Gatekeeper (Schließanlage)

Follow-Up zur SpaceNotification und altem Schließanlagen-Projekt, konkret geht es um die Schließanlage.

Meta

Maintainer
Open for adoption

Pad
https://pad.n39.eu/p/2014_Gatekeeper

Git-Repo
https://github.com/netz39/space_notification

Git-Repo
https://github.com/netz39/rollladensteuerung

Git-Repo
https://github.com/netz39/xmpp_space_control.git

Git-Repo
https://github.com/netz39/i2cbridge

ToDo

nächste Schritte/TODOs

- ~~Repositories aufräumen~~
- Debian-init-Scripte für den i3c_client
  - Prototypisch schon vorhanden (Tux fragen)
- Debian-Pakete für space-control-bibliothek, i3c_client und i3c_cli
  - derzeit nur über checkinstall verfügbar
  - da müssen richtige Debian-Pakete gebaut werden
- XMPP-Agenten für Umsetzung von I3C auf logische Devices
  - Ampel
  - SpaceStatus
  - Rollläden
  - Türschloss
- bessere Fehlererkennung/-Korrektur in der Firmware
  - am besten als Bibliothek/Modul umsetzen
- ~~Entprellung für Steuerleitungen im Tür-Controller~~
  - http://hackaday.com/2015/12/10/embed-with-elliot-debounce-your-noisy-buttons-part-ii/
  - https://github.com/netz39/rollladensteuerung/tree/master/debounce

    - fixed in https://github.com/netz39/rollladensteuerung/pull/2

übernächste Schritte

- Reviews
    - Firmware
    - space-control-bibliothek
    - i3c_client
    - i3c_cli
- space-control-bibliothek: message queues
- i3c_client: i3c-Erkennung

Aufbau und Funktionsweise

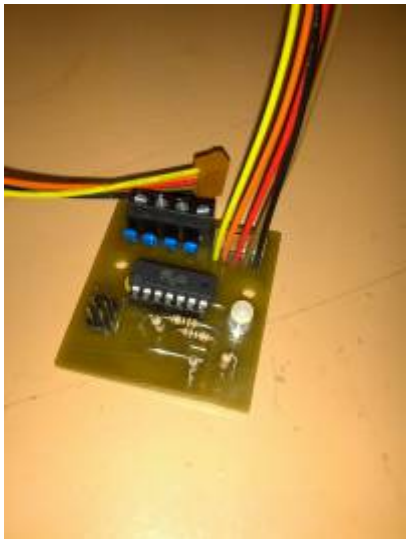
Software

Es gibt ein Failsafe-Script, das die Tür abschließt, wenn die Tür offen ist, aber seit 30 Sekunden die SpaceTime inaktiv (aka Ampel rot/aus) ist.

Komponenten kommunizieren per XMPP

README im GitHub

Hardware

Authentifizierung

Telefon

Ansprechpartner
Tux

SSH

Ansprechpartner
Basti

mit einer Webcam und einem QR-Code

Ansprechpartner
Michel

- Links:
  - http://www.jeremyblum.com/portfolio/libetech/
  - https://github.com/sciguy14/LibeTech-QR-Entry
  - https://github.com/sciguy14/LibeTech-QR-WebSystem

I3C-Bus

Der I3C-Bus besteht aus 5 Leitungen:

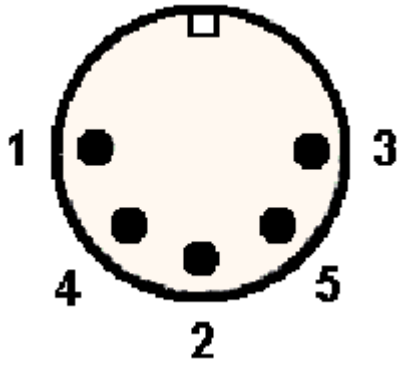
- SDA und SCL für I²C
- einer Interrupt-Leitung INT
- Versorgungsspannung 5V
- Masse

Für die Verbindung zwischen Geräten werden Steckverbinder oder 5-polige DIN-Buchsen (Reichelt MAB 5) verwendet.

Belegung Steckverbinder

1. SDA
2. SCL
3. INT
4. Vcc (5V)
5. GND

Belegung DIN-Buchse



(Draufsicht)

1. GND
2. INT
3. SDA
4. Vcc (5V)
5. SCL

I3C-Kommandos

Wenn kein data-Wert spezifiziert wird, ist der Parameter nicht relevant. Wenn kein output spezifiziert wird, bedeutet 1 Erfolg. 0 bedeutet immer Fehler einen Parity-Fehler und sollte zum erneuten Aufruf führen.

