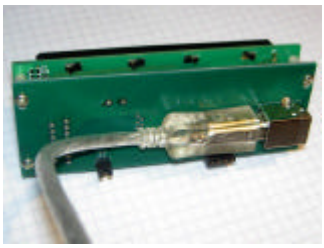
USB LCD character display



Features

- Ready made USB LCD display with 4x20-Zeichen
- High-quality LCD text module with backlight
- No contrast adjustment necessary
- Supply voltage: USB powered
- Display CONTROLLER: HD44780 compatible
- USB Chipset: CH341A
- Integrated I²C/TWI master interface
- System requirements: XP, 2000, Vista, WIN 7 32/64 bit
- Software drivers: ProfiLab Expert, LCD Hype, LCD Smartie, MediaPortal
- Simple programming with ActiveX control (OCX)
- Program examples in C++, Delphi and Visual basic.

Overview

The USB LCD display combines a high-quality LCD text module with 4x20 characters and background lighting with an USB interface circuitry and is ready for plug & play.
An additional I²C/TWI master interface allows connections to user extensions or ready-made I²C-circuits, like the MicroChip I²C demo board. Dimensions and position of USB connector make the device suitable for internal PC installations as well as for external use.

Download link for manual, driver and examples:

http://www.abacom-online.de/div/setup_usb_lcd.exe

Installation

- Before connecting the device to the USB download and install setup software please.
- After that connect the display to the USB and follow the Windows Plug & Play instructions.
- The display is now ready for use. Further information on how to use the display can be found in the manual installed with the software.

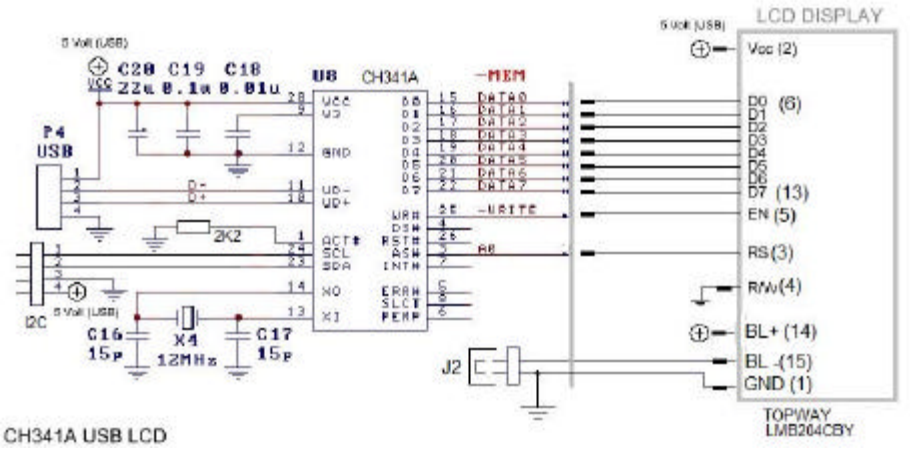
LCD

The device uses a HD44780 compatible display controller with 8 bit parallel data bus. The conversion of the serial data from the USB to the parallel bus of the display module is realized with a CH341A interface chip. This produces the eight data signals (D0…D7) for the display, as well as the signals RS (register SELECT) and EN (Enable). The R/W input of the display is bound to ground, so that data can be written to the display.

Background lighting

The backlight voltage is taken from the USB. The ground connection of the lighting can be removed with Jumper (J2), so it can be replaced with a mechanical switch or transistor, in order to make the lighting switch-able. The typical current of the backlight is approx. 120mA. I²C master connection is provided by a female pin connector together with USB voltage supply for extensions. The total current taken from an USN port must not exceed 500mA.

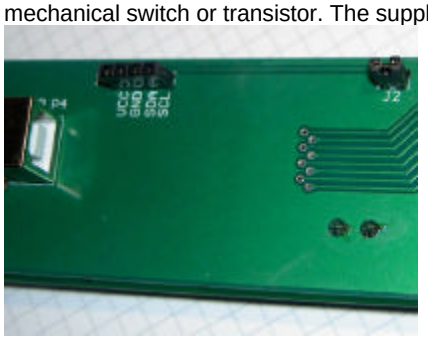
The I²C pull-up resistors (usually 2K2) must provided by the users extension circuitry.

