Stand 04.2022

EA eDIP160-7

Intelligentes HMI 160x104 , RS-232, SPI, I2C

EA eDIP160W-7LWTP

EA eDIP160B-7LWTP

Abmessung: 81,5x67,5x13,6mm

TECHNISCHE DATEN

* LCD-GRAFIKDISPLAY MIT DIVERSEN GRAFIKFUNKTIONEN
* 3 VERSCHIEDENE INTERFACES ONBOARD: RS-232, I²C-BUS ODER SPI-BUS
* 160x104 ODER 104X160 PIXEL IN ALLEN RICHTUNGEN EINBAUBAR
* WEISSE LED-BELEUCHTUNG BLAU NEGATIV ODER
* SCHWARZ-WEISS POSITIV, FSTN-TECHNIK
* 8 EINGEBAUTE FONTS
* FONT ZOOM VON ca. 2mm BIS ZU ca. 40mm, auch um 90° GEDREHT
* VERSORGUNG 3,3V/190mA/12mA ... 5V/125mA/20mA (MIT/OHNE LED BELEUCHTUNG)
* POWER-DOWN-MODE 25 µA, MIT WAKEUP PER TOUCH ODER I²C
* 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 (64kB 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

160x104 DOTS, WEISSE LED-BELEUCHTUNG, BLAU NEGATIV	**EA eDIP160B-7LW**
WIEVOR, JEDOCH MITTOUCH PANEL	**EA eDIP160B-7LWTP**
160x104 DOTS, WEISSE LED-BELEUCHTUNG, POSITIV MODE, FSTN	**EA eDIP160W-7LW**
WIEVOR, JEDOCH MITTOUCH PANEL	**EA eDIP160W-7LWTP**

STARTERKITS

ENTHÄLT EAeDIP160B-7LWTP, EVALUATION BOARD MIT USB FÜR DIREKTE PC-VERBINDUNG UND INTERFACE BOARDS FÜR ANBINDUNG AN DAS HOST-SYSTEM	**EA EVALeDIP160B**
WIEVOR, JEDOCH MIT EAeDIP160W-7LWTP	**EA EVALeDIP160W**

ZUBEHÖR

EINBAUBLENDE SCHWARZ, ELOXIERTES ALUMINIUM	**EA 0FP161-7SW**
BUCHSENLEISTE 1x20, 4.5 mm HOCH (1 STÜCK)	**EA B254-20**

Documentation of revision				
Date	**Type**	**Old**	**New**	**Reason / Description**
October, 2010	0.1			preliminary Version
August, 2011	1.0			first release

INHALT

ALLGEMEINES

EA eDIP160-7 ist ein 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 3,3V...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 +3,3V bis 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.

ANALOGES TOUCH PANEL

Die Versionen EA eDIP160B-7LWTP und EA eDIP160W-7LWTP sind mit einem integrierten Touch Panel ausgerüstet. Durch Berühren des Displays können hier Eingaben gemacht und Einstellungen per Menü oder Bargraph 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- UNDW-TYPEN

Alle Displays in blau-weiß (B) und schwarz-weiß (W) sind mit einer modernen und stromsparenden LED-Beleuchung ausgestattet. Während das Schwarz-Weiß-Display 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-Version. 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.

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 PROGRAMMER 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).

RS-232 INTERFACE

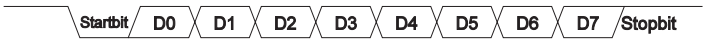
Wird das Display wie unten gezeigt beschaltet, so ist das RS-232 Interface ausgewählt. Die Pinbelegung ist in der Tabelle rechts angegeben. Die Leitungen RxD und TxD führen CMOS-Pegel (VDD) zur direkten Anbindung an z.B. einen Mikrokontoller.

Pinout eDIP160-7: RS-232/RS-485 mode

Pin	Symbol	In/Out	Function	Pin	Symbol	In/Out	Function
1	GND		Ground Potential for logic (0V)	21	GND		Ground (0V)
2	VDD		Power supply for logic (+3,3V..5V)	22	VDD		Power supply (+3,3..5V)
3	NC		do not connect	23	NC		do not connect
4	NC		do not connect	24	NC		
5	RESET	In	L: Reset	25	IN8 / OUT1		
6	BAUD0	In	Baud Rate 0	26	IN7 / OUT2		8 digital inputs (internal 20k..50k pullup)
7	BAUD1	In	Baud Rate 1	27	IN6 / OUT3		
8	BAUD2	In	Baud Rate 2	28	IN5 / OUT4		
9	ADR0	In	Address 0 for RS-485	29	IN4 / OUT5		alternativ up to 8 digital outputs maximum current: IOL = IOH = 10mA
10	RxD	In	Receive Data	30	IN3 / OUT6		
11	TxD	Out	Transmit Data	31	IN2 / OUT7		
12	EN485	Out	Transmit Enable for RS-485 driver	32	IN1 / OUT8		
13	WUP	In	L: (Power-On) disable Power-On-Macro L: Wakeup from Powerdownmode	33	NC		do not connect
14	ADR1	In	Address 1 for RS-485	34	NC		
15	ADR2	In	Address 2 for RS-485	35	NC		
16	BUZZ	Out	H: Buzzer output (L: Buzzer off)	36	NC		
17	DPROT	In	L: Disable Smallprotokoll do not connect for normal operation	37	NC		
18	PWR	Out	L: Normal Operation H: Powerdownmode	38	NC		
19	NC		do not connect	39	NC		
20	TEST SBUF	IN Out	open-drain with internal pullup 20k..50k IN (Power-On) L: Testmode OUT L: data in sendbuffer	40	NC		

BAUDRATEN

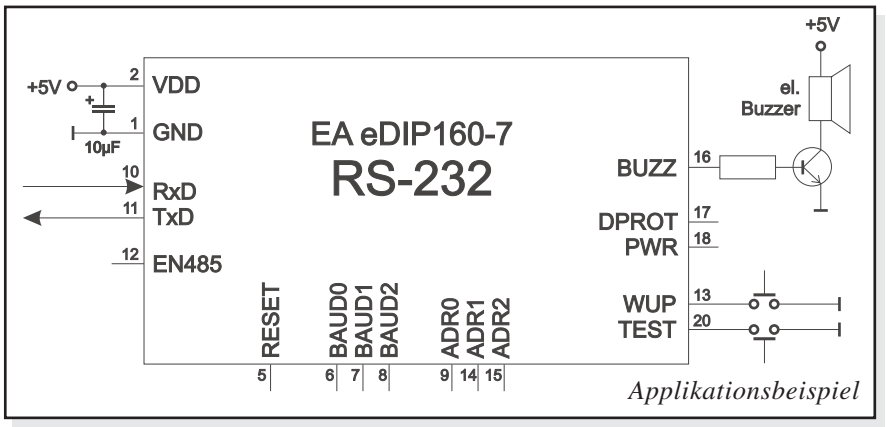
Die Baudrate wird über die Pins 6, 7 und 8 (Baud0..2) eingestellt. Das Datenformat ist fest eingestellt auf 8 Datenbits, 1 Stopbit, keine Parität.



Handshakeleitungen RTS/CTS sind nicht erforderlich. Die notwendige Steuerung wird von dem eingebauten Software-Protokoll übernommen.

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



Applikationsbeispiel

Hinweis:

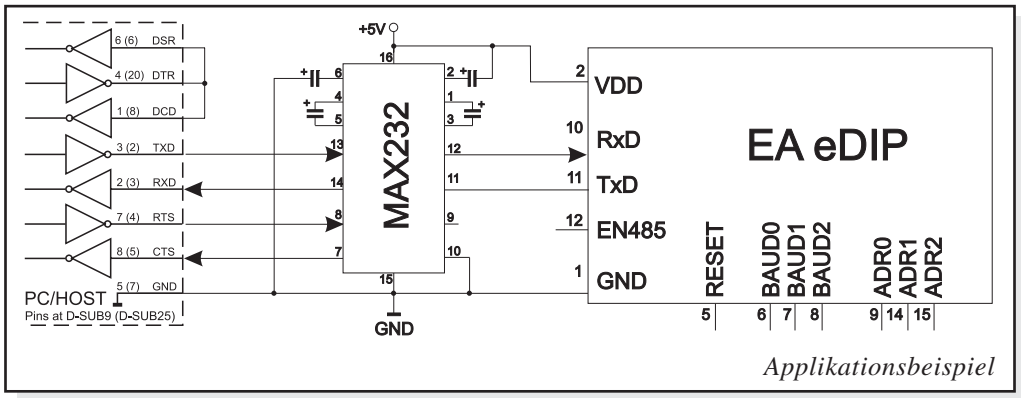
Die Pins BAUD0..2, ADR0..2, WUP und TEST/ SBUF haben einen internen Pull-UP, deshalb ist nur der LO-Pegel (0=GND) aktiv anzulegen. Für Hi-Pegel sind diese Pins offen zu lassen.

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.

APPLIKATIONSBEISPIEL „ECHTES“ RS-232 INTERFACE

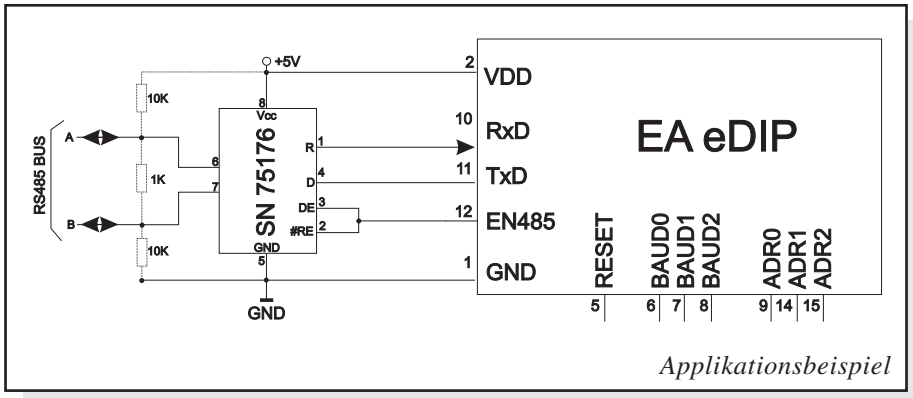
Das eDIP ist für den direkten Anschluss an eine RS-232 Schnittstelle mit CMOS-Pegeln (VDD) geeignet. Steht jedoch nur eine Schnittstelle mit ±12V Pegeln, so ist ein externer Pegelwandler erforderlich.



