

# FERTIG AUFGEBAUTE GRAFIKEINHEIT 128x64 MIT 3 FONTS, ZOOM UND LED-BACKLIGHT

**Bargraph Funktion**

**Text+Grafik mischen**

**Füllmuster**

**Font Zoom**



Abmessung:  
84x60x24 mm

**Bild Download**

## TECHNISCHE DATEN

- \* 3 VERSCHIEDENE FONTS INTEGRIERT
- \* ZOOM FUNKTION ALLER FONTS (2-, 3- UND 4-FACH)
- \* PROGRAMMIERUNG ÜBER DIVERSE EINGebaUTE GRAFIKFUNKTIONEN:
- \* GERADE, PUNKT, BEREICH, UND/ODER/EXOR, BARGRAPH, FÜLLMUSTER...
- \* TEXT UND GRAFIK MISCHEN
- \* 4-16 FREI DEFINIERBARE ZEICHEN (JE NACH GRÖßE)
- \* ANSTEUERUNG ÜBER RS-232 / CMOS-PEGEL
- \* BAUDRATE PROGRAMMIERBAR VON 1200 BIS 115.200 BAUD
- \* KEINE TIMINGPROBLEME BEI SCHNELLEM BUSSYSTEM
- \* 8 DIGITALE I/O'S ZUR FREIEN VERWENDUNG
- \* +5V / typ. 200mA (INKL. LED-BELEUCHTUNG)
- \* HARDWARE CODIERUNG VON BIS ZU 4 ADRESSEN
- \* DOWNLOAD VON KONVERTIERTEN WINDOWS-BMP GRAFIKEN

## ZUBEHÖR

- \* DISKETTE FÜR PC MIT KONVERTIERSOFTWARE FÜR WINDOWS-BMP GRAFIKEN: **EA DISK9719**
- \* RS-232 KABEL MIT D-SUB 9 (FEMALE) FÜR TEST AM PC: **EA KV24-9B**

## BESTELLBEZEICHNUNG

GRAFIKEINHEIT 128x64, 3 FONTS, RS-232  
DIP-SCHALTER STATT LÖTBRÜCKEN (z.B. BAUDRATEN)  
SIGNALGEBER AN I/O5 (I/O'S NICHT MEHR NUTZBAR)

**EA GE128-6N3V24**  
**EA OPT-DIP6**  
**EA OPT-SUMMER**

**ELECTRONIC  
ASSEMBLY** GMBH

LOCHHAMER SCHLAG 17 · D- 82 166 GRÄFELFING  
TELEFON 089/854 1991 · TELEFAX 089/854 1721

### ALLGEMEINES

Das Grafik-LCD EA GE128-6N3V24 ist für kleine bis mittlere Stückzahlen konzipiert. Aufgrund seiner kleinen Außenabmessungen, dem sehr guten Supertwistkontrast und der einfachen Programmierung ist es innerhalb weniger Stunden möglich, an nahezu jedes Prozessorsystem ein informativen und optisch ansprechenden Bildschirm anzuschließen. Die Ansteuerung erfolgt über die Standard Schnittstelle RS-232. Das Display enthält komplette Grafikroutinen zur Displayausgabe sowie verschiedenste Schriftgrößen.

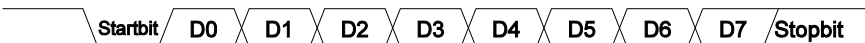
Die Programmierung erfolgt über hochsprachenähnliche Grafikbefehle; die zeitraubende Programmierung von Zeichensätzen und Grafikroutinen entfällt hier völlig. Doch nicht nur der Entwicklungsaufwand reduziert sich drastisch. Auch in der Serie sind die folgende Vorteile spürbar:

- keine Timingprobleme bei schnellem Prozessorbus
- keine Speicherplatzprobleme (Arbeitsspeicher und Speicher für den Zeichensatz v.a. bei  $\mu\text{C}$ )
- keine zeitaufwendigen Grafikberechnungen welche die Prozessorgeschwindigkeit belasten.

Es sind keine Treiber, Dekoder oder Portbausteine erforderlich. Im einfachsten Fall erfolgt die Displayansteuerung über nur 1 Leitung RxD.

### HARDWARE

Das Display ist für +5V Betriebsspannung ausgelegt. Die Datenübertragung erfolgt seriell asynchron im RS-232 Format mit echten V.24 Pegeln ( $\pm 10\text{V}$ ) oder über 5V CMOS Pegeln. Das Übertragungsformat ist fest auf 8 Datenbits, 1 Stopbit, no Parity eingestellt. Die Baudrate kann über 3 Lötbrücken von 1200 Baud bis zu 115.200 Baud ausgewählt werden. Handshakeleitungen RTS und CTS stehen zur Verfügung. Bei kleinen Datenmengen ist eine Auswertung nicht erforderlich.

Datenformat: 

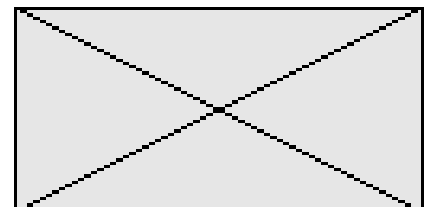
Zusätzlich sind an der Lötungenleiste J3 8 I/O-Ports zur freien Verwendung vorhanden. Diese können sowohl als Aus- als auch als Eingänge individuell geschaltet werden. Mögliche Anwendungen dafür sind das Schalten eines Transistors/Relais ( $I_{L_{\max}} = 10\text{mA}$ ) oder das Einlesen von Tasten/Schaltern. Das dafür notwendige Pullup Widerstandsnetzwerk kann auf der Platine eingelötet werden.

