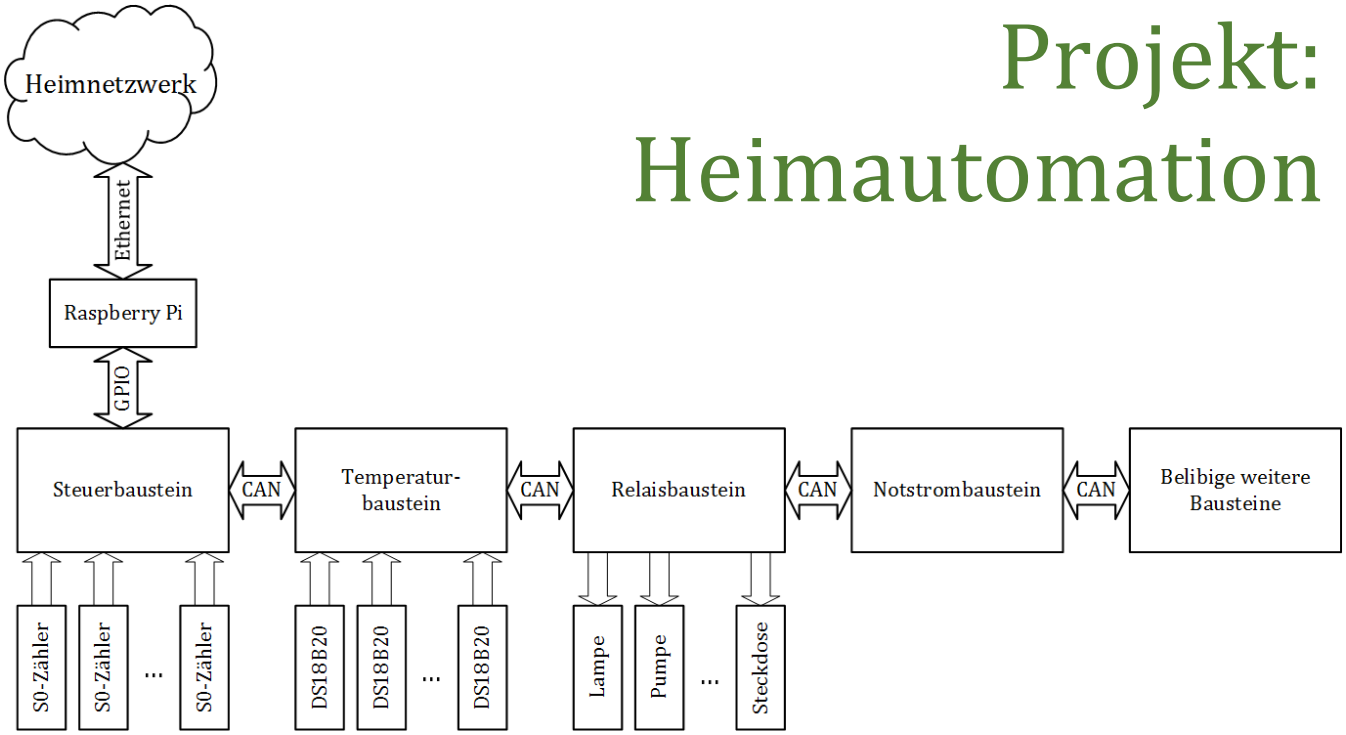
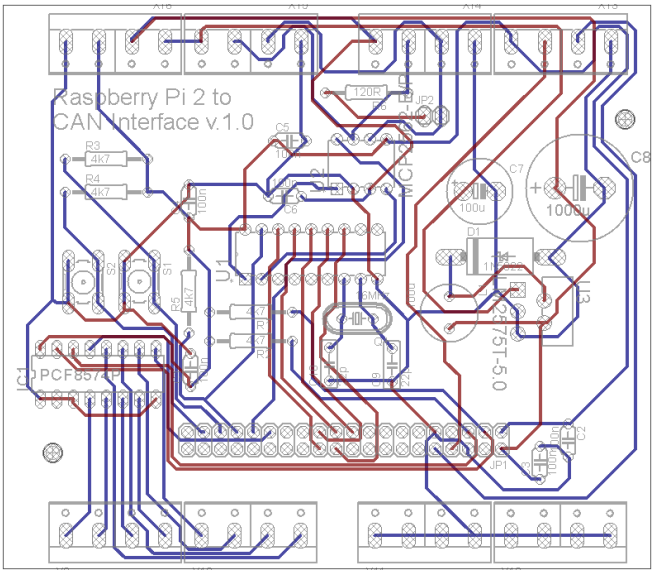
Projekt: Heimautomation



