# EA GE128-6N3V24

# GRAPHIC UNIT 128x64

## INCL.3 FONTS, ZOOM AND LED-BACKLIT

Bargraph Function

Merge Text+Graphic

Pattern

Font Zoom


*Dimensions:*
*84x60x24 mm*

Image Download

### FEATURES

* 3 DIFFERENT FONTS BUILT IN
* ZOOM FUNCTION FOR ALL FONTS (2x, 3x AND 4x)
* EASY PROGRAMMING OF VARIOUS BUILT IN GRAPH FUNCTIONS:
* STRAIGHT LINE, DOT, RANGE, AND/OR/EXOR, BAR GRAPH, PATTERNS...
* MIXING TEXT AND GRAPHICS
* 4-16 FREELY DEFINABLE CHARACTERS (DEPENDING ON SIZE)
* INPUT ON RS-232 / CMOS-LEVEL
* PROGRAMMABLE BAUDRATES FROM 1200 UP TO 115,200 BAUD
* NO TIMING PROBLEMS WITH FAST BUS SYSTEMS
* 8 DIGITAL I/O´S FREELY AVAILABLE FOR CUSTOM DESIGNS
* +5V / typ. 200mA
* HARDWARE CODES UP TO 4 ADRESSES
* DOWNLOAD OF CONVERTED WINDOWS-BMP GRAPHICS

### ACCESSORIES

* PC DISK: SOFTWARE TO CONVERT Windows-BMP GRAPHICS:**EA DISK9719**
* RS-232 CABLE WITH D-SUB 9 (FEMALE) TO TEST ON PC: **EA KV24-9B**

### ORDERING INFORMATION

| | |
|---|---|
| GRAPHIC UNIT 128x64, 3 FONTS, RS-232 | **EA GE128-6N3V24** |
| DIP-SWITCH INSTEAD OF SOLDER STRAPS (BAUDRATES) | **EA OPT-DIP6** |
| BUZZER AT I/O5 (I/O´S WON'T BE USEABLE) | **EA OPT-SUMMER** |

ELECTRONIC ASSEMBLY GMBH
ZEPPELINSTRASSE 19 · D-82205 GILCHING
TEL 08105/778090 ·FAX 08105/778099 · http://www.lcd-module.de

## GENERAL

Our graphics unit EA GE128-6N3V24 is designed for small up to medium quantities. Nearly all known processor systems can be connected within a few hours work to our attractive and informative looking display because of it's simple way to program, it's small outside dimensions and it's excellent supertwist contrast. Input ports accept a serial asynchronous RS-232 interface. The display contains complete graph routines to drive the display and includes also various character sizes.

Programming is made by high level programming language graph commands; time consuming programming of character sets and graph routines is not necessary anymore. Development costs for your products is reduced significant and additional features are gained on top of it:

- no timing problems with fast processor bus
- enough memory space (operating memory and characterset memory especially for µC)
- no time consuming graphic calculations which would slow down processor speed

Drivers, decoders or port modules are not required. Display control can be made in simple cases through one RxD line only.

## HARDWARE

Supply voltage of system is +5 Volts. Data transfer is asynchronous serial in RS-232 format at CMOS level with true RS-232 level (±10V) or with 5V CMOS level. Data format is set firmly to 8 databits, 1 stop bit and no parity. Baud rate can be selected on 3 solder pins from 1,200 Baud up to 115,200 Baud. Handshake lines RTS and CTS are available also. On small amounts of datas there is no interpretation required.

Data format:

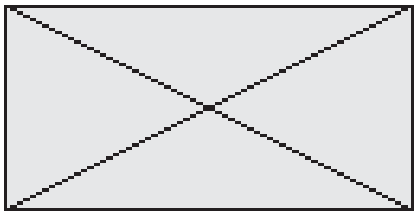

Additional 8 I/O-ports are available on J3 eyelet strip for freely usage. This may be wired as inputs or outputs on individual desire. Possible application is switching of a transistor / relais ($IL_{max}$=10mA) or read in of keystrokes / switches. Required pullup resistor network can be soldered dierctly onto pc- board.

