

EINBAU BEDIENEINHEIT

240x128 MIT TOUCH PANEL

Stand 06.2012



Abmessung:
113x70x12mm

TECHNISCHE DATEN

- * LCD-GRAFIKDISPLAY MIT DIVERSEN GRAFIKFUNKTIONEN
- * 8 EINGEBAUTE FONTS
- * FONT ZOOM VON ca. 2mm BIS ZU ca. 50mm, auch um 90° GEDREHT
- * 3 VERSCHIEDENE INTERFACE ONBOARD: RS-232, I²C-BUS ODER SPI-BUS
- * 240x128 PIXEL MIT LED-BELEUCHTUNG BLAU NEGATIV ODER SCHWARZ-WEISS POSITIV, FSTN-TECHNIK, AUCH IN AMBER
- * VERSORGUNG +5V@ typ. 75mA / 210mA (OHNE / MIT LED BELEUCHTUNG)
- * **PIXELGENAUE** POSITIONIERUNG BEI ALLEN FUNKTIONEN
- * GERADE, PUNKT, BEREICH, UND/ODER/EXOR, BARGRAPH...
- * CLIPBOARD FUNKTIONEN, PULL-DOWN MENÜS
- * BIS ZU 256 BILDER INTERN SPEICHERBAR
- * BIS ZU 256 MAKROS PROGRAMMIERBAR (32kB EEPROM ONBOARD)
- * TEXT UND GRAFIK MISCHEN, BLINKATTRIBUTE: EIN/AUS/ INVERS BLINKEN
- * BELEUCHTUNG PER SOFTWARE REGELBAR
- * ANALOGES TOUCH PANEL: VARIABLES RASTER
- * FREI DEFINIERBARE TASTEN UND SCHALTER

BESTELLBEZEICHNUNG

DISPLAYS

240x128 DOTS, WEISSE LED-BELEUCHTUNG, BLAU NEGATIV WIEVOR, JEDOCH MIT TOUCH PANEL

240x128 DOTS, WEISSE LED-BELEUCHTUNG, POSITIV MODE, FSTN WIEVOR, JEDOCH MIT TOUCH PANEL

240x128 DOTS, AMBER, POSITIV MODE, FSTN WIEVOR, JEDOCH MIT TOUCH PANEL

STARTERKITS

ENTHÄLT EAeDIP240B-7LWTP, EVALUATION BOARD MIT USB FÜR DIREKTE PC-VERBINDUNG UND INTERFACE BOARDS FÜR ANBINDUNG AN DAS HOST-SYSTEM

WIEVOR, JEDOCH MIT EAeDIP240J-7LWTP

ZUBEHÖR

EINBAUBLENDE SCHWARZ, ELOXIERTES ALUMINIUM
BUCHSENLEISTE 1x20, 4.5 mm HOCH (1 STÜCK)

EA eDIP240B-7LW
EA eDIP240B-7LWTP
EA eDIP240J-7LW
EA eDIP240J-7LWTP
EA eDIP240J-7LA
EA eDIP240J-7LATP

EA EVALeDIP240B
EA EVALeDIP240J

EA 0FP241-7SW
EA B254-20

**ELECTRONIC
ASSEMBLY**
making things easy

Documentation of revision				
Date	Type	Old	New	Reason / Description
15.02.04	V1.0			Preliminary version
24.11.04	V1.1	- Modulo 8	New Command Macro-Process #MD../#MZ../#MS.. Adaptor MAX232 circuit diagramm Modulo 256	new firmware - typing error in protocol description
18.01.05	V1.2		New Command Terminal-Cursor Save/Restore #TS/#TR New Command Bargraph send continous #AQ 2	new firmware
07.04.05	V1.3		New addressable 2-wire RS485 Interface with SN75176 New 32 additional I2C Addresses New Commands #AG, #SI, #KA	new firmware
13.05.05	V1.4		Bugfix in SPI- I2C-Mode after wrong Packet (NAK)	new firmware
04.10.05	V1.5		some problems with opertating >60°C (display corrupted) New Protocoll Info Command 'DC2 1 P bcc' Bugfix in #GZ (pointsize), #B RLOU (typ2+3 linewidth)	new firmware
18.10.05	V1.6		OUT-port functionality on not used configuration pins	new firmware
17.02.06	-		Drawing for mounting panel EA 0FP241-7SW included	-
27.04.06	-	V/A 61.0mm	Revised drawing (V/A = 60.4mm and pcb Rev.D)	
29.06.07	-		Insert EA eDIP240J-7LA	

INHALT

ALLGEMEINES	3
ELEKTRISCHE SPEZIFIKATIONEN	4
AUSGÄNGE	4
RS-232	5
SPI	6
I ² C	7
SOFTWARE PROTOKOLL	8 - 9
TOUCH PANEL	10
ZEICHENSÄTZE	11-12
BEFEHLE / FUNKTIONEN INTABELLENFORM	13 - 15
RÜCKANTWORTEN DES BEDIENPANELS	16
PROGRAMMIERBEISPIEL	17
MAKROPROGRAMMIERUNG	18 - 19
ABMESSUNGEN	20

ALLGEMEINES

EA eDIP240-7 ist das weltweit erste Display mit integrierter Intelligenz! Neben diversen eingebauten Schriften welche pixelgenau verwendet werden können, bietet es zudem eine ganze Reihe ausgefeilter Grafikfunktionen.

Das Display ist mit 5V sofort betriebsbereit. Die Ansteuerung erfolgt über eine der 3 eingebauten Schnittstellen RS-232, SPI oder I²C.

Die Programmierung erfolgt über hochsprachenähnliche Grafikbefehle; die zeitraubende Programmierung von Zeichensätzen und Grafikroutinen entfällt hier völlig. Die simple Verwendung dieses Displays samt Touchpanel verkürzt die Entwicklungszeit drastisch.

HARDWARE

Das Display ist für +5V Betriebsspannung ausgelegt. Die Datenübertragung erfolgt entweder seriell asynchron im RS-232 Format oder synchron via SPI oder I²C Spezifikation. Zur Erhöhung der Datensicherheit wird für alle Übertragungsvarianten ein einfaches Protokoll verwendet.

ANALOGESTOUCHPANEL

Die Versionen EA eDIP240B-7LWTP und EA eDIP240J-7LWTP sind mit einem integrierten Touch Panel ausgerüstet. Durch Berühren des Displays können hier Eingaben gemacht und Einstellungen per Menü oder Bargraphs getätigt werden. Die Beschriftung der "Tasten" ist flexibel und auch während der Laufzeit änderbar (verschiedene Sprachen, Icons). Das Zeichnen der einzelnen "Tasten", sowie das Beschriften wird von der eingebauten Software komplett übernommen.

LED-BELEUCHTUNG, B- UND J-TYPEN

Alle Displays in blau-weiß (B) und schwarz-weiß (J) sind mit einer modernen und stromsparenden LED-Beleuchtung ausgestattet. Während das Schwarz-Weiß-Display und das amber-farbige auch mit komplett abgeschalteter Beleuchtung noch lesbar ist, benötigt das blau-weiße Display dagegen zum Ablesen in jedem Fall eine minimale Beleuchtung. Die Beleuchtung ist per Befehl abschaltbar und die Helligkeit regelbar.

Für den Betrieb im direkten Sonnenlicht empfehlen wir die Schwarz-Weiß-Versionen. Für alle anderen Einsatzfälle kann auch die kontraststarke Version in blau-weiß verwendet werden. Im 24h Betrieb sollte zur Erhöhung der Lebensdauer der weißen Beleuchtung, diese sooft als möglich gedimmt bzw. abgeschaltet werden. Dies ist für die amberfarbige Beleuchtung nicht erforderlich.

SOFTWARE

Die Programmierung dieses Displays erfolgt über Befehle wie z.B. *Zeichne ein Rechteck von (0,0) nach (64, 15)*. Es ist keine zusätzliche Software oder Treiber erforderlich. Zeichenketten lassen sich **pixelgenau** platzieren. Blinkattribute können beliebig oft vergeben werden - auch für Grafiken. Das Mischen von Text und Grafik ist jederzeit möglich. Es können bis zu 16 verschiedene Zeichensätze verwendet werden. Jeder Zeichensatz kann wiederum 2- bis 4-fach gezoomt werden. Mit dem größten Zeichensatz lassen sich somit bildschirmfüllende Worte und Zahlen darstellen.

ZUBEHÖR: PROGRAMMIER FÜR INTERNES EEPROM

Evaluationboard (EVAL-Board) zur Programmierung des internen EEPROMS

Das Display wird fertig programmiert mit allen Fonts ausgeliefert. In der Regel ist also eine Programmierung des internen EEPROMS nicht erforderlich !

Sollen jedoch die internen Zeichensätze geändert oder erweitert werden, oder sollen intern Bilder/Animationen oder Makros abgelegt werden, brennen Sie über die kostenfrei erhältlichen „ELECTRONIC ASSEMBLY LCD-Tools“ und das als Zubehör erhältliche USB-Evaluationboard EA 9777-2USB die von Ihnen erstellten Daten/Bilder dauerhaft in das on-board EEPROM(64KB).

Das EVAL-Board wird an die USB-Schnittstelle des PC angeschlossen. Ein Schnittstellenkabel und die Installationssoftware sind im Lieferumfang des Programmers enthalten.

Zusätzliche Schnittstellenadapter EA 9777-2PE (im Starter-Kit enthalten)

Als weiteres Zubehör (EA 9777-2PE) ist ein Paket mit 5 zusätzlichen Schnittstellenadaptern für das EVAL-Board erhältlich: RS-232, RS-485, SPI, I²C, RS-232 (CMOS-Pegel).

SPEZIFIKATION UND GRENZWERTE

Characteristics					
Value	Condition	min.	typ.	max.	Unit
Operating Temperature		-20		+70	°C
Storage Temperature		-30		+80	°C
Storage Humidity	< 40°C			90	%RH
Operating Voltage		4.5	5.0	5.5	V
Input Low Voltage		-0.5		0.2*VDD	V
Input High Voltage	Pin Reset only	0.9*VDD		VDD+0.5	V
Input High Voltage	except Reset	0.6*VDD		VDD+0.5	V
Input Leakage Current	Pin MOSI only			1	µA
Input Pull-up Resistor		20		50	kOhms
Output Low Voltage				0.7	V
Output High Voltage		4.0			V
Output Current				20	mA
Current	Backlight off		75		mA
	Backlight on		210		mA

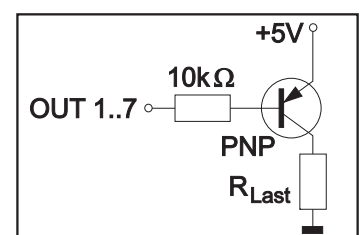
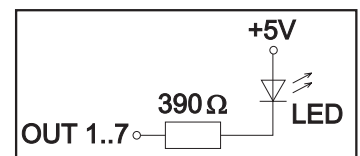
AUSGÄNGE

An das EA eDIP240 können ab Firmware V1.6 bis zu 7 Ausgänge z.B. zum Ansteuern von LEDs genutzt werden. Je nach gewähltem Interface RS232, SPI oder I2C werden dazu nicht benötigte Konfigurationspins als Ausgänge genutzt. Diese Konfigurationspins (Opendrain mit internem Pullup) werden zur Einstellung der Interfacemodi als 1=HIGH-Pegel gewertet.

Jeder Ausgang kann per Befehl 'ESC YW n1 n2' individuell angesteuert werden. Strom kann nur bei L-Pegel fließen (Opendrain mit internem Pullup). Jeder Ausgang kann max. 10mA liefern. Es ist somit möglich, mit einem Ausgang direkt eine LED zu schalten. Größere Ströme können durch Verwendung eines externen Transistors geschaltet werden.

Zuordnung Ausgang <-> Pin Nr. je nach Interface

Ausgang Nr.	RS232/RS422		SPI		I2C	
	Pin Nr.	Symbol	Pin Nr.	Symbol	Pin Nr.	Symbol
OUT1	6	BAUD0	10	DORD	6	BA0
OUT2	7	BAUD1	12	OUT2	7	BA1
OUT3	8	BAUD2	13	DPOM	8	SA0
OUT4	9	ADR0	14	CPOL	9	SA1
OUT5	13	DPOM	15	CPHA	10	SA2
OUT6	14	ADR1			11	BA2
OUT7	15	ADR2			13	DPOM



RS-232 INTERFACE

Wird das Display wie unten gezeigt beschaltet, so ist das RS-232 Interface ausgewählt. Die Pinbelegung ist in der Tabelle rechts angegeben.

Die Leitungen RxD und TxD führen 5V CMOS-Pegel zur direkten Anbindung an z.B. einen Mikrokontroller.

