FEATURES

* 5.7" COLOR LCD GRAPHIC DISPLAY WITH MANY GRAPHICS FUNCTIONS AND FONTS
* 320x240 PIXEL, 16 COLOURS WITH CFL BACKLIGHT
* FONT ZOOM OF approx. 2mm UPTO approx. 80mm, FONT ROTATION, 90° STEP
* SUPPLY VOLTAGE +5V±2% @ 320mA OR +9..35V OPTIONALLY
* RS-232 OR RS-422 WITH BAUD RATE 1,200~115,200 BD
* POSITIONING ACCURATE TO THE PIXEL WITH ALL FUNCTIONS
* STRAIGHT LINE, POINT, AREA, AND/OR/EXOR, BAR GRAPH...
* CLIPBOARD FUNCTIONS, PULL-DOWN MENUS
* STORE UPTO 256 BITMAPS
* UPTO 1380 MACROS PROGRAMMABLE (512kB FLASH ONBOARD)
* TEXT AND GRAPHIC CAN BE MIXED, FLASHING TEXT, INVERT TEXT
* SWITCH BACKLIGHT BY SOFTWARE CONTROL (OFF, ON, HALF BRIGHTNESS)
* ANALOGUETOUCHE PANEL: CAN BE SET INDIVIDUALLY (e.g. 10x8 FIELDS)
* DEFINE KEYS AND SWITCHES
* OPERATE MENU AND BARGRAPH ADJUSTMENT BY THE USE OF TOUCH PANEL
* 8 DIGITAL INPUT AND 8 OUTPUT LINES
* 2 ANALOGUE INPUT 0..200mV WITH SCALING FUNCTION
* CONNECTOR FOR ALPHANUMERIC DISPLAY WITH HD44780

ORDER INFORMATION

CONTROL UNIT COLOR 5.7", TOUCH PANEL, 320x240 DOTS, RS-232
SAME BUT WITHOUT TOUCH PANEL

SUPPLY VOLTAGE +9..35V= INSTEAD OF +5V=
RS-422 INTERFACE INSTEAD OF RS-232
OPTO COUPLER (8xIN, 8xOUT) ON BOARD
ALUMINIUM BEZEL FOR MOUNTING, BLACK ANODIZED
CABLE 1.5m WITH 9-PIN SUB-D (RS-232 FEMALE)

EA KIT320F-8LWTP  Outline dim. 155 x 115mm

EA KIT320F-8LW

EA OPT-9/35V

EA OPT-RS4224

EA OPT-OPTO16

EA 0FP320F-8SW

EA KV24-9B
CONTENT

GENERAL .................................................................................................................... 3
HARDWARE, CONNECTOR, PINOUT ........................................................................... 4
DIGITAL INPUT / OUTPUT, KEYBOARD INTERFACE ................................................. 5–7
TOUCH PANEL ......................................................................................................... 8, 9
MACRO PROGRAMMING .......................................................................................... 10, 11
CHARACTER SETS ................................................................................................. 12, 13
COLORS AND ATTRIBUTES .................................................................................. 14
COMMAND SYNTAX, PROGRAMMING EXAMPLE ................................................... 16
COMMAND TABLES ................................................................................................. 17–22
RESPONSE OF CONTROL UNIT ............................................................................ 23
DIMENSIONS .......................................................................................................... 24

Documentation of revision

<table>
<thead>
<tr>
<th>Date</th>
<th>Type</th>
<th>Old</th>
<th>New</th>
<th>Reason / Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.2.04</td>
<td>Simulator</td>
<td>F1: help function</td>
<td>Shift-F1: help function</td>
<td>continuous terminal functionality F1..F10</td>
</tr>
<tr>
<td>5.2.04</td>
<td>Data sheet</td>
<td>Page 6: Extension for analogue input description Page 22: Insert command for calibration</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6.2.04</td>
<td>Data sheet</td>
<td>Page 9: Insert &quot;Create own key form&quot;, &quot;Use bitmap as a key&quot;, &quot;Radio groups&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8.10.10</td>
<td>Display</td>
<td>EA KIT320F-8C</td>
<td>EA KIT320F-8LW</td>
<td>Backlight changed from CFL into LED type</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EA KIT320F-8CTP</td>
<td>EA KIT320F-8LWTP</td>
<td></td>
</tr>
</tbody>
</table>

5.2.04

Date Type Old New Reason / Description
5.2.04 Simulator F1: help function Shift-F1: help function continuous terminal functionality F1..F10
5.2.04 Data sheet Page 6: Extension for analogue input description Page 22: Insert command for calibration
6.2.04 Data sheet Page 9: Insert "Create own key form", "Use bitmap as a key", "Radio groups"
8.10.10 Display EA KIT320F-8C EA KIT320F-8LW Backlight changed from CFL into LED type
GENERAL
The EA KIT320F graphics kit is a fully assembled control and operating unit with a variety of integrated functions. The display has very compact dimensions and offers excellent super-twist contrast, which means the unit can be put into operation immediately. It is controlled via the standard RS-232 or RS-422 interface. In addition to complete graphics routines for display output, the graphics kit also contains a wide variety of fonts. Graphics command similar to those used in high-level programming languages are used for programming. There is thus no longer any need for the time-consuming programming of character sets and graphics routines. The ease of use offered by macros and input via touch panel make it a real power display.

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

```
Startbit D0 D1 D2 D3 D4 D5 D6 D7 Stopbit
```

TOUCH PANEL
The EA KIT320F-8LWTPis equipped with an integrated touch panel. You can make entries and choose menu settings by touching the display. The labeling of the "keys" is flexible and can also be changed during runtime (different languages, icons). The drawing of the individual "keys" and the labeling or grouping of several fields is handled by the integrated software.

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

ACCESSORIES
Front panel for mounting
A front panel made of anodized aluminum is available as an accessory. This allows the graphics kit to be mounted without any screws visible. Installing it is child’s play. The color for the EA 0FP320F-8SW front panel is black.
Creating macros with the Kit-Compiler
With the ELECTRONIC ASSEMBLY LCD-Tools*) individual macros and fonts may be created. More information on macro functionality is on the pages 10 and 11.
Simulator Software for Windows
To get familiar with this display, an easy, fast and cheap way is to download the ELECTRONIC ASSEMBLY LCD-Tools with its Simulator software from our web site. All KIT-functions can be simulated there!
Even later on for software development, this Simulator is a great utility.
Cable for PC
To enable simple connection to PCs (macro programming), we provide a 1.5m cable and a 9-pin SUB-D female connector (EA KV24-9B). Simply insert it into COM 1 or COM 2 and get started. Note: The cable is not suitable for the RS-422 version (EA OPT-RS4224).

*) full version is free available on web at http://www.lcd-module.com/products-touch.html
SUPPLY VOLTAGE / EA OPT-9/35V

In the standard model, the supply voltage of +5V is fed in via screw-type terminal J1. In the case of the version for 9..35V (EA OPT-9/35V), the power is supplied via J2.

Important: It is imperative that the polarity is correct. Polarity reversal, even for a very short time, can cause the immediate destruction of the entire display.

BAUD RATES

The baud rate can be set by means of the 3 DIP switches on the left. 9,600 baud is set at the factory (DIP 3 ON). Please note that the internal data buffer only holds 128 bytes. It is therefore imperative that the RTS handshake line be queried (a level of +10V means data can be accepted; a level of -10V means the display is busy).

The data format is fixed at 8 data bits, 1 stop bit and no parity.

RS-232/RS-422 CONNECTION

The graphics kit is shipped with an RS-232 interface as standard. The pin assignment of the plug connector (J3) is as shown in the table on the left. The J3 has a 2.54mm grid. If the graphics kit is ordered together with the EA OPT-RS4224 optional component, RS-422 drivers are fitted. In this case, the pin assignment is as shown in the table on the right.

The same serial data with 5V levels and TTL logic is available at the J5 eyelet strip. These levels are suitable for direct connection to a µC. However, if these signals are used, 4 solder links TXD, RXD, RTS, CTS had to be cut before!

### RS-232 J3 connection

<table>
<thead>
<tr>
<th>Pin</th>
<th>Symbol</th>
<th>In/Out</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VDD</td>
<td>-</td>
<td>+ 5V supply</td>
</tr>
<tr>
<td>2</td>
<td>DCD</td>
<td>-</td>
<td>Strap to DTR</td>
</tr>
<tr>
<td>3</td>
<td>DSR</td>
<td>-</td>
<td>Strap to DTR</td>
</tr>
<tr>
<td>4</td>
<td>TxD</td>
<td>Out</td>
<td>Transmit data</td>
</tr>
<tr>
<td>5</td>
<td>CTS</td>
<td>In</td>
<td>Clear to send</td>
</tr>
<tr>
<td>6</td>
<td>RxD</td>
<td>In</td>
<td>Receive data</td>
</tr>
<tr>
<td>7</td>
<td>RTS</td>
<td>Out</td>
<td>Request to send</td>
</tr>
<tr>
<td>8</td>
<td>DTR</td>
<td>-</td>
<td>See pin 2, pin 3</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>-</td>
<td>NC</td>
</tr>
<tr>
<td>10</td>
<td>GND</td>
<td>-</td>
<td>0V ground</td>
</tr>
</tbody>
</table>

### RS-422 J3 connection

<table>
<thead>
<tr>
<th>Pin</th>
<th>Symbol</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VDD</td>
<td>+ 5V supply</td>
</tr>
<tr>
<td>2</td>
<td>Data In</td>
<td>Receive data</td>
</tr>
<tr>
<td>3</td>
<td>Data In+</td>
<td>Receive data</td>
</tr>
<tr>
<td>4</td>
<td>Data Out</td>
<td>Transmit data</td>
</tr>
<tr>
<td>5</td>
<td>Data Out+</td>
<td>Transmit data</td>
</tr>
<tr>
<td>6</td>
<td>HS In-</td>
<td>Handshake</td>
</tr>
<tr>
<td>7</td>
<td>HS In+</td>
<td>Handshake</td>
</tr>
<tr>
<td>8</td>
<td>HS Out-</td>
<td>Handshake</td>
</tr>
<tr>
<td>9</td>
<td>HS Out+</td>
<td>Handshake</td>
</tr>
<tr>
<td>10</td>
<td>GND</td>
<td>0V ground</td>
</tr>
</tbody>
</table>

### J5 add-on

<table>
<thead>
<tr>
<th>Pin</th>
<th>Symbol</th>
<th>In/Out</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VU</td>
<td>-</td>
<td>9..35V supply</td>
</tr>
<tr>
<td>2</td>
<td>VDD</td>
<td>-</td>
<td>+ 5V supply</td>
</tr>
<tr>
<td>3</td>
<td>GND</td>
<td>-</td>
<td>0V, ground</td>
</tr>
<tr>
<td>4</td>
<td>TxDS</td>
<td>Out</td>
<td>Transmit data</td>
</tr>
<tr>
<td>5</td>
<td>RxD</td>
<td>In</td>
<td>Receive data</td>
</tr>
<tr>
<td>6</td>
<td>RTS</td>
<td>Out</td>
<td>Request to send</td>
</tr>
<tr>
<td>7</td>
<td>CTS</td>
<td>In</td>
<td>Clear to send</td>
</tr>
<tr>
<td>8</td>
<td>RESET</td>
<td>In</td>
<td>H reset</td>
</tr>
<tr>
<td>9</td>
<td>SCL</td>
<td>Out</td>
<td>2C bus, clock</td>
</tr>
<tr>
<td>10</td>
<td>SDA</td>
<td>In/Out</td>
<td>2C bus, data</td>
</tr>
</tbody>
</table>
DIGITAL INPUT AN OUTPUT

All control units EA KIT320F-8 are featured with 8 digital input and 8 digital output lines (5V CMOS level, grounded).

