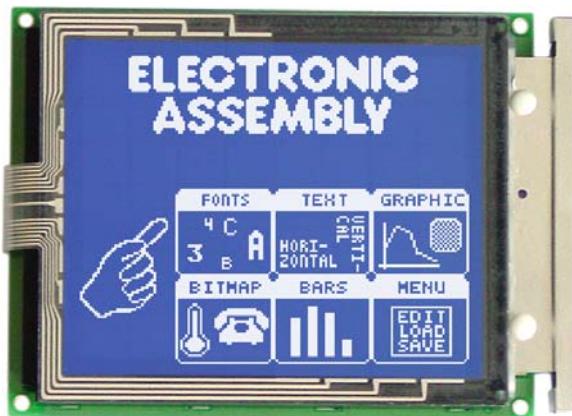


BEDIENEINHEIT MIT FONTS, GRAFIKBEFEHLEN UND MAKROS

5,1"
Touch Panel
integriert



EA KIT160-7LWTP
Abmessungen 140x102mm

TECHNISCHE DATEN

- * 160x128 PIXEL MIT CFL-BELEUCHTUNG BLAU NEGATIV
- * AUCH MIT LANGLEBIGER LED-BELEUCHTUNG WEISS/BLAU
- * INTEGRIERTESTOUCH PANEL MIT 8x7 FELDERN (ENTSPIEGELT, KRATZFEST)
- * FONT ZOOM VON ca. 3mm ÜBER ca. 5mm BIS ZU ca. 50mm
- * VERSORGUNGSSPANNUNG 5V/500mA(CFL)/300mA(LW) ODER 9..35V OPTIONAL
- * RS-232 ODER OPTIONAL RS-422 MIT BAUDRÄTEN 1200..115200 BD
- * **PIXELGENAUE** POSITIONIERUNG BEI ALLEN FUNKTIONEN
- * PROGRAMMIERUNG ÜBER HOCHSPRACHENÄHNLICHE BEFEHLE:
- * GERADE, PUNKT, BEREICH, UND/ODER/EXOR, BARGRAPH...
- * BIS ZU 256 MAKROS PROGRAMMIERBAR (EEPROM ONBOARD)
- * TEXT UND GRAFIK MISCHEN
- * 4 CLIPBOARD FUNKTIONEN, PULL-DOWN MENÜS
- * 8 DIGITALE EINGÄNGE UND 8 DIGITALE AUSGÄNGE
- * BELEUCHTUNG PER SOFTWARE SCHALTBAR

OPTIONEN/ZUBEHÖR

VERSORGUNG +9..35V = STATT +5V =	EA OPT-9/35V
RS-422 SCHNITTSTELLE STATT RS-232	EA OPT-RS4224
OPTOKOPPLER ONBOARD FÜR 8 EIN- UND 8 AUSGÄNGE	EA OPT-OPTO16
ALUMINIUM EINBAUBLENDE, MATT-SCHWARZ ELOXIERT	EA 0FP160-7SW
ALUMINIUM EINBAUBLENDE, BLAU ELOXIERT	EA 0FP160-7BL
KABEL (1,5m) FÜR ANSCHLUSS AN 9-POL. SUB-D (RS-232 FEMALE)	EA KV24-9B
DISKETTE FÜR MAKROPROGRAMMIERUNG (PC-DOS/WIN)	EA DISK240

BESTELLBEZEICHNUNG

160x128 DOTS MIT CFL-BELEUCHTUNG, BLAU NEGATIV, TOUCH PANEL WIE OBEN, JEDOCH OHNE TOUCH PANEL	EA KIT160-7CTP
160x128 DOTS MIT WEISSE LED-BEL., BLAU NEGATIV, TOUCH PANEL WIE OBEN, JEDOCH OHNE TOUCH PANEL	EA KIT160-7LWTP
	EA KIT160-7LW

ELECTRONIC
ASSEMBLY GMBH

LOCHHAMER SCHLAG 17 · D-82166 GRÄFELFING
TEL 089/8541991 · FAX 089/8541721 · <http://www.lcd-module.de>

ALLGEMEINES

EA KIT160 ist eine komplett aufgebaute Steuer- und Bedieneinheit mit diversen eingebauten Funktionen. Das kompakt aufgebaute Display bietet zusammen mit dem sehr guten Supertwistkontrast eine sofort einsetzbare Einheit. Die Ansteuerung erfolgt über die Standard Schnittstellen RS-232 oder RS-422. Die Bedieneinheit enthält neben kompletten Grafikroutinen zur Displayausgabe auch verschiedenste Schriften. Die Programmierung erfolgt über hochsprachenähnliche Grafikbefehle; die zeitraubende Programmierung von Zeichensätzen und Grafikroutinen entfällt hier völlig. Die simple Verwendung von Makros und die Eingabemöglichkeit über Touchpanel machen es zu einem richtigen Power Display.

DISPLAYVARIANTEN

CFL-Beleuchtung EA KIT160-7CTP: blauer Hintergrund mit weiß leuchtender Schrift. Extrem hell und kontraststark. Lebensdauer der Beleuchtung 10.000-20.000 Stunden.
Ersatzbeleuchtung unter EA CFL160-7 lieferbar. Stromverbrauch typ. 450mA.

LED-Beleuchtung EA KIT160-7LWTP: blauer Hintergrund mit weiß leuchtender Schrift. Guter Kontrast, Stromverbrauch typ. 250 mA.

HARDWARE

Die Bedieneinheit ist für +5V Betriebsspannung ausgelegt. Optional ist eine Versorgung mit 9..35V möglich. Die Datenübertragung erfolgt seriell im RS-232 oder RS-422 Format. Das Übertragungsformat ist fest auf 8 Datenbits, 1 Stopbit, no Parity eingestellt. Die Baudrate kann über DIP-Schalter von 1.200 Baud bis zu 115.200 Baud ausgewählt werden. Handshakeleitungen RTS und CTS stehen zur Verfügung. Datenformat:



TOUCH PANEL

Die Versionen EA KIT160-7CTP und -7LWTP sind mit einem integrierten Touch Panel ausgerüstet. Durch Berühren des Displays können hier Eingaben gemacht und Einstellungen per Menü getätigter werden. Die Beschriftung der "Tasten" ist flexibel und auch während der Laufzeit änderbar (verschiedene Sprachen, Icons). Das Zeichnen der einzelnen "Tasten", sowie das Beschriften oder Zusammenfassen mehrerer Felder wird von der eingebauten Software komplett übernommen.

SOFTWARE

Die Programmierung der Bedieneinheit erfolgt über Befehle wie z.B. *Zeichne ein Rechteck von (0,0) nach (64, 15)*. Es ist keine zusätzliche Software oder Treiber erforderlich. Zeichenketten lassen sich **pixelgenau** plazieren. Das Mischen von Text und Grafik ist jederzeit möglich. Es können bis zu 16 verschiedene Zeichensätze verwendet werden. Jeder Zeichensatz kann wiederum 2- bis 8-fach gezoomt werden.

ZUBEHÖR

Frontpanel zur Montage

Als Zubehör ist ein Frontpanel aus eloxiertem Aluminium erhältlich. Damit lässt sich die Bedieneinheit ohne sichtbare Schrauben montieren. Das Frontpanel EA OFP160-7 ist in den Farben schwarz (SW) und blau (BL) lieferbar.

Diskette zur Makroerstellung

Zur Makroprogrammierung ist eine Diskette EA DISK240 erforderlich^{*)}. Diese übersetzt die in eine Textdatei eingegebenen Befehle in einen für die Bedieneinheit lesbaren Code und brennt diesen dauerhaft ins EEPROM.

Kabel für PC

Für die einfache Anbindung an PC's (Makroprogrammierung) liefern wir ein ca. 1,5m langes Kabel mit 9-pol. SUB-D Stecker (female) EA KV24-9B. Einfach an die COM 1 oder COM 2 anstecken und loslegen. Hinweis: Das Kabel ist nicht für die RS-422 Version EA OPT-RS4224 geeignet.

^{*)} auch im Internet unter <http://www.lcd-module.de/deu/disk/disk240.zip>

ELECTRONIC ASSEMBLY

EXTERNETASTATUR

Am Steckanschluss J8 kann eine Tastatur (einzelne Tasten bis zur 8x7 Matrix-Tastatur) angeschlossen werden. Die angeschlossenen Tasten werden dabei per Software entprellt. Bitte beachten Sie, daß der Anschluß einer externen Tastatur nur bei den Versionen ohne integriertem Touch Panel möglich ist.

Jede Taste wird zwischen einen Ausgang und einen Eingang geschaltet. Jeder Eingang ist mit einem ca. 100k Ω Pullup abgeschlossen.

Um Doppeltastendrücke zu erkennen, müssen die Ausgänge voneinander entkoppelt werden. Dies geht am besten mit Schottky-Dioden (z.B. BAT 43).

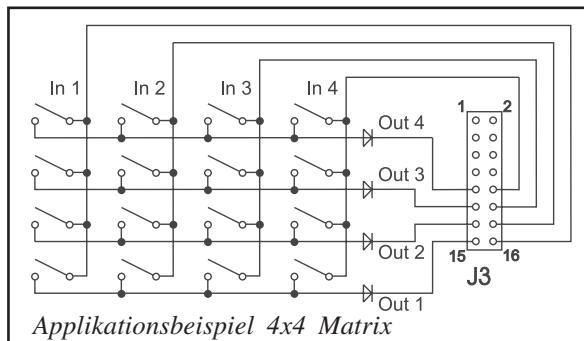
Senden der Tastendrücke

Bei jedem Druck einer Taste wird die dazugehörige Tastennummer (1..56) über die serielle Schnittstelle gesendet oder es wird ein internes Touch Makro, falls definiert, mit der Tastennummer (1..56) gestartet. Das Loslassen der Taste wird nicht gesendet. Soll auch das Loslassen gesendet werden, so kann das über die Definition des Touch Makros Nr.0 realisiert werden. Der automatische Tastaturscan lässt sich über den Befehl "ESC T A 0" deaktivieren. Falls die Handshakleleitung CTS das Senden nicht erlaubt, können Tastendrücke verloren gehen. Die Tastennummer kann folgendermaßen bestimmt werden:

Tastennummer = (Ausgang -1) * 8 + Eingang

(Ausgang: Zahl zwischen 1 und 7, Eingang: zahl zwischen 1 und 8).

Matrix - Tastaturanschluß J8					
Pin	Symbol	Funktion	Pin	Symbol	Funktion
1	-	nc	2	IN 8	Eingang Spalte 8
3	OUT 7	Ausgang Zeile 7	4	IN 7	Eingang Spalte 7
5	OUT 6	Ausgang Zeile 6	6	IN 6	Eingang Spalte 6
7	OUT 5	Ausgang Zeile 5	8	IN 5	Eingang Spalte 5
9	OUT 4	Ausgang Zeile 4	10	IN 4	Eingang Spalte 4
11	OUT 3	Ausgang Zeile 3	12	IN 3	Eingang Spalte 3
13	OUT 2	Ausgang Zeile 2	14	IN 2	Eingang Spalte 2
15	OUT 1	Ausgang Zeile 1	16	IN 1	Eingang Spalte 1



Die Versionen EA KIT160-7CTP und -7LWTP werden mit einem integrierten Touch Panel mit 56 Feldern geliefert. Die Bedieneinheit unterstützt dieses Touch Panel mit komfortablen Befehlen. So können z.B. mehrere Touch-Felder zu einer großen Gesamt-Taste zusammengefasst, die Taste gezeichnet und eine Beschriftung der Taste erfolgen. Ebenso kann dieser eben definierten Taste ein Return-Code (1..255) zugewiesen werden. Wird der Return-Code 0 zugewiesen, so ist die Taste deaktiviert und wird bei Betätigung nicht gemeldet.

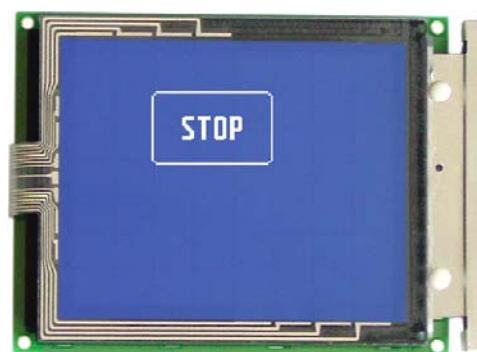
Beim Berühren der Touch-Tasten können diese automatisch invertiert werden und ein Summer signalisiert die Berührung. Der definierte Return-Code der Taste wird über die serielle Schnittstelle gesendet oder es wird ein internes Touch Makro mit der Nummer des Return-Codes gestartet.