Wenn "echte" RS-232 Pegel erwünscht sind (z.B. zur Anbindung an einen PC) ist ein externer Pegelwandler wie z.B. MAX232 erforderlich.

Hinweis:

Die Pins BAUD0..2, ADR0..2, DPOM und TEST/SBUF haben einen internen Pull-UP, deshalb ist nur der LO-Pegel (0=GND) aktiv anzulegen. Für Hi-Pegel sind diese Pins offen zu lassen.

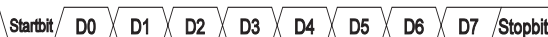
Für RS232 Betrieb (ohne Adressierung) sind die Pins ADR0..ADR2 offen zu lassen.

Am Pin 20 (SBUF) zeigt das Display mit einem low-Pegel, dass im internen Sendepuffer Daten zur Abholung bereit stehen. Diese Leitung kann z.B. mit einem Interrupteingang des Host Systems verbunden werden.

Pinout eDIP240-7						
RS-232 / RS-422 mode						
Pin	Symbol	In/Out	Function	Pin	Symbol	Function
1	GND	-	Ground Potential for logic (0V)	21	N.C.	not connected
2	VDD	-	Power supply for logic (+5V)	22	N.C.	not connected
3	VADJ	In	Operating voltage for LC driving (input)	23	N.C.	not connected
4	VOUT	Out	Output voltage for LC driving	24	N.C.	not connected
5	RESET	-	L: Reset	25	N.C.	not connected
6	BAUD0	In	Baud Rate 0	26	N.C.	not connected
7	BAUD1	In	Baud Rate 1	27	N.C.	not connected
8	BAUD2	In	Baud Rate 2	28	N.C.	not connected
9	ADR0	In	Address 0 for RS-485 (V1.3 or later)	29	N.C.	not connected
10	RxD	In	Receive Data	30	N.C.	not connected
11	TxD	Out	Transmit Data	31	N.C.	not connected
12	EN485	Out	Transmit Enable for RS-485 driver	32	N.C.	not connected
13	DPOM	In	L: disable Power-On-Macro do not connect for normal operation	33	N.C.	not connected
14	ADR1	In	Address 1 for RS-485 (V1.3 or later)	34	N.C.	not connected
15	ADR2	In	Address 2 for RS-485 (V1.3 or later)	35	N.C.	not connected
16	BUZZ	Out	Buzzer output	36	N.C.	not connected
17	EED_SDA	Bidir.	Serial Data Line for int. EEPROM	37	N.C.	not connected
18	EED_SCL	Out	Serial Clock Line for int. EEPROM	38	N.C.	not connected
19	EED_WP	In	H: Write Protect for int. EEPROM	39	N.C.	not connected
20	TEST SBUF	IN Out	open-drain with internal pullup 20..50k IN (Power-On) L: Testmode OUT L: data in sendbuffer	40	N.C.	not connected

BAUDRATEN

Die Baudrate wird über die Pins 6, 7 und 8 (Baud0..2).eingestellt. Das Datenformat ist fest eingestellt auf 8 Datenbits, 1 Stopbit, keine Parität. Handshakeleitungen RTS/CTS sind nicht erforderlich. Die notwendige Steuerung wird von dem eingebauten Software-Protokoll übernommen (siehe Seiten 8 und 9).



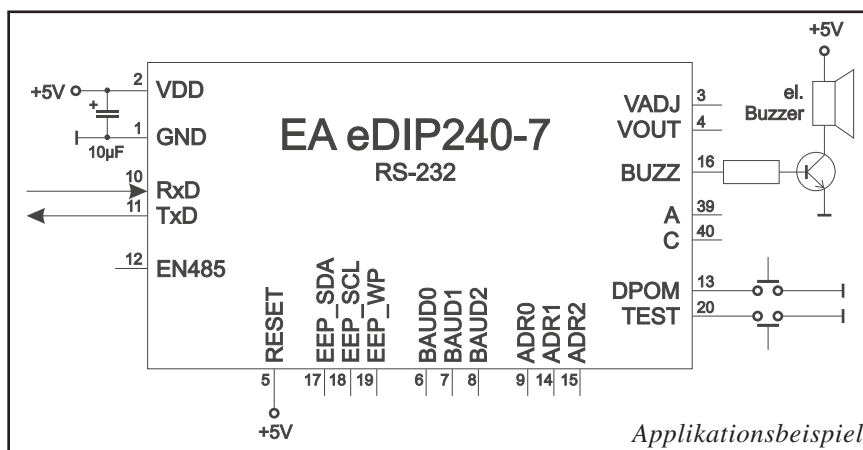
Baudraten			
Baud0	Baud1	Baud2	Datenformat 8,N,1
0	0	0	1200
1	0	0	2400
0	1	0	4800
1	1	0	9600
0	0	1	19200
1	0	1	38400
0	1	1	57600
1	1	1	115200

RS-485 INTERFACE

Mit einem externen Umsetzer z.B. SN75176 kann das EA eDIP240 an einen 2-Draht RS-485 Bus angeschlossen werden. Somit können grosse Entfernungen bis zu 1200m (Ferndisplay) realisiert werden. Betrieb von mehreren EA eDIP240 an einem RS-485 Bus durch Einstellen von Adressen.

Adressierung:

- Bis zu acht Hardware-Adressen (0..7) per Pins ADR0..ADR2 einstellbar
- Das eDIP mit Adresse 7 ist nach PowerOn selektiert und Empfangsbereit
- Die eDIPs mit Adresse 0..6 sind nach PowerOn deselektiert
- Bis zu 246 weitere Software-Adressen per Befehl '#KA adr' im PowerOnMakro einstellbar (nur bei eDIPs mit Adresse 0 möglich)



SPIINTERFACE

Wird das Display wie unten gezeigt beschaltet, ist der SPI-Mode aktiviert. Die Datenübertragung erfolgt dann über die serielle synchrone SPI-Schnittstelle. Mit den Pins DORD, CPOL, CPHA werden die Hardwarebedingungen an den Master angepasst.

Hinweis:

Die Pins DORD, CPOL, CPHA, DPOM und TEST/SBUF haben einen internen Pull-UP, deshalb ist nur der LO-Pegel (0=GND) aktiv anzulegen. Für Hi-Pegel sind diese Pins offen zu lassen.

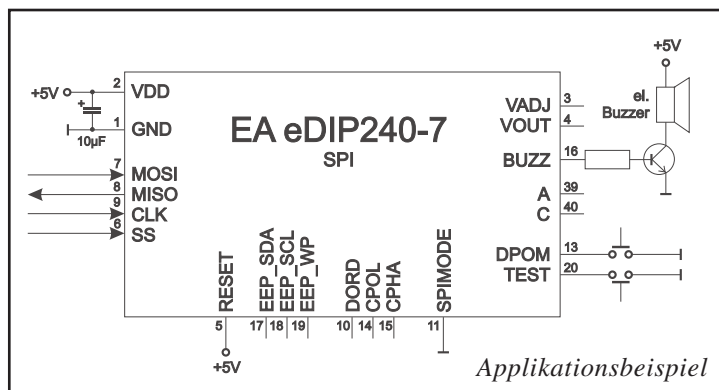
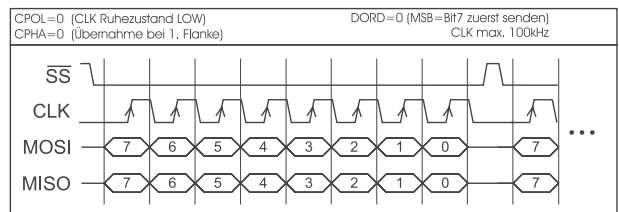
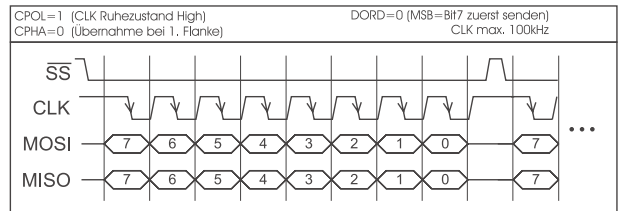
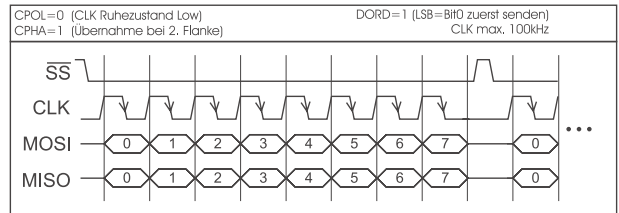
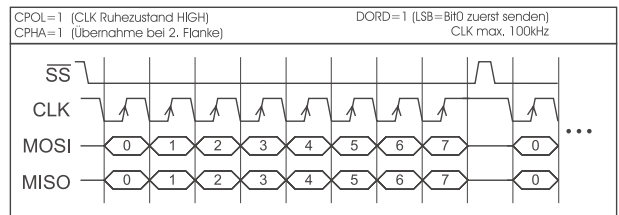
Am Pin 20 (SBUF) zeigt das Display mit einem low-Pegel, dass im internen Sendepuffer Daten zur Abholung bereit stehen. Diese Leitung kann z.B. mit einem Interrupteingang des Host Systems verbunden werden.

Pinout eDIP240-7						
SPI mode						
Pin	Symbol	In/Out	Function	Pin	Symbol	Function
1	GND	-	Ground Potential for logic (0V)	21	N.C.	not connected
2	VDD	-	Power supply for logic (+5V)	22	N.C.	not connected
3	VADJ	In	Operating voltage for LC driving (input)	23	N.C.	not connected
4	VOUT	Out	Output voltage for LC driving	24	N.C.	not connected
5	RESET	-	L: Reset	25	N.C.	not connected
6	SS	In	Slave Select	26	N.C.	not connected
7	MOSI	In	Serial In	27	N.C.	not connected
8	MISO	Out	Serial Out	28	N.C.	not connected
9	CLK	In	Shift Clock	29	N.C.	not connected
10	DORD	In	Data Order (0=MSB first; 1=LSB first)	30	N.C.	not connected
11	SPIMODE	In	connect to GND for SPI interface	31	N.C.	not connected
12	OUT2	Out	open-drain with internal pullup 20..50k (V1.6 or later)	32	N.C.	not connected
13	DPOM	In	L: disable Power-On-Macro do not connect for normal operation	33	N.C.	not connected
14	CPOL	In	Clock Polarity (0=LO 1=HI when idle)	34	N.C.	not connected
15	CPHA	In	Clock Phase (sampled on 0=1st 1=2nd edge)	35	N.C.	not connected
16	BUZZ	Out	Buzzer output	36	N.C.	not connected
17	EEP_SDA	Bidir.	Serial Data Line for int. EEPROM	37	N.C.	not connected
18	EEP_SCL	Out	Serial Clock Line for int. EEPROM	38	N.C.	not connected
19	EEP_WP	In	H: Write Protect for int. EEPROM	39	N.C.	not connected
20	TEST SBUF	IN Out	open-drain with internal pullup 20..50k (Power-On) L: Testmode OUT L: data in sendbuffer	40	N.C.	not connected

DATENÜBERTRAGUNG SPI

Eine Datenübertragung zum eDIP240 ist bis zu 100 kHz Nonstop möglich. Wenn jedoch zwischen den einzelnen Bytes während der Übertragung Pausen von jeweils min. 100 µs eingehalten werden, kann ein Byte mit bis zu 3 MHz übertragen werden.

Um Daten vom eDIP240 zu Lesen (z.B. das ACK-Byte) muss ein Dummy-Byte (z.B. 0xFF) gesendet werden. Das EA eDIP240-7 benötigt eine bestimmte Zeit um die Daten bereit zu stellen; deshalb muss vor jedem zu lesenden Byte zusätzlich mindestens 6µs gewartet werden (keine Aktivität auf der CLK Leitung). Dies gilt auch für 100kHz Betrieb.



I²C-BUS INTERFACE

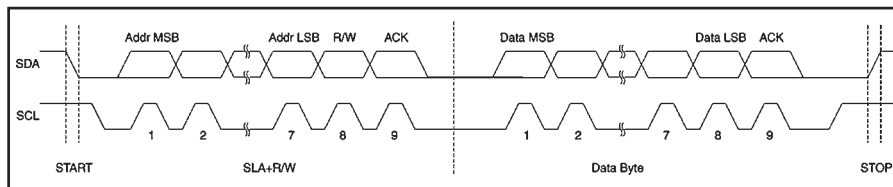
Eine Beschaltung des Displays wie unten ermöglicht den direkten Betrieb an einem I²C-Bus. Am Display kann zwischen 8 unterschiedlichen Basisadressen und 8 verschiedenen Slave-Adressen ausgewählt werden.

