Dipl.-Ing. Michael Zimmermann
Buchenstr. 15
42699 Solingen
☎ 0212 46267
https://kruemelsoft.hier-im-netz.de
BwMichelstadt@t-online.de

Michelstadt (Bw)

Übersicht

Diese Zusammenstellung wurde nach bestem Wissen
und ohne Funktionsgarantie in der Hoffnung erstellt, dass sie nützlich ist.
Wenn sie nicht nützlich ist – dann eben nicht.

Handregler für die Modellbahn – Software aufspielen („flashen“)

Seit der Verwendung des Digitalsystems DCC[1] werden bei mir und bei jedem Modultreffen Handregler verwendet, die an das LocoNET®[2] angeschlossen werden. Die verwendeten Handregler wurden vom FREMO[3] entwickelt und werden auch weiterentwickelt werden[4]. Diese Info soll zeigen, wie die Software auf den Handregler kommt – erläutert also das sogenannte „flashen“.

In dieser Info wird das flashen der FREMO-Handregler (kurz: FREDI) beschrieben, es wird nicht eingegangen auf:

- FREDs der ersten Generation („klassischer FRED“), die einen PIC als Controller haben
- Handregler, die nicht mit dem LocoNET® verwendbar sind
- kommerzielle Produkte wie z.B.:
  - Die von Roco bzw. Fleischmann verfügbare Lok- oder Funkmaus
  - Handregler von Digitrax, Uhlenbrock, Lenz usw.

Die Namensgebung der Handregler im FREMO ist einfach: der Grundbestandteil der Namen ist immer **FRED: Fremos einfacher Drehregler**.

[1] DCC = Digital Command Control, https://de.wikipedia.org/wiki/Digital_Command_Control
[2] LocoNET® = Bussystem, https://de.wikipedia.org/wiki/LocoNet
[3] FREMO = Freundeskreis europäischer Modellbahner, https://www.fremo-net.eu/
[4] Eine Übersicht der FREMO-Handregler gibt es hier: Krümelbahn Info 8 - Handregler für die Modellbahn (https://github.com/Kruemelbahn/Infoletter/blob/main/Krümelbahn%20Info%208%20-%20Handregler%20für%20die%20Modellbahn.pdf)

Voraussetzungen

Um die Software auf einen FREDI zu flashen werden eine entsprechende Hardware und natürlich die Software benötigt: Software um die Hardware zu bedienen und natürlich die Software, die auf den FREDI gebracht werden soll.

Benötigte Hardware

Grundvoraussetzung ist hier:

- ein PC oder Laptop, der wenigstens über eine USB-Schnittstelle verfügt. Tatsächlich ist auch ein RaspBerry Pi (ich verwende hier einen RaspBerry Pi 3 Model B Rev 1.2) einsetzbar.
- für das erstmalige flashen der Software wird auch ein AVR-Programmiergerät benötigt. Updates der Software können auch ohne Programmiergerät auf den FREDI gebracht werden, siehe hier.
  Als AVR-Programmer für FCalib2 kommt bei mir ein DIAMAX-Prog S2 zum Einsatz (siehe Anhang A), es können aber auch andere Programmiergeräte verwendet werden. Wichtig hierbei ist jedoch, dass das AVR-Programmiergerät die Möglichkeit hat, eine Spannung von 3,3V für den FREDI einzustellen.

Benötigte Software

Das verwendete Betriebssystem spielt hier eine untergeordnete Rolle, hier können sowohl Windows als auch Linux verwendet werden.