Beispiel:

Definieren einer Taste von Feld 11 bis 21, mit dem Return-Code 65= 'A' und dem Text "STOP". Anmerkung: Vor der Definition einzelner Tasten sollten alle Felder durch "ESC T R" deaktiviert sein.

Beispiel	Auszugebende Codes												Bemerkung
für Compiler	#TH 11, 21, 'A', 2, "STOP"												Die Endekennung 0 wird hier nicht angegeben! die Punkte ' ' stehen für nicht darzustellende ASCII-Zeichen
als ASCII	ESC	T	H	.	.	A	.	S	T	O	P	.	
in Hex	\$1B	\$54	\$48	\$0B	\$15	\$41	\$02	\$53	\$54	\$4F	\$50	\$00	
in Dezimal	27	84	72	11	21	65	2	83	84	79	80	0	
	Befehlskennung	Einleitung Touch-Befehl	horizontale Beschiftung	linke oberes Touchfeld	rechtes unteres Touchfeld	Return Code	Taste zeichnen mit Rahmen					Text Ende Kennung	

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



BAUDRATEN

Die Baudrate lässt sich über die linken 3 DIP Schalter einstellen. Im Auslieferungszustand sind 9.600 Baud eingestellt (DIP 3 ON). Bitte beachten Sie, daß der interne Datenpuffer lediglich 24 Byte umfaßt. Deshalb sollte unbedingt die Handshakeleitung RTS abgefragt werden (+10V Pegel: Daten können angenommen werden; -10V Pegel: Display ist Busy). Das Datenformat ist fest eingestellt auf 8 Datenbits, 1 Stopbit, keine Parität.

SCHREIBSCHUTZ FÜR MAKROPROGRAMMIERUNG

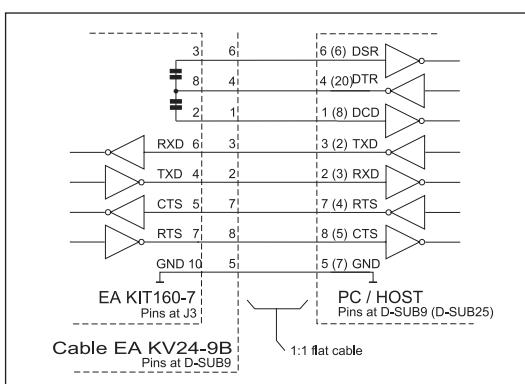
Über DIP Schalter 6 lässt sich ein versehentliches Überschreiben der Makros, Bilder und Fonts verhindern.

RS-232/RS-422 ANSCHLUSS

Standardmäßig wird die Bedieneinheit mit einer RS-232 Schnittstelle ausgeliefert. Die Stifteleiste J3 hat dann die Pinbelegung wie in der Tabelle links abgebildet. J3 ist im Raster 2,54mm ausgeführt. Wird die Bedieneinheit zusammen mit der Option EA OPT-RS4224 bestellt, sind spezielle RS-422 Treiber bestückt. Damit ist die Pinbelegung in der Tabelle rechts gültig.

An der Lötaugenleiste J5 stehen übrigens die gleichen seriellen Daten mit 5V Pegeln und TTL-Logik zur Verfügung. Diese Pegel sind für den direkten Anschluß an einen µC geeignet. Bei Verwendung dieser Signale müssen allerdings die Lötbrücken LB 5 + LB 6 geöffnet werden!

RS-232 Anschluß J3			
Pin	Symbol	In/Out	Funktion
1	VDD	-	+ 5V Versorgung
2	DCD	-	Brücke nach DTR
3	DSR	-	Brücke nach 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	-	siehe Pin 2, Pin 3
9	-	-	NC
10	GND	-	0V Masse



Baudraten			
DIP Schalter			
1	2	3	
ON	ON	ON	8,N,1
OFF		ON	1200
ON	OFF	ON	2400
OFF	ON	OFF	4800
OFF	OFF	ON	9600
ON	ON	OFF	19200
OFF	ON	OFF	38400
ON	OFF	OFF	57600
OFF	OFF	OFF	115200

Schreibschutz	
DIP	Schreibschutz für EEPROM
6	Ein keine Makroprog. mögl.
ON	Aus Makroprog. möglich
OFF	

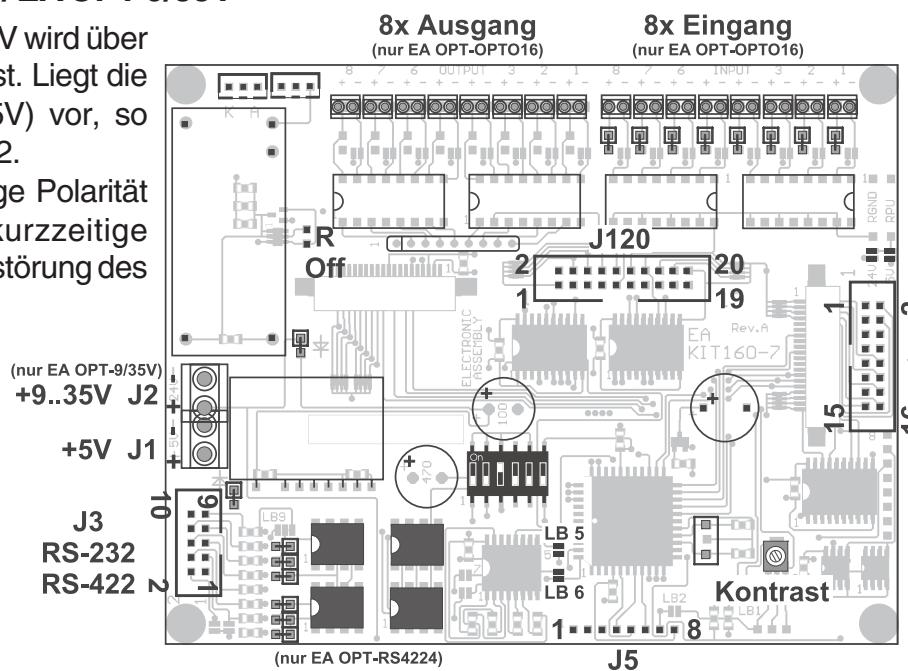
RS-422 Anschluß J3		
Pin	Symbol	Funktion
1	VDD	+ 5V Versorgung
2	Data In-	Receive Data
3	Data In+	Receive Data
4	Data Out-	Transmit Data
5	Data Out+	Transmit Data
6	HS In-	Handshake
7	HS In+	Handshake
8	HS Out-	Handshake
9	HS Out+	Handshake
10	GND	0V Masse

VERSORGUNGSSPANNUNG/EA OPT-9/35V

Die Versorgungsspannung von +5V wird über die Schraubklemme J1 eingespeist. Liegt die Version für 9..35V (EA OPT-9/35V) vor, so erfolgt die Stromversorgung über J2.

Achtung: Unbedingt auf die richtige Polarität achten! Eine auch noch so kurzzeitige Verpolung kann zur sofortigen Zerstörung des gesamten Displays führen.

Erweiterung J5			
Pin	Symbol	In/Out	Funktion
1	VU	-	9..35V Versorgung
2	VDD	-	+ 5V Versorgung
3	GND	-	0V, Masse
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



ELECTRONIC ASSEMBLY

EIN- UND AUSGÄNGE

Alle EA KIT160-7 werden mit 8 digitalen Ein- und 8 Ausgängen (5V CMOS Pegel, nicht potentialfrei) geliefert.

8 Ausgänge

Jede Leitung kann per Befehl "ESC Y W" individuell angesteuert werden. Pro Leitung kann ein Strom von max. 6mA geschaltet werden. Es ist somit möglich, mit einem Ausgang direkt eine LED (low current) zu schalten. Größere Ströme können mittels externen Transistors verstärkt werden.

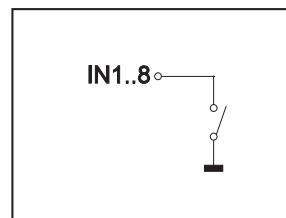
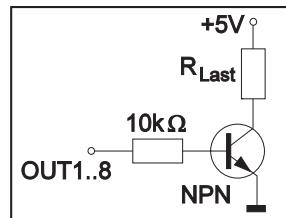
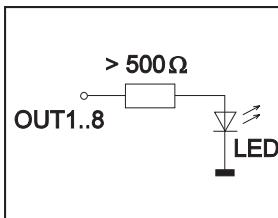
8 Eingänge

Die Eingänge können direkt über die serielle Schnittstelle abgefragt und ausgewertet werden ("ESC Y R"). Zusätzlich ist es möglich, bei Änderungen an den Eingängen ein Portmakro automatisch aufzurufen, durch die binäre Kombination von 8 Eingängen sind bis zu 256 Portmakros ansprechbar. Die automatische Portabfrage lässt sich mit dem Befehl "ESC Y A 0" deaktivieren.

Anmerkung:

Die Logik ist für langsame Vorgänge ausgelegt; d.h. mehr als 3 Änderungen pro Sekunde können nicht mehr sinnvoll ausgeführt werden. Falls ein Eingang offen ist, so ist dieser High (interner 100 kOhm PullUp).

Ein- und Ausgänge J120					
Pi	Symb	Funktion	Pin	Symb	Funktion
1	VDD	+5V Vers.	2	GND	0V, Masse
3	OUT 1	Ausgang 1	4	IN 1	Eingang 1
5	OUT 2	Ausgang 2	6	IN 2	Eingang 2
7	OUT 3	Ausgang 3	8	IN 3	Eingang 3
9	OUT 4	Ausgang 4	10	IN 4	Eingang 4
11	OUT 5	Ausgang 5	12	IN 5	Eingang 5
13	OUT 6	Ausgang 6	14	IN 6	Eingang 6
15	OUT 7	Ausgang 7	16	IN 7	Eingang 7
17	OUT 8	Ausgang 8	18	IN 8	Eingang 8
19	GND	0V, Masse	20	VDD	+5V Vers.



EIN- UND AUSGÄNGE ÜBER OPTOKOPPLER (EA OPT-OPTO16)

Die Ein- und Ausgänge können optional mit Optokopplern ausgestattet werden (EA OPT-OPTO16). Die Ein- und Ausgänge sind dann sowohl von der restlichen Elektronik, als auch untereinander isoliert. Der Anschluß erfolgt über 16 einzelne Schraublemmen.

An allen 8 Eingängen können direkt Spannungen von 5..35V angelegt werden. Pegel über 4V werden als H-Pegel erkannt, Pegel unter 2V gelten als L-Pegel. Spannungen zwischen 2 und 4V sind undefiniert.

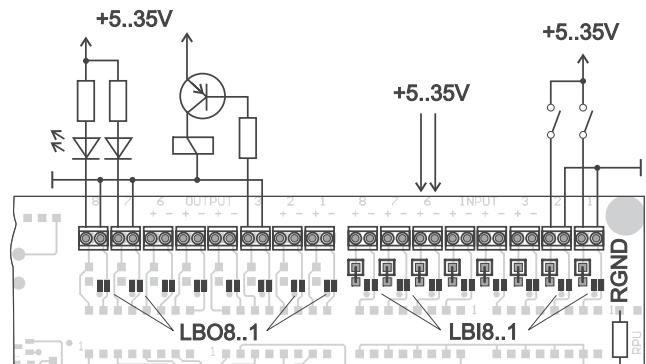
Als Ausgang ist jeweils der Kollektor (+) und Emitter (-) eines NPN-Transistors an den Schraubklemmen herausgeführt. Jeder Ausgang kann 10mA schalten. Hinweis: Der Minuspol jeder Schraubklemme kann durch Schließen der Lötbrücken LBI1..8 bzw. LBO1..8 zusammengeschalten werden. Zusätzlich können diese Lötbrücken auf die Systemmasse GND gelegt werden (0Ω Brücke RGND einlöten).

Anmerkung: Die Optokoppler invertieren die Eingangslogik (alle Eingänge offen: Portmakro 255). Hier empfieilt es sich (z.B. im Power-On-Makro) mit dem Befehl "ESC Y I 1" die Eingänge invertiert auszuwerten (d.h. alle Eingänge offen: Portmakro 0).

GRUNDEINSTELLUNGEN

Nach dem Einschalten bzw. nach einem manuell ausgelösten Reset werden die nebenstehenden Register auf einen bestimmten Wert voreingestellt.

Beachten Sie bitte, daß alle Einstellungen durch Erstellen eines Power-On-Makros (Normal-Makro Nr.0) überschrieben werden können.