Unterschiedliche Systeme zur Heimautomation gibt es bereits wie Sand am Meer. Was an meinem System besonders ist? Hier mein „Lastenheft“:

- Open Source: Ich lege alle Baupläne und Quellcodes offen. Wer will, kann es nachbauen.
- Standardisiere Schnittstellen: Es sollen nur Schnittstellen zum Einsatz kommen, die bereits weit verbreitet sind und keine zu speziellen Kenntnisse erfordern.
  - ✓ Zum Messen der Energie wird die S0-Schnittstelle verwendet. Diese ist bereits seit Jahren Standard und einfach zu verstehen.
  - ✓ Um Daten von einem Modul zum nächsten zu übertragen, soll der CAN-Bus verwendet werden. Er ist vor allem in der Automobiltechnik weit verbreitet, sehr resistent gegenüber elektromagnetischen Störungen und mit günstigen Bauteilen zu realisieren.
  - ✓ Um von außen auf die Module zuzugreifen soll Ethernet zum Einsatz kommen. Vorteil hier: Jeder PC hat diese Schnittstelle und in den meisten Haushalten ist auch eine Verkabelung vorhanden.
- Modularer Aufbau: Ich will im Laufe der Zeit unterschiedliche Module entwickeln, die eine Gewisse „Grundintelligenz“ besitzen, um den Ausfall einzelner Module verschmerzen zu können.
  - ✓ Steuerbaustein: Hier ist ein Raspberry Pi das eigentliche Gehirn. Er soll in erster Linie die Kommunikation zu den anderen Bausteinen managen, aber auch Protokollierungsaufgaben wahrnehmen. Zusätzlich soll auch noch die Schnittstelle zu den S0-Stromzählern implementiert werden.
  - ✓ Temperatur-Baustein: Auf Ebay gibt es günstige und praktische Temperatursonden (DS18B20), die bereits einen vorimplementierten 1-wire-Bus enthalten. Der Baustein soll die Temperaturen messen und per CAN an die anderen Bausteine übermitteln.
  - ✓ Relais-Baustein: Wie der Name schon sagt, der Baustein enthält mehrere Relais. Damit kann man z.B. Pumpen, Lampen, Steckdosen, … schalten. Um die Ausfallsicherheit zu erhöhen, kann auch hier eine Grundintelligenz integriert werden.
  - ✓ Notstrom-Baustein: Falls einmal die 12V-Stromversorgung ausfällt, kann dieser Baustein die Stromversorgung für eine gewisse Zeit gewährleisten.
  - ✓ Weitere Bausteine folgen nach Bedarf.

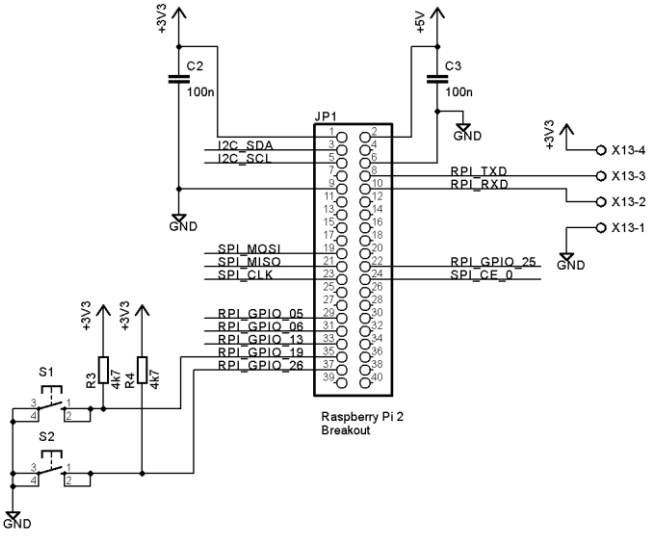
Auf den folgenden Seiten soll der Bau erklärt werden.



Der Steuerbaustein

Der Steuerbaustein soll, wie der Name schon sagt, Steuerungsaufgaben übernehmen. Über ihn soll mit der Steuerung kommuniziert werden. Es wird ein Raspberry Pi 2 zum Einsatz kommen, der die Steuerung über die GPIOs bedient. Im Folgenden erkläre ich den Aufbau des Boards.

