 $69 $67 $68 $74 $7C $4F $6B $00

ESC Z O/M/U x1 y1 text... NUL — Vertical string

Writes the string **text...** rotated by 90° degrees top justified (**O**), vertically centered (**M**) or bottom justified (**U**) at position **y1** taking into account the text mode (**ESC L**). Multi-line text can also be output, with the lines separated by the character '|' (=$7C). The string must be concluded with **NUL**= $00. Position **x1** is the right edge of the 1st line.

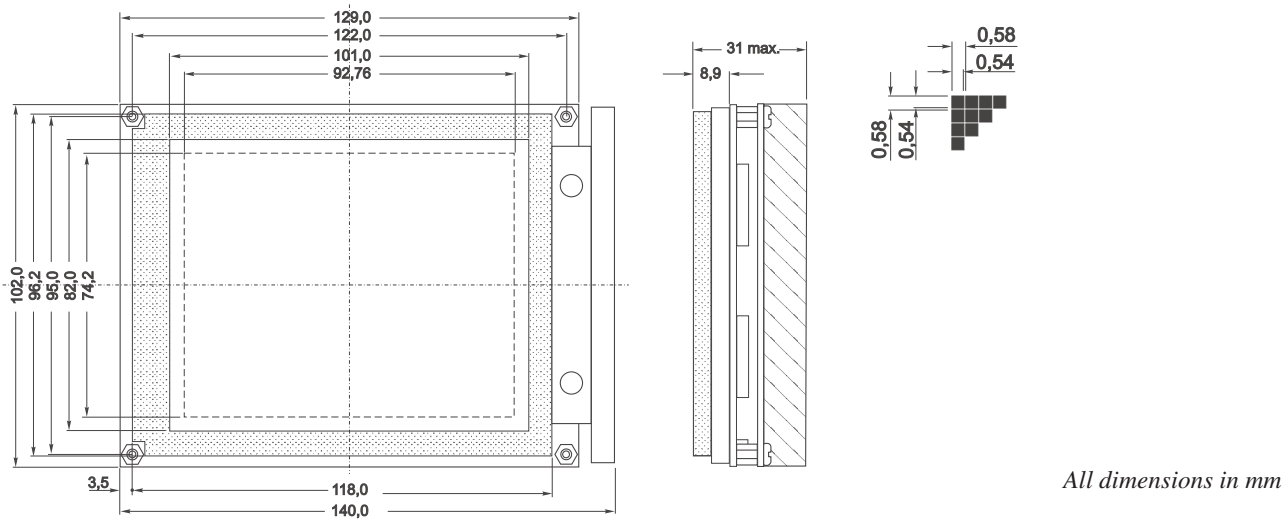
Example 1: Writes the text "Top|Ok" top justified at 49,0.
$1B $5A $4F $31 $00 $54 $6F $70 $7C $4F $6B $00

Example 2: Writes the text "Mid|Ok" vertically centered at 49,15.
$1B $5A $4D $31 $0F $4D $69 $64 $7C $4F $6B $00

Example 3: Writes the text "Bot|Ok" bottom justified at 49,31.
$1B $5A $55 $31 $1F $42 $6F $74 $7C $4F $6B $00