Grundeinstellungen		
Register	Befehl	nach Power-On / Reset
Text-Modus	ESC L	setzen, schwarz
Terminal Font	ESC FT	Font 3, kein Zoom
Cursor	ESC QC	ein
Blinkzeit	ESC QZ	0,6 sek.
Selbst definierte Zch.	ESC E	undefiniert
Grafik-Modus	ESC V	setzen
Grafik Font	ESC F	Font 3, kein Zoom
Last xy	ESC W	(0;0)
Bargraph 1..16	ESC B	undefiniert
Clipboard	ESC C	leer
Selekt/Deselekt	ESC K	selektiert
Ausgänge OUT1..8	ESC Y	L-Pegel

MAKRO PROGRAMMIERUNG

Einzelne oder mehrere Befehlsfolgen können als sog. Makros zusammengefasst und im EEPROM fest abgespeichert werden. Diese können dann mit den Befehlen *Makro ausführen* gestartet werden. Es gibt 3 verschiedene Makrotypen:

Touch Makro (1..255)

Start bei Berührung eines Touchfeldes (nur bei Versionen mit Touch Panel TP) oder bei Betätigung einer ext. angeschlossenen Taste/Matrixtastatur oder per Befehl 'ESC MT nr'. Das Touch Makro Nr.0 hat eine Sonderstellung: Beim Loslassen einer x-beliebigen Taste wird das Touch Makro Nr.0 gestartet.

Port Makro (0..255)

Start bei Anlegen/Ändern einer Spannung an den Eingängen IN 1..8 oder per Befehl 'ESC MP nr'.

Normal Makro (1..255)

Start per Befehl 'ESC MN nr' über die serielle Schnittstelle oder von einem anderen Makro aus. Es können auch mehrere hintereinander liegende Makros automatisch zyklisch aufgerufen werden (Movie, sich drehende Sanduhr, mehrseitiger Hilfetext).

Power-On-Makro

Das Normal Makro Nr.0 hat eine Sonderstellung: es wird automatisch nach dem Einschalten ausgeführt. Hier kann man z.B. den Cursor abschalten und einen Startbildschirm definieren.

Achtung: Wird im Power-On-Makro eine Endlosschleife programmiert, ist das Display nicht mehr ansprechbar. In diesen Fall hilft nur noch (ab REV. B): DIP Schalter 5 auf ON, Power off, Power on und dann DIP 5 wieder auf off. Jetzt müssen die Fonts und Makros wieder neu eingespielt werden.

256 BILDER FEST ABGELEGT

Um Übertragungszeiten der seriellen Schnittstelle zu verkürzen, oder auch um Speicherplatz im Prozessorsystem zu sparen, können bis zu 256 Bilder im internen EEPROM abgelegt werden. Der Aufruf erfolgt über den Befehl "ESC U E" über die serielle Schnittstelle oder aus einem Touch-/Port-/Normal-Makro heraus. Verwendet werden können alle Bilder im Windows BMP Format. Die Erstellung und Bearbeitung erfolgt über Standardsoftware wie z.B. Windows Paint oder Photoshop.

ERSTELLEN INDIVIDUELLER MAKROS

Um nun Ihre speziellen Makros erstellen zu können, benötigen Sie folgende Hilfsmittel:

- die Diskette EA DISK240^{*)}; sie enthält einen Compiler, Beispiele und Fonts
- einen PC mit serieller Schnittstelle COM1 oder COM2, mit ca. 500kB Platz auf der Festplatte
- einen Texteditor wie z.B. WordPad, Norton Editor o.ä.

Um eine Befehlsfolge als Makro zu definieren, werden alle Befehle auf dem PC in eine Datei z.B. DEMO.KMC geschrieben. Hier bestimmen Sie welche Zeichensätze eingebunden werden und in welchen Makros welche Befehlsfolgen stehen sollen. Sind die Makros definiert, startet man das Programm C:>KITCOMP DEMO.KMC. Dieses erzeugt eine EEPROM-Datei DEMO.EEP, welche dann automatisch mit der eingetragenen Baudrate in das Display-EEPROM gebrannt wird. Dieser Vorgang dauert nur wenige Sekunden und sofort danach können die selbstdefinierten Makros genutzt werden. Eine ausführliche Beschreibung zur Programmierung der Makros finden Sie zusammen mit Beispielen auf der Diskette EA DISK240^{*)} unter dem Namen DOKU.DOC (für WORD) bzw. DOKU.TXT (DOS).

```
;Makro Demo  
KIT160-7  
COM2: 115200 ; KIT festlegen  
; 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
```

^{*)} auch im Internet unter <http://www.lcd-module.de/disk/disk240.zip>

ELECTRONIC ASSEMBLY

INTEGRIERTE FONTS

In jeder Grafikeinheit sind standardmäßig 5 Zeichensätze integriert. Jeder Zeichensatz kann in 1- bis 8-facher Höhe verwendet werden. Unabhängig davon lässt sich auch die Breite verdoppeln bis verachtfachen.

Font 1: 4x6

+ 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)	ß	ä	ö	ü	ç	ñ	ñ	ñ	ñ	ñ	ñ	ñ	ñ	ñ	ñ	ñ
\$40 (dez: 64)	ó	á	é	í	ó	ú	ú	ú	ú	ú	ú	ú	ú	ú	ú	ú
\$50 (dez: 80)	þ	ð	ø	æ	þ	ø	ø	ø	ø	ø	ø	ø	ø	ø	ø	ø

+ 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)	ß	ä	ö	ü	ç	ñ	ñ	ñ	ñ	ñ	ñ	ñ	ñ	ñ	ñ	ñ
\$40 (dez: 64)	ó	á	é	í	ó	ú	ú	ú	ú	ú	ú	ú	ú	ú	ú	ú
\$50 (dez: 80)	þ	ð	ø	æ	þ	ø	ø	ø	ø	ø	ø	ø	ø	ø	ø	ø
\$60 (dez: 96)	ä	ë	ö	ë	é	ë	ë	ë	ë	ë	ë	ë	ë	ë	ë	ë
\$70 (dez: 112)	ä	ë	ö	ë	é	ë	ë	ë	ë	ë	ë	ë	ë	ë	ë	ë
\$80 (dez: 128)	ä	ë	ö	ë	é	ë	ë	ë	ë	ë	ë	ë	ë	ë	ë	ë
\$90 (dez: 144)	ä	ë	ö	ë	é	ë	ë	ë	ë	ë	ë	ë	ë	ë	ë	ë

Font 3: 6x8

Jeder Text lässt sich linksbündig, rechtsbündig und zentriert ausgeben. Auch eine 90° Drehung (vertikaler Einbau des Displays) ist möglich.

Die Makroprogrammierung erlaubt die Einbindung von weiteren 11 Fonts, sowie die komplette Umgestaltung der einzelnen Zeichen. Durch einen Fonteditor auf der Diskette EA DISKFONT6963 können alle nur erdenklichen Schriften mit bis zu 16x16 Pixeln Größe erstellt und einprogrammiert werden.

Font 5: 8x16

+ 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)	ß	ä	ö	ü	ç	ñ	ñ	ñ	ñ	ñ	ñ	ñ	ñ	ñ	ñ	ñ
\$40 (dez: 64)	ó	á	é	í	ó	ú	ú	ú	ú	ú	ú	ú	ú	ú	ú	ú
\$50 (dez: 80)	þ	ð	ø	æ	þ	ø	ø	ø	ø	ø	ø	ø	ø	ø	ø	ø
\$60 (dez: 96)	ä	ë	ö	ë	é	ë	ë	ë	ë	ë	ë	ë	ë	ë	ë	ë
\$70 (dez: 112)	ä	ë	ö	ë	é	ë	ë	ë	ë	ë	ë	ë	ë	ë	ë	ë
\$80 (dez: 128)	ä	ë	ö	ë	é	ë	ë	ë	ë	ë	ë	ë	ë	ë	ë	ë
\$90 (dez: 144)	ä	ë	ö	ë	é	ë	ë	ë	ë	ë	ë	ë	ë	ë	ë	ë

TIP: SCHRIFTEFFEKTE

Mit dem Befehl ESC L TEXT-Modus (Verknüpfung, Muster) können bei grossen Schriften interessante Effekte durch Überlagerung (mehrmaliges versetzes Schreiben eines Wortes) erzielt werden.

TEST — **TEST**

Orginalschrift 8x16 mit ZOOM 3
an Position 0,0 mit Muster Schwarz

Durch Überlagerung (EXOR) an
Pos.1,1 entstandene "Outline Schrift"

TEST

Nochmalige Überlagerung (EXOR) der
"Outline Schrift" an Pos.2,2. führt zu einer
"Outline Schrift mit Füllung"

TEST

Überlagerung (ODER) mit Muster 50% Grau
der "Outline Schrift" an Pos.0,0. führt zu
einer "Schrift mit Musterfüllung"

ALLE BEFEHLE AUF EINEN BLICK

Befehlstabelle für EA KIT160-7													
Befehl	Codes					Anmerkung							
Befehle für den Terminal Betrieb													
Formfeed FF (dez:12)	^L							Bildschirm wird gelöscht und der Cursor nach Pos. (1,1) gesetzt					
Carriage Return CR(13)	^M							Cursor ganz nach links zum Zeilenanfang					
Linefeed LF (dez:10)	^J							Cursor 1 Zeile tiefer, falls Cursor in letzter Zeile dann auf 1. Zeile setzen					
Cursor On / Off	ESC Q	C	n1							n1=0: Cursor ist unsichtbar; n1=1: Cursor blinkt (invers 6/10s);			
Cursor positionieren	ESC O	n1	n2							n1=Spalte; n2=Zeile; Ursprung links oben ist (1,1)			
Terminal Font einstellen	ESC F	T	n1							n1=1: Font Nr. n1 (1..16) für Terminal Betrieb einstellen			
Befehle zur Textausgabe													
Text-Modus	ESC L	n1	mst							Modus n1: 1=setzen; 2=löschen; 3=invers; 4=Replace; 5=Invers Replace; mst: Muster Nr. 0..7 verwenden;			
Font einstellen	ESC F	n1	n2	n3									
Zeichenkette horizontal ausgeben	ESC Z	L Z R	x1	y1 Text ...	NUL			Font mit der Nummer n1 (1..16) einstellen; n2=X n3=Y-Zommfaktor (1x..8x); Eine Zeichenkette (...) an x1,y1 ausgeben. 'NUL' (\$00)=Zeichenkettenende; Mehrere Zeilen werde durch das Zeichen 'I' (\$7C, dez: 124) getrennt; 'L'= Linksbündig an x1; 'Z'= Zentriert an x1; 'R'= Rechtsbündig an x1; y1 ist immer die Oberkannte der Zeichenkette					
Zeichenkette 90° gedreht (vertikal) ausgeben	ESC Z	O M U	x1	y1 Text ...	NUL			Eine Zeichenkette (...) um 90° gedreht an x1,y1 ausgeben; 'NUL' (\$00)=Ende; Mehrere Zeilen werde durch das Zeichen 'I' (\$7C, dez: 124) getrennt; 'O'= Oben-Bündig an y1; 'M'= Mittig an y1; 'U'= Unten-Bündig an y1; x1 ist immer die Rechte Kannte der Zeichenkette					
Zeichen definieren	ESC E	n1	daten ...						n1=Zeichen Nr.; daten=Anzahl Bytes je nach akt. Font				
Befehle zum Zeichnen													
Grafik-Modus	ESC V	n1							Zeichenmodus einstellen für die Befehle: 'Punkt setzen', 'Gerade zeichnen', 'Rechteck', 'Rundeck' und 'Bereich mit Füllmuster' n1: 1=setzen; 2=löschen; 3=invers; 4=Replace; 5=Invers Replace;				
Punkt setzen	ESC P	x1	y1							Ein Pixel an die Koordinaten x1, y1 setzen			
Gerade zeichnen	ESC G	x1	y1	x2	y2								
Gerade weiter zeichnen	ESC W	x1	y1							Eine Gerade vom letzten Endpunkt bis x1, y1 zeichnen			
Rechteck Befehle													
Rechteck zeichnen	ESC R	R	x1	y1	x2	y2							
Rundeck zeichnen		N	x1	y1	x2	y2							
Bereich löschen		L	x1	y1	x2	y2							
Bereich invertieren		I	x1	y1	x2	y2							
Bereich füllen		S	x1	y1	x2	y2							
Bereich m. Füllmuster		M	x1	y1	x2	y2	mst	Einen Bereich von x1,y1 nach x2,y2 mit Muster mst (0..7) zeichnen					
Box zeichnen		O	x1	y1	x2	y2	mst	Ein Rechteck mit Füllmuster mst (0..7) zeichnen; (immer Replace)					
Rundbox zeichnen		J	x1	y1	x2	y2	mst	Ein Rundeck mit Füllmuster mst (0..7) zeichnen; (immer Replace)					
Bitmap Bilder Befehle													
Bild aus EEPROM	ESC U	E	x1	y1	nr								
Bild laden		L	x1	y1	daten ...								
Hardcopy senden		H	x1	y1	x2	y2							
Display-Befehle (Wirkung auf das gesamte Display)													
Display löschen	ESC D	L							Displayinhalt löschen (alle Pixel aus)				
Display invertieren		I							Displayinhalt invertieren (alle Pixel umkehren)				
Display füllen		S							Displayinhalt füllen (alle Pixel ein)				
Display ausschalten		A							Displayinhalt wird unsichtbar bleibt aber erhalten, Befehle weiterhin möglich				
Display einschalten		E							Displayinhalt wird wieder sichtbar				
Display Clipboard		C							Inhalt des Clipboards wird dargestellt. Displayausgaben sind nicht mehr sichtbar				
Disp. Normaldarstellung		N							Aktuelles Bild wird dargestellt (Normalbetrieb). Alle Ausgaben wieder sichtbar				
Display Reset							Der Displaykontroller wird per Befehl rückgesetzt und neu initialisiert						
Makro Befehle													
Makro ausführen	ESC M	N	n1							Das (Normal-)Makro mit der Nummer n1 aufrufen (max. 7 Ebenen)			
Touch Makro ausführen		T	n1							Das Touch-Makro mit der Nummer n1 aufrufen (max. 7 Ebenen)			
Port Makro ausführen		P	n1							Das Port-Makro mit der Nummer n1 aufrufen (max. 7 Ebenen)			
autom. Makro zyklisch		A	n1	n2	n3								
autom. Makro pingpong							Makros n1..n2 automatisch zyklisch abarbeiten; n3=Pause in 1/10s						
							Makros autom. von n1..n2..n1 (PingPong) abarbeiten; n3=Pause in 1/10s						

