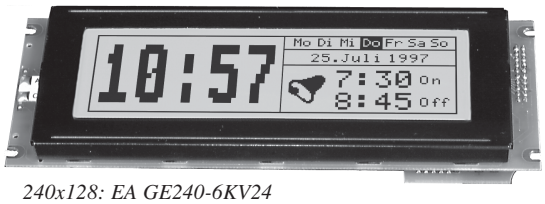
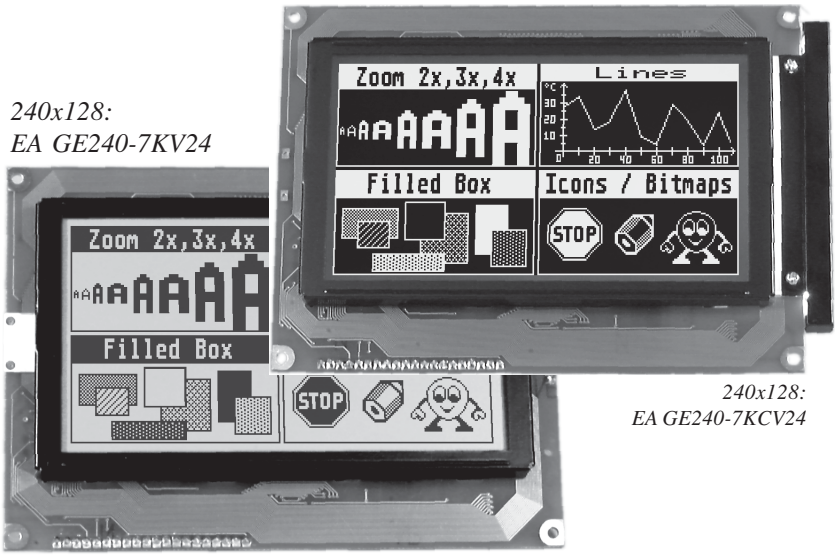
GRAPHIC UNIT

COMPLETE UNIT WITH 3 FONTS AND INTELLIGENT CONTROLLER



240x128: EA GE240-6KV24

240x128:
EA GE240-7KV24



240x128:
EA GE240-7KCV24

FEATURES

* LCD GRAPHICS DISPLAY SWITH EA IC6963 HIGH-LEVEL GRAPHICS CONTROLLER
* 240x128 PIXELS WITH CFL BEL. BLUE NEGATIVE, DIMENSIONS: 151 x 104 x 25 mm
* 240x128 PIXELS WITH LED ILLUMINATION GN/YL, DIMENSIONS: 144 x 104 x 25 mm
* 240x64 PIXELS WITH LED ILLUMINATION GN/YL, DIMENSIONS: 180 x 65 x 25 mm
* 3 FONTS (ZOOM) FROM about 2mm VIA about 5mm UP TO about 50mm
* SUPPLY VOLTAGE: +5V /500..1000mA
* RS-232 BAUD RATES 1200..115200 BD
* POSITIONING ACCURATE TO THE PIXEL WITH ALL FUNCTIONS
* PROGRAMMING BY MEANS OF COMMANDS SIMILAR TO HIGH-LEVEL LANGUAGE:
* STRAIGHT LINE, POINT, AREA, AND/OR/EXOR, BAR GRAPH...
* UP TO 21 FREELY DEFINABLE CHARACTERS
* COMBINATIONS OF TEXT AND GRAPHICS
* 6 CLIPBOARD FUNCTIONS, CURSOR FUNCTIONS

ACCESSORIES

* CABLE FOR CONNECTING TO 9-PIN SUB-D (FEMALE): **EA KV24-9B**
* DIP SWITCH, E.G. FOR SETTING THE BAUD RATE: **EA OPT-DIP6**

ORDER DESIGNATION

GRAPHICS UNIT 240x128 WITH CFL BEL., BLUE NEGATIVE	**EA GE240-7KCV24**
GRAPHICS UNIT 240x128 WITH LED ILLUMINATION GN/YL	**EA GE240-7KV24**
GRAPHICS UNIT 240x64 WITH LED ILLUMINATION GN/YL	**EA GE240-6KV24**

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

GRAPHIC UNIT

GENERAL

The graphics LCDs with EA IC6963 are completely assembled graphics units with a variety of integrated functions. The fact that they are small, have excellent supertwist contrast, and are simple to program means that it is possible to connect an informative and visually attractive display screen to almost any processor system in a matter of a few hours. They are accessed via a standard RS-232 interface. The display contains complete graphics routines for the display output, together with a wide range of font sizes.
Graphics commands similar to high-level language are used for programming. There is no longer any need for the time-consuming programming of character sets and graphics routines. But it is not just during development that time and effort are dramatically reduced; the following benefits are evident in series production as well:

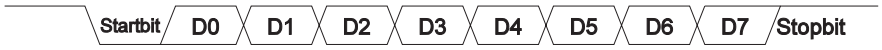
- There are no timing problems with the fast processor bus.
- There are no memory problems (RAM and memory for the character set, above all in the case of µC).
- There are no time-consuming graphics calculations that slow down the processor.

No drivers, decoders, or port components are needed. In the simplest case, the display is controlled by means of a single RxD line.

HARDWARE

The displays are designed to work with an operating voltage of +5V. Serial asynchronous data transfer is carried out in RS-232 format with real V.24 levels (±10V), or via 5V CMOS levels. The transmission format is set permanently to 8 data bits, 1 stop bit, and no parity. Rates between 1200 baud and 115,200 baud can be selected by means of three solder straps. RTS and CTS handshake lines are available. No analysis is needed in the case of small quantities of data.

Data format:



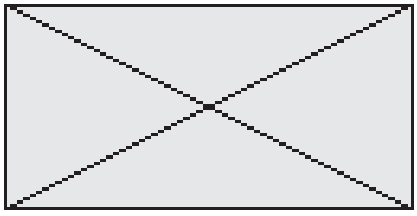
There are also eight I/O ports on the J3 eyelet strip that can be used freely and individually, either as inputs or outputs. They can, for instance, be employed to connect a transistor / relay (IL_{max}=10mA), or to read in keys / switches.

SOFTWARE

The graphics unit is programmed by means of commands, such as 'Draw a rectangle from (0,0) to (64,15)'. The origin is located at the top left-hand corner of the display. The following bytes have to be sent over the serial interface for this purpose: $52 $00 $00 $40 $0F. Character strings can be placed precisely to the pixel. Text and graphics can be combined at any time. Three different character sets can be used. Each one can be zoomed from 2 to 8 times. When the 8-times zoom is used with the largest character set (16x8), the words and numbers displayed will fill the screen (= 128x64).

TEST MODE

If solder strap 6 (pin RTS5) is closed (connected to GND) while power-on or reset, the display will be in test mode, and a rectangle with two diagonal lines will flash. If the solder strap is opened, then the display will return to normal mode. However, you will still be able to see the test picture.