Zusätzlich zum Betriebssystem werden nachfolgende Softwarepakete benötigt, diese sind als FreeWare im Internet verfügbar:

- AvrDude (https://www.nongnu.org/avrdude/)
  Eine Software, die als Schnittstelle zwischen FClaib2 und der Hardware des AVR-Programmiergerätes benötigt wird.
- FCalib2 (https://sourceforge.net/projects/fremodcc/files/FCalib2/)
  Hiermit können:
  - Die FREDIs verwaltet werden
  - Den FREDIs eine ID („Throttle-ID“) zugewiesen werde
  - Die Software auf den FREDI gebracht werden (sowohl initial als auch ein Update)

  Fcalib2 beinhaltet weiterhin alle Software-Versionen für die FREDIs.
- JMRI (https://www.jmri.org/)
  Hiermit können u.a.:
  - Einstellungen für FREDIs gelesen, geschrieben und verwaltet werden
  - Ein Up- oder Downgrade der FREDI-Software durchgeführt werden, ohne dass ein AVR-Programmiergerät benötigt wird.

Alle Softwarepakete sind auf den Betriebssystemen Windows und Linux[5] einsetzbar. Voraussetzung ist:

- Es wird zwingend ein JAVA-RuntimeEnvironment (JRE, https://www.oracle.com/) benötigt, ein JAVA-SDK ist nicht erforderlich (kann aber an Stelle des JRE auch verwendet werden…)
  - Für die aktuelle JMRI-Version wird JAVA-Version 11 benötigt (siehe auch hier: https://www.jmri.org/java/index.shtml).
  - FCalib2 läuft auch zusammen mit älteren JAVA-Versionen.

[5] Siehe Anhang B

FCalib2

Wird einem FREDI erstmals eine Software aufgespielt, so empfiehlt sich die Verwendung von FCalib2 zusammen mit einem AVR-Programmiergerät (siehe Anhang A).

Diese Vorgehensweise hat den Vorteil, dass

- die FREDI-Id eingegeben werden kann
- die dann zusammen mit der ausgewählten Software auf den AVR geflasht wird
- und auch die AVR-Fuses[6] gesetzt werden

ohne, dass man sich um weitere Besonderheiten eines AVR kümmern muss.

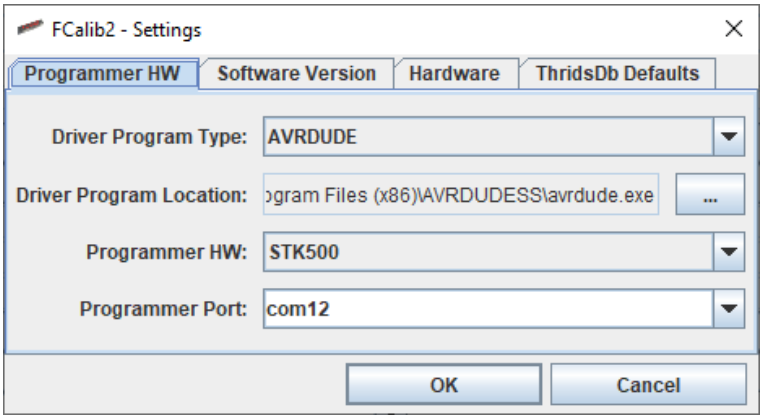
FCalib2 kann mit einer Batch-Datei gestartet werden (`start-fcalib2.bat` unter Windows oder `start-fcalib2.sh` unter Linux). Vor dem allerersten Start sind hier die Pfade anzupassen für:

- JAVA
  ```
  REM: in der nächsten Zeile den Pfad auf JAVA anpassen!
  SET JAVA_EXE=%ProgramFiles(x86)%\<path>\java.exe
  ```
- FCalib2
  ```
  ECHO "%JAVA_EXE%" -jar "%APP_DIR%FCalib2-<version>.jar"
  "%JAVA_EXE%" "-Dlog.dir=%LOG_DIR%" -jar "%APP_DIR%FCalib2-↵
  <version>.jar"
  ```

Mit FCalib2 flashen

Nach dem Start von FCalib2 sind zunächst Einstellungen vorzunehmen:

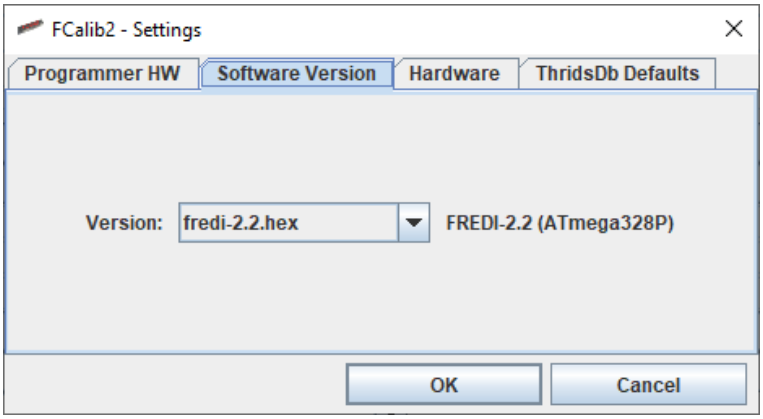
- Es ist (einmalig) das verwendete AVR-Programmiergerät samt Software einzustellen[7]:



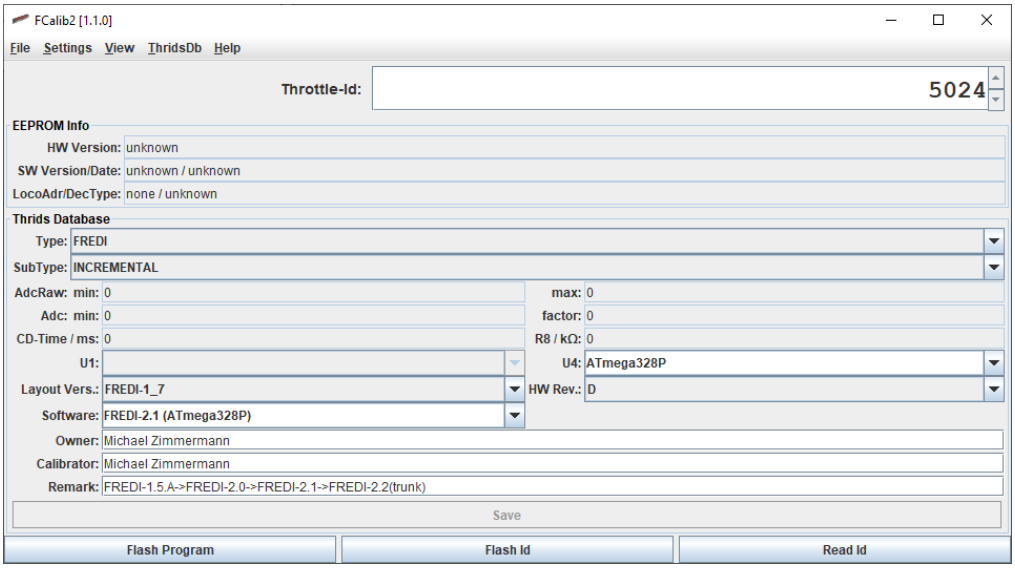
[6] AVR-Fuses sind die Hardwareeinstellungen direkt im Prozessor und vergleichbar mit den Konfigurationsbits bei einem PIC

[7] Angaben in diesem und zu den weiteren Screenshots dienen als Beispiel und sind immer den tatsächlichen Verhältnissen und Geräten anzupassen!

- Es ist die zu verwendende FREDI-Software aus der Drop-Down-Liste auszuwählen (aktuellste Version ist 2.2.2):



- Ausgehend vom Hauptbildschirm:



  - wird mit Read Id die Throttle-Id gelesen. Ist der Handregler bekannt, werden die zugehörigen Daten angezeigt.
    Weiterhin ist dies der erste und wichtige Schritt, ins besonders wenn der AVR-Prozessor getauscht werden soll (damit der neue Prozessor die ‚alte/bestehende' Id bekommen kann).
  - Kann mit Flash Program die Software auf den AVR geschrieben werden. Zusätzlich zum Programm wird auch die Throttle-Id und die Fuses gesetzt.
    Daher ist es wichtig, immer zuerst die Throttle-Id zu lesen oder bei einem neuen einzugeben. Neue oder geänderte FREDI-Daten werden mit Save in die Datenbank geschrieben
  - Mit Flash Id wird die eingegebene Throttle-Id geschrieben. Dies wird aber auch bereits mit Flash Program erledigt.