### SOFTWARE

Die Programmierung der Grafikeinheit erfolgt über Befehle wie z.B. Zeichne ein Rechteck von (0,0) nach (64,15). Der Ursprung liegt im linken oberen Eck des Displays. Über die serielle Schnittstelle müssen somit folgende Bytes gesendet werden: \$52 \$00 \$00 \$40 \$0F. Zeichenketten lassen sich ebenso pixelgenau plazieren. Das Mischen von Text und Grafik ist jederzeit möglich. Es können 3 verschiedene Zeichensätze verwendet werden. Jeder Zeichensatz kann wiederum 2-, 3- und 4-fach gezoomt werden. Mit dem größten Zeichensatz 16x8 lassen sich somit bei 4-fach Zoom (=64x32) bildschirmfüllende Worte und Zahlen darstellen.

### TESTMODE

Solange die Lötbrücke 6 (Pin RTS5) nach dem Power-On oder Reset geschlossen ist (mit GND verbunden ist), befindet sich das Display im Testmode: es blinkt ein Rechteck mit zwei diagonalen Linien. Wird die Lötbrücke geöffnet, dann kehrt das Display in den Normalbetrieb zurück. Das Testbild ist aber immer noch zu sehen.



### INTEGRIERTE FONTS

Im der Grafikeinheit EA GE128-6N3V24 sind bereits 3 Zeichensätze integriert (Font1: 4x6 Pixel; Font2: 6x8 Pixel und Font 3 8x16 Pixel). Jeder Zeichensatz kann in 1-, 2-, 3- oder 4-facher Höhe verwendet werden. Unabhängig davon läßt sich auch die Breite verdoppeln, verdreifachen oder vervierfachen. Zusätzlich können 4-16 eigene Zeichen definiert werden, die solange erhalten bleiben, bis die Versorgungsspannung abgeschaltet wird. (Siehe Befehl 'E').

+ Lower Upper	\$0 (0)	\$1 (1)	\$2 (2)	\$3 (3)	\$4 (4)	\$5 (5)	\$6 (6)	\$7 (7)	\$8 (8)	\$9 (9)	\$A (10)	\$B (11)	\$C (12)	\$D (13)	\$E (14)	\$F (15)
\$20 (dez: 32)		!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
\$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
\$40 (dez: 64)	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	
\$50 (dez: 80)	p	q	r	s	t	u	v	w	x	y	z	[	\	]	^	_

Font 1

+ Lower Upper	\$0 (0)	\$1 (1)	\$2 (2)	\$3 (3)	\$4 (4)	\$5 (5)	\$6 (6)	\$7 (7)	\$8 (8)	\$9 (9)	\$A (10)	\$B (11)	\$C (12)	\$D (13)	\$E (14)	\$F (15)
\$20 (dez: 32)		!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
\$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
\$40 (dez: 64)	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	
\$50 (dez: 80)	p	q	r	s	t	u	v	w	x	y	z	[	\	]	^	_
\$60 (dez: 96)	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
\$70 (dez: 112)	p	q	r	s	t	u	v	w	x	y	z	[	\	]	^	_
\$80 (dez: 128)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
\$90 (dez: 144)	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	
\$A0 (dez: 160)	p	q	r	s	t	u	v	w	x	y	z	[	\	]	^	_
\$B0 (dez: 176)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
\$C0 (dez: 192)	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	
\$D0 (dez: 208)	p	q	r	s	t	u	v	w	x	y	z	[	\	]	^	_
\$E0 (dez: 224)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
\$F0 (dez: 240)	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	

Jedes Zeichen kann pixelgenau plaziert werden. Texte und Grafiken können beliebig gemischt dargestellt werden. Auch mehrere verschiedene Schriftgrößen lassen sich gemeinsam darstellen.

Font 2

+ Lower Upper	\$0 (0)	\$1 (1)	\$2 (2)	\$3 (3)	\$4 (4)	\$5 (5)	\$6 (6)	\$7 (7)	\$8 (8)	\$9 (9)	\$A (10)	\$B (11)	\$C (12)	\$D (13)	\$E (14)	\$F (15)
\$20 (dez: 32)		!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
\$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
\$40 (dez: 64)	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	
\$50 (dez: 80)	p	q	r	s	t	u	v	w	x	y	z	[	\	]	^	_
\$60 (dez: 96)	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
\$70 (dez: 112)	p	q	r	s	t	u	v	w	x	y	z	[	\	]	^	_
\$80 (dez: 128)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
\$90 (dez: 144)	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	

Font 3

## ALLE GRAFIKFUNKTIONEN AUF EINEN BLICK

Die Grafikeinheit läßt sich über diverse eingebaute Befehle programmieren. Jeder Befehl beginnt mit einem Befehlsbuchstaben, gefolgt von einigen Parametern.