Eine Datenübertragung ist bis zu 100 kHz möglich. Wenn jedoch zwischen den einzelnen Bytes während der Übertragung Pausen von jeweils min. 100 µs eingehalten werden, kann ein Byte mit bis zu 400 kHz übertragen werden.

Pinout eDIP240-7						
I ² C-Bus mode						
Pin	Symbol	In/Out	Function	Pin	Symbol	Function
1	GND	-	Ground Potential for logic (0V)	21	N.C.	not connected
2	VDD	-	Power supply for logic (+5V)	22	N.C.	not connected
3	VADJ	In	Operating voltage for LC driving (input)	23	N.C.	not connected
4	VOUT	Out	Output voltage for LC driving	24	N.C.	not connected
5	RESET	-	L: Reset	25	N.C.	not connected
6	BA0	In	Basic Address 0	26	N.C.	not connected
7	BA1	In	Basic Address 1	27	N.C.	not connected
8	SA0	In	Slave Address 0	28	N.C.	not connected
9	SA1	In	Slave Address 1	29	N.C.	not connected
10	SA2	In	Slave Address 2	30	N.C.	not connected
11	BA2	In	Basic Address 2 (V1.3 or later)	31	N.C.	not connected
12	I2CMODE	In	connect to GND for I ² C interface	32	N.C.	not connected
13	DPOM	In	L: disable Power-On-Macro do not connect for normal operation	33	N.C.	not connected
14	SDA	Bidir.	Serial Data Line	34	N.C.	not connected
15	SCL	In	Serial Clock Line	35	N.C.	not connected
16	BUZZ	Out	Buzzer output	36	N.C.	not connected
17	EEP_SDA	Bidir.	Serial Data Line for int. EEPROM	37	N.C.	not connected
18	EEP_SCL	Out	Serial Clock Line for int. EEPROM	38	N.C.	not connected
19	EEP_WP	In	H: Write Protect for int. EEPROM	39	N.C.	not connected
20	TEST SBUF	IN Out	open-drain with internal pullup 20..50k IIN (Power-On) L: Testmode OUT L: data in sendbuffer	40	N.C.	not connected

Hinweis:

Die Pins BA0..2, SA0..2, DPOM, DPROT und TEST/SBUF haben einen internen Pull-Up, deshalb ist nur der LO-Pegel (L=0=GND) aktiv anzulegen. Für Hi-Pegel (H=1) sind diese Pins offen zu lassen. Am Pin 20 (SBUF) zeigt das Display mit einem LO-Pegel, dass im internen Sendepuffer Daten zur Abholung bereit stehen. Diese Leitung kann z.B. mit einem Interrupteingang des Host Systems verbunden werden.



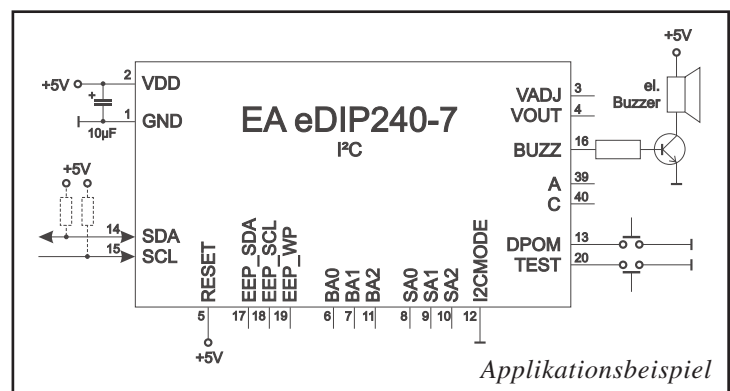
I ² C - Address													
Pin 11,7,6			Base address	I ² C address									
BA2	BA1	BA0		D7	D6	D5	D4	D3	D2	D1	D0		
L	L	L	\$10	0	0	0	1	S	A	1	S	A	0
L	L	H	\$20	0	0	1	0						
L	H	L	\$30	0	0	1	1	S	A	2	S	A	0
L	H	H	\$40	0	1	0	0						
H	L	L	\$70	0	1	1	1	S	A	1	S	A	0
H	L	H	\$90	1	0	0	1						
H	H	L	\$B0	1	0	1	1	S	A	1	S	A	0
H	H	H	\$D0	1	1	0	1						

DATENÜBERTRAGUNG I²C-BUS

So funktioniert prinzipiell die Übertragung:

- I²C-Start
- Master-Transmit: Display-I²C-Adr. (z.B. \$DE), Smallprotokollpaket (Daten) senden
- I²C-Stop
- I²C-Start
- Master-Read: Display-I²C-Adr. (z.B. \$DF), ACK-Byte und evtl. Smallprotokollpaket (Daten) lesen
- I²C-Stop

Das Display benötigt eine bestimmte Zeit um die Daten bereit zu stellen; deshalb muss vor jedem zu lesenden Byte mindestens 6µs gewartet werden (keine Aktivität auf der SCL Leitung).



Applikationsbeispiel

DATENÜBERTRAGUNGSPROTOKOLL (SMALL PROTOKOLL)

Das Protokoll ist für alle 3 Schnittstellenarten RS-232, SPI und I²C identisch aufgebaut. Die Datenübertragung ist jeweils eingebettet in einen festen Rahmen mit Prüfsumme „bcc“. Das EA eDIP240-7 quittiert dieses Paket mit dem Zeichen <ACK> (= \$06) bei erfolgreichem Empfang oder <NAK> (= \$15) bei fehlerhafter Prüfsumme oder Empfangspufferüberlauf. In jedem Fall wird bei <NAK> das komplette Paket verworfen und muss nochmal gesendet werden.

Ein <ACK> bestätigt lediglich die korrekte Übertragung. Ein Syntax-Check erfolgt nicht.

Hinweis: <ACK> muß eingelesen werden.

Empfängt der Hostrechner keine Quittierung, so ist mindestens ein Byte verloren gegangen. In diesem Fall muss die eingestellte Timeoutzeit abgewartet werden, bevor das Paket komplett wiederholt wird.

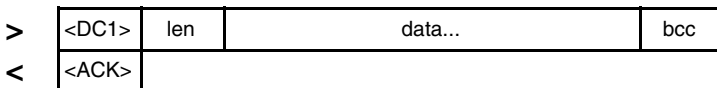
Die Anzahl (len) der Rohdaten pro Paket kann max. 64 Byte betragen. Befehle die grösser als 64 Byte (z.B. Bild laden ESC UL ...) müssen auf mehrere Pakete aufgeteilt werden. Alle Daten in den Paketen werden nach korrektem Empfang von eDIP wieder zusammengefügt.

SMALL PROTOKOLL DEAKTIVIEREN

Das Protokoll ist für alle drei Schnittstellen RS-232, I²C und SPI identisch. Für Tests kann das Protokoll durch Schliessen der Lötbrücke J2 (siehe Seite 20) abgeschaltet werden. Im normalen Betrieb ist allerdings die Aktivierung des Protokolls unbedingt zu empfehlen. Andernfalls wäre ein möglicher Überlauf des Empfangspuffers nicht zu erkennen.

DIE PAKETVARIANTEN IN EINZELNEN

Befehle/Daten zum Display senden

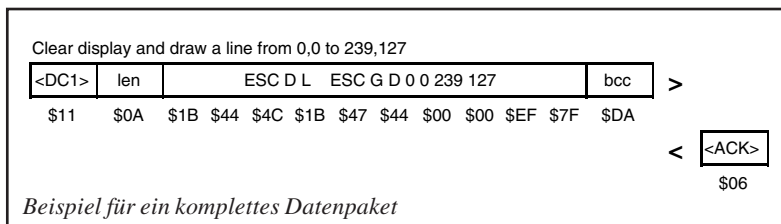


<DC1> = 17(dez.) = \$11

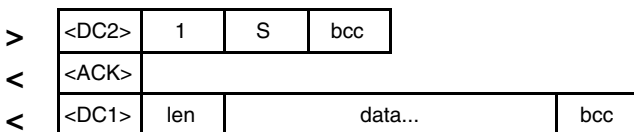
<ACK> = 6(dez.) = \$06

len = Anzahl der Nutzdaten in Byte (ohne Prüfsumme, ohne <DC1>)

bcc = 1 Byte = Summe aus allen Bytes inkl. <DC1> und len, Modulo 256



Inhalt des Sendepuffers anfordern

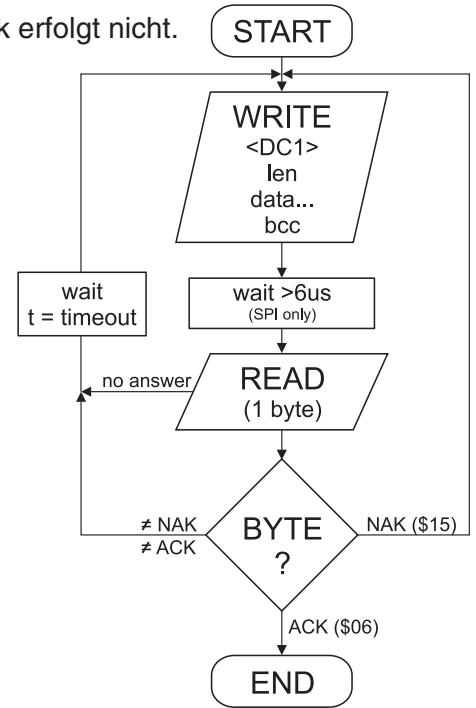


<DC2> = 18(dez.) = \$12 1 = 1(dez.) = \$01 S = 83(dez.) = \$53

<ACK> = 6(dez.) = \$06

len = Anzahl der Nutzdaten in Byte (ohne Prüfsumme, ohne <DC1>)

bcc = 1 Byte = Summe aus allen Bytes inkl. <DC1> und len, Modulo 256



Eingerahmt von <DC1>, der Anzahl der Daten "len" und der Prüfsumme "bcc" werden die jeweiligen Nutzdaten übertragen. Als Antwort sendet das Display <ACK> zurück.

```
void sendData(unsigned char *buf, unsigned char len)
{
    unsigned char i, bcc;

    SendByte(0x11);           // Send DC1
    bcc = 0x11;

    SendByte(len);           // Send data length
    bcc = bcc + len;

    for(i=0; i < len; i++)    // Send buf
    {
        SendByte(buf[i]);
        bcc = bcc + buf[i];
    }

    SendByte(bcc);           // Send checksum
}
```

C-Beispiel zum Senden eines Datenpaketes

Die Befehlsfolge <DC2>, 1, S, bcc entleert den Sendepuffer des Displays. Das Display antwortet zuerst mit der Quittierung <ACK> und beginnt dann alle gesammelten Daten wie z.B. Touchtastendrucke zu senden.

Pufferinformationen anfordern

>	<DC2>	1	I	bcc	
<	<ACK>				
<	<DC2>	2	send buffer bytes ready	receive buffer bytes free	bcc

<DC2> = 18(dez.) = \$12 I = 1(dez.) = \$01 I = 73(dez.) = \$49

<ACK> = 6(dez.) = \$06

send buffer bytes ready = Anzahl abholbereiter Bytes

receive buffer bytes free = verfügbarer Platz im Empfangspuffer

bcc = 1 Byte = Summe aus allen Bytes inkl. <DC2> Modulo 256

Mit diesem Befehl wird abgefragt, ob Nutzdaten zur Abholung bereit stehen und wie voll der Empfangspuffer des Displays bereits ist.

Protokolleinstellungen

>	<DC2>	3	D	packet size for send buffer	timeout	bcc
<	<ACK>					

<DC2> = 18(dez.) = \$12 3 = 3(dez.) = \$03 D = 68(dez.) = \$44

packet size for send buffer = 1..64 (Standard: 64)

timeout = 1..255 in 1/100 Sekunden (Standard: 200 = 2 Sekunden)

bcc = 1 Byte = Summe aus allen Bytes inkl. <DC2>, Modulo 256

<ACK> = 6(dez.) = \$06

Hierüber läßt sich die maximale Paketgröße welche das Display senden darf begrenzen. Voreingestellt ist eine Paketgröße mit bis zu 64 Byte Nutzdaten. Weiterhin läßt sich der Timeout in 1/100s einstellen. Der Timeout spricht an, wenn einzelne Bytes verloren gegangen sind. Danach muß das gesamte Paket nochmals übertragen werden.