Applikationsbeispiel

APPLIKATIONSBEISPIEL: RS-485 INTERFACE

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



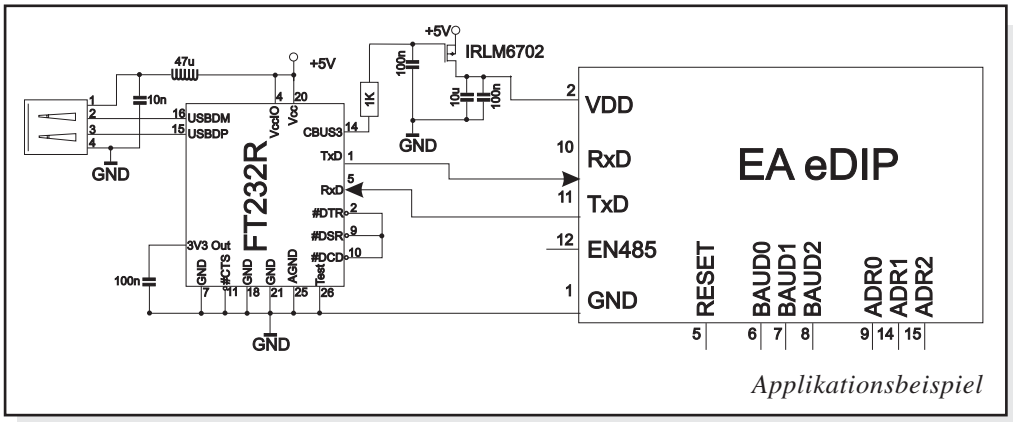
Applikationsbeispiel

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 (eDIP extern auf Adresse 0 setzen)*

APPLIKATIONSBEISPIEL: USB ANSCHLUSS

Mit einem externen Umsetzer (z.B. FTZ232R) von FTDI kann das eDIP an einen USB-Bus angeschlossen werden. Virtuelle-COM-Port Treiber gibt es für viele Betriebssyteme auf der FTDI Homepagehttp://www.ftdichip.com/drivers/vcp.htm.



Applikationsbeispiel

SPI INTERFACE

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.

Pinout eDIP160-7: SPI mode

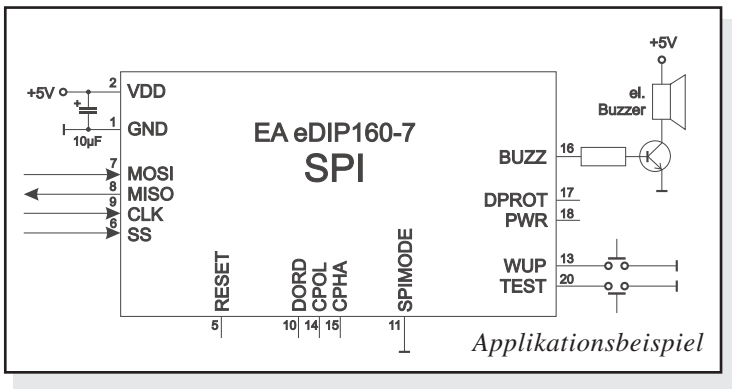
Pin	Symbol	In/Out	Function	Pin	Symbol	In/Out	Function
1	GND		Ground Potential for logic (0V)	21	GND		Ground (0V)
2	VDD		Power supply for logic (+3,3V..5V)	22	VDD		Power supply (+3,3..5V)
3	NC		do not connect	23	NC		do not connect
4	NC		do not connect	24	NC		do not connect
5	RESET	In	L: Reset	25	IN8 / OUT1		
6	SS	In	Slave Select	26	IN7 / OUT2		8 digital inputs
7	MOSI	In	Serial In	27	IN6 / OUT3		(internal 20k..50k pullup)
8	MISO	Out	Serial Out	28	IN5 / OUT4		
9	CLK	In	Shift Clock	29	IN4 / OUT5		alternativ up to 8 digital outputs
10	DORD	In	Data Order (0=MSB first; 1=LSB first)	30	IN3 / OUT6		maximum current:
11	SPIMOD	In	connect to GND for SPI interface	31	IN2 / OUT7		IOL = IOH = 10mA
12	NC		do not connect	32	IN1 / OUT8		
13	WUP	In	L: (Power-On) disable Power-On-Macro L: Wakeup from Powerdownmode	33	NC		do not connect
14	CPOL	In	Clock Polarity (0=LO 1=HI when idle)	34	NC		do not connect
15	CPHA	In	Clock Phase sample 0=1st;1=2nd edge	35	NC		do not connect
16	BUZZ	Out	H: Buzzer output (L: Buzzer off)	36	NC		do not connect
17	DPROT	In	L: Disable Smallprotokoll do not connect for normal operation	37	NC		do not connect
18	PWR	Out	L: Normal Operation H: Powerdownmode	38	NC		do not connect
19	NC		do not connect	39	NC		do not connect
20	TEST SBUF	IN Out	open-drain with internal pullup 20k..50k IN (Power-On) L: Testmode OUT L: data in sendbuffer	40	NC		do not connect

Hinweis:
Die Pins DORD, CPOL, CPHA, WUP 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.

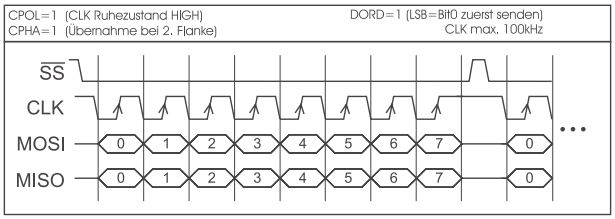
DATENÜBERTRAGUNG SPI

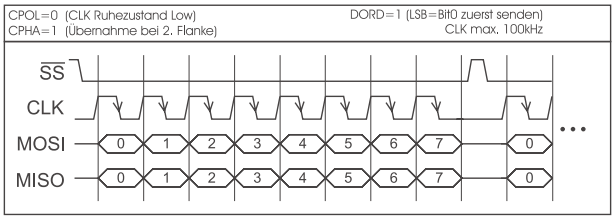
Eine Datenübertragung zum eDIP ist bis zu 100 kHz Nonstop möglich. Wenn jedoch zwischen den einzelnen Bytes während der Übertragung Pausen von jeweils min. 100 µs eingehalten werden, kann ein Byte mit bis zu 3 MHz übertragen werden.

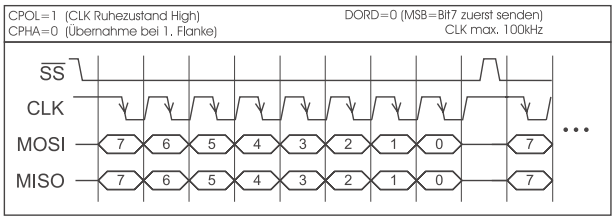
Um Daten vom eDIP zu Lesen (z.B. das ACK-Byte) muss ein Dummy-Byte (z.B. 0xFF) gesendet werden. Das eDIP benötigt eine bestimmte Zeit um die Daten bereit zu stellen; deshalb muss vor jedem zu lesenden Byte zusätzlich mindestens 6µs gewartet werden (keine Aktivität auf der CLK Leitung). Dies gilt auch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

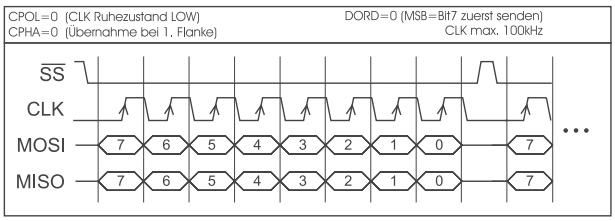


Applikationsbeispiel









I²C-BUS INTERFACE

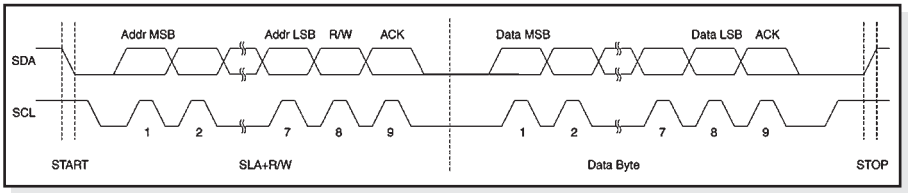
Eine Beschaltung des Displays wie unten ermöglicht den direkten Betrieb an einem I²C-Bus.
Am Display kann zwischen 8 unterschiedlichen Basisadressen und 8 verschiedenen Slave-Adressen ausgewählt werden.

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 eDIP160-7: I2C mode

Pin	Symbol	In/Out	Function	Pin	Symbol	In/Out	Function
1	GND		Ground Potential for logic (0V)	21	GND		Ground (0V)
2	VDD		Power supply for logic (+5V)	22	VDD		Power supply (+3,3..5V)
3	NC		do not connect	23	NC		do not connect
4	NC		do not connect	24	NC		
5	RESET	In	L: Reset	25	IN8 / OUT1		
6	BA0	In	Basic Address 0	26	IN7 / OUT2		8 digital inputs (internal 20k..50k pullup)
7	BA1	In	Basic Address 1	27	IN6 / OUT3		
8	SA0	In	Slave Address 0	28	IN5 / OUT4		
9	SA1	In	Slave Address 1	29	IN4 / OUT5		alternativ up to 8 digital outputs maximum current: IOL = IOH = 10mA
10	SA2	In	Slave Address 2	30	IN3 / OUT6		
11	BA2	In	Basic Address 2	31	IN2 / OUT7		
12	I2CMOD	In	connect to GND for I²C interface	32	IN1 / OUT8		
13	WUP	In	L: (Power-On) disable Power-On-Macro L: Wakeup from Powerdownmode	33	NC		do not connect
14	SDA	Bidir.	Serial Data Line	34	NC		
15	SCL	In	Serial Clock Line	35	NC		
16	BUZZ	Out	H: Buzzer output (L: Buzzer off)	36	NC		
17	DPROT	In	L: Disable Smallprotokoll do not connect for normal operation	37	NC		
18	PWR	Out	L: Normal Operation H: Powerdownmode	38	NC		
19	NC		do not connect	39	NC		
20	TEST SBUF	IN Out	open-drain with internal pullup 20k..50k IN (Power-On) L: Testmode OUT L: data in sendbuffer	40	NC		

Hinweis:
Die Pins BA0..2, SA0..2, WUP, DPROT 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 LO-Pegel, dass im internen Sendepuffer Daten zur Abholung bereit stehen. Diese Leitung kann z.B. mit einem Interrupteingang des Host Systems verbunden werden.