8 outputs

Each line can be controlled individually using the "ESC Y W" command. A maximum current of 6mA can be switched per line. This gives the opportunity to drive a low power LED in direct way. To source higher current please use an external transistor (see application below).

8 Eingänge

The inputs can also be queried and evaluated directly via the serial interface ("ESC Y R"). In addition to that every port change may start an individual port - or bit- macro. Each of these port macros can change the contents of the screen or switch an output, thus enabling a wide range of control functions. Command "ESC Y A 0" disables automatic port query.

Port Macro: when the 8 lines are combined, up to 256 port macros can thus be addressed.

Bit Macro: watching a single line. Bit Macro 1..8 will be started when one of the lines 1..8 changes to high (rising edge). Bit Macro 9..16 will be started when one of these lines is going to low (falling edge).

If both macros (Port and Bit macro) are defined, every change will start Bit Macro first and then Port Makro. If there's no macro defined, port status will be sent via RS232/RS422.

To create the port macros you need a PC and the EA DISK320F floppy disk.

Note: The logic circuitry is designed for slow operations; in other words, more than 3 changes per second cannot be easily executed. Open input are high because of internal 100 kΩ pull-up.

INPUT AND OUTPUT VIA OPTOCOUPLE (EA OPT-OPTO16)

Digital input and output are optionally featured with optocoupler (EA OPT-OPTO16). All the 8 inputs and 8 outputs are isolated from the rest of the electronic components as well as each other. The connection is made via 16 different screw-type terminals.

All input lines can be connected with 5..35V directly. Level above 4V are H-level, a voltage below 2V stay for L-level. Voltage between 2V and 4V are undefined.

The collector and emitter of a transistor is brought out at the screw-type terminals as the output. Each output can switch a maximum of 10mA.

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

Note: The optocouplers invert the input logic (all inputs open: port macro 255). It is advisable here (in the power-on macro, for example) to use the "ESC Y I 1" command to evaluate the inputs inversely (all inputs open: port macro 0).
INTERFACE FOR TEXT DISPLAY WITH HD44780

Eyelet J7 is a interface for an external alphanumeric LCD with HD44780 controller onboard. All popular sizes from 1x8 up to 4x20 (2x40) characters are supported. Software with terminal functions via command ‘ESC T xx‘ is already built-in. By command ‘ESC L xx‘ low level programming for controller HD44780 is possible. Potentiometer for contrast adjustment is built-in, too. Customer is able to add a series resistor $R_{\text{LED}}$ for LED backlight on-board.

### Table: HD44780 LCD-interface J6 + J7

<table>
<thead>
<tr>
<th>Pin</th>
<th>Symbol</th>
<th>Level</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VSS</td>
<td>L</td>
<td>Power supply, Ground</td>
</tr>
<tr>
<td>2</td>
<td>VDD</td>
<td>H</td>
<td>Power supply +5V</td>
</tr>
<tr>
<td>3</td>
<td>VEE</td>
<td>-</td>
<td>Contrast voltage 0V~5V</td>
</tr>
<tr>
<td>4</td>
<td>RS</td>
<td>H / L</td>
<td>Register Select</td>
</tr>
<tr>
<td>5</td>
<td>R/W</td>
<td>H / L</td>
<td>H: Read / L: Write</td>
</tr>
<tr>
<td>6</td>
<td>E</td>
<td>H</td>
<td>Enable</td>
</tr>
<tr>
<td>7</td>
<td>D0</td>
<td>H / L</td>
<td>Data line 0 (LSB)</td>
</tr>
<tr>
<td>8</td>
<td>D1</td>
<td>H / L</td>
<td>Data line 1</td>
</tr>
<tr>
<td>9</td>
<td>D2</td>
<td>H / L</td>
<td>Data line 2</td>
</tr>
<tr>
<td>10</td>
<td>D3</td>
<td>H / L</td>
<td>Data line 3</td>
</tr>
<tr>
<td>11</td>
<td>D4</td>
<td>H / L</td>
<td>Data line 4</td>
</tr>
<tr>
<td>12</td>
<td>D5</td>
<td>H / L</td>
<td>Data line 5</td>
</tr>
<tr>
<td>13</td>
<td>D6</td>
<td>H / L</td>
<td>Data line 6</td>
</tr>
<tr>
<td>14</td>
<td>D7</td>
<td>H / L</td>
<td>Data line 7</td>
</tr>
<tr>
<td>15</td>
<td>A</td>
<td>-</td>
<td>Anode for LED (RLED)</td>
</tr>
<tr>
<td>16</td>
<td>K</td>
<td>L</td>
<td>Cathode f. LED (=VSS)</td>
</tr>
</tbody>
</table>

ANALOGUE INPUT AIN1 AUND AIN2 ➔ PAGE 20

For analogue measurement 2 inputs with a range of 0...+250mV are available. Each input is grounded (GND) and DC impedance is 10kΩ. Please make sure that only positive voltages will be supplied there. Internal resolution is 10 Bit, equal to a 3-digit DVM module. Linearity (after adjustment) is around 0.5%.

**Adjustment**

Analogue inputs are not calibrated when shipped out. A procedure for adjustment may be like that:
1.) Put a well known voltage within a range of 150-250mV to analogu input (example: 200mV, AIN1)
2.) Run command for calibration (see page 20). Example: "ESC V @ 1 200.0 NUL". This command may be put into a Touch-macro, too, which will be started touching the display.

**Measurement**

Each input query can be done via RS-232 (RS-422) or directly shown on display (as digits or bargraph in various colors and sizes).

Best way for direct visualisation are Process-macros or one of Analogue-macros (e.g. starting at every voltage change, or above/below a limit).

Both input lines are scaleable from 0 to ±9999.9. Scaling will be done via definition at 2 voltages (value1 > string1, value2 > string2).

---

Application example

![Application example](attachment:image1.png)

Divider 1:100 for voltage 0..25V

![Divider 1:100](attachment:image2.png)
EXTERNAL KEYBOARD

A keyboard (anything from individual keys to a 8x8 matrix keyboard) can be connected at the plug-in connector J120. Command 'ESC Y M n1 n2 n3' define the count of input lines (n1=1..8) and output lines (n2=1..8, see page 5). n3 set debounce function with 50ms steps (n3=0..7). Please note that count of digital input and output lines will be reduced while connecting an external keyboard at the same port.

Each key is connected with 1 output and 1 input. All inputs are terminated with a 100kΩ pull-up resistor. For double-keytroke function decoupling of outputs is necessary. For that please use schottky diodes (e.g. BAT 46).

Transmitting the keystrokes

At each keystroke, the associated key number (1..64, see page 21 “ESC M”) is transmitted or - if a corresponding Matrix-Macro is defined, Matrix-Macro will be started. The release of the key is not transmitted. If the release of the key is to be transmitted as well, this can be done by defining Matrix-Macro no. 0.

Note: If the handshake line (e.g. CTS) does not permit transmission, up to 8 keystrokes will be stored in internal buffer. While buffer overrun keystrokes may be lost!

Calculation of key numbers:

**Key number = (output -1) * count of inputs + input**  (output=MOx, count of inputs=MIx).

Example: Connecting 4 keypads in 3 ways

- **2x2 matrix**: Command 'ESCY M 2 2 ..' defines the 2x2 matrix. Keypad will need input lines MI1, MI2 and output lines MO1, MO2. Output lines are decoupled by diodes; this is for double-keystrokes necessary, only. 6 input and 6 output lines remain free for other requirements.

- **1x4 matrix**: Command 'ESCY M 1 4 ..' defines the 1x4 matrix. Keypad will need output lines MO1..MO4 and a single input line MI1. With that connection 7 input and 4 output lines remain free for other requirements.

- **4x0 matrix**: Using one single output only (physically 4x1 Matrix), all keys can switch to GND. So no output line is necessary and comman 'ESCY M 4 0 ..' defines 4 input lines onyl. With that connection 4 input and 8 output lines remain free for other requirements.
EA KIT320F-8 COLOR

TOUCH PANEL (EA KIT320F-8LWTP ONLY)

Version EA KIT320F-8CTP comes with an integrated touch panel, analogue resistive type. Up to 80 touch areas can be defined (summary of all keys, switch, menu, bargraph-input). This control unit supports all of them with many comfortable commands (see page 18). When the touch keys are touched, they can be automatically inverted and a tone can sound, indicating they have been touched. At the same time, the defined return code of the key is transmitted via the serial interface, or an internal touch macro with the number of the return code is started.

TOUCH PANEL ADJUSTMENT

Touch panel is ready to use and well adjusted when module will be shipped out. Because of aging and attrition a re-adjustment may be necessary.

Adjustment procedure:

1. While switching on display, touch the display surface and keep it touched until display shows the question "touch adjustment ?" (or transmit ‘ESC @’ command).
2. Within 1 second touch again the display surface for min. 1 second.
3. Follow instructions for adjustment (touch 2 points topleft and bottomright).