ELECTRONIC ASSEMBLY

Bargraph Befehle											
Bargraph definieren	ESC	B	R L O U	nr	x1	y1	x2	y2	aw	ew	mst
											Bargraph nach L(inks), R(echts), O(ben), U(nten) mit der 'nr' (1..16) definieren. x1,y1,x2,y2 sind das umschließende Rechteck des Bargraphs. aw,ew sind die Werte für 0% und 100%. mst=Muster (0..7)
Clipboard Befehle (Zwischenspeicher für Bildbereiche)											
Displayinhalt sichern	ESC	C	B								Der gesamte Displayinhalt wird als Bildbereich ins Clipboard kopiert
Bereich sichern			S	x1	y1	x2	y2				Der Bildbereich von x1, y1 bis nach x2, y2 wird ins Clipboard kopiert
Display restaurieren			R								Der Bildbereich im Clipboard wird wieder ins Display zurückkopiert
Bereich kopieren			K	x1	y1						Der Bildbereich im Clipboard wird ins Display nach x1, y1 kopiert
Tastatur / Touch-Panel Befehle											
Touch-Taste mit horizontaler Beschriftung definieren	ESC	T	H	f1	f2	Ret. Cod.	Form	Text ...	NUL		Die Touch-Felder f1 bis f2 (gegenüberliegenden Eckfelder), werden zu einer Touch-Taste mit dem Rückgabewert 'Ret. Code' (=1..255) zusammengefasst (Ret.Code=0 Touch-Taste nicht aktiv).
Touch-Taste mit vertikaler (90° gedreht) Beschriftung definieren			V								'Form':Touch-Taste (=0 nichts; =1 löschen; =2 mit Rahmen) zeichnen 'Text': es folgt eine Zeichenkette die zentriert mit dem akt. Font in der Touch-Taste plaziert wird, mehrzeilige Texte werden mit dem Zeichen ';' (\$7C, dez: 124) getrennt; Zeichen NUL (\$00) = Zeichenkettenende
Touch-Tasten (P)Reset			P								Alle Touch-Tasten werden aufsteigend aktiviert (Felder mit Code 1..60)
			R								Alle Touch-Tasten werden deaktiviert (alle Felder mit Code 0)
Touch-Tasten Reaktion			I	n1							n1=0: kein invertieren beim Berühren der Touch-Taste n1=1: Touch-Taste wird beim Berühren automatisch invertiert
Touch-Taste Invertieren			S	n1							n1=0: kein Summer beim Berühren einer (Touch-)Taste n1=1: Summer piepst kurz beim Berühren einer (Touch-)Taste
Taste manuell abfragen			M	n1							Die Touch-Taste mit dem zugeordnetem Return-Code n1 wird manuell Invertiert
Tasten-Abfrage Ein/Aus			W								Die momentan gedrückte (Touch-)Taste wird auf der RS-232/RS-422 gesendet
			A	n1							Tastaturabfrage wird n1=0:deaktiviert; n1=1:aktiviert, Tastendrücke werden automatisch gesendet; n1=2:aktiviert, Tastendrücke werden nicht gesendet (mit ESC T W abfragen)
Menü / Popup Befehle											
Menü mit horizontalen Einträgen definieren	ESC	N	H	x1	y1	nr	Text ...	NUL			Ein Menü wird ab der Ecke x1,y1 (Horizontales Menü = linke obere Ecke; Vertikale menü = rechte obere Ecke) mit dem akt. Font gezeichnet. nr:= aktuell invertierter Eintrag (z.B: 1 = 1. Eintrag)
Menü mit vertikalen (90° gedrehten) Einträgen definieren			V								Text:= Zeichenkette mit den Menüeinträgen. Die einzelnen Einträge sind durch Zeichen ';' (\$7C,dez:124) getrennt z.B. "Eintrag1 Eintrag2 Eintrag3" Der Hintergrund des Menüs wird automatisch ins Clipboard gesichert.
Menübox invertieren			I								Ist bereits ein Menü definiert, wird dieses automatisch abgebrochen+entfernt. Die gesamte Menübox wird invertiert. Sinnvoll für negative Darstellung
nächster Eintrag			N								Der nächste Eintrag wird invertiert oder bleibt am Ende stehen
vorheriger Eintrag			P								Der vorherige Eintrag wird invertiert oder bleibt am Anfang stehen
Menüende / Senden			S								Das Menü wird vom Display entfernt und durch den Clipboardinhalt ersetzt der aktuelle Eintrag wird als Nummer (1..n) gesendet (0=kein Menü dargestellt)
Menüende / Makro			M	nr							Das Menü wird vom Display entfernt und durch den Clipboardinhalt ersetzt Für Eintrag 1 wird Makro 'nr' aufrufen; für Eintrag 2 Makro nr+1 usw.
Menüende / Abbrechen			A								Das Menü wird vom Display entfernt und durch den Clipboardinhalt ersetzt
Kontroll- / Definitions-Befehle											
Automatisch blinkender Bereich (Cursor-Funktion)	ESC	Q	D	x1	y1	x2	y2				Definiert einen Blinkbereich x1,y1 bis x2,y2; Blinkfunktion aktivieren
			Z	n1							Einstellen der Blinkzeit n1= 1..15 in 1/10s; 0=Blinkfunktion deaktivieren
			M	I mst							Invers-Modus (Blinkbereich wird invertiert); Blinkfunktion aktivieren Clipboard-Modus mst=Muster (0..7) des Blockursors; Blinken aktivieren
			C	n1							Automatisch blinkender Bereich als Cursor für den Terminal Betrieb n1=0: Blinkfunktion deaktivieren; n1=1: Blinkfunktion aktivieren (Invers, 6/10s)
Selekt / Deselekt	ESC	K	S	adr							Kit mit Adresse n1 aktivieren (n1=255: alle)
			D	adr							Kit mit Adresse n1 deaktivieren (n1=255: alle)
			A	adr							Neue Adresse adr zuweisen (z.B. im Power-On Makro)
Warten (Pause)	ESC	X	n1								n1 Zehntel-Sekunden abwarten bevor der nächste Befehl ausgeführt wird.
Summer Ein / Aus	ESC	J	n1								n1=0:Summer Aus; n1=1:Summer Ein; n1=2..255:für n1 1/10s lang Ein
Bytes senden	ESC	S	anz			daten ...					Es werden anz (1..255; 0=256) Bytes auf der RS-232/RS-422 gesendet daten ... = anz Bytes (z.B Ansteuerung eines externen seriellen Druckers)
Port-Befehle											
Output-Port schreiben	ESC	Y	W	n1	n2						n1=0: Alle 8 Ausgabe-Ports entsprechend n2 (=8-Bit Binärwert) einstellen
Eingabe-Port lesen			R	n1							n1=1..8: Ausgabe-Port n1 rücksetzen (n2=0); setzen (n2=1); invertieren (n2=2)
Port Scan Ein/Aus			A	n1							n1=0: Alle 8 Eingabe-Ports als 8-Bit Binärwert einlesen n1=1..8: Eingabe-Port <n1> einlesen (1=H-Pegel=5V, 0=L-Pegel=0V)
Eingabe-Port invers			I	n1							Der automatische Scan des Eingabe-Port wird n1=0: deaktiviert; n1=1: aktiviert
Beleuchtung Ein/Aus			L	n1							Der Eingabe-Port wird n1=0: normal; n1=1: invertiert ausgewertet CFL/LED-Beleuchtung n1=0: AUS; n1=1: EIN; n1=2: INVERTIEREN; n1=3..255: Beleuchtung für n1 Zehntel Sek.. lang einschalten

PARAMETER

Die Bedieneinheit lässt sich über diverse eingebaute Befehle programmieren. Jeder Befehl beginnt mit ESC gefolgt von einem oder zwei Befehlsbuchstaben und einigen Parametern. Alle Befehle und deren Parameter wie Koordinaten und sonstige Übergabewerte werden immer als Bytes erwartet. Dazwischen dürfen keine Trennzeichen z.B. Leerzeichen oder Kommas verwendet werden. Die Befehle benötigen auch **kein Abschlussbyte** wie z.B. Carriage Return (außer Zeichenkette: \$00).

A..Z, L/R/O/U Alle Befehle werden als ASCII-Zeichen übertragen.

Beispiel: G= 71 (dez.) = \$47 leitet den Geraden-Befehl ein.

x1, x2, y1, y2 Koordinatenangaben werden mit 1 Byte übertragen.

Beispiel: x1= 10 (dez.) = \$0A

ESC 1 Byte: 27(dez.) = \$1B

n1,n2,nr,aw,ew,wert,mst,ret,

frm,daten Nummernwerte werden mit 1 Byte übertragen.

Beispiel: n1=15(dez.) = \$0F

PROGRAMMIERBEISPIEL

In der nachfolgenden Tabelle ist ein Beispiel zu sehen welches die Zeichenkette "Test" linksbündig an den Koordinaten 7,3 ausgibt.

Beispiel	Auszugebende Codes									
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 Dezimal	27	90	76	7	3	84	101	115	116	0
für Turbo-Pascal	write(aux, chr(27), 'Z', 'L', chr(7), chr(3), 'Test', chr(0));									
für 'C'	fprintf(stdaux, "\x1BZL%c%c%s\x00", 7, 3, "Test");									
für 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)									

MUSTER

Bei diversen Befehlen kann als Parameter ein Mustertyp (mst = 0..7) eingestellt werden. So können rechteckige Bereiche, Bargraphs und sogar Texte mit unterschiedlichen Mustern verknüpft und dargestellt werden.

Folgende Füllmuster stehen dabei zur Verfügung:

mst=0

mst=1

mst=2

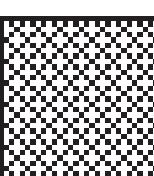
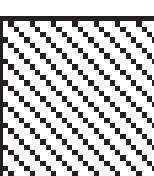
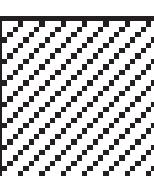
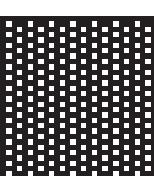
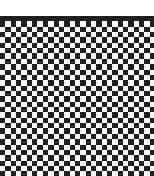
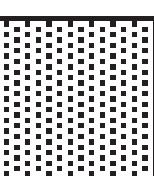
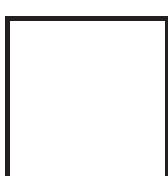
mst=3

mst=4

mst=5

mst=6

mst=7



weiss

schwarz

25% grau

50% grau

75% grau

45° rechts

45° links

45° kreuz

ELECTRONIC ASSEMBLY

BESCHREIBUNG DER EINZELNEN GRAFIKFUNKTIONEN

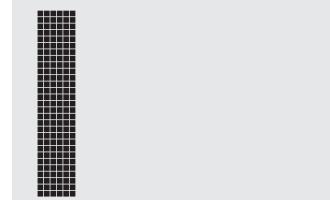
Auf den nächsten Seiten befindet sich eine detaillierte alphabetisch sortierte Beschreibung zu jeder einzelnen Funktion. Als Beispiel wird jeweils ein vergößerter Bildausschnitt von 50x32 Pixeln als Hardcopy gezeigt der den Displayinhalt nach Ausführung des Befehls darstellt. In den Beispielen sind die zu übertragenden Bytes als Hex-Werte abgebildet.