I²C - Address

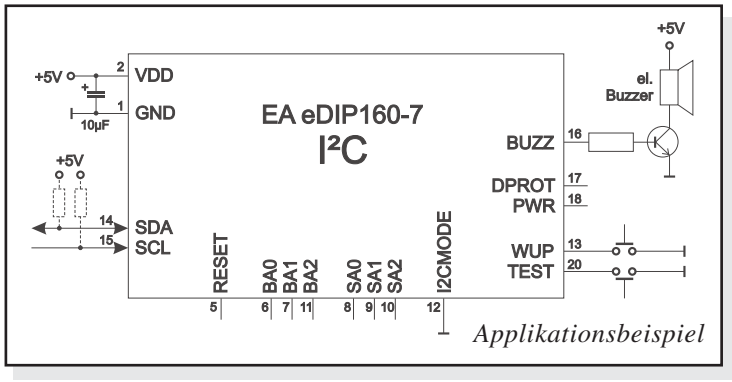
Pin 11,7,6			Base address		I²C address							
BA2	BA1	BA0			D7	D6	D5	D4	D3	D2	D1	D0
L	L	L	$10		0	0	0	1	SA2	SA1	SA0	RW
L	L	H	$20		0	0	1	0				
L	H	L	$30		0	0	1	1				
L	H	H	$40		0	1	0	0				
H	L	L	$70		0	1	1	1				
H	L	H	$90		1	0	0	1				
H	H	L	$B0		1	0	1	1				
H	H	H	$D0		1	1	0	1				

alle Pins offen: Schreiben $DE
Lesen $DF

DATENÜBERTRAGUNG I²C-BUS

Prinzip der 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).

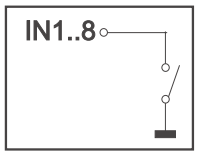


Applikationsbeispiel

EIN- UND AUSGÄNGE

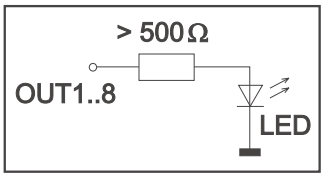
Das eDIP160-7 hat 8 digitale Ein- oder Ausgänge (CMOS Pegel, nicht potentialfrei). Sie können in beliebiger Anzahl umdefiniert werden.

Eingänge

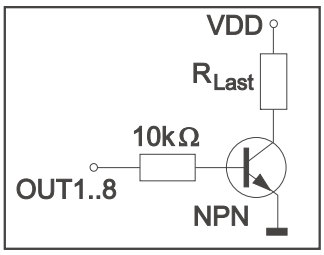


Im Auslieferungszustand sind alle Leitungen als Eingänge definiert. Jeder Eingang hat einen ca. 20..50kΩ Pullup, somit ist es möglich Taster und Schalter direkt nach GND anzuschliessen. Die Eingänge können mit dem Befehl "ESC Y R" abgefragt und ausgewertet werden.
Zusätzlich ist es möglich, bei Änderungen an den Eingängen ein Bit- / Portmakro automatisch aufzurufen (siehe Seite 24). Die automatische Portabfrage läßt sich mit dem Befehl "ESC Y A 1" aktivieren. Bei jeder Änderung des Eingangports werden zuerst die Bitmakros und dann das Portmakro ausgeführt. Ist kein Makro definiert so wird der neue Portzustand in den Sendepuffer gestellt (siehe auch Seite 18: Antworten/Rückmeldungen).
Anmerkung: Die Logik ist für langsame Vorgänge ausgelegt; d.h. mehr als 3 Änderungen pro Sekunde können nicht mehr sinnvoll ausgeführt werden.

Ausgänge



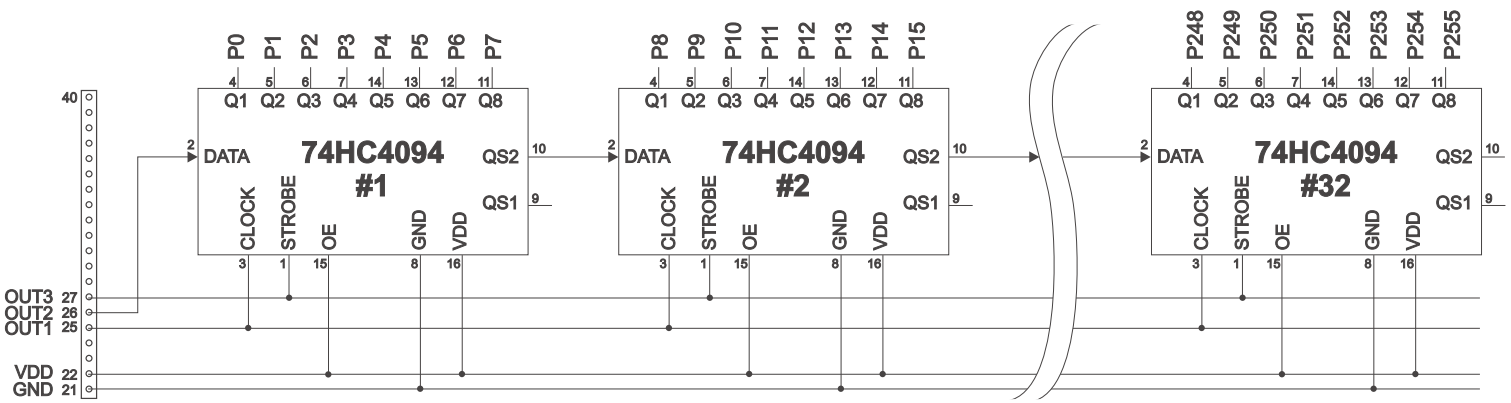
Über den Befehl "ESC Y M anz" können ein oder mehrere Eingänge als Ausgang umdefiniert werden. Dabei werden immer die höherwertigen Eingänge als Ausgänge genutzt. 'ESC Y M 3' schaltet zum Beispiel IN8, IN7 und IN6 als Ausgänge OUT1, OUT2 und OUT3.



Jeder Ausgang kann per Befehl "ESC Y W" individuell angesteuert werden. Pro Leitung kann ein Strom von max. 10mA geschaltet werden. Es ist somit möglich, mit einem Ausgang direkt eine LED (low current) zu schalten. Für höhere Ströme mus ein externer Transistor verwendet werden.

NOCH MEHR AUSGÄNGE (PORTERWEITERUNG)

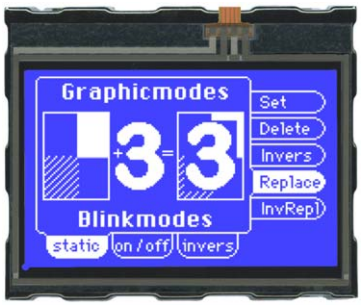
Es können 1 bis 32 Bausteine vom Typ 74HC4094 an das eDIP160 (OUT1...OUT3) angeschlossen werden, damit sind 8 bis 256 weitere Ausgänge möglich. Mit dem Befehl "ESC Y E n1 n2 n3" (siehe Seite 16) können diese Ports komfortabel angesteuert werden.



TOPVIEW UND GEDREHTER EINBAU

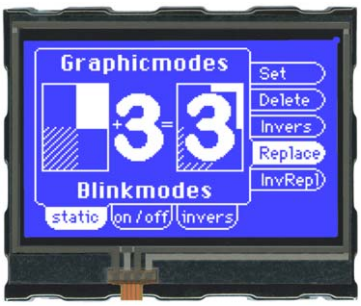
Die Vorzugsblickrichtung des eDIP160 ist schräg von unten (BottomView, 6 Uhr).
Das eDIP160 kann um **180° gedreht eingebaut** werden um die Blickrichtung von Oben (TopView, 12 Uhr) zu erhalten. Zur Richtigstellung des Bildinhaltes muss der Befehl 'ESC DO 2' (siehe Seite 13) ausgeführt werden (z.B. im PowerOnMakro).
Genauso ist es möglich das eDIP160 um **90°** oder **270°** gedreht einzubauen, um ein hochkant-Display mit 104x160 Pixeln zu erhalten.

0°: 'ESC DO 0'



90°: 'ESC DO 1'

180°: 'ESC DO 2'



270°: 'ESC DO 3'

POWER-DOWN-MODE

Um Strom zu sparen (Betrieb mit Akku) kann man mit dem Befehl 'ESC PD mode n2' (siehe Seite 15 unten) verschiedene Power-down-modes aktivieren.

Mode 0 (25µA): Die LED-Beleuchtung wird ausgeschaltet und der Displayinhalt ist nicht mehr sichtbar, bleibt jedoch erhalten. Das eDIP160 benötigt typ. 40µA. Durch die integrierten Suppressordioden kann der Querstrom aber auch 1000µA und mehr betragen. Die Suppressordioden können durch Entfernen der beiden 0Ω Widerstände deaktiviert werden, dann wird ein Power-down-strom von typ. 25µA erreicht. Sie sind mit R_{pd} bezeichnet.
Achtung: Bei deaktivierten Supressordioden unbedingt auf die richtige Polarität des Displays VDD,GND (Pin1+2) achten! Eine auch noch so kurzzeitige Verpolung oder Überspannung kann dann zur sofortigen Zerstörung des gesamten Displays führen.

Mode 1 (1mA): Die LED-Beleuchtung wird ausgeschaltet, der Displayinhalt ist aber weiterhin sichtbar. Der Stromverbrauch reduziert sich dabei auf unter 1mA. Dieser Powerdownmode ist vor allem für die Versionen EA eDIP160J mit dem positiven Display sinnvoll, weil dieses auch ohne Hintergrundbeleuchtung lesbar bleibt.

Mode 2 (2mA): Die LED-Beleuchtung bleibt eingeschaltet und der Displayinhalt ist sichtbar. Der Stromverbrauch reduziert sich dabei auf ca. 2-3mA plus dem eingestellten LED-Strom. Damit läßt sich das Display auch im Dunkeln mit gedimmter LED-Beleuchtung unter 10mA betreiben.

Das eDIP160 kann durch L-Pegel an Pin13 (WUP) und durch Ansprechen der eingestellten I^2C Adresse aus dem Power-down-mode aufgeweckt werden. Zusätzlich kann das eDIP160 auch durch Berührung des Touches (unabhängig von der Position) aufgeweckt werden, wenn dies gewünscht wird. Nach dem Aufwecken können spezielle WakeUpMakros ausgeführt werden (siehe Seite 24).

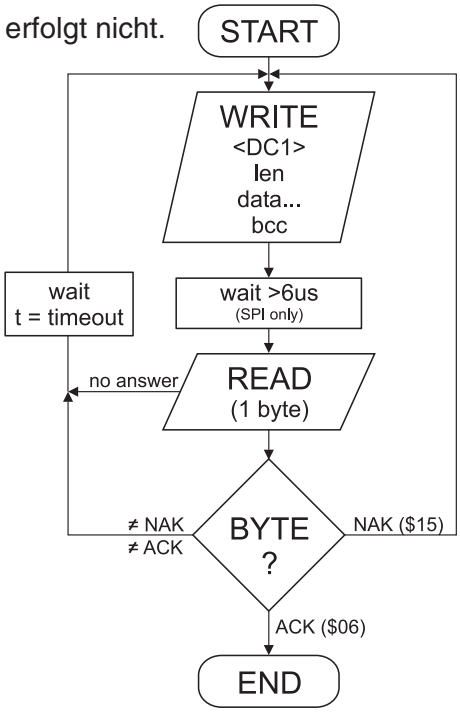
DATENÜBERTRAGUNGSPROTOKOLL (SMALL PROTOKOLL)

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 eDIP160-7 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 erneut 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 sind (z.B. Bild laden ESC UL ...) müssen auf mehrere Pakete aufgeteilt werden. Alle Daten in den Paketen werden nach korrektem Empfang im Display wieder zusammengefügt.