Schnittstelle zum Raspberry Pi 2

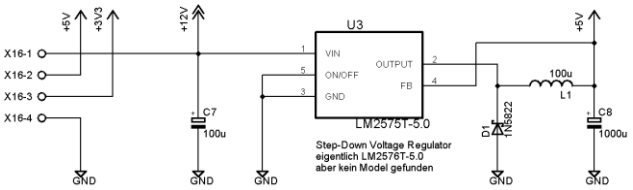


Die Schnittstelle zum RPi ist letztendlich schon vorgegeben. Die GPIOs haben bereits ihre Funktionen, so dass ich mich nur entscheiden muss, welche Funktionen ich nach außen führen möchte. Das wären:

- Die I2C-Schnittstelle, auf den Pins 3 und 5. Hierbei ist darauf zu achten, dass der I2C-Bus noch Pull-up-Widerstände benötigt. Die habe ich nicht vergessen. Den I2C-Bus benötige ich, um mit dem PCF8574 zu kommunizieren.
- Die SPI-Schnittstelle befindet sich auf den Pins 19, 21, 23 und 24, wobei der Pin 24 der CS-Pin ist und auch leicht ausgetauscht werden könnte. Die SPI-Schnittstelle benutze ich, um mit dem MCP2515 zu kommunizieren.
- Die Serial-Schnittstelle (Pins 8 und 10) benötige ich nicht wirklich, sie soll nur zu nach außen geführt werden, um später weitere Funktionen einbauen zu können. Momentan ist allerdings nichts geplant.
- Auf den Pins 35 und 37 habe ich zwei Schalter verbaut. Diese dienen nur zum Debuggen. Man könnte auch noch weitere Funktionen implementieren, aber da die Platine später in einem Hutschienengehäuse untergebracht werden soll, ist das eher unpraktisch.
- Die Stromversorgung des RPi befindet sich auf den Pins 1 (+3,3V), 2 (+5V), 6 und 9 (GND). Hier ist jeweils noch ein Abblockkondensator verbaut, um hochfrequente Schwankungen in der Spannung zu verringern.
- Die restlichen Pins sind wie folgt belegt:
  - ✓ RPI_GPIO_05 – Pin 29: ist der Interrupt-Pin für den PCF8574. Wenn sich dessen Zustand ändert (also ein S0-Impuls detektiert wird), ist das am Interrupt-Pin zu erkennen.
  - ✓ RPI_GPIO_06 – Pin 31: ist der StandBy-Pin des MCP2562. Er kann später verwendet werden, um Strom zu sparen.
  - ✓ RPI_GPIO_13 – Pin 33: ist der Reset-Pin des MCP2515. Er hat vorerst noch keine Funktion.
  - ✓ RPi_GPIO_25 – Pin 22: ist der Interrupt-Pin des MCP2515. Falls eine CAN-Botschaft empfangen wird, ist das über diesen Pin zu bemerken.

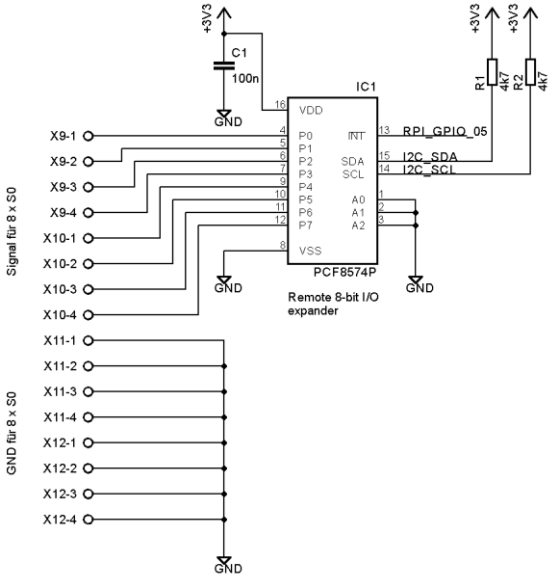
Als Bauteil habe ich einen 40-Poligen Wannenstecker verwendet. Die Kabel hierfür kann man sich mit wenig Aufwand selbst herstellen.

Stromversorgung



Die Stromversorgung gestaltet sich recht einfach – ich habe lediglich die Schaltung aus dem Datenblatt übernommen. Zu beachten ist, dass der Kondensator C8 recht groß (ziemlich hoch) ist. Das sollte man beim Design des Boards beachten, damit man nicht mit dem Gehäuse in Konflikt kommt.

