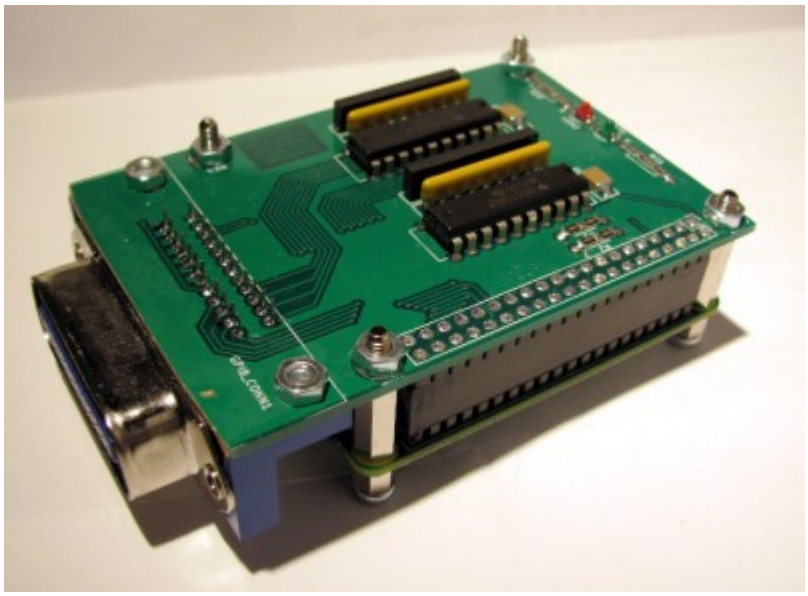
GPIB mit dem Raspberry PI

Vor einiger Zeit sind wir auf das Raspberry Pi Shield von Thomas Klima (github) gestoßen. Da viele Messgeräte GPIB besitzen und man damit gut Geräte automatisieren kann, hat Stefan Bausätze zusammengestellt, die wir aufgebaut haben. Da nicht alles auf den ersten Blick sofort ersichtlich ist, soll hier ein möglicher Weg zur erfolgreichen Inbetriebnahme dokumentiert werden.

Aufbau der Leiterplatte

Die doppelseitige Platine wurde vom Autor bezogen und ist professionell gefertigt worden, was die Bestückung zum Kinderspiel macht. Die Platine ist als Aufsteckplatine für Raspberry Pi 2+ und 3+ entwickelt. Dazu werden extra hohe Buchsenleisten benötigt. Bei der Verwendung eines Raspberry Zero entfällt dies allerdings und es können Buchsenleiste mit einer Standardhöhe verwendet werden. Anders als in der Originalbestückung haben wir eine Centronics-Buchse verwendet, die man direkt ohne Kabel an ein Gerät stecken kann. Dafür muss die Buchse auf der Unterseite bestückt werden.



Das Bild zeigt ein Sandwich aus GPIB Shield und Raspberry Pi Zero.

Installation der Software und Funktionstest

Aufbauend auf einer Raspbian-Installation ist auf github ein Treiber für die Platine verfügbar. Es werden mehrere Pakete benötigt, die mit folgendem Befehl installiert werden können:

```
$ apt-get install raspberrypi-kernel-headers bison byacc
```

Die aktuellen Quellen von linux-gpib müssen von Sourceforge geladen und entpackt werden. Mittels

```
$ ./configure
$ patch -p1 < gpib_bitbang_legacy-4.1.0.patch
```

```
$ make
$ make install
```

wird eine linux-gpib-Version mit den benötigten Treibern gebaut und in das System installiert.

```
$ sudo modprobe gpib_common
$ sudo modprobe gpib_bitbang
$ sudo modprobe gpio
$ sudo gpib_config
```

Dies lädt die Treibermodule, was in der Debugkonsole mit „gpib: registered gpib_bitbang interface“ bestätigt wird.

In */etc/gpib.conf* muss die eigene GPIB-Konfiguration abgelegt werden. Beispiel Die Einstellungen werden mit dem Befehl *gpib_config* geladen.

Schließt man den Pi an ein Gerät mit der Adresse 19 an, so kann man zum Funktionstest dessen Kennung auslesen:

```
ibterm -d19
Attempting to open /dev/gpib0
pad = 19, sad = 0, timeout = 10, send_eoi = 1, eos_mode = 0x0000
ibterm>*IDN?
Hewlett-Packard, US37230463, ESG-D4000A, A.01.12
```

Um via Python Daten von GPIB-Geräte auszulesen lohnt es sich dieses Tutorial zu befolgen. Desweiteren wurde ein Repository angelegt um verschiedene Geräte-Libaries zu sammeln.