INTEGRATED FONTS

Each graphics unit has three integrated character sets. The characters in these sets can be used at their normal height or can increased up to eight times, while their width can be increased from two to eight times, irrespective of the height.

Upper \ + Lower	$0 (0)	$1 (1)	$2 (2)	$3 (3)	$4 (4)	$5 (5)	$6 (6)	$7 (7)	$8 (8)	$9 (9)	$A (10)	$B (11)	$C (12)	$D (13)	$E (14)	$F (15)
$20 (dez: 32)		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
$40 (dez: 64)	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
$50 (dez: 80)	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_

Font 1: 4x6

Upper \ + Lower	$0 (0)	$1 (1)	$2 (2)	$3 (3)	$4 (4)	$5 (5)	$6 (6)	$7 (7)	$8 (8)	$9 (9)	$A (10)	$B (11)	$C (12)	$D (13)	$E (14)	$F (15)
$20 (dez: 32)		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
$40 (dez: 64)	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
$50 (dez: 80)	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
$60 (dez: 96)	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
$70 (dez: 112)	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	⌂
$80 (dez: 128)	Ç	ü	é	â	ä	à	å	ç	ê	ë	è	ï	î	ì	Ä	Å
$90 (dez: 144)	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	¢	£	¥	ß	

In addition, you can define up to twenty-one characters of your own, depending on the font being used. These characters will remain until the supply voltage is switched off. (See command 'E'.)

Font 2: 6x8

Each character can be placed precisely to the pixel. A combination of text and graphics can be displayed as required. Several different font sizes can also be displayed together.

Upper \ + Lower	$0 (0)	$1 (1)	$2 (2)	$3 (3)	$4 (4)	$5 (5)	$6 (6)	$7 (7)	$8 (8)	$9 (9)	$A (10)	$B (11)	$C (12)	$D (13)	$E (14)	$F (15)
$20 (dez: 32)		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
$40 (dez: 64)	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
$50 (dez: 80)	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
$60 (dez: 96)	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
$70 (dez: 112)	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	⌂
$80 (dez: 128)	Ç	ü	é	â	ä	à	å	ç	ê	ë	è	ï	î	ì	Ä	Å
$90 (dez: 144)	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	¢	£	¥	ß	

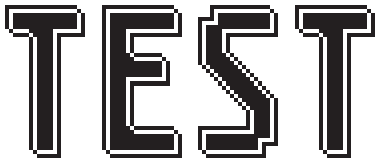
Font 3: 8x16

TIP: FONT EFFECTS

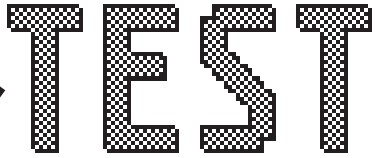
With large fonts, the command 'T', TEXT mode (link, pattern), can be used to produce interesting effects through overlaying (writing and offsetting a word several times).

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

"Outline font" resulting from overlaying **(EXOR)** at **pos. 1,1**.



When the "outline font" is overlaid again **(EXOR)** at **pos. 2,2**, this results in an "outline font with filling".



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

Command table EA IC6963

Command											Remarks
Functions for outputting text											
Text mode	T	R L O U	n1	ptn							R/L/O/U: Write character string (R)ight, (L)eft, (O)ben (up), (U)nten (down); n1: overlay combination mode for text output 1=set; 2=delete; 3=inverse; 4=replace; 5=inverse replace; ptn: use pattern no. 0..7;
Set font	F	n1	n2	n3							Set font no. n1; n1=1:4x6 font; n1=2:6x8 font; n2=3:8x16 font n2+n3=zoom factor (1..8); n2=X factor; n3=Y factor;
Set ASCII characters	A	x1	y1	n1							The character n1 will be set at coordinate x1,y1. (Reference top left)
Set character string	Z	x1	y1	...	NUL						Output character string (...) to x1,y1; character ´NUL´ ($00)=end
Define character	E	n1	data ...								n1=character no.; data =number of bytes dep. on current font
Graphics commands with overlay mode											
Graphics mode	V	n1									n1: 1=set; 2=delete; 3=inverse; 4=replace; 5=inverse replace;
Set point	P	x1	y1								Set one pixel at coordinates x1, y1
Draw straight line	G	x1	y1	x2	y2						Draw straight line from x1,y1 to x2,y2
Continue straight line	W	x1	y1								Draw a straight line from last end point to x1, y1
Draw rectangle	R	x1	y1	x2	y2						Draw a rectangle; x1,y1,x2,y2 = opposite corner points
Draw round corner	N	x1	y1	x2	y2						Draw a rectangle with round corners; x1,y1,x2,y2 = corner points
Area with fill pattern	M	x1	y1	x2	y2	ptn					Draw area with pattern ptn (0..7); x1,y1,x2,y2 = corner points
Other graphics commands											
Delete display	D	L									Delete entire contents of display (set to white);
Invert display	D	I									Invert entire contents of display;
Fill display	D	S									Fill entire contents of display; (set to black);
Delete area	L	x1	y1	x2	y2						Delete an area; x1,y1,x2,y2 = opposite corner points
Invert area	I	x1	y1	x2	y2						Invert an area; x1,y1,x2,y2 = opposite corner points
Fill area	S	x1	y1	x2	y2						Fill an area; x1,y1,x2,y2 = opposite corner points
Draw box	O	x1	y1	x2	y2	ptn					Draw a rectangle with fill pattern ptn (0..7); (always replace)
Draw round box	J	x1	y1	x2	y2	ptn					Draw a round corner with fill pattern ptn (0..7); (always replace)
Draw bar graph	B	nr	valu								Set the bar graph with the ´nr´ (1..8) to the new user ´value'
Upload picture area	U	x1	y1	data ...							Load a picture area to x1,y1; see picture structure for picture data
Control / definition commands											
Define bar graph	B	R L O U	nr	x1	y1	x2	y2	aw	ew	ptn	Define bar graph to L(eft), R(ight), O(up), U(down) with the ´nr´ (1..8). x1,y1,x2,y2 form the rectangle enclosing the bar graph. aw, ew are the values for 0% and 100%. ptn=pattern (0..7).
Clipboard commands *) (buffer for picture areas)	C	B									The entire contents of the display will be copied to the clipboard
		S	x1	y1	x2	y2					Picture area extending from x1, y1 to x2, y2 will be copied to the clipboard
		R									The picture area on the clipboard will be copied back to the display
		K	x1	y1							The picture area on the clipboard will be copied to x1, y1 in the display
		H									The picture area on the clipboard will be sent as hard copy via RS232
		L	data...								Load a picture area to the clipboard; see picture structure for picture data
Automatic flashing area (cursor function)	Q	D	x1	y1	x2	y2					Defines a flashing area x1,y1 to x2,y2; activate flashing function
		Z	n1								Set the flashing time n1= 1..15 in 1/10s; 0=deactivate flashing function
		M	I								Inverse mode (flashing area will be inverted); activate flashing function
			ptn								Clipboard mode*) ptn=pattern(0..7) of the block cursor; activate flashing
Select / deselect graphics lcd	K	S	n1								Activate display with address n1 (n1=0..3; n1=255: all)
		D									Deactivate display with address n1 (n1=0..3; n1=255: all)
Write I/O port	Y	n1	n2								n1=0..7: reset I/O port n1 (n2=0); set (n2=1); invert (n2=2) n1=8: Set all 8 I/O ports in accordance with n2 (=8 bit binary value)
Set display type	!	n1	n2	LO	HI						Another display can be set. n1=X resolution (64..240); n2=Y resolution (16..128); LO, HI 16-bit picture start address (normally $0000)
Send commands											
Hard copy	H	x1	y1	x2	y2						An area is requested as a picture. The width and height are sent in pixels first of all, followed by the actual picture data, via RS232.
Read I/O port	X	n1									n1=0..7: load I/O port <n1> (1=H level=5V, 0=L level=0V) n1=8: load all 8 I/O ports I/O0..I/O7 as 8-bit binary value
Query display type	?										This command is used to query the display type. 3 bytes are sent back: X resolution, Y resol., 'H' (e.g. 240, 64 (pixels), horizontal picture)