PCF8574 – S0-Zähler



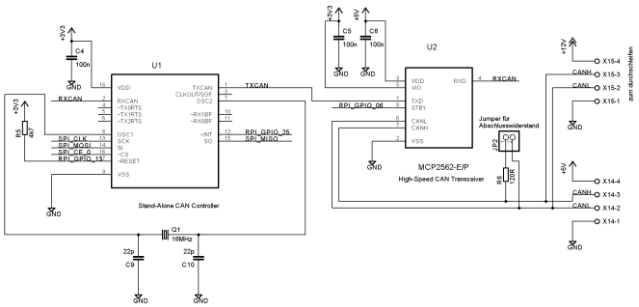
Da der RPi mit 3,3V arbeitet und alle Pins nur für 3,3V ausgelegt sind, habe ich mich dafür entschieden, ebenfalls 3,3V zu verwenden, jedenfalls dort wo möglich. Daher läuft auch der PCF8574 mit 3,3V. An der Spannungsversorgung ist hier wieder ein Abblockkondensator verbaut. Ansonsten ist hier gar nicht so viel Spannendes zu finden: Die Pins P0-P7 sind über Schraubklemmen nach außen geführt um dort die S0-Zähler anzuschließen. Damit man nicht mehrere Kabel an eine Schraubklemme fummeln muss, habe ich zusätzliche acht Schraubklemmen vorgesehen, die nur auf GND liegen. So kann ein S0-Zähler beispielsweise an X9-1 und X11-1 angeschlossen werden.

Der INT (Interrupt)-Pin wurde weiter oben schon kurz erklärt. Die Pins SDA und SCL sind für den I2C-Bus reserviert. Hier sind auch die Pull-up-Widerstände verbaut.

Die Pins A0, A1 und A2 sind für die Festlegung der I2C-Adresse gedacht. Je nachdem, wie sie angeschlossen werden (Low oder High) wird eine andere Adresse eingestellt. Das ist wichtig, wenn man mehrere PCF8574 verbauen will. Bis zu acht Stück sind an einem Bus möglich.

Weiteres zum PCF8574, speziell die Ansteuerung, werde ich im Abschnitt zur Programmierung erklären.

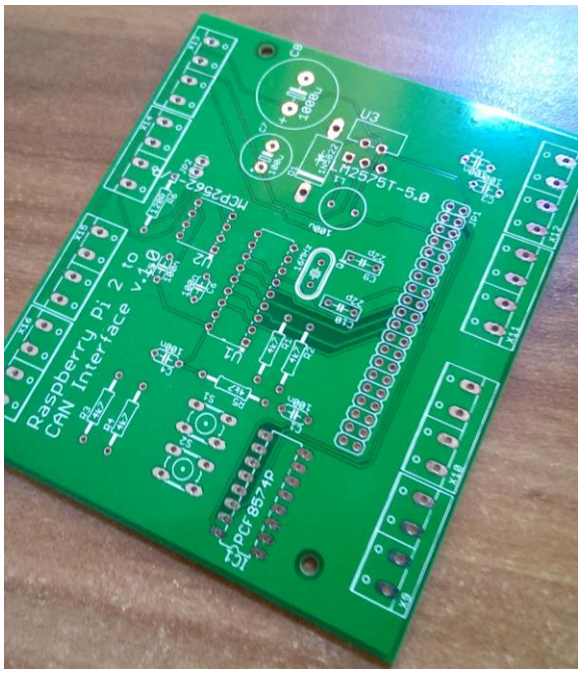
CAN-Interface



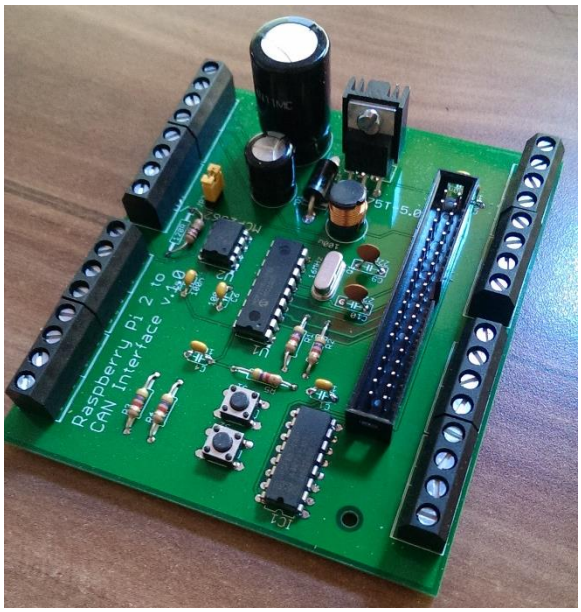
Das CAN-Interface besteht im Wesentlichen aus zwei Bauteilen, dem MCP2515 (CAN-Controller) und dem MCP2562 (CAN-Transceiver). Die Beschaltung ist wie im Datenblatt beschrieben. Zusätzlich ist ein Jumper verbaut, der es ermöglicht einen 120Ω-Abschlusswiderstand direkt auf dem Board zu nutzen.