SMALL PROTOLKOLL DEAKTIVIEREN

Das Protokoll ist für alle drei Schnittstellen RS-232, I²C und SPI identisch. Für Tests kann das Protokoll durch L-Pegel an Pin17(DPROT) abgeschaltet werden. Im normalen Betrieb ist allerdings die Aktivierung des Protokolls unbedingt zu empfehlen. Andernfalls wäre ein möglicher Überlauf des Empfangspuffers oder eine fehlerhafte Datenübertragung nicht zu erkennen.

DIE PAKETVARIANTEN IN EINZELNEN

Befehle/Daten zum Display senden

>	<DC1>	len	data...	bcc
<	<ACK>			

<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

Clear display and draw a line from 0,0 to 159,103

<DC1>	len	ESC D L		ESC G D 0 0 159 103							bcc	>
$11	$0A	$1B	$44	$4C	$1B	$47	$44	$00	$00	$9F	$67	$72
												< <ACK>
												$06

Beispiel für ein komplettes Datenpaket

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

Inhalt des Sendepuffers anfordern

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

<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

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. Touchtastendrücke zu senden.

Pufferinformationen anfordern

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

<DC2> = 18(dez.) = $12 1 = 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 1 = 1(dez.) = $01 P = 80(dez.) = $50
<ACK> = 6(dez.) = $06
max. packet size = maximale Anzahl der Nutzdaten eines Protokollpaketes (eDIP160-7 = 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 Protokoll-einstellungen abgefragt.

Letztes Datenpaket wiederholen

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

<DC2> = 18(dez.) = $12 1 = 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 komlette 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

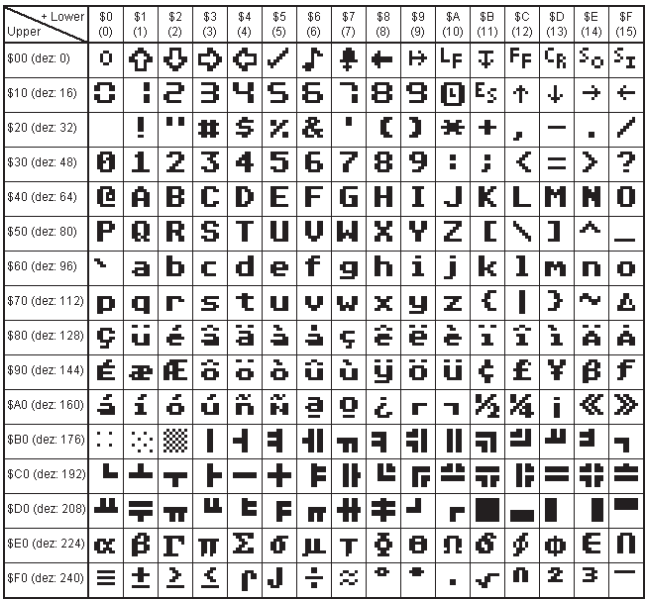
>	<DC2>	3	A	select or deselect	adr	bcc
<	<ACK>					

<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 läst sich das eDIP mit der Adresse adr Selektieren oder Deselektieren.

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 oder ein abgeschaltetes Protokoll (siehe Seite 10+11).
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 '##'.

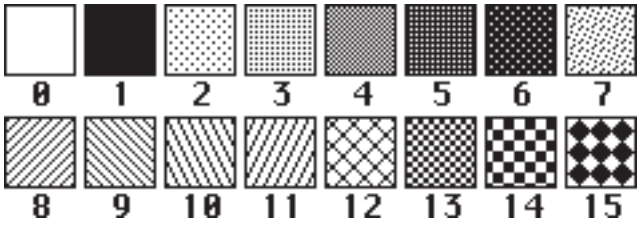


Terminal-Font (Font 0): 8x8 monospaced

Das Terminal besitzt eine eigene Ebene zur Darstellung und ist somit völlig unabhänging 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).

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.



BEFEHLE ÜBER DIE SERIELLE SCHNITTSTELLE SENDEN

Das eDIP läßt sich über diverse eingebaute Befehle programmieren. Jeder Befehl beginnt mit ESCAPE gefolgt von einem oder zwei Befehlsbuchstaben und einigen Parametern.
Es gibt 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.: **#GD0,0,159,103,**
- 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 und 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 **kein Abschlussbyte** wie z.B Carriage Return (außer Zeichenkette: $00).

ALLE BEFEHLE AUF EINEN BLICK

Die eingebaute Intelligenz erlaubt den Aufbau eines Bildschirmes über unten stehende Befehle. Alle Befehle können sowohl über die serielle Schnittstelle als auch in selbst-definierten Makros verwendet werden.

Terminal Befehle									nach Reset
Befehl	**Codes**							**Anmerkung**	
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	ESC	T	C	n1				n1=0: Cursor ist unsichtbar; n1=1: Cursor blinkt;	1
Cursorposition sichern	ESC	T	S					die aktuelle Cursorposition wird gesichert	
Cursorposition restoren	ESC	T	R					die letzte gesicherte Cursorposition wird wieder hergestellt	
Terminal AUS	ESC	T	A					Terminal Anzeige ist ausgeschalten; Ausgaben werden verworfen	
Terminal EIN	ESC	T	E					Terminal Anzeige ist eingeschalten;	Ein
Version ausgeben	ESC	T	V					Die Versions-Nr. wird im Terminal ausgegeben z.B "EA eDIP160-7 V1.0 Rev.A"	
Projektname anzeigen	ESC	T	J					Der Makro-Projektname wird im Terminal ausgegeben z.B. "init / delivery state"	
Informationen anzeigen	ESC	T	I					Das Terminal wird initialisiert und gelöscht, die Software Version, Hardware Revision, der Makro-Projektname und die CRC-Checksummen werden im Terminal ausgegeben.	

Displayfunktionen (Wirkung auf das gesamte Display)						nach Reset
Befehl	**Codes**				**Anmerkung**	
Display Orientierung	ESC	D	O	n1	n1=0: 0°; n1=1: 90°; n1=2: 180°; n1=3: 270°; (0°+180°=160x104; 90°+270°=104x160)	0°
Display Kontrast	ESC	D	K	n1	n1=0..40: Displaykontrast auf Wert n1 setzen n1='+': Kontrast erhöhen; n1='-': Kontrast verringern	20
Disp. Graustufenmodus	ESC	D	G	n1	n1=0: Graustufenmodus AUS; blinken möglich n1=1: Graustufenmodus EIN; blinken nicht mehr möglich	0
Display löschen	ESC	D	L		Displayinhalt löschen (alle Pixel aus)	
Display invertieren	ESC	D	I		Displayinhalt invertieren (alle Pixel umkehren)	
Display füllen	ESC	D	S		Displayinhalt füllen (alle Pixel ein)	
Display ausschalten	ESC	D	A		Displayinhalt wird unsichtbar bleibt aber erhalten, Befehle weiterhin möglich	
Display einschalten	ESC	D	E		Displayinhalt wird wieder sichtbar	Ein
Display Clipboard	ESC	D	C		Inhalt des Clipboards wird dargestellt. Displayausgaben sind nicht mehr sichtbar	
Disp. Normaldarstellung	ESC	D	N		Aktuelles Bild wird dargestellt (Normalbetrieb). Alle Ausgaben wieder sichtbar	

Clipboard Befehle (Zwischenspeicher für Bildbereiche)										nach Reset
Befehl	**Codes**								**Anmerkung**	
Displayinhalt sichern	ESC	C	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	ESC	C	R						Der Bildbereich im Clipboard wird wieder ins Display kopiert	
Bereich kopieren	ESC	C	K	x1	y1				Der Bildbereich im Clipboard wird ins Display nach x1,y1 kopiert	

Geradenfunktionen										nach Reset
Befehl	**Codes**								**Anmerkung**	
Einstellungen										
Punktgröße / Liniendicke	ESC	G	Z	n1	n2				n1 = X-Punktgröße (1..15); n2 = Y-Punktgröße (1..15);	1,1
Verknüpfungsmodus	ESC	G	V	n1					Zeichenmodus einstellen n1: 1=setzen; 2=löschen; 3=invers;	1
Blinkmodus	ESC	G	B	n1					n1:0=kein blinken; 1=An/Aus; 2=Invertierend blinken; 3=Aus/An Phasenverschoben	0
Graudarstellung	ESC	G	B	n1					n1:0=Schwarz; 1=Dunkelgrau; 3=Hellgrau (siehe Befehl ESC DG)	
Geraden und Punkte zeichnen										
Punkt zeichnen	ESC	G	P	x1	y1				Ein Punkt an die Koordinaten x1, y1 setzen	
Gerade zeichnen	ESC	G	D	x1	y1	x2	y2		Eine Gerade von x1,y1 nach x2,y2 zeichnen	
Gerade weiter zeichnen	ESC	G	W	x1	y1				Eine Gerade vom letzten Endpunkt bis x1, y1 zeichnen	0, 0
Rechteck zeichnen	ESC	G	R	x1	y1	x2	y2		Vier Geraden als Rechteck von x1,y1 nach x2,y2 zeichnen	

Bereichsfunktionen										nach Reset
Befehl	**Codes**								**Anmerkung**	
Bereich löschen	ESC	R	L	x1	y1	x2	y2		Einen Bereich von x1,y1 nach x2,y2 löschen (alle Pixel aus)	
Bereich invertieren	ESC	R	I	x1	y1	x2	y2		Einen Bereich von x1,y1 nach x2,y2 invertieren (alle Pixel umkehren)	
Bereich füllen	ESC	R	S	x1	y1	x2	y2		Einen Bereich von x1,y1 nach x2,y2 füllen (alle Pixel ein)	
Bereich m. Füllmuster	ESC	R	M	x1	y1	x2	y2	n1	Einen Bereich von x1,y1 nach x2,y2 mit Muster n1 zeichnen (immer setzen)	
Box zeichnen	ESC	R	O	x1	y1	x2	y2	n1	Ein Rechteck von x1,y1 nach x2,y2 mit Muster n1 zeichnen; (immer Replace)	
Rahmen zeichnen	ESC	R	R	x1	y1	x2	y2	n1	Einen Rahmen Typ n1 von x1,y1 nach x2,y2 zeichnen (immer setzen)	
Rahmenbox zeichnen	ESC	R	T	x1	y1	x2	y2	n1	Eine Rahmenbox Typ n1 von x1,y1 nach x2,y2 zeichnen; (immer Replace)	