## SOFTWARE

Programming of this High-Level graphics controller is performed by commands like i.e. "plot a rectangular box from (0,0) to (64,15) which origins in top left hand corner of display". Therefore the serial interface has to transmit this sequence of bytes: $52 $00 $00 $40 $0F. Character strings can be placed exactly to the pixel. Merge of graphic images with text elements is possible anytime. Three different character sets are available where each of them can be zoomed from 2x, 3x to 4x. The biggest characterset 16x8 shows when using 4x zoom (=64x32) a totally filled display with letters and numbers.

## TESTMODE

As long as pin 15 (RTS) is after Power-On or after Reset connected with GND, the graphics controller is in test mode. Display shows now a cross marked flashing box. When connection Pin 15 (RTS) to GND is removed, the Graphics controller returns to normal operation mode but testbox remains still visible.



## INTEGRATED FONTS

Graphics device EA GE128-6N3V24 contents three integrated character sets (font1: 4x6 pixel; font 2: 6x8 pixel and font 3 8x16 pixel). Each character set can be used in 1-, 2-, 3- or 4-times height. Their width can be doubled, tripled or quadrupled, independent of height. In addition you can define 4-16 characters on your own which remain alive as long as power is on (see command 'E').

| 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*

| 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 | ƒ |
| $A0 (dez: 160) | á | í | ó | ú | ñ | Ñ | ª | º | ¿ | ⌐ | ¬ | ½ | ¼ | ¡ | « | » |
| $B0 (dez: 176) | ░ | ▒ | ▓ | │ | ┤ | ╡ | ╢ | ╖ | ╕ | ╣ | ║ | ╗ | ╝ | ╜ | ╛ | ┐ |
| $C0 (dez: 192) | └ | ┴ | ┬ | ├ | ─ | ┼ | ╞ | ╟ | ╚ | ╔ | ╩ | ╦ | ╠ | ═ | ╬ | ╧ |
| $D0 (dez: 208) | ╨ | ╤ | ╥ | ╙ | ╘ | ╒ | ╓ | ╫ | ╪ | ┘ | ┌ | ▀ | ▄ | ▌ | ▐ | ▪ |
| $E0 (dez: 224) | α | ß | Γ | π | Σ | σ | µ | τ | Φ | Θ | Ω | δ | ∞ | ø | ε | ∩ |
| $F0 (dez: 240) | ≡ | ± | ≥ | ≤ | ⌠ | ⌡ | ÷ | ≈ | ° | ∙ | · | √ | n | ² | ■ | |

*Font 2*

Each individual character can be placed precisely to the pixel. You may mix text with graphics in any way at your desire. Several different character 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*

## OVERVIEW OF ALL GRAPHIC FUNCTIONS

This graphic unit can be programmed by a number of built in commands. Each command starts with a command letter and will be extended by several parameters.

**Command table EA GE128-6N3V24**

| 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..4); 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). |
| Display control | C | n1 | | | | | | | | | n1=0: display off (entire contents unchanged) n1=1: display on |
| 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) |
| Power Save Mode | Q | n1 | | | | | | | | | n1=1: Power Save for controller; RTS->high<br>n1=2: Power Save for controller and display; RTS->high, LCDON->low |
| 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) |
| **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: 128 64 V (128x64 dots, vertical picture) |

## PARAMETER

All commands with parameters, coordinates and other handover datas are expected in form of Bytes. No space characters are allowed, i.e. no space bars, no commas. End of command **does not need a closing byte** such as a Carrige Return.

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

**x1, x2, y1, y2 .....** Coordinates are transmitted with one byte, applicable values are 0..127 for x- resp. 0..63 for y- coordinates.
Example: x1= 10 (dec.) = $0A

**n1,n2,nr,aw,ew,value,ptn,data .....** Parameters with numbers are transmitted with one byte.
Example: n1= 15 (dec.) = $0F

## EXAMPLE OF PROGRAMMING

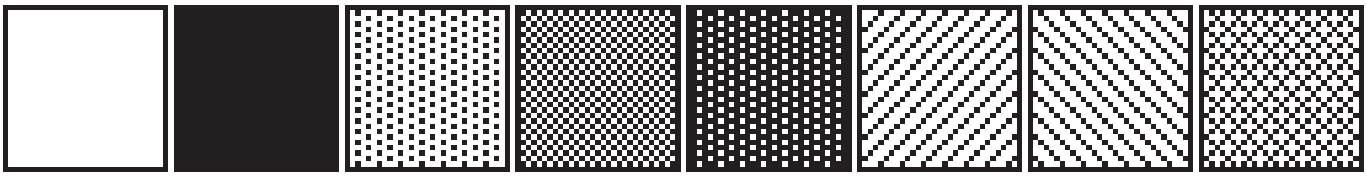
Below table shows the character string "Test" which is displayed at coordinates 7,3.