FRAMES AND KEY STYLES

Both commands draw box frame and draw touch defines key style. You can choose one to 20 internal defines frame types; furthermore some personal styles can be defined via compiler k320comp.exe (see DOKU.DOC on floppy disk EA DISK320F: Compiler code "BORDER")

All of them can be used in various sizes via coordinates. Each frame is split into 4 segments: the edges (not for type 1-5), outer frame, inner frame and filling.

Each segment will get an individual colour and attribute. This will give the user the opportunity, when touching a field, the individual part of the key will be inverted only.
SELF-DEFINABLE SHAPES

With the help of Kitcompiler k320comp.exe you are able to generate individually created shapes for touch-fields (see K320COMP.DOC on floppy disk EA DISK320F: compiler directive "BORDER"). Each of these new shapes will be stored as a Windows-BMP file with 16 colours and 24x24 dots resolution (9 segments with 8x8 dots each: 4x edge, 4x middle part, 1x filling). Scaling for bigger touch fields will be done by repetition of these 8x8 dot segments. This makes it necessary to keep the 8x8 size in every case.

The first 4 colors will be used for the 4 frame colors f1..f4 (see also command table on page 20). Auf der Diskette EA DISK320F sind einige Beispielrahmen im Verzeichnis 'DISK320F\BITMAPS\border' vorhanden.

BITMAPS ALSTASTEN

Ausser den Rahmentypen, die in der Grösse frei skalierbar sind, gibt es noch die Möglichkeit beliebige Bitmaps als Touch-Tasten oder -Schalter zu verwenden. Über den Kitcompiler k320comp.exe können bis zu 127 eigene Buttons eingebunden werden (siehe K320COMP.DOC auf der Diskette EA DISK320F: Compileranweisung "BUTTON").

Ein Button besteht immer aus zwei gleich grossen 16-farbigen Windows-BMPs (ein Bitmap für die gedrückte Touchtaste und ein Bitmap für normale Darstellung der Touchtaste). Die aktive Fläche der Touchtaste ergibt sich automatisch aus der Grösse der Button-Bitmaps.

Werden Buttons nachträglich beschriftet, so muss das Farbattribut SOLID gesetzt sein um ein Überschreiben des Textes beim Drücken zu verhindern.


Auf der Diskette EA DISK320F sind einige Beispielbuttons im Verzeichnis 'DISK320F\BITMAPS\button' vorhanden.

SCHALTER IN GRUPPEN (RADIO GROUP)

MACRO PROGRAMMING

Single or multiple command sequences can be grouped together in macros and stored in the Data-FLASH. You can then start them by using the Run macro commands. There are several different types of macros:

Normal Macros (0..255)
These are started by means of a command via the serial interface (ESC MN xx) or from another macro. A series of macros occurring one after the other can be called cyclically (movie, hourglass, multi-page help text). This kind of automatic macro will run until data from RS-232/422 will be received or until another macro will be started (Touch- Port- or matrix-Macro).

Touch Macros (1..255)
These are started when you touch a touch field (in versions with a touch panel - TP) or command ‘ESC MT xx’ will be received. Touch macro no. 0 is different: It is started when you release a key.

Menu Macros (1..255)
will start after a choice in menu entry or by command ‘ESC MM xx’.

Bit Macros (1..8) and (9..16)
will start voltage at a single line IN 1..8 (bitweise) will change or by command ‘ESC MB xx’. Bit- Macro 1..8 are good for rising edge and Bit Macro 9..16 are good for falling edge at input 1..8.

Port macros (0..255)
These are started when voltage (binary) is applied to IN 1..8 or by command ‘ESC MP xx’.

Matrix Macros (0..64)
Matrix Macro 1..64: start when keypressed or by command ‘ESC MX xx’.
Matrix Macro 0: start after release of key or by command.
For more details please refer to page 7.

Analogue Macros (0..19)
will start whenever voltage changes or limit exceeds or by command ‘ESC MV xx’. See table at the right:

Process Macros (0..255)
automatic start at fixed periode (0.1s up to 25s) or by command ‘ESC MC xx’. Up to 16 individual process may be defined by command ‘ESC MD ..’ . These Process Makro will never be stopped by other commands or activities.

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

Reset Macro
Start after external reset or power supply break-down below 4.7V (VDD-VSS).

Watchdog Macro
Start after system error.

Note: Doing with Power-On-, Reset- or Watchdog Macro an endless loop, display can no longer be reached. In that case: set DIP switch no. 5 to ON position, power-off, power-on, and then DIP 5 back to OFF. All Macros are need to be downloaded again.

<table>
<thead>
<tr>
<th>Analogue Macro</th>
<th>Macro starts at</th>
</tr>
</thead>
<tbody>
<tr>
<td>Macro No.</td>
<td>AIN1</td>
</tr>
<tr>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>2</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>13</td>
</tr>
<tr>
<td>4</td>
<td>14</td>
</tr>
<tr>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td>6</td>
<td>16</td>
</tr>
<tr>
<td>7</td>
<td>17</td>
</tr>
<tr>
<td>8</td>
<td>18</td>
</tr>
<tr>
<td>9</td>
<td>19</td>
</tr>
</tbody>
</table>
CREATING INDIVIDUAL MACROS AND IMAGES

- ELECTRONIC ASSEMBLY LCD-Tools\(^\d\), which contains a kit editor, kit compiler and examples and fonts (for Windows PCs)
- A PC with a serial COM interface

To define a sequence of commands as a macro, all the commands are written to a file on the PC (e.g. DEMO.KMC). You specify which character sets are to be integrated and which command sequences are to be in which macros.

If the macros are defined using the kit editor, you start the eDIP320 compiler using F5. This creates a file called DEMO.DF. If the display is connected to the PC, this file is automatically downloaded in the display’s data flash memory.

You will find a detailed description of the programming of the macros together with examples in the ELECTRONIC ASSEMBLY LCD-Tools\(^\d\) help system.

WRITE PROTECTION FOR PROGRAMMED MACROS

You can use DIP switch # 6 and set to OFF position to prevent the programmed macros, images and fonts from being inadvertently overwritten. Re-programming the FLASH memory need to have the DIP switch #6 at ON.

STORING 256 IMAGES IN THE ON-BOARD FLASH PROM

To reduce the transmission times of the serial interface or to save storage space in the processor system, up to 256 images can be stored in internal FLASH PROM. They can be called using the "ESC U I" command via the serial interface or from within any macro. All the images can be used in the Windows .BMP format (monochrome or 16 colors). They can be created and edit using widely available software such as Windows Paint or Photoshop.

\(^\d\) full version is free available on web at [http://www.lcd-module.com/products/touch.html](http://www.lcd-module.com/products/touch.html)
INTEGRATED FONTS

6 mono-spaced, 6 proportional character sets and 1 big numbers are built-in. Terminal mode will display mono-spaced fonts only. In graphics mode all fonts can be used accurate to the pixel. Imaging of proportional font is more beatiful. And it is space-saving (e.g. a narrow "i" do need less room as the fat "W").

Each character set can be used in graphics mode from normal up to 8-times height. Independently of the height, the width can also be increased two to eight times.

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

<table>
<thead>
<tr>
<th>No.</th>
<th>Name</th>
<th>Char. height</th>
<th>Lines x chars.</th>
<th>Size in pixels</th>
<th>ASCII area</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>FONT4x6</td>
<td>2.1 mm</td>
<td>40 x 80</td>
<td>4 x 6</td>
<td>32 - 158</td>
<td>micro size, mono-spaced</td>
</tr>
<tr>
<td>2</td>
<td>FONT5x6</td>
<td>2.1 mm</td>
<td>40 x 64</td>
<td>5 x 6</td>
<td>32 - 158</td>
<td>mini size, mono-spaced</td>
</tr>
<tr>
<td>3</td>
<td>FONT6x8</td>
<td>2.8 mm</td>
<td>30 x 53</td>
<td>6 x 8</td>
<td>32 - 255</td>
<td>extended ASCII code without frame symols (176-223) mono-spaced</td>
</tr>
<tr>
<td>4</td>
<td>FONT8x8</td>
<td>2.8 mm</td>
<td>30 x 40</td>
<td>8 x 8</td>
<td>32 - 255</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>FONT7x12</td>
<td>4.3 mm</td>
<td>20 x 45</td>
<td>7 x 12</td>
<td>32 - 255</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>FONT8x16</td>
<td>5.7 mm</td>
<td>15 x 40</td>
<td>8 x 16</td>
<td>32 - 255</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>10 free definable character sets</td>
</tr>
<tr>
<td>17</td>
<td>GENEVA12</td>
<td>4.3 mm</td>
<td>20 x 32&gt;</td>
<td>&lt;10 x 12</td>
<td>32 - 168</td>
<td>Geneva, proportional</td>
</tr>
<tr>
<td>18</td>
<td>CHICAGO16</td>
<td>5.7 mm</td>
<td>15 x 26&gt;</td>
<td>&lt;12 x 16</td>
<td>32 - 168</td>
<td>Chicago, proportional</td>
</tr>
<tr>
<td>19</td>
<td>TIMES20</td>
<td>7.2 mm</td>
<td>12 x 17&gt;</td>
<td>&lt;18 x 20</td>
<td>32 - 168</td>
<td>Times, proportional</td>
</tr>
<tr>
<td>20</td>
<td>TIMES26</td>
<td>9.3 mm</td>
<td>9 x 13&gt;</td>
<td>&lt;24 x 26</td>
<td>32 - 168</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>SWISS28</td>
<td>10.0 mm</td>
<td>8 x 11&gt;</td>
<td>&lt;27 x 28</td>
<td>32 - 168</td>
<td>Swiss, proportional</td>
</tr>
<tr>
<td>22</td>
<td>SWISS38</td>
<td>13.6 mm</td>
<td>6 x 10&gt;</td>
<td>&lt;33 x 38</td>
<td>32 - 168</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>BIGZIF57</td>
<td>20.5 mm</td>
<td>4 x 8&gt;</td>
<td>&lt;37 x 57</td>
<td>43 - 58</td>
<td>Big numbers '0'-'9' and '+,-.:'</td>
</tr>
</tbody>
</table>

EXTERNAL FONTS

Macro programming permits the inclusion of up to 10 additional fonts (7..16) and the complete redesign of the individual characters. Via text editor you are able to create and program in any font you like with a size of up to 256x240 pixels. These fonts will be stored into display-FLASH with K320COMP.EXE software.