Textfunktionen												nach Reset
Befehl	**Codes**										**Anmerkung**	
Einstellungen												
Font einstellen	ESC	Z	F	n1							Font mit der Nummer n1 (0..15) einstellen	0
Font-Zoomfaktor	ESC	Z	Z	n1	n2						n1 = X-Zoomfaktor (1x..4x); n2 = Y-Zoomfaktor (1x..4x)	1,1
zus. Zeilenabstand	ESC	Z	Y	n1							zwischen zwei Textzeilen n1 Pixel (0..15) als zusätzlichen Zeilenabstand einfügen	
Leerzeichenbreite	ESC	Z	J	n1							Leerzeichenbreite: n1=0 aus Zeichensatz; n1=1 wie Ziffer; n1>=2 Breite in Pixel	0
Text-Winkel	ESC	Z	W	n1							Text-Ausgabewinkel: n1=0: 0°; n1=1: 90°;	0
Text-Verknüpfungsmodus	ESC	Z	V	n1							Modus n1: 1=setzen; 2=löschen; 3=invers; 4=Replace; 5=Invers Replace;	4
Text-Blinkattribut	ESC	Z	B	n1							n1:0=kein blinken; 1=An/Aus; 2=Invertierend blinken; 3=Aus/An Phasenverschoben	0
Graudarstellung											n1:0=Schwarz; 1=Dunkelgrau; 3=Hellgrau (siehe Befehl ESC DG)	
Ausgabe von Zeichenketten												
Zeichenkette ausgeben L: Linksbündig C: Zentriert R: Rechtsbündig	ESC	Z	L C R	x1	y1	Text ...	NUL				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 '&' ($26) Zeichen stehen blinken Aus/An Phasenverschoben; Texte die zwischen zwei '@' ($40) Zeichen stehen blinken Invertierend; Das Backslash-Zeichen '\' ($5C) hebt die Sonderfunkion der Zeichen '|~@\' auf; z.B. "name\@test.de" => "name@test.de"	
Zeichenkette für Terminal	ESC	Z	T	Text ...							Befehl um eine Zeichenkette in einem Makro an das Terminal ausgeben zu können	

Bildfunktionen											nach Reset
Befehl	**Codes**									**Anmerkung**	
Einstellungen											
Bild-Zoomfaktor	ESC	U	Z	n1	n2					n1 = X-Zoomfaktor (1x..4x); n2 = Y-Zoomfaktor (1x..4x)	1,1
Bild-Winkel	ESC	U	W	n1						Ausgabewinkel des Bildes: n1=0: 0°; n1=1: 90°	0
Bild-Verknüpfungsmodus	ESC	U	V	n1						Modus n1: 1=setzen; 2=löschen; 3=invers; 4=Replace; 5=Invers Replace;	4
Bild-Blinkattribut	ESC	U	B	n1						n1:0=kein blinken; 1=An/Aus; 2=Invertierend blinken; 3=Aus/An Phasenverschoben	0
Graudarstellung										n1:0=Schwarz; 1=Dunkelgrau; 3=Hellgrau (siehe Befehl ESC DG)	
Ausgabe											
Bild aus Clipboard	ESC	U	C	x1	y1					Der akt. Clipboardinhalt wird mit allen Bildattributen nach x1,y1 geladen	
internes Bild laden	ESC	U	I	x1	y1	nr				internes Bild mit der nr (0..255) aus dem EEPROM nach x1,y1 laden	
Bild laden	ESC	U	L	x1	y1	BLH daten ...				Ein Bild nach x1,y1 laden; daten des Bildes siehe Bildaufbau	
Hardcopy											
Hardcopy senden	ESC	U	H	x1	y1	x2	y2			Es wird ein Bild angefordert. Zuerst werden die Breite und Höhe in Pixel und dann die eigentlichen Bilddaten gesendet.	

Bargraphfunktionen														nach Reset
Befehl	**Codes**												**Anmerkung**	
Definition														
Bargraph definieren	ESC	B	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 löschen	ESC	B	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	
Anwendung														
Bargraph aktualisieren	ESC	B	A	n1	wert								Bargraph mit der Nummer n1 auf den neuen Benutzer-´wert´ setzen und zeichnen.	
Bargraph neu zeichnen	ESC	B	Z	n1									Den Bargraph mit der Nummer n1 komplett neu zeichnen	
Bargraphwert senden	ESC	B	S	n1									Den aktuellen Wert des Bargraph Nr. n1 senden	

Blinkbereiche										nach Reset
Befehl	**Codes**								**Anmerkung**	
Einstellungen										
Blinkzeit einstellen	ESC	Q	Z	n1					Einstellen der Blinkzeit n1= 1..15 in 1/10s; 0=Blinkfunktion deaktivieren	6
Blinkbereichs Befehle										
Blinkattribut löschen	ESC	Q	L	x1	y1	x2	y2		Löscht das Blinkattribut von x1,y1 bis x2,y2. Nicht für Phasenverschobene Blinkbereiche! (Kopiert den Bereich von der Grafik- in die Blinkebene)	
Invertierender Blinkbereich	ESC	Q	I	x1	y1	x2	y2		Definiert einen invertierenden Blinkbereich von x1,y1 bis x2,y2. (Kopiert den invertierten Bereich von der Grafik- in die Blinkebene)	
Muster Blinkbereich	ESC	Q	M	x1	y1	x2	y2	n1	Definiert einen Blinkbereich mit Muster n1 (An/Aus) von x1,y1 bis x2,y2 (Füllt einen Bereich mit dem Muster n1 in der Blinkebebe)	
Phasenverschobene Blinkbereiche										
Restore Blinkbereich	ESC	Q	R	x1	y1	x2	y2		Löscht das Phasenverschobene Blinken von x1,y1 bis x2,y2. Nicht für andere Blinkattribute verwenden! (Kopiert den Bereich von der Blink- in die Grafikebene)	
Phasenverschoben Invertierend	ESC	Q	E	x1	y1	x2	y2		Definiert einen Phasenverschobenen invertierenden Blinkbereich von x1,y1 bis x2,y2. (Kopiert den invertierten Bereich von der Blink- in die Grafikebene)	
Phasenverschoben mit Muster	ESC	Q	P	x1	y1	x2	y2	n1	Definiert einen Phasenverschobenen Blinkbereich mit Muster n1 (Aus/An) von x1,y1 bis x2,y2. (Füllt einen Bereich mit dem Muster n1 in der Grafikebebe)	

Menü Befehle											nach Reset
Befehl	**Codes**									**Anmerkung**	
Einstellungen für Menübox / Touchmenü											
Menü-Font einstellen	ESC	N	F	n1						Font mit der Nummer n1 (0..15) für Menüdarstellung einstellen	0
Menüfont-Zoomfaktor	ESC	N	Z	n1	n2					n1 = X-Zoomfaktor (1x..4x); n2 = Y-Zoomfaktor (1x..4x)	1,1
zus. Zeilenabstand	ESC	N	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	ESC	N	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	ESC	N	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	ESC	N	P							Der vorherige Eintrag wird invertiert oder bleibt am Anfang stehen	
Menüende / Senden	ESC	N	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	ESC	N	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	ESC	N	A							Das Menü wird entfernt und durch den ursprünglichen Hintergrund ersetzt	

Makro Befehle											nach Reset
Befehl	**Codes**									**Anmerkung**	
Makros aufrufen											
Normal Makro ausführen	ESC	M	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	ESC	M	M	n1						Das Menü-Makro mit der Nummer n1 (0..255) aufrufen (max. 7 Ebenen)	
Port Makro ausführen	ESC	M	P	n1						Das Port-Makro mit der Nummer n1 (0..255) aufrufen (max. 7 Ebenen)	
Bit Makro ausführen	ESC	M	B	n1						Das Bit-Makro mit der Nummer n1 (0..255) aufrufen (max. 7 Ebenen)	
automatische (Normal-) Makros											
Makro mit Verzögerung	ESC	M	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 eimal abarbeiten; n3=Pause in 1/10s. Die Ausführung wird durch Befehle (z.B durch Empfang oder Touchmakros) gestoppt.	
autom. Makros zyklisch	ESC	M	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	ESC	M	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											
Makroprozess definieren	ESC	M	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	ESC	M	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

Allgemeine Befehle											nach Reset
Befehl	**Codes**									**Anmerkung**	
Hintergrundbeleuchtung											
Beleuchtung Helligkeit	ESC	Y	H	n1						Helligkeit der LED-Beleuchtung auf n1=0..100% einstellen	100
Softdimmzeit	ESC	Y	Z	n1						n1=0..31: Zeit zum Ändern der LED-Helligkeit von 0..100% in 1/10s	5
Beleuchtung Ein/Aus	ESC	Y	L	n1						Beleuchtung n1=0: AUS; n1=1: EIN; n1=2..255: für n1/10s lang einschalten	1
Parameter speichern	ESC	Y	@							Die aktuelle LED-Helligkeit und Änderungszeit als Startwert im EEPROM speichern	
Sende-Befehle											
Bytes senden	ESC	S	B	anz	daten ...					Es werden anz (=1..255) Bytes zum Sendepuffer gesendet; im Quelltext der Makroprogrammierung darf die Anzahl anz nicht angegeben werden, diese wird vom eDIP-Compiler automatisch eingetragen.	
Version senden	ESC	S	V							Version wird als String gesendet z.B."EA eDIP160-7 V1.0 Rev.A TP+" (Sendepuffer)	
Projektname senden	ESC	S	J							Es wird der Makro-Projektname als String gesendet z.B. "init / delivery" (Sendepuffer)	
Interne Infos senden	ESC	S	I							Es werden interne Informationen vom eDIP gesendet (landen im Sendepuffer)	
Sonstige-Befehle											
Warten (Pause)	ESC	X	n1							n1/10s abwarten bevor der nächste Befehl ausgeführt wird.	
RS485 Adresse einstellen	ESC	K	A	adr						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	ESC	Y	S	n1						Summerausgang (PIN16) wird n1=0:AUS;n1=1:EIN;n1=2..255;für n1/10s eingeschaltet	AUS
Power Down	ESC	P	D	n1	n2					Nach diesem Befehl geht das eDIP in den Power-down mode n1=0..2 (siehe Seite 9) n2=0: kein WakeUp per Touch; n2=1: WakeUp per Touch möglich	

