ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

TIK Institut für Technische Informatik und Kommunikationsnetze

Bachelor Gruppenarbeit
am Departement für Informationstechnologie und Elektrotechnik

Mountain View
Eine Kamera für das PermaSense Projekt

Michel Müller, Stefan Kronig

Betreut durch:
Dr. Jan Beutel und Matthias Keller
Prof. Dr. Lothar Thiele

7. Januar 2009

Inhaltsverzeichnis

1 Einleitung

PermaSense ist ein gemeinsames Projekt der ETH Zürich und der Universitäten Zürich und Basel. Es verfolgt zwei Hauptziele [1]:

1. Bau und Konfiguration eines Sensornetzwerks, das in schwierig zu erreichenden Gebieten eingesetzt – und deshalb mittels einer Funktechnologie gesteuert – werden kann.

2. Sammeln von geologischen Daten zur Erforschung des Zusammenhangs zwischen der Klimaerwärmung und Gesteinsverschiebungen in Permafrostzonen.

Gegenwärtig sind zwei Sensornetzwerke auf den Standorten Jungfraujoch und Matterhorn aktiv. Die Sensorknoten, basierend auf der TinyNode-Plattform, bilden zusammen ein Ad-hoc-Funknetzwerk und bleiben dadurch in Verbindung mit einer Basisstation, die die Daten sammelt und mittels einer weiteren Übertragungstechnik (GSM, WLAN oder Ethernet) an einen ETH-Server sendet. Die Sensorknoten beschränken sich bis jetzt auf Temperatur-, Feuchtigkeits- und Druckmessungen.

Die Bandbreite an gesammelten Informationen soll nun verbreitert werden. Das Ziel unseres Projektes ist es, die bestehende Infrastruktur um ein System zur Erfassung und Übertragung von Bildern zu erweitern, damit in Zukunft auch optische Informationen zu Gesteinsverschiebungen ausgewertet werden können. Die Hardware dieses System hat in erster Linie folgenden Anforderungen zu genügen:

1. Möglichst hohe Auflösung und Farbtiefe, um auch in kontrastarmen Konstellationen differenzierte Informationen liefern zu können. Konkret sollte die Farbtiefe mehr als 8bit pro Farbkanal bieten.

2. Widerstandsfähigkeit gegenüber extremen Temperaturbedingungen (bis zu -30°C) und Windgeschwindigkeiten (bis zu 200 km/h), Verhinderung des Eindringens von Feuchtigkeit

3. Temperatur- und Feuchtigkeitsbedingungen im Gehäuse überwachbar

4. Notfalls steuern der Temperatur mittels einer Heizung möglich

5. Ferngesteurter “Reset” der Kamera möglich

6. Möglichst geringer Stromverbrauch im ausgeschalteten Zustand

7. Möglichst einfache Montage, insbesondere möglichst wenige Boxen und Verbindungskabel

An die Steuerung werden folgende Anforderungen gestellt:

1. Autonome Bilderfassung zu festgelegten Zeitpunkten möglich
2. Manuelle Kontrolle der Kamerafunktionen möglich

Ziel dieser Gruppenarbeit ist es, gemäss diesen Kriterien sowohl einen Prototypen der Hardware, als auch die Steuersoftware zu implementieren.

Schon aufgrund der ersten Anforderung an die Hardware wird klar, dass eine handelsübliche Webcam hier nicht genügen wird und stattdessen eine den Kriterien entsprechende Digitalkamera herangezogen werden muss. Zur Wahl der Plattform für die Steuerung einer Kamera und die Zwischenspeicherung der Bilder sind grundsätzlich zwei Möglichkeiten vorhanden: Basisstation oder Sensorknoten. Die Sensorknoten scheiden allerdings aufgrund der zu eingeschränkten Stromversorgung und Rechenleistung aus. Somit bleibt die Basisstation als einzige Möglichkeit für diese Aufgabe. Diese wird durch eine Solaranlage mit 12V-Batterie mit Energie versorgt und bietet genügend Rechenleistung.

Die Bilddaten sollen zunächst von der Kamera zur Basisstation übertragen werden, welche selber mittels einer SD-Memorycard als Pufferspeicher dienen kann. Danach können die Bilder mit der gleichen Übertragungstechnik wie die übrigen Messdaten zum Server gelangen. Ob die Übertragung ebenfalls autonom geschieht, oder auf Befehl des Benutzers erfolgt, steht zum Zeitpunkt der Projektplanung noch nicht fest. Deshalb muss davon ausgegangen werden, dass Vorschaubilder zur Ermöglichung der Auswahl benötigt werden.

2 Evaluation und Messungen

Nachfolgend werden die Abwägungen zu den am System beteiligten Komponenten dargelegt. Die Reihenfolge der Unterkapitel entspricht im Wesentlichen dem Gedankengang während den Evaluationen. Das Ziel dieses Kapitels ist die Erläuterung der gesammelten Erkenntnisse, die im Hinblick auf die Lösung von Bedeutung sind.

2.1 Die Steuerung und Datenspeicherung

Folgende Funktionen werden von den Steuerkomponenten gefordert:

1. Feststellen des freien Speicherplatzes auf der Kamera
2. Bei Bedarf Entfernen der ältesten Bilder, um freien Speicherplatz zu gewinnen
3. Auslösen eines Fotos
4. Erzeugung eines Vorschaubildes
5. Übertragung der Daten auf den lokalen Speicher der Basisstation

Die dafür verwendeten Komponenten werden im Verlauf dieses Kapitels erläutert.

2.1.1 Die Basisstation

Die Kernkomponente der Basisstation ist ein Embedded- PC des Typs “Gumstix Verdex Pro XM4” [2] mit 400MHz XScale Prozessor. Zum Testen stehen uns neben diesem Mainboard ausserdem Erweiterungsboards zur Verfügung, um USB, Netzwerk und SD Memory Cards benützen zu können.

Das verwendete Betriebssystem ist ein OpenEmbedded Linux (Kernel Version 2.6.21) mit integriertem Packetmanager (ipkg). Aufgrund der begrenzten Ressourcen (Speicherplatz und Rechenleistung) wird auf eine Kompilierumgebung verzichtet. Binäre Programme für die Basisstation müssen stattdessen auf einem Fremdsystem mittels einer emulierten Umgebung kompiliert werden.

2.1.2 Die Übertragungstechnik

Alle getesteten Kameras verwenden zur Übertragung von Daten auf den PC die USB-Schnittstelle, weshalb im Folgenden von dieser Übertragungstechnik ausgegangen wird. Auf der Seite der Basisstation wird für den Prototyp dafür die Erweiterungskarte "Breakout-vx" verwendet. Die Gumstix-Verdex-Plattform unterstützt allerdings lediglich USB 1.1 (12MBit/s).

Eine andere Option wären die für aktuelle Kameras erhältlichen WLAN-Module. Im Rahmen dieses Projekts ist allerdings keine grosse Distanz zwischen der Kamerabox und der Basisstation vorgesehen, so dass USB hier die einfachste Lösung darstellt.

2.1.3 Das Softwareinterface

Die meisten Digitalkameras unterstützen zur Kommunikation mit anderen Geräten (in der Regel PCs oder Drucker) das "Picture Transfer Protocol (PTP)", welches nicht nur zur Datenübertragung, sondern auch zum Senden von Steuersignalen an die Kamera verwendet werden kann.

Als Softwareschnittstelle, die sowohl die Datenübertragung als auch die Steuerung realisiert, eignet sich GPhoto. Dies ist eine Sammlung von freien Programmen zur Kommunikation mit Digitalkameras, welche in der aktuellen Version 2 mehr als 1000 Kameramodelle unterstützt. GPhoto ist architekturunabhängig und kann mittels der OpenEmbedded-Plattform auch auf dem für dieses Projekt verwendeten Embedded-System verwendet werden. Im Folgenden wird von der Verwendung dieser Software ausgegangen.

Zur Implementation der im Kapitel 2.1 erwähnten Kamerafunktionen stellt GPhoto folgende Shell-Aufrufe bereit:

- `gphoto2 --storage-info`
- `gphoto2 --list-files`
- `gphoto2 --delete-file`
- `gphoto2 --capture-image`
- `gphoto2 --get-thumbnail`
- `gphoto2 --get-file`

Um den Anforderungen bezüglich Farbtiefe zu entsprechen, muss es möglich sein, die Bilder in einem Rohformat speichern zu können. Dies erfordert Einstellungen, welche üblicherweise im Menü der Kamera vorgenommen werden. Selbst wenn diese ohne externe Stromversorgung [1] erhalten bleiben würden, wäre es bei einem Absturz der Kamera-Firmware denkbar, dass Einstellungen verloren gehen.

[1] der Akkumulator der Kamera darf nicht mit eingebaut werden, da sonst die Kamera nicht wie gewünscht ein- und ausgeschaltet werden kann (siehe Abschnitt 2.4)

Die Einstellungen der Kamera müssen also ebenfalls über die Steuersoftware vorgenommen werden können. Dies kann mittels dem GPhoto- Befehl `gphoto2 --set-config [VARIABLE]=[WERT]` realisiert werden, wobei die möglichen Konfigurationsvariablen und Werte mittels `gphoto2 --list-config` und `gphoto2 --get-config [VARIABLE]` herausgefunden werden können. Über dieses Interface lassen sich je nach Kameramodell diverse Einstellungen vornehmen, unter Anderem auch die Systemzeit der Kamera, um diese mit derjenigen der Basisstation synchronisiseren zu können.

Für alle hier aufgezeigten Möglichkeiten von GPhoto gilt die Einschränkung, dass die Software von den Herstellern nicht offiziell unterstützt wird und für die Realisierung einer ferngesteuerten Kontrolle von Digitalkameras kein einheitliches Protokoll existiert. Ob und wie diese Funktionalitäten also bei einem bestimmten Kameramodell vorhanden sind, hängt von den GPhoto- Entwicklern und der Implementation der Kamera-Firmware ab. Es ist also zu evaluieren, wie gut eine Kamera mittels GPhoto zu steuern ist und ob die essentiellen Funktionen umgesetzt werden können.

2.2 Die Kamera

2.2.1 Die Vorgaben

Die zu wählende Digitalkamera muss also folgende **Anforderungen** erfüllen:

- Möglichst hohe Auflösung und Farbtiefe, mehr als 8bit pro Farbkanal erforderlich
- Unproblematisches Verhalten bei plötzlichem Verlieren der Stromversorgung [2]
- Fernsteuerung mittels GPhoto möglich

Weitere Aspekte zur Wahl:

- Möglichst verlustfreie Anbindung an eine 12V- Stromversorgung möglich
- Wiederherstellung der Einstellungen ferngesteuert möglich
- Direkter Abruf von Vorschaubildern möglich (um Berechnungszeit auf der Basisstation zu sparen)

Uns wurden drei **Modelle** zum Testen zur Verfügung gestellt:

- Sony DSC-H2
- Nikon D70s
- Nikon D200

Die Sony scheidet bei näherer Betrachtung aus, da sie keine Möglichkeit bietet, über USB gesteuert Fotos auszulösen. Deshalb gehen wir im Folgenden von den beiden Nikon-Modellen aus.

[2] Ist nötig, da alle getesteten Kameras nur auf diesem Weg ausgeschaltet werden können. vgl. Abschnitt 2.4

2.2.2 Tests und Messungen

In der Tabelle 2.1 befinden sich alle für das Design des Prototyps wichtigen Resultate der Tests und Messungen an den Kameras. Abbildungen 2.1 - 2.4 zeigen einige ausgewählte Stromverläufe im Zeitbereich. Für einen tieferen Einblick in den Verlauf der Messungen sei auf das Messprotokoll im Anhang verwiesen.

Tabelle 2.1: Messungen und Tests an den Kameras

	Nikon D70s	**Nikon D200**
Hardware		
Auflösung[3]	3'008 * 2'000 (6.1 MP)	3872 * 2,592 (10.0 MP)
Farbtiefe[3]	12bit / Kanal	12bit / Kanal
Betriebsspannung nominell	9V	13.5V
Betriebsspannung gemessen am Herstellernetzteil	≈9.4V	≈13.7V
Betrieb mit 12V getestet	nein	ja, Erfolg
Stromverbrauch Standby[4]	≈0.16A ($U_{in} \approx 9.4V$) (vgl. Abb. 2.1)	≈0.265A ($U_{in} \approx 12V$) (vgl. Abb. 2.3)
Leistungsaufnahme Standby[4]	1.5W	3.18W
Energiesparmodus im Netzbetrieb[5]	nein	nein
Stromspitze bei der Bildaufnahme[6]	3.49A (vgl. Abb. 2.2)	2.59A (vgl. Abb. 2.4)
Test Ein-/ Ausschalten[7]	Erfolg, 124 Fotos	Erfolg, 100 Fotos
Software		
Unterstützung GPhoto- Funktionen	ja[8]	ja[8]
Abruf von Vorschaubildern möglich	ja	ja

[3] Um die maximale Bildqualität zu erhalten, muss das Speicherformat der Kamera auf "raw" gestellt werden. Nikon verwendet dafür proprietäre "NEF"-Dateien (Nikon Electronic Format).

[4] Mit Standby ist hier der eingeschaltete Betrieb ohne Aktivität gemeint.

[5] Um dies zu testen, wurden die Kameras einige Stunden ohne Aktivität eingeschaltet gelassen und der Verlauf des Stromverbrauchs aufgezeichnet.

[6] Die Aufnahme-Stromspitze war in allen Messungen die Höchste.

[7] Hier wurde mittels eines Power Analysers die Kamera eingeschaltet, ein Photo mittels eines Skriptes ausgelöst, dann wieder ausgeschaltet, also die für das Projekt benötigte Arbeitsweise simuliert.

[8] Die essentiellen GPhoto- Funktionen, einschliesslich den Einstellungen zur Bildqualität und -format arbeiten auf beiden Kameras ohne Fehler. Hingegen wurde ein Problem bei der Einstellung der Kamerazeit festgestellt. Der Befehl bleibt ohne Wirkung, es wird aber auch keine Fehlermeldung angezeigt (siehe auch Kapitel 4.4).

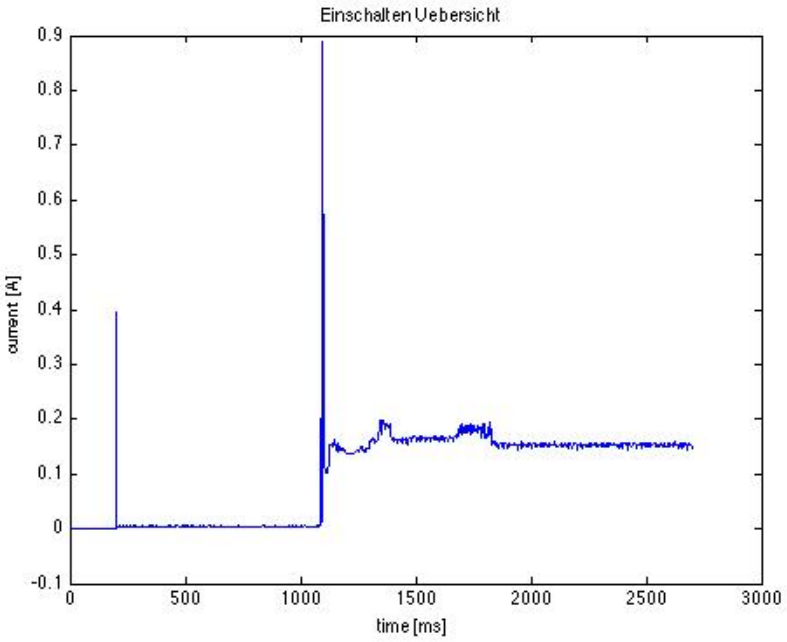


Abbildung 2.1: Stromverlauf eines D70s- Einschaltprozesses mit $U_{in} = \sigma(t) \cdot 9.4V$

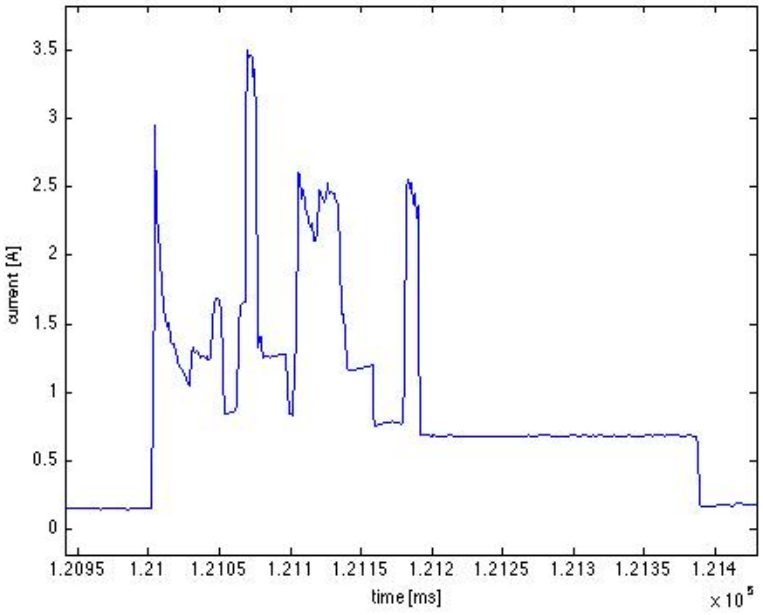


Abbildung 2.2: Stromverlauf einer D70s während des Auslösens eines Fotos ($U_{in} = 9.4V$)

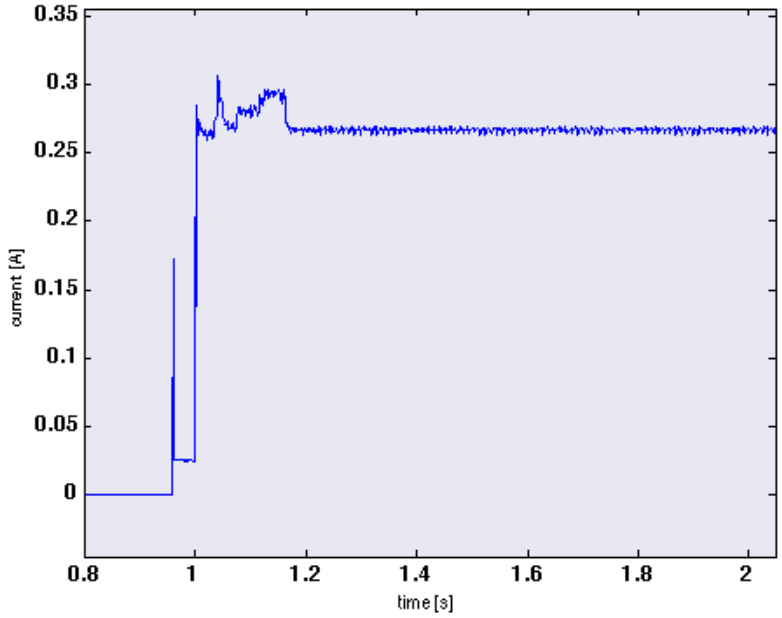


Abbildung 2.3: Stromverlauf eines D200- Einschaltprozesses mit $U_{in} = \sigma(t) \cdot 12V$

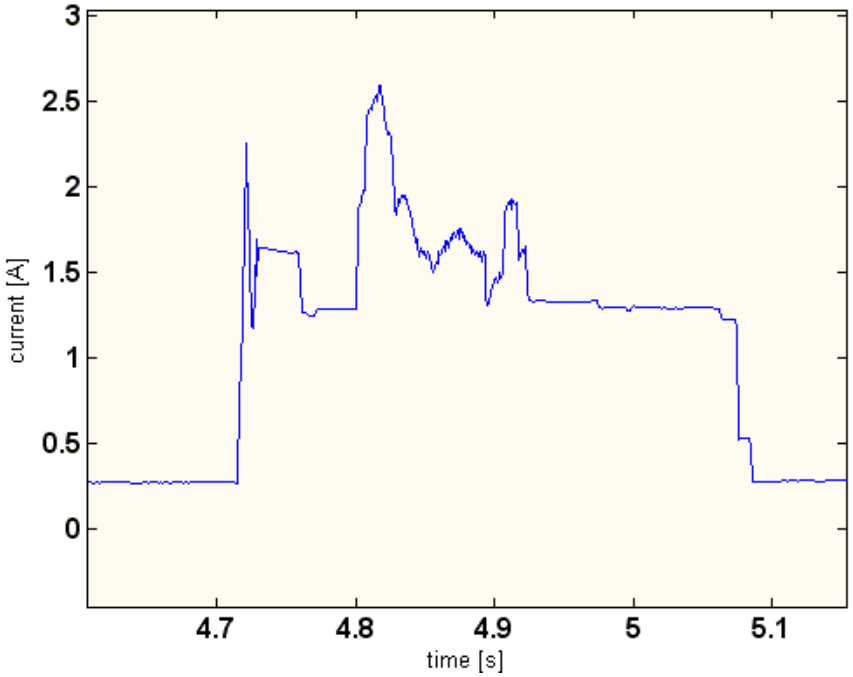


Abbildung 2.4: Stromverlauf einer D200 während des Auslösens eines Fotos ($U_{in} = 12V$)

2.2.3 Fazit

Sowohl die D200, also auch die D70s erfüllen die grundsätzlichen Anforderungen. Die D200 schneidet bei der Auflösung und der einfachen Anbindung an die 12V-Stromversorgung besser ab, die D70s beim Verbrauch. Der Vorteil der D70s muss aber relativiert werden, da für die Erzeugung der benötigten 9.4V ein Spannungswandler eingesetzt werden muss, was einerseits die Komplexität des Systems und andererseits den Verbrauch wieder erhöht.

