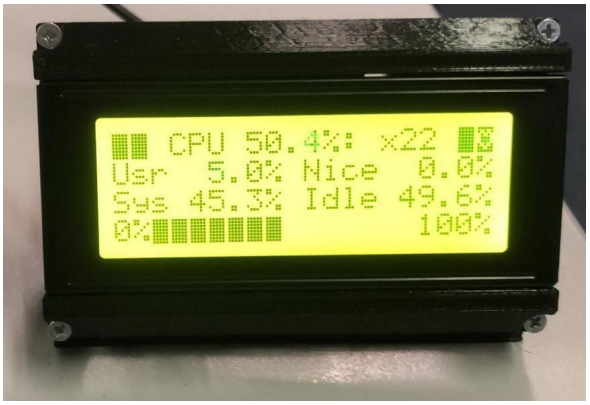
I2C Workshop

Labortage 2021
Mark Hoffmann



Einrichten der Hardware (Verlöten)

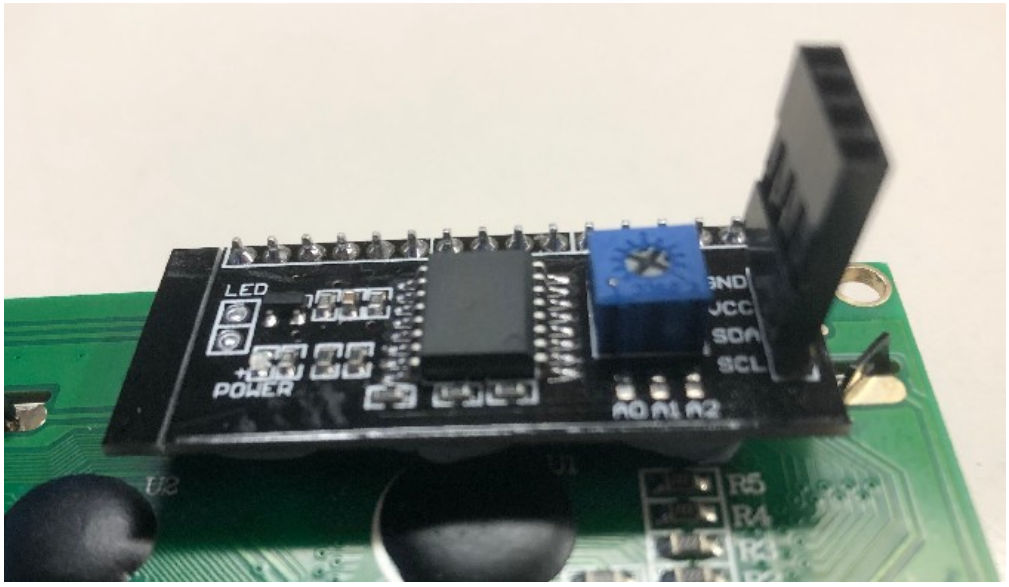
ACHTUNG! Bitte zuerst lesen, dann löten! Es drohen Kurzschluss (also permanentes Versagen des Displays) und die Option darauf, dass das Display nicht ins Gehäuse passt.

I2C-HD44780 Adapterplatine

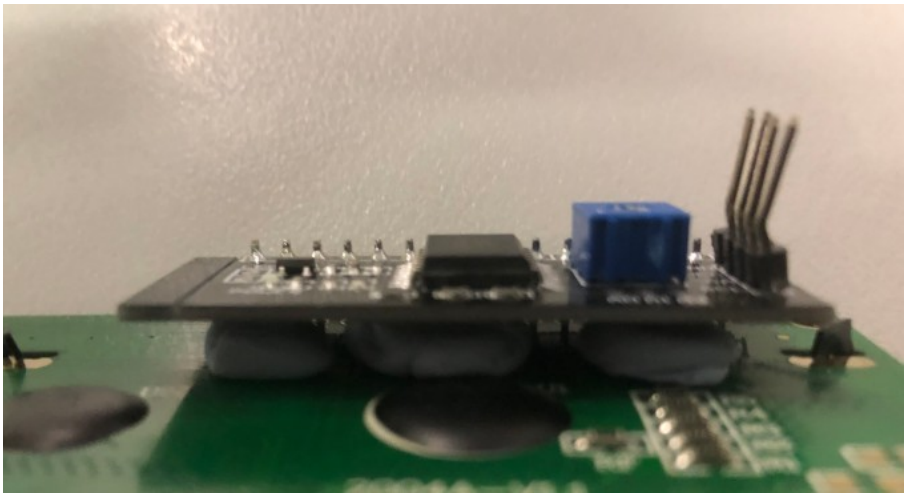
Zuerst bitte das Display mit dem I2C / HD44780Adapter verlöten.