Port-Befehle									nach Reset
Befehl	**Codes**							**Anmerkung**	
Eingangs Ports									
Eingabe Port lesen	ESC	Y	R	n1				n1=0: Alle Eingabe-Ports als Binärwert einlesen (landet im Sendepuffer) n1=1..8: IN-Port n1 einlesen (1=H-Pegel=5V, 0=L-Pegel=0V)	
Port Scan Ein/Aus	ESC	Y	A	n1				Der automatiche Scan des Eingabe-Port wird n1=0: deaktiviert; n1=1: aktiviert	1
Eingabe Port invers	ESC	Y	I	n1				Der Eingabe-Port wird n1=0: normal; n1=1: invertiert ausgewertet	0
Bit-Makros für Eingänge umdefinieren	ESC	Y	D	n1	n2	n3		Eingang n1=1..8 wird bei fallender Flanke n2=0 das Bitmakro n3=0..255 zugewiesen Eingang n1=1..8 wird bei steigender Flanke n2=1 das Bitmakro n3=0..255 zugewiesen	
Ausgangs Ports									
Ausgabe Ports defineren	ESC	Y	M	n1				n1=0: Alle 8 I/O-Ports sind Eingänge IN1..IN8 (=default nach Power-On / Reset) n1=1..8: Anzahl n1 I/O-Ports als Ausgabe-Port benutzen (ab OUT1 steigend)	
Ausgabe Port schreiben	ESC	Y	W	n1	n2			n1=0: Alle OUT-Ports entsprechend n2 (=8-Bit Binärwert) einstellen n1=1..8: OUT-Port n1 rücksetzen (n2=0); setzen (n2=1); invertieren (n2=2)	
Port Erweiterung mit 74HC4094									
zusätzliche Ausgänge setzen	ESC	Y	E	n1	n2	n3		zusätzliche Ausgänge des 74HC4094 (siehe Porterweiterung S.8) von Port n1=0..255 bis Port n2=0..255 einstellen; n3=0: rücksetzen; n3=1: setzen; n3=2: invertieren;	

TOUCH PANEL(NUR EAeDIP160x-7xxTP)

Die Versionen -7xxTP werden mit einem analogen resistiven Touchpanel geliefert. Bis zu 40 Touchbereiche (Tasten, Schalter, Menüs, Bargrapheingaben), können gleichzeitig und pixelgenau definiert werden. Das eDIP unterstützt die Darstellung mit komfortablen Befehlen. 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.

RAHMEN UND TASTENFORMEN

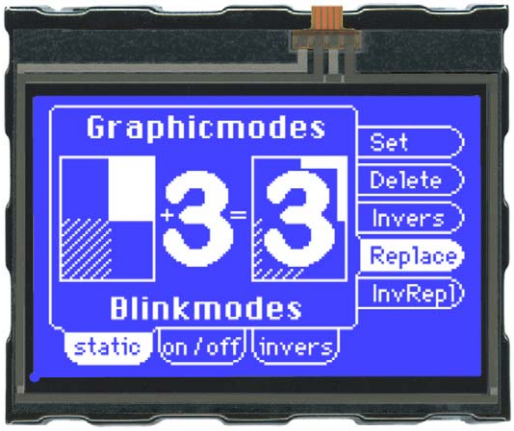
Mit den Befehlen *Rahmen /Rahmenbox zeichnen* sowie beim Zeichnen von Touchtasten kann ein Rahmentyp eingestellt werden. Es stehen dabei 18 Rahmentypen zur Verfügung (0= keinen Rahmen zeichnen). Die Rahmengröße muß mindestens 16x16 Pixel betragen.

BITMAPS ALS TASTEN

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.



Befehle für das Touch-Panel														nach Reset
Befehl	**Codes**												**Anmerkung**	
Voreinstellungen														
Touch-Rahmen Form	ESC	A	E	n1									mit n1 wird der Rahmentyp für die Darstellung von Touch-Tasten/Schaltern eingestellt	1
Radiogroup für Schalter	ESC	A	R	nr									Innerhalb einer Gruppe ist immer nur 1 Schalter aktiv, alle anderen werden deaktiviert 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	0
Voreinstellungen 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
Touchbereiche definieren														
Touch-Taste definieren (Taste ist gedrückt solange der Touch berührt wird)	ESC	A	T	x1	y1	x2	y2	dow Cod	up Cod	Text ...	NUL		'T': Der Bereich von x1,y1 nach x2,y2 wird als Taste definiert. 'K': Der Bereich von x1,y1 nach x2,y2 wird als Schalter definiert. 'U': Das Bild Nr. n1 wird nach x1,y2 geladen und als Taste definiert. 'J': Das Bild Nr. n1 wird nach x1,y2 geladen und als Schalter 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; 'NUL': ($00) = Zeichenkettenende	
			U	x1	y1	n1	dow Cod	up Cod	Text ...	NUL				
Touch-Schalter definieren (Zustand der Schalter toggelt nach jeder Berührung)	ESC	A	K	x1	y1	x2	y2	dow Cod	up Cod	Text ...	NUL			
			J	x1	y1	n1	dow Cod	up Cod	Text ...	NUL				
Touch-Taste mit Menüfunktion definieren	ESC	A	M	x1	y1	x2	y2	dow Cod	up Cod	mnu Cod	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/Menumakro+(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 Richung 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 im Clipboard gesichert.	
Zeichenbereich definieren	ESC	A	D	x1	y1	x2	y2	n1					Ein Zeichenbereich wird definiert. Innerhalb der Eck-Koodinaten x1,y1 und x2,y2 kann dann mit der Strichstärke n1 gezeichnet werden.	
Freien Touchbereich def.	ESC	A	H	x1	y1	x2	y2						Ein frei benutzbarer Touchbereich wird definiert. Touchaktionen (down, up und drag) innerhalb der Eck-Koodinaten x1,y1 und x2,y2 werden gesendet.	
Bar per Touch einstellbar	ESC	A	B	nr									Der Bargraph mit der Nr. n1 wird zur Eingabe per Touchpanel definiert.	
Globale Einstellungen														
Touch-Abfrage Ein/Aus	ESC	A	A	n1									Touchrabfrage wird n1=0:deaktiviert; n1=1:aktiviert;	1
Touch-Tasten Reaktion	ESC	A	I	n1									automatisches Invertieren beim Berühren der Touch-Taste: n1=0=AUS; n1=1=EIN;	1
			S	n1									Summer piepst kurz beim Berühren einer Touch-Taste: n1=0=AUS; n1=1=EIN	1
Barwert automatisch in den Sendepuffer stellen	ESC	A	Q	n1									das automatische Speichern eines neuen Bargraphwertes per Toucheingabe wird n1=0:deaktiviert; n1=1:neuer Wert wird nach dem Einstellen in den Sendepuffer gestellt n1=2: jede Änderung landet während des Einstellens im Sendepuffer.	1
sonstige Funktionen														
Touch-Taste Invertieren	ESC	A	N	Cod									Die Touch-Taste mit dem zugeordnetem Return-Code wird manuell Invertiert	
Touch-Schalter einstellen			P	Cod	n1								Zustand des Schalters wird per Befehl geändert n1=0=Aus; n1=1=Ein.	
Touch-Schalter abfragen			X	Cod									Zustand des Schalters (Aus=0; Ein=1) wird in den Sendepuffer gestellt.	
Radiogroup abfragen			G	nr									Der downcode des aktivierten Schalters aus der Radiogroup mit der Nummer nr wird in den Sendepuffer gestellt.	
Touch-Bereich Löschen			L	Cod	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.	
			V	x1	y1	n1							Touchbereich der die Koordinaten x1,y1 umschliesst aus der Touchabfrage entfernen n1=0: Bereich bleibt sichtbar; n1=1: Bereich löschen	

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 A @' 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).

ANTWORTEN/RÜCKMELDUNGEN

Alle Antworten des eDIP160 werden in einen Sendepuffer gestellt. Über das Small-Protokoll werden diese dann vom Host angefordert (siehe Seite 10). Dies kann per „Polling“ geschehen, oder altenativ dazu zeigt der Pin 20 „SBUF“ mit einem LO-Pegel an, dass Daten zur Abholung bereit stehen.

Antworten über die serielle Schnittstelle							
Kennung		**anz**	**daten**				**Anmerkung**
Selbstständige Antworten (landen im Sendepuffer)							
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	**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	**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	**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	**P**	1	wert				Nach Änderung des Eingangs-Port wird der neue 8-Bit Wert gesendet. Der Port-Scan muß aktiviert sein siehe Befehl 'ESC Y A n1'. Es wird nur gesendet wenn kein Port-Makro mit der Nr. wert definiert ist !
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 (landen im Sendepuffer)							
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			Nach dem Befehl 'ESC A G nr' wird der code des aktiven Touch-Schalters von der Radiogroup nr gesendet.
ESC	**Y**	2	nr	wert			Nach dem Befehl 'ESC Y R' wird der angeforderte Eingangs-Port gesendet. nr=0: wert ist ein Binärwert aller Eingänge. nr=1..8: wert ist 0 oder 1 je nach Zustand des Eingans nr
ESC	**V**	anz	Zeichenkette...				Nach dem Befehl 'ESC S V' wird die Version der eDIP-Firmware als Zeichenkette gesendet. z.B. "EA eDIP160-7 V1.0 Rev.A TP+"
ESC	**J**	anz	Zeichenkette Projektname...				Nach dem Befehl 'ESC S J' wird der Makro-Projektname als Zeichenkette gesendet. z.B. "init / delivery state"
ESC	**I**	anz	X-Pixel, Y-Pixel, Version, Touchinfo, CRC-ROM, CRC-ROMsoll EEP in KB, CRC-EEP, CRC-EEPsoll, EEPanz				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 = Hardwarerevisonsbuchstabe 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 Hardcopys (Linke obere Ecke) BLH-Bildaten: 2 Byte: breite, höhe (in Pixel) + anzahl Bytes Bilddaten anzahl = ((breite+7)/8*höhe);
ESC	**U**	G	x1	y1	BLH-Bild (2 x Daten)...		Im Graustufenmodus wird ein zweites mal 'anzahl Bytes Bilddaten' für die Blinkebene gesendet.

VORGELADENE 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 ist möglich.