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

## PATTERN

Several commands allow setting of pattern type parameters (ptn = 0..7). They will link and display rectangular areas, bargraphs and even text lines with various pattern.

This pattern are available:



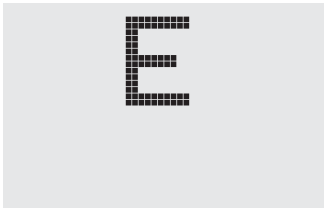
ptn=0 white | ptn=1 black | ptn=2 25% black | ptn=3 50% black | ptn=4 75% black | ptn=5 45° right | ptn=6 45° left | ptn=7 45°crosshatch

## DESCRIPTION OF INDIVIDUAL GRAPH FUNCTIONS

Coming pages show detailed descriptions in alphabetical order for each individual function. Examples are shown as hardcopy in an enlarged window of 50 x 32 pixel once the command has been executed. Examples show transfered Bytes all in Hex codes.

### A x1 y1 n1 — Set ASCII-Characters

A character **n1** will be displayed on coordinates **x1**,**y1** with preset font 'F' and text mode 'T' (set / delete / invert / replace / invers replace / pattern). Origin is (0,0) at top left hand corner of display. Datas for coordinates apply also to top left hand corner of a given character. Note: Font No.1 shows capital letters only.



Example: $41 $13 $02 $45

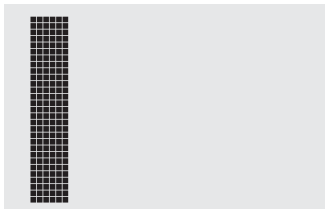
Character 'E' will be displayed at coordinates 19,2

Preset font: 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 8 bar graphs (**nr**=1..8) can be defined. These can extend **L**=left, **R**=right, **O**=top or **U**=bottom direction. Bar graph full level range coordinates are described from **x1**,**y1** to **x2**,**y2**. Scaling of bar graph is performed by starting zero position **aw** (=0..254) and max. ending position (full size) **ew** (=0..254). Bar graph always is displayed in inverse-mode using the **ptn-**pattern type: the background remains preserved in any case. (Note: executing this command, only the bar graph range is defined but will not yet be visible on display).



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

Defines bar graph no. 1 which extends upwards. At full level its coordinates ranges from 4,2 to 9,30. Displayed start- and end- values represent a current value of 4..20 mA. (Hardcopy shows bargraph at its full level operating at $42 $01 $14).

### B nr value — Draw Bar graph

Bar graph number **n1** (1..8) will be adjusted to a new value (aw <= **value** <= ew).

If **value** > ew, than final value value will be displayed. Shape of bar graph must be defined first, see above example.

Example: $42 $01 $0A

Above defined bargraph no. 1 is set here to value 10.

### C n1 — Display Control

switches the display ON (**n1**=1) or OFF (**n1**=0); all datas remain in display and commands can be still executed.

Example: $43 $00

Content of display will be invisible, however datas are preserved.

## D L/I/S — Display command

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

Example: $44 $49

This will invert the entire content of the display.