Protokollinformationen anfordern

>	<DC2>	1	P	bcc		
<	<ACK>					
<	<DC2>	3	max. packet size	akt. send packet size	akt. timeout	bcc

<DC2> = 18(dez.) = \$12 I = 1(dez.) = \$01 P = 80(dez.) = \$50

<ACK> = 6(dez.) = \$06

max. packet size = maximale Anzahl der Nutzdaten eines Protokollpaketes (eDIP240-7 = 64)

akt. send packet size = eingestellte Paketgröße zum Senden

akt. timeout = eingestellter timeout in 1/100 Sekunden

bcc = 1 Byte = Summe aus allen Bytes inkl. <DC2>, Modulo 256

Mit diesem Befehl werden Protokolleinstellungen abgefragt.

Letztes Datenpaket wiederholen

>	<DC2>	1	R	bcc
<	<ACK>			
<	<DC1> <DC2>	len	data...	bcc

<DC2> = 18(dez.) = \$12 I = 1(dez.) = \$01 R = 82(dez.) = \$52

<ACK> = 6(dez.) = \$06

<DC1> = 17(dez.) = \$11

len = Anzahl der Nutzdaten in Byte (ohne Prüfsumme, ohne <DC1> bzw. <DC2>)

bcc = 1 Byte = Summe aus allen Bytes inkl. <DC2> und len, Modulo 256

Falls das zuletzt angeforderte Paket eine falsche Prüfsumme enthielt, kann das komplette Paket nochmals angefordert werden. Die Antwort kann dann der Inhalt des Sendepuffers (<DC1>) oder die Puffer-/Protokoll-Information (<DC2>) sein.

Adressierung nur bei RS232/RS485 Betrieb

>	<DC2>	3	A	select or deselect	adr	bcc
<	<ACK>					

<DC2> = 18(dez.) = \$12 3 = 3(dez.) = \$03 A = 65(dez.) = \$41

select or deselect: 'S' = 83(dez.) = \$53 oder 'D' = 68(dez.) = \$44

adr = 0..255

bcc = 1 Byte = Summe aus allen Bytes inkl. <DC2> und len, Modulo 256

<ACK> = 6(dez.) = \$06

Mit diesem Befehl läßt sich das eDIP mit der Adresse adr Selektieren oder Deselektieren.

TOUCH PANEL(NUR EAeDIP240x-7xxTP)

Die Versionen -7xxTP werden mit einem analogen resistiven Touchpanel geliefert. Bis zu 60 Touchbereiche (Tasten, Schalter, Menüs, Bargrapheingaben), können gleichzeitig definiert werden. Eine pixelgenaue Definition ist möglich. Das Display unterstützt die Darstellung mit komfortablen Befehlen (siehe Seite 15). Beim Berühren der Touch-"Tasten" können diese automatisch invertiert werden und ein externer Summer (Pin 16) signalisiert die Berührung. Der zuvor definierte Return-Code der "Taste" wird über die Schnittstelle gesendet oder es wird statt dessen ein internes Touch Makro mit der Nummer des Return-Codes gestartet (siehe Seite 18, *Makroprogrammierung*).

TOUCHPANELABGLEICH

Das Touchpanel ist bei Auslieferung abgeglichen und sofort einsatzbereit. Durch Alterung und Abnutzung kann es nötig sein, dass das Touchpanel neu abgeglichen werden muss.

Abgleichprozedur:

1. Beim Einschalten Touch berühren und gedrückt halten. Nach Erscheinen der Meldung "*touch adjustment ?*" den Touch wieder loslassen (alternativ den Befehl 'ESC @' senden).
2. Innerhalb 1 Sekunde den Touch nochmals für mindestens 1 Sekunde berühren.
3. Den Anweisungen zum Abgleich folgen (2 Punkte *Linksoben* und *Rechtsunten* betätigen).

RAHMEN UNDTASTENFORMEN

Mit den Befehlen *Rahmen /Rahmenbox zeichnen* sowie beim Zeichnen von Touchtasten kann ein Rahmentyp eingestellt werden. Es stehen dabei 18 Rahmentypen zur Verfügung (0= keinen Rahmen zeichnen). Die Rahmengröße muß mindestens 16x16 Pixel betragen.

BITMAPS ALSTASTEN

Ausser den Rahmentypen, die in der Grösse frei skalierbar sind, gibt es noch die Möglichkeit beliebige Bitmaps (jeweils 2 Stück für *nicht-gedrückt* und *gedrückt*) als Touch-Tasten oder -Schalter zu verwenden.

Über die ELECTRONIC ASSEMBLY LCD-Tools*) können eigene Buttons als Bilder eingebunden werden (Compileranweisung "PICTURE"). Ein Button besteht immer

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



SCHALTER IN GRUPPEN (RADIO GROUP)

Touch-Schalter ändern ihren Zustand bei jeder Berührung von *EIN* in *AUS* und umgekehrt. Mehrere Touchschalter können zu einer Gruppe zusammengefasst werden (Befehl: 'ESC A R nr'). Wird nun ein Touch-Schalter innerhalb einer Gruppe 'nr' eingeschaltet, dann werden automatisch alle andern Touch-Schalter dieser Gruppe ausgeschaltet. Es ist also automatisch immer nur ein Schalter gesetzt.

*) im Internet unter <http://www.lcd-module.de/deu/touch/touch.htm>

INTEGRIERTE UND EXTERNE FONTS

Es sind standardmäßig, außer dem 8x8 Terminalfont (Font-Nr. 0), noch 3 monospaced, 3 proportionale Zeichensätze und 1 grosser Ziffernfont integriert. Die proportionalen Zeichensätze ergeben ein schöneres Schriftbild, gleichzeitig benötigen sie weniger Platz auf dem Bildschirm (z.B. schmales "i" und breites "W"). Jedes Zeichen kann **pixelgenau** platziert werden und in der Höhe und Breite von 1- bis 4-fach vergrößert werden.

Texte lassen sich linksbündig, rechtsbündig und zentriert ausgeben. Auch eine 90° Drehung, z.B. für vertikalen Einbau des Displays, ist möglich.

Die Makroprogrammierung erlaubt die Einbindung von weiteren Fonts (max. 15). Diese können mit einem Texteditor erstellt und über den **KIT-Compiler*** geladen werden (EA 9777-1USB).

+ Lower Upper	\$0 (0)	\$1 (1)	\$2 (2)	\$3 (3)	\$4 (4)	\$5 (5)	\$6 (6)	\$7 (7)	\$8 (8)	\$9 (9)	\$A (10)	\$B (11)	\$C (12)	\$D (13)	\$E (14)	\$F (15)
\$20 (dez: 32)		!	"	#	\$	%	&	'	()	*	+	,	-	.	/
\$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
\$40 (dez: 64)	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
\$50 (dez: 80)	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
\$60 (dez: 96)	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
\$70 (dez: 112)	p	q	r	s	t	u	v	w	x	y	z	{		}	~	Δ
\$80 (dez: 128)	€	ü	é	â	ä	à	ã	ç	ê	ë	è	ï	î	í	ñ	ñ
\$90 (dez: 144)	É	æ	Æ	ô	ö	ò	ó	ù	û	ý	ÿ	ƒ	ƒ	ƒ	ƒ	ƒ
\$A0 (dez: 160)	á	í	ó	ú	ñ	ñ	ã	ø	¿	¡	½	¼	¾	¼	¼	¼
\$B0 (dez: 176)																
\$C0 (dez: 192)																
\$D0 (dez: 208)																
\$E0 (dez: 224)	α	β	Γ	π	Σ	σ	μ	τ	ξ	θ	η	δ	φ	ψ	ε	π
\$F0 (dez: 240)	≡	±	≥	≤	Γ	J	÷	≈	°	•	•	•	•	•	•	•

Font 1: 4x6 monospaced

+ Lower Upper	\$0 (0)	\$1 (1)	\$2 (2)	\$3 (3)	\$4 (4)	\$5 (5)	\$6 (6)	\$7 (7)	\$8 (8)	\$9 (9)	\$A (10)	\$B (11)	\$C (12)	\$D (13)	\$E (14)	\$F (15)
\$20 (dez: 32)		!	"	#	\$	%	&	'	()	*	+	,	-	.	/
\$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
\$40 (dez: 64)	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
\$50 (dez: 80)	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
\$60 (dez: 96)	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
\$70 (dez: 112)	p	q	r	s	t	u	v	w	x	y	z	{		}	~	Δ
\$80 (dez: 128)	€	ü	é	â	ä	à	ã	ç	ê	ë	è	ï	î	í	ñ	ñ
\$90 (dez: 144)	É	æ	Æ	ô	ö	ò	ó	ù	û	ý	ÿ	ƒ	ƒ	ƒ	ƒ	ƒ
\$A0 (dez: 160)	á	í	ó	ú	ñ	ñ	ã	ø	¿	¡	½	¼	¾	¼	¼	¼
\$B0 (dez: 176)																
\$C0 (dez: 192)																
\$D0 (dez: 208)																
\$E0 (dez: 224)	α	β	Γ	π	Σ	σ	μ	τ	ξ	θ	η	δ	φ	ψ	ε	π
\$F0 (dez: 240)	≡	±	≥	≤	Γ	J	÷	≈	°	•	•	•	•	•	•	•

Font 3: 7x12 monospaced

+ Lower Upper	\$0 (0)	\$1 (1)	\$2 (2)	\$3 (3)	\$4 (4)	\$5 (5)	\$6 (6)	\$7 (7)	\$8 (8)	\$9 (9)	\$A (10)	\$B (11)	\$C (12)	\$D (13)	\$E (14)	\$F (15)
\$20 (dez: 32)		!	"	#	\$	%	&	'	()	*	+	,	-	.	/
\$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
\$40 (dez: 64)	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
\$50 (dez: 80)	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
\$60 (dez: 96)	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
\$70 (dez: 112)	p	q	r	s	t	u	v	w	x	y	z	{		}	~	Δ
\$80 (dez: 128)	€	ü	é	â	ä	à	ã	ç	ê	ë	è	ï	î	í	ñ	ñ
\$90 (dez: 144)	É	æ	Æ	ô	ö	ò	ó	ù	û	ý	ÿ	ƒ	ƒ	ƒ	ƒ	ƒ
\$A0 (dez: 160)	á	í	ó	ú	ñ	ñ	ã	ø	¿	¡	½	¼	¾	¼	¼	¼
\$B0 (dez: 176)																
\$C0 (dez: 192)																
\$D0 (dez: 208)																
\$E0 (dez: 224)	α	β	Γ	π	Σ	σ	μ	τ	ξ	θ	η	δ	φ	ψ	ε	π
\$F0 (dez: 240)	≡	±	≥	≤	Γ	J	÷	≈	°	•	•	•	•	•	•	•

Font 2: 6x8 monospaced

+ Lower Upper	\$0 (0)	\$1 (1)	\$2 (2)	\$3 (3)	\$4 (4)	\$5 (5)	\$6 (6)	\$7 (7)	\$8 (8)	\$9 (9)	\$A (10)	\$B (11)	\$C (12)	\$D (13)	\$E (14)	\$F (15)
\$20 (dez: 32)		!	"	#	\$	%	&	'	()	*	+	,	-	.	/
\$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
\$40 (dez: 64)	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
\$50 (dez: 80)	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
\$60 (dez: 96)	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
\$70 (dez: 112)	p	q	r	s	t	u	v	w	x	y	z	{		}	~	Δ
\$80 (dez: 128)	€	ü	é	â	ä	à	ã	ç	ê	ë	è	ï	î	í	ñ	ñ
\$90 (dez: 144)	É	æ	Æ	ô	ö	ò	ó	ù	û	ý	ÿ	ƒ	ƒ	ƒ	ƒ	ƒ
\$A0 (dez: 160)	á	í	ó	ú	ñ	ñ	ã	ø	¿	¡	½	¼	¾	¼	¼	¼
\$B0 (dez: 176)																
\$C0 (dez: 192)																
\$D0 (dez: 208)																
\$E0 (dez: 224)	α	β	Γ	π	Σ	σ	μ	τ	ξ	θ	η	δ	φ	ψ	ε	π
\$F0 (dez: 240)	≡	±	≥	≤	Γ	J	÷	≈	°	•	•	•	•	•	•	•

Font 4: GENEVA10 proportional

+ Lower Upper	\$0 (0)	\$1 (1)	\$2 (2)	\$3 (3)	\$4 (4)	\$5 (5)	\$6 (6)	\$7 (7)	\$8 (8)	\$9 (9)	\$A (10)	\$B (11)	\$C (12)	\$D (13)	\$E (14)	\$F (15)
\$20 (dez: 32)		!	"	#	\$	%	&	'	()	*	+	,	-	.	/	
\$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
\$40 (dez: 64)	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
\$50 (dez: 80)	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_		
\$60 (dez: 96)	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
\$70 (dez: 112)	p	q	r	s	t	u	v	w	x	y	z	{ }	~	Δ		
\$80 (dez: 128)	€	ü	é	â	ä	à	ã	ç	ê	ë	è	ï	î	í	Ä	Å
\$90 (dez: 144)	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü					
\$A0 (dez: 160)	á	í	ó	ú	ñ	Ñ	ä	ö								
\$B0 (dez: 176)																
\$C0 (dez: 192)																
\$D0 (dez: 208)																
\$E0 (dez: 224)		ß														
\$F0 (dez: 240)								°								