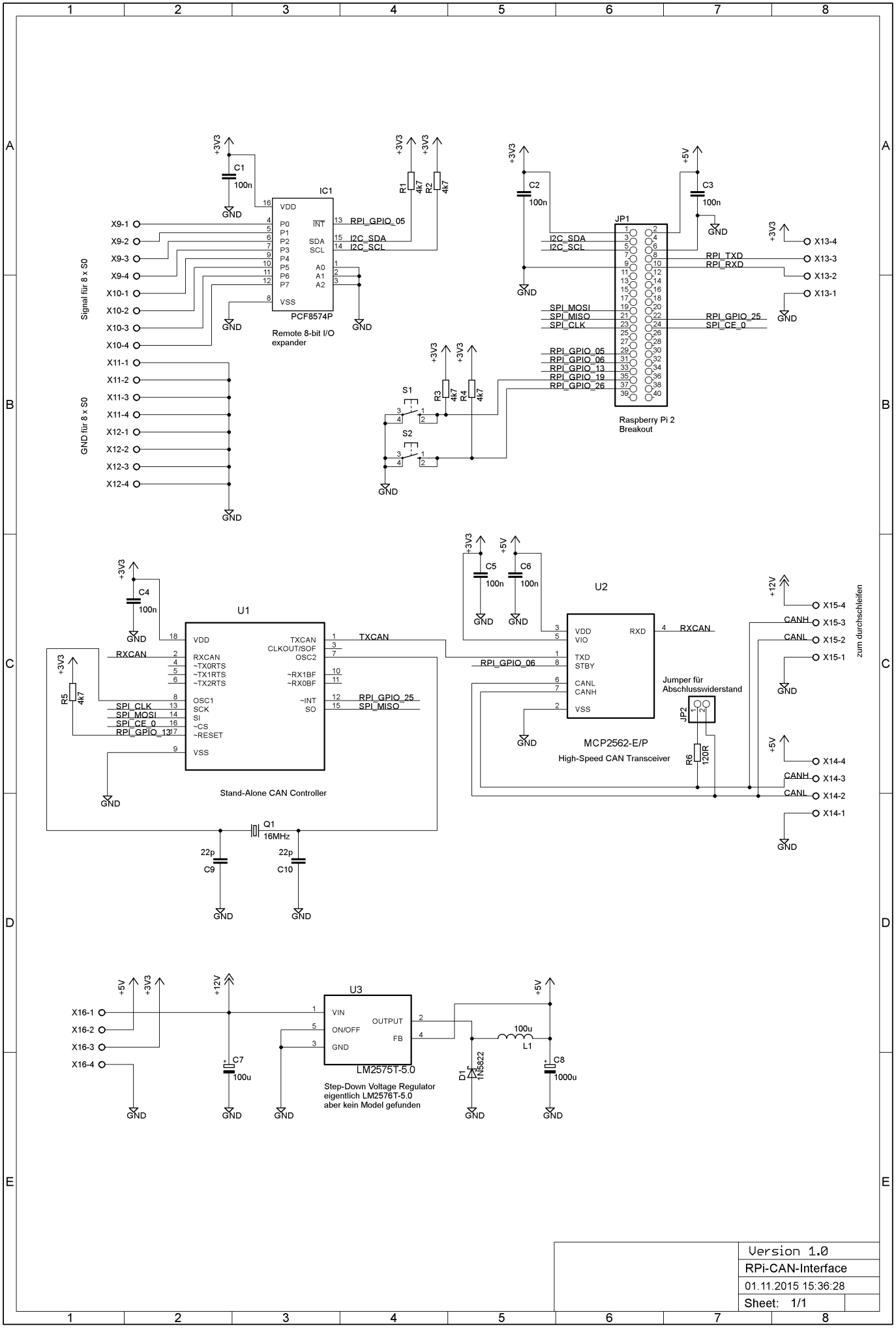
Board-Layout

Das Board habe ich mithilfe von EAGLE gebaut und in China fertigen lassen. Mit der Qualität bin ich sehr zufrieden. Hier die Ergebnisse:



Und bestückt:




IC1
PCF8574P
Remote 8-bit I/O expander
Signal für 8 x S0
GND für 8 x S0
JP1
Raspberry Pi 2 Breakout
S1
S2
U1
Stand-Alone CAN Controller
U2
MCP2562-E/P
High-Speed CAN Transceiver
Jumper für Abschlusswiderstand
zum durchschleifen
U3
LM2575T-5.0
Step-Down Voltage Regulator
eigentlich LM2576T-5.0
aber kein Model gefunden
Version 1.0
RPi-CAN-Interface
01.11.2015 15:36:28
Sheet: 1/1