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

Font 1: 4x6 monospaced

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

Font 3: 7x12 monospaced

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

Font 7: grosse Ziffern BigZif57

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

Font 4: GENEVA10 proportional

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

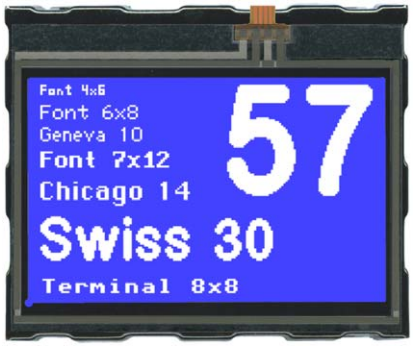
Font 5: GENEVA13 proportional

+ 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	{	|	}	~	△
$80 (dez: 128)	€	ü	é	â	ä	à	å	ç	ê	ë	è	ï	î	ì	Ä	Å
$90 (dez: 144)	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü					
$A0 (dez: 160)	á	í	ó	ú	ñ	Ñ	ª	º								
$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	å	ç	ê	ë	è	ï	î	ì	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	ª	º								
$B0 (dez: 176)																
$C0 (dez: 192)																
$D0 (dez: 208)																
$E0 (dez: 224)		ß														
$F0 (dez: 240)									o							

Font 6: Swiss30 Bold proportional

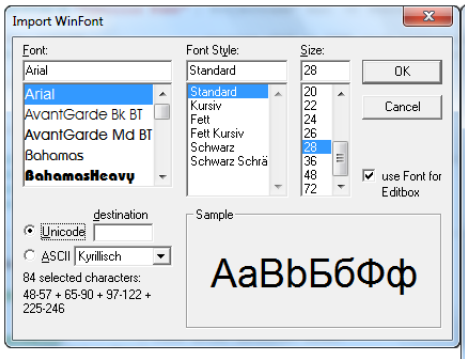


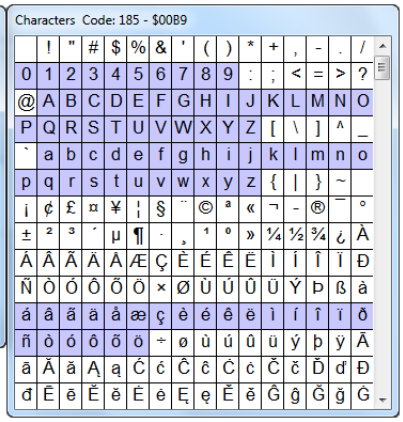
integrierte Schriften im Auslieferungszustand

LADBARE ZEICHENSÄTZE

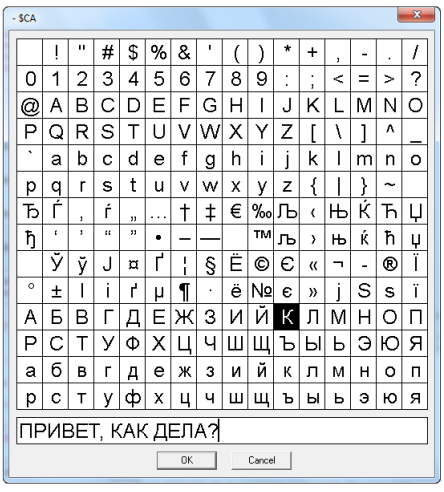
Compileranweisung "WinFont:"

Damit ist es möglich, TrueType-Fonts in verschiedenen Größen zu rastern und einzubinden. Sie können entweder den kompletten Zeichensatz (ASCII) einbinden oder Sie wählen aus dem gesamten Unicode-Zeichensatz bestimmte Zeichen aus. Ein Doppelclick im KitEditor auf den Fontnamen öffnet dazu die Font-Auswahlbox. Um die Verwendung dieser Zeichensätze zu vereinfachen gibt es die kompfortable Möglichkeit einer Zeichen-Auswahlbox. Wird im KitEditor ein String ausgegeben (z.B. #ZL 5,5, "Hallo") kann mit einem Doppelclick auf den String diese geöffnet werden. Es können nun die gewünschten Zeichen ausgewählt werden. Dies ist vor allem bei kyrillischen, asiatischen oder Symbolschriftarten zu empfehelen. Der KitEditor setzt darauf hin automatischen den richtigen ASCII-Code ein. Alternativ zu den Anführungsstrichen können geschweifte Klemmern genutzt werden (z.B. +ZL5,5, {48616C6C6F}).





Import WinFonts



Edit Box

Compileranweisung "Font:"

Verwendet werden kann folgendes Font-Format:

- FXT: Textfont von eDIP240/eDIP320 und KIT-Serie

DISPLAY BLINKMODUS

Nach dem Einschalten oder dem Befehl **'ESC DG 0'** ist das eDIP160 im Blinkmodus. Zwei Bildinhalte werden abwechselnd in einem einstellbaren Zeitraum Nacheinader angezeigt.

Blinkattribute werden mit den Befehlen 'ESC ZB,UB,GB n1' eingestellt:
n1=0: kein blinken
n1=1: An/Aus blinken
n1=2: Invertierend blinken
n1=3: Aus/An blinken (Phasenverschoben)

Innerhalb von Zeichenketten (ESC ZL,ZC,ZR ...) kann das Blinken lokal aktiviert werden:
Texte zwischen zwei '~' ($7E) blinken An/Aus.
Texte zwischen zwei '&' ($26) blinken Aus/An Phasenverschoben.
Texte zwischen zwei '@' ($40) blinken Invertierend;

Außerdem kann mit den Blinkbereichsfunktionen 'ESC Q...' rechteckigen Bereichen nachträglich ein Blinkattribut zugewiesen oder entfernt werden.

DISPLAY GRAUSTUFENMODUS

Nach dem Befehl **'ESC DG 1'** ist das eDIP160 im Graustufenmodus. Blinken ist dann nicht mehr möglich, stattdessen werden die Blinkinformationen zur Darstellung von zwei Graustufen verwendet.

Graue Farben werden mit Hilfe der Blinkattribute 'ESC ZB,UB,GB n1' erzeugt:
n1=0: Schwarz
n1=1: Dungelgrau
n1=3: Hellgrau

Innerhalb von Zeichenketten (ESC ZL,ZC,ZR ...) können graue Texte lokal verwendet werden:
Texte zwischen zwei '~' ($7E) sind Dungelgrau.
Texte zwischen zwei '&' ($26) sind Hellgrau.

Außerdem können mit den Blinkbereichsfunktionen 'ESC Q...' rechteckige Bereiche nachträglich grau oder schwarz eingefärbt werden.


```
Makro:  MnAutoStart
    #TA                                  ; Terminal off

    #AR 1                                ; define radiogroup
    #AE 13                               ; define frame
    #AF  CHICAGO14                       ; define touchlabelfont
    #AK   20,30,  70,50,  1,0,  "BLINK"  ; place button „blink"
    #AK  100,30,150,50,  2,0,  "GRAY"    ; place button „gray"

    #ZF  SWISS30B                        ; select textfont
    #ZC  45,35,    "~OnOff~|&OffOn&"     ; place blink text
    #ZC  125,35,   "~dark~|&light&"      ; place gray text

Touchmakro:  1   ;  Flashing
    #DG 0        ;  grayscalemode off -> flashing possible

Touchmakro:  2   ;  Grayscale modus
    #DG 1        ;  grayscalemode on -> flashing impossible
```

Beispielmakros von den Hardcopys

MAKRO PROGRAMMIERUNG

Einzelne oder mehrere Befehlsfolgen können als sog. Makros zusammengefasst und im EEPROM fest abgespeichert werden. Diese können dann mit den Befehlen *Makro ausführen* 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 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'.

Bit Makro `BitMakro:`

Start bei Anlegen/Änderung einer Spannung an einzelnen Eingängen IN1..8 (Bitweise) oder per Befehl 'ESC MB xx'. Die Bit-Makros 1..8 reagieren auf fallende Flanke, Bit-Makros 9..16 auf die steigende Flanke der Eingänge 1..8. Mit dem Befehl 'ESC YD n1 n2 n3' die Zuordung der Eingänge zu den Bitmakros umdefiniert werden (siehe Seite 16).

Port Makro `PortMakro:`

Start bei Anlegen/Änderung einer Spannung an den 8 Eingängen IN 1..8 (binär kombiniert) oder per Befehl 'ESC MP xx'.

Power-On-Makro `PowerOnMakro:`

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

Reset-Makro `ResetMakro:`

Start nach einem externen Reset.

Watchdog-Makro `WatchdogMakro:`

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

Brown-Out-Makro `BrownOutMakro:`

Start nach einem Spannungseinbruch <3V.

Achtung: Wird im PowerOn-, Reset-, Watchdog- oder BrownOut-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 WUP:
-PowerOff - Pin13 (WUP) auf GND legen
-PowerOn - Pin13 (WUP) wieder öffnen.

WakeUpPin-Makro `WakeupPinMakro:`

Start nach dem Aufwachen aus dem Power-Down-Mode per Pin13 (WUP).

WakeUpTouch-Makro `WakeupTouchMakro:`

Start nach dem Aufwachen aus dem Power-Down-Mode per Touch-Berührung (gesamte Touchfläche ist aktiv).

WakeUpI2C-Makro `WakeupI2CMakro:`

Start aus dem Power-Down-Mode über den I²C Bus

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

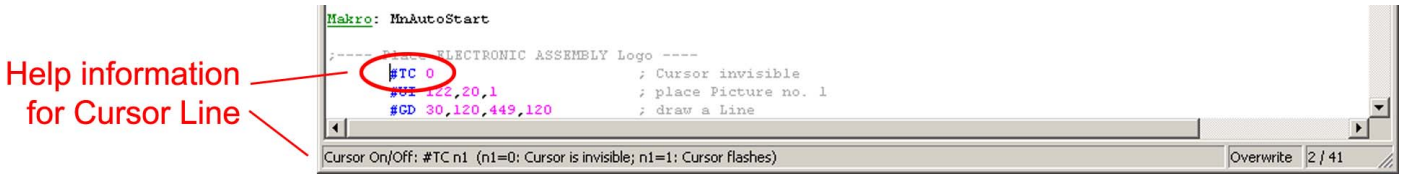
Um nun Ihre speziellen Makros erstellen zu können, benötigen Sie folgende Hilfsmittel:

- um das Display an den PC anschliessen zu können benötigen Sie das als Zubehör erhältliche Evaluationboard EA 9777-2USB oder einen selbstgebauten Adapter mit Pegelwandler MAX232 (Applikationsbeispiel siehe S. 5).
- die Software ELECTRONIC ASSMBLY LCD-Tools*); sie enthält einen KitEditor, Compiler 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 KitEditor definiert, startet man über F5 den Compiler. Dieser erzeugt eine Datei DEMO.EEP, ist auch das Evaluationboard EA 9777-2USB angeschlossen, oder das Display über einen MAX232 an den PC angeschlossen, dann wird diese Datei automatisch in das EEPROM des Displays gebrannt. Der 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.

HILFE IM KIT-EDITOR (ELECTRONIC ASSEMBLY LCDTOOLS)

In der Statuszeile am unteren Rand des Editorfensters werden für den aktuellen Befehl mögliche Parameter kurz erläutert. Der Cursor muss dazu in der entsprechenden Zeile stehen.
Für mehr Informationen drücken Sie F1.