*) *All clipboard commands require a display RAM of at least 8 KB. The clipboard commands cannot be used with displays having a smaller RAM (e.g. 2 KB).*

PARAMETERS

Various in-built commands can be used to program the high-level graphics controller. Each command starts with a command letter, which is followed by a number of parameters. All the commands and their parameters - such as coordinates and other transfer values - are always expected as bytes. No separating characters, such as spaces or commas, must be used between them. The commands require **no final byte**, such as a carriage return (apart from the character string: $00).

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

x1, x2, y1, y2 ..... Coordinate details will be transmitted with 1 byte.
Example: x1= 10 (dec.) = $0A

n1,n2,nr,aw,ew,value,ptn,data ..... Number values will be transmitted with 1 byte.
Example: n1=15(dec.) = $0F

EXAMPLE OF PROGRAMMING

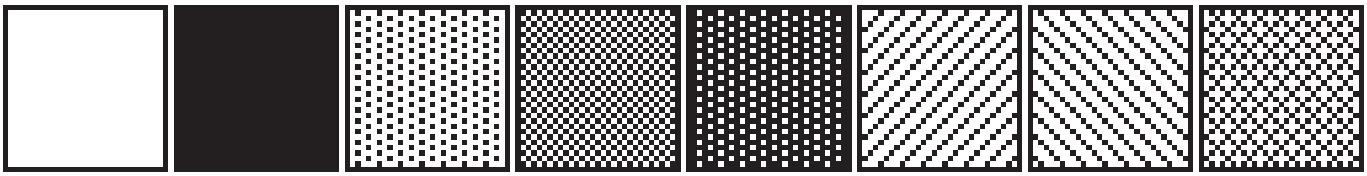
The following table shows an example in which the character string “Test” is output at coordinate 7,3.

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

PATTERN

With different commands, a pattern type (ptn = 0..7) can be set as a parameter. Thus, rectangles, areas, bar graphs, and even text can be linked to different patterns, and then displayed.

The following fill patterns are available for this purpose:



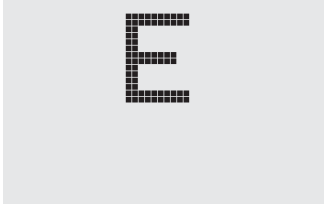
ptn=0	**ptn=1**	**ptn=2**	**ptn=3**	**ptn=4**	**ptn=5**	**ptn=6**	**ptn=7**
white	black	25% gray	50% gray	75% gray	45° right	45° left	45° cross

DESCRIPTION OF THE VARIOUS GRAPHICS FUNCTIONS

On the following pages, you will find detailed descriptions of all the functions, set out in alphabetical order. An enlarged extract from the picture in 50x32 pixels is shown as a hard copy example of what the display contains once the command has been executed. In the examples, the bytes to be transmitted are shown as hex values.

A x1 y1 n1 — Set ASCII character

A character **n1** will be output to coordinate **x1**,**y1**, taking into account the fonts that have been set, 'F', and the text mode, 'T' (set / delete / invert / replace / inverse replace / fill pattern). The origin (0,0) is at the top left-hand corner of the display. The coordinate details refer to the top left-hand corner of the character. Note that font no. 1 displays upper case letters only.



Example: $41 $13 $02 $45

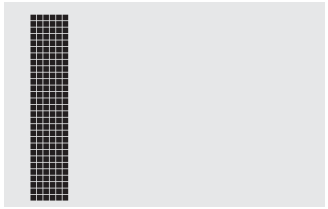
Character 'E' will be output to coordinate 19,2.

Font set: 6x8 with double width and double height.

Text mode: Replace and black pattern.

B L/R/O/U nr x1 y1 x2 y2 aw ew ptn — Define bar graph

Up to eight bar graphs (**nr**=1..8) can be defined. These can extend **L**=left, **R**=right, **O**=up, or **U**=down. At its full extent, the bar graph will occupy an area from coordinate **x1**,**y1** to coordinate **x2**,**y2**. It will be scaled with the start value (no extension), **aw**, (=0..254) and the end value (full extension), **ew**, (=0..254). It will always be drawn in inverse mode, with the pattern, **ptn**. The background will therefore always be retained. (Note: After this command has been executed, the bar graph will only be defined, and will not yet be visible in the display.)

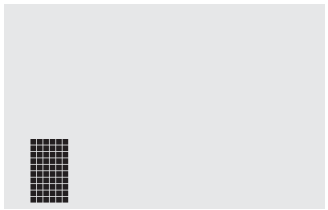


Example: $42 $4F $01 $04 $02 $09 $1E $04 $14 $01

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

B nr wert — Draw bar graph

The bar graph with the number **n1** (1..8) will be set to the new value (wert) (aw <= **value** <= ew). If **value** > ew, then the end value, ew, will be displayed. The bar graph must have been defined beforehand (see above).



Example: $42 $01 $0A

Bar graph no. 1 defined in the above example will be set to value 10.

C B*) — Save contents of display to clipboard

The entire contents of the display will be copied to the clipboard (buffer).

Example: $43 $42

This will save the entire contents of the display to the clipboard so that the screen can be restored later. The contents of the display will not be altered in the process.

**) All clipboard commands require a display RAM of at least 8 KB. The clipboard commands cannot be used with displays having a smaller RAM (e.g. 2 KB).*

C S x1 y1 x2 y2*) — Save area to clipboard

An area extending from the top left-hand corner **x1,y1** to the bottom right-hand corner **x2,y2** will be copied to the clipboard (buffer).

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

This will save the area extending from 0,0 to 23,27, so that the screen can be restored later. The contents of the display will not be altered.