ESC B L/R/O/U nr x1 y1 x2 y2 aw ew mst Bargraph definieren

Es können bis zu 16 Bargraphs (**nr=1..16**) definiert werden, welche nach **L**=links, **R**=rechts, **O**=oben oder **U**=unten ausschlagen können. Der Bargraph beansprucht bei Vollauschlag einen Bereich mit den Koordinaten **x1,y1** bis **x2,y2**. Mit dem Anfangswert (kein Ausschlag) **aw** (=0..254) und dem Endwert (Vollausschlag) **ew** (=0..254) wird der Bargraph skaliert. Der Bargraph wird immer im Inversmodus mit dem Muster **mst** gezeichnet: Der Hintergrund bleibt somit in jedem Fall erhalten. (Achtung! Nach diesem Befehl ist der Bargraph nur definiert, am Display ist er aber noch nicht zu sehen).

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

Es wird der Bargraph Nr. 1 der nach oben ausschlägt definiert. Bei Vollausschlag nimmt er einen Bereich von den Koordinaten 4,2 bis 9,30 ein. Anfangs- und Endwert entspricht einer 4..20 mA Anzeige. (Das Bild zeigt den Bargraph im Vollausschlag wie er mit \$42 \$01 \$14 dargestellt wird)



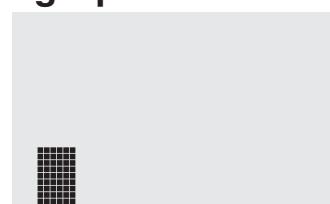
ESC B nr wert

Der Bargraph mit der Nummer **nr** (1..16) wird auf den neuen Wert eingestellt (**aw <= wert <= ew**). Ist **wert > ew** dann wird Endwert **ew** angezeigt. Der Bargraph muss vorher definiert worden sein (siehe oben).

Beispiel: \$1B \$42 \$01 \$0A

Der im oberen Beispiel definierte Bargraph Nr. 1 wird auf den Wert 10 gestellt.

Bargraph zeichnen



ESC C B

Displayinhalt ins Clipboard sichern

kopiert den gesamten Displayinhalt in das Clipboard (ZwischenSpeicher).

Beispiel: \$1B \$43 \$42

sichert den gesamten Displayinhalt für ein späteres Wiederherstellen des Bildschirms ins Clipboard. Der Displayinhalt wird dabei nicht verändert.

ESC C S x1 y1 x2 y2

Bereich ins Clipboard sichern

kopiert einen Bereich von der linken oberen Ecke **x1,y1** bis zur rechten unteren Ecke **x2,y2** in das Clipboard (ZwischenSpeicher).

Beispiel: \$1B \$43 \$53 \$00 \$00 \$17 \$1B

sichert den Bereich von 0,0 nach 23,27 für ein späteres Wiederherstellen des Bildschirms. Der Displayinhalt wird nicht verändert.

ESC C R

Bereich wiederherstellen

kopiert den zuletzt gespeicherten Bereich vom Clipboard (ZwischenSpeicher) in das Display zurück. Ziel: Ursprüngliche Koordinaten.

Beispiel: \$1B \$43 \$52

stellt den zuletzt gesicherten Bereich wieder her.

ESC C K x1 y1

Bereich vom Clipboard kopieren

kopiert den zuletzt gespeicherten Bereich im Clipboard (ZwischenSpeicher) an eine neue Position **x1,y1** des Displays.

Beispiel: \$1B \$43 \$4B \$0A \$20

kopiert den zuletzt gesicherten Bereich an die Koordinate 10,32.

ESC D L/I/S

Der gesamte Displayinhalt wird **L**=gelöscht (weiss), **I**=invertiert (umkehren) oder **S**=gefüllt (schwarz)

Beispiel: \$1B \$44 \$49

invertiert den gesamten Displayinhalt

Displayinhalt verändern

ESC D A/E

Der Displayinhalt wird **A**=ausgeschalten (unsichtbar) oder **E**=eingeschalten (sichtbar). Ausgaben sind auch im ausgeschaltenem Zustand weiterhin möglich.

Beispiel: \$1B \$44 \$41

Nach diesem Befehl ist der Displayinhalt nicht mehr sichtbar.

Display Aus- / Einschalten

ESC D N/C

Displayanzeige Normal/Clipboard

Im Display wird der **N**=normale (aktuelle) Inhalt oder **C**=der Clipboardinhalt dargestellt. Mit diesem Befehl ist es möglich verdeckt zu zeichnen. Beispiel: Der aktuelle Displayinhalt wird mit **ESC C B** ins Clipboard gesichert, danach wird das Clipboard mit **ESC D C** dargestellt. Alle weiteren Ausgaben auf das Display sind nun unsichtbar, erst nach dem Befehl **ESC D N** ist der aktuelle Inhalt wieder sichtbar.

Beispiel: \$1B \$44 \$49

Display zeigt nun den Inhalt des Clipboards an (nur komplette Bilder sind erkennbar).

ESC E n1 daten

Zeichen definieren

Es ist möglich bis zu 21 Zeichen selbst zu definieren (je nach Fontgröße). Diese Zeichen haben dann die ASCII Codes 1 bis max.21 und bleiben bis zum Abschalten der Versorgungsspannung in einem 128 Byte großen unsichtbaren Bildschirm-RAM erhalten. Bei einem 4x6 Font können bis zu 21 Zeichen definiert werden, bei einem 8x16 Font bis zu 8 Zeichen. Achtung! Sollen mehrere Zeichen aus unterschiedlichen Fonts definiert werden, so ist darauf zu achten daß z.B. ein Zeichen mit Code 1 vom 8x16 Font denselben Platz im RAM benötigt wie die Zeichen mit den Codes 1 bis 3 vom 4x6 Font (siehe Tabelle nebenan)!

Beispiel 1:

\$1B \$45 \$01

\$20 \$70 \$A8 \$20 \$20 \$20 \$20 \$00

definiert einen Pfeil nach oben für ASCII-Nr. 1 bei eingestelltem 6x8 Zeichensatz.

Beispiel 2:

\$1B \$45 \$02

\$10 \$10 \$10 \$10 \$10 \$10 \$10 \$10 \$10 \$10 \$92 \$54 \$38 \$10 \$00 \$00

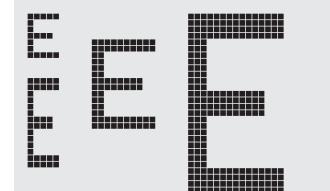
definiert einen Pfeil nach unten für ASCII-Nr. 2, bei eingestelltem 8x16 Zeichensatz.

BIT NR.							
7 6 5 4 3 2 1 0							
Byte 1							
Byte 2							
Byte 3							
Byte 4							
Byte 5							
Byte 6							
Byte 7							
Byte 8							

BIT NR.							
7 6 5 4 3 2 1 0							
Byte 1							
Byte 2							
Byte 3							
Byte 4							
Byte 5							
Byte 6							
Byte 7							
Byte 8							
Byte 9							
Byte 10							
Byte 11							
Byte 12							
Byte 13							
Byte 14							
Byte 15							
Byte 16							

Selbstdefinierbare Zeichen (Code)							
4x6	6x8	8x1	16x16				
1	1						
2	2						
3							
4	3						
5	4						
6							
7	5						
8	6						
9	7						
10	8						
11							
12	9						
13	10						
14	11						
15							
16	12						
17	13						
18	14						
19							
20	15						
21	16						

Font einstellen



ESC F n1 n2 n3

Es wird der Font mit der Nr. **n1** eingestellt. Außerdem wird ein Vergrößerungsfaktor (1..8-fach) für die Breite **n2** und für die Höhe **n3** getrennt eingestellt.

Beispiel: \$1B \$46 \$02 \$03 \$04

ab sofort ist der 6x8- Font mit 3-facher Breite und 4-facher Höhe eingestellt.

Im Bild nebenan ist das Zeichen 'E' aus dem 6x8 Font mit unterschiedlichen Vergrößerungen dargestellt.

ESC F T n1

Es wird der Font mit der Nr. **n1** für den Terminal Betrieb eingestellt. Der Font für das Terminal wird immer ohne Zoom und im REPLACE Modus benutzt.

Beispiel: \$1B \$46 \$54 \$03

ab sofort ist der 6x8 Font als Terminalfont eingestellt.

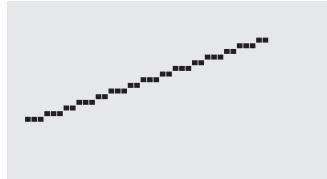
Terminal-Font einstellen

ELECTRONIC ASSEMBLY**ESC G x1 y1 x2 y2**

Eine Gerade wird von den Koordinaten **x1,y1** nach **x2,y2** unter Beachtung des eingestellten Grafikmodus 'V' (setzen / löschen / invers) gezeichnet.

Beispiel: \$1B \$47 \$03 \$14 \$28 \$06

Es wird eine Gerade von 3,20 nach 50,6 gezeichnet.

Gerade zeichnen**ESC H x1 y1 x2 y2****Hardcopy vom Displayinhalt erstellen**

Der Bereich von der linken oberen Ecke **x1,y1** bis zu rechten unteren Ecke **x2,y2** wird angefordert. Der Grafikchip sendet daraufhin sofort die Breite und Höhe des Bildausschnittes und danach die Bilddaten. Zum Aufbau der Bilddaten siehe den Befehl Bild Upload 'U'.

Beispiel: \$1B \$48 \$00 \$00 \$1F \$0F

und sofort wird der linke obere Teil des Bildschirms mit der Grösse 32 x 16 Pixel über RS-232 gesendet.

ESC J n1**Summer manuell Ein-/Ausschalten**

Der Summer wird **n1=0** ausgeschaltet, **n1=1** dauerhaft eingeschaltet oder mit **n1=2..255** für **n1/10** Sekunden lang eingeschaltet (nur bei den Versionen mit Touchpanel EA KIT160-7xxTP).

Beispiel: \$1B \$4A \$0A

nach diesem Befehl ertönt der Summer 1s lang.

ESC K A adr**Adresse zuweisen**

Dem wird die Adresse **adr** (0..254) zugewiesen. Dieser Befehl befindet sich am bestem im Power-On Makro.

Beispiel: \$1B \$4B \$41 \$01

Das KIT160 kann ab sofort unter der Adresse \$01 angesprochen werden.

ESC K S/Dadr**KIT (de)selektieren**

Das KIT160 mit der Adresse **adr** (0..254) wird **S=selektiert** oder **D=deselektiert**; Die Adresse 255=\$FF ist eine Masteradresse mit der alle KIT160 angesprochen werden.

Beispiel: \$1B \$4B \$44 \$01

alle Befehle werden für das KIT160 mit der Adresse \$01 ab sofort ignoriert.

ESC L n1 mst**Text-Modus einstellen**

Der Verknüpfungsmodus **n1** und das Muster **mst** wird für die Textfunktion Zeichenkette ausgeben **ESC Z** eingestellt.

Beispiel: \$1B \$4C \$03 \$03

stellt den Verknüpfungsmodus für alle folgenden Textfunktionen auf graue Zeichen (Muster 3 = 50% Grau) invertiert mit dem Hintergrund.