Font 5: CHICAGO14 proportional

+ Lower Upper	\$0 (0)	\$1 (1)	\$2 (2)	\$3 (3)	\$4 (4)	\$5 (5)	\$6 (6)	\$7 (7)	\$8 (8)	\$9 (9)	\$A (10)	\$B (11)	\$C (12)	\$D (13)	\$E (14)	\$F (15)
\$20 (dez: 32)		!	"	#	\$	%	&	'	()	*	+	,	-	.	/	
\$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
\$40 (dez: 64)	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
\$50 (dez: 80)	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_		
\$60 (dez: 96)	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
\$70 (dez: 112)	p	q	r	s	t	u	v	w	x	y	z	{ }	~	Δ		
\$80 (dez: 128)	€	ü	é	â	ä	à	ã	ç	ê	ë	è	ï	î	í	Ä	Å
\$90 (dez: 144)	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü					
\$A0 (dez: 160)	á	í	ó	ú	ñ	Ñ	ä	ö								
\$B0 (dez: 176)																
\$C0 (dez: 192)																
\$D0 (dez: 208)																
\$E0 (dez: 224)		ß														
\$F0 (dez: 240)								°								

Font 6: Swiss30 Bold proportional

+ Lower Upper	\$0 (0)	\$1 (1)	\$2 (2)	\$3 (3)	\$4 (4)	\$5 (5)	\$6 (6)	\$7 (7)	\$8 (8)	\$9 (9)	\$A (10)	\$B (11)	\$C (12)	\$D (13)	\$E (14)	\$F (15)
\$20 (dez: 32)												+	-	.		
\$30 (dez: 48)	0	1	2	3	4	5	6	7	8	9	:					

Font 7: grosse Ziffern BigZif57

SCHRIFTBILD

Diese Hardcopy zeigt alle eingebauten Standard Schriften.

Die Makroprogrammierung erlaubt die Einbindung von weiteren Fonts. Es können alle nur erdenklichen Schriften (einschl. kyrillisch und chinesisches) mit einem Texteditor erstellt und über den Kitcompiler / LCD-Toolkit*) programmiert werden (Programmer EA 9777-1USB notwendig).



*) im Internet unter <http://www.lcd-module.de/deu/touch/touch.htm>

ALLE BEFEHLE AUF EINEN BLICK

Die eingebaute Intelligenz erlaubt den Aufbau eines Bildschirms über unten stehende Befehle. Alle Befehle können sowohl über die serielle Schnittstelle (vgl. Seite 17) als auch in selbst-definierten Makros (vgl. Seiten 18/19) verwendet werden.

EA eDIP240-7: Befehlstabelle 1							nach			
Befehl	Codes	Anmerkung					Rese			
Befehle für den Terminal Betrieb										
Formfeed FF (dez:12)	^L	Bildschirm wird gelöscht und der Cursor nach Pos. (1,1) gesetzt								
Carriage Return CR(13)	^M	Cursor ganz nach links zum Zeilenanfang								
Linefeed LF (dez:10)	^J	Cursor 1 Zeile tiefer, falls Cursor in letzter Zeile dann wird gescrollt								
Cursor positionieren	ESC	T	P	n1	n2	n1=Spalte; n2=Zeile; Ursprung links oben ist (1,1)	1,1			
Cursor On / Off			C	n1		n1=0: Cursor ist unsichtbar; n1=1: Cursor blinkt;	1			
Cursorposition sichern			S	die aktuelle Cursorposition wird gesichert (ab V1.2)						
Cursorposition restoren			R	die letzte gesicherte Cursorposition wird wieder hergestellt (ab V1.2)						
Terminal AUS			A	Terminal Anzeige ist ausgeschaltet; Ausgaben werden verworfen						
Terminal EIN			E	Terminal Anzeige ist eingeschaltet;				Ein		
Version ausgeben	V	Die Versions-Nr. wird im Terminal ausgegeben z.B "EA eDIP240-7 V1.1 Rev.B"								
Befehle zur Ausgabe von Zeichenketten										
Zeichenkette ausgeben L: Linksbündig C: Zentriert R: Rechtsbündig	ESC	Z	L	x1	y1	Text ... NU	Eine Zeichenkette (...) an x1,y1 ausgegeben; Zeichenkettenende: 'NUL' (\$00), 'LF' (\$0A) oder 'CR' (\$0D); Mehrere Zeilen werden durch das Zeichen ' ' (\$7C) getrennt; Texte die zwischen zwei '-' (\$7E) Zeichen stehen blinken An/Aus; Texte die zwischen zwei '@' (\$40) Zeichen stehen blinken Invertierend; Das Backslash-Zeichen '\' (\$5C) hebt die Sonderfunktion der Zeichen ' ~@\' auf; z.B. "name@test.de" => "name@test.de"			
Font einstellen			F	n1			Font mit der Nummer n1 (0..15) einstellen	0		
Font-Zoomfaktor			Z	n1	n2		n1 = X-Zoomfaktor (1x..4x); n2 = Y-Zoomfaktor (1x..4x)	1,1		
zus. Zeilenabstand			Y	n1			zwischen zwei Textzeilen n1 Pixel (0..15) als zusätzlichen Zeilenabstand einfügen			
Text-Winkel			W	n1			Text-Ausgabewinkel: n1=0: 0°; n1=1: 90°;	0		
Text-Verknüpfungsmodu Text-Blinkattribut			V	n1			Modus n1: 1=setzen; 2=löschen; 3=invers; 4=Replace; 5=Invers Replace; n1: 0=blinken Aus; 1=Text blinkt An/Aus; 2=Text blinkt Invertierend;	4 0		
Zeichenkette für Terminal	ESC	Z	T	Text ...			Befehl um eine Zeichenkette in einem Makro an das Terminal ausgeben zu können			
Geraden und Punkte zeichnen										
Rechteck zeichnen	ESC	G	R	x1	y1	x2	y2	Vier Geraden als Rechteck von x1,y1 nach x2,y2 zeichnen		
Gerade zeichnen			D	x1	y1	x2	y2	Eine Gerade von x1,y1 nach x2,y2 zeichnen		
Gerade weiter zeichnen			W	x1	y1			Eine Gerade vom letzten Endpunkt bis x1, y1 zeichnen	0	
Punkt zeichnen			P	x1	y1			Ein Punkt an die Koordinaten x1, y1 setzen		
Punktgröße / Liniendicke			Z	n1	n2			n1 = X-Punktgröße (1..15); n2 = Y-Punktgröße (1..15);	1,1	
Verknüpfungsmodus			V	n1				Zeichenmodus einstellen n1: 1=setzen; 2=löschen; 3=invers;	1	
Rechteckige Bereiche verändern / zeichnen										
Bereich löschen	ESC	R	L	x1	y1	x2	y2	Einen Bereich von x1,y1 nach x2,y2 löschen (alle Pixel aus)		
Bereich invertieren			I	x1	y1	x2	y2	Einen Bereich von x1,y1 nach x2,y2 invertieren (alle Pixel umkehren)		
Bereich füllen			S	x1	y1	x2	y2	Einen Bereich von x1,y1 nach x2,y2 füllen (alle Pixel ein)		
Bereich m. Füllmuster			M	x1	y1	x2	y2	n1	Einen Bereich von x1,y1 nach x2,y2 mit Muster n1 zeichnen (immer setzen)	
Box zeichnen			O	x1	y1	x2	y2	n1	Ein Rechteck von x1,y1 nach x2,y2 mit Muster n1 zeichnen; (immer Replace)	
Rahmen zeichnen Rahmenbox zeichnen			R T	x1	y1	x2	y2	n1	Einen Rahmen Typ n1 von x1,y1 nach x2,y2 zeichnen (immer setzen) Eine Rahmenbox Typ n1 von x1,y1 nach x2,y2 zeichnen; (immer Replace)	
Bitmap Bilder Befehle										
Bild aus Clipboard	ESC	U	C	x1	y1			Der akt. Clipboardinhalt wird mit allen Bildattributen nach x1,y1 geladen		
internes Bild laden			I	x1	y1	nr			internes Bild mit der nr (0..255) aus dem EEPROM nach x1,y1 laden	
Bild laden			L	x1	y1	BLH	daten ...		Ein Bild nach x1,y1 laden; daten des Bildes siehe Bildaufbau	
Bild-Zoomfaktor			Z	n1	n2				n1 = X-Zoomfaktor (1x..4x); n2 = Y-Zoomfaktor (1x..4x)	1,1
Bild-Winkel			W	n1					Ausgabewinkel des Bildes: n1=0: 0°; n1=1: 90°	0
Bild-Verknüpfungsmodus			V	n1					Modus n1: 1=setzen; 2=löschen; 3=invers; 4=Replace; 5=Invers Replace;	4
Bild-Blinkattribut	B	n1					n1=0: Bild blinken Aus; n1=1: Bild blinkt An/Aus; n1=2: Bild blinkt Invertierend	0		
Hardcopy senden	H	x1	y1	x2	y2		Es wird ein Bild angefordert. Zuerst werden die Breite und Höhe in Pixel und dann die eigentlichen Bilddaten gesendet.			
Display-Befehle (Wirkung auf das gesamte Display)										
Display löschen	ESC	D	L					Displayinhalt löschen (alle Pixel aus)		
Display invertieren			I					Displayinhalt invertieren (alle Pixel umkehren)		
Display füllen			S					Displayinhalt füllen (alle Pixel ein)		
Display ausschalten			A					Displayinhalt wird unsichtbar bleibt aber erhalten, Befehle weiterhin möglich		
Display einschalten			E					Displayinhalt wird wieder sichtbar	Ein	
Display Clipboard Disp. Normaldarstellung			C N					Inhalt des Clipboards wird dargestellt. Displayausgaben sind nicht mehr sichtbar Aktuelles Bild wird dargestellt (Normalbetrieb). Alle Ausgaben wieder sichtbar		
Blinkbereichs-Befehle										
Blinkattribut löschen	ESC	Q	L	x1	y1	x2	y2	Löscht das Blinkattribut von x1,y1 bis x2,y2		
Invertierender Blinkbereich			I	x1	y1	x2	y2		Definiert einen invertierenden Blinkbereich von x1,y1 bis x2,y2	
Muster Blinkbereich			M	x1	y1	x2	y2	n1	Definiert einen Blinkbereich mit Muster n1 (An/Aus) von x1,y1 bis x2,y2	
Blinkzeit einstellen			Z	n1					Einstellen der Blinkzeit n1= 1..15 in 1/10s; 0=Blinkfunktion deaktivieren	6