Schlussendlich wurde für den weiteren Bau des Prototyps zugunsten der D200 entschieden. Aus den Messungen wird ersichtlich, dass die Kamera auch im inaktiven Zustand (eingeschaltet, ohne Aktivität) eine für das Einsatzgebiet zu hohe Leistungsaufnahme aufweist. Aus diesem Grund muss die Kamera in jedem Fall bei Nichtgebrauch abgeschaltet werden. Die zu erstellende Schaltung muss aufgrund der Messungen mindestens 0.3A kontinuierlichen Strom und 3A kurzzeitige Stromspitzen am Ausgang unterstützen können.

2.3 Die Überwachung und Steuerung von Temperatur und Feuchtigkeit im Gehäuse

Als Gehäuse für die Kamera wurde ein für Ausseneinsätze geeignetes Gehäuse gewählt. Es ist der gleiche Typ Gehäuse, welcher bereits für die Basisstation und die Sensorknoten verwendet wird. Die Planung der daran nötigen physischen Modifikationen fällt allerdings nicht mehr in den Rahmen dieser Gruppenarbeit. Das folgende Kapitel beschäftigt sich deshalb lediglich mit den geforderten Massnahmen (siehe Kapitel 1) zur Überwachung und Steuerung von Temperatur und Feuchtigkeit im Gehäuse.

Um Fehlfunktionen durch Kälte oder Feuchtigkeit (z. B. beschlagene Scheibe) entgegenzuwirken, werden Temperatur und Feuchtigkeit im Gehäuse überwacht und geregelt. Die Überwachung geschieht mit einem Sensorknoten[9], welcher in das Gehäuse integriert wird. Die Sensorknoten haben die Funktion der Temperatur- und Feuchtigkeitsüberwachung innerhalb ihres eigenen Gehäuses bereits implementiert. Durch die Nutzung des bereits bestehenden Systems kann diese Überwachung einfacher in das gesamte Projekt integriert werden, als wenn man die Sensoren über USB anschliessen würde.

Um die Temperatur nicht nur überwachen, sondern auch regeln zu können, wird eine (bereits vorgegebene) 12V-Elektroheizung in das Gehäuse integriert, für die ebenfalls eine Schaltung entsprechend erstellt werden muss. Dabei handelt es sich um eine PTC-Heizung, die ihre Temperatur selbst regelt (basierend auf den Eigenschaften des verwendeten Halbleiters). Gemäss eigenen Messungen benötigt die Heizung beim Einschaltvorgang (auf Zimmertemperatur) kurzzeitig fast 5A. Danach flacht der Strom exponentiell ab und strebt gegen ≈0.9A. Abbildung 2.5 Zeigt diesen Stromverlauf[10].

[9] Nur die Elektronik, ohne externe Sensoren

[10] Für die Dimensionierung der Schaltung darf also nicht der im Datenblatt [3] angegebene Maximalwert von 18W zur Berechnung der Stromspitze herangezogen werden. Dieser ergäbe eine Spitze von ≈1.5A bei 12V-Versorgung. Der im Datenblatt angegebene Wert ist ein theoretischer Maximalwert für die kontinuierliche Ausgangsleistung bei sehr tiefen Aussentemperaturen.

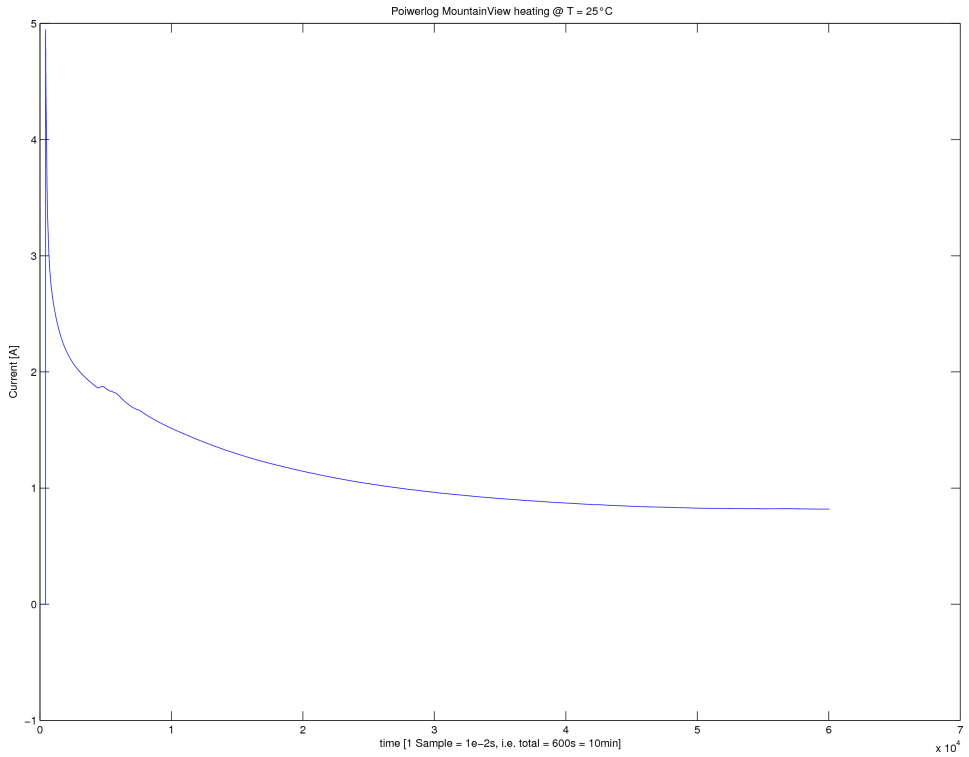


Abbildung 2.5: Stromverlauf beim Einschaltvorgang der Heizung mit $U_{in} = \sigma(t-4.3s) \cdot 12V$ und $T_{begin} \approx 300K$ während 10 Minuten.

2.4 Das ferngesteuerte Ein- und Ausschalten

Neben der Heizung muss auch die Kamera aus zwei Gründen schaltbar sein: Einerseits wäre sonst der Verbrauch gemäss Abschnitt 2.2.3 zu hoch, andererseits muss für die Kamera ein ferngesteuerter "Reset" möglich sein (siehe Kapitel 1). Da der Hauptschalter der Kamera mechanisch funktioniert, wird dies so gelöst, dass dieser immer eingeschaltet bleibt und die Stromversorgung getrennt wird, sobald die Kamera ausschalten soll (der Akkumulator der Kamera darf zu diesem Zweck nicht eingelegt sein). Es muss also eine Lösung für das getrennte Ein- und Ausschalten der Kamera und Heizung gefunden werden.

Um eine möglichst einfache Montage realisieren zu können, soll die Kamerabox möglichst wenige Anschlüsse aufweisen. Deshalb soll das Schalten über die USB-Schnittstelle realisiert werden, die auch für die Datenübertragung verwendet wird. Demzufolge wird auch ein USB-Hub benötigt.

Es wurden folgende zwei Möglichkeiten evaluiert, um über USB einzelne Leitungen ein- und ausgeschalten zu können:

- Es kann ein USB-to-Serial-Adapter verwendet werden[11]. Diese Möglichkeit hat allerdings den Nachteil, dass die gesetzten Werte nur erhalten bleiben, wenn ein (speziell dafür geschriebener) Prozess die serielle Schnittstelle durchgehend für sich beansprucht und die Leitungen entsprechend steuert.

- Es kann eine GPIO[12]-Schnittstelle verwendet werden. Diese Lösung hat den Vorteil, dass die Pins ihren Wert auch behalten, wenn kein Prozess darauf zugreift[13] oder der PC neu gestartet werden muss.

Sowohl die Serielle Schnittstelle als auch die GPIOs sind "active low", d.h. nach dem Einschalten des CP2103 liegt an den Ausgängen eine Spannung an und nach der Initialisierung der seriellen Schnittstelle, respektive einem entsprechenden GPIO-Befehl, werden die Pins geerdet. Somit muss das Signal an den Eingängen invertiert werden, um ein Einschalten der Kamera und Heizung beim Anschliessen der Komponenten zu verhindern.

Schlussendlich wurde zugunsten einer Steuerung mittels GPIOs entschieden. Das dafür verwendete Gerät ist ein CP2103 von Silicon Labs. Dies ist ein USB-to-UART Adapter, der zusätzlich 4 GPIOs bereitstellt [5]. Zudem wandelt der CP2103 intern 5V in 3.3V um. Diese Spannung kann auch zur Speisung anderer Geräte verwendet werden.

[11] Man kann, z. B. mit PySerial [4], die RTS- und DTR-Leitungen schalten.
[12] General Purpose Input/Output - generische Ein- und Ausgabe
[13] Im Kapitel 4.2 wird ersichtlich, wieso die Kamera bei Beenden des darauf zugreifenden Prozesses nicht abschalten darf.

3 Auslegung der Schaltung

3.1 Die Vorgaben

Ausgehend von den Messungen und Evaluationen im Kapitel 2 sowie den Anforderungen in Kapitel 1 muss also eine Schaltung zum Ein- und Ausschalten der Kamera und der Heizung entwickelt werden. Zusammenfassend ergeben sich dazu folgende Vorgaben:

1. Schaltung der Kamera (Nikon D200): 12V, Stromverbrauch ≈0.3A kontinuierlich, ≈3A Stromspitze
2. Schaltung der Heizung: 12V, Stromverbrauch bis zu 5A kontinuierlich
3. Als Eingänge werden die GPIO-Signale verwendet. Diese sind "active low", müssen also invertiert werden (siehe auch Abschnitt 2.4).
4. Vorhandene Versorgungsspannungen sind: 12V, 5V (USB), sowie 3.3V, die der CP2103 bereitstellt. Die Schaltung soll daher mit diesen Spannungen arbeiten.
5. Möglichst geringer Stromverbrauch der Schaltung (namentlich der Inverter). Konkret: Max. 1mA im ausgeschalteten Zustand.
6. Beide Ausgänge sollen ausgeschaltet sein, wenn USB nicht angeschlossen ist. Dies macht nicht nur dann Sinn, wenn die Kamera gerade montiert wird und USB noch nicht verbunden ist, sondern kann auch hilfreich sein, falls der USB-Hub oder der CP2103 ausfallen sollte. USB kann nämlich an der Basisstation "virtuell ausgesteckt" werden, sprich die Versorgungsspannung des USB kann ausgeschaltet werden, um somit notfalls die Geräte zu deaktivieren.

Es wurde zugunsten einer Schaltung der Ausgänge mittels Power-MOSFETs entschieden (als Alternative zu Relais).

Im weiteren Verlauf dieses Kapitels folgt eine Beschreibung der Schaltung sowie die Überlegungen dazu. Abbildung 3.1 zeigt das aktuelle Schema[1]. Die im Rahmen dieser Gruppenarbeit gefertigte Schaltung ist noch ein Prototyp, daher dienen einige der Komponenten nur dem "Debugging". Sie sind im Schema blau eingezeichnet und werden in 3.4.5 genauer erläutert. Anschliessend folgen im Abschnitt 3.4 die Überlegungen zu den einzelnen Komponenten in Form einer Liste und in Abschnitt 3.5 eine (foto-)grafische Übersicht.

[1] Eine grössere Version ist im Anhang auf Seite 65 zu finden, die Originaldateien befinden sich auf der CD-ROM im Verzeichnis "schematic".

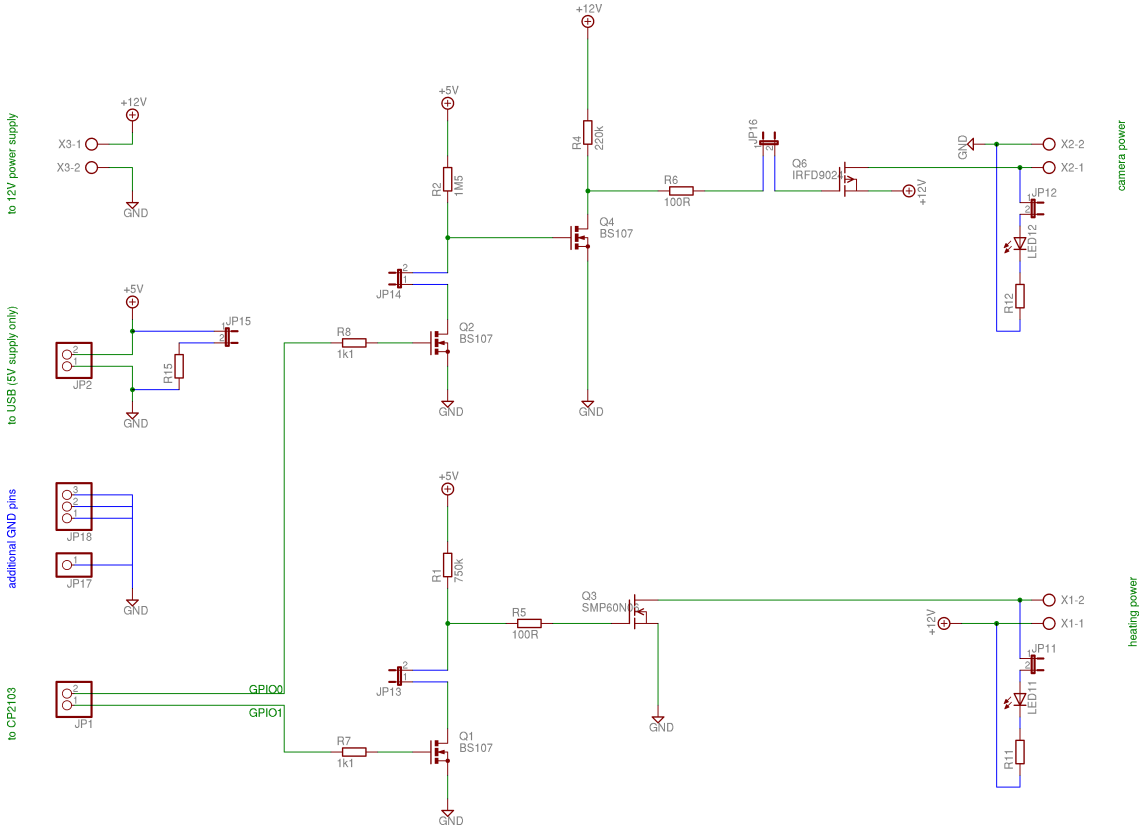


Abbildung 3.1: Schema des Prototyps. Die blau eingezeichneten Komponenten dienen dem “Debugging”

3.2 Heizungssteuerung (GPIO1, ungerade Nummern im Schema)

Q1-R1 bilden einen Inverter, da die GPIOs "active low" arbeiten. Die Signale der GPIOs müssen invertiert werden, da sonst sowohl Kamera als auch Heizung einschalten würden, wenn die Schaltung an die Stromversorgung angeschlossen wird (Vorgabe 3). Die Heizung wird "low-side", also an der Erdung, mittels Q3 geschaltet. Der 12V-Anschluss ist immer verbunden.

3.3 Kamerasteuerung (GPIO0, gerade Nummern im Schema)

Die Kamera muss – im Gegensatz zur Heizung – "high-side" geschaltet werden. Eine "low-side"-Schaltung (mit nur einem Transistor) ist nicht möglich, da die Kamera über die USB-Schnittstelle eine durchgehende Verbindung zu GND aufweist und somit immer eingeschaltet bleiben würde. Demzufolge wird ein P-Kanal MOSFET benötigt.

Da der P-Kanal MOSFET sperrt, wenn V_{GS} 12V (also "logisch high") beträgt und leitet, wenn V_{GS} einen gewissen Threshold unterschreitet (sprich: "logisch low" ist), kann direkt der (logische) Wert des GPIO zur Steuerung verwendet werden. Jedoch muss die Höhe der Steuerungsspannung angepasst, nämlich von 3.3V auf 12V erhöht, werden. Dies geschieht mit den beiden Invertern Q2-R2 und Q4-R4, wobei R4 an 12V und R2 an 5V angeschlossen wird.

R4 beträgt lediglich 220kΩ (im Vergleich zu den anderen Widerstände in der Grössenordnung MΩ), da bei höheren Widerständen V_{GS} an Q6 unter dessen Threshold-Spannung liegen würde. Der ("on-state-")Verluststrom ist auch so $\ll$ 1mA, und ist somit im Vergleich zum kontinuierlichen Stromverbrauch der Kamera vernachlässigbar.

3.4 Überlegungen zu den einzelnen Komponenten

3.4.1 Leitungen (Knoten) im Schema

+5V (Anschluss: JP2, rote Litze): 5V vom USB. Wird verwendet, damit die Ausgänge abschalten, wenn USB nicht verbunden ist (siehe Vorgabe 6). Der 3.3V-Ausgang des CP2103 kann nicht verwendet werden, da diese Spannung unter der Schwellspannung von Q3 liegen würde.

+12V (Anschluss: X3, gelbe Litze): Versorgungsspannung für Kamera und Heizung, verbunden mit der Versorgung der Basisstation. Beide zu schaltenden Geräte sind für eine Spannung von 12V geeignet (vgl. Vorgaben 1 und 2).

GND (schwarze Litze): Die Erdungen aller Geräte sind miteinander verbunden. Da keine galvanische Trennung benötigt wird, ist dies die einfachste Lösung.

3.4.2 Anschlüsse (JP, X)

JP1 Anschluss der GPIO Pins des CP2103. Hier ist keine GND-Verbindung nötig, da die Schaltung und der CP2103 schon mittels USB-GND miteinander verbunden sind.

JP2 USB-Anschluss. Es werden nur 5V und GND verwendet, die Datenleitungen sind nicht verbunden.

X1 Anschluss der Heizung. X1-1: Immer mit +12V verbunden; X1-2: geschalteter GND ("low-side switch").

X2 Anschluss der Kamera. X2-1: geschalteter +12V; X2-2: Immer mit GND verbunden ("high-side switch").

X3 Anschluss der 12V-Stromversorgung

3.4.3 Widerstände (R)

R2 1.5 MΩ Widerstände. Dies sind die höchsten zur Verfügung stehenden, um den Verlustrom im ausgeschalteten Zustand zu minimieren[2]. Der Verlustrom befindet sich so im μA-Bereich.

R1, R4 Können nicht so hoch gewählt werden wie R2, da sonst die Thresholdspannungen unter- (Q3) bzw überschritten (Q6) würden.

R5, R6 Widerstände vor den Gates der Power-MOSFETs, um den maximalen Strom durch Q1 bzw. Q4 beim Laden und Entladen der Gates zu begrenzen. Im Allgemeinen haben Power-MOSFETs eine hohe Gate-Kapazität, daher werden diese Widerstände empfohlen [6].

R7, R8 Widerstände zum Schutz der Gates von Q1 und Q2. Diese Widerstände werden in [6] bei Leitungen empfohlen, welche zu einem externen Teil der Schaltung führen (in diesem Fall zum CP2103).

3.4.4 Transistoren (Q)

Die hier angegebenen Spezifikationen beziehen sich jeweils auf Raumtemperatur (T_A = 25°C). Für detailliertere Informationen sei auf die jeweiligen Datenblätter verwiesen.