Dazu die vier Pins für den I2C Anschluss am Adapter zurechtbiegen mit der vierfach Jumper Wire Hülse . Dies ist nötig, da die vier Jumper Wire vom I2C-Bus nach Einbau ins Gehäuse sonst nicht angeschlossen werden können.

Des weiteren braucht es Abstandshalter zwischen Adapterplatine und Displayplatine – gelöst mit Blu Tack (englische Klebeknete, liegt dem Bausatz bei).



Dies ist nötig, damit 1. kein Kurschluss entsteht (das Display lässt sich in dem Fall nicht mehr richtig ansprechen … gar nicht mehr) und 2. die Pins der Adapterplatine nicht zu weit vorne rausstehen und damit dann die Platine nicht in das Gehäuse passt. **ACHTUNG – dies beachten!** Entlöten macht eher weniger Spaß! Ein „gesundes“ Herausstehen der Pins auf der Vorderseite ist für das Verlöten empfohlen.



Pull-Up Widerstände

Für die Pull-Up Widerstände gibt es drei Varianten: Realisierung mit 2x 10kΩ, 2x 4,7kΩ oder keine Verwendung von Widerständen. Interessant ist die Herangehensweise, zu ermitteln, ob I2C Slave Geräte bereits einen

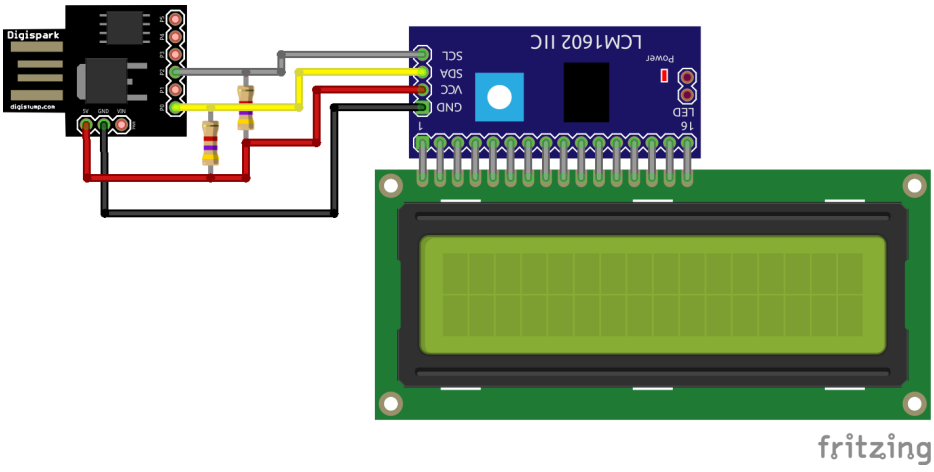
integrierten Pull-Up Widerstand haben. Dafür einfach mit einem Multimeter den Widerstand zwischen +5V und SDA oder SCL bestimmen.

Die hier verwendeten Displays haben bereits einen internen Pull-Up Widerstand mit dem Wert 6,84kΩ. Alle I2C Knoten an einem Bus tragen mit ihren internen Pull-Up Widerständen zu einer Verringerung des Gesamt-Pull-Up Widerstands hinzu, sie erzeugen einen parallelen Widerstand. Dieser sollte nicht weniger als 2,2kΩ betragen. Der ATtiny85 hat ebenfalls einen integrierten Pull-Up Widerstand.

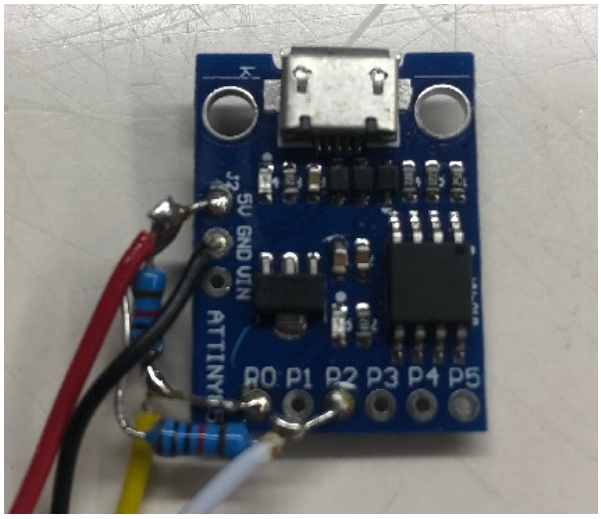
Bei der Verwendung der Jumper Wire habe ich mir angewöhnt für SDA (Daten) ein gelbes zu nehmen und für SCL (Clock) ein weißes; für VCC ein rotes und für GND ein schwarzes. Gelb „schlägt“ Weiß - ist eine „intensivere“ Farbe und Daten sind „wichtiger“ als Clock Signale. Natürlich müssen SDA und SCL nicht wie bei anderen Bus Arten (z.B. UART) gekreuzt werden.