STRING PARAMETER

Within a character string color and attribute may be changed locally. This does not affect global definition (ESC FZ). Command will start with \ (backslash) followed by a capital character from table besides. Please find all options at right table.

<table>
<thead>
<tr>
<th>CMD</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>New line</td>
</tr>
<tr>
<td>@...@</td>
<td>Text between both ´@´ will blink invers (Text color and background color)</td>
</tr>
<tr>
<td><del>...</del></td>
<td>Text between both ´~´ will blink on/off (Text color and background color)</td>
</tr>
<tr>
<td>\</td>
<td>Character ´\´ (pipe)</td>
</tr>
<tr>
<td>@</td>
<td>Character ´@´ (at)</td>
</tr>
<tr>
<td></td>
<td>Character ´\´</td>
</tr>
<tr>
<td>\</td>
<td>Character ´\´ (backslash)</td>
</tr>
<tr>
<td>W</td>
<td>Attribute- and color for text (=default)</td>
</tr>
<tr>
<td>H</td>
<td>Attribute- and color for background</td>
</tr>
<tr>
<td>L</td>
<td>Attribute 0: Clear attribute</td>
</tr>
<tr>
<td>S</td>
<td>Attribute 16: SOLID (non invertible)</td>
</tr>
<tr>
<td>V</td>
<td>Attribute 32: blink INVERS</td>
</tr>
<tr>
<td>O</td>
<td>Attribute 48: blink ON/OFF</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CMD</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Color 0: Black</td>
</tr>
<tr>
<td>01</td>
<td>Color 1: Blue</td>
</tr>
<tr>
<td>02</td>
<td>Color 2: Red</td>
</tr>
<tr>
<td>03</td>
<td>Color 3: Green</td>
</tr>
<tr>
<td>04</td>
<td>Color 4: Magenta</td>
</tr>
<tr>
<td>05</td>
<td>Color 5: Cyan</td>
</tr>
<tr>
<td>06</td>
<td>Color 6: Yellow</td>
</tr>
<tr>
<td>07</td>
<td>Color 7: White</td>
</tr>
<tr>
<td>08</td>
<td>Color 8: Dark gray</td>
</tr>
<tr>
<td>09</td>
<td>Color 9: Orange</td>
</tr>
<tr>
<td>0A</td>
<td>Color 10: Pink</td>
</tr>
<tr>
<td>0B</td>
<td>Color 11:</td>
</tr>
<tr>
<td>0C</td>
<td>Color 12:</td>
</tr>
<tr>
<td>0D</td>
<td>Color 13: Yellow/green</td>
</tr>
<tr>
<td>0E</td>
<td>Color 14: Light blue</td>
</tr>
<tr>
<td>0F</td>
<td>Color 15: Light gray</td>
</tr>
<tr>
<td>10</td>
<td>Color 64: Transparent</td>
</tr>
</tbody>
</table>
## CHARACTER SET

Below are shown the built-in character sets.

### ASCII codes for monospaced fonts 1 and 2

<table>
<thead>
<tr>
<th>Character</th>
<th>ASCII Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>!</td>
<td>33</td>
</tr>
<tr>
<td>#</td>
<td>35</td>
</tr>
<tr>
<td>$</td>
<td>36</td>
</tr>
<tr>
<td>%</td>
<td>37</td>
</tr>
<tr>
<td>&amp;</td>
<td>38</td>
</tr>
<tr>
<td>(</td>
<td>40</td>
</tr>
<tr>
<td>)</td>
<td>41</td>
</tr>
<tr>
<td>+</td>
<td>43</td>
</tr>
<tr>
<td>,</td>
<td>44</td>
</tr>
<tr>
<td>-</td>
<td>45</td>
</tr>
<tr>
<td>.</td>
<td>46</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Character</th>
<th>ASCII Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>48</td>
</tr>
<tr>
<td>1</td>
<td>49</td>
</tr>
<tr>
<td>2</td>
<td>50</td>
</tr>
<tr>
<td>3</td>
<td>51</td>
</tr>
<tr>
<td>4</td>
<td>52</td>
</tr>
<tr>
<td>5</td>
<td>53</td>
</tr>
<tr>
<td>6</td>
<td>54</td>
</tr>
<tr>
<td>7</td>
<td>55</td>
</tr>
<tr>
<td>8</td>
<td>56</td>
</tr>
<tr>
<td>9</td>
<td>57</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Character</th>
<th>ASCII Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>:</td>
<td>58</td>
</tr>
<tr>
<td>;</td>
<td>59</td>
</tr>
<tr>
<td>&lt;</td>
<td>60</td>
</tr>
<tr>
<td>=</td>
<td>61</td>
</tr>
<tr>
<td>&gt;</td>
<td>62</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Character</th>
<th>ASCII Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
<td>63</td>
</tr>
</tbody>
</table>

### ASCII codes for monospaced fonts 3 - 6

<table>
<thead>
<tr>
<th>Character</th>
<th>ASCII Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>!</td>
<td>33</td>
</tr>
<tr>
<td>#</td>
<td>35</td>
</tr>
<tr>
<td>$</td>
<td>36</td>
</tr>
<tr>
<td>%</td>
<td>37</td>
</tr>
<tr>
<td>&amp;</td>
<td>38</td>
</tr>
<tr>
<td>(</td>
<td>40</td>
</tr>
<tr>
<td>)</td>
<td>41</td>
</tr>
<tr>
<td>+</td>
<td>43</td>
</tr>
<tr>
<td>,</td>
<td>44</td>
</tr>
<tr>
<td>-</td>
<td>45</td>
</tr>
<tr>
<td>.</td>
<td>46</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Character</th>
<th>ASCII Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>48</td>
</tr>
<tr>
<td>1</td>
<td>49</td>
</tr>
<tr>
<td>2</td>
<td>50</td>
</tr>
<tr>
<td>3</td>
<td>51</td>
</tr>
<tr>
<td>4</td>
<td>52</td>
</tr>
<tr>
<td>5</td>
<td>53</td>
</tr>
<tr>
<td>6</td>
<td>54</td>
</tr>
<tr>
<td>7</td>
<td>55</td>
</tr>
<tr>
<td>8</td>
<td>56</td>
</tr>
<tr>
<td>9</td>
<td>57</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Character</th>
<th>ASCII Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>:</td>
<td>58</td>
</tr>
<tr>
<td>;</td>
<td>59</td>
</tr>
<tr>
<td>&lt;</td>
<td>60</td>
</tr>
<tr>
<td>=</td>
<td>61</td>
</tr>
<tr>
<td>&gt;</td>
<td>62</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Character</th>
<th>ASCII Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
<td>63</td>
</tr>
</tbody>
</table>

### ASCII codes for font 23 (BigZif57)

<table>
<thead>
<tr>
<th>Character</th>
<th>ASCII Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>!</td>
<td>33</td>
</tr>
<tr>
<td>#</td>
<td>35</td>
</tr>
<tr>
<td>$</td>
<td>36</td>
</tr>
<tr>
<td>%</td>
<td>37</td>
</tr>
<tr>
<td>&amp;</td>
<td>38</td>
</tr>
<tr>
<td>(</td>
<td>40</td>
</tr>
<tr>
<td>)</td>
<td>41</td>
</tr>
<tr>
<td>+</td>
<td>43</td>
</tr>
<tr>
<td>,</td>
<td>44</td>
</tr>
<tr>
<td>-</td>
<td>45</td>
</tr>
<tr>
<td>.</td>
<td>46</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Character</th>
<th>ASCII Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>48</td>
</tr>
<tr>
<td>1</td>
<td>49</td>
</tr>
<tr>
<td>2</td>
<td>50</td>
</tr>
<tr>
<td>3</td>
<td>51</td>
</tr>
<tr>
<td>4</td>
<td>52</td>
</tr>
<tr>
<td>5</td>
<td>53</td>
</tr>
<tr>
<td>6</td>
<td>54</td>
</tr>
<tr>
<td>7</td>
<td>55</td>
</tr>
<tr>
<td>8</td>
<td>56</td>
</tr>
<tr>
<td>9</td>
<td>57</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Character</th>
<th>ASCII Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>:</td>
<td>58</td>
</tr>
<tr>
<td>;</td>
<td>59</td>
</tr>
<tr>
<td>&lt;</td>
<td>60</td>
</tr>
<tr>
<td>=</td>
<td>61</td>
</tr>
<tr>
<td>&gt;</td>
<td>62</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Character</th>
<th>ASCII Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
<td>63</td>
</tr>
</tbody>
</table>

### ASCII codes for proportional fonts 17 - 22

<table>
<thead>
<tr>
<th>Character</th>
<th>ASCII Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>!</td>
<td>33</td>
</tr>
<tr>
<td>#</td>
<td>35</td>
</tr>
<tr>
<td>$</td>
<td>36</td>
</tr>
<tr>
<td>%</td>
<td>37</td>
</tr>
<tr>
<td>&amp;</td>
<td>38</td>
</tr>
<tr>
<td>(</td>
<td>40</td>
</tr>
<tr>
<td>)</td>
<td>41</td>
</tr>
<tr>
<td>+</td>
<td>43</td>
</tr>
<tr>
<td>,</td>
<td>44</td>
</tr>
<tr>
<td>-</td>
<td>45</td>
</tr>
<tr>
<td>.</td>
<td>46</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Character</th>
<th>ASCII Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>48</td>
</tr>
<tr>
<td>1</td>
<td>49</td>
</tr>
<tr>
<td>2</td>
<td>50</td>
</tr>
<tr>
<td>3</td>
<td>51</td>
</tr>
<tr>
<td>4</td>
<td>52</td>
</tr>
<tr>
<td>5</td>
<td>53</td>
</tr>
<tr>
<td>6</td>
<td>54</td>
</tr>
<tr>
<td>7</td>
<td>55</td>
</tr>
<tr>
<td>8</td>
<td>56</td>
</tr>
<tr>
<td>9</td>
<td>57</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Character</th>
<th>ASCII Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>:</td>
<td>58</td>
</tr>
<tr>
<td>;</td>
<td>59</td>
</tr>
<tr>
<td>&lt;</td>
<td>60</td>
</tr>
<tr>
<td>=</td>
<td>61</td>
</tr>
<tr>
<td>&gt;</td>
<td>62</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Character</th>
<th>ASCII Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
<td>63</td>
</tr>
</tbody>
</table>

### TYPING EXAMPLE

This hardcopy shows all 13 built-in fonts

- Schriftprobe mit Font 16x8
- Schriftprobe mit Font 17x12
- Schriftprobe mit Font 8x8
- Schriftprobe mit Font 8x16
- Schriftprobe mit Geneva 12
- Schrift Times 20
- Schrift Times 26
- Schrift Swiss 28
- Schrift Swiss 38
COLORS
EA KIT320F-8 is able to work with 16 colors (0..15) used for text and graphic output functions. Color no. 64 = transparent is special and can be used for background of character, e.g. that means that for placing a character no rectangular field will be deleted around the character itself.