Verknüpfungsmodus n1:

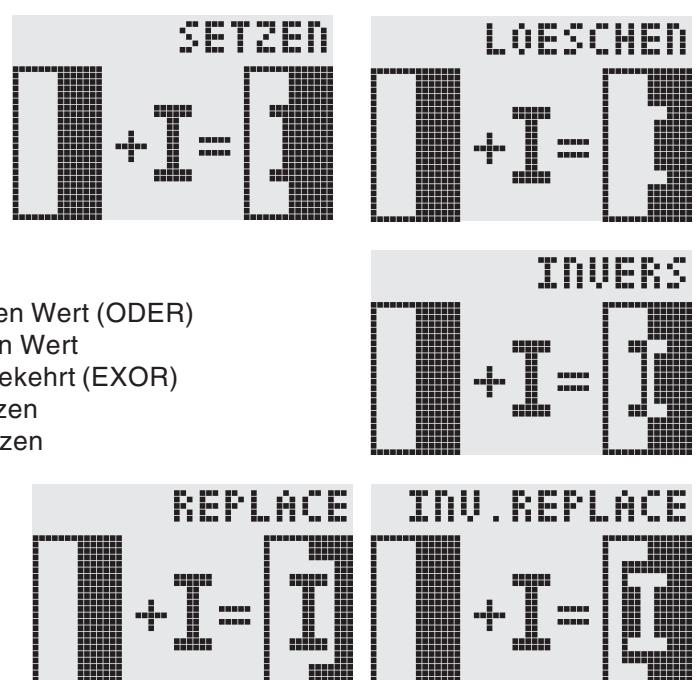
1 = setzen: schwarze Pixel ohne Rücksicht auf den vorigen Wert (ODER)

2 = löschen: weißes Pixel ohne Rücksicht auf den vorigen Wert

3 = invers: aus schwarzen Pixeln werden Weiße und umgekehrt (EXOR)

4 = replace: Hintergrund löschen und schwarze Pixel setzen

5 = invers replace: Hintergrund füllen und weiße Pixel setzen



ESC M N/T/P n1

Makro aufrufen

Das **N**=Normal-Makro, **T**=Touch-Makro oder **P**=Port-Makro mit der Nummern**n1** (0..255) wird aufgerufen.

Beispiel: \$1B \$4D \$4E \$0F

Das (Normal)Makro mit der Nummer 15 wird ausgeführt.

ESC M A/J n1 n2 n3

Makros automatisch ausführen

Die Normal-Makros mit den Nummern**n1** bis**n2** werden automatisch allen3/10 Sekunden aufgerufen.

A=zyklischer Aufruf (z.B. 1,2,3,4,1,2,3,4 usw.); **J**=Pingpong Aufruf (z.B. 1,2,3,4,3,2,1,2,3,4 usw.).

Die automatische Ausführung wird beendet: - wenn ein Zeichen von der RS-232 Schnittstelle empfangen wird.

- eine Touchberührung automatisch ein Touchmakro ausführt.

- oder eine Eingangsänderung ein Portmakro ausführt

Beispiel: \$1B \$4D \$41 \$01 \$03 \$05

Die Makros mit den Nummern 1, 2 und 3 werden automatisch mit einer Pause vom 1/2 Sekunde ausgeführt.

ESC N H/Vx1 y1 nr Text... NUL

Menü darstellen

Ein Menü wird definiert und mit dem aktuellen Font dargestellt. Der Hintergrund der Menübox wird automatisch im Clipboard gesichert (**evtl. vorhandener Clipboardinhalt geht verloren !**):

H=horizontales Menü an **x1,y1** (linke obere Ecke) oder **V**=vertikales Menü (90°

gedreht) an **x1,y1** (rechte obere Ecke). **nr**=aktuell invertierter Eintrag;

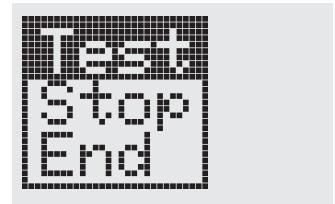
Text...=Zeichenkette mit den Einträgen. Die einzelnen Einträge werden durch das Zeichen '|' (=7C) getrennt. die Zeichenkette muss mit **NUL**= \$00 beendet werden

Beispiel 1 Horizontale Menü:

\$1B \$4E \$48 \$02 \$02 \$01

\$54 \$65 \$73 \$74 \$7C \$53 \$74 \$6F \$70 \$7C \$45 \$6E \$64 \$00

definiert ein horizontales Menü mit den Einträgen "Test", "Stop" und "End" an der Position 2,2. Der 1. Eintrag ist invertiert.

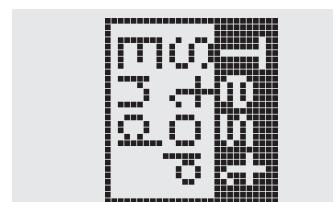


Beispiel 2 Vertikales Menü:

\$1B \$4E \$56 \$28 \$01 \$01

\$54 \$65 \$73 \$74 \$7C \$53 \$74 \$6F \$70 \$7C \$45 \$6E \$64 \$00

definiert ein vertikales Menü mit den Einträgen "Test", "Stop" und "End" an der Position 40,1. Der 1. Eintrag ist invertiert.



ESC N N/P

nächster/vorheriger Menü-Eintrag

N=der nächste oder **P**=der vorherige Menüeintrag wird invertiert. Falls schon der letzte/erste Eintrag invertiert ist dann wird der Befehl ignoriert.

Beispiel: \$1B \$4E \$4E Der nächste Menüeintrag wird invertiert.

ESC N I

Menübox invers darstellen

Die gesamte Menübox wird invertiert.

Beispiel: \$1B \$4E \$49

ESC N S

Menü beenden und senden

Das dargestellte Menü wird vom Display entfernt und der Hintergrund vom Clipboardinhalt ersetzt der aktuell ausgewählte Eintrag wird als Nummer (1..max. Eintrag) über die RS 232 Schnittstelle gesendet.

Beispiel: \$1B \$4E \$53

ESC N M n1

Menü beenden und Makro aufrufen

Das dargestellte Menü wird vom Display entfernt und der Hintergrund vom Clipboardinhalt ersetzt. Ist Eintrag 1 ausgewählt so wird das (Normal)Makro mit der Nummer **n1** aufgerufen, für Eintrag 2 das Makro **n1+1** usw.

Beispiel: \$1B \$4E \$4D \$0A

ESC N A

Menü abbrechen

Das dargestellte Menü wird vom Display entfernt und der Hintergrund vom Clipboardinhalt ersetzt.

Beispiel: \$1B \$4E \$41

ELECTRONIC ASSEMBLY**ESC O n1 n2**

Der Cursor wird für den Terminal-Betrieb auf Spalte **n1** und Zeile **n2** gesetzt. Der Ursprung links oben ist (1,1).

Beispiel: \$1B \$4F \$03 \$05

setzt den Cursor auf die 3. Spalte in Zeile 5.

Cursor positionieren**ESC P x1 y1**

Ein Pixel wird an der Koordinate **x1,y1** unter Beachtung des eingestellten Grafikmodus 'ESC V' (setzen / löschen / invertieren) gesetzt.

Beispiel: \$50 \$11 \$0D

setzt den Pixel an der Koordinate 17,13.

Punkt setzen**ESC Q C n1**

n1=1: der Cursor wird eingeschaltet, er blinkt an der aktuellen Zeichenposition im Terminal.

n1=0: der Cursor wird ausgeschaltet.

Beispiel: \$1B \$51 \$43 \$01

Der Cursor wird eingeschaltet.

Cursor EIN/AUS**ESC Q D x1 y1 x2 y2****Blinkbereich definieren**

Der Bereich von der linken oberen Ecke **x1,y1** bis zur rechten unteren Ecke **x2,y2** wird als automatischer Blinkbereich festgelegt. Zugleich wird die Blinkfunktion gestartet. Der Terminal-Cursor wird dadurch deaktiviert.

Beispiel: \$1B \$51 \$44 \$00 \$0F \$07 \$10

Definiert den Blinkbereich von 0,15 nach 7,16.

ESC Q Z n1**Blinkzeit einstellen**

Stellt die Blinkzeit auf **n1** (=1..15) zehntel Sekunden ein. Bei **n1= 0** wird die Blinkfunktion deaktiviert und der Orginal Bildschirm wieder hergestellt.

Beispiel: \$1B \$51 \$5A \$03

stellt die Blinkzeit auf 0,3 Sekunden ein.

ESC Q M I**Blinkmodus Invers**

Der definierte Blinkbereich wird zyklisch mit der eingestellten Blinkzeit automatisch invertiert. Zugleich wird die Blinkfunktion gestartet.

Beispiel: \$1B \$51 \$49

Der Blinkmodus invers wird eingestellt.

ESC Q M mst**Blinkmodus Blockcursor**

Der definierte Blinkbereich wird gesichert. Mit der eingestellten Blinkzeit wird zyklisch zwischen dem Orginal Bereich und dem Muster **mst** (=0..7) umgeschaltet. Dadurch kann z.B ein Blockcursor simuliert werden (**mst=1** schwarz) oder ein blinkendes Wort angezeigt werden (**mst=0** weiss). Zugleich wird die Blinkfunktion gestartet.

Beispiel: \$1B \$51 \$43 \$00

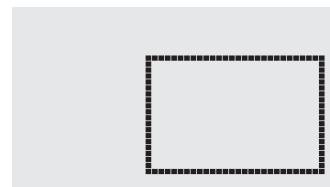
Der Blinkmodus Blockcursor mit dem Muster weiss wird eingestellt. Dadurch wird erreicht, daß der eingestellte Bereich auf weissem Hintergrund blinkt.

ESC R R x1 y1 x2 y2**Rechteck zeichnen**

Ein Rechteck wird von der linken oberen Ecke **x1,y1** bis zur rechten unteren Ecke **x2,y2** unter Beachtung des eingestellten Grafikmodus 'V' (setzen / löschen / invers) gezeichnet. Der Inhalt des Rechtecks wird dabei nicht verändert. Vergleiche 'ESC R O' Box zeichnen.

Beispiel: \$1B \$52 \$52 \$15 \$08 \$30 \$25

zeichnet ein Rechteck von 21,8 nach 48,37.



EA KIT160-7

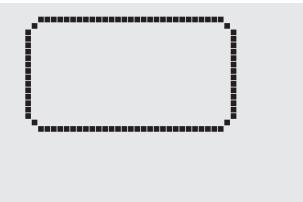
ELECTRONIC ASSEMBLY

ESC R N x1 y1 x2 y2

Ein Rechteck mit abgerundeten Ecken wird von der linken oberen Ecke **x1,y1** bis zur rechten unteren Ecke **x2,y2** unter Beachtung des eingestellten Grafikmodus 'V' (setzen / löschen / invers) gezeichnet. Der Inhalt des Rundecks wird nicht verändert. Vergleiche 'ESC R J' Rundbox zeichnen.

Beispiel: \$1B \$52 \$4E \$06 \$02 \$26 \$13
zeichnet ein Rundeck von 6,2 nach 38,19.

Rundeck zeichnen



ESC R L x1 y1 x2 y2

Der Bereich von der linken oberen Ecke **x1,y1** bis zur rechten unteren Ecke **x2,y2** wird gelöscht.

Beispiel: \$1B \$44 \$53 \$1B \$52 \$4C \$06 \$04 \$28 \$19
das Display wird mit **ESC D S** gefüllt und dann von 6,4 nach 40,25 gelöscht .

Bereich löschen

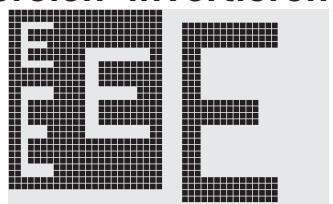


ESC R I x1 y1 x2 y2

Der Bereich von der linken oberen Ecke **x1,y1** bis zur rechten unteren Ecke **x2,y2** wird invertiert (aus schwarzen Pixeln werden Weiße und umgekehrt).

Beispiel: \$1B \$52 \$49 \$00 \$00 \$17 \$1B
invertiert bei vorhandenem Displayinhalt aus dem Beispiel "Font einstellen" den Bereich von 0,0 nach 23,27.

Bereich invertieren

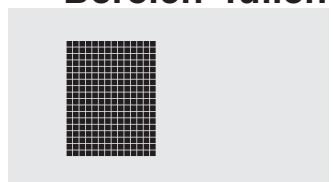


ESC R S x1 y1 x2 y2

Der Bereich von der linken oberen Ecke **x1,y1** bis zur rechten unteren Ecke **x2,y2** wird gefüllt (auf schwarze Pixel gesetzt).

Beispiel: \$1B \$52 \$53 \$09 \$05 \$16 \$16
setzt den Bereich von 9,5 nach 22,22 auf schwarz.

Bereich füllen

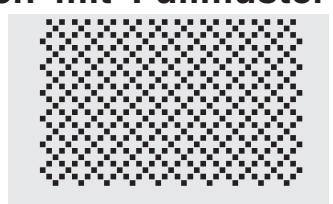


ESC R M x1 y1 x2 y2 mst

Ein rechteckiger Bereich wird von der linken oberen Ecke **x1,y1** bis zur rechten unteren Ecke **x2,y2** mit dem Muster **mst** unter Beachtung des eingestellten Grafikmodus 'ESC V' (setzen/löschen/invertieren/replace/invers replace) gezeichnet.