Befehlstabelle EA GE128-6N3V24											
Befehl							Anmerkung				
<b>Funktionen zur Textausgabe</b>											
Text-Modus	T	R L O U	n1	mst			R/L/O/U: Zeichenkette nach (R)echts,(L)inks,(O)ben, (U)nten schreiben; n1: Verknüpfungsmodus für Textausgabe 1=setzen; 2=löschen; 3=invers; 4=Replace; 5=Invers Replace; mst: Muster Nr. 0..7 verwenden;				
Font einstellen	F		n1	n2	n3			Font Nr. n1 einstellen; n1=1:4x6 Font; n1=2:6x8 Font; n2=3:8x16 Font n2+n3=Zoomfaktor (1..4); n2=X-Faktor; n3=Y-Faktor;			
ASCII-Zeichen setzen	A		x1	y1	n1			Das Zeichen n1 wird an Koordinate x1,y1 gesetzt. (Bezug links oben)			
Zeichenkette ausgeben	Z		x1	y1	...	NUL			Eine Zeichenkette (...) an x1,y1 ausgeben; Zeichen 'NUL' (\$00)=Ende		
Zeichen definieren	E		n1	daten ...				n1=Zeichen Nr.; daten=Anzahl Bytes je nach akt. Font			
<b>Grafik-Befehle mit Verknüpfungsmodus</b>											
Grafik-Modus	V		n1					n1: 1=setzen; 2=löschen; 3=invers; 4=Replace; 5=Invers Replace;			
Punkt setzen	P		x1	y1					Ein Pixel an die Koordinaten x1, y1 setzen		
Gerade zeichnen	G		x1	y1	x2	y2			Eine Gerade von x1,y1 nach x2,y2 zeichnen		
Gerade weiter zeichnen	W		x1	y1					Eine Gerade vom letzten Endpunkt bis x1, y1 zeichnen		
Rechteck zeichnen	R		x1	y1	x2	y2			Ein Rechteck zeichnen; x1,y1,x2,y2 = Gegenüberliegende Eckpunkte		
Rundeck zeichnen	N		x1	y1	x2	y2			Ein Rechteck mit runden Ecken zeichnen; x1,y1,x2,y2 = Eckpunkte		
Bereich m. Füllmuster	M		x1	y1	x2	y2	mst			Ein Bereich mit Muster mst (0..7) zeichnen; x1,y1,x2,y2 = Eckpunkte	
<b>sonstige Grafik-Befehle</b>											
Display löschen	D	L							Gesamten Displayinhalt löschen (auf weiß setzen);		
Display invertieren	D	I							Gesamten Displayinhalt invertieren;		
Display füllen	D	S							Gesamten Displayinhalt füllen; (auf schwarz setzen);		
Bereich löschen	L		x1	y1	x2	y2			Einen Bereich löschen; x1,y1,x2,y2 = Gegenüberliegende Eckpunkte		
Bereich invertieren	I		x1	y1	x2	y2			Einen Bereich invertieren; x1,y1,x2,y2 = Gegenüberliegende Eckpunkte		
Bereich füllen	S		x1	y1	x2	y2			Einen Bereich füllen; x1,y1,x2,y2 = Gegenüberliegende Eckpunkte		
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)	
Bargraph zeichnen	B		nr	wert					Den Bargraph mit der 'nr' (1..8) auf den neuen Benutzer-'wert' setzen		
Bildbereich Uploaden	U		x1	y1	daten ...				Einen Bildbereich nach x1,y1 laden; daten des Bildes siehe Bildaufbau		
<b>Kontroll- / Definitions-Befehle</b>											
Bargraph definieren	B	R L O U	nr	x1	y1	x2	y2	aw	ew	mst	Einen Bargraph nach L(inks), R(echts), O(ben), U(nten) mit der 'nr' (1..8) definieren. x1,y1,x2,y2 sind das umschließende Rechteck des Bargraphs. aw,ew sind die Werte für 0% und 100%. mst=Muster Nr. (0..7)
Display Control	C		n1							n1=0:Display Aus, Inhalt bleibt erhalten; n1=1:Display Ein, Inhalt sichtbar	
Selekt / Deselekt	K	S	n1							Kontroller mit Adresse n1 (n1=0..3; n1=4: alle) aktivieren	
Grafikkontroller		D								Kontroller mit Adresse n1 (n1=0..3; n1=4: alle) deaktivieren	
I/O-Port schreiben	Y		n1	n2					n1=0..7: I/O-Port n1 rücksetzen (n2=0); setzen (n2=1); invertieren (n2=2) n1=8: Alle 8 I/O-Ports I/O0..I/O7 als 8-Bit Binärwert einstellen		
<b>Sende-Befehle</b>											
Hardcopy	H		x1	y1	x2	y2			Es wird der angegebene Bildinhalt angefordert. Zuerst werden die Breite und Höhe in Pixel und dann die eigentlichen Bilddaten gesendet.		
I/O-Port lesen	X		n1							n1=0..7: I/O-Port <n1> einlesen (1=H-Pegel=5V, 0=L-Pegel=0V) n1=8: Alle 8 I/O-Ports I/O0..I/O7 als 8-Bit Binärwert einlesen	
Displaytyp abfragen	?									mit diesem Befehl wird der Displaytyp abgefragt. Zurückgesendet werden 3 Bytes: 128, 64, 'V' (128x64 Pixel Auflösung, vertikal Organisiert)	

### PARAMETER

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.