Device 0x20: Ampel

```
#define CMD_I3C_RESET 0x00
#define CMD_GETLIGHT  0x01
#define CMD_SETLIGHT  0x02
```

- I3C_RESET: Interrupt-Leitung zurücksetzen (sollte bei der Ampel derzeit nicht nötig sein)
- GETLIGHT: Ampelstatus zurückliefern
- SETLIGHT: Ampelstatus setzen

```
data (DDDD)
  1 bit blink-Status
  3 bit Farbe: 0=keine, 1=rot, 2=grün
```

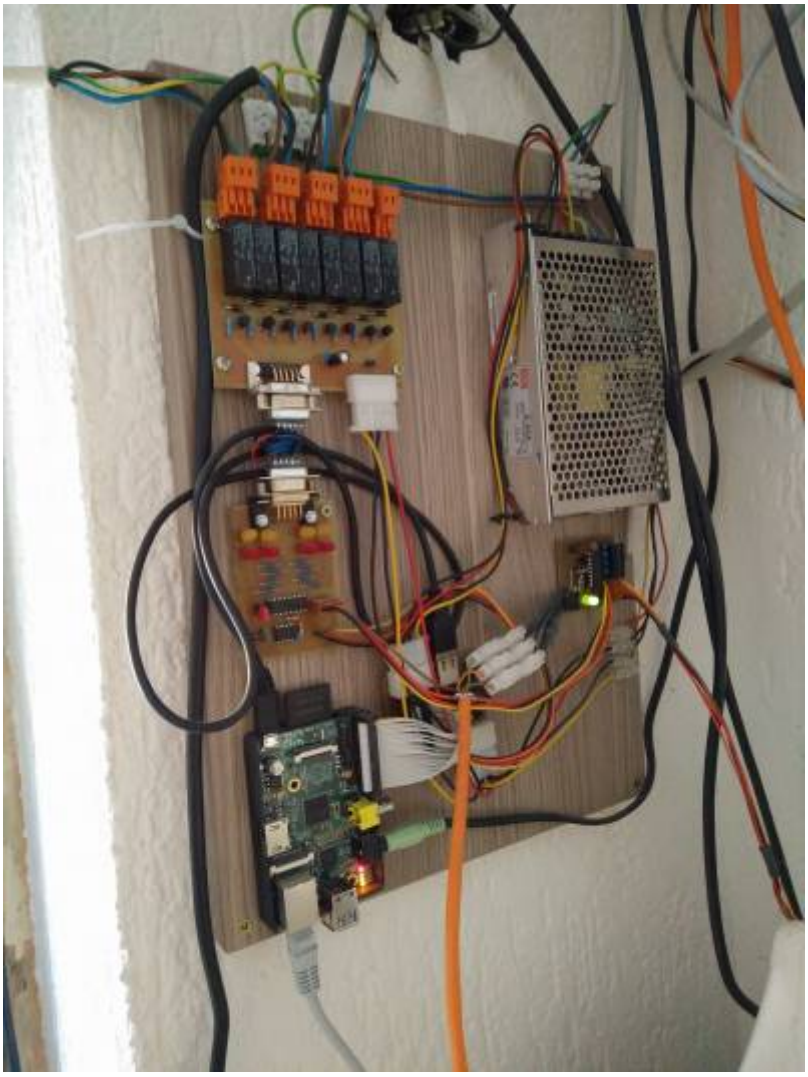

oboldine
Floki
DES LEBENS
KDE
Perl

Device 0x21: Controller Rollläden

```
#define CMD_ALL_STOP  0x0
#define CMD_STOP      0x1
#define CMD_UP        0x2
#define CMD_DOWN      0x3
```

- ALL_STOP: alle Rollläden stoppen
- STOP: Rollladen aus *data* anhalten
- UP: Rollladen aus *data* hochfahren
- DOWN: Rollladen aus *data* herunterfahren

data gibt jeweils die Nummer des Rollladens (0: fenster bastelbereich, 1: tür bastelbereich, 2: tür lounge, 3: fenster lounge) an Ausgabe: 0 == fehler, 1 == erfolg



Device 0x22: Manuellsteuerung

```
#define CMD_RESET       0x00
#define CMD_BEEP        0x01
#define CMD_MANUAL_MODE 0x02
#define CMD_GET_SWITCH  0x03
#define CMD_MANUAL_SW   0x05
```

- RESET: I3C-Interrupt-Status zurücksetzen
- BEEP: Summer nach Muster aktivieren
  - data enthält das Bitmuster für die Aktivierung
- MANUAL_MODE: LED-Anzeige setzen
  - data: 0 = aus, 1 = langsam blinken, 2 = schnell blinken, 3 = an
- GET_SWITCH: Schalterstellung auslesen
  - data: Nummer des Schalters, korrespondierend zur Rollladen-Nummer
  - output: 1 = hoch, 2 = runter, 3 = neutral
- MANUAL_SW: Status des Tasters setzen/auslesen
  - data: 1 = Blockstatus löschen, 2 = Blockstatus setzen, sonst keine Änderung
  - output: 1 = Blockstatus gesetzt, 2 = Blockstatus gelöscht (Wert vor Manipulation)