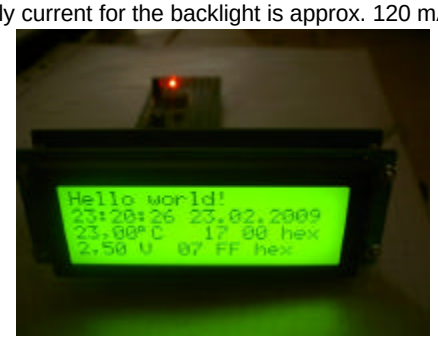


CH341A USB LCD

Backlight

The backlight is power-supplied from the USB voltage as well. The ground connection of the backlight supply voltage can be split removing jumper J2, to replace it with a mechanical switch or transistor. The supply current for the backlight is approx. 120 mA.





I²C master connector

The USB power (Vcc, GND) and the I²C signals (SCA and SCL) are available at a female pin connector. Make sure that the overall current from USB does not exceed 500 mA. The I²C pull-up resistors (usually 2K2) must be provided by the user circuitry.

Software

The software setup copies the following files to your hard-disc:

API
This directory contains files and examples made for some programming languages.

DOC
Contains some useful datasheets.

DRIVER:
This folder contains files necessary for Windows plug & play installation. Connecting the display to the USB these driver files for CH341A chipset installed. The display is then ready for use. In software applications the display is addressed with a device number. This is a consecutive number of available CH341A chips starting from zero. Applications open a data channel to the CH341A chip that will then execute the data transport to the HD4470 compatible display controller. In most applications the device no. is adjustable.

TEST
A simple test program can be found in this folder. It shows the device no. on each display.

PLUGINS
For use with LCD hype, LCD smartie and MedialPortal software additional files must be copied to the software. The files can be found here. See ReadMe files for further information.

Functional test
For a simple device test and to find out a displays device no. use the USB_LCD_TEST.EXE from the \TEST folder. Clicking the test button display will show their device no. (Device #)

ProfiLab Expert
Examples for ProfiLab Expert 4.0 can be found in the directory \PROFILAB.

Controlling the display with ProfiLab mainly two ProfiLab components are necessary. The components ”LCD display” generates data signals for a HD44780 display controller. A ”CH341A” component is used to output the data over USB to the display controller.

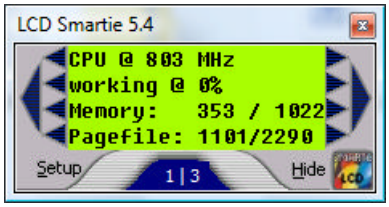
The HD44780 display controller is controlled by two registers (control and data register). To write data to this registers two functions from the CH341A API are used:

CH341AMEMWRITEADDR0
CH341AMEMWRITEADDR1

The I²C bus can also be controlled, using corresponding CH341A API functions.

Other software
The LCD can be used with some programs from other manufacturers. Files necessary for that purpose can be found in directory \PLUGINS. Also refer to the ReadMe files in those folders.

LCD-Hype is a freeware controlling LCD´s. (http://www.lcdhype.de.vu)



LCD Smartie is another free and open-source software
(http://lcdsmartie.sourceforge.net) .

MediaPortal is a multimedia application supporting LCD´s.
(http://www.team-mediaportal.de)

Programming interface (API)
Programming examples for some languages can be found in the \API directory of the software installation.

The display and the I²C interface is programmed using a ActiveX control (file ABACOM_USB_LCD.OCX). The ActiveX control is registered as ”ABACOM USB LCD” during software installation process and can be imported with any programming language that support ActiveX controls. Please refer to the manual of your programming language how ActiveX controls a handled.

The following code snippets are meant as illustration for the functional principles and may fail to compile. Remember the code examples.

To program the LCD an object ”ABACOM USB LCD” (USBLCDX) must be created. This could be done dynamically at run-time or with the IDE´s form designer.

```
TUSBLCDX *LCD ;

LCD = new TUSBLCDX(this);
LCD->Parent = this;
LCD->Left = 24;
LCD->Top = 40;
```

The LCD object defines some properties and methods for general control administration (left, top, etc.) and specialized ones for LCD and I²C control., like

```
LCD->LCD_ROW1TEXT = ...
```

or

```
LCD->I2C_Write(...)
```

Methods and Properties are named with leading ”LCD_” and ”I2C_”.corresponding to their purpose.