The senseless combination of transparent background and transparent foreground is used to invert all dots (=complementary). Two times inverted will end same as action was started (original drawing is restored).

COLOR ATTRIBUTES
There are 3 different attributes for all colors. Just add attribute value to color value. Please note that only one attribute can be added to color value!

<table>
<thead>
<tr>
<th>Value</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>+0</td>
<td>no attribute</td>
<td></td>
</tr>
<tr>
<td>+16</td>
<td>SOLID</td>
<td>Color will never be inverted</td>
</tr>
<tr>
<td>+32</td>
<td>BLINKINVERS</td>
<td>Color will blink invers. Color for invers will be set by command 'ESC QE'</td>
</tr>
<tr>
<td>+48</td>
<td>BLINKONOFF</td>
<td>color will blink ON/OFF. Color for OFF will be set by command 'ESC QF'</td>
</tr>
</tbody>
</table>

Please find below some examples to write the letter "A" onto a fixed background:

PATTERN
A pattern type (pat = 1..16) can be set as a parameter with some commands. In this way, rectangular areas, bar graphs and even texts can be linked to different patterns and displayed.

Some more own pattern can be defined via text editor and flashed by KITcomp.exe
THIS PAGE IS INTENTIONALLY LEFT BLANK
PARAMETERS

The graphics kit can be programmed by means of various integrated commands. Each command begins with ESC followed by one or two command letters and then some parameters. There are two ways to encode parameters:

1. ASCII Mode
   - Escape code is ‘#’ (hex: $23, dez: 35).
   - Command letter will follow directly.
   - Parameter will be expected as ASCII codes with separation like ‘,’
   - Strings (text) will be transmitted without any question mark “ but terminated by CR (hex: $0D) or LF (hex: $0A).

2. Binary Mode
   - Escape code is ESC (hex: $1B, dez: 27).
   - Command letter will follow directly.
   - Coordinates xx and yy will follow binary (16 bit, starting with LOW byte and then HIGH byte.
   - All other parameter are binary 8 bit.
   - Strings (text) will be put between two question marks “ and terminated by CR (hex: $0D) or LF (hex: $0A) or NUL (hex:$00)

In binary mode there must be no separation codes like ‘,’ ‘ ’ or ‘;’. Commands do not need any terminating byte like Carrige Return (except “place a string”: $00).

PROGRAMMING EXAMPLE

The following table shows an example in which the string "Test" is output left justified at coordinate 117 / 32.

<table>
<thead>
<tr>
<th>Example</th>
<th>Codes can be output in ASCII mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>for terminal.exe</td>
<td>#ZL117, 32,Test &lt;Return&gt;</td>
</tr>
<tr>
<td>for Turbo-Pascal</td>
<td>write(aux, '#ZL117, 32,Test', chr(13));</td>
</tr>
<tr>
<td>for ‘C’</td>
<td>fprintf(stdaux, &quot;#ZL%d,%d,%s\0D&quot;, 117, 32, &quot;Test&quot;);</td>
</tr>
<tr>
<td>for Q-Basic</td>
<td>OPEN &quot;COM:9600,N,8,1,BIN&quot; FOR RANDOM AS #1 PRINT #1,&quot;#ZL117,32,Test&quot;+CHR$(13)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Example</th>
<th>Codes can be output in binary mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>in ASCII</td>
<td>ESC</td>
</tr>
<tr>
<td>in Hex</td>
<td>$1B</td>
</tr>
<tr>
<td>in Decimal</td>
<td>27</td>
</tr>
<tr>
<td>for Turbo-Pascal</td>
<td>write(aux, chr(27), 'Z', 'L', chr(117), chr(0), chr(32), chr(0), 'Test', chr(0));</td>
</tr>
<tr>
<td>for ‘C’</td>
<td>fprintf(stdaux, &quot;x1BZL%c%c%c%c%s\x00&quot;, 117, 0, 32, 0, &quot;Test&quot;);</td>
</tr>
<tr>
<td>for Q-Basic</td>
<td>OPEN &quot;COM:9600,N,8,1,BIN&quot; FOR RANDOM AS #1 PRINT #1,CHR$(27)+&quot;ZL&quot;+CHR$(117)+CHR$(0)+CHR$(32)+CHR$(0)+&quot;Test&quot;+CHR$(0)</td>
</tr>
</tbody>
</table>
**TERMINAL MODE**

After power-on a cursor flashes at the top left corner and display is ready to receive some data. All ASCII codes will be displayed with terminal function (exception: CR, LF, FF, ESC, '#'). Linefeed will be done automatically or by command 'LF' ($0A). If terminal window is full, text will scroll. Code 'FF' (Formfeed, $0C) clears window and places cursor to the top left.

Size of window can be set by command 'ESC TW'.

**Attention:** Graphic commands are able to draw inside terminal window. For example 'ESC DL' will delete terminal window, too.

Character '#' will be used as Escape code and cannot be displayed direct in terminal mode. To show this code, just send it double like '##'.

### EA KIT320F-8: Command table for terminal mode

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Commands for terminal mode</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FF: Form feed (dec:12)</td>
<td>^L</td>
<td>The contents of the terminal area are deleted and the cursor is placed at pos. (1,1)</td>
</tr>
<tr>
<td>CR: carriage return (d:13)</td>
<td>^M</td>
<td>Cursor to the beginning of the line on the extreme left</td>
</tr>
<tr>
<td>LF: line feed (dec:10)</td>
<td>^J</td>
<td>Cursor is set to the next line</td>
</tr>
<tr>
<td>Position cursor</td>
<td>ESC</td>
<td></td>
</tr>
<tr>
<td>P n1 n2</td>
<td></td>
<td>n1 = column; n2 = line; origin upper-left corner (1,1)</td>
</tr>
<tr>
<td>Save cursor position</td>
<td>S</td>
<td>saves current cursor position</td>
</tr>
<tr>
<td>Restore cursor position</td>
<td>R</td>
<td>places cursor to saved position</td>
</tr>
<tr>
<td>Cursor on/off</td>
<td>ESC</td>
<td></td>
</tr>
<tr>
<td>C n1</td>
<td></td>
<td>n1 = 0: Cursor is invisible; n1 = 1: Cursor flashes:</td>
</tr>
<tr>
<td>Terminal mode</td>
<td>ESC</td>
<td></td>
</tr>
<tr>
<td>M n1</td>
<td></td>
<td>n1 = 0: Clear mode; n1 = 1: Overwrite mode; n1 = 2: Scroll mode</td>
</tr>
<tr>
<td>Autom. line feed</td>
<td>ESC</td>
<td></td>
</tr>
<tr>
<td>Z n1</td>
<td></td>
<td>The automatic line feed is switched on (n1 = 1) or off (n1 = 0)</td>
</tr>
<tr>
<td>Terminal invisible</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td>Terminal display is visible again;</td>
</tr>
<tr>
<td>Terminal visible</td>
<td>E</td>
<td></td>
</tr>
<tr>
<td>Suppress terminal</td>
<td>ESC</td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td>ASCII characters, FF, CR, LF are suppressed. Commands (ESC T) are executed</td>
</tr>
<tr>
<td>Terminal output internal</td>
<td>ESC</td>
<td></td>
</tr>
<tr>
<td>I</td>
<td></td>
<td>All terminal outputs/commands affect the internal terminal of the EA KIT320F-8</td>
</tr>
<tr>
<td>Terminal output external</td>
<td>ESC</td>
<td></td>
</tr>
<tr>
<td>X</td>
<td></td>
<td>All terminal outputs/commands affect the external dot-matrix module</td>
</tr>
</tbody>
</table>

### Redirect terminal outputs

- **Supress terminal**
  - **Terminal output internal:**
  - **Terminal output external:**

### Settings for the internal terminal

- **Set terminal color**
  - **Set font**
  - **Add. line spacing**
  - **Define window**

### Settings for the external dot-matrix module (optionally to J6 or J7)

- **Initialize dot-matrix module**
# EA KIT320F-8 COLOR

## All Commands at a Glance

The following command tables will give an overview of all built-in functions of EA KIT320F-8. After power-on or reset, some functions are set to a particular value (see last column 'After reset' in table). Please note that all the settings can be overwritten by creating a power-on macro.