BS107 (Q1, Q2, Q4) Kleinsignal N-Kanal MOSFETs, die als Logik-Bauteile verwendet werden (sprich: es fliessen keine hohen Ströme durch den Drain). Sehr hoher Drain-Source Widerstand im gesperrten Zustand ($\gg$ 1.5 MΩ). I_D max. 0.12A (kontinuierlich); Datenblatt unter [7].

SUP60N06 N-Kanal Leistungs-MOSFET, der den Minus-Pin der Heizung steuert. I_D max. 50A (kontinuierlich), erfüllt also Vorgabe (2). Datenblatt unter [8].

IRFD9024 P-Kanal MOSFET, um die positive Versorgungsspannung der Kamera zu steuern. Gate-Source-Widerstand in der Grössenordnung 10 MΩ. I_D max. 1.5A (kontinuierlich), 13A (Stromspitze), erfüllt also Vorgabe (1); Datenblatt unter [9].

Alle Transistoren sind für Temperaturen von -55°C – 150°C geeignet.

3.4.5 "Debug"-Komponenten (Nummern $\geq$ 10), blaue Leitungen im Schema

LED11, JP11, R11 Einschalt-Anzeige (LED mit Vorwiderstand) für die Heizung. Mit dem Jumper kann die LED für Verbrauchsmessungen deaktiviert werden.

LED12, JP12, R12 Einschalt-Anzeige für die Kamera. Analog LED11.

JP13 Unterbruch zur Messung des Drain-Stromes durch Q1. Normalerweise verbunden.

JP14 Analog JP13

JP16 Unterbruch zur Messung des Gate-Stromes durch Q6. Normalerweise verbunden.

R15, JP15 Ein Verbraucher, mit dem Zweck, die 5V-USB-Versorgungsspannung abzubauen, falls USB ausgesteckt ist. Die USB-Spannung muss dann geerdet sein, damit die Power-MOSFETs sperren, wenn USB nicht verbunden ist (vgl. Vorgabe 6). JP15 dient zum deaktivieren dieses Features. Bisherige Messungen

[2] "ausgeschaltet" heisst hier, dass das angeschlossene Gerät (Heizung oder Kamera) ausgeschaltet ist, nicht etwa der Transistor. In Bezug auf einzelne Transistoren werden die Begriffe "leitend" oder "sperrend" benutzt.

des Spannungsverlaufes mit offenem Jumper haben jedoch die Notwendigkeit dieser Verbindung nicht bestätigen können.

JP17, JP18 Zusätzliche GND-Pins, um GND verschiedener Geräte zu verbinden oder Spannungen zu messen.

3.5 Übersicht zum Prototyp

Abbildung 3.2 zeigt eine Aufsicht des Prototyps mit Bezeichnungen der einzelnen Elemente. Abbildung 3.3 zeigt die Pinbelegung der Anschlüsse, darin nicht bezeichnete Pins sind nicht verbunden. Die "Debug"-Komponenten (vgl. 3.4.5) sind hier gelb eingezeichnet.

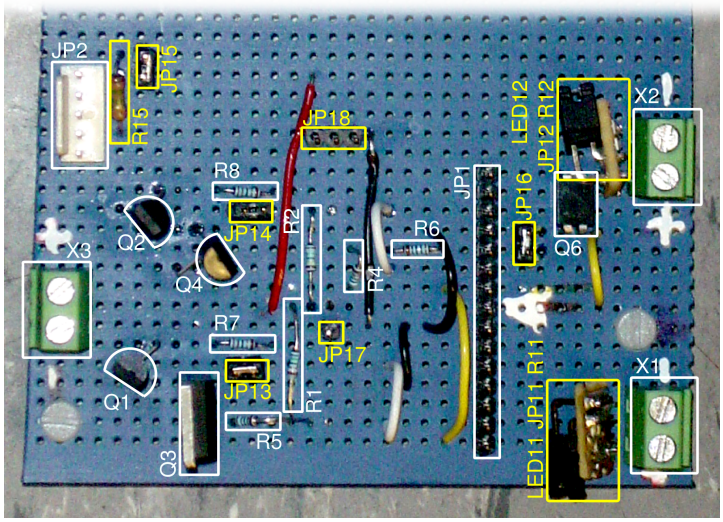


Abbildung 3.2: Foto des Prototyps mit Bezeichnungen der einzelnen Komponenten

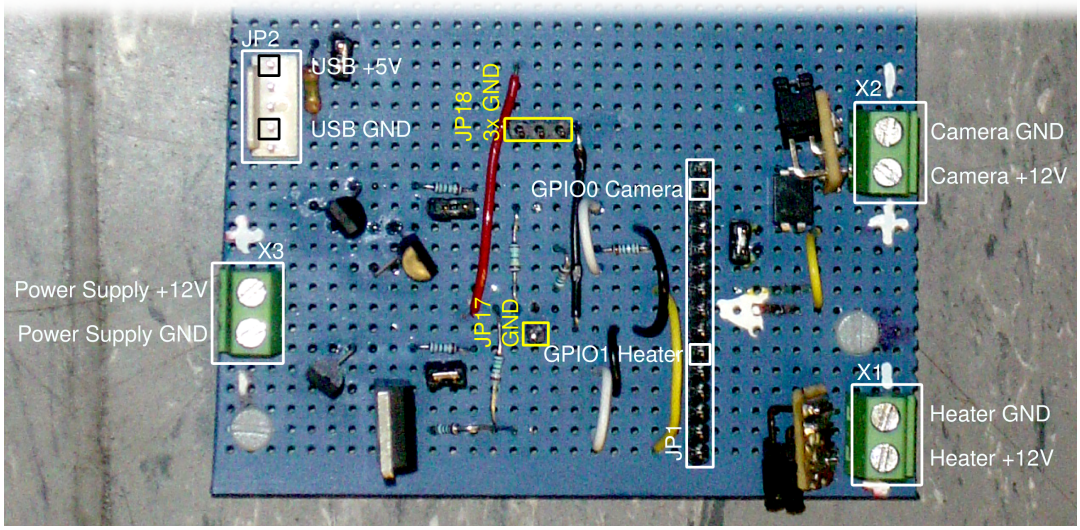


Abbildung 3.3: Foto des Prototyps, Pinbelegungen der Anschlüsse

3.6 Weitere Überlegungen und bekannte Probleme

3.6.1 Warum nicht zwei identische Schaltungen?

Die Frage ist insofern berechtigt, als dass eine Schaltung (in doppelter Ausführung) den Planungs- und Testaufwand halbieren würde. Die Schaltung der Kamera kann allerdings nicht einfach für die Heizung kopiert werden, da der P-Kanal MOSFET (IRFD9024) nicht solch hohe kontinuierliche Ströme am Drain unterstützt. Denkbar wäre eine Lösung mit zwei identischen, stärkeren P-Kanal MOSFETs.

3.6.2 Bekannte Probleme

Messungen haben ergeben, dass der Heizungskanal beim Ausschalten einige μs lang schwingt. Da dies für die Heizung kein Problem darstellt (vgl. Tests im Kapitel 5.3), ist dieses Verhalten so belassen worden. Die Messung ist in Abb. 3.4 zu sehen.

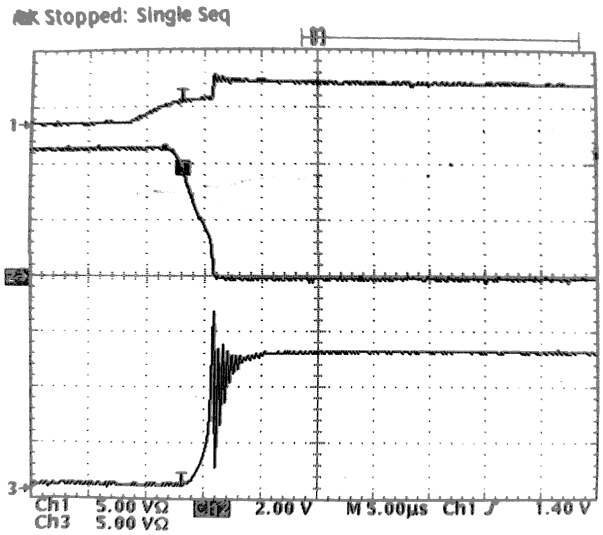


Abbildung 3.4: Spannungsverläufe beim Ausschalten der Heizung:
Ch1: am Eingang GPIO1, Ch2: an JP13, Ch3: am Ausgang X1-2

Ein weiteres Problem ist, dass die Gates von Q1 und Q2, trotz Vorwiderständen, sehr empfindlich gegenüber Potentialunterschieden und elektrostatischen Entladungen sind. Zur Vorbeugung gegen Schäden lassen sich, wie auf Abbildung 3.5 gezeigt, die Gates mit GND verbinden, wenn die Schaltung nicht benutzt wird. Zu beachten sind in diesem Zusammenhang auch die Hinweise unter 5.1.

3.6.3 Ausblick

Für eine definitive Schaltung wird empfohlen, die gesamte Schaltung (inkl. USB-Hub und CP2103) auf einer einzigen Platine zu realisieren. Dies würde die

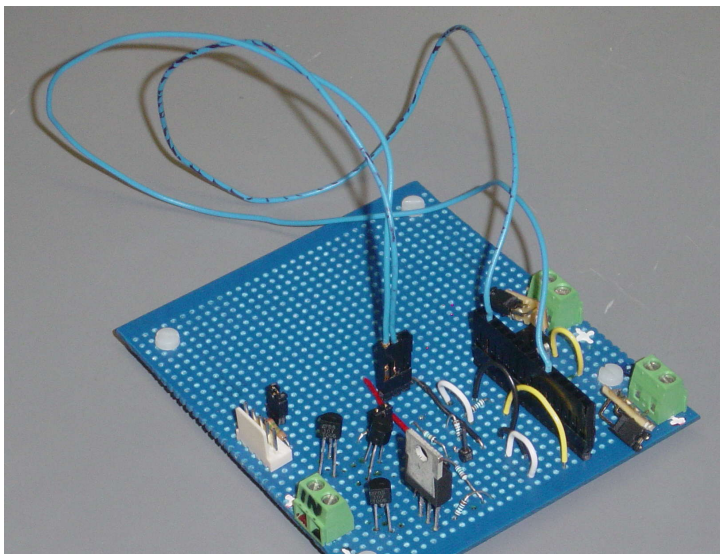


Abbildung 3.5: Schutz des Prototyps vor Elektrostatischen Entladungen (ESD)

Wahrscheinlichkeit von Problemen (insbesondere durch defekte Anschlüsse und Verpolung) wesentlich verringern.

4 Auslegung des Steuerungsprogramms

4.1 Die Vorgaben

Die Softwarekomponenten haben folgende Aufgaben und Anforderungen zu erfüllen (vgl. Kapitel 1):

1. Es sollen in periodischen Zeitabständen Bilder automatisch erfasst und gespeichert werden können.
2. Die Kamera soll bei Bedarf über das Ansteuern des entsprechenden GPIO ein- und ausgeschaltet werden.
3. Korrigierende Eingriffe in den automatischen Ablauf sollen möglichst einfach zu realisieren sein.
4. Das Erfassen und Speichern von Bildern soll auch manuell möglich sein. Im manuellen Modus sollen möglichst alle steuerbaren Funktionen der Hardware zugänglich sein.
5. Da der Ablauf zum Abrufen der Daten noch nicht feststeht, müssen Vorschaubilder bereitgestellt werden, um selektives Herunterladen zu ermöglichen.
6. Exaktes protokollieren der Abläufe ist erforderlich, um Fehler auffinden zu können.
7. Die Heizung soll manuell ein- und ausgeschaltet werden können.

4.2 Frühe Entscheidungen zum Softwaredesign

Aufgrund der obigen Vorgaben und den bereits in Kapitel 2 evaluierten Möglichkeiten ergeben sich folgende (frühe) Entschlüsse im Bezug auf die Ausarbeitung der Software:

- Zur Steuerung der Kamerafunktionen soll gemäss Kapitel 2.1.3 GPhoto verwendet werden.
- Das Ein- und Ausschalten wird mittels des Treibers der USB-to-GPIO-Komponente (CP2103) zu realisieren sein (siehe auch Kapitel 2.4).

- Aufgrund von Anforderung (3) wird ersichtlich, dass die Abarbeitung der Befehle mittels eines Skriptes erfolgen soll. Dies ermöglicht einerseits schnelle Änderungen am Ablauf (kompilieren wäre, wie in Kapitel 2.1.1 erwähnt, nur in emulierten Umgebungen möglich), andererseits sind die Geschwindigkeitseinbussen im Vergleich zu kompilierten Programmen vernachlässigbar, denn das begrenzende Element im Bezug auf die Geschwindigkeit sind die GPhoto-Befehle. Als Skriptsprache wird "Bourne-Shell"-Skript eingesetzt.

Aus den Anforderungen (1) und (2) ergibt sich eine erweiterte Problemstellung: Ab dem Zeitpunkt des Einschaltens der Kamera dauert es eine zunächst unbestimmte Zeit, bis sie als USB-Gerät von der Basisstation erkannt wird. Ein Skript, das sowohl das Einschalten der Kamera, als auch die nachfolgenden GPhoto- Aktionen ausführt, müsste also entweder den Zustand des Geräts "pollen" oder eine zu definierende Zeitperiode warten, nach der die Kamera mit sehr grosser Wahrscheinlichkeit gefunden wird.

Aktuelle Linux-Systeme (wozu auch das OpenEmbedded-Betriebsystem der Basisstation zählt) bieten für solche Aufgaben eine Alternative: Beim Erkennen eines neuen Gerätes durchläuft ein Gerätemanager (genannt `udev`) eine Reihe von Regeln, die die Einbindung des Gerätes bestimmen (z. B. den Namen des Geräts unter `/dev`). Diese Regeln lassen sich editieren, um beispielsweise ein Gerät mittels einer zu bestimmenden, eindeutigen Identifizierung immer unter dem gleichen Namen einbinden zu können. Zudem lassen sich beliebige Skripte definieren, die nach der Einbindung ausgeführt werden sollen.

Es eignet sich also, das Anschalten der Kamera von den übrigen Aktionen zu trennen, um diese erst mittels `udev` bei Erkennung des Geräts zu starten. Dieses Verfahren ist nicht abhängig von Zeitfaktoren und spart die eigene Implementierung einer Statusabfrage der Kamera.

Aus dieser Zweiteilung der Software ergibt sich die Notwendigkeit, dem von `udev` ausgelösten Skript die vom Benutzer gewünschten Aktionen zu kommunizieren (manuelles Benutzen der Kamera oder reguläre Abarbeitung des Skriptes). Die dafür einfachste Möglichkeit ist das Erzeugen einer Datei, die den Modus des Kameraskriptes bestimmt.

Zur Verhinderung des gleichzeitigen Zugriffs mehrerer Benutzer auf die Kamera (z. B. bei einer Konkurrenzierung des automatischen Aufrufs mit einem manuellen) wurde ausserdem ein Mutex implementiert.

4.3 Die beteiligten Komponenten

Im Folgenden werden die beteiligten Skripte und Komponenten erläutert. Die Abbildung 4.1 stellt das Zusammenarbeiten der einzelnen, im Text erklärten Komponenten schematisch dar. Dabei wurde in der Darstellung auf das Anzeigen einiger der beteiligten Dateien verzichtet, um die Grafik übersichtlich zu halten. Ein exakteres Schema zur Software befindet sich in Anhang A.5.

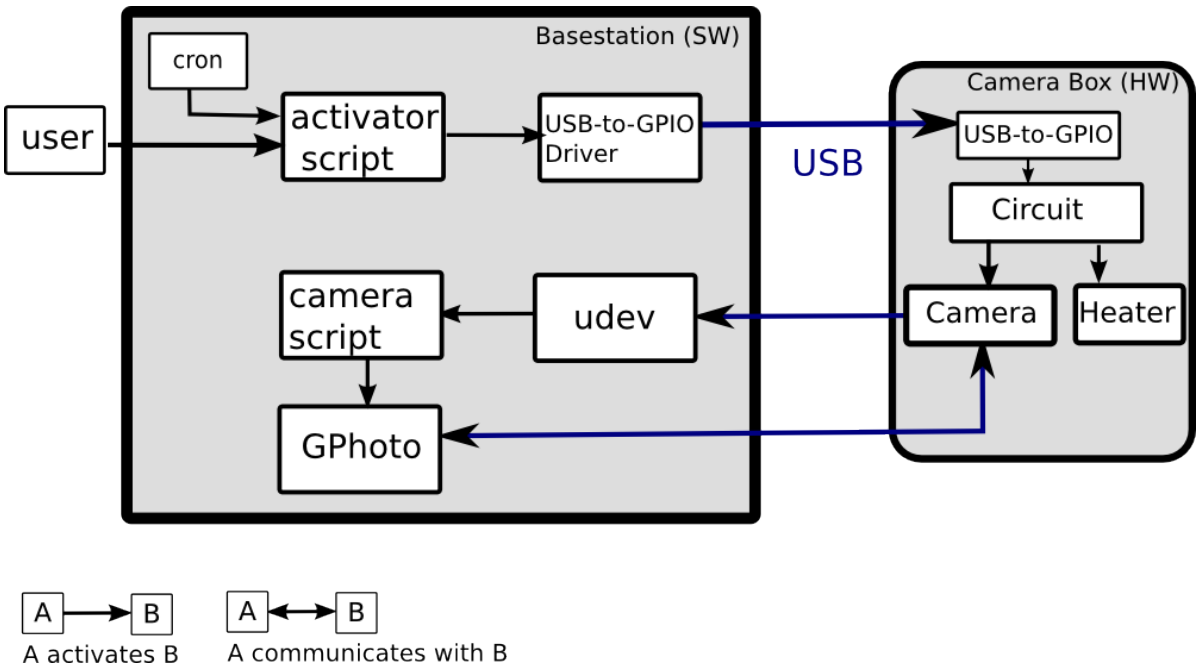


Abbildung 4.1: Diagramm der grundlegenden Softwarekomponenten

USB-to-GPIO-Treiber Dieser Treiber bietet ein Interface zur Steuerung der GPIO-Pins am CP2103- Modul (siehe Kapitel 2.4). Es handelt sich um eine von Mustafa Yuecel (Mitarbeiter des Instituts für Technische Informatik und Kommunikationsnetze am Departement ITET) erweiterte Version des CP210x-Herstellertreibers. Als Interface dient eine Datei, die bei Anschliessen des Geräts an die USB-Schnittstelle automatisch unter `\proc\cp2103` eingelinkt wird. Zur Benutzung dieses Interfaces seien folgende Beispiele angeführt:

- `cat /proc/cp2103` gibt den aktuellen Status der GPIOs aus.
- `echo "GPIO0 clear" > /proc/cp2103` setzt GPIO0 auf "low". Dies würde in der besprochenen Konfiguration (siehe Kapitel 3.4) also die Kamera einschalten.
- `echo "GPIO0 set" > /proc/cp2103` setzt GPIO0 auf "high", würde hier also die Kamera wieder ausschalten.
- `echo "GPIO0 set GPIO1 set GPIO2 set GPIO3 set" > /proc/cp2103` setzt alle GPIOs auf "high" (GPIO2 und GPIO3 werden in der besprochenen Konfiguration nicht benutzt).

Aktivierungsskript (/usr/bin/mview.sh) Dieses Skript bietet dem Benutzer ein einheitliches Interface zur Steuerung der Kamera und der Heizung. Zur Erläuterung der Bedienung sei auf das Kapitel 5.2.4 verwiesen.

Konfigurationsdatei (/etc/mview/config.sh) Eine von beiden Skripten (mview.sh, camera.sh) aufgerufene Datei zur Konfiguration des automatischen Modus (z.B.

zum Festlegen des Speicherortes der Bilder).

Modusdatei (MODEFILE) Wird vom Aktivierungsskript bei manuellem Starten der Kamera geschrieben, um dem Kameraskript mitzuteilen, dass es seine Aktionen schon zu Beginn abbrechen und die Kamera für Benutzeraktionen freigeben soll.

Mutexdatei (MUTEXFILE) Wird vom Aktivierungsskript geschrieben, wenn auf die Kamera zugegriffen wird. Im automatischen Modus löscht das Kameraskript nach Beendung seiner Aktionen die Mutexdatei. Im manuellen Modus wird sie gelöscht, wenn der Benutzer die Kamera durch einen Ausschalt- Befehl an das Aktivierungsskript wieder freigibt. Falls noch ein anderer Aktivierungsprozess auf die Kamera zugreifen will, muss dieser warten, bis die Mutexdatei wieder gelöscht wird.