C R*) — Restore area

The area that was saved last will be copied from the clipboard (buffer) back to the display. Target: original coordinates.

Example: $43 $52

This will restore the area last saved.

C K x1 y1*) — Copy area from clipboard

The area last saved to the clipboard (buffer) will be copied to a new position at **x1,y1** in the display.

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

This will take the area that was last saved and copy it to coordinate 10,32.

C L data*) — Load picture onto the clipboard

This will take the data that now follows, and will load it onto the clipboard (buffer).

Example: $43 $4C as with the upload command ´U´.

This means that even with a low baud rate (slow), a picture can be loaded into an invisible area, and can then be displayed “suddenly” at one or more places by means of the command ´C´, ´K´.

C H*) — Send picture from the clipboard as hard copy

This requests the data from the clipboard (buffer). The function is similar to the ´H´, hard copy, command.

Example: $43 $48

And the picture on the clipboard will be sent immediately via RS-232.

D L/I/S — Display command

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

Example: $44 $49

This will invert the entire contents of the display.

**) All clipboard commands require a display RAM of at least 8 KB. The clipboard commands cannot be used with displays having a smaller RAM (e.g. 2 KB).*

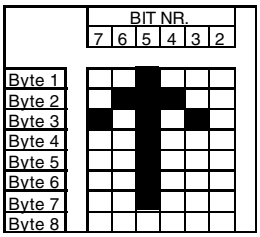
E n1 data — Define character

You can define up to twenty-one characters yourself (depending on the size of the font). These characters will then have the ASCII codes 1 to 21, and will remain in an invisible screen RAM of 128 bytes until the supply voltage is switched off. In the case of font 1, up to twenty-one characters can be defined; with font 2, the figure is sixteen; and with font 3, the largest, it is eight characters. Please note that if you specify several characters from different fonts, then you must bear in mind that a character with code 1 of the 8x16 font, for example, will need the same amount of RAM as characters with the codes 1 to 3 of the 4x6 font (see the table alongside).

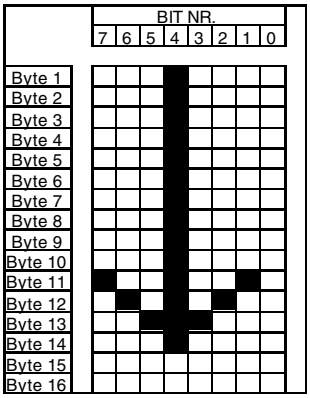
Define characters (ASCII)

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

Example 1:
$45 $01
$20 $70 $A8 $20 $20 $20 $20 $00
This defines an up arrow for ASCII no. 1, using the character set 6x8.

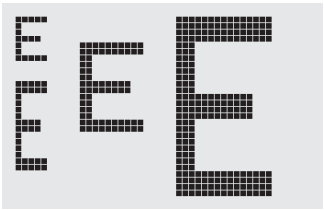


Example 2:
$45 $02
$10 $10 $10 $10 $10 $10 $10 $10 $10 $10 $92 $54 $38 $10 $00 $00
This defines a down arrow for ASCII no. 2, using the character set 8x16.



F n1 n2 n3 — Set font

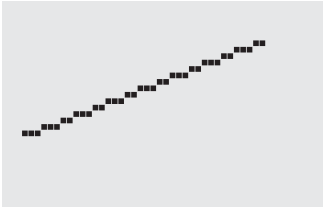
The font with the no. **n1** (1=4x6 upper-case letters only; 2=6x8; 3=8x16) will be set. In addition, an enlargement factor (1..8 times) for the width **n2** and the height **n3** will be set separately.



Example: $46 $02 $03 $04
The 6x8 font with the width enlarged three time and the height enlarged four times will be set with immediate effect.
In the diagram alongside, the character 'E' from the 6x8 font is shown with different enlargements.

G x1 y1 x2 y2 — Draw straight line

A straight line will be drawn from coordinate **x1,y1** to coordinate **x2,y2**, taking into account the graphics mode 'V' that has been set (set / delete / inverse).



Example: $47 $03 $14 $28 $06
A straight line will be drawn from 3,20 to 50,6.

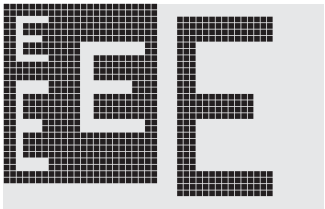
H x1 y1 x2 y2 — Produce a hard copy of the display contents

This requests the area extending from the top left-hand corner **x1,y1** to the bottom right-hand corner **x2,y2**. The graphics chip will immediately send the width and height of the section of the picture, followed by the picture data. See the upload picture command, 'U', for the structure of the picture data.

Example: $48 $00 $00 $1F $0F
The top left-hand part of the screen, measuring 32 x 16 pixels, will be sent immediately via RS-232.

I x1 y1 x2 y2 — Invert area

The area extending from the top left-hand corner **x1,y1** to the bottom right-hand corner **x2,y2** will be inverted (black pixels will become white, and vice versa).



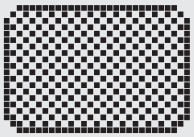
Example: $49 $00 $00 $17 $1B
This will invert the area extending from 0,0 to 23,27 when the contents of the display are as shown in the example under "Set font".

J x1 y1 x2 y2 ptn — Draw round box

A rectangle with rounded corners will be drawn from the top left-hand corner **x1,y1** to the bottom right-hand corner **x2,y2**, with the pattern **ptn**. The background will be deleted. Compare 'N, Draw round corner'.

Example: $4A $07 $03 $23 $16 $03

This will draw a round box extending from 7,3 to 35,22, with the pattern 3=50% gray.



K S/D n1 — (De)select graphics controller

The graphics controller with the hardware address **n1** (0..3) will be **S**=selected or **D**=deselected; The address 255=$FF is a master address that is used to access all graphics controllers. The address is set by hardware (pins ADR0/1, see page 16).

Example: $4B $44 $00

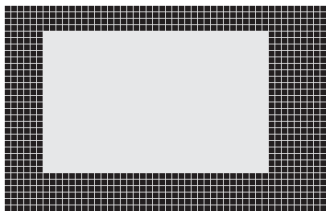
All commands for the graphics controller with the address $00 will be ignored with immediate effect.

L x1 y1 x2 y2 — Delete area

The area extending from the top left-hand corner **x1,y1** to the bottom right-hand corner **x2,y2** will be deleted.

Example: $44 $53 $4C $06 $04 $28 $19

To begin with, the display will be filled with 'D', 'S', and then the area extending from 6,4 to 40,25 will be deleted.

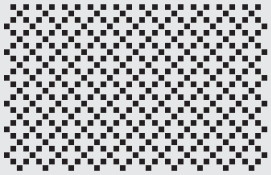


M x1 y1 x2 y2 ptn — Area with fill pattern

A rectangular area will be drawn from the top left-hand corner **x1,y1** to the bottom right-hand corner **x2,y2**, with the pattern **ptn**, and taking into account the graphics mode 'V' that has been set (set / delete / invert / replace / inverse replace).