**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; gültig sind Werte von 0..127 für x- bzw. 0..63 für y-Koordinaten.  
Beispiel: x1= 10 (dez.) = \$0A

**n1,n2,nr,aw,ew,wert,mst,daten** ..... Nummernwerte werden mit 1 Byte übertragen.  
Beispiel: n1=15(dez.) = \$0F

### PROGRAMMIERBEISPIEL

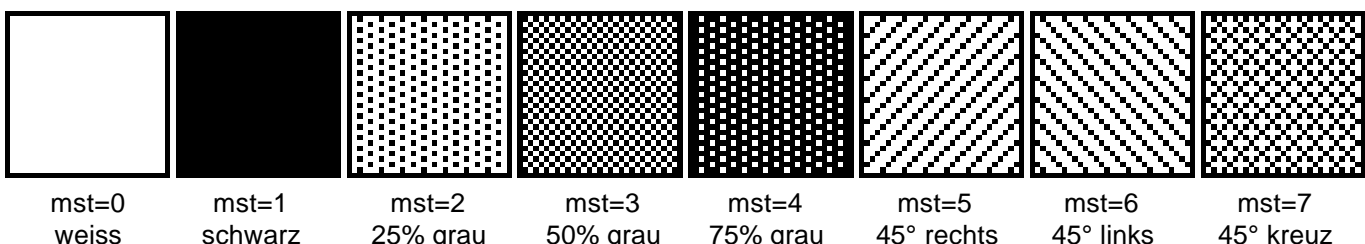
In der nachfolgenden Tabelle ist ein Beispiel zu sehen, welches die Zeichenkette "Test" an den Koordinaten 7,3 ausgibt.

Beispiel	Auszugebende Codes							
	Z	BEL	ETX	T	e	s	t	NUL
in ASCII	Z	BEL	ETX	T	e	s	t	NUL
in Hex	\$5A	\$07	\$03	\$54	\$65	\$73	\$74	\$00
in Dezimal	90	7	3	84	101	115	116	0
für Turbo-Pascal	write(aux, 'Z', chr(7), chr(3), 'Test', chr(0));							
für 'C'	fprintf(stdaux, "%c%c%c%s%c", 'Z', 7, 3, "Test", 0);							
für Q-Basic	OPEN "COM1:1200,N,8,2,BIN" FOR RANDOM AS #1 PRINT #1,"Z"+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:



### BESCHREIBUNG DER EINZELNEN GRAFIKFUNKTIONEN

Auf den nächsten Seiten befindet sich eine detaillierte alphabetisch sortierte Beschreibung zu jeder einzelnen Funktion. Als Beispiel wird jeweils ein Bildausschnitt von 50 x 32 Pixeln als Hardcopy gezeigt der den Displayinhalt nach Ausführung des Befehls darstellt. In den Beispielen sind die zu übertragenden Bytes als Hex-Werte abgebildet.

#### A x1 y1 n1

Ein Zeichen **n1** wird an die Koordinate **x1,y1** unter Beachtung des eingestellten Fonts 'F' und des Textmodus 'T' (setzen / löschen / invertieren / replace / invers replace / Füllmuster) ausgegeben. Der Ursprung (0,0) liegt im linken oberen Eck des Displays. Die Koordinatenangaben beziehen sich auf das linke obere Eck des Zeichens. Achtung: Font Nr.1 zeigt nur Großbuchstaben.

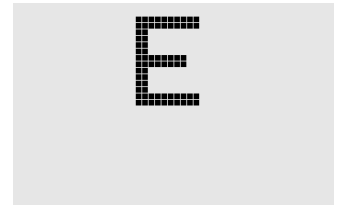
Beispiel: \$41 \$13 \$02 \$45

Zeichen 'E' wird an Koordinate 19,2 ausgegeben.

Eingestellter Font: 6x8 mit 2-facher Breite und 2-facher Höhe

Textmodus: Replace und Muster Schwarz

#### ASCII-Zeichen setzen



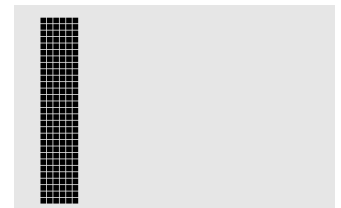
#### B L/R/O/U nr x1 y1 x2 y2 aw ew mst

Es können bis zu 8 Bargraphs (**nr=1..8**) definiert werden, welche nach **L=links**, **R=rechts**, **O=oben** oder **U=unten** ausschlagen können. Der Bargraph beansprucht bei Vollausschlag 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 dem Ausführen dieses Befehles ist der Bargraph nur definiert, am Display ist er aber noch nicht zu sehen).

Beispiel: \$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)

#### Bargraph definieren



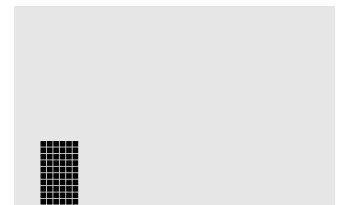
#### B nr wert

Der Bargraph mit der Nummer **n1** (1..8) wird auf den neuen Wert eingestellt (**aw <= wert <= ew**). Wenn **wert > ew** dann wird der Endwert **ew** dargestellt. Der Bargraph muss vorher definiert worden sein (siehe oben).

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

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

#### Bargraph zeichnen



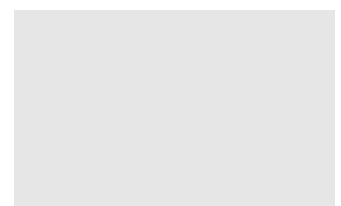
#### C n1

schaltet das Display Ein (**n1=1**) oder Aus (**n1=0**); alle Displaydaten bleiben erhalten und es können weiterhin Befehle ausgeführt werden.

Beispiel: \$43 \$00

Der Displayinhalt wird unsichtbar, der Inhalt bleibt jedoch erhalten.