### EA KIT320F-8: Command table 1

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display commands (effect on the entire display)</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>Set display color</td>
<td>ESC</td>
<td>D fg bg</td>
</tr>
<tr>
<td>Delete display</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fill display</td>
<td>L</td>
<td>yx1 yy1 xx2 yy2</td>
</tr>
<tr>
<td>Fill display with color</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>Fill invert display</td>
<td>F</td>
<td>co</td>
</tr>
<tr>
<td>Switch display off</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>Switch display on</td>
<td>E</td>
<td></td>
</tr>
<tr>
<td>Display update</td>
<td>U</td>
<td>n1</td>
</tr>
</tbody>
</table>

### Rechteckige Bereiche verändern / zeichnen

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete area</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fill area</td>
<td>L</td>
<td>x1 x1 y1 x2 y2</td>
</tr>
<tr>
<td>Fill area with color</td>
<td>S</td>
<td>x1 x1 y1 x2 y2</td>
</tr>
<tr>
<td>Fill invert area</td>
<td>F</td>
<td>x1 x1 y1 x2 y2 co</td>
</tr>
<tr>
<td>Area with fill pattern</td>
<td>M</td>
<td>x1 x1 y1 x2 y2 n1</td>
</tr>
<tr>
<td>Draw box</td>
<td>R</td>
<td>O x1 y1 x2 y2 n1</td>
</tr>
<tr>
<td>Set color for frame</td>
<td>F</td>
<td>R r1 r2 r3 r4</td>
</tr>
<tr>
<td>Draw frame box</td>
<td>R</td>
<td>T x1 x1 y1 x2 y2 n1</td>
</tr>
</tbody>
</table>

### Straight lines and points

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set text color</td>
<td>ESC</td>
<td>F Z fg bg</td>
</tr>
<tr>
<td>Output string L: left justified, C: centered R: right justified</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>Text output angle</td>
<td></td>
<td>7,0,7</td>
</tr>
<tr>
<td>Text zoom factor</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Add. line spacing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Text pattern</td>
<td></td>
<td></td>
</tr>
<tr>
<td>String for terminal</td>
<td>ESC</td>
<td>Z T text...</td>
</tr>
</tbody>
</table>

### Bitmap image commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load image</td>
<td>ESC</td>
<td>L x1 y1 data...</td>
</tr>
<tr>
<td>Load internal image</td>
<td>I</td>
<td>x1 y1</td>
</tr>
<tr>
<td>Send hard copy</td>
<td>H</td>
<td></td>
</tr>
</tbody>
</table>

### Commands for monochrome bitmaps

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set bitmap colors</td>
<td>ESC</td>
<td>F U fg bg</td>
</tr>
<tr>
<td>Image zoom factor</td>
<td>ESC</td>
<td>U Z n1 n2</td>
</tr>
<tr>
<td>Image angle</td>
<td>W</td>
<td>n1</td>
</tr>
<tr>
<td>Image pattern</td>
<td>M</td>
<td>n1</td>
</tr>
</tbody>
</table>

### Commands for colored bitmaps (16 colors)

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attrib for colored bitmap</td>
<td>ESC</td>
<td>A n1</td>
</tr>
<tr>
<td>Set color palette</td>
<td>ESC</td>
<td>P n1</td>
</tr>
<tr>
<td>Transparency for bitmap</td>
<td>T</td>
<td>n1</td>
</tr>
</tbody>
</table>
## EA KIT320F-8: Command table 2

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bar graph commands</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set color for bargraph</td>
<td>ESC F B fg bg fc</td>
<td>colors: fg = foreground; bg = background; fc = color for frame</td>
</tr>
<tr>
<td>Define bar graph</td>
<td>R L O n1 xx1 yy1 xx2 yy2 sv ev Typ. pat</td>
<td>Define bar graph to L(left), R(right), O(oven), U(untlet) (down) with the &quot;n&quot; n1. xx1,yy1,xx2,yy2 form the rectangle enclosing the bar graph. sv, ev are the values for 0% and 100%. Type=0: bar; type=1: bar in rectangle; pat=bar pattern type=2: line; type=3: line in rectangle; pat= line width</td>
</tr>
<tr>
<td>Update bar graph</td>
<td>ESC B A n0 valu</td>
<td>Set and draw the bar graph with the number no to the new user &quot;value.&quot;</td>
</tr>
<tr>
<td>Draw new bar graph</td>
<td>Z no valu</td>
<td>Draw the bar graph with the number no completely</td>
</tr>
<tr>
<td>Send bar graph value</td>
<td>S no valu</td>
<td>Send the current value of bar graph no on the serial interface</td>
</tr>
<tr>
<td>Delete bargraph</td>
<td>D n1 n2</td>
<td>Undefine bargraph no. n1. If bargraph was defined as an analogue ouch input, even this touch field will be cleared</td>
</tr>
<tr>
<td><strong>Clipboard commands (buffer for image areas)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Select clipboard no.</td>
<td>ESC C N no</td>
<td>2 clipboards are available, the current clipboard is selected with no= (1,2).</td>
</tr>
<tr>
<td>Save display contents</td>
<td>B</td>
<td>The entire contents of the display are copied to the clipboard as an image area</td>
</tr>
<tr>
<td>Save area</td>
<td>S xx1 yy1 xx2 yy2</td>
<td>The image area from xx1, yy1 to xx2, yy2 is copied to the clipboard</td>
</tr>
<tr>
<td>Restore area</td>
<td>R xx1 yy1</td>
<td>The image on the clipboard is copied back its original position in the display</td>
</tr>
<tr>
<td>Copy area</td>
<td>K xx1 yy1</td>
<td>The image area on the clipboard is copied to xx1, yy1 in the display</td>
</tr>
<tr>
<td>Restore color palette</td>
<td>P n1</td>
<td>n1=0: palette will not be restored; n1=1: palette will always be restored; n1=2: palette will be restored for full bitmap (320x240 dots) only</td>
</tr>
<tr>
<td><strong>Flashing area commands / attributes</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flashing time</td>
<td>ESC Q Z n1</td>
<td>Set the flashing time n1=1..15 in 1/10s; 0=deactivate flashing function</td>
</tr>
<tr>
<td>Blink attribute On / Off</td>
<td>ESC Q O xx1 yy1 xx2 yy2</td>
<td>Set blink area (on/off) for xx1,yy1 to xx2,yy2 (Attribute BLINKOFF)</td>
</tr>
<tr>
<td>Blink color for Off</td>
<td>ESC Q F bg</td>
<td>Set background color for on/off blinking</td>
</tr>
<tr>
<td>Blink attribute inverting</td>
<td>ESC Q I xx1 yy1 xx2 yy2</td>
<td>Set blink area (invers) for xx1,yy1 to xx2,yy2 (Attribute BLINKINVERS)</td>
</tr>
<tr>
<td>Blink color for inverting</td>
<td>ESC Q E c1 c2</td>
<td>Color c1=0..15 will be defined as inverting color c2=0..15</td>
</tr>
<tr>
<td>Attribute solid</td>
<td>ESC Q S xx1 yy1 xx2 yy2</td>
<td>Set fixed area xx1, yy1 to xx2, yy2 which will not be inverted (Attribute SOLID)</td>
</tr>
<tr>
<td>Clear attributes</td>
<td>ESC Q L xx1 yy1 xx2 yy2</td>
<td>Clears all attributes BLINKONOFF, BLINKINVERS and SOLID for xx1,yy1 to xx2,yy2 (Color numbers from f1 to f2 = (0..15) within palette n1=0..3 will rotate with n2/10sec. speed)</td>
</tr>
<tr>
<td>Rotate palette</td>
<td>ESC Q R n1</td>
<td>Color numbers from f1 to f2 = (0..15) within palette n1=0..3 will rotate with n2/10sec. speed)</td>
</tr>
<tr>
<td><strong>Menu/pop-up commands</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set colors for menu</td>
<td>ESC F N fg bg fc</td>
<td>fg = foreground; bg = background; fc = color for frame</td>
</tr>
<tr>
<td>Set menu font</td>
<td>ESC F F no</td>
<td>Set font with the number no (1..23) for menu display</td>
</tr>
<tr>
<td>Menu font zoom factor</td>
<td>ESC N Z n1 n2</td>
<td>n1 = X zoom factor (1x..8x); n2 = Y zoom factor (1x..8x)</td>
</tr>
<tr>
<td>Add. line spacing</td>
<td>ESC N Y n1</td>
<td>Insert n1 pixels between two menu items as additional line spacing</td>
</tr>
<tr>
<td>Menu angle</td>
<td>ESC N W n1</td>
<td>Menu display angle: n1=0: 0°; n1=1: 90°; n1=2: 180°; n1=3: 270°;</td>
</tr>
<tr>
<td>Automatic function for touch menu</td>
<td>ESC Q T n1</td>
<td>n1=1: touch menu will open automatically; n1=0: touch menu will not open automatically; ESC T '0' will be sent via RS-232/422 to host; after that touch menu can be opened with command 'ESC N T 2' but not ESC T 0</td>
</tr>
<tr>
<td><strong>Menu/pop-up commands (not for touch-controlled menu)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Define menu and display</td>
<td>ESC N D xx1 yy1 no text NUL</td>
<td>A menu is drawn as of the corner xx1,yy1 with the current menu font. no: currently inverted entry (e.g.: 1 = 1st entry) Text=string with menu items. The different items are separated by the character &quot;</td>
</tr>
<tr>
<td>Next item</td>
<td>ESC N N</td>
<td>The next item is inverted or remains at the end</td>
</tr>
<tr>
<td>Previous item</td>
<td>ESC P</td>
<td>The previous item is inverted or remains at the beginning</td>
</tr>
<tr>
<td>End of menu/send</td>
<td>ESC S</td>
<td>The menu is removed from the display and replaced with the original background. The current item is sent as a number (1..n) (0=no menu displayed)</td>
</tr>
<tr>
<td>End of menu/macro</td>
<td>ESC M n1</td>
<td>The menu is removed from the display and replaced with the original background. Menu macro n1 is called for item 1, menu macro n+1 for entry 2, and so on</td>
</tr>
<tr>
<td>End of menu/cancel</td>
<td>ESC A</td>
<td>The menu is removed from the display and replaced with the original background</td>
</tr>
</tbody>
</table>
### EA KIT320F-8: Commands for the touch panel

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Touch: Define areas</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Define touch key (key remains depressed as long as there is contact)</td>
<td>ESC A</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>C f1 l2 dow code</td>
<td>up text code...</td>
<td>NUL</td>
</tr>
<tr>
<td></td>
<td>T xx1 yy1 xx2 yy2 dow code</td>
<td>up text code...</td>
<td>NUL</td>
</tr>
<tr>
<td></td>
<td>U xx1 yy1 no</td>
<td>dow up code code</td>
<td>text...</td>
</tr>
<tr>
<td>Define touch switch (status of the switch toggles after each contact on/off)</td>
<td>ESC A</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>G f1 l2 dow code</td>
<td>up text code...</td>
<td>NUL</td>
</tr>
<tr>
<td></td>
<td>K xx1 yy1 xx2 yy2 dow code</td>
<td>up text code...</td>
<td>NUL</td>
</tr>
<tr>
<td></td>
<td>J xx1 yy1 n1 dow up code code</td>
<td>text...</td>
<td>NUL</td>
</tr>
<tr>
<td>Define touch key with menu function</td>
<td>ESC A M xx1 yy1 xx2 yy2 dow code</td>
<td>up text code...</td>
<td>NUL</td>
</tr>
<tr>
<td></td>
<td>N1 n2</td>
<td>dow up code code</td>
<td>text...</td>
</tr>
<tr>
<td>Define drawing area</td>
<td>ESC A D xx1 yy1 xx2 yy2 dow code</td>
<td>up text code...</td>
<td>NUL</td>
</tr>
<tr>
<td>Define free touch area</td>
<td>ESC A H xx1 yy1 xx2 yy2 dow code</td>
<td>up text code...</td>
<td>NUL</td>
</tr>
<tr>
<td>Set bar graph by touch</td>
<td>ESC A B no dow up code code</td>
<td>text...</td>
<td>NUL</td>
</tr>
</tbody>
</table>