Beispiel: \$1B \$52 \$4D \$05 \$01 \$2D \$1A \$07
zeichnet das Muster 7=45°Kreuz von 5,1 nach 45,26.

Bereich mit Füllmuster

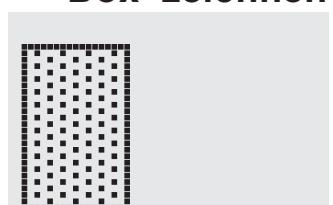


ESC R O x1 y1 x2 y2 mst

Ein Rechteck wird von der linken oberen Ecke **x1,y1** bis zur rechten unteren Ecke **x2,y2** mit dem Muster **mst** gezeichnet. Der Hintergrund der Box wird dabei gelöscht. Vergleiche 'ESC R R' Rechteck zeichnen.

Beispiel: \$1B \$52 \$4F \$02 \$05 \$12 \$1E \$02
zeichnet eine Box von 2,5 nach 18,30 mit dem Muster 2=25%Grau.

Box zeichnen



ESC R J x1 y1 x2 y2 mst

Ein Rechteck mit abgerundeten Ecken wird von der linken oberen Ecke **x1,y1** bis zur rechten unteren Ecke **x2,y2** mit dem Muster **mst** gezeichnet. Der Hintergrund wird dabei gelöscht. Vergleiche 'ESC R N' Rundeck zeichnen.

Beispiel: \$1B \$52 \$4A \$07 \$03 \$23 \$16 \$03
zeichnet eine Rundbox von 7,3 nach 35,22 mit dem Muster 3=50%Grau.

Rundbox zeichnen



ESC S anz daten...

Die nachfolgenden **anz** (1..255, 0=256) Bytes werden auf der seriellen Schnittstelle ausgegeben.

Beispiel: \$1B \$53 \$04 \$54 \$45 \$53 \$54

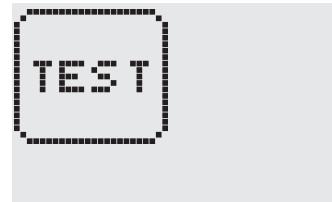
Das Wort 'TEST' wird über die RS-232C Schnittstelle gesendet.

Bytes über RS-232 senden

ELECTRONIC ASSEMBLY

ESC T H/Vf1 f2 ret frm text... NUL Touch-Taste definieren

Ein Touch-Taste wird definiert und mit dem aktuellen Font beschriftet. **H**=horizontale oder **V**=vertikale Beschriftung (90° gedreht). Mehrere Touch-Felder können als eine einzige Touch-Taste zusammengefasst werden **f1**=linkes obere Touchfeld, **f2**=rechtes untere Touchfeld der neuen Touchtaste. Diese Touchtaste wird mit **ret** ein Returncode zugewiesen (1..255). Beim Berühren der Touchtaste wird dann das Touchmakro mit der Nummer **ret** aufgerufen oder, falls kein Touchmakro definiert ist, dieser Returncode über die RS232 gesendet. Mit **frm** wird die Darstellung der Touchtaste festgelegt (frm=0: nichts zeichnen; frm=1: Touchtaste löschen; frm=2: Touchtaste löschen und mit Rahmen zeichnen). **text...**=Zeichenkette mit der Beschriftung (wird immer in der Touchtaste zentriert). Die Beschriftung kann auch mehrzeilig sein, die einzelnen Zeilen werden durch das Zeichen 'l' (=7C) getrennt. Die Zeichenkette muss mit **NUL**= \$00 beendet werden. Siehe Beispiel auf Seite 3.



Beispiel 1 Horizontale Touchtaste:

\$1B \$54 \$48 \$01 \$01 \$41 \$02 \$54 \$45 \$53 \$54 \$00

definiert eine horizontale Touchtaste (nur Feld Nr. 1) mit dem Returncode 65='A'. Die Touchtaste wird mit Rahmen gezeichnet und mit dem Wort 'TEST' beschriftet.



Beispiel 2 Vertikale Touchtaste:

\$1B \$54 \$56 \$02 \$02 \$42 \$02 \$54 \$45 \$53 \$54 \$00

definiert eine vertikale Touchtaste (nur Touchfeld Nr. 2) mit dem Returncode 66='B'. Die Touchtaste wird mit Rahmen gezeichnet und mit dem Wort 'TEST' beschriftet.

ESC T P/R

Touchfelder Vorbelegen/Reset

Alle 60 Touchfleder werden mit **P**=aufsteigendem Returncode belegt (1..60) oder **R**=rückgesetzt alle Touchfelder erhalten den Returncode 0 d.h sie sind deaktiviert.

Beispiel: \$1B \$54 \$52

Alle Touchfelder sind nach diesem Befehl deaktiviert und werden nicht mehr erkannt.

ESC T I/S n1

Touchtasten Reaktion

Mit diesem Befehlen wird die automatische Reaktion des Touchpanels beim Berühren eingestellt. Es können beide Reaktionen gleichzeitig aktiviert werden.

I=automatisches Invertieren beim Berühren der Touchtaste **n1=0**: AUS oder **n1=1**: EIN.

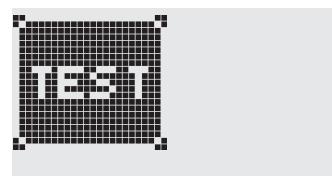
S=automatischer Signalton beim Berühren **n1=0**: AUS oder **n1=1**: EIN

Beispiel: \$1B \$54 \$49 \$01

Nach diesem Befehl ertönt der Summer beim Berühren einer Touchtaste.

ESC T M ret

Touchtaste manuell invertieren



Die Touchtaste mit dem Returncode **ret** kann mit diesem Befehl manuell invertiert werden.

Beispiel: \$1B \$54 \$4D \$41

Die Touchtaste aus obigen Beispiel mit dem Returncode 65='A' wird invertiert.

ESC T A n1

(Touch)Tastenabfrage Ein/Aus

Die (Touch)Tastenabfrage wird mit diesem Befehl eingestellt:

n1=0: Tastenabfrage ist komplett abgeschaltet: keine Touchmakros, keine manuelle Tastenabfrage möglich.

n1=1: Tastenabfrage ist aktiv: Tastendrücke lösen Touchmakros aus oder werden über RS232 gesendet.

n1=2: Tastenabfrage ist aktiv: Tastendrücke lösen Touchmakros aus, müssen manuell abgefragt werden.

Beispiel: \$1B \$54 \$41 \$02

Die (Touch)Tastenfrage wird aktiviert, die Tastendrücke werden nicht automatisch über RS232 gesendet, sie müssen manuell mit dem Befehl **ESC T W** angefordert werden.

ESC T W

Touchtaste manuell abfragen

Der Returncode der momentan gedrückten Touchtaste wird auf der RS232 gesendet.

Beispiel: \$1B \$54 \$57

ESC U E x1 y1 n1

Bild aus EEPROM

Das gespeicherte Bild im EEPROM mit der Nummer **n1** (0..255) wird an die Koordinate **x1,y1** geladen.

Beispiel: \$1B \$55 \$45 \$02 \$03 \$0E

Das Bild Nummer 14 aus dem EEPROM wird an Koordinate 2,3 angezeigt.

ESC U L x1 y1 daten...

Ein Bild wird an die Koordinate **x1,y1** geladen.

daten...: - 1 Byte für die Bildbreite in Pixeln

- 1 Byte für die Bildhöhe in Pixeln

- Bilddaten: Anzahl = ((Breite+7) / 8) * Höhe Bytes.

1 Byte steht für 8 waagrechte Pixel am Bildschirm; 0=weiß, 1=schwarz;
MSB: links, LSB: rechts; das Bild ist von oben nach unten abgelegt.

Das Programm BMP2BLH.EXE auf der als Zubehör erhältlichen Diskette
EA DISK240 erzeugt aus monochromen Windows-Bitmap- Grafiken
(*.BMP) die Bilddaten inkl. der Angabe von Breite und Höhe.

Beispiel: \$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

lädt das nebenstehende Bild an die Koordinate 9,4.

Bild Upload



	Bit Nr. 7 6 5 4 3 2 1 0	Bit Nr. 7 6 5 4
Byte 1	[Binary Data]	[Binary Data]
Byte 2	[Binary Data]	[Binary Data]
Byte 3	[Binary Data]	[Binary Data]
Byte 4	[Binary Data]	[Binary Data]
Byte 5	[Binary Data]	[Binary Data]
Byte 6	[Binary Data]	[Binary Data]
Byte 7	[Binary Data]	[Binary Data]
Byte 8	[Binary Data]	[Binary Data]
Byte 9	[Binary Data]	[Binary Data]
Byte 10	[Binary Data]	[Binary Data]
Byte 11	[Binary Data]	[Binary Data]
Byte 12	[Binary Data]	[Binary Data]
Byte 13	[Binary Data]	[Binary Data]
Byte 14	[Binary Data]	[Binary Data]
Byte 15	[Binary Data]	[Binary Data]
Byte 16	[Binary Data]	[Binary Data]
Byte 17	[Binary Data]	[Binary Data]
Byte 18	[Binary Data]	[Binary Data]
Byte 19	[Binary Data]	[Binary Data]
Byte 20	[Binary Data]	[Binary Data]
Byte 21	[Binary Data]	[Binary Data]
Byte 22	[Binary Data]	[Binary Data]
Byte 23	[Binary Data]	[Binary Data]
Byte 24	[Binary Data]	[Binary Data]

ESC V n1

Einstellen des Verknüpfungsmodus **n1** für folgende
Grafikfunktionen: Punkt setzen ESC P, Gerade zeichnen
ESC G, Gerade weiter zeichnen ESC W, Rechteck
zeichnen ESC R R, Rundeck zeichnen ESC R N, Bereich
mit Füllmuster ESC R M.

Als Beispiel wird nebenan ein Rechteck mit den
Verknüpfungsmodi setzen, löschen und invers auf einen
vorhandenen Hintergrund gezeichnet.

Verknüpfungsmodus n1:

1=setzen: schwarze Pixel ohne Rücksicht auf den vorigen Wert (ODER)

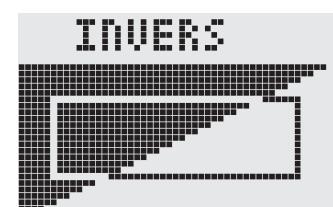
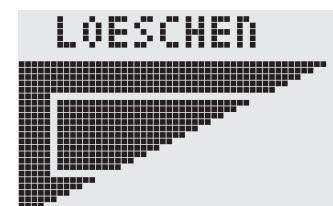
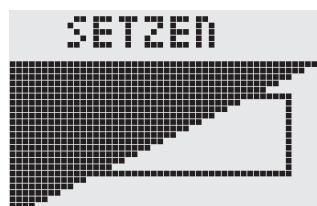
2=löschen: weißes Pixel ohne Rücksicht auf den vorigen Wert

3=invers: aus schwarzen Pixeln werden Weiße und umgekehrt (EXOR)

4=replace: Hintergrund löschen und Pixel setzen; nur Bereich mit Füllmuster 'mst'

5=invers replace: Hintergrund füllen, Pixel löschen; nur Bereich mit Füllmuster 'mst'

Grafik-Modus einstellen



ESC W x1 y1

Zieht eine Gerade vom zuletzt gezeichneten Geradenende bzw. Punkt bis nach **x1,y1**
unter Beachtung des eingestellten Grafik-Modus 'V'

Beispiel: \$1B \$47 \$00 \$00 \$10 \$04

\$1B \$57 \$16 \$1B

\$1B \$57 \$30 \$0F

Eine Gerade von 0,0 nach 16,4 zeichnen. Dann weiter nach 22,27 und nach 48,15.

ESC X n1

Mit diesem Befehl wird das KIT160 für **n1**/10 Sekunden angehalten.

Beispiel: \$1B \$58 \$0A

Nach diesem Befehl wartet das KIT160 eine Sekunde bevor der nächste Befehl abgearbeitet wird.

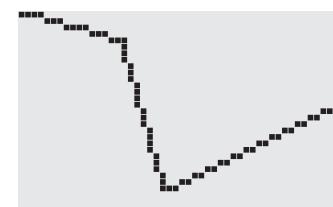
ESC Y R n1

Liest den Eingangs-Port (**n1**=1..8 = IN1..IN8) ein. Wenn **n1**=0, werden alle Eingänge als 8-Bit Binärwert
eingelese (MSB:IN8...In1:LSB); Siehe Applikation auf Seite 5. Achtung: Die Optokoppler invertieren die
Eingangslogik (Eingang offen: 1). Der Befehl "ESC Y 1 1" stellt dies richtig (Eingang offen: 0).