## E n1 data — Define character

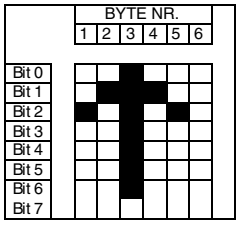
You can define up to 16 characters yourself (depending on size of font). These characters will then have the ASCII codes 1 to max.16, and will remain in an invisible screen RAM of 64 bytes until the supply voltage is switched off. In the case of font 1, up to 16 characters can be defined; with font 2 up to 10 characters; and with font 3, the largest, up to 4 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 4 of the 4x6 font (see the table alongside).

| Define characters (ASCII) | | |
|---|---|---|
| 4x6 | 6x8 | 8x16 |
| 1 | 1 | 1 |
| 2 | | |
| 3 | 2 | |
| 4 | 3 | 2 |
| 5 | | |
| 6 | 4 | |
| 7 | 5 | |
| 8 | | |
| 9 | 6 | 3 |
| 10 | 7 | |
| 11 | | |
| 12 | 8 | |
| 13 | 9 | 4 |
| 14 | | |
| 15 | 10 | |
| 16 | | |

Example 1:

Commands

$45 $03
$04 $02 $7F $02 $04 $00
define an up arrow for ASCII no. 1, using the character set 6x8.



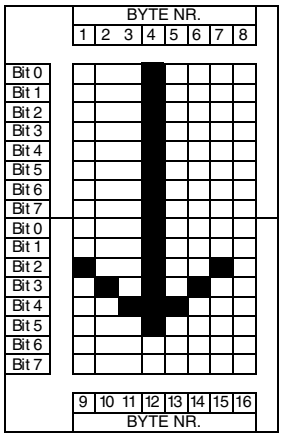

Example 2:

Commands

$45 $02
$00 $00 $00 $FF $00 $00 $00 $00
$04 $08 $10 $3F $10 $08 $04 $00

define a down arrow for ASCII no. 2, using the character set 8x16.

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

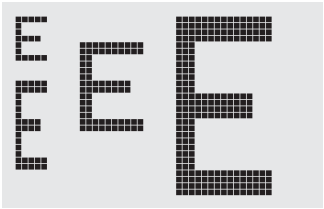



## F n1 n2 n3 — Set font

Font no. **n1** (1=4x6 capital letters only; 2=6x8; 3=8x16) will be set. In addition, a zoom factor (1..4 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 times 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 zoom factors.

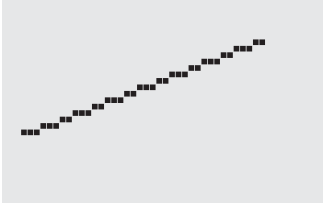


## G x1 y1 x2 y2 — Draw straight line

A straight line will be drawn from coordinates **x1,y1** to coordinates **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 — Get a hard copy of the display

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 transmit the width and height of the section of the image, followed by the image data. See the upload image command, 'U', for the structure of the image data.

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

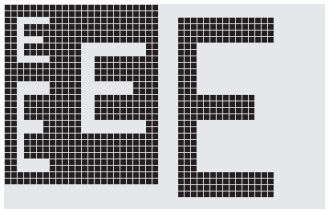
The top left-hand part of the screen, measuring 32 x 16 pixels, will be sent immediately via RS-232.

## 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 is shown as in example under "Set font".

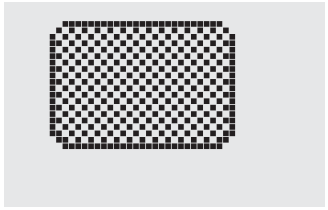


## J x1 y1 x2 y2 ptn — Draw rounded box

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 box with rounded corners'.

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

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



## K S/D n1 — Select / deselect 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 (see page 12, Adressing).

Example: $4B $44 $00

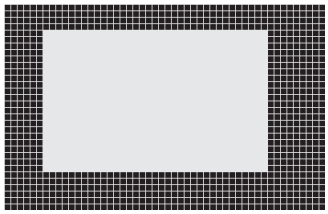
All commands for the graphics controller with address $00 will be ignored effective immediate.

## L x1 y1 x2 y2 — Clear specified display 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.

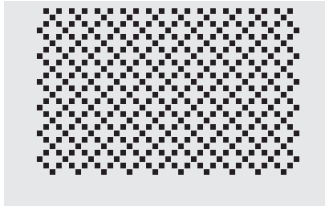


## M x1 y1 x2 y2 ptn — Area with fill pattern

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.

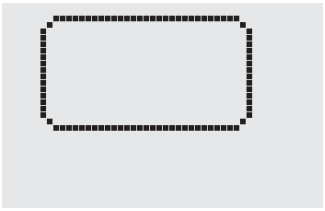


## N x1 y1 x2 y2 — Draw round corner

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 box will not be altered. Compare 'J, Draw rounded box'.