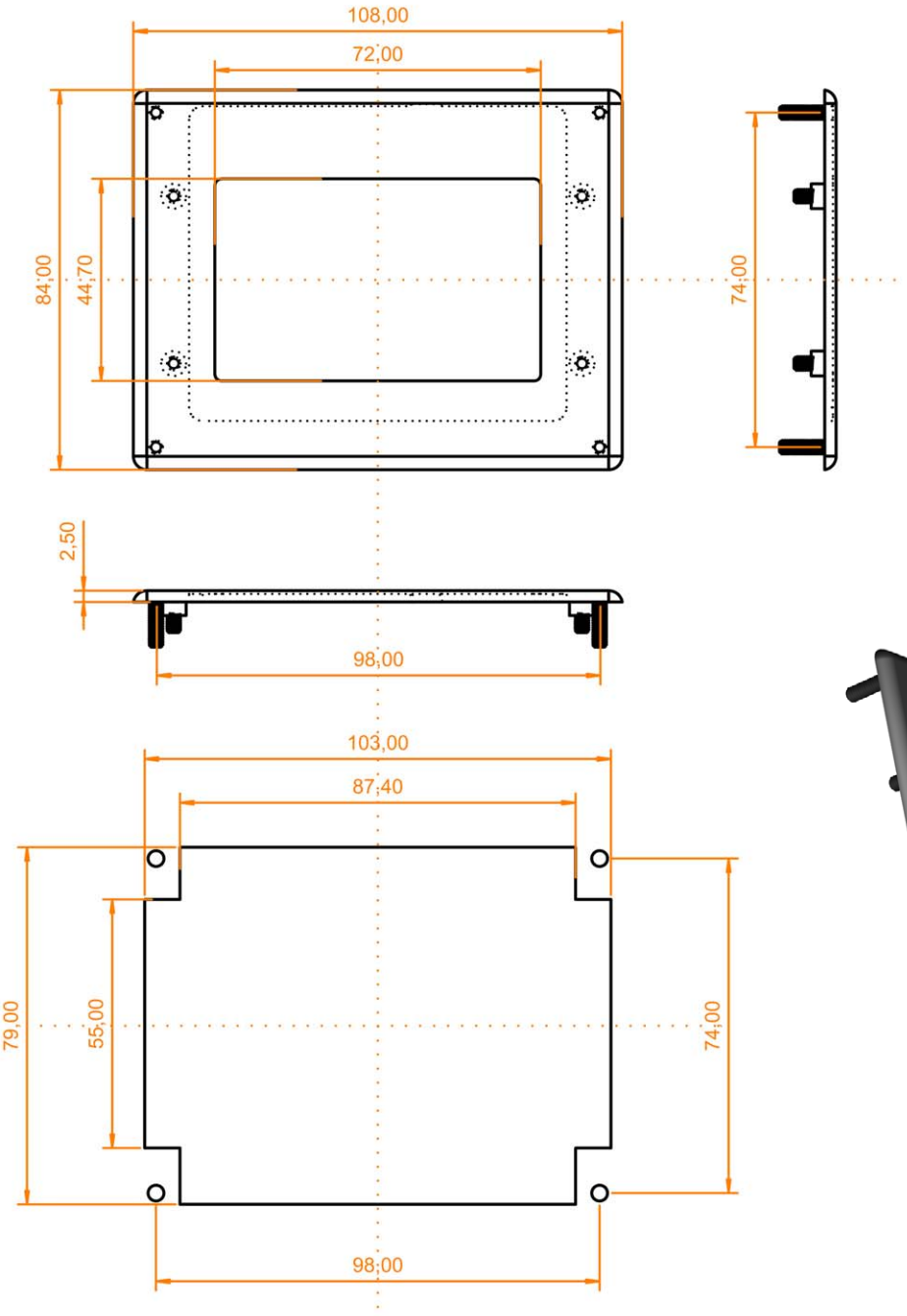
*) *im Internet unter*http://www.lcd-module.de/deu/dip/edip.htm

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		3.2		5.2	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				1	uA
Input Pull-up Resistor		20		50	kOhms
Output Low Voltage				0.7	V
Output High Voltage	VDD = 3,3V VDD = 5V	2.5 4.2			V
Output Current				20	mA
Current Backlight on	VDD = 3,3V VDD = 5V		190 125		mA
Current Backlight off	VDD = 3,3V VDD = 5V		10 15		mA
Power Down	Mode 0		25		µA

EINBAUBLENDE (ZUBEHÖR EA 0FP161-7SW)

aus eloxiertem Aluminium, schwarz



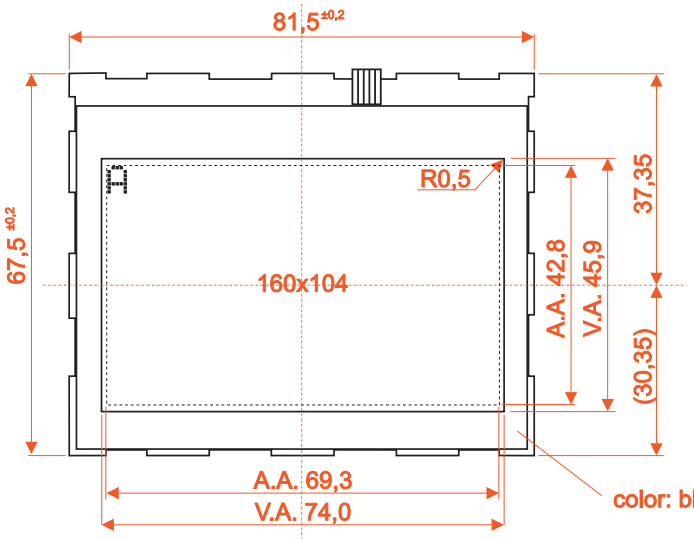
alle Maße in mm

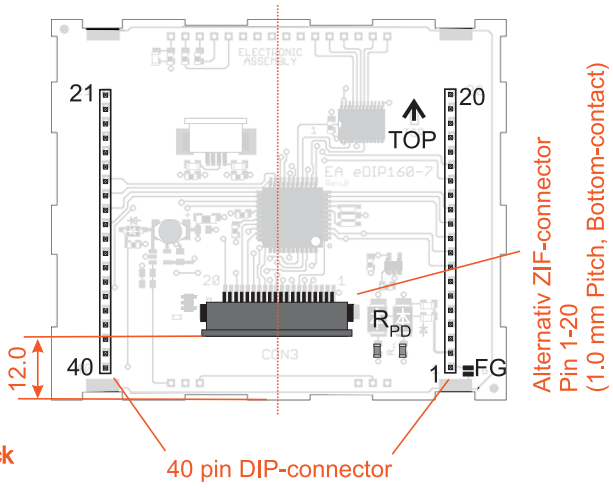
Hinweise zur Handhabung und zum Betrieb

- *Zur elektrischen Zerstörungs des Moduls kann führen: Verpolung oder Überspannung der Stromversorgung, Überspannung oder Verpolung bzw. statische Entladung an den Eingängen, Kurzschließen der Ausgänge.*
- *Vor dem Abstecken desModuls 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.*

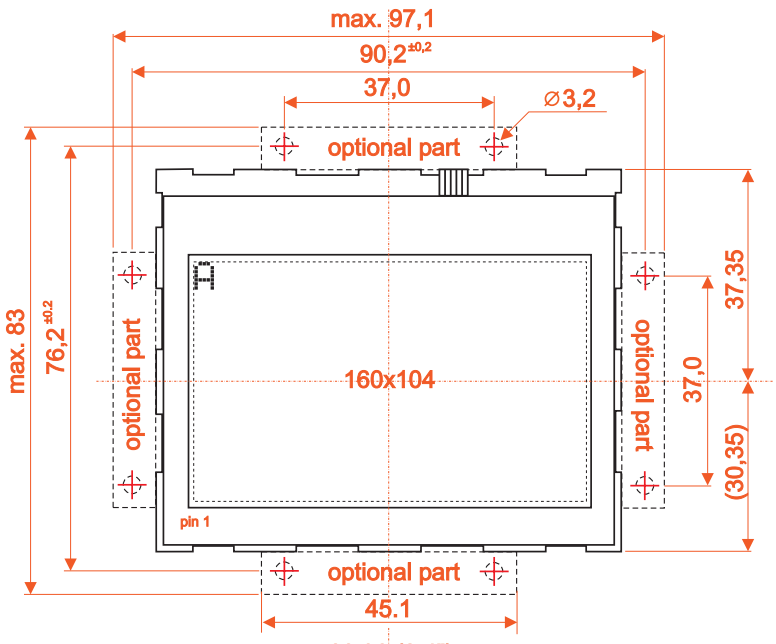
EA eDIP160-7 INTELLIGENTES HMI

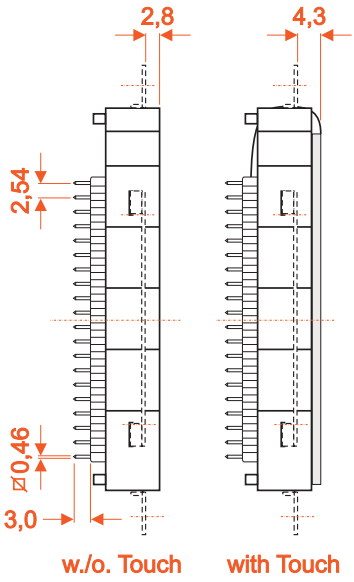
ABMESSUNGEN

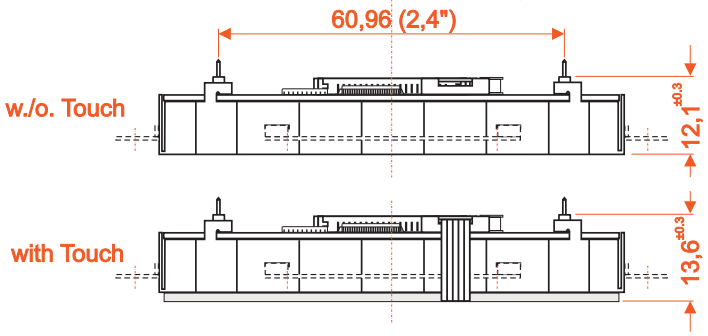




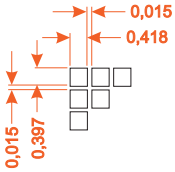
FG: Verbindung Metallrahmen und GND (spezielle ESD / EMV Anforderungen)







alle Maße in mm



Zwei Montagelaschen liegen der Lieferung bei.

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

DISPLAY VISIONS GmbH
Zeppelinstraße 19
D-82205 Gilching
Germany

Fon: +49 (0)8105-778090
Fax: +49 (0)8105-778099
e-Mail: info@lcd-module.de
Web: www.lcd-module.de