### Touch: settings

#### Set touch frame colors

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set touch frame colors</td>
<td>ESC F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>E f1 l2 f3 f4</td>
<td>f1=edges; f2=frame outside; f3=frame inside; f4=filling</td>
<td></td>
</tr>
</tbody>
</table>

#### Touch frame color

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set touch frame colors</td>
<td>ESC F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>E n1</td>
<td>The frame type for the display of touch keys/switches is set with n1</td>
<td>1</td>
</tr>
</tbody>
</table>

#### Touch key response

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set touch frame colors</td>
<td>ESC F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>I n1</td>
<td>Automatic inversion when touch key touched: n1=0=OFF, n1=1=ON;</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>S n1</td>
<td>Tone sounds briefly when a touch key is touched: n1=0=OFF, n1=1=ON;</td>
<td>1</td>
</tr>
</tbody>
</table>

#### Invert touch key

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set touch frame colors</td>
<td>ESC F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>N Cod</td>
<td>The touch key with the assigned return code is inverted manually</td>
<td></td>
</tr>
</tbody>
</table>

#### Query touch switch

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set touch frame colors</td>
<td>ESC F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>X Cod</td>
<td>The status of the switch (off=0; on=1) is sent via the serial interface.</td>
<td>1</td>
</tr>
</tbody>
</table>

#### Set touch switch

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set touch frame colors</td>
<td>ESC F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P Cod n1</td>
<td>The status of the switch is changed by means of a command n1=0=off, n1=1=on.</td>
<td>0</td>
</tr>
</tbody>
</table>

#### Define radiogroup

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set touch frame colors</td>
<td>ESC F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R n1</td>
<td>Within a group only one single switch will be active; rel of them will be deactivated n1=0: next switch definitions will keep free of all groups n1=1..255: next switch definitions will join to group no. n1</td>
<td>0</td>
</tr>
</tbody>
</table>

#### Delete touch area

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set touch frame colors</td>
<td>ESC F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>L Cod n1</td>
<td>The touch area with the return code (code=0: all touch areas) is removed from the touch query. When n1=0, the area remains visible on the display; when n1=1, the area is deleted from the display.</td>
<td>0</td>
</tr>
</tbody>
</table>

#### Send bar value on/off

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set touch frame colors</td>
<td>ESC F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Q n1</td>
<td>Automatic transmission of a new bar graph value by touch input is deactivated (n1=0) or activated (n1=1)</td>
<td>1</td>
</tr>
</tbody>
</table>

#### Touch query on/off

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set touch frame colors</td>
<td>ESC F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>A n1</td>
<td>Touch query is deactivated (n1=0) or activated (n1=1)</td>
<td>1</td>
</tr>
</tbody>
</table>

### Touch: Label font

#### Font color

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set touch frame colors</td>
<td>ESC F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>A fg bg</td>
<td>Color for touch labeling. fg=foreground; bg=background color</td>
<td></td>
</tr>
</tbody>
</table>

#### Label font

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set touch frame colors</td>
<td>ESC F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>F no</td>
<td>Set font with the number no=1..23 for touch key label</td>
<td>5</td>
</tr>
</tbody>
</table>

#### Label zoom factor

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set touch frame colors</td>
<td>ESC F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Z n1 n2</td>
<td>n1 = X zoom factor (1..8); n2 = Y zoom factor (1..8)</td>
<td>1.1</td>
</tr>
</tbody>
</table>

#### Add line spacing

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set touch frame colors</td>
<td>ESC F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Y n1</td>
<td>Insert n1 pixels between two lines of text as additional line spacing</td>
<td>0</td>
</tr>
</tbody>
</table>

#### Label angle

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set touch frame colors</td>
<td>ESC F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W n1</td>
<td>Text output angle: n1=0=0°; n1=1=90°; n1=2=180°; n1=3=270°</td>
<td>0</td>
</tr>
</tbody>
</table>
### EA KIT320F-8: Commands for Macro, Port and Misc

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Macro commands</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Run macro</td>
<td>N no</td>
<td>Call the (normal) macro with the number no (0..255) (max. 7 levels)</td>
</tr>
<tr>
<td>Run touch macros</td>
<td>T no</td>
<td>Call the touch macro with the number no (0..255) (max. 7 levels)</td>
</tr>
<tr>
<td>Run port macro</td>
<td>P no</td>
<td>Call the port macro with the number no (0..255) (max. 7 levels)</td>
</tr>
<tr>
<td>Run bit macro</td>
<td>B no</td>
<td>Call the bit macro with the number no (1..16) (max. 7 levels)</td>
</tr>
<tr>
<td>Run menu macro</td>
<td>M no</td>
<td>Call the menu macro with the number no (0..255) (max. 7 levels)</td>
</tr>
<tr>
<td>Run matrix macro</td>
<td>X no</td>
<td>Call the matrix macro with the number no (0..64) (max. 7 levels)</td>
</tr>
<tr>
<td>Run analogue macro</td>
<td>V no</td>
<td>Call the analogue macro with the number no (0..19) (max. 7 levels)</td>
</tr>
<tr>
<td>Run process macro</td>
<td>C no</td>
<td>Call the process macro with the number no (0..255) (max. 7 levels)</td>
</tr>
<tr>
<td><strong>Inhibit macro</strong></td>
<td>ESC M</td>
<td>inhibit macro execution for type=N,T,P,B,M,X,V or C (type=A alle types) will be inhibited from no. n1 to n2 (no longer executed)</td>
</tr>
<tr>
<td><strong>Enable macro</strong></td>
<td>ESC M</td>
<td>Enables macro execution for type=N,T,P,B,M,X,V or C (type=A alle types) from no. n1 to n2</td>
</tr>
<tr>
<td><strong>Automatic (normal-) macro</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Macro execution with delay</td>
<td>G no n2</td>
<td>Normal macro with no=0..255 will be executed after n2/10sec.</td>
</tr>
<tr>
<td>Run autom. macros once</td>
<td>E n1 n2 n3</td>
<td>Run all macros n1..n2 automatically one after another once; n3=pause in 1/10sec. steps.</td>
</tr>
<tr>
<td>Run autom. macro cyclical</td>
<td>A n1 n2 n3</td>
<td>Run all macros n1..n2 automatically one after another cyclically; n3=pause in 1/10sec. steps.</td>
</tr>
<tr>
<td>Run auto. macro pingpong</td>
<td>J n1 n2 n3</td>
<td>Run all macros n1..n2 automatically one after another (pingpong mode: e.g. n1,n2,n3,n4,n5,n6,n7,n8,..); n3=pause in 1/10sec. steps.</td>
</tr>
<tr>
<td><strong>Define process macro</strong></td>
<td>ESC M</td>
<td>Define process macro no (no=1..16, 1=highest priority)</td>
</tr>
<tr>
<td><strong>Set pause</strong></td>
<td>Z no zs</td>
<td>Set new pause (zs/10 sec.) for macro process no (no=1..16). zs=0 will stop execution</td>
</tr>
<tr>
<td><strong>Stop process macro</strong></td>
<td>S n1</td>
<td>Stop all process macros with n1=0 gestoppt and restart with n1=1; important for some settings and outputs via RS-232/RS-422 that may not be interrupted</td>
</tr>
<tr>
<td><strong>Port commands</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Write output port</td>
<td>W n1 n2</td>
<td>n1=0: Set all 8 output ports in accordance with n2 (8-bit binary value)</td>
</tr>
<tr>
<td>Read input port</td>
<td>R n1</td>
<td>n1=0: Read all 8 input ports as 8-bit binary value</td>
</tr>
<tr>
<td>Port scan on/off</td>
<td>A n1</td>
<td>The automatic scan of the input port is n1=0: deactived; n1=1: activated</td>
</tr>
<tr>
<td>Invert input port</td>
<td>I n1</td>
<td>The input port is n1=0: normal; n1=1: evaluated inverted</td>
</tr>
<tr>
<td>Matrix keyboard</td>
<td>M n1 n2 n3</td>
<td>Specifies an external matrix keyboard at the inputs and outputs. n1=number of inputs (255); n2=number of outputs (255); n3=deboooning (0..7)</td>
</tr>
<tr>
<td>Illumination on/off/half</td>
<td>L n1</td>
<td>CFL/LED illumination n1=0: OFF; n1=1: ON; n1=2: half brightness;</td>
</tr>
<tr>
<td>BEEP on/off</td>
<td>S n1</td>
<td>n1=1..255: Tone on for n1 1/10s</td>
</tr>
<tr>
<td><strong>Other commands</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Redefine color</td>
<td>ESC F</td>
<td>Set RGB value (r16,g16,b16=0..15) for color no. n1=0..15</td>
</tr>
<tr>
<td>Increase contrast</td>
<td>ESC P</td>
<td>Increase contrast for 1 step (more bright)</td>
</tr>
<tr>
<td>Decrease contrast</td>
<td>ESC D</td>
<td>Decrease contrast for 1 step (more dark)</td>
</tr>
<tr>
<td>Set contrast</td>
<td>ESC S</td>
<td>Set contrast to value n1=0..40</td>
</tr>
<tr>
<td>Wait (pause)</td>
<td>ESC X</td>
<td>Wait n1 tenths of a second before the next command is executed</td>
</tr>
<tr>
<td>Send bytes</td>
<td>ESC S</td>
<td>num (=1..255) bytes are sent on the RS-232/RS-422</td>
</tr>
<tr>
<td>Send version</td>
<td>ESC V</td>
<td>The software version no. + date is sent as a string on the RS-232/RS-422</td>
</tr>
<tr>
<td>Commands to HD44780</td>
<td>ESC L</td>
<td>num (=1..255) commands are sent to the ext. dot-matrix module with HD44780.</td>
</tr>
<tr>
<td>Data to HD44780</td>
<td>ESC D</td>
<td>num (=1..255) data is sent to the ext. dot-matrix module with HD44780.</td>
</tr>
<tr>
<td>Read EEPROM</td>
<td>ESC R</td>
<td>num (=1..255) bytes are requested from the internal user EEPROM as of the address and sent via the RS-232/RS-422.</td>
</tr>
<tr>
<td>Write EEPROM</td>
<td>ESC W</td>
<td>num (=1..255) bytes are written to the internal user EEPROM as of the address addr.</td>
</tr>
<tr>
<td>Read I2C bus</td>
<td>ESC R</td>
<td>num (=1..255) bytes are requested from the block on the I2C bus with the device address addr and sent via the RS-232/RS-422.</td>
</tr>
<tr>
<td>Write I2C bus</td>
<td>ESC W</td>
<td>num (=1..255) bytes are sent on the I2C bus for the block with the device address addr.</td>
</tr>
</tbody>
</table>
COMMAND FOR ANALOGUE INPUTS

Table below shows functionality for analogue inputs AIN1 and AIN2. Range is 0..+250mV and resolution is 10 bit (like a 3 digit DVM).