#### Display Control



## ELECTRONIC ASSEMBLY

### D L/I/S Display Befehl

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

Beispiel: \$44 \$49

invertiert den gesamten Displayinhalt

### E n1 daten

Es ist möglich, bis zu 16 Zeichen selbst zu definieren (je nach Fontgröße). Diese Zeichen haben dann die ASCII Codes 1 bis max.16 und bleiben bis zum Abschalten der Versorgungsspannung in einem 64 Byte großen internen RAM-Bereich erhalten. Bei Font 1 können bis zu 16 Zeichen definiert werden, bei Font 2 noch 10 Zeichen und beim größten Font 3 immer noch 4 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 4 vom 4x6 Font (siehe Tabelle nebenan) !

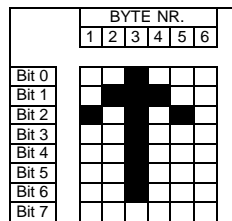
Beispiel 1:

Mit dem Befehl

\$45 \$03

\$04 \$02 \$7F \$02 \$04 \$00

wird für ASCII-Nr. 3, bei eingestelltem 6x8 Zeichensatz, ein Pfeil nach oben definiert.



Beispiel 2:

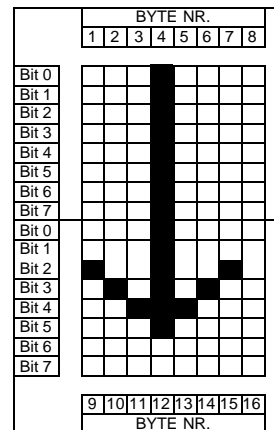
Mit dem Befehl

\$45 \$02

\$00 \$00 \$00 \$FF \$00 \$00 \$00 \$00

\$04 \$08 \$10 \$3F \$10 \$08 \$04 \$00

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



### Zeichen definieren

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

### F n1 n2 n3

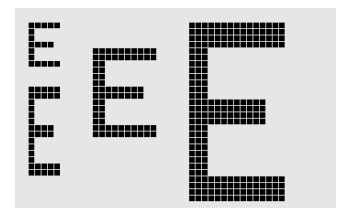
Es wird der Font mit der Nr. **n1** (1=4x6 nur Großbuchstaben; 2=6x8; 3=8x16) eingestellt. Ausserdem wird ein Vergrößerungsfaktor (1..4-fach) für die Breite **n2** und für die Höhe **n3** getrennt eingestellt.

Beispiel: \$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.

### Font einstellen



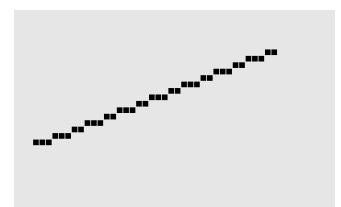
### G x1 y1 x2 y2

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

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

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

### Gerade zeichnen



### 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 die Höhe des Bildausschnittes und danach die Bilddaten. Zum Aufbau der Bilddaten siehe den Befehl Bild Upload 'U'.

Beispiel: \$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.

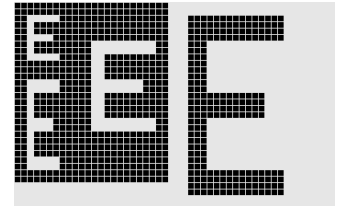
### I x1 y1 x2 y2

### Bereich invertieren

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: \$49 \$00 \$00 \$17 \$1B

invertiert bei vorhandenem Displayinhalt aus dem Beispiel "Font einstellen" den Bereich von 0,0 nach 23,27.



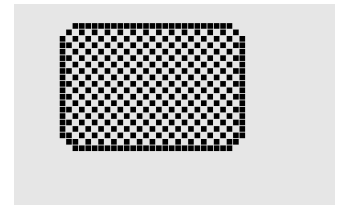
### J x1 y1 x2 y2 mst

### Rundbox zeichnen

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 'N' Runddeck zeichnen.

Beispiel: \$4A \$07 \$03 \$23 \$16 \$03

zeichnet eine Rundbox von 7,3 nach 35,22 mit dem Muster 3=50%Grau.



### K S/D n1 Grafikkontroller (de)selektieren

Der Grafikkontroller mit der Hardwareadresse **n1** (0..3) wird **S**=selektiert oder **D**=deselektiert; Die Adresse 255=\$FF ist eine Masteradresse mit der alle Grafikkontroller angesprochen werden. Die Adresseinstellung erfolgt per Hardware (Pins ADR0/1 siehe Seite 16).

Beispiel: \$4B \$44 \$00

alle Befehle werden für den Grafikkontroller mit der Adresse \$00 ab sofort ignoriert.

### L x1 y1 x2 y2

### Bereich löschen

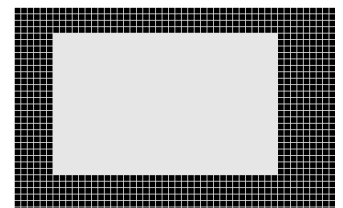
Der Bereich von der linken oberen Ecke **x1,y1** bis zur rechten unteren Ecke **x2,y2** wird gelöscht.

Beispiel:

\$44 \$53

\$4C \$06 \$04 \$28 \$19

Zuerst wird das Display mit 'D', 'S' gefüllt und dann der Bereich von 6,4 nach 40,25 gelöscht .



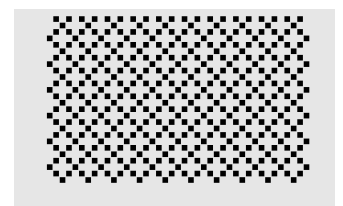
### M x1 y1 x2 y2 mst

### Bereich mit Füllmuster

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 'V' (setzen / löschen / invertieren / replace / invers replace) gezeichnet.