Es gibt zwei mögliche Lötvarianten:

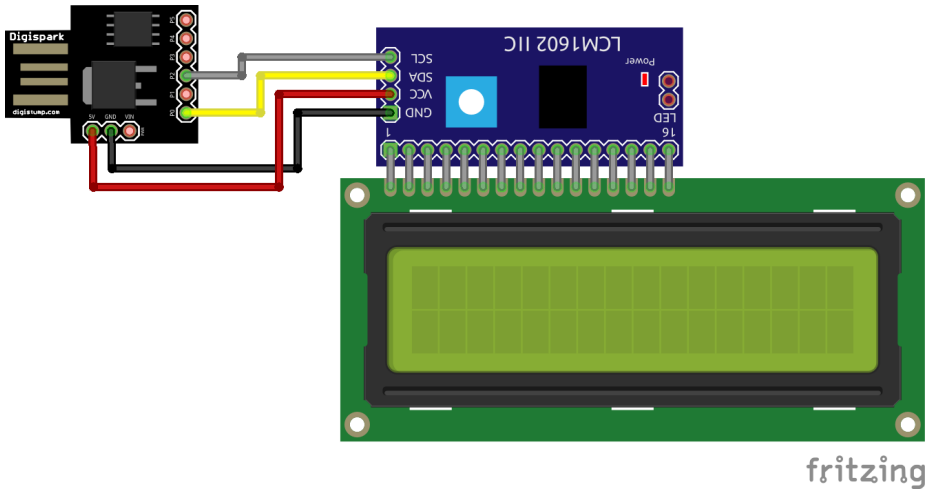
1.) Mit Pull-Up Widerständen – für die Verwendung mehrer Busteilnehmer



Das kann dann so aussehen. Es empfiehlt sich, Klebeband um die Platine bei folgendem Aufbau zu wickeln, da die Jumper Wire gerne schon mal beim Wackeln abbrechen.



2.) Ohne Pull-Up Widerstände – für die Verwendung nur eines Busteilnehmers, z.B. des Displays. Hierzu einfach die (wahlweise abgewinkelten) Pfostenleisten an die dafür vorgesehenen Löcher löten und direkt per Jumper Wire abgreifen.



Nun kann man das Display in das Gehäuse einbauen. Dafür setzt man an den beiden oberen Löchern (an der Oberseite des Gehäuses) zwei der vier Schrauben ein und dreht sie in die zwei Muttern. In den unteren zwei Löchern werden nur die Schrauben, nicht aber Muttern eingesetzt.

Jetzt kann die Verdrahtung stattfinden und die Inbetriebnahme mit dem LCDd Daemon.

Präparieren des Digisparks

Herunterladen des Flashertools Micronucleus:

wget https://github.com/micronucleus/micronucleus/raw/master/commandline/builds/x86_64-linux-gnu/micronucleus

chmod +x micronucleus
sudo cp micronucleus /usr/local/bin

Herunterladen der Till Harbaum'schen Digispark I2C-Tiny-USB Firmware:

wget https://raw.githubusercontent.com/harbaum/I2C-Tiny-USB/master/digispark/main.hex

Flashen der Firmware per:

sudo micronucleus --run --dump-progress --type intel-hex main.hex

Hierbei Digispark erst nach Aufforderung einstecken.

Einrichtung der Software (Linux)

LCDproc Daemon und I2C-Buszugriffssoftware installieren per:

Ubuntu Linux Pakete: sudo apt install lcdproc i2c-tools libftdi1
Das Paket python-smbus findet sich nicht in den aktuellen Repositories; man nehme von https://is.gd/zuyato die folgende Datei und installiere diese per sudo dpkg -i python-smbus_4.0-2_amd64.deb.

Arch Linux: sudo pacman -S i2c-tools lm_sensors libftdi
Das Paket python-smbus ist bereits in i2c-tools bzw. enthalten.

Einspielen der alternativen Config-Datei LCDd.conf nach /etc/.

Emitteln der Nummer des I2C-Buses per i2cdetect -l
Wichtig ist die Zeile, bei der i2c-tiny-usb steht, z.B. i2c-**10**.