Dieses Verfahren sollte Ressourcenkonflikte weitestgehend verhindern. Die Befehle zum Erzeugen und Löschen der Datei sind allerdings nicht atomar, insofern können Verklemmungen (“Deadlocks”) nicht ausgeschlossen werden. Bei Tests mit bis zu 10 konkurrierenden Prozessen sind allerdings keine solchen Probleme festgestellt worden. Eine Lösung zur Erzeugung von atomaren Mutex mit der “Bourne-Shell” ist bis anhin nicht gefunden worden.

udev Der Linux-Gerätemanager. Bei Erkennung der Kamera als USB-Gerät wird das Kameraskript ausgelöst. Zur Festlegung der udev-Regeln siehe auch Kapitel 5.2.1.

Kameraskript (/etc/udev/scripts/camera.sh) Implementiert den automatischen Ablauf der Kamerafunktionen. Besteht im Wesentlichen aus folgenden Operationen:

```
01 lade "config.sh"; -> LOKALER_PFAD, MODUSDATEINAME, MINIMALER_SPEICHERPLATZ,
       MINIMALER_SPEICHERPLATZ_KAMERA
02 lösche alle Umgebungsvariablen;
03 setze PATH- und HOME-Umgebungsvariablen neu;
04 wenn ([MODUSDATEINAME] gefunden und Inhalt von [MODUSDATEINAME] ist "MODE=FREE")
05  lösche [MODUSDATEINAME];
06  breche ab;
07 speichere aktuelle Zeit -> TIMESTAMP;
08 frage freien Speicherplatz im lokalen Dateisystem ab -> FREIER_SPEICHERPLATZ;
09 wenn ([FREIER_SPEICHERPLATZ] tiefer als [MINIMALER_SPEICHERPLATZ])
10  lösche das älteste Bild vom lokalen Ordner;
11 detektiere Kamera;
12 frage freien Speicherplatz auf Kamera ab -> FREIER_SPEICHERPLATZ_KAMERA;
13 wenn ([FREIER_SPEICHERPLATZ_KAMERA] tiefer als [MINIMALER_SPEICHERPLATZ_KAMERA])
14  lösche die 10 ältesten Bilder auf der Kamera;
15 stelle Kamera-Speicherort auf "SD-Karte";
16 stelle Kamera-Bildqualität auf "raw";
17 erzeuge ein Bild -> KAMERADATEINAME;
18 lade das neueste Bild herunter, speichere in [LOKALER_PFAD]. Dateiname: [TIMESTAMP]_
       [KAMERADATEINAME].NEF;
19 lade das Thumbnail des neuesten Bildes herunter, speichere in [LOKALER_PFAD].
       Dateiname: thumb_[TIMESTAMP]_[KAMERADATEINAME].NEF;
20 lösche [MUTEXFILE];
```

Anmerkungen:

1. Zur Begründung der Manipulation der Umgebungsvariablen siehe auch Punkt 3 im Kapitel 4.4.
2. Die hier benutzten GPhoto-Operationen werden in Kapitel 2.1.3 aufgelistet.
3. Dieses Skript unterstützt zumindest die beiden Kameramodelle D70s und D200 von Nikon. Andere Digitalkameras müssen zuerst auf die Kompatibilität mit den verwendeten GPhoto2-Befehlen hin getestet werden.

CRON Die zeitbasierte Jobsteuerung von UNIX-artigen Betriebssystemen. Wird für die Automatisierung der Bilderfassung verwendet (siehe Kapitel 5.2.4).

Logdaten Die Abläufe werden bei jedem Durchgang in eine separate Datei gespeichert. Die Identifizierung der Logdatei wird mittels eines Zeitstempels im Dateinamen gewährleistet. Die Logdaten befinden sich in einem Unterordner des lokalen Pfades (kann in der Konfigurationsdatei eingestellt werden). Ein Beispiel einer Logdatei ist in Anhang A.3 zu finden.

Liste aller Bilder auf der Kamera Zusätzlich zu den Logs wird durch `camera.sh` auch eine Liste der Bilder, die sich auf der Speicherkarte der Kamera befinden, geführt. Sie befindet sich im gleichen Ordner wie die Logdateien.

4.4 Bekannte Probleme und mögliche Verbesserungen

1. Die aktuelle Software synchronisiert die Zeit auf der Kamera nicht mit der Zeit auf der Basisstation. Der Befehl `gphoto2 --set-config time="ZEITSTRING"` hat keinen Effekt, womöglich verlangt dieser Befehl aber ein spezielles Format für den Eingabestring, welches nicht den üblichen UNIX- Konventionen entspricht (siehe `man date`). Hier wäre zu prüfen, ob trotzdem eine funktionierende Möglichkeit existiert. Die GPhoto-Dokumentation [10] liefert hierzu keine Informationen, womöglich ist dies auch abhängig vom Gerät.

   Während unseren Testläufen hatte die fehlende Zeitsynchronisierung jedoch keine Auswirkung, da die Kameras auch ohne Stromversorgung ihre Zeiteinstellung behalten. Ob dies jedoch auch über mehrere Wochen und unter tiefen Temperaturen der Fall ist, wird zu prüfen sein. Als “Workaround”, falls keine andere Lösung gefunden wird, könnten auch die Zeiteinträge in den Metadaten nach dem Speichern auf die Basisstation angepasst werden.

2. Die jetzt implementierte Skriptsteuerung arbeitet – zumindest auf dem von uns getesteten Gumstix – langsam. Ein Durchlauf dauert, je nach Vorbedingungen (müssen auf der Kamera noch Bilder gelöscht werden?), 3-4 Minuten. Zur Verbesserung seien hier folgende Vorschläge angefügt:

- Die bereits implementierte, mitgeführte Liste der Bilder auf der Kamera könnte zur Beschleunigung beitragen, indem einerseits auf eine Übertragung der Liste im Falle des Löschens von Bildern verzichtet, andererseits der freie Speicherplatz direkt aus dieser Liste errechnet werden könnte. Dieses Verhalten wäre aber fehleranfälliger, da die Daten nicht selbstkorrigierend wären.
- Anstatt die Standardbefehle von GPhoto2 zu benützen, könnte auf Basis der libgphoto-API oder auf Basis von PTP-Befehlen ein gerätespezifisches Softwareinterface programmiert werden.
- Bei der Wahl der Plattform sollte auf die Unterstützung von USB 2.0 geachtet werden. Die Geschwindigkeit der Skripte konnte im Rahmen dieses Projektes nicht auf einem Embedded System mit USB 2.0 getestet werden.

3. Der Speicherort der Bilder und Logs für die automatische Erfassung (in der Konfigurationsdatei mittels `LOCALPATH` angegeben) darf sich aus unbekannten Gründen nicht im `home`-Verzeichnis befinden, sonst können die Bilder nicht richtig gespeichert werden. Bei manuellem Benutzen der GPhoto-Funktionen in der Shell tritt dieses Problem nicht auf. Wir vermuten einen Zusammenhang mit den Umgebungsvariablen, die bei `udev`-aktivierten Skripten nicht identisch sind mit denjenigen, die ein Shellbenutzer sieht. Deshalb ist es notwendig, zu Beginn des Skriptes einige Umgebungsvariablen neu zu setzen, was das Funktionieren der GPhoto-Operationen überhaupt erst ermöglicht. Eine Lösung, um das `home`-Verzeichnis benutzen zu können, konnte hingegen bis anhin noch nicht gefunden werden.

4. Zur Verhinderung von dauerhaft hohem Stromverbrauch bei Fehlfunktionen sollte ein "Wächter" implementiert werden, der im automatischen Modus die Kamera nach einer gewissen Zeitdauer automatisch abschaltet, falls sie bis dann noch nicht freigegeben wurde. In der aktuellen Implementation müsste im Falle einer Verklemmung (z. B. bei Absturz des Kameraskripts im automatischen Modus) die Mutex manuell mit dem Befehl `mview.sh FORCE_UNLOCK` gelöst werden (siehe auch Kapitel 5.2.4).

5. Die vorhandene Mutex wäre zur garantierten Verhinderung von Verklemmungen durch eine atomare zu ersetzen. (siehe Kapitel 4.3).

6. Die Liste der Bilder auf der Kamera ist nicht selbstkorrigierend. Bei entdeckten Unstimmigkeiten wird lediglich eine Warnung geloggt. Dies könnte durch eine erneute Generierung der Liste mittels `gphoto2 --list-files` gelöst werden.

7. Eine reduzierte Variante der Logs von `camera.sh` könnte auch in die UNIX-Systemlogs integriert werden.

5 Anleitungen und Tests

5.1 Hardware: Überblick und Installation

Um das System in Betrieb zu nehmen, schliesse man die Komponenten so an, wie es auf den Abbildungen 5.1 und 5.2 gezeigt wird. Die meisten Anschlüsse sind selbsterklärend, bei Unklarheiten bezüglich der Pinbelegung sei auf Abbildung 3.3 verwiesen. Folgende Punkte sind beim Zusammenbau zu beachten:

- Die erläuterte Schaltung ist empfindlich auf elektrostatische Entladungen (ESD) und Potentialunterschiede, dies betrifft insbesondere das Anschliessen der Schaltung an den CP2103. Um dem vorzubeugen, sollten in jedem Fall zuerst die Erdungen (GND) der beiden Geräte verbunden werden, entweder indem sie an den selben USB-Hub angeschlossen, oder (mit einem zusätzlichen Kabel) die GND-Pins miteinander verbunden werden. Erst danach dürfen die GPIO-Verbindungen (blaue Litze auf dem Foto) hergestellt werden. Beim Abbau ist in umgekehrter Reihenfolge vorzugehen.
- Um die Schaltung, wenn sie nicht benutzt wird, zusätzlich vor ESD zu schützen, sollten die Kabel gemäss Abbildung 3.5 an GND angeschlossen werden.
- Die GPIO Verbindungen werden gemäss Abbildung 5.3 hergestellt. Die Anschlüsse auf dem CP2103 sind mit “J5, GPIO_1 sowie GPIO_0” beschriftet. Bei Benutzung des beigelegten Kabels beachte man die Pfeile auf dem Kabel und auf dem Prototyp.
- Die Polarität des (durchtrennten) Nikon-Netzteilkabels ist: rot +, schwarz GND.
- Die Polarität der Heizung muss nicht beachtet werden.
- Die USB-Schnittstelle der verwendeten Gumstix-Breakout-Erweiterungskarte ist im mini-B-Format, was üblicherweise auf Seite der Peripherie – nicht des PCs – verwendet wird. Deshalb müssen, um eine USB-Verbindung mit einem Peripherie-Gerät zu realisieren, zwei USB-Kabel mit dazwischengeschaltetem “gender switch Adapter” verwendet werden (siehe Abb. 5.4).
- Vor der definitiven Montage ist zu beachten, dass einige Einstellungen (z. B. Zoom und Fokus) an der Kamera nur mannuell vorgenommen werden können. Ausserdem muss die Speicherkarte formatiert und von der Kamera erkannt werden, der Akku darf nicht eingelegt sein und der Hauptschalter muss auf “ON” stehen.

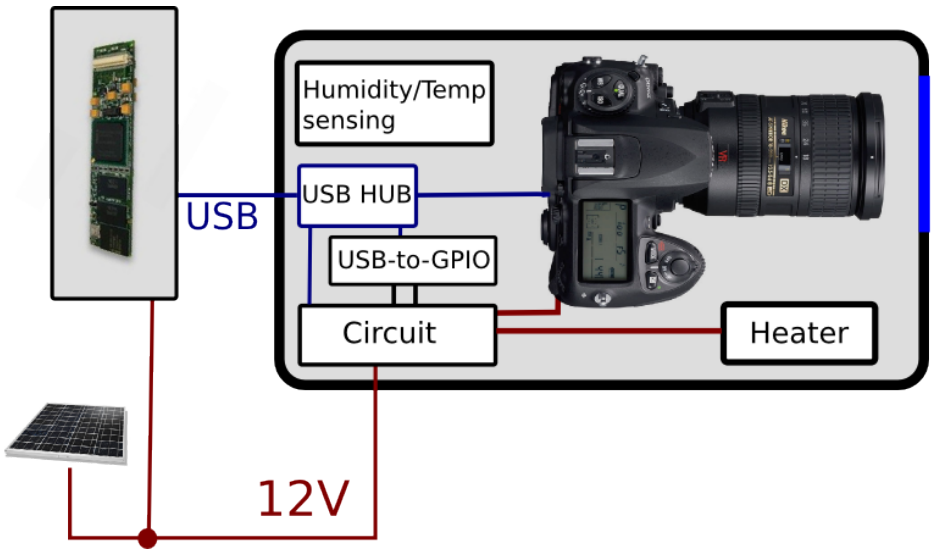


Abbildung 5.1: Schematischer Überblick der Hardware-Komponenten

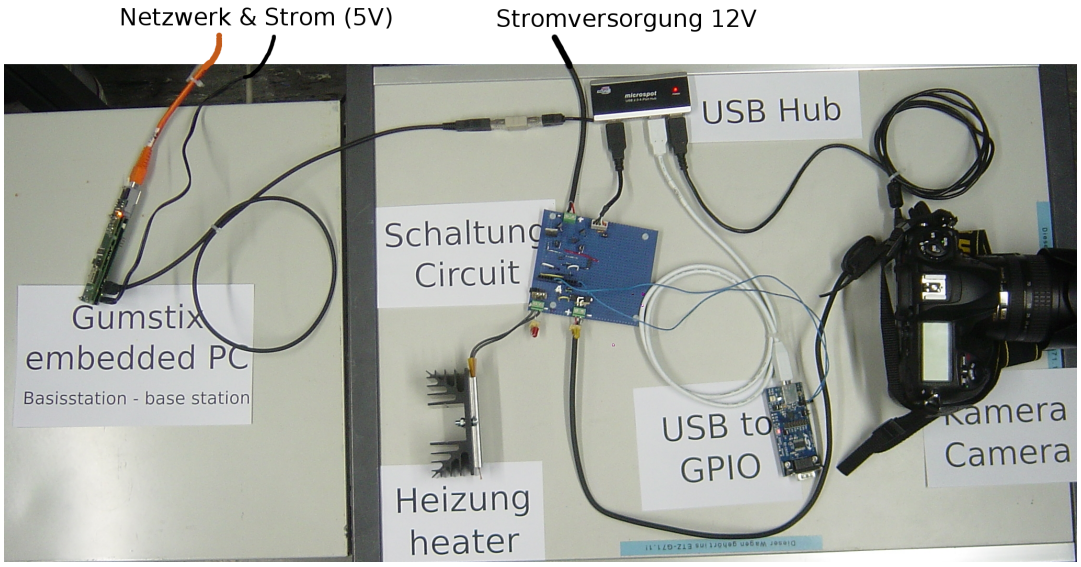


Abbildung 5.2: Foto der Hardware-Komponenten, fertig zusammengebaut

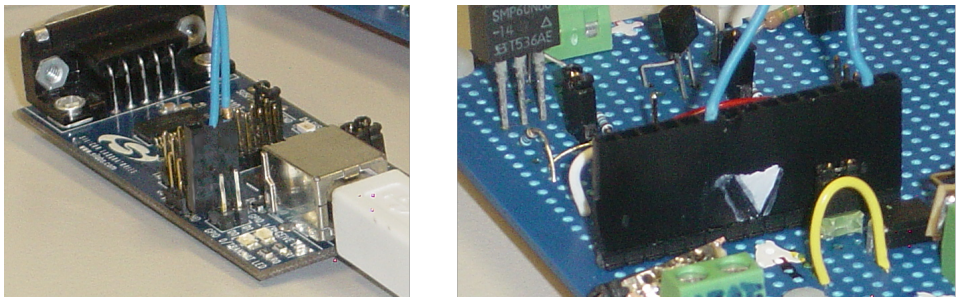


Abbildung 5.3: Anschliessen des Prototyps an den CP2103 mit dem beigelegten Kabel

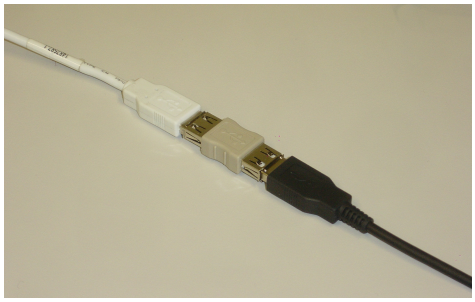



Abbildung 5.4: Funktionsweise des USB-“gender switch Adapters”

5.2 Software: Installation und Bedienung

5.2.1 Installation

In diesem Kapitel werden alle nötigen Schritte aufgelistet, um einen Embedded-PC mit installiertem Linux, Kernel Version 2.6.21, und Paketmanager “ipkg” bereit zu machen für die Steuerung des “Mountainview”-Systems.

CP2103 Der “Treiber” für die GPIOs des CP2103 ist ein Patch für den Linux Kernel, er ist auf der CD-ROM im Verzeichnis “software” enthalten. Zum Installieren gehe man wie folgt vor:

- den Patch auf den Linux Kernel 2.6.21 anwenden
- `CONFIG_USB_SERIAL_CP2101` muss im `defconfig` gesetzt sein (ob static oder module ist nicht wichtig)
- den Kernel kompilieren
- den Kernel auf die Zielplattform installieren
- das Kernelmodul `cp2101` neu laden, falls es als Modul kompiliert wurde

Nach erfolgreicher Installation wird beim Anschliessen eines CP2103 automatisch die Datei `/proc/cp2103` erstellt. Diese Datei wird als Interface verwendet, um die GPIO des CP2103 zu steuern (siehe auch Kapitel 4.3).

GPhoto Um GPhoto zu installieren genügt bei funktionierendem ipkg-Manager die Eingabe von `ipkg install gphoto2`.

Skripte Für die Installation der Software-Skripte (sie befinden sich auf der CD unter `software/scripts.tar.gz`) gehe man wie folgt vor:

1. `config.sh`, sowie `helptext` in den Ordner `/etc/mview/` kopieren
2. `mview.sh` in den Ordner `/usr/bin/` kopieren
3. `camera.sh`, sowie `camera_frame.sh` in den Ordner `/etc/udev/scripts` kopieren

udev

Bei Verwendung einer Nikon D200 kann die im Anhang unter `software/scripts.tar.gz` enthaltene Datei `10-mountainview.rules` in den Ordner `/etc/udev/rules.d/` kopiert werden. Mit dieser Regel lässt `udev` bem Anschliessen einer Nikon D200 das Skript `\etc\udev\camera_frame.sh` ausführen, welches wiederum `\etc\udev\camera.sh` im Hintergrund ausführt. Dieser Umweg ist notwendig, da sonst `udev` mit der Einlinkung der Kamera warten würde, bis das Skript beendet ist.

Wenn eine andere Kamera verwendet wird, sind folgende Schritte[1] notwendig (diese können auf einem beliebigem PC mit aktuellem Linux-Betriebsystem ausgeführt werden):

1. `ls /sys/bus/usb/devices > temp.txt` ausführen
2. Kamera über USB mit dem PC verbinden und einschalten
3. `ls /sys/bus/usb/devices` erneut ausführen und vergleichen mit dem Inhalt von `temp.txt`; Der Pfad des neu hinzugekommenen Gerätes soll vorgemerkt werden (Beispiel: `/sys/bus/usb/devices/4-4`).
4. `udevinfo -a -p [Systempfad des Geräts]` ausführen (Beispiel: `udevinfo -a -p /sys/bus/usb/devices/4-4`); Aus den ausgegebenen Informationen lassen sich `ATTR{idVendor}` und `ATTR{idProduct}` ablesen.
5. `ATTR{idVendor}` und `ATTR{idProduct}` in `10-mountainview.rules` durch die soeben gesammelten Informationen ersetzen.
6. Die (abgeänderte) Datei `10-mountainview.rules` in den Ordner `/etc/udev/rules.d/` kopieren.

5.2.2 Test der Installation

Sofern alle in den Kapiteln 5.1 und 5.2.1 beschriebenen Schritte erfolgreich ausgeführt wurden, sollte ein erster Aufruf von `mview.sh STATUS` jetzt den Status der Kamera und der Heizung anzeigen: Beide "Powerlines" sollten im Moment im "off"-Zustand sein. `mview.sh HELP` sollte einen Hilfetext zur Bedienung ausgeben. Falls diese beiden Befehle ohne Fehlermeldungen funktionieren, heisst dies, dass zumindest der CP2103 und die Skripte korrekt installiert wurden.