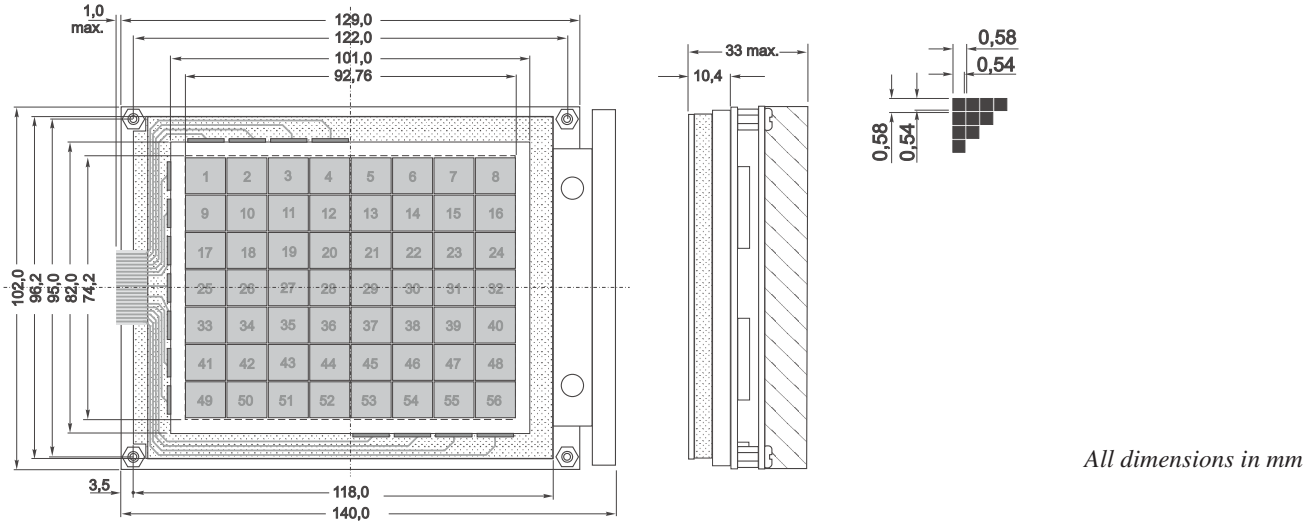
EA KIT160-7

DIMENSIONSWITHOUTTOUCH PANEL



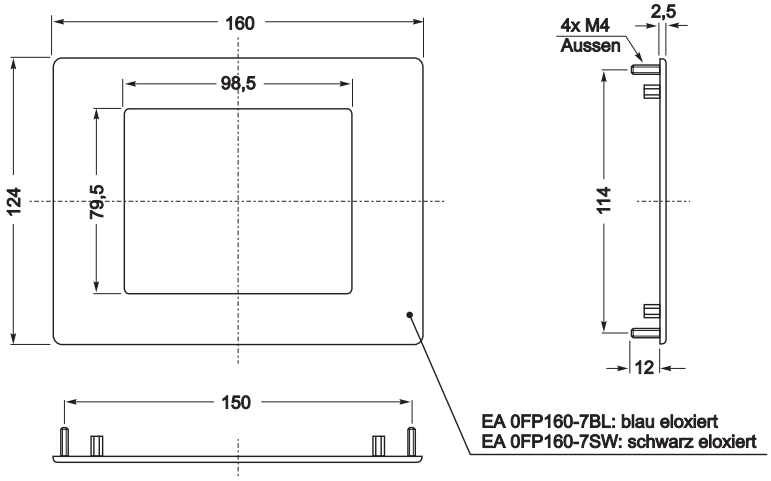
All dimensions in mm

DIMENSIONSWITHTOUCH PANEL

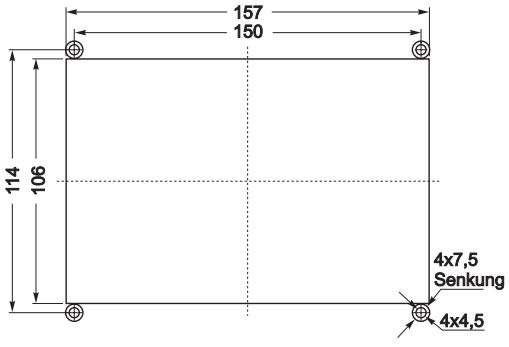


All dimensions in mm

EA 0FP160-7 FRONT PANEL



PANEL CUTOUT



All dimensions in mm

ZEPPELINSTRASSE 19 · D-82205 GILCHING
PHONE +49-8105-778090 · FAX +49-8105-778099 · http://www.lcd-module.de

ELECTRONIC ASSEMBLY GMBH