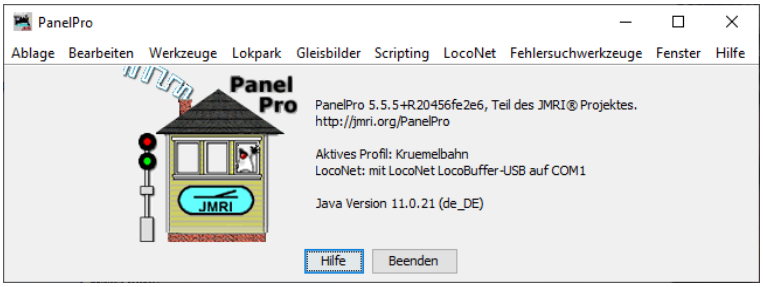
Nach dem Flashen ist der Selbsttest durchzuführen, erkennbar an den langsamen Lauflicht der LEDs. Das ist nun der Moment, alle Tasten, Schalter und Drehelemente zu betätigen. Der Nothalt muss als letzter Taster betätigt werden! Den erfolgreichen Selbsttest erkennt man an einer Geschwindigkeitsänderung des Lauflichts (von langsam zu schnell). Beim nächsten Anstecken des FREDI an das LocoNET®-Kabel geht dieser in seinen Grundzustand (rote LED leuchtet) und ist nun betriebsbereit.
*Hinweis: betätigt man den Nothalt **vor** dem Anschließen des FREDI an das LocoNET® und hält den Nothalt weiter für mehr als 10 Sekunden fest, geht der FREDI (erneut) in den Selbsttest.*

JMRI

JMRI ist ein vielseitiges und umfassendes Werkzeug[8] für alles, was rund um die Modellbahn und ins besonders der Ansteuerung von Zentralen und Handreglern. JMRI beinhaltet auch den wiThrottle-Server, der für die Anbindung der WLAN-FREDs benötigt wird, siehe auch hier:

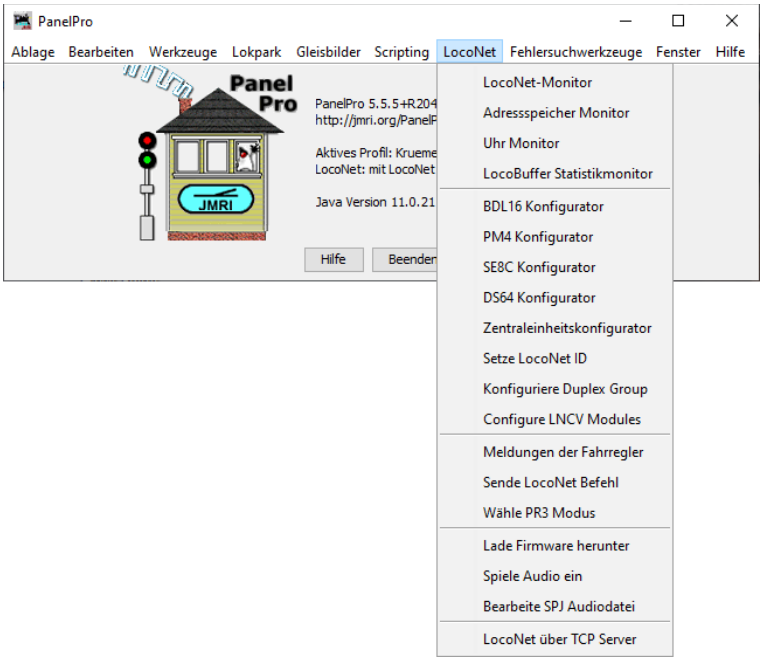
- Krümelbahn Info 3 - RaspBerry als wiThrottle-Server https://github.com/Kruemelbahn/Infoletter/blob/main/Krümelbahn%20Info%203%20-%20RaspBerry%20als%20wiThrottle-Server.pdf bzw. hier:
- Krümelbahn Info 18 – PiLocoBuffer https://github.com/Kruemelbahn/Infoletter/blob/main/Krümelbahn%20Info%2018%20–%20PiLocoBuffer.pdf