Als nächstes kann mittels `mview.sh CAMERA_SCRIPTED` (oder einfach `mview.sh`) getestet werden, ob die `udev`-Regel funktioniert. Der Zustand der Systems kann mittels `watch mview.sh STATUS` überwacht werden. Mit der Kommandozeile `mview.sh CAMERA_ON` kann getestet werden, ob die Übermittlung des Programmmodus an das Kameraskript funktioniert (das Aktivierungsskript muss dann über die Modusdatei dem Kameraskript mitteilen, dass es schon zu Beginn abbrechen soll). Falls zu diesem Zeitpunkt das automatische Skript noch arbeitet, muss der neue Prozess warten.

[1] in [11] und [12] können allgemeinere Anleitungen zur Bedienung von udev gefunden werden.

5.2.3 Automation

Um `mview.sh` automatisch zu festgelegten Zeitpunkten zu aktivieren, kann der Linux-Jobdaemon `cron` benutzt werden. Dazu wird mittels `crontab -e` die Regeldatei des Benutzers editiert. Die Einträge müssen dazu wie folgt aussehen:

```
[Minute] [Stunde] [Tag] [Monat] [Wochentag] [Kommando]
```

Asteriske (*) werden eingesetzt, um jede Instanz einer Zeitperiode zu benutzen (Beispiel: Ausführung zu jeder Stunde). Um mehrere Instanzen einer Zeitperiode anzugeben, können diese mit Kommas getrennt eingefügt werden. Kommandos können beliebige Operationen der Kommandozeile sein, also z. B. auch das Aufrufen von Skripten. [13]
Beispiel:

```
0,30 * * * * /home/user/skript.sh
```

Führt jeweils zur vollen und halben Stunde `/home/user/skript.sh` aus.

5.2.4 Übersicht über die Befehle

CAMERA_SCRIPTED (default) schaltet die Kamera an und aktiviert das automatische Kameraskript. Nach Beendung wird die Kamera wieder abgeschaltet.

CAMERA_ON schaltet nur die Kamera an. Danach ist sie frei verfügbar, um z.B. gphoto2-Operationen manuell ausführen zu können. Am Ende muss unbedingt `mview.sh CAMERA_OFF` ausgeführt werden, um die Kamera wieder für andere Prozesse freizugeben.

CAMERA_OFF schaltet die Kamera ab und gibt sie frei (wird nur ausgeführt, wenn sich die Kamera im manuellen Modus befindet, also durch `mview.sh CAMERA_ON` aktiviert wurde).

HEATER_ON schaltet die Heizung ein.

HEATER_OFF schaltet die Heizung aus.

FORCE_UNLOCK erzwungenes Abschalten der Kamera und Löschen von MUTEX- und Modusdatei. Dieser Befehl ist nur für den Fall gedacht, dass `camera.sh` abstürzt und danach der Zugriff auf die Kamera blockiert ist (MUTEX wird nicht freigegeben).

STATUS zeigt den aktuellen Status des Systems an, inklusive der Logdatei, falls zu diesem Zeitpunkt das Kameraskript aktiv ist.

HELP zeigt einen Hilfetext an.

5.3 Tests zum Gesamtsystem

Es wurde eine Reihe von Funktionstests des Gesamtsystems durchgeführt. Eine Übersicht der Tests und Resultate ist in Tabelle 5.1 zu sehen.

Tabelle 5.1: Tests am Gesamtsystem

Test Nr.	Beschreibung	Start	Ende	Ergebnis
1	Test der Heizungssteuerung im angeschalteten Zustand	8.12.2008 20:30	8.12.2008 21:00	Heizungs-LED eingeschaltet, Heizung ist aktiv, keine Probleme beim Ausschalten
2	Langzeittest der Heizungssteuerung im angeschalteten Zustand	8.12.2008 22:30	9.12.2008 11:00	Heizung aktiv bis zum Ende, keine Probleme beim Ausschalten
3	Test der Kamerasteuerung mittels cronjob: 1 Durchlauf des Scriptes pro 10min	9.12.08 23:00	10.12.08 11:00	keine Fehlfunktion von HW, Bug beim Handling der Kameraspeicherliste gefunden und korrigiert
4	Belastungstest der Schaltung: Heizung an/aus im 4s-Takt plus ein Bild pro 10min mittels cronjob	14.12.08 20:00	15.12.08 10:00	keine Fehlfunktion von HW & SW, R_{GS} aller MOSFETS unmessbar hoch
5	Test der SW auf Deadlocks: Starten von 10 konkurrierenden Kamera-Prozessen mittels `mview.sh`	15.12.08 11:00	15.12.08 12:00	10 Bilder erfolgreich gespeichert

Der Stromverbrauch der Schaltung im "Standby"[2], ohne USB-Komponenten, ist $\ll$ 1mA. Auf Messungen des Stromverbrauchs am Gesamtsystem wurde verzichtet, da auf den Platinen des USB-Hubs und des CP2103 noch unbenutzte Peripherie (und Anzeige-LEDs) vorhanden ist, die die Aussage der Messung zumindest im "Standby" wesentlich verfälschen würde.

Aus Zeitmangel wurden auch keine Tests bei extrem tiefen Temperaturen durchgeführt.

[2]sprich: angeschlossen und betriebsbereit, aber weder Kamera noch Heizung eingeschaltet

6 Fazit und Ausblick

6.1 Fazit

Zusammenfassend betrachtet konnte im Rahmen dieser Gruppenarbeit gezeigt werden, dass eine hochauflösende, ferngesteuerte, Energie sparende und flexibel zu bedienende Bilderfassung technisch machbar ist. Konkret kann dies anhand des erstellten Systems gezeigt werden. Momentan nicht erfüllt bleibt die Wetterfestigkeit des Systems. Ein entsprechendes Gehäuse befindet sich zum Zeitpunkt der Abgabe dieser Arbeit jedoch bereits in der Fertigung. Es wird anschliessend anhand von Tests in einem Kältelabor, sowie Praxistests, zu zeigen sein, wie gut das System die Anforderungen aus Kapitel 1 tatsächlich erfüllt, insbesondere im Umgang mit sehr tiefen Temperaturen[1]. Beim Auftreten von Problemen aufgrund von Feuchtigkeit oder Kälte kann bereits jetzt, dank der vorhandenen Überwachungsmöglichkeiten (siehe Kapitel 2.3), die Ursache festgestellt und bestenfalls, mittels der eingebauten Heizung, sofort behoben werden.

Die Vorgaben in Bezug auf die Bildqualität und die Bedienung von Hard- und Software konnten bereits im vorliegenden Systems erfüllt werden (einfache Montage, ferngesteuerter "Reset" der Kamera möglich, Implementierung der automatischen Bilderfassung und Steuermöglichkeiten aus der Ferne). Raum für Verbesserungen eröffnet sich in folgenden Punkten:

- schnellere und sicherere Steuerungssoftware (siehe Kapitel 4.4)
- Zusammenfassung der Hardwarekomponenten auf einer einheitlichen Platine

6.2 Ausblick

Die nächsten Schritte zur Verbesserung des Systems werden sein:

- Anbindung über WLAN, was die mögliche Distanz zur Basisstation erheblich vergrössert.
- Einbau der Solarstromverorgung in das Gehäuse, um eine noch einfachere Montage gewährleisten zu können.

[1] Zumindest die für die Schaltung verwendeten Komponenten sind gemäss Datenblatt für die vorhandenen Temperaturbedingungen geeignet.

- Ausbau der Kamera zu einem eigenständigen System, das ohne externe Basisstation arbeitet.

Das Ziel ist also ein Bilderfassungssystem, das unabhängig von den gegenwärtigen Installationen eingesetzt werden kann.

A Anhang

A.1 Messprotokoll

Projekt „Mountainview“

Messprotokoll

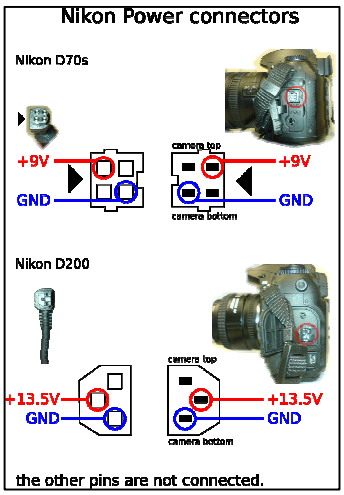
Stefan Kronig & Michel Müller

Einleitung

Bei unseren Messungen geht es um vier entscheidende Punkte, die für den weiteren Verlauf des Designprozesses ausschlaggebend sind:

1. Verhalten der Kameras im Eingeschalteten Zustand ohne Aktivität.
2. Ermittlung der maximalen Ströme, welche z. B. ein Schalter für die Kamera verkraften muss.
3. Langzeitverhalten der Kameras, wenn sie periodisch ein- und ausgeschaltet werden.
4. Langzeitverhalten in Bezug auf die Steuerung.

Getestet haben wir zwei Kameras der Marke Nikon, eine D70s und eine D200.



Bevor wir überhaupt mit Messen anfangen konnten, gab es einige Probleme zu lösen. Wie genau muss man diese Kameras mit Strom versorgen? Welche Polung haben die Anschlüsse? Als erstes kauften wir also die Netzteile für beide Kameras, damit wir ihre genauen Spannungen und Pinbelegungen messen konnten und falls nötig die Stecker abschneiden und benutzen konnten. Ein übersichtliches Schema mit den Pinbelegungen beider Kameras & Netzteile (siehe links) haben wir ins SVN hochgeladen.

Es stellte sich heraus, dass man die Stecker nicht einzeln kaufen kann. Da wir zum ausprobieren aber nicht beide, über CHF 100.- teuren, Netzteile zerschneiden wollten, suchten wir nach einer Möglichkeit, selbst eine Steckverbindung zu improvisieren. Mit etwas Kabel und Lochrasterplatinen konnten wir für beide Kameras passende Stecker bauen, allerdings sind diese weniger für den definitiven Einsatz als mehr für die provisorische Stromversorgung bei den Messungen geeignet. Im fertigen Einsatz wird man wohl nicht um das Zerschneiden eines Netzteil- Kabels herumkommen.

Um den Bedingungen auf dem Berg am ehesten gerecht zu werden, haben wir die Kameras zum Ausschalten jeweils elektronisch mit Hilfe des Power Analysers vom Netz genommen und den Schalter an der Kamera immer eingeschaltet gelassen. Den Blitz haben wir ausgeschaltet, da dieser für Fotos über grössere Entfernungen zu schwach wäre.

Bei der D70s haben wir mit 9V, 9.4V und 9.7V Versorgungsspannung gemessen. 9.4 ist die Spannung des Original- Netzteils, 9 und 9.7V haben wir eingestellt um zu sehen, ob die Kamera auch mit 5-10% mehr oder weniger Spannung arbeiten kann. Bei der D200 haben wir stets mit 12V gemessen, da wir diese Kamera möglichst direkt an die 12V-Stromversorgung der Basisstation anschliessen möchten.

Messreihe 1

Zielwissen

Mit dieser ersten Messreihe wollten wir das Verhalten der Kameras im eingeschalteten Zustand im Netzbetrieb ohne Aktivität ermitteln, um zum Beispiel allfällige Energiesparmodi sichtbar zu machen. Vom Batteriebetrieb der D70s wissen wir, dass es einen solchen Energiesparmodus gibt, da die Batterie auch nach einer Nacht im ON- Zustand noch zu ca. 75% voll war. Würde die Kamera in diesem Zustand so viel Strom verbrauchen, wie im Netzbetrieb, wäre die Batterie hingegen schon nach 10h leer.

Versuchsaufbau

Hier galt es, den Versuchsaufbau möglichst einfach zu halten. Deshalb schlossen wir die Kamera lediglich an einen Agilent N6705A Power Analyser an. Bei dieser Messung genügte ein Port mit 1.5A Ausgangsstrom.

Für die D70s wurde die Ausgangsspannung konstant auf 9.4V gesetzt, für die D200 auf 12V.

Die Messdauer betrug jeweils 2h.

Resultat

	Min(I)	**Max(I)**	**Min(P)**	**Max(P)**
D70s	0.15A	0.16A	1.82W	1.98W
D200	0.26A	0.31A	3.13W	3.67W

Interpretation

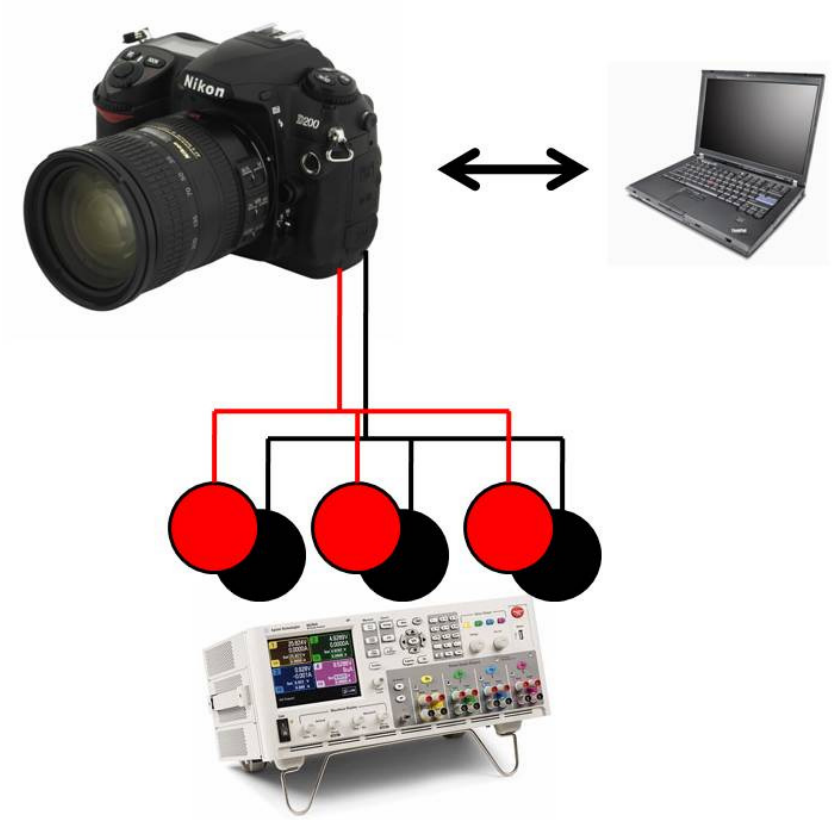
Energiesparmodi waren zumindest in unserem Versuchsaufbau also keine zu ermitteln. Unter den gegebenen Bedingungen ist das elektronische Ausschalten nach jedem Capturing also unumgänglich.

Messreihe 2

Zielwissen

Ziel dieser Messung war erstens, die Maximalströme zu messen, und zweitens, den Energieverbrauch im Betrieb im PTP- Modus festzustellen. Da wir bei der Messung 1 keine plötzlichen Sprünge im Stromverbrauch festgestellt hatten, waren Maxima nur beim Einschalten, Fotografieren und Datentransfer zu erwarten. Ausserdem sollen auch allfällige Rückströme ins Netz gemessen werden.

Versuchsaufbau



Um diesen Versuch durchführen zu können, mussten wir zuerst ein Problem lösen: Der Agilent N6705A Power Analyser kann in der aktuellen Konfiguration höchstens 1.5A Strom pro Port liefern. Die Nikon-Netzteile liefern aber bis zu 4.5A (D70s) resp. 5A (D200). Manuelle Versuche mit der Kamera an nur einem Port angeschlossen ergaben dann auch entsprechende Fehlfunktionen. Durch Ausprobieren fanden wir heraus, dass man mit einer parallel geschalteten Kapazität von mehr als 6.6mF (bei der D70s) die Stromspitze genügend stark ausgleichen kann, um ohne Fehlermeldungen ein Foto schiessen zu können. Allerdings wollten wir mit dieser Messung einen möglichst unverfälschten Stromverlauf analysieren können. Die Lösung war dann die Gruppierung von drei Ausgängen des Agilent N6705A Power Analysers. Diese werden dann intern zu einer virtuellen Schaltung kombiniert, für die Messung werden die Ströme addiert und das Einschalten ist mit nur einem Knopfdruck möglich.

Damit die Messung möglichst dem Betriebsmodus entspricht, den wir in der finalen Komposition verwenden wollen, liessen wir die Kamera im PTP- Modus mittels einem für die Messung geschriebenen Shellscript steuern (auf Laptop des Instituts). Die Funktion dieses Skriptes lässt sich mit einigen Zeilen Pseudocode beschreiben (tatsächlich eingesetzter Code befindet sich im Anhang):

```
Wartezeit_zwischen_Bildern = arg1
While(true)
{
        Überprüfe, ob Kamera angeschlossen.
        Wenn ja:
        {
                Logge die aktuelle Zeit und „versuche Bild zu schiessen“
                Schiesse Bild
                Speichere Bild lokal
                Wenn Speichern erfolgreich:
                        Logge „erfolg, aktuelle Zeit, benötigte Zeit“
                Sonst:
                        Logge „misserfolg, aktuelle Zeit, benötigte Zeit“
                Warte Wartezeit_zwischen_Bildern
        }
        Wenn nein:
                Warte 3 sek.
}
```

Der Agilent N6705A Power Analyser wurde für die Messung mit einer Rechteck- Pulse- Funktion betrieben:

10s: 0V;
30s: 9.7V (D70s) / 12V (D200);
Repeat 3 times;

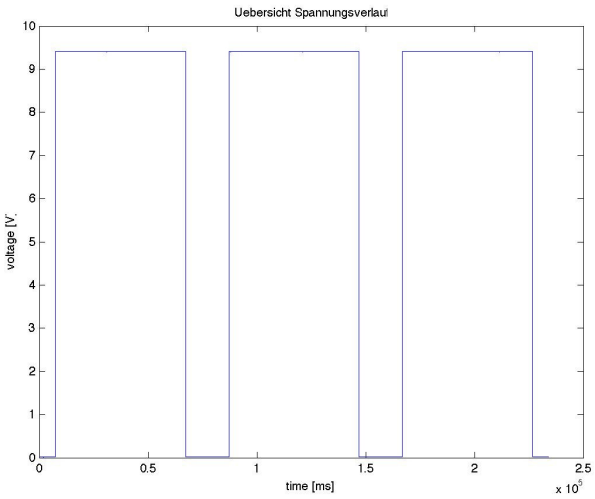
Die Sampling- Rate wurde auf die für den Power Analyser maximale Auflösung eingestellt: 1ms.

Resultate

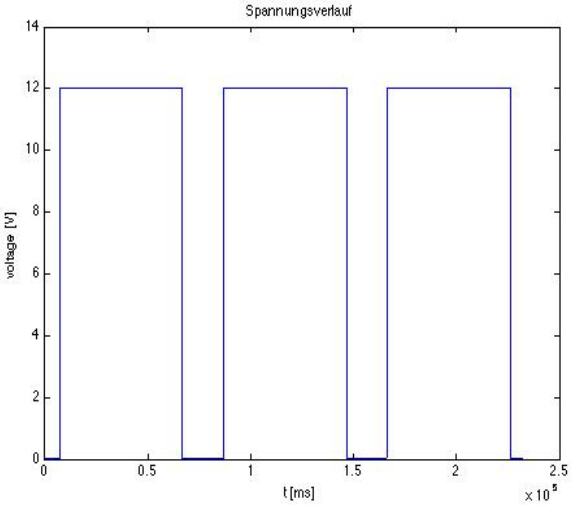
	Einschalt- Spitze (worst case in 3 Versuchen)	**Capturing- Spitze (worst case)**	**Rückfluss-Stromspitze (worst case)**	**Energieverbrauch während 60s**
D70s	0.88 A	3.4914 A	-0.0623 A	88.7516 J
D200	0.9173 A	2.5930 A	-0.4404 A	197.3793 J

Übersicht Gesamtverlauf

D70s Spannungsverlauf

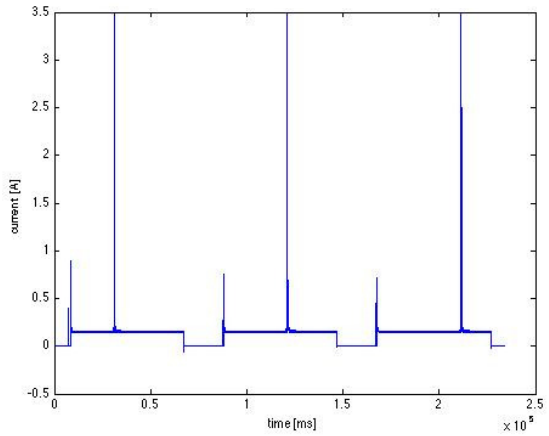


D200 Spannungsverlauf

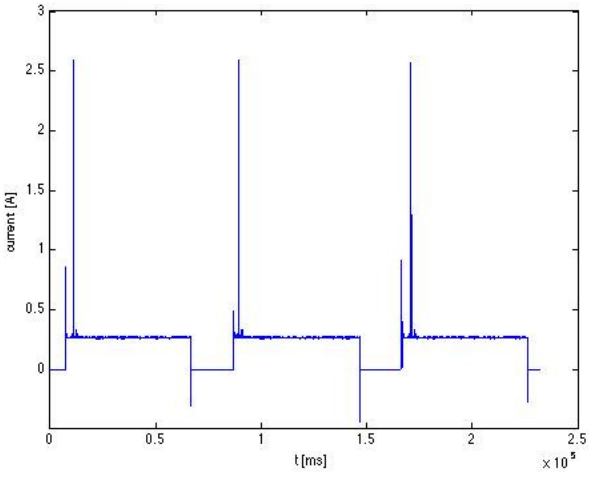


Anmerkung: Offensichtlich sind keine Spannungseinbrüche festzustellen; Die Leistungskurve ist also proportional zum Stromverlauf.