EA eDIP240-7: Befehlstabelle 2										nach				
Befehl	Codes		Anmerkung							Reset				
Bargraph Befehle														
Bargraph definieren			R L O U	n1	x1	y1	x2	y2	aw	ew	typ	mst	Bargraph nach L(inks), R(echts), O(ben), U(nten) mit der Nr. n1 (1..32) definieren. x1,y1,x2,y2 sind das umschließende Rechteck des Bars. aw,ew sind die Werte für 0% und 100%. typ=0:Balken; typ=1:Balken im Rechteck; mst=Balkenmuster typ=2:Strich; typ=3:Strich im Rechteck; mst=Strichbreite	kein Bar definiert
Bargraph aktualisieren	ESC	B	A	n1	wert	Bargraph mit der Nummer n1 auf den neuen Benutzer- 'wert' setzen und zeichnen.								
Bargraph neu zeichnen			Z	n1	Den Bargraph mit der Nummer n1 komplett neu zeichnen									
Bargraphwert senden			S	n1	Den aktuellen Wert des Bargraph Nr. n1 senden									
Bargraph löschen			D	n1	n2	Die Definition des Bargraph mit der Nummer n1 wird ungültig. War der Bargraph als Eingabe mit Touch definiert so wird auch dieses Touchfeld gelöscht. n2=0: Bar weiterhin sichtbar; n2=1: Bar wird gelöscht								
Clipboard Befehle (Zwischenspeicher für Bildbereiche)														
Displayinhalt sichern			B	Der gesamte Displayinhalt wird als Bildbereich ins Clipboard kopiert										
Bereich sichern	ESC	C	S	x1	y1	x2	y2	Der Bildbereich von x1,y1 bis nach x2,y2 wird ins Clipboard kopiert						
Bereich restaurieren			R	Der Bildbereich im Clipboard wird wieder ins Display kopiert										
Bereich kopieren			K	x1	y1	Der Bildbereich im Clipboard wird ins Display nach x1,y1 kopiert								
Einstellungen für Menübox / Touchmenü														
Menü-Font einstellen			F	n1	Font mit der Nummer n1 (0..15) für Menüdarstellung einstellen						0			
Menüfont-Zoomfaktor			Z	n1	n2	n1 = X-Zoomfaktor (1x..4x); n2 = Y-Zoomfaktor (1x..4x)					1,1			
zus. Zeilenabstand			Y	n1	zwischen Menüeinträgen n1 Pixel (0..15) als zusätzlichen Zeilenabstand einfügen									
Menü-Winkel	ESC	N	W	n1	Menüdarstellung Winkel: n1=0: 0°; n1=1: 90°;						0			
Touchmenü-Automatik			T	n1	n1=1: Touchmenü öffnet automatisch; n1=0: Touchmenü öffnet nicht automatisch stattdessen wird die Anforderung 'ESC T 0' zum Öffnen den Hostrechner gesendet, dieser kann dann mit 'ESC N T 2' das Touchmenü öffnen.						1			
Menübox Befehle (Steuerung mit Tasten nicht per Touch)														
Menü definieren und Darstellen			D	x1	y1	nr	Text ...	NUL	Ein Menü wird ab der Ecke x1,y1 mit dem akt. Menüfont gezeichnet. nr:= aktuell invertierter Eintrag (z.B: 1 = 1. Eintrag) Text:= Zeichenkette mit den Menüeinträgen. Die einzelnen Einträge sind durch Zeichen ' ' (\$7C,dez:124) getrennt z.B. "Eintrag1 Eintrag2 Eintrag3" Der Hintergrund des Menüs wird automatisch gesichert. Ist bereits ein Menü definiert, wird dieses automatisch abgebrochen+entfernt.					
nächster Eintrag	ESC	N	N	Der nächste Eintrag wird invertiert oder bleibt am Ende stehen										
vorheriger Eintrag			P	Der vorherige Eintrag wird invertiert oder bleibt am Anfang stehen										
Menüende / Senden			S	Das Menü wird entfernt und durch den ursprünglichen Hintergrund ersetzt der aktuelle Eintrag wird als Nummer (1..n) gesendet (0=kein Menü dargestellt)										
Menüende / Makro			M	n1	Das Menü wird entfernt und durch den ursprünglichen Hintergrund ersetzt. Für Eintrag 1 wird das Menü-Makro n1 aufgerufen, für Eintrag 2 Menü-Makro nr+1 usw.									
Menüende / Abbrechen			A	Das Menü wird entfernt und durch den ursprünglichen Hintergrund ersetzt										
Makro Befehle														
Normal Makro ausführen			N	n1	Das (Normal-)Makro mit der Nummer n1 (0..255) aufrufen (max. 7 Ebenen)									
Touch Makro ausführen	ESC	M	T	n1	Das Touch-Makro mit der Nummer n1 (0..255) aufrufen (max. 7 Ebenen)									
Menü Makro ausführen			M	n1	Das Menü-Makro mit der Nummer n1 (0..255) aufrufen (max. 7 Ebenen)									
automatische (Normal-) Makros														
Makro mit Verzögerung			G	n1	n2	Das (Normal-)Makro mit der Nummer n1 (0..255) in n2/10s aufrufen. Die Ausführung wird durch Befehle (z.B durch Empfang oder Touchmakros) gestoppt.								
autom. Makros einmal	ESC	M	E	n1	n2	n3	Makros n1..n2 automatisch einmal abarbeiten; n3=Pause in 1/10s. Die Ausführung wird durch Befehle (z.B durch Empfang oder Touchmakros) gestoppt.							
autom. Makros zyklisch			A	n1	n2	n3	Makros n1..n2 automatisch zyklisch abarbeiten; n3=Pause in 1/10s. Die Ausführung wird durch Befehle (z.B durch Empfang oder Touchmakros) gestoppt.							
autom. Makros pingpong			J	n1	n2	n3	Makros autom. von n1..n2..n1 (PingPong) abarbeiten; n3=Pause in 1/10s. Die Ausführung wird durch Befehle (z.B durch Empfang oder Touchmakros) gestoppt.							
Makro Prozesse (ab V1.1)														
Makroprozess definieren			D	nr	typ	n3	n4	zs	Ein Makroprozess mit der Nummer nr (1..4) wird definiert (1=höchste Priorität). Die (Normal-) Makros n3 bis n4 werden nacheinander alle zs/10s ausgeführt. typ: 1=einmal; 2=zyklisch; 3=pingpong n3..n4..n3					
Makroprozess Zeitintervall	ESC	M	Z	nr	zs	Dem Makroprozess mit der Nummer nr (1..4) wird eine neue Zeit zs in 1/10s zugeordnet. Ist die Zeit zs=0 so wird die Ausführung angehalten.								
Makroprozesse anhalten			S	n1	Alle Makroprozesse werden mit n1=0 gestoppt und n1=1 wieder gestartet. z.B. um Einstellungen und Ausgaben über die Schnittstelle ungestört auszuführen						1			
Sonstige-Befehle														
Warten (Pause)	ESC	X	n1	n1 Zehntel-Sekunden abwarten bevor der nächste Befehl ausgeführt wird.										
RS485 Adresse einstellen	ESC	K	A	adr	ab V1.3 und nur für RS232/RS485 Betrieb und nur bei Hardwareadresse 0 möglich Dem eDIP wird eine neue Adresse adr zugewiesen (im PowerOn-Makro).									
Summer Ein / Aus			S	n1	Der Summerausgang (PIN16) wird n1=0: AUS; n1=1: EIN; n1=2..255: für n1 Zehntel Sek. lang eingeschaltet						AUS			
Beleuchtung Ein/Aus			L	n1	LED-Beleuchtung n1=0: AUS; n1=1: EIN; n1=2..255: Beleuchtung für n1 Zehntel Sek. lang einschalten						1			
Beleuchtung Helligkeit	ESC	Y	H	n1	Helligkeit der LED-Beleuchtung einstellen n1=0..100% (ab V1.3: n1=254 LED sofort AUS; n1=255 sofort auf 100% stellen).						100			
Output-Port schreiben (ab V1.6)			W	n1	n2	n1=0: Alle Ausgabe-Ports entsprechend n2 (=5/7-Bit Binärwert) einstellen n1=1..5/7: Ausgabe-Port n1 rücksetzen (n2=0); setzen (n2=1); invertieren (n2=2)					auf 1			
Bytes senden			B	anz	daten ...	Es werden anz (=1..255) Bytes zum Sendepuffer gesendet daten ... = anz Bytes Im Quelltext der Makroprogrammierung darf die Anzahl anz nicht angegeben werden, diese wird vom eDIP-Compiler gezählt und eingetragen.								
Version senden	ESC	S	V	Es wird die Version als String gesendet z.B "EA eDIP240-7 V1.3 Rev.B TP+"										
interne Infos senden			I	(ab V1.3) Es werden interne Informationen vom eDIP gesendet.										

EA eDIP240-7: Befehle für das Touch-Panel											nach Reset		
Befehl	Codes		Anmerkung										
Touch: Bereiche definieren													
Touch-Taste definieren (Taste ist gedrückt solange der Touch berührt wird)	ESC	A	T	x1	y1	x2	y2	down Code	up Code	Text ... NUL	T: Der Bereich von x1,y1 nach x2,y2 wird als Taste definiert. U: Das Bild Nr. n1 wird nach x1,y2 geladen und als Taste definiert. 'down Code':(1-255) Rückgabe/Touchmakro beim Drücken der Taste. 'up Code': (1-255) Rückgabe/Touchmakro beim Loslassen der Taste. (down-/up-Code = 0 drücken/loslassen wird nicht gemeldet). 'Text': Das erste Zeichen bestimmt die Ausrichtung des Textes (C=zentriert L=linksbündig R=rechtsbündig) danach folgt eine Zeichenkette die mit dem akt. Touch-Font in der Taste plaziert wird. Mehrzeilige Texte werden mit dem Zeichen ' ' (\$7C, dez: 124) getrennt;		
Touch-Schalter definieren (Zustand der Schalter toggelt nach jeder Berührung)			U	x1	y1	n1	down Code	up Code	Text ... NUL	NUL	'K': Der Bereich von x1,y1 nach x2,y2 wird als Schalter definiert. 'J': Das Bild n1 wird nach x1,y2 geladen und als Schalter definiert. 'down Code': (1-255) Rückgabe/Touchmakro beim Einschalten. 'up Code': (1-255) Rückgabe/Touchmakro beim Ausschalten. (down-/up-Code = 0 Ein-/Ausschalten wird nicht gemeldet). 'Text': Das erste Zeichen bestimmt die Ausrichtung des Textes (C=zentriert L=linksbündig R=rechtsbündig) danach folgt eine Zeichenkette die mit dem akt. Touch-Font in der Taste plaziert wird. Mehrzeilige Texte werden mit dem Zeichen ' ' (\$7C, dez: 124) getrennt;		
			K	x1	y1	x2	y2	down Code	up Code	Text ... NUL	NUL	'K': Der Bereich von x1,y1 nach x2,y2 wird als Schalter definiert. 'J': Das Bild n1 wird nach x1,y2 geladen und als Schalter definiert. 'down Code': (1-255) Rückgabe/Touchmakro beim Einschalten. 'up Code': (1-255) Rückgabe/Touchmakro beim Ausschalten. (down-/up-Code = 0 Ein-/Ausschalten wird nicht gemeldet). 'Text': Das erste Zeichen bestimmt die Ausrichtung des Textes (C=zentriert L=linksbündig R=rechtsbündig) danach folgt eine Zeichenkette die mit dem akt. Touch-Font in der Taste plaziert wird. Mehrzeilige Texte werden mit dem Zeichen ' ' (\$7C, dez: 124) getrennt;	
			J	x1	y1	n1	down Code	up Code	Text ... NUL	NUL	'J': Das Bild n1 wird nach x1,y2 geladen und als Schalter definiert. 'down Code': (1-255) Rückgabe/Touchmakro beim Einschalten. 'up Code': (1-255) Rückgabe/Touchmakro beim Ausschalten. (down-/up-Code = 0 Ein-/Ausschalten wird nicht gemeldet). 'Text': Das erste Zeichen bestimmt die Ausrichtung des Textes (C=zentriert L=linksbündig R=rechtsbündig) danach folgt eine Zeichenkette die mit dem akt. Touch-Font in der Taste plaziert wird. Mehrzeilige Texte werden mit dem Zeichen ' ' (\$7C, dez: 124) getrennt;		
Touch-Taste mit Menüfunktion definieren	ESC	A	M	x1	y1	x2	y2	down Code	up Code	mnuü Code	Text ... NUL	Der Bereich x1,y1 nach x2,y2 wird als Menü-Taste definiert. 'down Code':(1-255)Rückgabe/Touchmakro beim Drücken. 'up Code':(1-255) Rückgabe/Touchmakro beim Menü-Abbruch 'mnu Code':(1-255) Rückgabe/Menu Makro+(EintragsNr-1) nach Auswahl eines Menü-Eintrages. (down-/up-Code=0:Aktivieren/Abbruch wird nicht gemeldet. 'Text':= Zeichenkette mit den Tastentext und den Menüeinträgen. Das erste Zeichen bestimmt die Richtung in der das Menü aufklappt (R=rechts L=links O=oben U=Unten). Das zweite Zeichen bestimmt die Ausrichtung des Touchtasten-Textes (C=zentriert L=linksbündig R=rechtsbündig). Die Menü-Einträge sind durch Zeichen ' ' (\$7C,dez:124) getrennt. z.B. "UCTaste Eintrag1 Eintrag2 Eintrag3" Der Tastentext wird mit dem akt. Touchfont und die Menü-Einträge mit dem akt. Menüfont gezeichnet. Der Hintergrund des Menüs wird automatisch gesichert.	
Zeichenbereich definieren	ESC	A	D	x1	y1	x2	y2	n1	Ein Zeichenbereich wird definiert. Innerhalb der Eck-Koordinaten x1,y1 und x2,y2 kann dann mit der Strichstärke n1 gezeichnet werden.				
Freien Touchbereich def. Bar per Touch einstellbar	ESC	A	H	x1	y1	x2	y2	Ein frei benutzbarer Touchbereich wird definiert. Touchaktionen (down, up und drag) innerhalb der Eck-Koordinaten x1,y1 und x2,y2 werden gesendet.					
	ESC	A	B	nr	Der Bargraph mit der Nr. n1 wird zur Eingabe per Touchpanel definiert.								
Touch: Einstellungen													
Touch-Rahmen Form	ESC	A	E	n1	mit n1 wird der Rahmentyp für die Darstellung von Touch-Tasten/Schaltern eingestellt						1		
Touch-Tasten Reaktion			I	n1	automatisches Invertieren beim Berühren der Touch-Taste: n1=0=AUS; n1=1=EIN;						1		
Touch-Taste Invertieren			S	n1	Summer piepst kurz beim Berühren einer Touch-Taste: n1=0=AUS; n1=1=EIN						1		
Touch-Schalter abfragen			N	Code	Die Touch-Taste mit dem zugeordnetem Return-Code wird manuell Invertiert								0
Touch-Schalter einstellen			X	Code	Zustand des Schalters (Aus=0; Ein=1) wird in den Sendepuffer gestellt.								
			P	Code n1	Zustand des Schalters wird per Befehl geändert n1=0=Aus; n1=1=Ein. Innerhalb einer Gruppe ist immer nur 1 Schalter aktiv, alle anderen werden deaktiviert								
Radiogroup für Schalter			R	nr	nr=0: neu definierte Schalter gehören keiner Gruppe an. nr=1..255: neu definierte Schalter gehören der Gruppe mit der Nummer nr an. Bei Schalter in einer Gruppe wird nur der downcode beachtet, der upcode wird ignoriert								
Radiogroup abfragen			G	nr	(ab V1.3) der downcode des aktivierten Schalters aus der Radiogroup mit der Nummer nr wird in den Sendepuffer gestellt.								
Touch-Bereich Löschen			L	Code n1	Der Touchbereich mit dem Return-Code (Code=0: alle Touchbereiche) wird aus der Touchabfrage entfernt. Mit n1=0 bleibt der Bereich am Display sichtbar, mit n1=1 wird der Bereich gelöscht.								
Barwert automatisch in den Sendepuffer stellen			V	x1 y1 n1	Touchbereich der die Koordinaten x1,y1 umschließt aus der Touchabfrage entfernen n1=0: Bereich bleibt sichtbar; n1=1: Bereich löschen das automatische Speichern eines neuen Bargraphwertes per Toucheingabe wird n1=0:deaktiviert; n1=1:neuer Wert wird nach dem Einstellen in den Sendepuffer gestellt (ab V1.2) n1=2: jede Änderung landet während des Einstellens im Sendepuffer.								1
Touch-Abfrage Ein/Aus	A	n1	Touchabfrage wird n1=0:deaktiviert; n1=1:aktiviert;								1		
Touch: Beschriftungs-Font													
Beschriftungs Font	ESC	A	F	nr	Font mit der Nummer nr (0..15) für Touchtastenbeschriftung einstellen						0		
Beschriftungs-Zoomfaktor			Z	n1 n2	n1 = X-Zoomfaktor (1x..4x); n2 = Y-Zoomfaktor (1x..4x)						1,1		
zus. Zeilenabstand			Y	n1	zwischen zwei Textzeilen n1 Pixel (0..15) als zusätzlichen Zeilenabstand einfügen								
Beschriftungs-Winkel			W	n1	Text-Ausgabewinkel: n1=0: 0°; n1=1: 90°;						0		