Beispiel: \$1B \$59 \$52 \$03

liest den Port IN3 ein. Ergebnis wird über RS232 gesendet.

Gerade weiterzeichnen



Warten / Pause

Eingabe-Port lesen

ELECTRONIC ASSEMBLY**ESC Y W n1 n2**

Ändert den Ausgabe Port (**n1**=1..8 = OUT1..OUT8) auf den Wert **n2** (0=L-Pegel; 1=H-Pegel; 2=Port invertieren). Wenn **n1**=0, werden alle Ausgänge als Binärwert **n2** (MSB:OUT8...OUT1:LSB) ausgegeben; Siehe Applikation auf Seite 5.

Beispiel: \$1B \$59 \$57 \$02 \$01
schaltet den Ausgabe Port OUT2 auf H-Pegel.

ESC Y A n1**automatische Portabfrage EIN/AUS**

Jede Änderung am Eingabeport (8-Bit Binärwert IN8..IN1) kann ein Portmakro (0..255) aufrufen. Mit diesem Befehl wird die automatische Portabfrage **n1**=1 aktiviert oder mit **n1**=0 deaktiviert. Nach dem Einschalten wird der aktuelle Portzustand gelesen und sofort das dazugehörige Portmakro ausgeführt.

Beispiel: \$1B \$59 \$41 \$01

Die automatische Portabfrage wird aktiviert und das anliegende Portmakro wird ausgeführt.

ESC Y I n1**Eingabe-Port invers**

Mit diesem Befehl kann die Logik des Eingabe-Ports umgekehrt werden (**n1**=0 normal oder mit **n1**=1 invers).

Sinnvoll z.B. bei den Optokoppler Eingängen.

Beispiel: \$1B \$59 \$49 \$01

Die Logik Eingabe-Ports wird invertiert.

ESC Y L n1**Beleuchtung Ein-/Ausschalten**

Die Hintergrundbeleuchtung wird **n1**=0 ausgeschaltet, **n1**=1 dauerhaft eingeschaltet; **n1**=2 umgekehrt: EIN->AUS bzw. AUS->EIN oder mit **n1**=3..255 für **n1**/10 Sekunden lang eingeschaltet. Wird ein Widerstand (z.B. 47 Ohm) bei R Off (siehe S. 4 Lageplan) eingelötet so ist die Beleuchtung nicht aus sondern dunkler (2 Helligkeitstufen!).

Beispiel: \$1B \$4A \$64

nach diesem Befehl leuchtet die Hintergrundbeleuchtung 10s lang.

ESC Z L/Z/R x1 y1 text... NUL**Zeichenkette horizontal**

Schreibt die Zeichenkette **text...**, **L**=Linksbündig, **Z**=Zentriert oder **R**=Rechtsbündig an der Koordinate **x1** unter Beachtung des eingestellten Textmodus **ESC L**. Es können auch mehrzeilige Texte ausgegeben werden, die einzelnen Zeilen sind durch das Zeichen ';' (=7C) getrennt. Die Zeichenkette muss mit **NUL**= \$00 beendet werden. Die Koordinate **y1** ist die Oberkante der 1. Zeile.

Beispiel 1: schreibt an 0,0 linksbündig den Text "Left|Ok"
\$1B \$5A \$4C \$00 \$00 \$4C \$65 \$66 \$74 \$7C \$4F \$6B \$00

Beispiel 2: schreibt an 25,0 zentriert "Center|Ok"
\$1B \$5A \$5A \$19 \$00 \$43 \$65 \$6E \$74 \$65 \$72 \$7C \$4F \$6B \$00

Beispiel 3: schreibt an 49,0 rechtsbündig "Right|Ok"
\$1B \$5A \$52 \$31 \$00 \$52 \$69 \$67 \$68 \$74 \$7C \$4F \$6B \$00

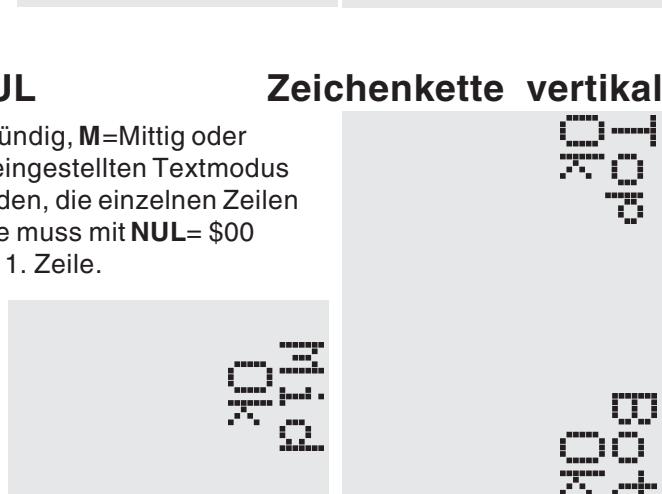
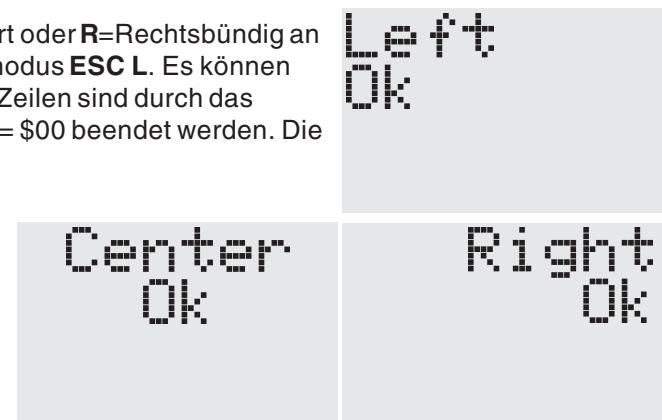
ESC Z O/M/U x1 y1 text... NUL**Zeichenkette vertikal**

Schreibt die Zeichenkette **text...** um 90° gedreht, **O**=Obenbündig, **M**=Mittig oder **U**=Untenbündig an der Koordinate **y1** unter Beachtung des eingestellten Textmodus **ESC L**. Es können auch mehrzeilige Texte ausgegeben werden, die einzelnen Zeilen sind durch das Zeichen ';' (=7C) getrennt. Die Zeichenkette muss mit **NUL**= \$00 beendet werden. Die Koordinate **x1** ist die rechte Kante der 1. Zeile.

Beispiel 1: schreibt an 49,0 obenbündig "Top|Ok"
\$1B \$5A \$4F \$31 \$00 \$54 \$6F \$70 \$7C \$4F \$6B \$00

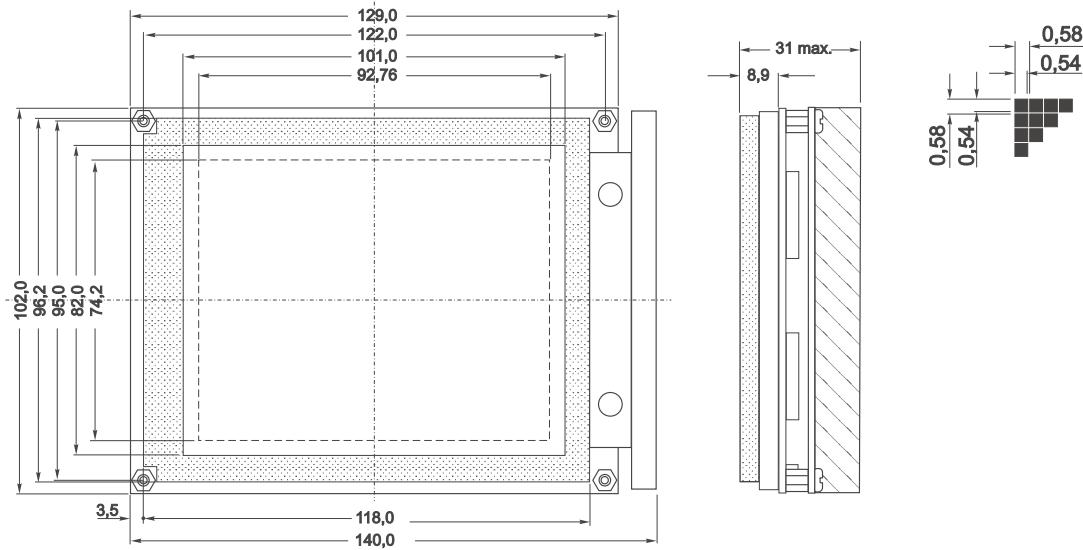
Beispiel 2: schreibt an 49,15 mittig "Mid|Ok"
\$1B \$5A \$4D \$31 \$0F \$4D \$69 \$64 \$7C \$4F \$6B \$00

Beispiel 3: schreibt an 49,31 untenbündig "Bot|Ok"
\$1B \$5A \$55 \$31 \$1F \$42 \$6F \$74 \$7C \$4F \$6B \$00

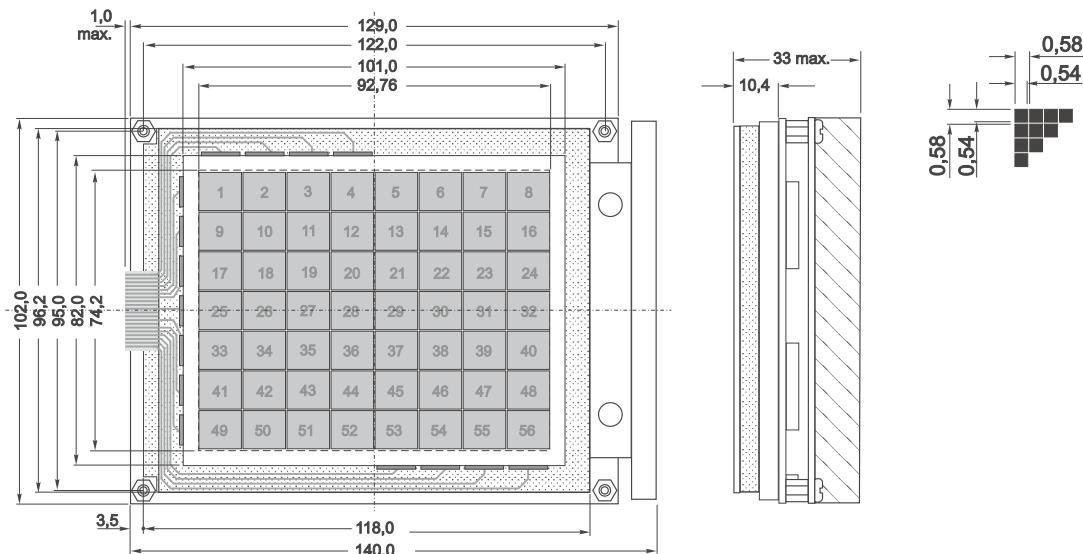


EA KIT160-7

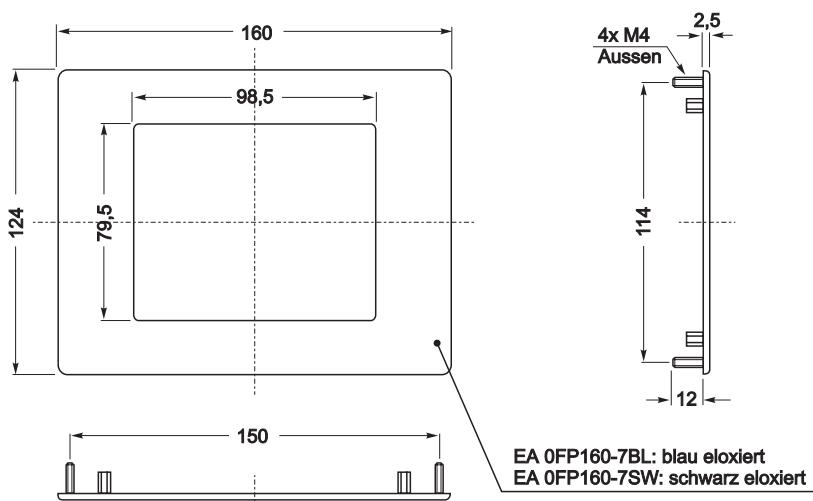
ABMESSUNGEN OHNE TOUCH PANEL



ABMESSUNGEN MIT TOUCH PANEL



FRONTPANEL EA OFP160-7



PANEL CUT OUT