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

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

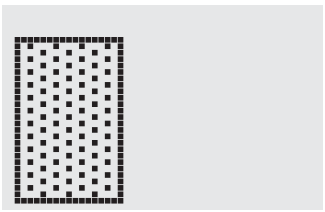


## O x1 y1 x2 y2 ptn — Draw box

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 coordinates 2,5 to 18,30, with the pattern 2=25% black.



## P x1 y1 — Place a dot

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

Example: $50 $OD $11

This will set the pixel at coordinates 17,13.

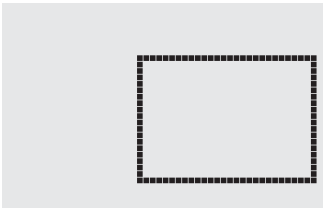


## R x1 y1 x2 y2 — Draw rectangle

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 this procedure. Compare 'N, Draw round corner' on page 12.

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

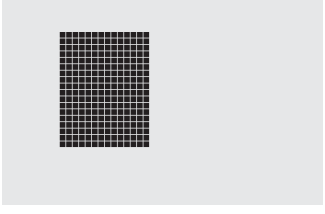
This will draw a rectangle from position 21,8 to 48,37.



## 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 with black pixels.

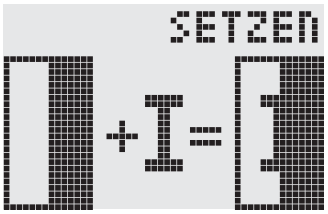
Example: $53 $09 $05 $16 $16
fills an area extending from 9,5 to 22,22 with black pixels.

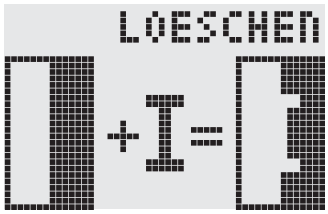


## T L/R/O/U n1 ptn

### Set text mode

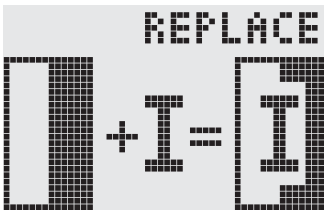
The overlay combination mode **n1**, and pattern mode **ptn** will be set for the text functions 'A' (set ASCII- character) and 'Z' (set character string). In addition, the write direction is stipulated for the command 'Z': **L**=left, **R**=right, **O**=up, and **U**=down.

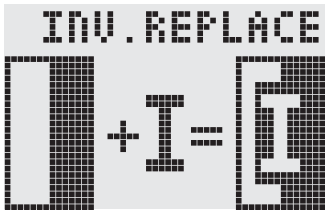



Example: $54 $52 $03 $03

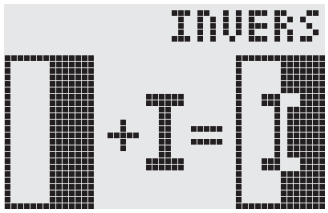
This will set the overlay combination mode for all of the following text functions to gray characters (pattern 3 = 50% black), inverts the background and writes the character string from left to right.




Overlay combination mode n1:

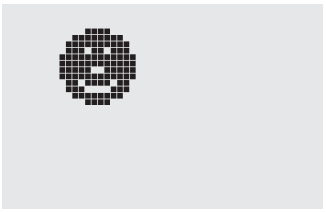
1 = set: Black pixels, irrespective of the previous value (OR).
2 = delete: White pixels, irrespective of the previous parameter.
3 = inverse: Black pixels become white, and vice versa (EXOR).
4 = replace: Delete background, and set black pixels.
5 = inverse replace: Fill background, and set white pixels.

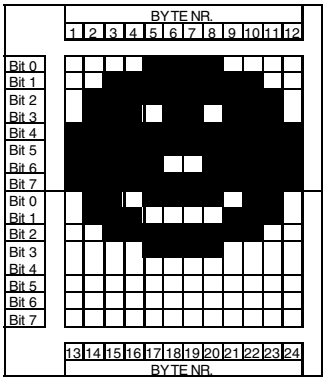



## U x1 y1 data

### Upload picture

An image will be loaded to coordinates **x1,y1**.