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

This will draw the pattern 7=45°cross from 5,1 to 45,26.



N x1 y1 x2 y2 — Draw round corner

A rectangle with rounded corners will be drawn from the top left-hand corner **x1,y1** to the bottom right-hand corner **x2,y2**, taking into account the graphics mode 'V' that has been set (set / delete / inverse). The contents of the round corner will not be altered. Compare 'J, Draw round box'.

Example: $4E $06 $02 $26 $13

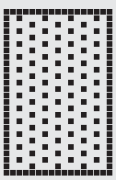
This will draw a round corner from 6,2 to 38,19.

O x1 y1 x2 y2 ptn — Draw box

A rectangle will be drawn from the top left-hand corner **x1,y1** to the bottom right-hand corner **x2,y2**, with the pattern **ptn**. The background of the box will be deleted. Compare 'R, Draw rectangle'.

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

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



P x1 y1 — Set point

A pixel will be set at coordinate **x1,y1**, taking into account the graphics mode 'V' that has been set (set / delete / invert).

Example: $50 $11 $0D

This will set the pixel at coordinate 17,13.

Q D x1 y1 x2 y2 — Define flashing area

This specifies the area extending from the top left-hand corner **x1,y1** to the bottom right-hand corner **x2,y2** as an automatic flashing area. At the same time, it starts the flashing function. This makes it possible to represent a "cursor" when entries are being made.

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

This defines the flashing area from 0,15 to 7,16. (Simulation of an underscore cursor for the 8x16 font, with a character at position 0,0.)

Q Z n1 — Set flashing time

This sets the flashing time to **n1** (=1..15) tenth seconds. At **n1**= 0, the flashing function will be deactivated, and the original screen will be restored.

Example: $51 $5A $05
This will set the flashing time to ½ second.

Q M I — Inverse flashing mode

This automatically inverts the specified flashing area, using the flashing time that has been set. At the same time, it starts the flashing function.

Example: $51 $49
This will set the inverse flashing mode.

Q M ptn*) — Block cursor flashing mode

This saves the defined flashing area to the clipboard. A cyclical changeover will be carried out between the original area and the pattern ptn (=0..7), using the flashing time that has been set. This means, for example, that a flashing cursor can be simulated (ptn=1 black), or a flashing word can be displayed (ptn=0 white). At the same time, the flashing function will be started. It will then no longer be possible to use the clipboard commands!

Example: $51 $43 $00

This will set the flashing mode block cursor with the white pattern. The area that has been set will therefore flash on a white background.

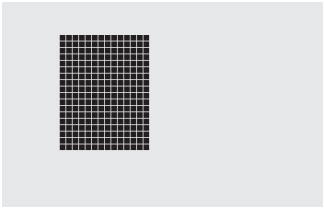
R x1 y1 x2 y2 — Draw rectangle

This draws a rectangle from the top left-hand corner **x1,y1** to the bottom right-hand corner **x2,y2,** taking into account the graphics mode 'V' that has been set (set / delete / inverse). The contents of the rectangle will not be altered in the process. Compare 'O, Draw round corner'.

Example: $52 $15 $08 $30 $25
This will draw a rectangle from 21,8 to 48,37.

**) The command Q M mst requires a display RAM of at least 8 KB. This command cannot be used with displays having a smaller RAM (e.g. 2 KB).*

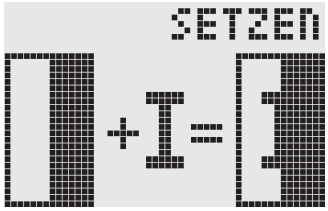
S x1 y1 x2 y2 — Fill area

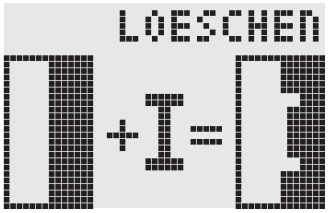


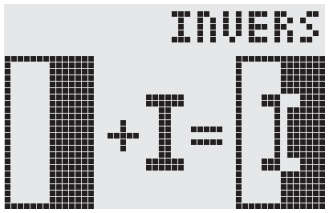
The area extending from the top left-hand corner **x1,y1** to the bottom right-hand corner **x2,y2** will be filled (set to black pixels).

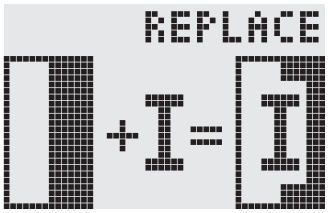
Example: $53 $09 $05 $16 $16
This will set the area extending from 9,5 to 22,22 to black.

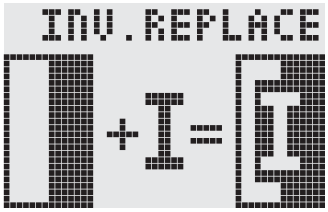
T L/R/O/U n1 ptn — Set text mode











The overlay combination mode, **n1**, and the pattern, **ptn**, will be set for the text functions 'A' (set ASCII character) and 'Z' (output character string). In addition, the write direction is stipulated for the command 'Z' (output character string): **L**=left, **R**=right, **O**=oben (up), and **U**=unten (down).

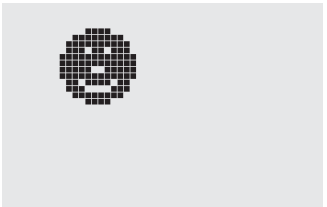
Example: $54 $52 $03 $03

This will set the overlay mode for all of the following text functions to gray characters (pattern 3 = 50% gray) inverted with the background. Character strings are written towards the right.

Overlay combination mode n1:

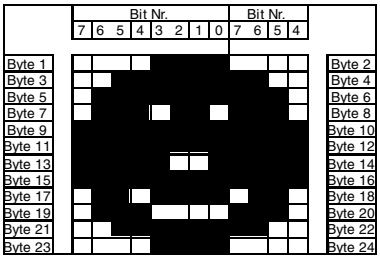
1 = set: Black pixels, irrespective of the previous value (OR).
2 = delete: White pixels, irrespective of the previous value.
3 = inverse: Black pixels become white, and vice versa (EXOR).
4 = replace: Delete background, and set black pixels.
5 = inverse replace: Fill background, and set white pixels.

U x1 y1 data — Upload picture



A picture will be loaded to coordinate **x1,y1**.

data: - 1 byte for the width of the picture in pixels.
- 1 byte for the height of the picture in pixels.
- Picture data: number = ((width+7) / 8) * height bytes.
1 byte stands for 8 horizontal pixels on the screen; 0=white, 1=black;
MSB: left, LSB: right; the picture is stored from top to bottom.
The program BMP2BLH.EXE on disk EA DISKIC1, which is available as an accessory, generates the picture data - including details of width and height - from monochrome Windows bitmap graphics (*.BMP).