LCD programming
Each of the four LCD rows has a corresponding VARIANT text property.
Setting these properties text is written to the display:

```
LCD->LCD_ROW1TEXT = ”Text for row 1”;
LCD->LCD_ROW2TEXT = ”Text for row 2”;
LCD->LCD_ROW3TEXT = ”Text for row 3”;
LCD->LCD_ROW4TEXT = ”Text for row 4”;
```

The display format can be adjusted like this:

```
LCD->LCD_Columns = 20;
LCD->LCD_Rows = 4;
```

4x20 characters is set by default.

In case the length of a text exceeds the number of display columns, the text wil be scrolled automatically.

Another way to write text to the display is to ”PRINT” it. Text is output at the bottom line and lines above are shifted upwards.

```
LCD->LCD_Print( time );
```

A text row can easily be cleared assigning an empty string:

```
LCD->LCD_ROW1TEXT = ””;
```

Mutated vowels and ß are replaced automatically:

```
LCD->LCD_ROW1TEXT = ”Olivenöl”;
```

Is displayed as ”Olivenoel”. Apart from that character translations can be defined to transform Windows characters to the display font. An example is the ° degree character that is transformed from chr(176) to chr(223). Such translations can be added to a list of the LCD object:

```
LCD->LCD_AddCharTranslation(176,223); // replace degree char
LCD->LCD_AddCharTranslation(65 , 66); // make ”A” a ”B”
```

...
As soon as the list is emptied, no translations are performed anymore.

```
LCD->LCD_ClearCharTranslations;
```

The HD44780 allows eight user character pattern to be defined - chr(0)...chr(7).The pattern is defined by eight data bytes for each character:

```
LCD-LCD_DefineCustomChar(1, ”1F 1F 1F 1F 1F 1F 1F 1F”);
```

A block character is defined by above example. That is display wherever a chr(1) is output to the display. The following link may be helpful to define a character pattern:
http://www.quinapalus.com/hd44780udg.html
The pattern data is transferred in readable hex dump format, where each byte is written in its hexadecimal representation separated by spaces.

Additional functions of the HD44780 display controller can be executed writing directly to its control register:

```
if (CheckBox1->Checked)
  {LCD->LCD_WriteToControlRegister(12); // Display ON
else
  {LCD->LCD_WriteToControlRegister(8); // Display OFF
```

HD44780 are documented in the controllers datasheet.

In the need to use more than one display, additional LCD objects must be created. To address a display the device number must be set:

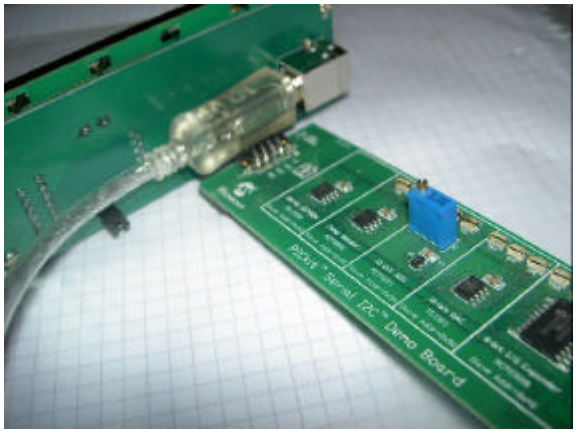
```
LCD1->LCD_DeviceNo = 0; //1st display
LCD2->LCD_DeviceNo = 1; //second display
etc.
```

I²C (TWI) programming

The device is equipped with an integrated I²C(TWI) master interface, which makes it possible to connect additional peripheral chips and control them with PC software. Some basic knowledge in I²C technology and understanding of chips datasheets is necessary for that.

The programming examples are based on a ready-made I²C demonstration board from MicroChip (TM), which can be purchased separately from our web-shop.
(http://www.electronic-software-shop.com/product_info.php?pName=microchip-ic-board-p-17&cName=hardware-ic-twi-c-2_13)

Instead of that, any other I²C circuitry could be connected and programmed.



The following I²C methods are usable for that purpose:

Read/write bytes from/to chip registers

The method

```
LCD->I2C_Write(0x20,0x09,LEDStatus)
```

Writes a byte (here LEDStatus) to the chip (slave) address 0x20 into a chip register (here 0x09).

```
LCD->I2C_Read(0x20,0x09,InData)
```

Read a register (here 0x09) of a chip, with chip address 0x20. The return value InData contains a readable ASCII string (HEX dump; e.g. ”A7”)

Direct read/write data over i²C bus
The following stream function allows direct read and/or write over the I²C bus:

```
OutData = "92 00";
LCD->I2C_Stream(outData, 2, inData);
```

The parameters (inData; outData) are readable ASCII strings (Hex dumps). The example writes two bytes 0x92 and 0x00 out on the bus. The number of bytes to write is implied with the the length OutData hex dump.