**Datas:** - 1 byte for width of picture in pixels.
- 1 byte for height of picture in pixels.
- picture data: pixel qty= ((height+7) / 8) x width bytes.
1 byte stands for 8 vertical pixels on the screen;
0=white, 1=black; LSB: top, MSB:bottom.
The image builds up from left hand side to right hand side.
Programme BMP2BLV.EXE generates the image data - including details of width and height - from monochrome Windows bitmap graphics.

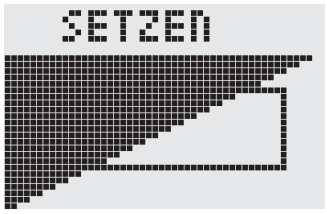
Example: $55 $09 $04 $00 $00
$F0 $FC $FE $FE $F7 $BF $BF $F7 $FE $FE $FC $F0
$00 $03 $07 $06 $0D $0D $0D $0D $06 $07 $03 $00

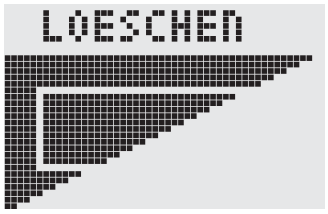
Loads beside shown image to coordinates 9.4.

## V n1

### Set graphics mode

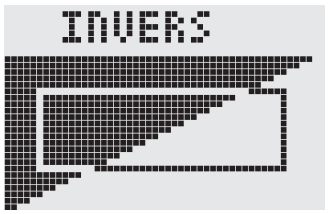
This sets the overlay combination mode **n1** for the following graphics functions: set point ('P'), draw straight line ('G'), continue drawing straight line ('W'), draw rectangle ('R'), draw round corner ('N'), fill area with pattern ('M').




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

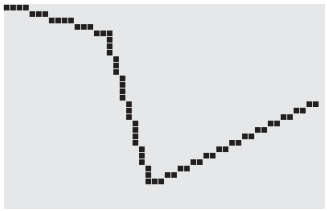
As an example, a rectangle is drawn here on an existing background with overlay combination mode, set, delete, and inverse.




Overlay combination mode n1:

1=set: Black pixels, irrespective of previous parameter (OR).
2=delete: White pixels, irrespective of previous parameter.
3=inverse: Black pixels are changed to white, and vice versa (EXOR).
4=replace: Clear background and set pixels inside area with fill pattern 'ptn' only.
5=inverse replace: Fill background, delete pixels from 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 (see page 7 Draw straight line) to **x1,y1**, taking into account the graphics mode 'V' (set / delete / inverse) that has been set.

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

Draws first a straight line from 0,0 to 16,4. It will then be continued to 22,27 and to 48,15.

## X n1 — Read I/O port

This reads in 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.
Application note on request.

Example: $58 $02

This will read in signals at I/O 2, and will transmit $00 in the case of level L and $01 in the case of level H via RS-232.

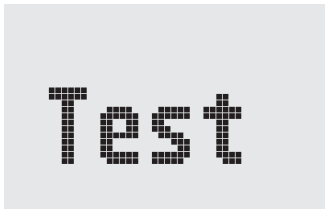
## Y n1 n2 — Set I/O port

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

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 coordinates **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 top left-hand corner of display. Datas of coordinates refer to top left-hand corner of character. Note that font 1 has capital letters only.

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

This will write the character string "Test" starting at coordinates 6,11.
Font set is: 8x16 with normal width and height.
Text mode: Written from left to right in link mode replace and with black pattern.

## ? — Query display type

This queries the resolution of the display and the type of image structure. Resolution of EA 128-6N3V24 is always 128 x 64 pixel with vertical image organisation. External programs can access the High-Level-Graphic-Controller with this command, i.E. IC6963-PGH is designed for displays with Toshiba Graphic Controller featuring a variable resolution up to 240 x 128 pixel with horizontal image build up.

Example: $3F

This command transmits first the X-(128) and Y-(64) resolution over the RS-232 interface and afterwards the type of image structure ('V') for vertical organisation.