Antworten des EA eDIP240-7 über die serielle Schnittstelle						
Kennung	anz	daten			Anmerkung	
automatische Antworten						
ESC	A	1	code			Antwort vom Analogen Touchpanel wenn eine Taste/Schalter gedrückt wurde. code = down oder up Code der Taste/Schalter. Es wird nur gesendet wenn kein Touch-Makro mit der Nr. code definiert ist !
ESC	N	1	code			Nach dem Auswählen eines Menüeintrages per Touch wird der ausgewählte Menüeintrag code gesendet. Es wird nur gesendet wenn kein Menü-Makro mit der Nr. code definiert ist !
ESC	B	2	nr	wert		Nach dem Einstellen eines Bargraph per Touch wird der aktuelle wert des Bars mit der nr gesendet. Barwert Senden muß aktiviert sein siehe Befehl 'ESC A Q n1'.
ESC	T	0				Falls das automatische Öffnen eines Touchmenüs deaktiviert ist (siehe Befehl 'ESC N T n1'), so wird diese Anforderung an den Hostrechner gesendet. Dieser kann dann das Touchmenü mit dem Befehl 'ESC N T 2' öffnen.
ESC	H	3	typ	x1	y1	Bei einem freien Touchbereich-Ereignis wird folgendes gesendet: typ=0 ist Loslassen; typ=1 ist Berühren; typ=2 ist Draggen innerhalb des freien Touchbereiches an den Koordinaten x1,y1
Antworten nur nach Anforderung per Befehl						
ESC	N	1	nr			Nach dem Befehl 'ESC N S' wird der aktuell ausgewählte Menüeintrag gesendet. nr=0: kein Menüeintrag ist ausgewählt.
ESC	B	2	nr	wert		Nach dem Befehl 'ESC B S n1' wird der aktuelle Wert Bars mit der Nr. nr gesendet.
ESC	X	2	code	wert		Nach dem Befehl 'ESC A X code' wird der aktuelle Zustand des Touch-Schalters mit dem Return-Code code gesendet. wert = 0 oder 1
ESC	G	2	nr	code		(ab V1.3) Nach dem Befehl 'ESC A G nr' wird der code des aktiven Touch-Schalters von der Radiogroup nr gesendet.
ESC	V	anz	Zeichenkette...			Nach dem Befehl 'ESC S V' wird die Version der eDIP-Firmware als Zeichenkette gesendet. z.B "EA eDIP240-7 V1.3 Rev.B TP+"
ESC	I	anz	X-Pixel, Y-Pixel, Version, Touchinfo, CRC-ROM, CRC-ROMsoll, EEP in KB, (abV1.4) CRC-EEP, CRC-EEPsoll, EEPanz			(V1.3: anz=14; ab V1.4: anz = 21) Nach dem Befehl 'ESC S I' werden interne Informationen vom eDIP gesendet (16-Bit integer Werte LO- HI-Byte) Version: LO-Byte = Versionsnr. Software; HI-Byte = Hardware revisionsbuchstabe Touchinfo: LO-Byte = '- +' X-Richtung erkannt; HI-Byte = '- +' Y-Richtung erkannt EEPanz: Anzahl benutzter Bytes im EEPROM (3 Byte: LO-, MID- HI-Byte)
Antworten ohne Längenangabe (anz)						
ESC	U	L	x1	y1	BLH-Bilddaten...	
Nach dem Befehl 'ESC UH....' wird ein Hardcopy gesendet. x1,y1 = Startkoordinaten des Hardcopies (Linke obere Ecke) BLH-Bilddaten: 2 Byte: breite, höhe (in Pixel) + anzahl Bytes Bilddaten anzahl = ((breite+7)/8)*höhe						

TERMINAL-BETRIEB

Das Display enthält eine integrierte Terminalfunktion. Nach dem Einschalten blinkt ein Cursor in der ersten Zeile und das Display ist empfangsbereit. Alle ankommenden Zeichen werden als ASCII's dargestellt (Ausnahme: CR,LF,FF,ESC,'#'). Voraussetzung dafür ist ein funktionierender Portokollrahmen (Seiten 8 und 9) oder ein abgeschaltetes Protokoll (Lötbrücke J2 schliessen, Seiten 8 und 20).

Der Zeilenvorschub erfolgt automatisch oder durch das Zeichen 'LF'. Ist die letzte Zeile voll, scrollt der Terminalinhalt nach oben. Beim Zeichen 'FF' wird das Terminal gelöscht.

Das Zeichen '#' wird als Escape-Zeichen benutzt und ist somit nicht direkt im Terminal darstellbar. Soll das Zeichen '#' im Terminal ausgegeben werden, so muß es doppelt gesendet werden '##'. Das Terminal besitzt eine eigene Ebene zur Darstellung und ist somit völlig unabhängig von den Grafikausgaben. Wird z.B. der Grafikbildschirm mit 'ESC DL' gelöscht, so beeinflusst das nicht den Inhalt des Terminalfensters.

Der Terminalfont ist fest im ROM vorhanden und kann auch für Grafikausgaben 'ESC Z...' verwendet werden (FONT nr=0 einstellen).

	+ 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)
Upper																	
\$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	{		}	~	Δ
\$80 (dez: 128)		€	ü	é	û	ä	à	ç	ë	è	ì	í	î	ï	ä	Å	
\$90 (dez: 144)		É	æ	Æ	ö	ö	û	ù	ÿ	ö	ü	ç	£	¥	β	ƒ	
\$A0 (dez: 160)		á	í	ó	ú	ñ	ñ	ë	ë	í	í	½	¼	i	«	»	
\$B0 (dez: 176)																	
\$C0 (dez: 192)																	
\$D0 (dez: 208)																	
\$E0 (dez: 224)		α	β	Γ	Π	Σ	σ	ι	τ	ϕ	θ	η	δ	φ	ε	π	
\$F0 (dez: 240)		≡	±	≥	≤	ρ	∫	÷	∞	°	*	.	√	n	z	≅	-

Terminal-Font (Font 0): 8x8 monospaced

BEFEHLE ÜBER DIE SERIELLE SCHNITTSTELLE SENDEN

Das eDIP240-7 läßt sich über diverse eingebaute Befehle programmieren. Jeder Befehl beginnt mit ESCAPE oder RAUTE gefolgt von einem oder zwei Befehlsbuchstaben und einigen Parametern. Es gibt somit zwei Möglichkeiten Befehle zu senden:

1. ASCII-Modus

- Das Escape-Zeichen entspricht dem Zeichen '#' (hex: \$23, dez: 35).
- Die Befehlsbuchstaben folgen direkt im Anschluss an das '#' Zeichen.
- Die Parameter werden im Klartext (mehrere ASCII Ziffern) mit einem nachfolgenden Trennzeichen (z.B. das Komma ',') gesendet - auch hinter dem letzten Parameter z.B.: #GD0,0,239,127,
- Zeichenketten (Texte) werden direkt ohne Anführungsstrichen geschrieben und mit CR (hex: \$0D), oder LF (hex: \$0A) abgeschlossen.

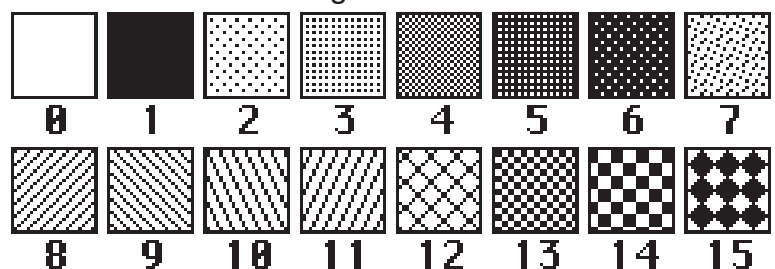
2. Binär-Modus

- Das Escape-Zeichen entspricht dem Zeichen ESC (hex: \$1B, dez: 27).
- Die Befehlsbuchstaben werden direkt gesendet.
- Die Koodinaten x, y und alle anderen Parameter werden als 8-Bit Binärwert (1 Byte) gesendet.
- Zeichenketten (Texte) werden mit CR (hex: \$0D), LF (hex: \$0A) oder NUL (hex: \$00) abgeschlossen.

Im Binär-Modus dürfen keine Trennzeichen z.B. Leerzeichen oder Kommas verwendet werden. Die Befehle benötigen auch **kein Abschlussbyte** wie z.B Carrige Return (außer Zeichenkette: \$00).

FÜLLMUSTER

Bei diversen Befehlen kann als Parameter ein Mustertyp eingestellt werden. So können z.B. rechteckige Bereiche und Bargraphs mit unterschiedlichen Mustern gefüllt werden. Dabei stehen 16 interne Füllmuster zur Verfügung.



MAKRO PROGRAMMIERUNG

Einzelne oder mehrere Befehlsfolgen können als sog. Makros zusammengefasst und im EEPROM fest abgespeichert werden. Diese können dann mit den Befehlen *Makro ausführe* gestartet werden. Es gibt verschiedene Makrotypen (Compileranweisungen sind grün geschrieben):

Normal Makro (0..255) *Makro*:

Start per Befehl 'ESC MN xx' über serielle Schnittstelle oder von einem anderen Makro aus.

Es können auch mehrere hintereinander liegende Makros automatisch zyklisch aufgerufen werden (Movie, sich drehende Sanduhr, mehrseitiger Hilfetext). Diese automatischen Makros werden solange abgearbeitet bis ein Befehl über die Schnittstelle empfangen wird, oder ein Touchmakro mit entsprechendem Return-Code ausgelöst wird.

Ausserdem werden diese Makros von Makro-Prozessen (ab V1.1) in definierten Intervallen aufgerufen. Makro-Prozesse werden nicht durch Empfang von Befehlen von der Schnittstelle oder von ausgelösten Touchmakros unterbrochen.