Example: $55 $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

This will load the picture alongside to coordinate 9.4.

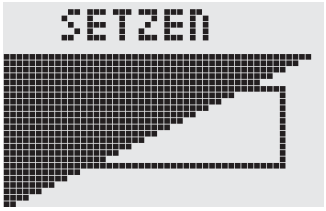
? — Query display type

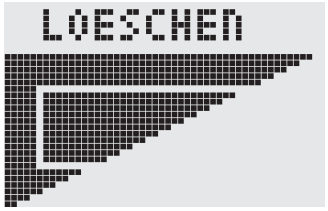
This queries the resolution of the display and the type of picture structure.

Example: $3F

After this command, the X and Y resolution will be sent first over the RS-232 interface, followed by the type of picture structure ('H') for the horizontal organization.

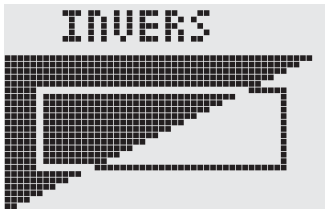
V n1 — Set graphics mode





This sets the overlay combination mode **n1** for the following graphics functions: set point ('P'), draw straight line ('G'), continue straight line ('W'), draw rectangle ('R'), draw round corner ('N'), fill area with pattern ('M').

Example: $56 $03
This will set the overlay mode to inverse.

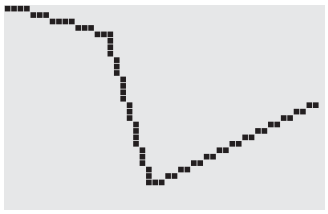


As an example, a rectangle is drawn here on an existing background, with the overlay modes set, delete, and inverse.

Overlay combination mode n1:

1=set: Black pixels, irrespective of the previous value (OR).
2=delete: White pixels, irrespective of the previous value.
3=inverse: Black pixels are changed to white, and vice versa (EXOR).
4=replace: Delete background, and set pixels; area with fill pattern 'ptn' only.
5=inverse replace: Fill background, delete pixels; area with fill pattern 'ptn' only.

W x1 y1 — Continue straight line



This continues a straight line, from the point or the end of the line last drawn, to **x1,y1**, taking into account the graphics mode 'V' that has been set.

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

X n1 — I/O Read port

This reads in a port (**n1**: 0..7 = I/O: 0..7). If **n1** = 8, all I/O 0..7 will be read in as a binary value; I/O 0: LSB, I/O 7: MSB. See application on page 13.

Example: $58 $02
This will read in the level at I/O 2, and will send $00 in the case of level L and $01 in the case of level H via RS-232.

Y n1 n2 — I/O Set port

This changes the port (**n1**: 0..7 = I/O: 0..7) to the value **n2** (0=L level; 1=H level; 2=port invert). If **n1**= 8, all I/O 0..7 will be output as a binary value **n2**; I/O 0: LSB, I/O 7: MSB. See application on page 13.

Example: $59 $02 $01
This will switch the port I/O 2 to H level.

Z x1 y1 ASCII... NUL — Write character string

This writes the character string **ASCII...** to coordinate **x1,y1**, taking into account the text mode 'T' that has been set (set / delete / invert / replace / inverse replace / fill pattern / direction). The character string must be terminated with **NUL** (zero) ($00). The origin (0,0) is at the top left-hand corner of the display. The details of the coordinates refer to the top left-hand corner of the character.

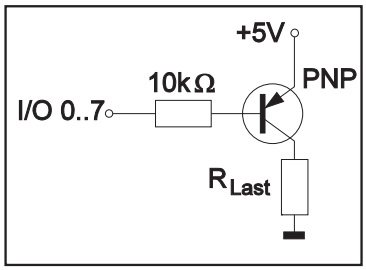
Example: $5A $06 $0B $54 $65 $73 $74 $00
This will write the character string "Test" to coordinate 6,11. Font set: 8x16 with normal width and height. Text mode: Written to the right with overlay mode replace and with black pattern.

DIGITAL INPUTS / OUTPUTS IO 0..7

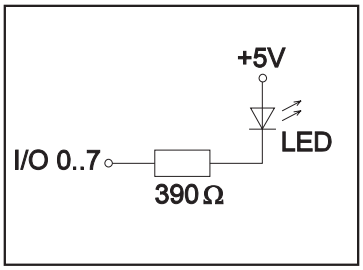
Eight pins on the high-level graphics controller can be used as freely programmable inputs and outputs. They can also be used in various combinations - for instance, three outputs and five inputs.

Connecting as an output

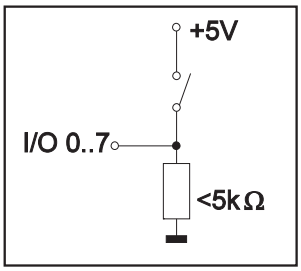
The command "Y n1 n2"[1] can be used to connect each IO 0..7 pin on H or L level. Current can only flow when the level is at L (internal pull-up).



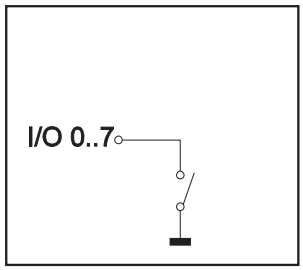
Each pin can supply up to 10mA. The load on all the pins together must not exceed 26mA (e.g. 2x10mA and 1x6mA). It is therefore possible to connect an LED direct to an output. Higher currents can be connected by using an external transistor. After power-on, or after a RESET, all outputs will be at H level.



Connecting as an input



The voltage levels at the input may be between -0.5V and +0.2V*VDD-0.1V. The leakage current can be up to ±10µA. The switching threshold is <0.2*VDD-0.1V for the L level and >0.2*VDD+0.9V for the H level. The command "X n1"[1] can be used to input each IO 0..7 pin. During the entire input process, the voltage level must be stable. There is no in-built debounce function.