# EA GE128-6N3V24

## BAUDRATES

Baudrate is set on 3 left hand located solder bridges. Factory setting is 9.600 Baud. Note that the internal data buffer holds 20 bytes only. Make sure that the handshake line RTS is queried when transmitting higher data volumes (+10V level: accept datas; -10V level: display is busy). Data format is fixed 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 4 displays can be connected onto one serial interface. Respective addresses are allocated by solder bridges 4 and 5.
Attention! Simple parallel-connecting of handshake lines RTS resp.transmitter-lines TxD two outputs

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

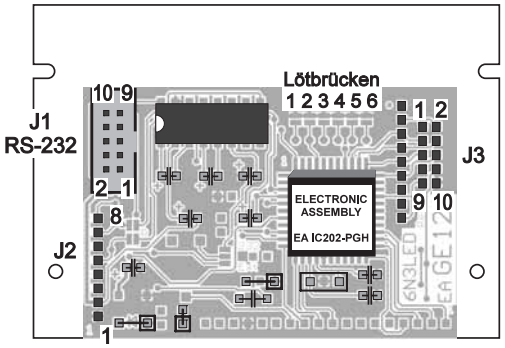
## PINNING

J1 features true RS-232 level (±10V). J2 is designed for 5V-direct connection to an µC. Solder bridges "R" and "C" must be opened or circuit 232 must be removed in case of using J2!
When feeding J1 or J2 with contrast voltage V0, you must change the solder bridge from "232" to "Ext".

| RS-232 Connector J1 | | | |
|---|---|---|---|
| Pin | Symb | 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 | V0 | In | approx. -9V f. contrast |
| 10 | GND | - | 0V Masse |

| Connector J2 | | | |
|---|---|---|---|
| Pin | Symb | In/Out | Function |
| 1 | GND | - | 0V Ground |
| 2 | VDD | - | + 5V Supply |
| 3 | V0 | In | approx. -9V 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 |

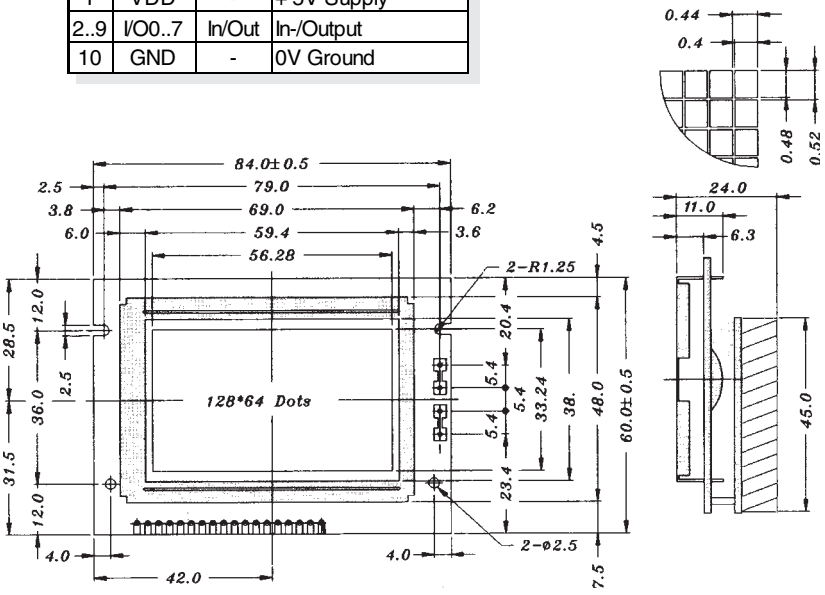
| Connector J3 | | | |
|---|---|---|---|
| Pin | Symbo | In/Out | Function |
| 1 | VDD | - | + 5V Supply |
| 2..9 | I/O0..7 | In/Out | In-/Output |
| 10 | GND | - | 0V Ground |




*Bottom view*

## DIMENSIONS

all sizes in mm




**ATTENTION**

handling precautions!


ZEPPELINSTRASSE 19 · D-82205 GILCHING
TEL 08105/778090 ·FAX 08105/778099 · http://www.lcd-module.de


ELECTRONIC ASSEMBLY GMBH