Beispiel: \$4D \$05 \$01 \$2D \$1A \$07

zeichnet das Muster 7=45°Kreuz von 5,1 nach 45,26.



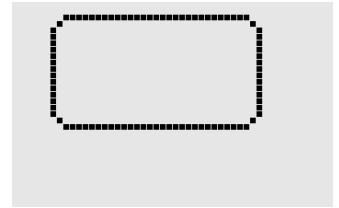
## ELECTRONIC ASSEMBLY

**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 Rundercks wird nicht verändert. Vergleiche 'J' Rundbox zeichnen.

Beispiel: \$4E \$06 \$02 \$26 \$13

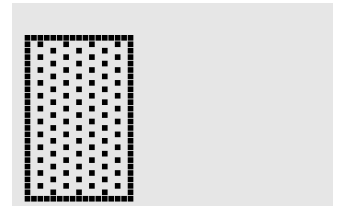
zeichnet ein Runderck von 6,2 nach 38,19.

**Runderck zeichnen****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 'R' Rechteck zeichnen.

Beispiel: \$4F \$02 \$05 \$12 \$1E \$02

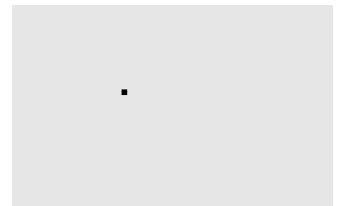
zeichnet eine Box von 2,5 nach 18,30 mit dem Muster 2=25%Grau.

**Box zeichnen****P x1 y1**

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

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

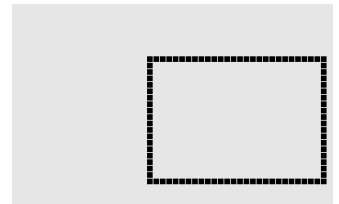
setzt den Pixel an der Koordinate 17,13.

**Punkt setzen****R x1 y1 x2 y2**

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 'O' Runderck zeichnen.

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

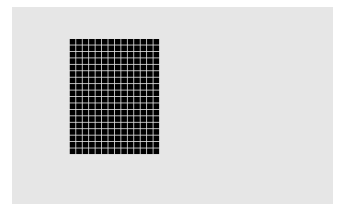
zeichnet ein Rechteck von 21,8 nach 48,37.

**Rechteck zeichnen****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: \$53 \$09 \$05 \$16 \$16

setzt den Bereich von 9,5 nach 22,22 auf schwarz.

**Bereich füllen**

## T L/R/O/U n1 mst

Der Verknüpfungsmodus **n1** und das Muster **mst** wird für Textfunktionen ASCII-Zeichen setzen 'A' und Zeichenkette ausgeben 'Z' eingestellt. Für den Befehl Zeichenkette ausgeben 'Z' wird außerdem die Schreibrichtung angegeben: **L**=links, **R**=rechts, **O**=oben und **U**=unten.

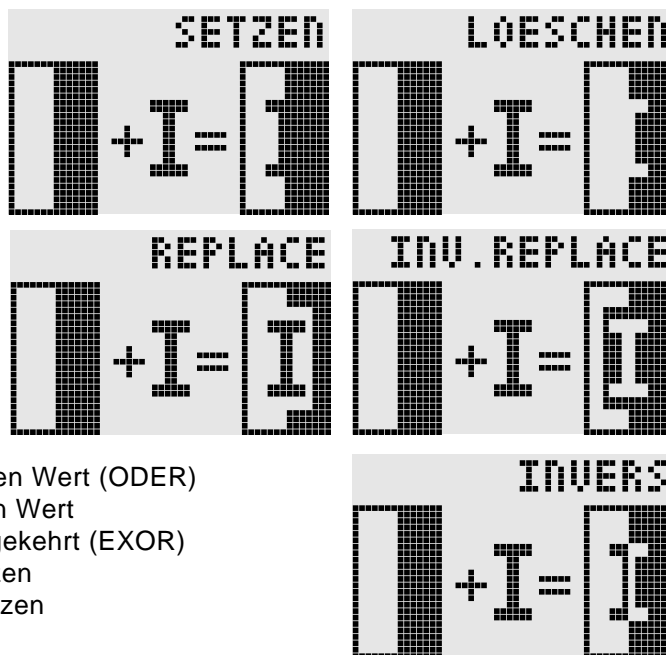
Beispiel: \$54 \$52 \$03 \$03

stellt den Verknüpfungsmodus für folgende Textfunktionen auf graue Zeichen (Muster 3 = 50%Gru) invertiert mit dem Hintergrund, Zeichenketten werden nach rechts geschrieben.

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

## Text-Modus einstellen



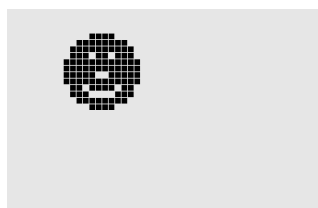
## U x1 y1 daten

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

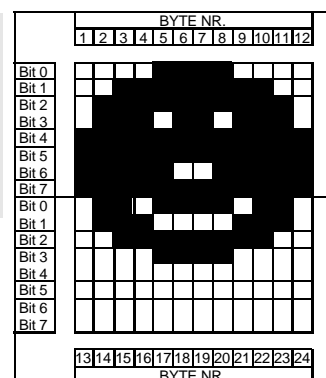
**daten:** - 1 Byte für die Bildbreite in Pixeln  
 - 1 Byte für die Bildhöhe in Pixeln  
 - Bilddaten: Anzahl = ((Höhe+7) / 8) \* Breite Bytes.  
 1 Byte steht für 8 senkrechte Pixel am Bildschirm;  
 0=weiß, 1=schwarz; LSB: oben, MSB: unten;  
 Das Bild ist von links nach rechts abgelegt.  
 Das Programm BMP2BLV.EXE erzeugt aus monochromen Windows-Bitmap-Grafiken die Bilddaten inkl. der Angabe von Breite und Höhe.