D70s Stromverlauf

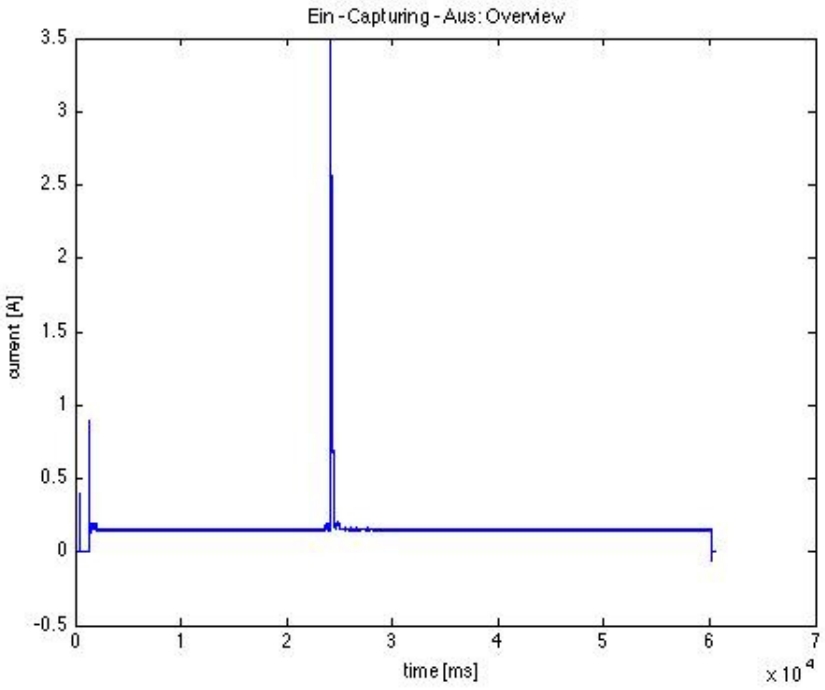


D200 Stromverlauf

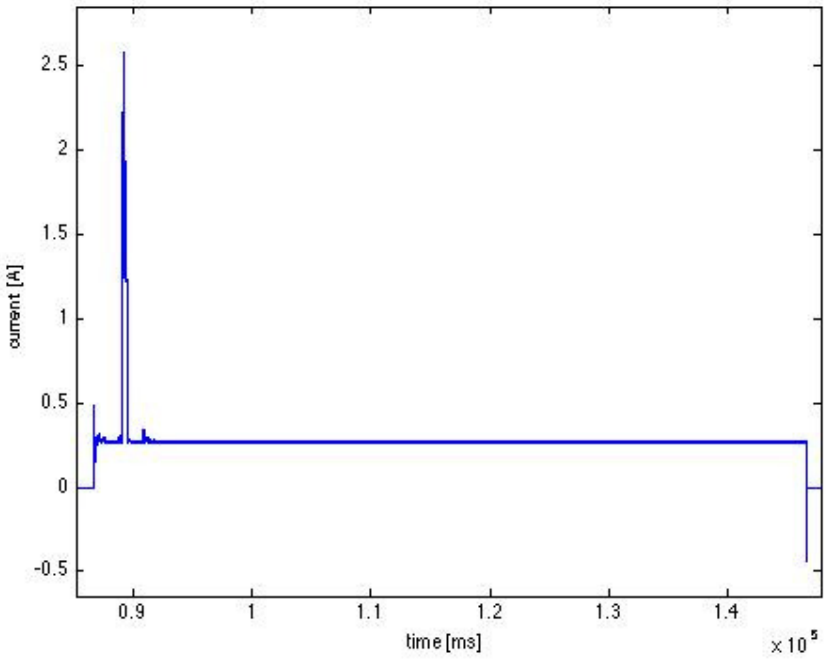


Übersicht über eine Periode

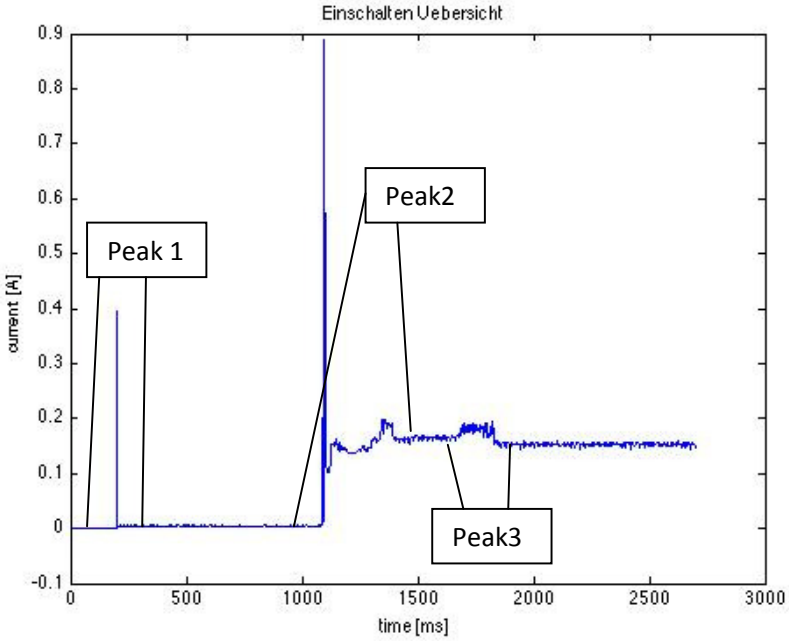
D70s:

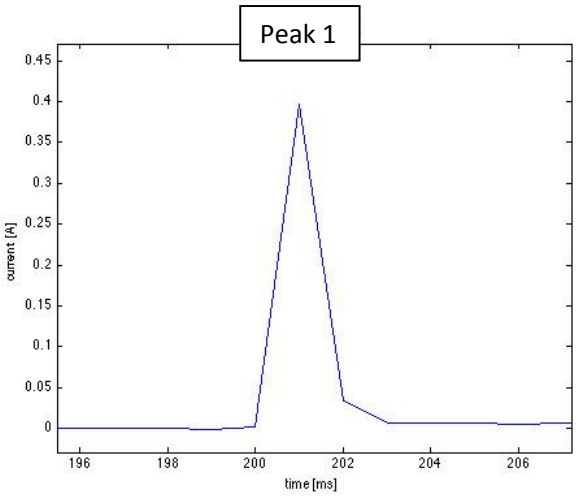


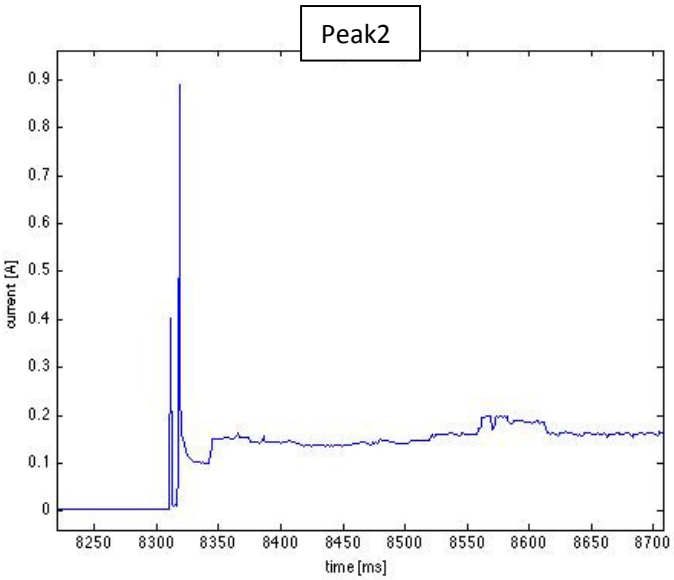
D200:

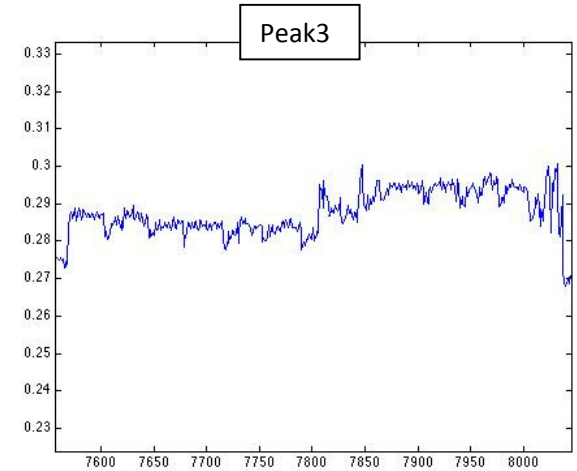


Einschaltprozess D70s

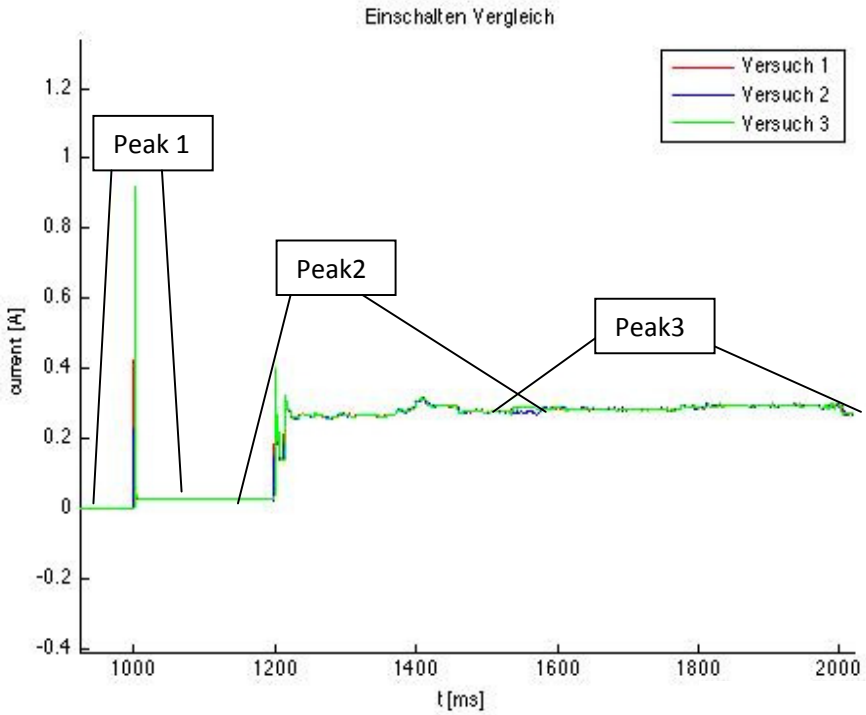


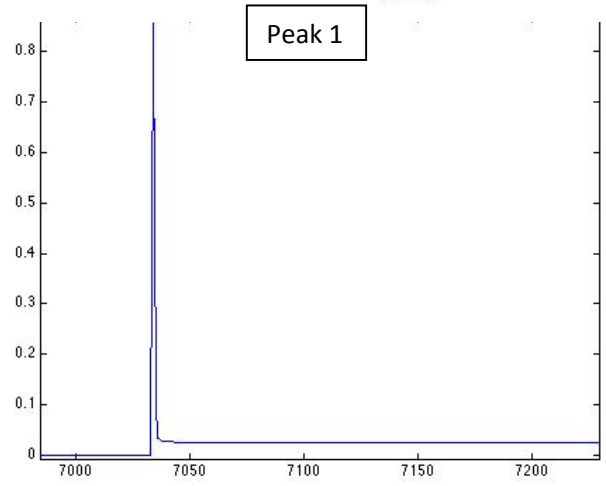


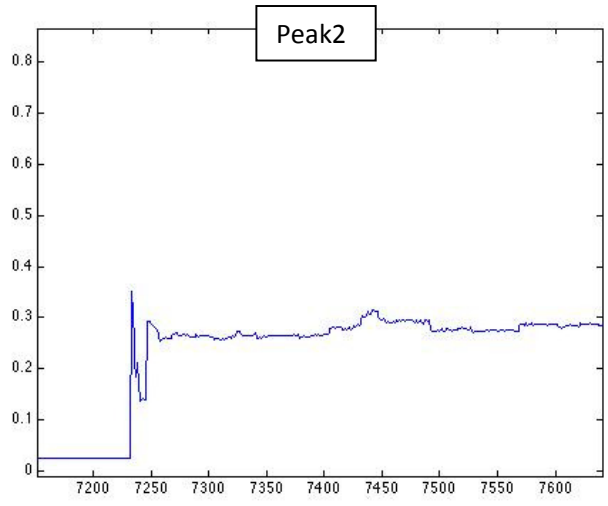


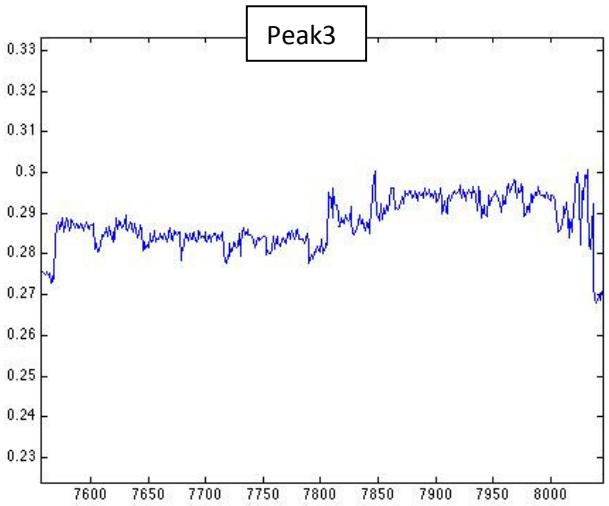


Einschaltprozess D200

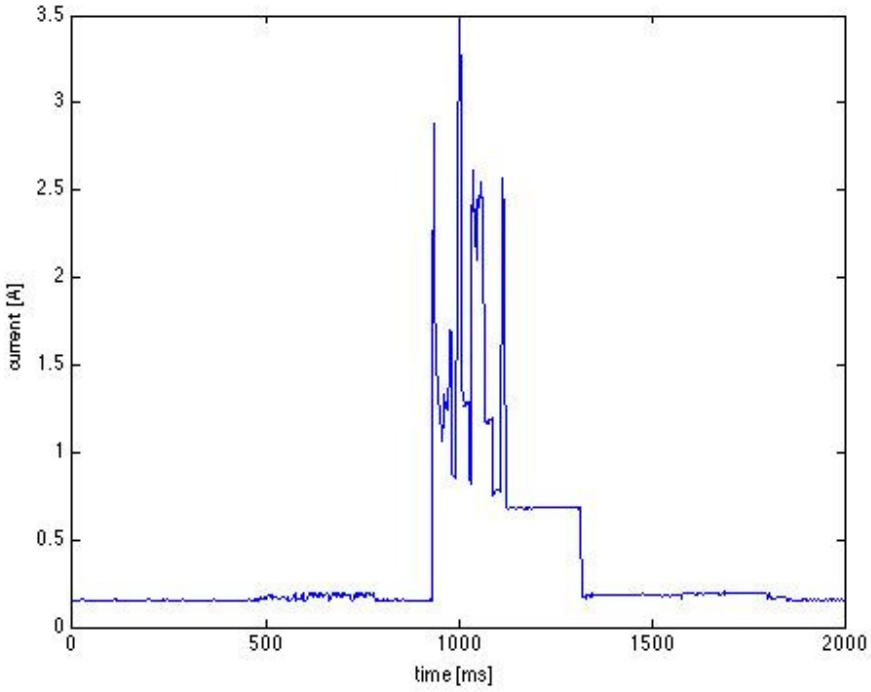


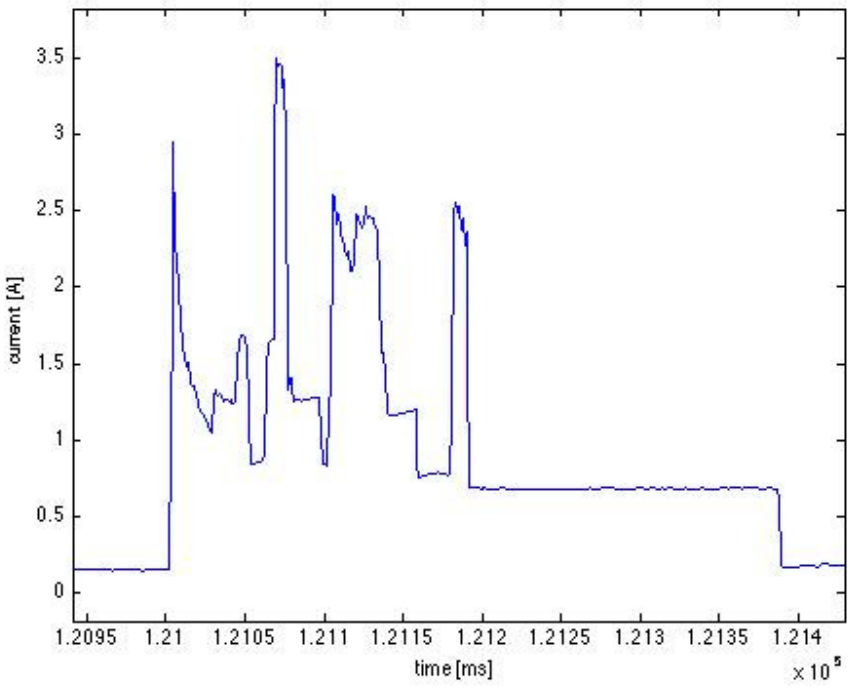




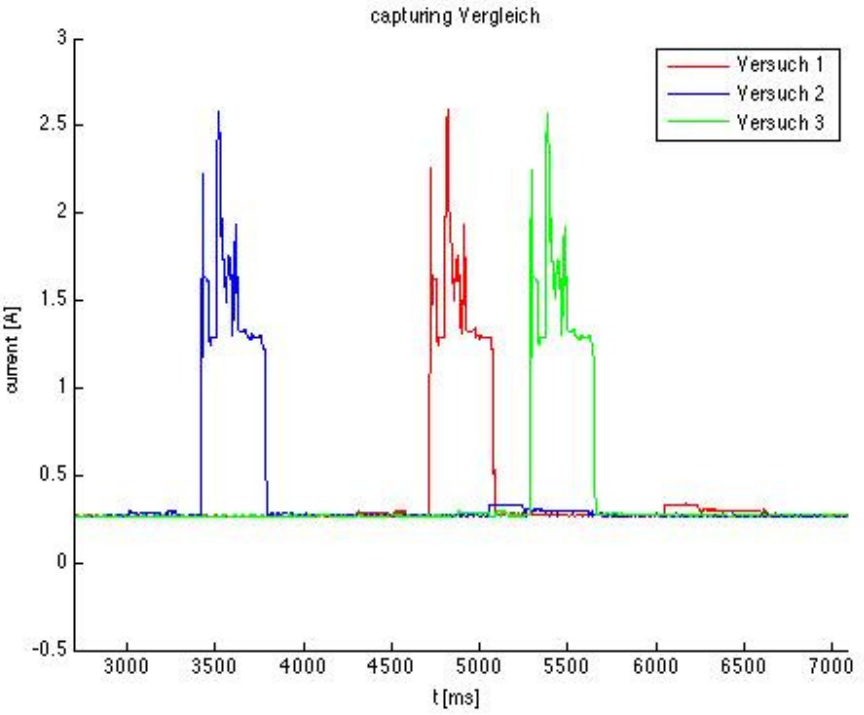


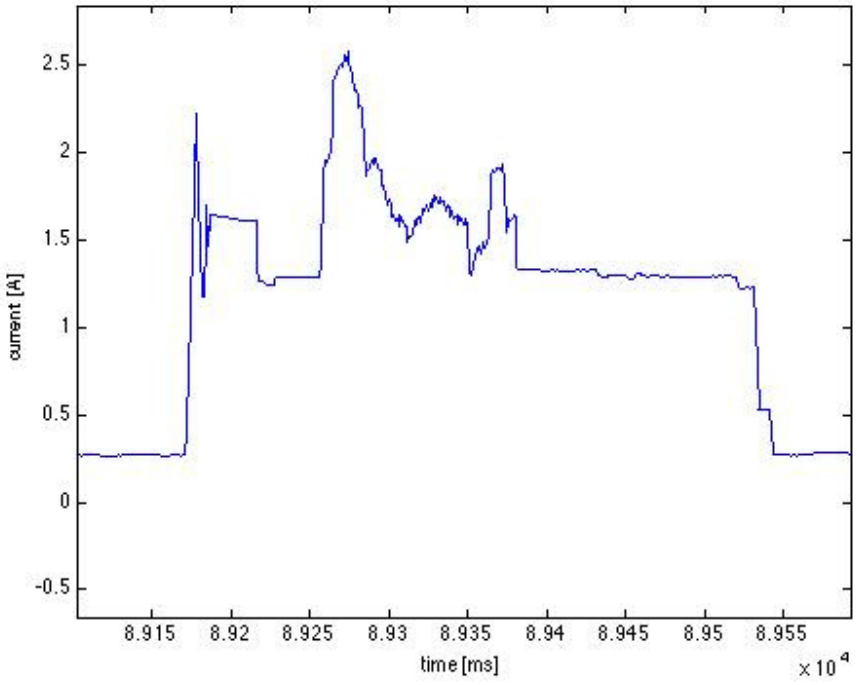
Capturingprozess D70s





Capturingprozess D200



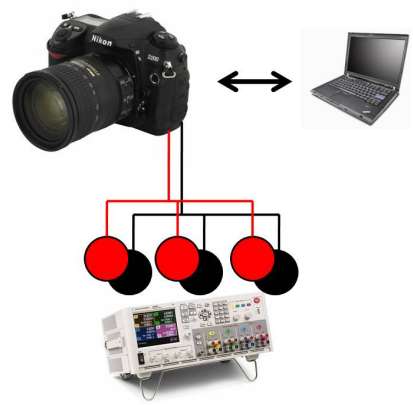


Interpretation

Der Energieverbrauch wie gemessen fällt unter den gegebenen Randbedingungen (zur Verfügung stehende Batterie, lange Capturing- Intervalle) kaum ins Gewicht. Auch die Stromspitzen beim Einschalten sind eher klein im Vergleich zum Capturing. Das zu lösende Problem für die Ansteuerung werden also die Stromspitzen beim Capturing sein. Allenfalls müssen auch die Rückflussströme berücksichtigt werden.

Messreihe 3: Langzeitverhalten

Versuch 3.1/3.2: Capturing mit immer eingeschalteter Kamera



Zielwissen:

Belastbarkeit des USB- Interfaces. Test der D70s im 9V- Betrieb.

Versuchsaufbau:

Capturing alle 10 Minuten mittels dem gphoto2-Befehl

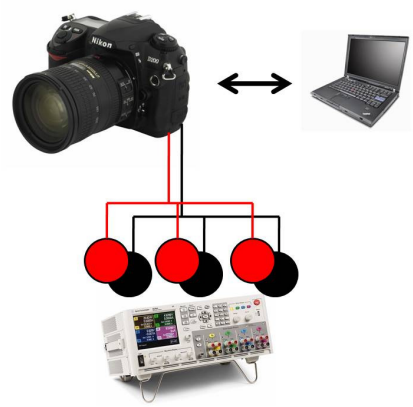
Gphoto2 --interval 600

D70s: konstant 9V; D200: konstant 12V

Resultat:

Keine Probleme festgestellt.

Versuch 3.3: Capturing mit Ein-/Aus- Funktion



Zielwissen:

Langzeitverhalten einer zu entwickelnden Scriptsteuerung, Test der D70 im 9.7V- Betrieb.

Versuchsaufbau:

Kamera: D70s
Spannung: 270s: 0V ; **30s**: 9.7V; Repeat;
Script: siehe Anhang, File "ScriptV1" (entspricht weitgehend dem in Messreihe 2 beschriebenen Script bis auf das Logging- Verhalten).

Resultat (Logs, Bilderliste, erstes & letztes Bild: siehe Anhang):

28 Fotos erfolgreich gespeichert. Danach gab die gphoto2 bei jedem weiteren Aufruf der Capturing-Funktion den I/O- Fehler „Could not claim USB- Device" aus. Jedes weitere Aufrufen von „gphoto2 –capture-image" führte zum gleichen Fehler. „gphoto2 –auto-detect" führte hingegen zu keinem Fehler. Ein Ein-/Ausschalten der Kamera löste das Problem ebenfalls nicht. Ein Neustart des Laptops behob es allerdings.

Interpretation:

Mögliche Ursachen dieses Verhaltens:

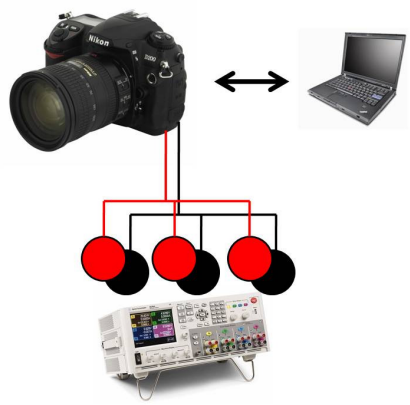
1) Zu kurze Perioden für das Schiessen der Bilder.

2) Software- Fehler auf user- Ebene, respektive der Gnome- Benutzeroberfläche.

3) Fehler auf libusb- Ebene.

Einen Fehler im Skript konnten wir ausschliessen, da auch das manuelle Aufrufen von gPhoto denselben Fehler lieferte.
Einen Fehler auf der Kamera konnten wir auch ausschliessen, da ein Neustart des Rechners, nicht aber ein Neustart der Kamera den Fehler behob.

Versuch 3.4



Zielwissen:

Einschränkung der möglichen Fehlerursachen, insbesondere von Fehler 1) in Versuch 3.3.

Versuchsaufbau:

Kamera: D70s
Spannung: 520s: 0V ; **60s**: 9.7V; Repeat;
Script: siehe Anhang, File "ScriptV2" (entspricht weitgehend dem in Messreihe 2 beschriebenen Script, bis auf das Logging- Verhalten).

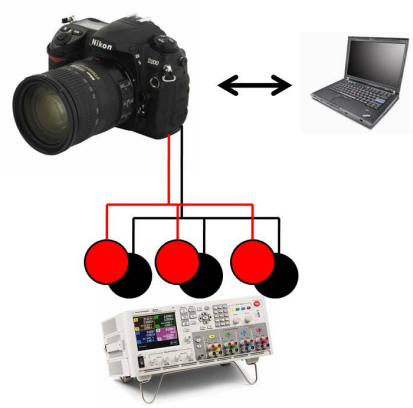
Resultat (Logs, Bilderliste, erstes & letztes Bild: siehe Anhang):

26 Fotos wurden erfolgreich gespeichert. Danach kam die gleiche Fehlermeldung wie in Versuch 3.3. Dieses Mal haben wir allerdings zum Test nur den User ausgeloggt, anstatt den Rechner neu zu booten. Resultat: Die Fehlermeldung war verschwunden.

Interpretation:

Die Wiederholung des gleichen Fehlers mit sowohl doppelt so langen Perioden zum Schiessen der Bilder, als auch verdoppelung der Pausen dazwischen, lässt Ursache Nr. 1, zu kurze Perioden, mit grosser Wahrscheinlichkeit ausschliessen. (Dies auch aufgrund der Erfahrung, dass das Bild immer schon wenige Sekunden nach Einschalten ausgelöst wird, wenn die Kamera am Laptop betrieben wird). Das Verhalten nach aus- und wieder einloggen lässt dagegen auf einen Softwarefehler unter Gnome, respektive der verwendeten USB- Library.

Versuch 3.5



Zielwissen:

Zur Sicherheit soll überprüft werden, ob das aktuelle Skript in einem Durchgang ohne Abschalten der Kamera ohne Fehler durchläuft.

Versuchsaufbau:

Kamera: D70s
Spannung: 9.7V constant
Skript: siehe Anhang, File "ScriptV3" (entspricht weitgehend dem in Messreihe beschriebenen Skript). Schiesst ein Foto in ca. 5 Minuten.

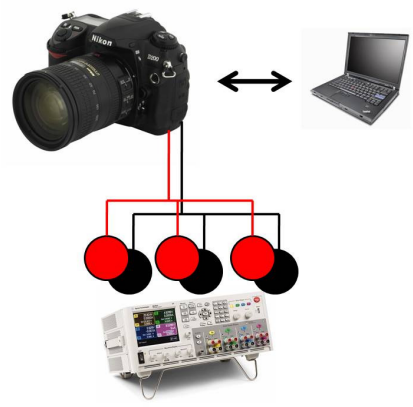
Resultat (Logs, Bilderliste, erstes & letztes Bild: siehe Anhang):

52 Fotos wurden ohne Fehlermeldung gespeichert.

Interpretation:

Der Fehler tritt also nur unter oftmaligem Verlust der USB- Verbindung auf.

Versuch 3.6



Zielwissen:

Überprüfe, ob der Fehler auch bei kurzen Ein-/Ausschalt- Perioden auftritt.

Versuchsaufbau:

Kamera: D70s
Spannung: 20s: 0V; 60s: 9.7V; Repeat;
Script: siehe Anhang, File "ScriptV3"

Resultat (Logs, Bilderliste, erstes & letztes Bild: siehe Anhang):

50 Fotos wurden ohne Fehlermeldung gespeichert.

Interpretation:

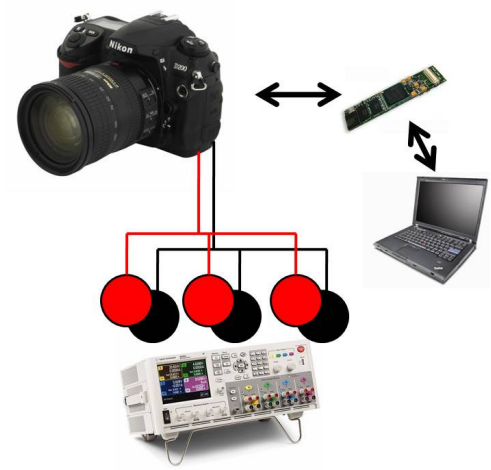
Der Fehler scheint unter anderem auch zeitabhängig zu sein. Um dies noch genauer zu untersuchen wäre aber eine zeitraubende weitere Messreihe nötig.

Arbeitshypothese

Aufgrund der gewonnen Erkenntnissen aus den Versuchen 3.3 bis 3.6 stellen wir folgende Arbeitshypothese auf:

Die Fehlermeldung „Could not claim USB- Device“ tritt in Zusammenhang mit der User- Oberfläche auf dem Laptop auf. Der Verursacher scheint weder die Kamera, noch gPhoto oder unser Script zu sein. Unter diesen Annahme sollte dasselbe Script auf dem Gumstix keine Fehlermeldung zeigen.

Versuch 3.7/3.8



Zielwissen:

Test der aufgestellten Arbeitshypothese

Versuchsaufbau:

Neu lassen wir die Kameras von einem Gumstix Embedded Linux steuern. Die Schnittstellen sind dieselben wie mit dem Laptop: PTP Protokoll über USB. Der Laptop dient nur noch zur Überwachung des Gumstix mittels Minicom und einer „CP2103 USB to UART Bridge“ von Silicon Laboratories.

Spannung: 520s : 0V; 120s: 9.7V (D70s) / 12V (D200); Repeat;

Script: siehe Anhang, File “ScriptV3”

Anmerkung zur Spannungsfunktion: Wir mussten feststellen, dass der Gumstix für das Abarbeiten des Skripts wesentlich länger als der Laptop benötigt. Das Auslösen des Fotos dauert nach Einschalten oft 30 Sekunden. Dies warscheinlich deswegen, weil der Aufruf „gphoto2 –auto-detect“, der zum Detektieren der Kamera benötigt wird, einen grossen Overhead erzeugt. Aus diesem Grund haben wir die eingeschaltete Periode zur Sicherheit auf 120s erhöht. Die Off- Periode von 520s ist die maximal mögliche für den Agilent N6705A Power Analyser (um den Speicherplatz des Gumstix möglichst lange ausnützen zu können).

Mit der D70s wurde diese Konfiguration während einem ganzen Wochenende getestet. Für die D200 haben wir dafür hingegen in Anbetracht des Resultats der D70s nur noch etwa 16h aufgewendet.

Resultat (Logs, Bilderliste, erstes & letztes Bild: siehe Anhang):

D70s: nach dem 100. Foto war der Speicher des Gumstix voll. Danach wurde zwar gemäss Konsolen-Output die Bilder weiter geschossen aber nicht mehr gespeichert. Dies wurde im Laufe des Wochenendes noch 620 Mal wiederholt.

D200: in ca. 17h hat die D200 mit dieser Konfiguration genau 100 Fotos geschossen und abgespeichert, Fehler sind keine aufgetreten.

Interpretation

Wir können aufgrund dieser Ergebnisse sagen, dass der Fehler auf dem Gumstix nicht auftreten sollte und unsere Arbeitshypothese wurde bestätigt.

Zusammenfassung der gewonnnen Erkenntnisse

- Ohne Aktivität verbrauchen die Kameras >1.8 Watt (D70s), respektive >3.1W (D200) im eingeschalteten Zustand, was für einen Langzeitbetrieb unter den gegebenen Umständen sicherlich zuviel ist. Ein-/Ausschalten bei Bedarf wird unumgänglich sein.

- Eine solche Schaltung muss darauf ausgelegt sein, bis zu 4A Stromspitzen (D70s), respektive 3A (D200) aushalten zu können.

- Im Langzeitbetrieb erscheint die Ansteuerung mit einem Embedded System, wie wir es einsetzen werden, fehlerresistenter zu sein als mit einem PC. Der Einsatz von gPhoto mit einem Gumstix- System ist problemlos möglich.

Anhang

Logdaten, Dateilisten, erste und letzte Bilder zur Messreihe 3, sowie alle Scriptversionen können unter der URL

http://people.ee.ethz.ch/~muellmic/muellermichel/uploads/mountainview/Anhang.zip

heruntergeladen werden.

Messungen zum Projekt PermaSense: Mountainview

Nr.	Zielwissen	Dauer	Datum	Modell	U	Samples	Steuergerät	Resultat
1.1	Energieverbrauch im eingeschalteten Zustand ohne Aktivität. Ermittlung von eventuellen automatischen Energiesparmodi.	2h	16.10.08	D70S	9.4V konst	50ms	Keines	Die Kamera verbraucht durchgehend gleich viel Leistung (1.82 – 1.98 W), es gibt keinen Energiesparmodus.
1.2	Siehe 1.1	2h	17.10.08	D200	12V konst	50ms	Keines	Siehe 1.1. (3.13 – 3.67 W)
2.1	Stromspitzen beim Capturing mittels PTP über USB. Stromspitzen beim Einschaltvorgang. Energieverbrauch beim Capturing.	ca. 2.5 min	24.10.08	D70S	10s: 0V; 30s: 9.7V; Repeat 3x;	1 ms	Ubuntu- Laptop, Messungsscript v 3 Spannung ein/aus mit Power Analyser	3.49A Spitze beim Capturing. Einschaltstrom- Maxima und Energieverbrauch nicht relevant.
2.2	Siehe 2.1	ca. 2.5 min	24.10.08	D200	10s: 0V; 30s: 12V; Repeat 3x;	1 ms	Ubuntu- Laptop, Messungsscript v 3 Spannung ein/aus mit Power Analyser	Siehe 2.1 (2.59A Spitze beim Capturing)
3.1	Langzeitverhalten beim Capturing im immer eingeschalteten Zustand	5h	15.10.08	D70S	9 V konst	n.a.	Ubuntu- Laptop, gphoto2 im Intervall- Modus	Fotos werden erfolgreich gespeichert.
3.2	Siehe 3.1	5h	15.10.08	D200	12V konst	n.a.	Ubuntu- Laptop, gphoto2 im Intervall- Modus	Fotos werden erfolgreich gespeichert.
3.3	Langzeitverhalten beim Ein- / Ausschalten plus Capturing.	10h	22.10.08	D70S	270s: 0V; 30s: 9.7V; Repeat;	n.a.	Ubuntu- Laptop, Messungsscript v 1 Spannung ein/aus mit Power Analyser	28 Fotos erfolgreich gespeichert, dann Fehlermeldung bei jedem weiteren.
3.4	Anpassung an Testresultat 3.4: Verlängerung der ON- Periode auf 60s. Test mit neuem Script, der die Capturing- Zeitpunkte loggt zwecks Debugging.	10h	23.10.08	D70S	520s: 0V; 60s: 9.4V ; Repeat;	n.a.	Ubuntu- Laptop, Messungsscript v 2 Spannung ein/aus mit Power Analyser	26 Fotos erfolgreich gespeichert, dann Fehlermeldung bei jedem weiteren.

3.5	Schritt zurück: Test des Verhaltens mit aktuellem Script im immer eingeschalteten Zustand	Ca. 1h	24.10.08	D70S	9.7V Konst	n.a.	Ubuntu- Laptop, Messungsscript v 3	52 Fotos erfolgreich gespeichert, keine Fehlermeldung.
3.6	Test des aktuellen Scripts beim Ein-/ Ausschalten, kurze OFF-Perioden	Ca. 1.5h	24.10.08	D70S	20s: 0V; 60s: 9.7V; Repeat;	n.a.	Ubuntu- Laptop, Messungsscript v 3 Spannung ein/aus mit Power Analyser	50 Fotos erfolgreich gespeichert, keine Fehlermeldung.
3.7	Hypothese: Fehlermeldung tritt auf auf user-Ebene auf bei OS mit grafischer Oberfläche im Langzeitbetrieb. Deshalb: Test des Langzeitverhaltens auf OpenEmbedded-Gumstix.	>48h	27.10.08 Bis 28.10.08	D70s	520s: 0V; 120s: 12V ; Repeat;	n.a.	OpenEmbedded-Gumstix, Messungsscript v 3 Spannung ein/aus mit Power Analyser	124 Fotos erfolgreich gespeichert, danach Memory voll. In der restlichen Zeit 604 Fotos erfolgreich ausgelöst aber nicht gespeichert.
3.8	Siehe 3.7	>16h	24.10.08 bis 26.10.08	D200	520s: 0V; 120s: 12V ; Repeat;	n.a.	OpenEmbedded-Gumstix, Messungsscript v 3 Spannung ein/aus mit Power Analyser	100 Fotos erfolgreich gespeichert, keine Fehlermeldung.