[1] *You will find a description of the commands on page 12.*

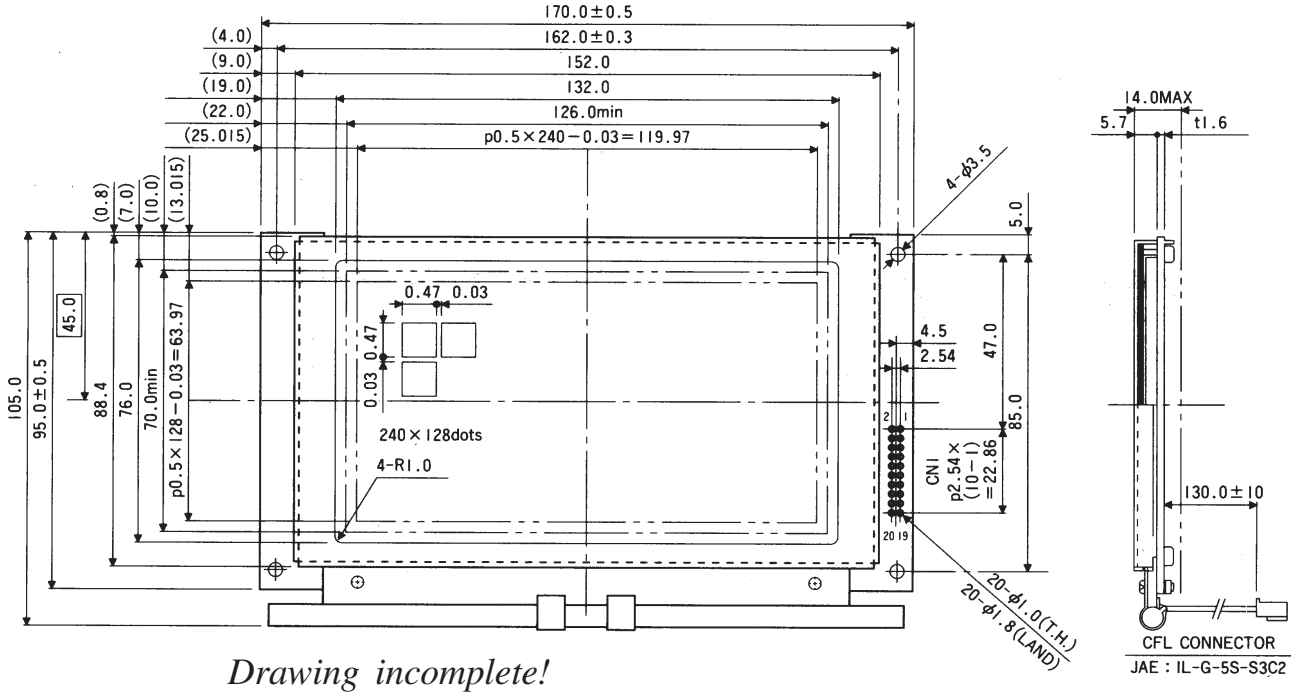
DEFAULT SETTINGS

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

Default settings		
Register	**command**	**after power-on / reset**
Text mode	T	right, set, black
Graphics mode	V	set
Font	F	6x8
Font factor width/height	F	1/1
Last xy	W	(0;0)
Self-defined character	E	undefined
Bar graph 1..8	B	undefined
High-level graphics controller	K	selected
Flashing area	QD	(0;0)
Flashing mode	QC	inverse
Flashing time	QZ	0.6 sec.
Clipboard	C	empty
Inputs / outputs I/O0..7	Y	H level

EA GE240-7K2CV24

240x128 WITH CFL-BACKLIGHT, NEGATIV BLUE

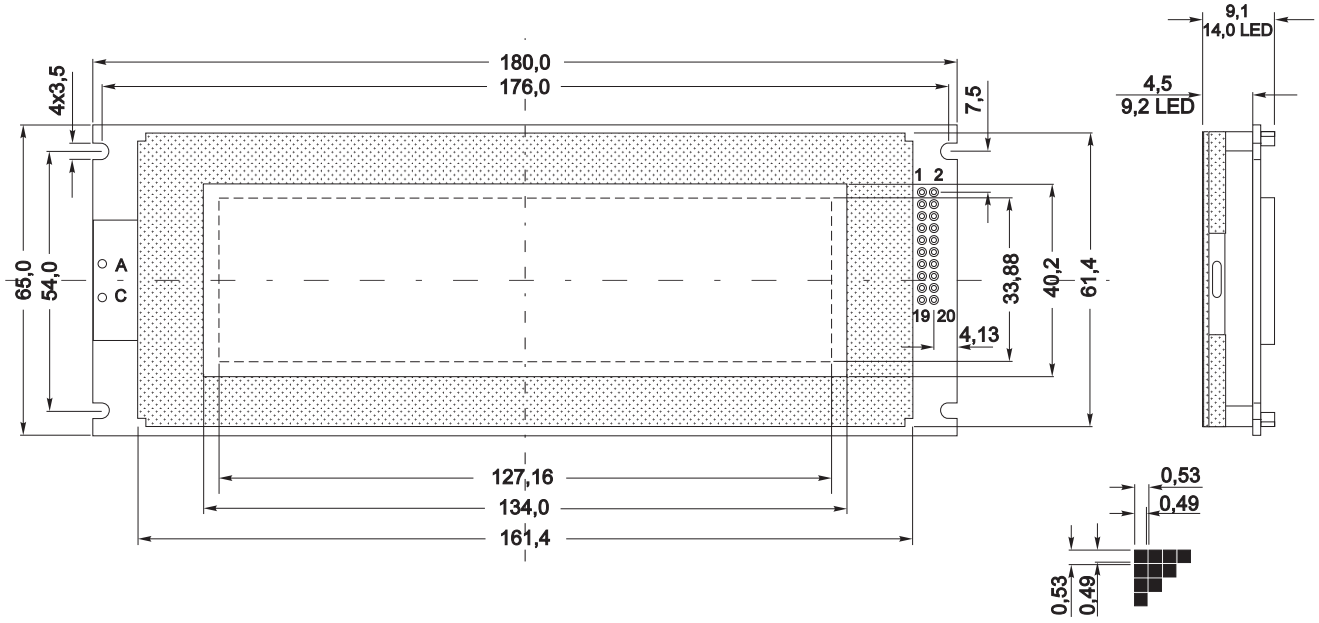


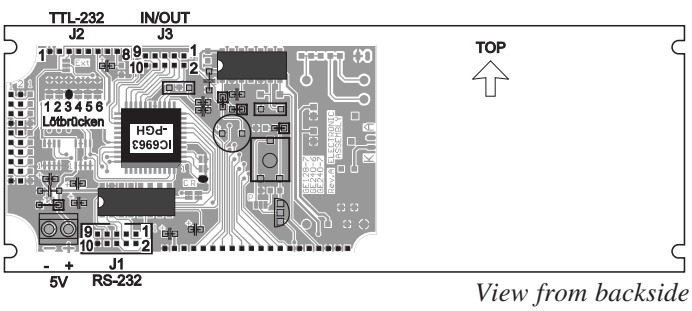
Drawing incomplete!
Thickness max. 30mm!