Beispiel: \$55 \$09 \$04 \$00 \$00  
 \$F0 \$FC \$FE \$FE \$F7 \$BF \$BF \$F7 \$FE \$FE \$FC \$F0  
 \$00 \$03 \$07 \$06 \$0D \$0D \$0D \$0D \$06 \$07 \$03 \$00

lädt das nebenstehende Bild an die Koordinate 9,4.



## Bild Upload



## V n1

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

Beispiel: \$56 \$03

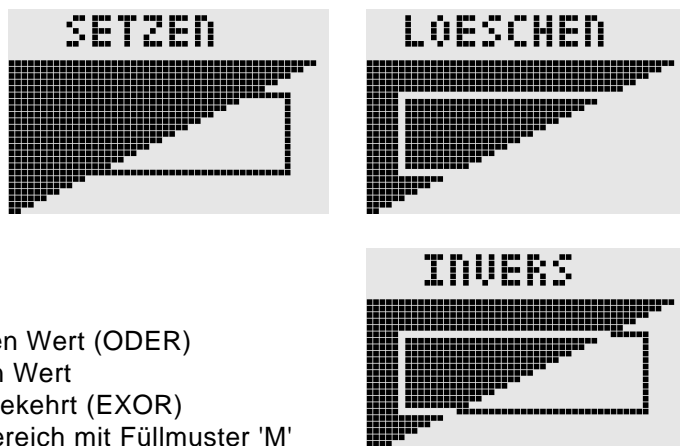
stellt den Verknüpfungsmodus auf invers.

Als Beispiel wird nebenan ein Rechteck mit den Verknüpfungsmodi setzen, löschen und invers auf einen vorhandenem 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 'M'
- 5=invers replace: Hintergrund füllen, Pixel löschen; nur Bereich mit Füllmuster 'M'

## Grafik-Modus einstellen

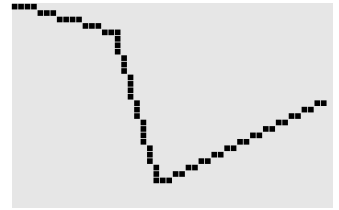


**W x1 y1****Gerade weiterzeichnen**

Zieht eine Gerade vom zuletzt gezeichneten Geradenende bzw. Punkt (siehe Seite 3 Last xy) bis nach **x1,y1** unter Beachtung des eingestellten Grafik-Modus 'V' (setzen / löschen / invers).

Beispiel:

```
$47 $00 $00 $10 $04
$57 $16 $1B
$57 $30 $0F
```



Zuerst wird eine Gerade von 0,0 nach 16,4 gezeichnet. Dann weiter nach 22,27 und nach 48,15.

**X n1****I/O Port lesen**

Liest einen Port (**n1**: 0..7 = I/O: 0..7) ein.

Wenn **n1** = 8, werden alle I/O 0..7 als Binärwert eingelesen; I/O 0: LSB, I/O 7: MSB

Siehe Applikation auf Seite 4.

Beispiel: \$58 \$02

liest den Pegel an I/O 2 ein und sendet bei L-Pegel ein \$00 und bei H-Pegel ein \$01 über RS-232

**Y n1 n2****I/O Port einstellen**

Ändert den Port (**n1**: 0..7 = I/O: 0..7) auf den Wert **n2** (0=L-Pegel; 1=H-Pegel; 2=Port invertieren).

Wenn **n1** = 8, werden alle I/O 0..7 als Binärwert **n2** ausgegeben; I/O 0: LSB, I/O 7: MSB

Siehe Applikation auf Seite 4.

Beispiel: \$59 \$02 \$01

schaltet den Port I/O 2 auf H-Pegel

**Z x1 y1 ASCII... NUL****Zeichenkette schreiben**

Schreibt an die Koordinate **x1,y1** die Zeichenkette **ASCII...** unter Beachtung des eingestellten Textmodus 'T' (setzen / löschen / invertieren / replace / invers replace / Füllmuster/ Richtung). Die Zeichenkette muß mit **NUL** (\$00) abgeschlossen werden. Der Ursprung (0,0) liegt im linken oberen Eck des Displays. Die Koordinatenangaben beziehen sich auf das linke obere Eck des Zeichens. Achtung: Font Nr.1 zeigt nur Großbuchstaben.

Beispiel: \$5A \$06 \$0B \$54 \$65 \$73 \$74 \$00

schreibt an die Koordinate 6,11 die Zeichenkette "Test"

Eingestellter Font: 8x16 mit normaler Breite und Höhe

Textmodus: Schreibrichtung nach Rechts, Verknüpfung Replace mit Muster Schwarz

**?****Displaytyp abfragen**

Die Auflösung des Displays und die Art des Bildaufbaus wird abgefragt. Bei dem High-Level-Grafikkontroller IC202-PGH ist die Auflösung immer 128 x 64 Pixel und der Bildaufbau vertikal organisiert. Dieser Befehl ist für externe Programme die auf den High-Level-Grafikkontroller zugreifen gedacht. Das IC6963-PGH ist z.B. für Displays mit Toshibacontroller bei dem die Auflösung variabel bis 240x128 Pixel und die Organisation horizontal ist.