<table>
<thead>
<tr>
<th>Command</th>
<th>Codes</th>
<th>Remarks</th>
<th>After reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable/disable analogue-in</td>
<td>A n1</td>
<td>n1=0 disables input scan for AIN1 and AIN2; n1=1 enable input scan</td>
<td>1</td>
</tr>
<tr>
<td>Send analog value</td>
<td>D ch</td>
<td>Voltage [mV] will be sent for channel ch=1..2 via RS-232/RS-422</td>
<td></td>
</tr>
<tr>
<td>Limit for analog macro</td>
<td>ESC V K ch n1 n2 n3</td>
<td>Sets limits for channel ch=1..2. n1 = lower limit [mV] n2 = upper limit [mV] n3 = hysteresis [mV]</td>
<td>0</td>
</tr>
<tr>
<td>Bargraph for AIN1/AIN2</td>
<td>ESC V R L O U ch n1 xx1 yy1 xx2 yy2 sv ev type pat</td>
<td>Define bargraph for channel ch=1..2 to L(eft), R(ight), O(ben) (up), U(nten) (down) with no. n1. xx1,yy1,xx2,yy2 form the rectangle enclosing the bar graph. sv, ev are the values for 0% and 100% [mV]. Type=0: bar; type=1: bar in rectangle; pat=bar pattern type=2: line; type=3: line in rectangle; pat= line width</td>
<td>no bar graph defined</td>
</tr>
<tr>
<td>Redraw bargraph</td>
<td>ESC V B ch</td>
<td>redraw all bar graphs defined for channel ch=1..2</td>
<td></td>
</tr>
<tr>
<td>Calibration</td>
<td>ESC V @ ch String NUL</td>
<td>Calibration procedure is as follows: 1.) Apply defined voltage to AIN1 (channel1, ch=1) or AIN2 (channel2, ch=2) 2.) Run this command with channel information ch=1..2 and &quot;String&quot;; &quot;String&quot; are ASCII characters like e.g. &quot;200.0&quot;</td>
<td>nicht kalibriert</td>
</tr>
<tr>
<td>Scaling</td>
<td>ESC V E ch Format String... NUL</td>
<td>Set scaling factor for channel ch=1..2. Assign 2 voltages (0..200mV) 2 numerical strings (max. 5 digits + decimal point + sign) Format String: mV1=voltage1;mV2=voltage2. 'NUL' ($00) = termination Example: display for 200 mV input should be '123.45' and '0.00' for 100mV Format String: &quot;200=123.45:100=0&quot;</td>
<td>0; 200=200.0</td>
</tr>
<tr>
<td>Send string</td>
<td>ESC V S ch</td>
<td>This will send current voltage as formatted string for channel ch=1..2 via RS-232/RS-422</td>
<td></td>
</tr>
<tr>
<td>Display on terminal</td>
<td>ESC V T ch</td>
<td>Show formatted string of channel ch=1..2 on terminal window</td>
<td></td>
</tr>
<tr>
<td>Display on graphic layer</td>
<td>ESC V G ch xx1 yy1</td>
<td>Show formatted string of channel ch=1..2 at coordinate xx1,yy1</td>
<td></td>
</tr>
<tr>
<td>String color</td>
<td>ESC F V ch fg bg</td>
<td>Set color for string output of channel ch=1..2; fg= foreground, bg= background color</td>
<td>7 / 0</td>
</tr>
<tr>
<td>Font</td>
<td>ESC F F ch n1</td>
<td>Set font n1 for channel ch=1..2</td>
<td>5</td>
</tr>
<tr>
<td>Foorm factor</td>
<td>ESC V Z ch n1 n2</td>
<td>Set zoom factor for channel ch=1..2. n1 = X zoom factor (1x..8x); n2 = Y zoom factor (1x..8x)</td>
<td>1,1</td>
</tr>
<tr>
<td>Text pattern</td>
<td>ESC V M ch pat</td>
<td>Set fill-pattern for string of channel ch=1..2; pat= pattern no.; 0 = no pattern</td>
<td>0</td>
</tr>
<tr>
<td>Text angle</td>
<td>ESC V W ch n1</td>
<td>Set writing angle for channel ch=1..2; n1=0: 0°; n1=1: 90°; n1=2: 180°; n1=3: 270°;</td>
<td>0</td>
</tr>
</tbody>
</table>
RESPONSE OF THE EA KIT320F-8 VIA SERIAL INTERFACE

The table below contains all response codes of EA KIT320F-8. Some response data will come automatically some others on request. In addition to that with command 'ESC SB ...' user is able to transmit individual data packages via RS-232/RS-422.

<table>
<thead>
<tr>
<th>Id</th>
<th>num</th>
<th>data</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>Automatic response from the KIT320</strong></td>
</tr>
<tr>
<td>ESC</td>
<td>A</td>
<td>1</td>
<td>code</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Response from the analog touch panel when a key/switch is pressed. code = down or up code of the key/switch. Only transmitted if no corresponding touch macro is defined !</td>
</tr>
<tr>
<td>ESC</td>
<td>N</td>
<td>1</td>
<td>code</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>After a menu item is selected by touch, the selected menu item code is transmitted. Only transmitted if no corresponding touch macro is defined !</td>
</tr>
<tr>
<td>ESC</td>
<td>P</td>
<td>1</td>
<td>value</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>After the input port is changed, the new 8-bit value is transmitted. The automatic port scan must be activated. See the 'ESC Y A n1' command. It is only transmitted when there is no corresponding port macro defined !</td>
</tr>
<tr>
<td>ESC</td>
<td>M</td>
<td>1</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>When a keystroke of the external matrix keyboard is detected, the newly pressed key number nr is transmitted. Only transmitted if no corresponding matrix macro is defined !</td>
</tr>
<tr>
<td>ESC</td>
<td>B</td>
<td>2</td>
<td>no value</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>When a bar graph is set by touch, the current value of the bar is transmitted with no Transmission of the bar value must be activated (see the 'ESC A O n1' command).</td>
</tr>
<tr>
<td>ESC</td>
<td>T</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>After a menu item is selected by touch, the selected menu item code is transmitted. Only transmitted if no corresponding touch macro is defined !</td>
</tr>
<tr>
<td>ESC</td>
<td>H</td>
<td>5</td>
<td>Typ. xLO xHI yLO yHI</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>The following is transmitted in the case of a free touch area event: type=0 release; type=1 is touch; type=2 is drag within the free touch area at the x,y coordinates (16-bit values).</td>
</tr>
</tbody>
</table>

**Response only when requested by command**

<table>
<thead>
<tr>
<th>Id</th>
<th>num</th>
<th>data</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESC</td>
<td>N</td>
<td>1</td>
<td>no</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>After the ‘ESC N S’ command, the currently selected menu item is transmitted. no=0: no menu item is selected.</td>
</tr>
<tr>
<td>ESC</td>
<td>B</td>
<td>2</td>
<td>no value</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>After the ‘ESC B S n1’ command, the current value of the bar is transmitted with no.</td>
</tr>
<tr>
<td>ESC</td>
<td>X</td>
<td>2</td>
<td>code value</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>After the ‘ESC A X command, the current status of the touch switch is transmitted with code (the return code): value = 0 or 1</td>
</tr>
<tr>
<td>ESC</td>
<td>Y</td>
<td>2</td>
<td>no value</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>After the ‘ESC Y command, the requested input port is transmitted. no=0: value is an 8-bit binary value of all 8 inputs. no=1..8: value is 0 or 1 depending on the status of the input no.</td>
</tr>
<tr>
<td>ESC</td>
<td>D</td>
<td>3</td>
<td>ch LO-byt Hi-byt</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>After the ‘ESC S D ch’ command, the requested voltage of channel ch=1..2 will be sent (value = 0..2500 means 0..250mV)</td>
</tr>
<tr>
<td>ESC</td>
<td>V</td>
<td>cnt</td>
<td>ch scaled ASCII string ...</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>After the ‘ESC V S ch’ command, the requested voltage of channel ch=1..2 will be sent as scaled ASCII characters (length of string = cnt-1).</td>
</tr>
<tr>
<td>ESC</td>
<td>E</td>
<td>num</td>
<td>data ...</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>After the ‘ESC E R addr num’ command, the requested bytes are transmitted from the user FLASH-PROM.</td>
</tr>
<tr>
<td>ESC</td>
<td>I</td>
<td>num</td>
<td>data ...</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>After the ‘ESC I R addr num’ command, the requested bytes are transmitted from the I2C bus.</td>
</tr>
</tbody>
</table>

**Response without ESC and length specification (num)**

<table>
<thead>
<tr>
<th>Id</th>
<th>num</th>
<th>data</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>M</td>
<td>77,876 bytes of image data</td>
<td>After the ‘ESC U H’ command, 77,876 bytes of image data are transmitted (=320x240 BMP image with 256 colors). The first two bytes of the BMP image always begin with ‘BM’</td>
</tr>
<tr>
<td>E</td>
<td>A</td>
<td>String ..</td>
<td>NUL</td>
</tr>
</tbody>
</table>
DIMENSION

all dimensions are in mm

MOUNTING BEZEL EA 0FP320F-8SW

(Also good for blue/white version EA 320F-8CTP)

Panel Cut Out

All dimensions are in mm