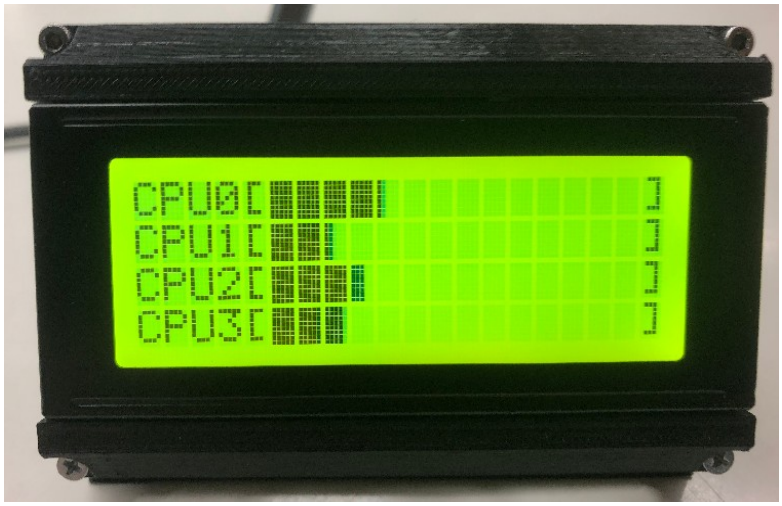
Eintragen dieser Nummer in die Datei /etc/LCDd.conf; Beispiel: Device=/dev/i2c-**10**.

Einspielen der alternativen Treiberdatei hd44780.so (md5sum: b9313acca34b40b01870bc265f910ad6) in den Ordner /lib/x86_64-linux-gnu/lcdproc/. Diese wird benötigt, um abweichende Pinbelegungen des PCF8574T (I2C 8-Bit Port Expander) in verschiedenen Varianten einzustellen (vgl. Diskussion bei https://is.gd/osoxen). Durch Ändern des Eintrags DriverPath=/usr/lib/x86_64-linux-gnu/lcdproc/ in der Datei /etc/LCDd.conf ist dieser Pfad (Standardwert bei Installation unter Ubuntu) anpassbar, so lange er zu der Datei hd44780.so führt. Es ist zu beachten, dass viele Projekte, die LCDproc nutzen, dies mit einer anderen CPU-Architektur verwenden, z.B. ARM beim Raspberry Pi. Hier bietet sich der Befehl file hd44780.so an, der die jeweilige Architektur verrät. Der Befehl ldd hd44780.so zeigt eventuell nicht erfüllte Abhängigkeiten von anderen .so Dateien.

Evtl. wird ein Kernelmodul Nachladen per modprobe i2c-dev benötigt.

Testen des Displays per sudo LCDd -f -r 5 -s 0. Das Display sollte Text anzeigen; oder - falls nicht, sollte es im Terminalfenster detailliertere Informationen geben, warum dies nicht der Fall ist (!!!). Das folgende Bild erscheint bei erfolgreicher Verbindung:

Man kann Statistiken anzeigen, wie z.B. CPU-Auslastung durch Verwendung des zugehörige Programms lcdproc Programms. Dazu muss (!) die Datei /etc/lcdproc.conf verändert werden; lcdproc Parameter zur Anzeige des Screens werden ignoriert. Im Standardfall wird alle paar Sekunden eine andere Statistikenart angezeigt.



Zum Anzeigen von beliebigem Text per Netcat (der LCDd muss laufen) die Datei hello_netcat.txt mit folgendem Inhalt anlegen:

```
hello
screen_add Screen01
widget_add Screen01 Number string
widget_add Screen01 Name string
widget_set Screen01 Number 1 1 "Hello Labortage \'21"
widget_set Screen01 Name 1 2   "... LCDproc Demo …"
```

Der Befehl nc localhost 13666 < hello_labortage.txt zeigt dann den gewünschten Text an.

Es ist auch möglich den LCDd Daemon übers Netzwerk zu betreiben; dazu den Befehl sudo LCDd -a 10.0.1.94 einsetzen (IP nur Beispiel), wobei diese angegebene IP auf den Wert des Rechners gesetzt werden muss, an den das Display auch angeschlossen ist. Man kann also das Display von fremden Rechnern ansprechen per nc 10.0.1.94 13666 < hello_labortage.txt.

Das verwendete Protokoll findet sich unter:
http://lcdproc.omnipotent.net/download/netstuff.txt

Es sein folgende Seiten empfohlen (Danke an CyReVolt!):

https://padcom.aplaline.com/2015/12/running-lcdproc-on-digispark.html?m=1
Diese Seite ist für Personen geeignet, die ihre ausführbaren Dateien lieber selber kompilieren wollen.

Zudem findet sich eine Komplettlösung unter:
https://gist.github.com/orangecms/a9a24eb2a0dd86e0c69fd14251e21189