Beispiel: \$3F

Nach diesem Befehl wird zuerst die X- (128) und Y-Auflösung (64) und dann die Art des Bildaufbaus ('V') für die vertikale Organisation über die RS-232 Schnittstelle gesendet.

# EA GE128-6N3V24

## BAUDRATEN

Die Baudrate läßt sich über die linken 3 Lötbrücken einstellen. Im Auslieferungszustand sind 9.600 Baud eingestellt. Bitte beachten Sie, daß der interne Datenpuffer lediglich 20 Byte umfaßt. Beim Senden größerer Datenmengen 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.

Baudraten			
Lötbrücken			Datenformat
1	2	3	8,N,1
zu	zu	zu	1200
offen	zu	zu	2400
zu	offen	zu	4800
offen	offen	zu	9600
zu	zu	offen	19200
offen	zu	offen	38400
zu	offen	offen	57600
offen	offen	offen	115200

## ADRESSIERUNG

Bis zu 4 Displays können an einer seriellen Schnittstelle adressiert betrieben werden. Die jeweilige Adresse wird über die Lötbrücken 4 und 5 eingestellt.

Adressen		
Lötbrücke		Adresse
4	5	
zu	zu	0
zu	offen	1
offen	zu	2
offen	offen	3

**Achtung!** Da beim einfachen Parallelschalten der Handshakeleitungen RTS bzw. der Sendeleitungen TxD zwei Ausgänge gegeneinander arbeiten würden, muß durch eine zusätzliche Hardware sichergestellt werden, daß es zu keinem Datencrash kommen kann. Sinnvoll ist z.B. eine Verknüpfung über ODER-Logik bei RTS bzw. über UND-Logik bei TxD.

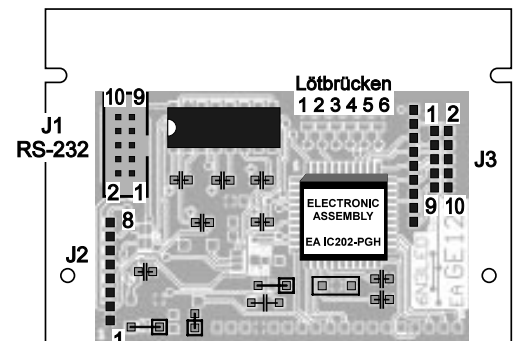
## PINBELEGUNG

J1 liefert "echte" RS-232 Pegel ( $\pm 10V$ ). J2 ist für den direkten 5V-Anschluß an einen  $\mu C$  konzipiert. Bei Verwendung von J2 müssen die Lötbrücken "R" und "C" geöffnet oder der Baustein 232 entfernt werden! Wird die Kontrastspannung V0 an J2 eingespeist, so muß die Lötbrücke von "232" auf "Ext" umgestellt werden.

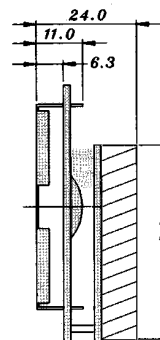
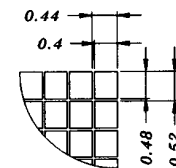
RS-232 Anschluß J1			
Pin	Symbo	In/Out	Funktion
1	VDD	-	+ 5V Versorgung
2	DCD	-	Brücke nach DTR
3	DSR	-	Brücke nach DTR
4	TxD	Out	Transmit Data
5	CTS	In	Clear To Send
6	RxD	In	Receive Data
7	RTS	Out	Request To Send
8	DTR	-	siehe Pin 2, Pin 3
9	V0	In	ca. -9V Displayspg.
10	GND	-	0V Masse

Anschluß J2			
Pin	Symbo	In/Out	Funktion
1	GND	-	0V Masse
2	VDD	-	+ 5V Versorgung
3	V0	In	ca. -9V Displayspg.
4	TxD5	Out	Transmit Data CMOS
5	RxD5	In	Receive Data CMOS
6	RTS5	Out	Request To Send CMOS
7	CTS5	In	Clear To Send CMOS
8	RESET	In	Reset Controller

Anschluß J3			
Pin	Symbol	In/Out	Funktion
1	VDD	-	+ 5V Versorgung
2..9	I/O0..7	In/Out	Ein-/Ausgang
10	GND	-	0V Masse

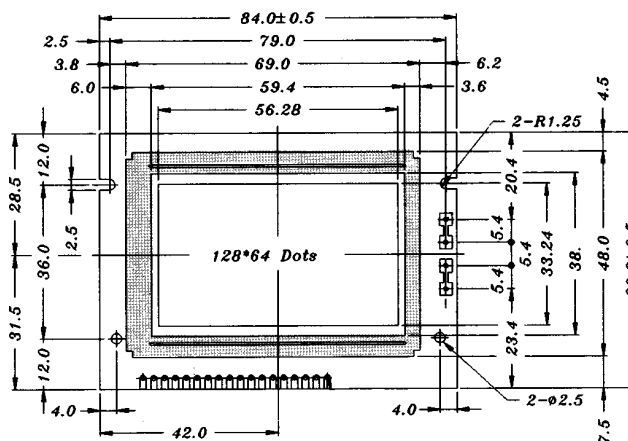


Ansicht von hinten



## ABMESSUNGEN

alle Maße in mm



**Achtung!**  
 Handhabungsvorschriften beachten  
 Elektrostatisch gefährdete Bauelemente