Device 0x23: Tür-Controller

```
#define CMD_RESET       0x00
#define CMD_OPEN        0x01
#define CMD_CLOSE       0x02
#define CMD_STATE       0x03
```

- RESET: Tür-Status zurücksetzen, inklusive I3C-Interrupt
- OPEN: Tür öffnen
- CLOSE: Tür schließen
- STATE: Tür-Status zurückliefern. Löscht auch den I3C-Interrupt
    - output enthält Bitmaske:

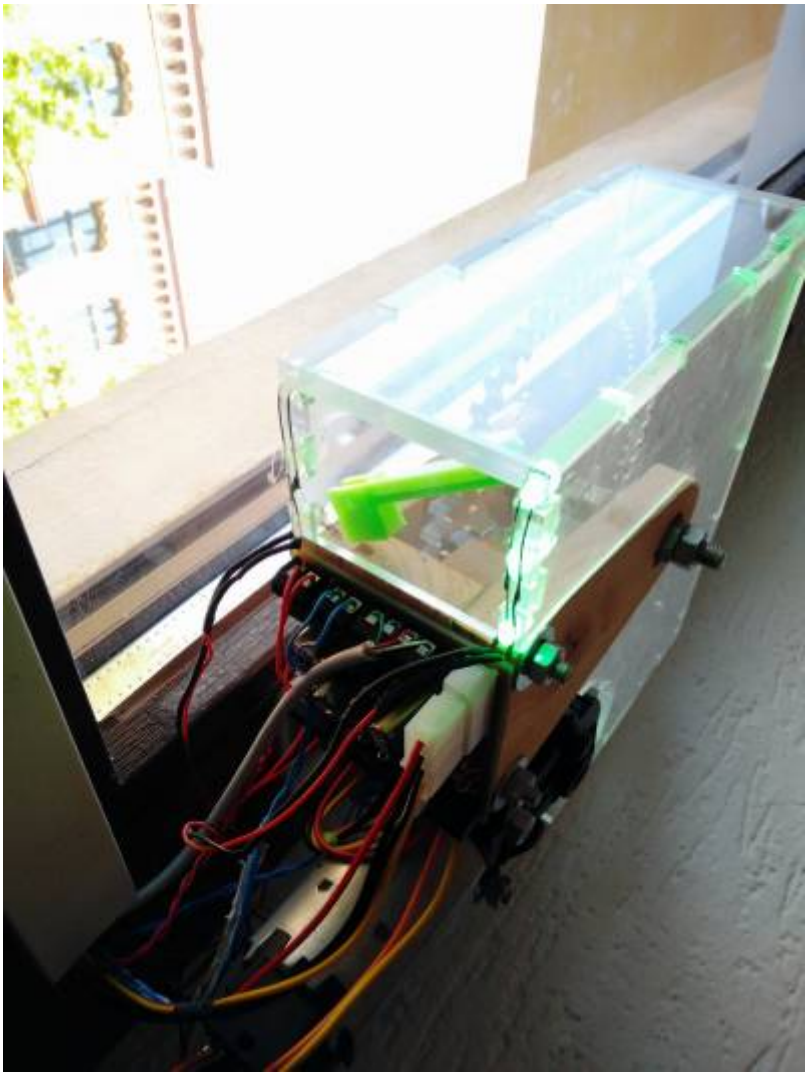
```
Input Status Byte (ISB)

 +-----+----+----+----+----+----+----+
 | 7-6 | 5  | 4  | 3  | 2  | 1  | 0  |
 | res | GB | RB | DC | LO | FC | FO |
 +-----+----+----+----+----+----+----+

 GB Green Button active (Force-open door)
```

```
RB Red Button active (Force-close door)
DO Door Open
LC Lock Closed
FC Force Close
FO Force Open
```

- Bit-Bedeutung
  - DO: Tür steht offen (1) oder ist geschlossen (0)
  - LC: Schloss verriegelt (1) / offen (0)
  - FC: Force Close – Signalleitung "Verriegeln" ist aktiv
  - FO: Force Open – Signalleitung "Öffnen" ist aktiv
  - RB, GB sind die beiden Buttons (rot/grün) an der Tür



Device 0x24: SpaceStatus-Switch

```
#define CMD_RESET        0x00
#define CMD_GETSTATE     0x01
#define CMD_SETSTATE     0x02
```

- RESET: Tür-Status zurücksetzen, inklusive I3C-Interrupt
- GETSTATE: Aktivierten SpaceStatus zurückgeben (1 closed, 2 open, 3 unbekannt)
- SETSTATE: Setzt einen SpaceStatus (verhindert Notifications über Änderungen, wenn der eig. Status schon angepasst ist)

Ideen

- USV
  - https://hackaday.com/2016/11/26/diying-a-raspberry-pi-power-bank/