A.2 Präsentationsfolien

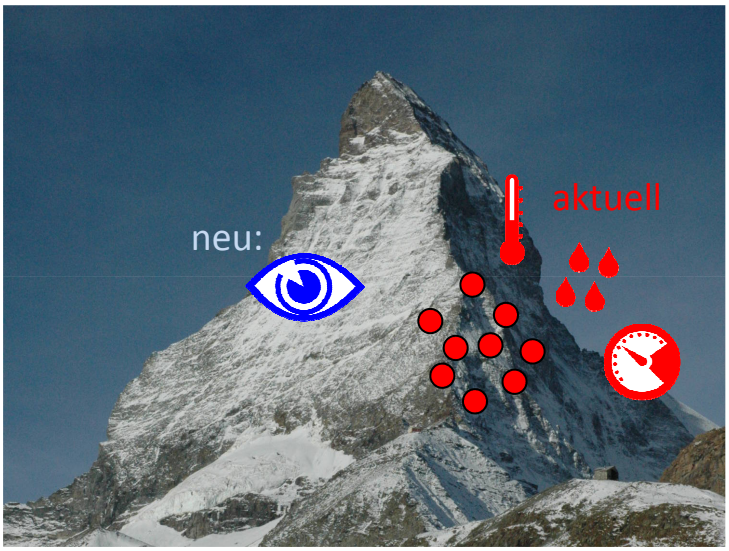
Gruppenarbeit "Mountainview"

Eine Kamera für das PermaSense Projekt

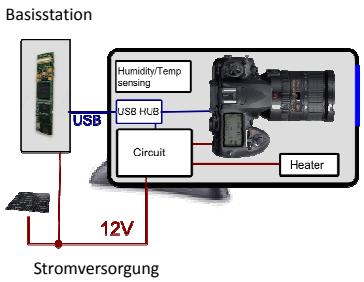
Stefan Kronig, Michel Müller
Tutoren: Jan Beutel, Matthias Keller

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

TIK Institut für Technische Informatik und Kommunikationsnetze



Vorgaben auf Hardwareebene



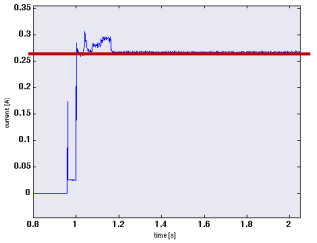
Funktion
- **Bildqualität**
- **Wetterfestigkeit**
- **Einfache Montage**

Zuverlässigkeit
- Ferngesteuertes „Reset“
- Temp. / Feuchtigkeit überwachbar + steuerbar
- Fail-Safe-Verhalten

Optimierung
- Stromverbrauch im Standby

Strommessungen D200

Kontinuierlicher Verbrauch

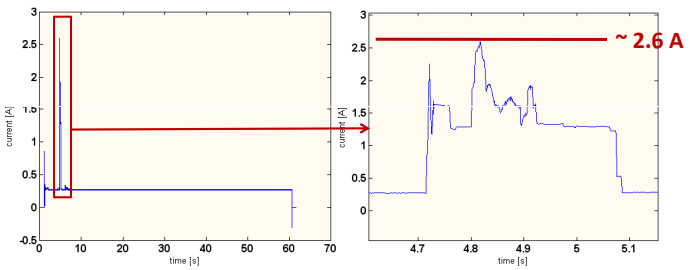


- Kein Energiesparmodus im Netzbetrieb
- Kontinuierlicher Stromverbrauch zu hoch für permanenten Betrieb
- Manuelles Ausschalten erforderlich

Optim

Strommessungen D200

Gesamtverlauf | Auslösen eines Fotos



Schaltung muss mindestens auf 3A-Spitzen ausgelegt werden.

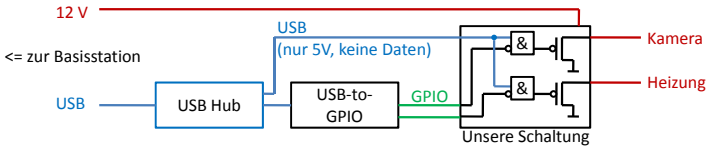
Zuverl.

Funktionen der Hardware

- Kamera & Heizung schalten
  - Mittels FETs, geeignet dimensionieren (Reserve, Stromverbrauch im Standby)
- Anforderung: nur 1 Box mit 2 Anschlüssen
  - 1 Stromkabel (12 V) & 1 Datenkabel (USB)
- Leitungen müssen also via USB schaltbar sein.
  - Modul zur Kontrolle von Steuerleitungen (GPIO) über USB
  - Steuerleitungen treiben Gates von FETs zur Steuerung der 12V- Leitungen.

Funktion

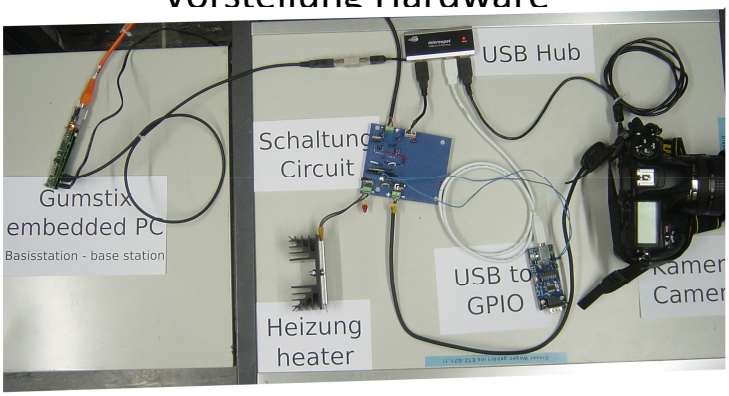
Optim.

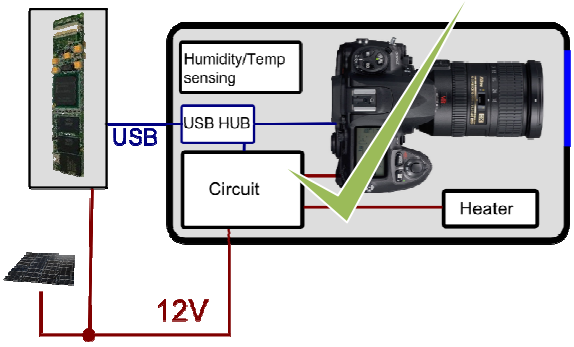


- „Notfallplan“: Störung am USB- Hub oder am USB-to-GPIO- Modul
  - USB ist an der Basisstation „virtuell aussteckbar“.
  - Schaltung so gebaut, dass ohne USB- Spannung die FETs sperren -> Kamera, Heizung sind aus.
- Überwachung Temperatur & Feuchtigkeit
  - Nutzt bereits vorhandenes System: TinyNode

Zuverl.

Vorstellung Hardware





Funktion
Zuverl.
Optim.

Vorgaben auf Softwareebene

- Autonomie
- Manuelles Eingreifen möglich

Funktion

- Logging

Zuverlässigkeit

- Geschwindigkeit (Stromverbrauch)

Optimierung

Steuerung auf Softwareebene

OpenEmbedded Linux Umgebung

Software- Interface zur Steuerung der Kamera

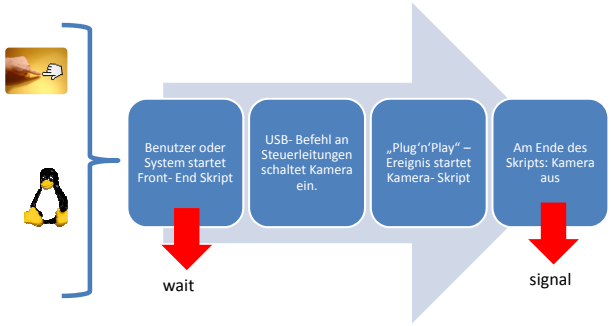
Bourne Shell Scriptsteuerung → Flexibilität, Anpassungsfähigkeit

Triggerung der USB- Ereignisse mittels „Plug‘n‘Play“ (udev) → Kein Polling oder Eventhandling nötig

Doppelte Speicherung: Auf SD- Card der Kamera & Gumstix
→ Restore- Funktionalität

Funktion
Zuverl.
Optim.

Demo / Ablauf der Software



Demonstration / Fazit

	Software	Hardware
Funktion	•Autonomie •Manuelles Eingreifen möglich	•Bildqualität •Wetterfestigkeit •Einfache Montage
Zuverlässigkeit	•Logging	•Ferngesteuertes „Reset" •Temp. / Feuchtigkeit überwachbar + steuerbar •Fail-Safe-Verhalten
Optimierung	•Geschwindigkeit (Stromverbrauch)	•Stromverbrauch im Standby

A.3 Beispiel einer Logdatei

```
01 ==>check local memory<==================================
02 on the local filesystem, there are 6461 kB available
03 ============================================================
04 ==>free local memory<===================================
05 WARNING: free memory on local filesystem is unsufficient. one RAW- Picture will be deleted
       now.
06 ..Operation rm successfully executed
07 ..Operation rm successfully executed
08 1229333102_DSC_0140.NEF and thumb_1229333102_DSC_0140.NEF deleted.
09 ============================================================
10 ==>detecting camera<====================================
11 ..Operation gphoto2 --autodetect successfully executed
12 ============================================================
13 ==>determine free space on camera<=======================
14 ..Operation gphoto2 --storage-info successfully executed
15 on the camera, there are 64992 kB available
16 ============================================================
17 ==>memory warning<========================================
18 WARNING: free memory on camera is low. Oldest pictures will be lost soon.
19 ============================================================
20 ==>adjusting camerasettings<=============================
21 ..Operation gphoto2 --set-config imgquality=x successfully executed
22 ..Operation gphoto2 --set-config capturetarget=x successfully executed
23 ============================================================
24 ==>shooting picture<======================================
25 Picture saved on camera as expected. filename: DSC_0148.NEF
26 ..Operation gphoto2 --capture-image successfully executed
27 ============================================================
28 ==>downloading picture<===================================
29 picture successfully saved in /mview_testrun8/images/1229344940_DSC_0148.NEF
30 ..Operation gphoto2 --get-file successfully executed
31 ============================================================
32 ==>downloading thumbnail<=================================
33 thumbnail successfully saved in /mview_testrun8/thumbs/thumb_1229344940_DSC_0148.NEF
34 ..Operation gphoto2 --get-thumbnail successfully executed
35 ============================================================
```

A.4 Schema des Prototyps

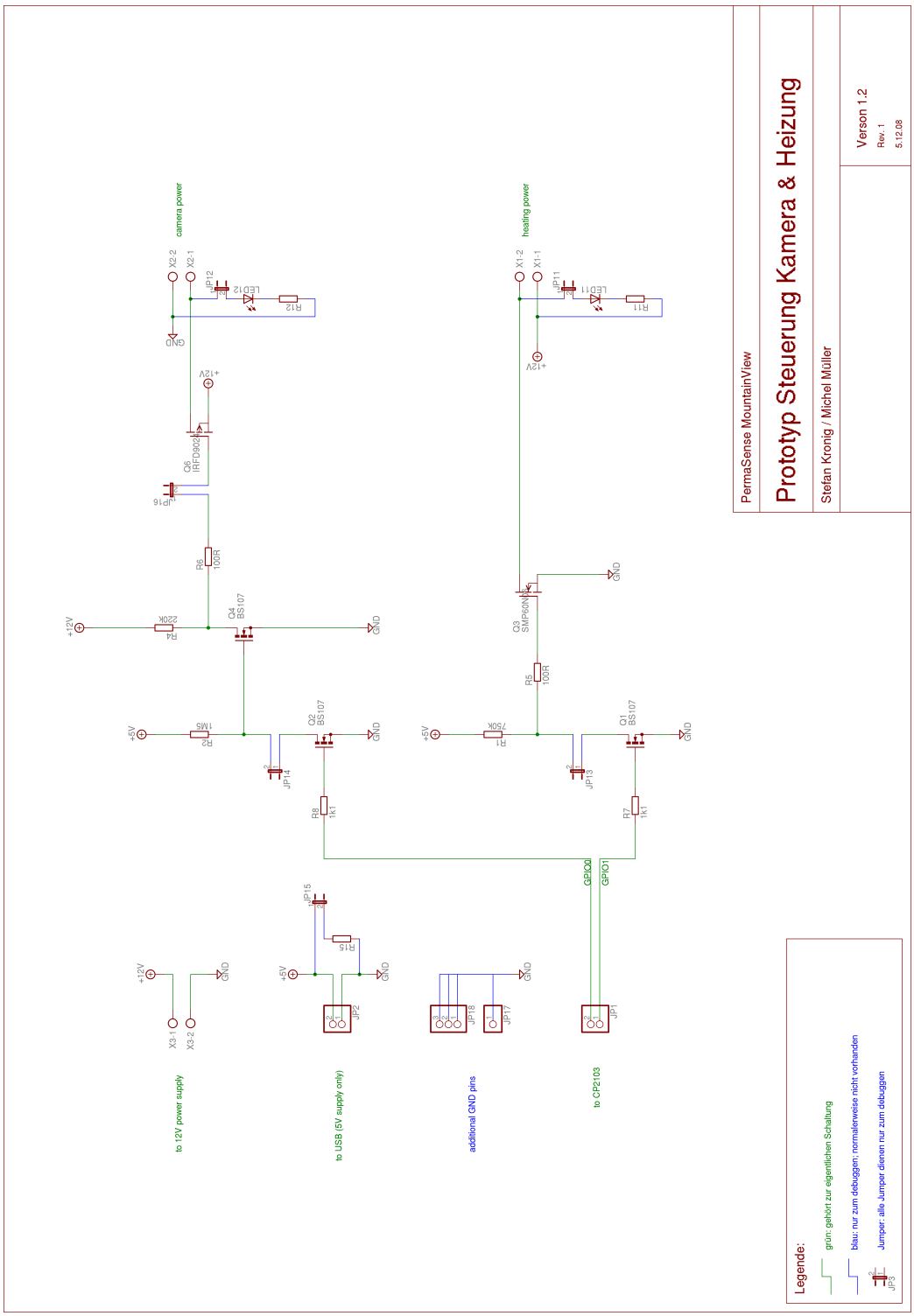


Abbildung A.1: Schema der Hardware

A.5 Übersicht der Software-Komponenten

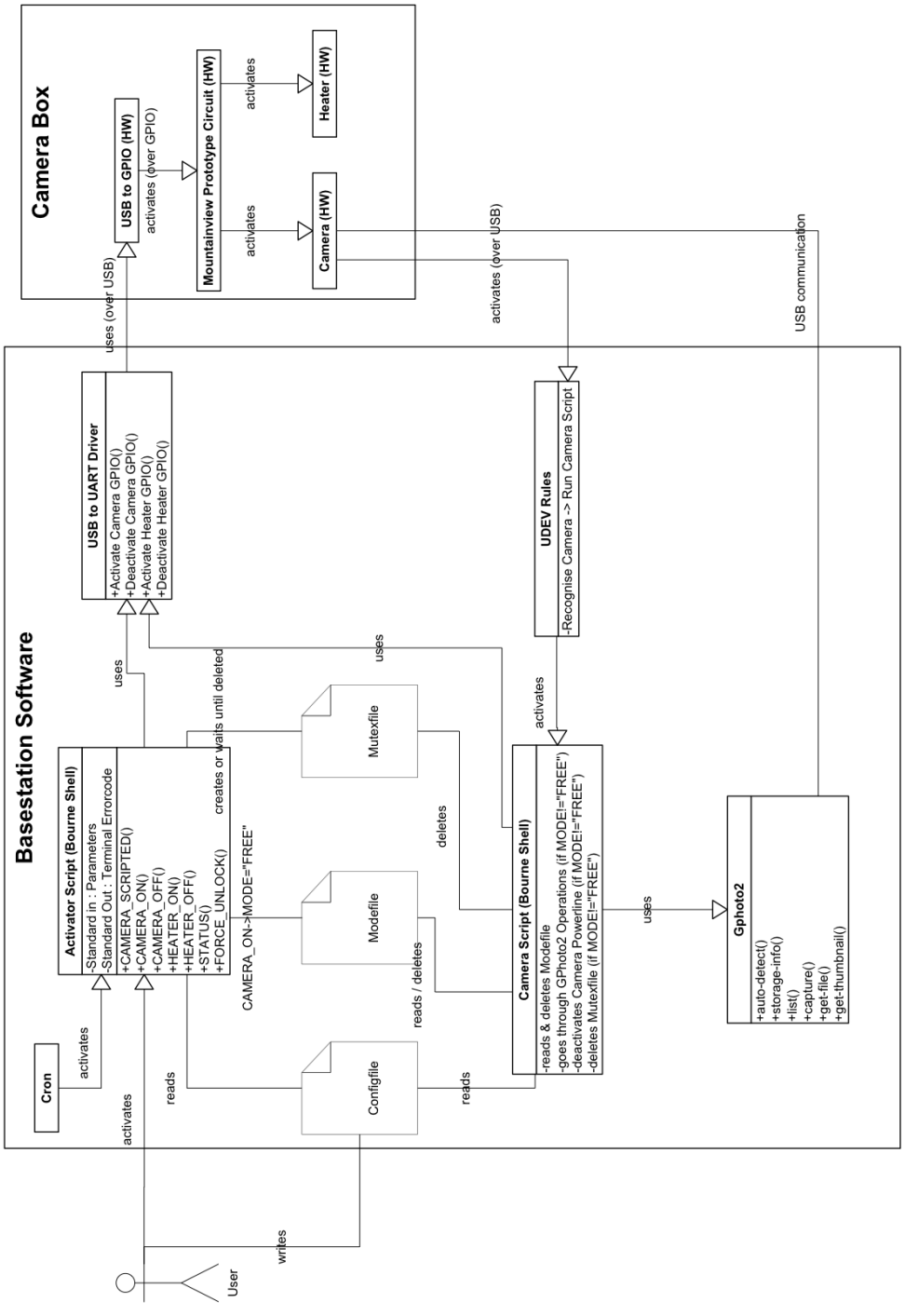


Abbildung A.2: Diagramm der Softwarekomponenten

Tabellenverzeichnis

Abbildungsverzeichnis

Abkürzungen

API	Application Programming Interface
ESD	Electrostatic Discharge (elektrostatische Entladung)
GND	Ground (Erdung)
GPIO	General Purpose Input/Output
GSM	Global System for Mobile communications (in der Schweiz auch bekannt als Natel D)
HW	Hardware
ITET	Departement für Informationstechnologie und Elektrotechnik der ETH Zürich
MP	Megapixels (1 Million Pixel)
Mutex	Mutual Exclusion (gegenseitiger Ausschluss)
NEF	Nikon Electronic Format (“Raw”-Format der Nikon-Kameras)
PTC	Positive Temperature Coefficient
PTP	Picture Transfer Protocol
SD (Mem. Card)	Secure Digital Memory Card
SW	Software
UART	Universal Asynchronous Receiver Transmitter serielle Kommunikationsschnittstelle
USB	Universal Serial Bus
WLAN	Wireless Local Area Network

Literaturverzeichnis

[1] Igor Talzi, Andreas Hasler, Stephan Gruber, Christian Tschudin: *PermaSense: Investigating Permafrost with a WSN in the Swiss Alps*. Seite 1.

[2] Gumstix, inc: *Verdex Pro Overview*. Website, Oktober 2008. http://www.gumstix.net/Hardware/view/Hardware-Specifications/Verdex-Pro-Specifications/112.html.

[3] European Thermodynamics Ltd.: *HP Series of PTC Heaters*. Website, 2002. http://www.europeanthermodynamics.com/heaters/HP%20Products.pdf.

[4] Liechti, Chris: *pySerial*. Website, Oktober 2008. http://pyserial.wiki.sourceforge.net.

[5] Silicon Laboratories Inc.: *CP2103 Datasheet*. Website, Mai 2007. https://www.silabs.com/Support%20Documents/TechnicalDocs/cp2103.pdf.

[6] Paul Horowitz, Winfried Hill: *The Art of Electronics*, Kapitel 3. The Press Syndicate of the University of Cambridge, 2. Auflage, 1989.

[7] Vishay Siliconix: *BS107 Datasheet*. Website. http://www.datasheetcatalog.org/datasheet/vishay/70215.pdf.

[8] Vishay Siliconix: *SMP60N06 Datasheet*. Website. http://www.datasheetcatalog.org/datasheets/320/500027_DS.pdf.

[9] IRF / Vishay Siliconix: *IRFD9024 Pbf Datasheet*. Website. http://www.vishay.com/docs/91137/91137.pdf.

[10] Tim Waugh, Hans Ulrich Niedermann, Michael J. Rensing u.a.: *GPhoto2 Dokumentation*. Website, Dezember 2006. http://www.gphoto.org/doc/manual.

[11] Carla Schroder: *Manage Linux Hardware with udev*. Website, Oktober 2006. http://www.enterprisenetworkingplanet.com/nethub/article.php/3635686.

[12] Drake, Daniel: *Writing udev rules*. Website. http://reactivated.net/writing_udev_rules.html.

[13] Ubuntu Community: *cron HowTo*. Website, Dezember 2008. https://help.ubuntu.com/community/CronHowto.