EA GE240-6KV24

240x64 WITH LED-BACKLIGHT YELLOW/GN

Accessories: Bezel EA 017-10UKE





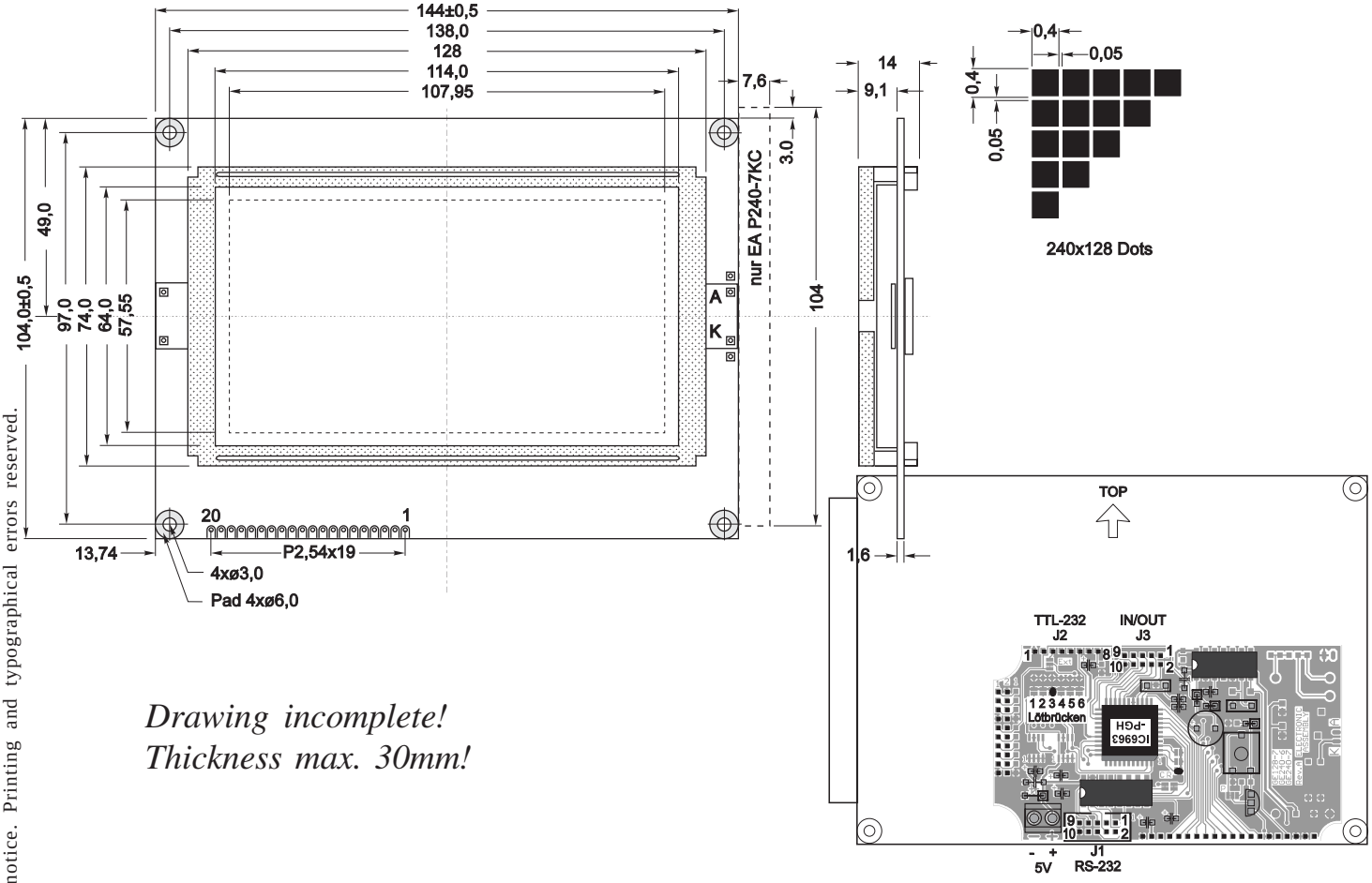
View from backside

Drawing incomplete!
Thickness max. 25mm!

ELECTRONIC ASSEMBLY

EA GE240-7KCV24
EA GE240-7KV24

240x128 WITH CFL-BACKLIGHT, NEGATIV BLUE
240x128 WITH LED-BACKLIGHT YELLOW/GN



Drawing incomplete!
Thickness max. 30mm!

View from backside

GRAPHIC UNIT

BAUD RATES

The baud rate can be set by means of the three left-hand solder straps. When the equipment is delivered, the setting is 9,600 baud. Please note that the internal data buffer is only 56 bytes. When larger quantities of data are being transmitted, the RTS handshake line must be queried (+10V level: data can be accepted; -10V level: display is busy). The data format is set permanently to 8 data bits, 1 stop bit, no parity.

Baudrate			
Solder Bridges			Data Format 8,N,1
1	2	3	
short	short	short	1200
cut	short	short	2400
short	cut	short	4800
cut	cut	short	9600
short	short	cut	19200
cut	short	cut	38400
short	cut	cut	57600
cut	cut	cut	115200

ADDRESSING

Up to four displays can be operated on a serial interface. The address in question is set by means of solder straps 4 and 5.

Adress		
Solder Bridge		Adress
4	5	
short	short	0
short	cut	1
cut	short	2
cut	cut	3

Note: With parallel connection of the RTS handshake lines or the TxD transmission lines, two outputs would work against each other. For this reason, additional hardware must be used to ensure that a data crash cannot occur. It is advisable, for instance, to have a link via OR logic in the case of RTS, or via AND logic in the case of TxD.

DIP SWITCH EA OPT-DIP6

Setting the baud rate and the address by means of the DIP switch spares the board, and means that the settings can be made even when there is no soldering iron at hand. Of course, the test mode can be activated and deactivated at any time in this way as well (DIP switch no. 6). EA OPT-DIP6 is available as an option. (Please state whether you require it when placing your order.)

PIN CONFIGURATION

The supply voltage of +5V is fed in via two screw clamps. The RS-232 data with “real” ±10V levels goes to pin strip J1. J2 has been designed for direct connection to a µC for RS-232 data with 5V levels. When J2 is being used, the solder straps “C” and “R” must be open, or component 232 must be removed! If the contrast voltage V0 is fed to J2, the solder bridge "Ext" must be changed over (cut and solder).

The eight inputs-outputs I/O1..8 are available on connection J3. You will find a more detailed description of the inputs and outputs on page 13.

RS-232 Connector J1			
Pin	Symbol	In/Out	Function
1	VDD	-	+ 5V Supply
2	DCD	-	Connection to DTR
3	DSR	-	Connection to DTR
4	TxD	Out	Transmit Data
5	CTS	In	Clear To Send
6	RxD	In	Receive Data
7	RTS	Out	Request To Send
8	DTR	-	Connection to Pin 2&3
9	-	-	no connection
10	GND	-	0V Masse

Connector J2			
Pin	Symbol	In/Out	Function
1	GND	-	0V Ground
2	VDD	-	+ 5V Supply
3	V0	In	approx.-15V f. contrast
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

Connector J3			
Pin	Symbol	In/Out	Function
1	VDD	-	+ 5V Supply
2..9	I/O0..7	In/Out	In-/Output
10	GND	-	0V Ground

ATTENTION

handling precautions!

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

ELECTRONIC ASSEMBLY GMBH