Touch Makro (1..255) *TouchMakro*:

Start beim Berühren/Loslassen eines Touchfeldes (nur bei Versionen mit Touch Panel TP) oder per Befehl 'ESC MT xx'.

Menü Makro (1..255) *MenuMakro*:

Start bei Auswahl eines Menüeintrages oder per Befehl 'ESC MM xx'.

Power-On-Makro *PowerOnMakro*:

Start nach dem Einschalten Power-On. Hier kann man zB. den Cursor abschalten und einen Startbildschirm definieren.

Reset-Makro *ResetMakro*:

Start nach einem externen Reset oder nach einem Spannungseinbruch unter 4,7V (VDD-VSS).

Watchdog-Makro *WatchdogMakro*:

Start nach einem Fehlerfall (z.B. Absturz).

Brown-Out-Makro *BrownOutMakro*:

Start nach einem Spannungseinbruch <4V.

Achtung: Wird im Power-On-, Reset- oder Watchdog-Makro eine Endlosschleife programmiert, ist das Display nicht mehr ansprechbar. In diesen Fall muss die Ausführung des Power-On Makros unterdrückt werden. Das erreicht man durch die Beschaltung von DPOM:

PowerOff - Pin 13 (DPOM) auf GND legen - PowerOn -Pin 13 wieder öffnen.

SCHREIBSCHUTZ FÜR MAKROPROGRAMMIERUNG UND FONTS

Ein VDD-Pegel am Pin 19 (EEP_WP) verhindert ein versehentliches Überschreiben der Makros, Bilder und Fonts im EEPROM (in jedem Fall empfohlen !).

SPEICHERERWEITERUNG

Der interne EEPROM Speicher beträgt 32kB. In der Regel steht dadurch ausreichend Platz für viele Bilder und Makros zur Verfügung. Wenn jedoch sehr viele Bilder (vor allem Vollbilder) abgelegt werden sollen, kann es erforderlich sein Speicher nachzurüsten. Möglich ist eine Verdopplung durch direktes Einlöten eines SMD-EEPROM's aus der Serie 24C256 auf dem eDIP (siehe S.20 Abmessungszeichnung U12).

Alternativ kann der Anschluß auch extern über die Pins 17, 18 und 19 erfolgen (das EEPROM muss auf die I2C-Adresse \$A6 eingestellt sein).

BILDER IM EEPROM ABGELEGT

Um die Übertragungszeiten der Schnittstelle zu verkürzen, oder auch um Speicherplatz im Prozessorsystem zu sparen, können bis zu 256 Bilder im internen EEPROM abgelegt werden. Der Aufruf erfolgt über den Befehl "ESC U I" oder aus einem Makro heraus. Verwendet werden können alle Bilder im Windows BMP-Format (nur monochrome Bilder). Die Erstellung und Bearbeitung erfolgt über Standardsoftware wie z.B. Windows Paint oder Photoshop (nur schwarz/weiss = 1 Bit).

ERSTELLEN INDIVIDUELLER MAKROS UND BILDER

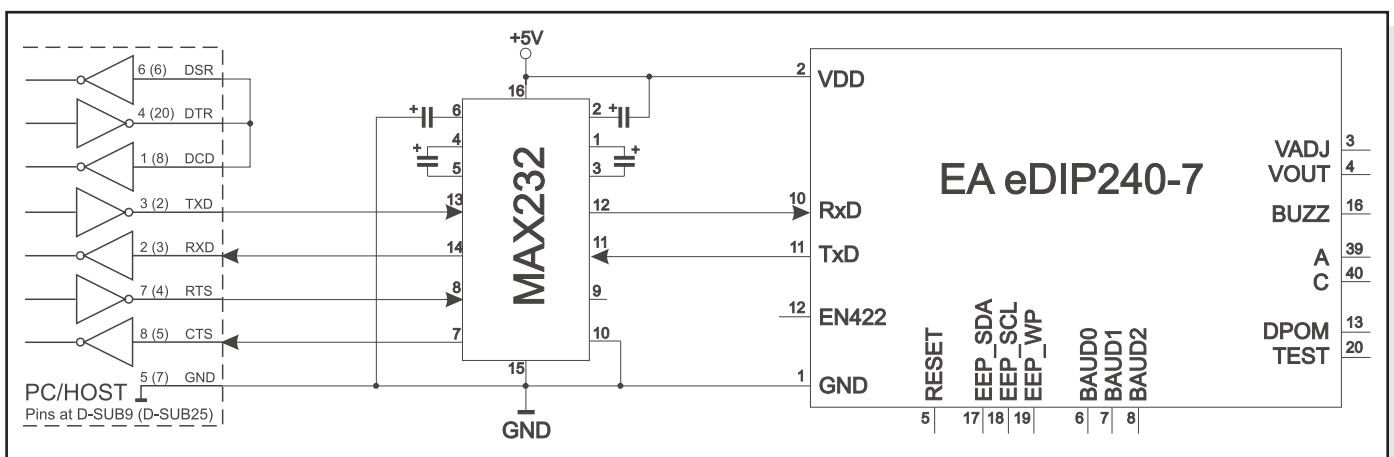
Um nun Ihre speziellen Makros erstellen zu können, benötigen Sie folgende Hilfsmittel:

- um das Display an den PC anschliessen zu können benötigen Sie den als Zubehör erhältlichen USB-Programmer EA 9777-1USB oder einen selbstgebauten Adapter mit Pegelwandler MAX232 (Applikationsbeispiel unten).
- die Software ELECTRONIC ASSMBLY LCD-Tools[®]; sie enthält einen Kit-Editor, Kit-Compiler, Simulator, sowie Beispiele und Fonts (für PC-Win)
- einen PC mit USB oder serieller Schnittstelle COM

Um eine Befehlsfolge als Makro zu definieren, werden alle Befehle auf dem PC in eine Datei z.B. DEMO.KMC geschrieben. Hier bestimmen Sie, welche Zeichensätze eingebunden werden und in welchen Makros welche Befehlsfolgen stehen sollen.

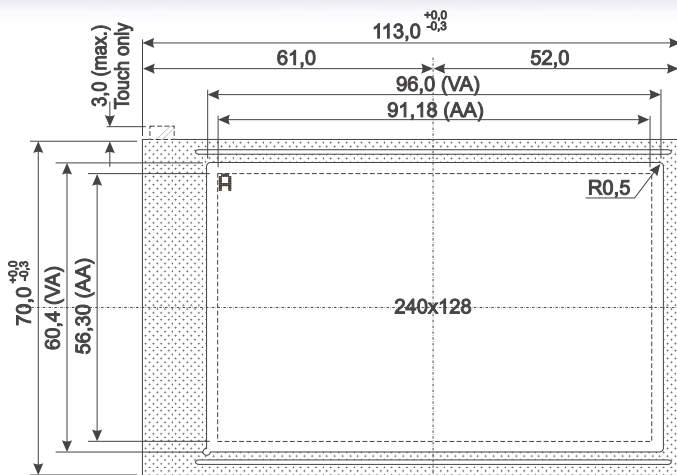
Sind die Makros über den Kit-Editor definiert, startet man über F5 den Kit-Compiler. Dieser erzeugt eine Datei DEMO.EEP, welcher das Ergebnis in einem Simulatorfenster (virtuelles Display) sofort anzeigt. Ist auch ein Programmer EA 9777-1USB angeschlossen, oder das Display über einen MAX232 an den PC angeschlossen, dann wird diese Datei automatisch in das EEPROM des Displays gebrannt. Der Kit-Compiler erkennt das Display mit und ohne eingeschaltetem Small-Protokoll.

Der Programmiervorgang selbst dauert nur wenige Sekunden und sofort danach können die selbstdefinierten Makros und Bilder auch im Display genutzt werden. Eine ausführliche Beschreibung zur Programmierung der Makros finden Sie zusammen mit Beispielen in der Hilfefunktion der ELECTRONIC ASSEMBLY LCD-Tools[®] Software.



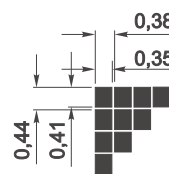
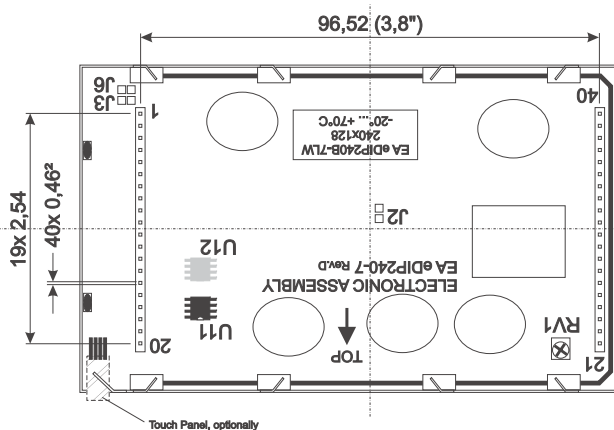
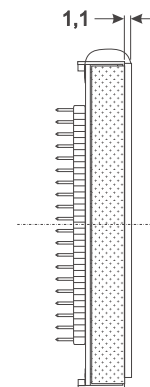
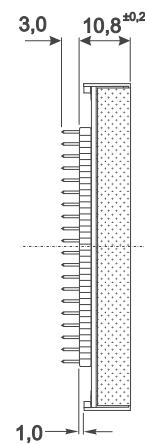
Adapter zum Selberbauen für direkten PC-Anschluss

ABMESSUNGEN



ohne Touchpanel

mit Touchpanel



- J2: Small Protokoll deaktivieren
- J6: Verbindung Metallrahmen und GND (spezielle ESD / EMV Anforderungen)
- J3: externe Kontrasteinstellung

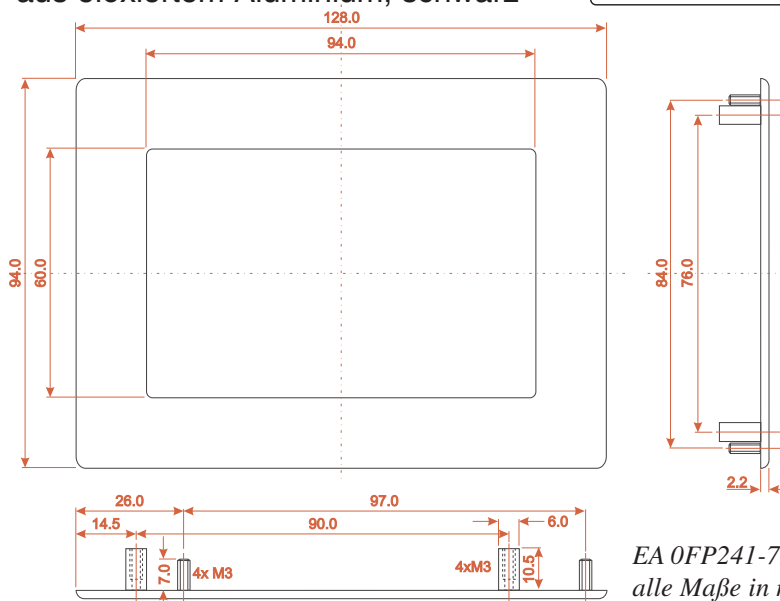
Hinweis:

LC-Displays sind generell nicht geeignet für Wellen- oder Reflowlötlötung. Temperaturen über 90°C können bleibende Schäden hinterlassen.

alle Maße in mm

EINBAUBLENDE (ZUBEHÖR)

aus eloxiertem Aluminium, schwarz



EA 0FP241-7SW:
alle Maße in mm

Hinweise zur Handhabung und zum Betrieb

- Zur elektrischen Zerstörung des Moduls kann führen: Verpolung oder Überspannung der Stromversorgung, Überspannung oder Verpolung bzw. statische Entladung an den Eingängen, Kurzschließen der Ausgänge.
- Vor dem Abstecken des Moduls muß unbedingt die Stromversorgung abgeschaltet sein. Ebenso müssen alle Eingänge stromlos sein.
- Das Display und der Touchscreen bestehen aus Kunststoff und dürfen nicht mit harten Gegenständen in Berührung kommen. Die Oberflächen können mit einem weichen Tuch ohne Verwendung von Lösungsmitteln gereinigt werden.
- Das Modul ist ausschließlich für den Betrieb innerhalb von Gebäuden konzipiert. Für den Betrieb im Freien müssen zusätzliche Vorkehrungen getroffen werden. Der maximale Temperaturbereich darf nicht überschritten werden. Bei Einsatz in feuchter Umgebung kann es zu Funktionsstörungen und zum Ausfall des Moduls kommen. Das Display ist vor direkter Sonneneinstrahlung zu schützen.

ELECTRONIC ASSEMBLY

making things easy