The number of bytes to read must be given in the second parameter (here: two). The return value is a readable hex dump string, like ”FE B6” of the read bytes.

This function writes/reads bytes – without certain protocol – over the I²C bus and is therefore usable with almost any peripheral chip type.

EEPROM
Read/write to EEPROM can be done with

```
LCD->I2C_WriteEEPROM(eprom24C02, 0, ”FF FE”);
LCD->I2C_ReadEEPROM(eprom24C02, 0, 7, Data);
```

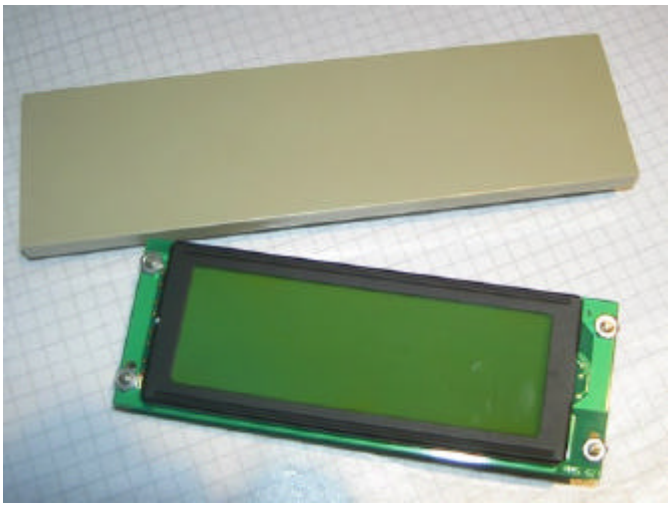
For write operation the EEPROM type, the memory (start) address and the write data must be fed. For read operations the number of bytes to read must be given in addition (here: seven). Read and write data are readable hex dump strings again.

Clock frequency
The I²C speed can be adjusted in four steps:

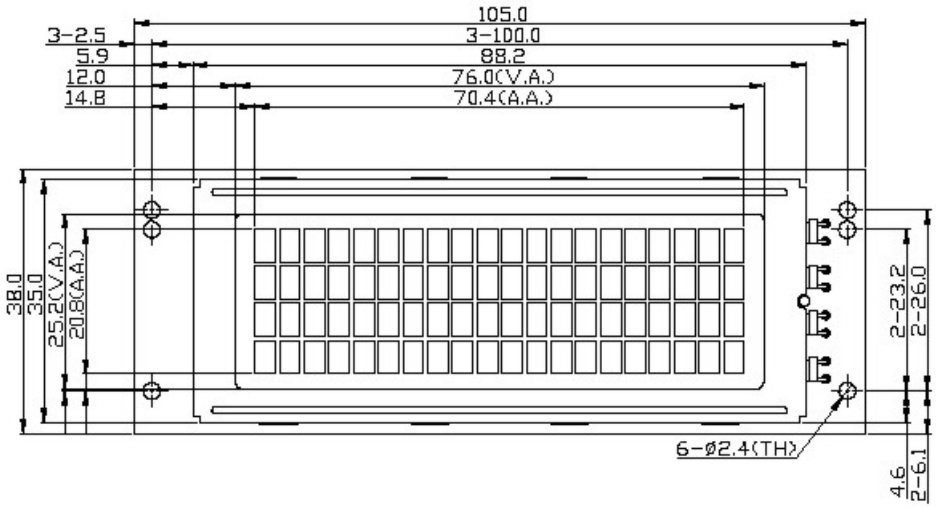
```
LCD->I2C_Speed = slow_20kHz
LCD->I2C_Speed = normal_100kHz
LCD->I2C_Speed = fast_400kHz
LCD->I2C_Speed = high_750kHz
```

Fast and high clock rates may not be supported by all I²C chips.

Dimensions



Size compared with an access panel of a 5,25” PC slot.
Mounting depth is approx. 35 mm.



Links

http://www.sprut.de/electronic/lcd/
http://www.quinapalus.com/hd44780udg.html
http://en.wikipedia.org/wiki/HD44780_Character_LCD
http://www.electronic-software-shop.com/product_info.php?pName=ch341a-usbinterfacemodule-dip28-p-30
http://www.electronic-software-shop.com/product_info.php?pName=microchip-ic-board-p-17&cName=hardware-ic-twi-c-2_13