Nach der Installation von JMRI gibt es i.d.R. zwei Icons auf dem Desktop: DecoderPro® und PanelPro™. Der Unterschied besteht lediglich in den Startbildschirmen und den Anfangs verfügbaren Funktionen. In den kommenden Abschnitten wird immer von einem Start von PanelPro™ ausgegangen:



Mit JMRI flashen

Startpunkt ist für uns hier der Menüpunkt LocoNet:

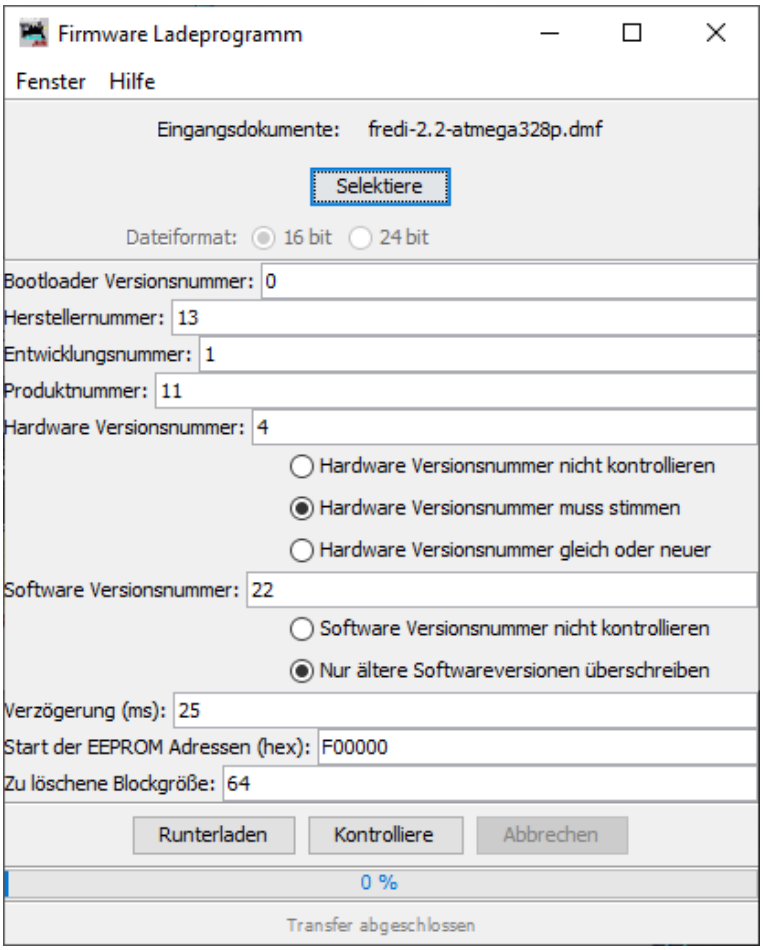


Neben dem LocoNet-Monitor (den wir hier nicht weiter betrachten) gibt es weiter unten den Menüpunkt Lade Firmware herunter. „Herunterladen“ ist hier m.E. eigentlich nicht der richtige Begriff: immerhin geht es darum, eine Software auf ein Gerät zu bringen – also eher ein „Hochladen“.

[8] JMRI wird mit seinen Möglichkeiten erläutert in: Krümelbahn Info 11 - JMRI - Universalwerkzeug für die Modellbahn (https://github.com/Kruemelbahn/Infoletter/blob/main/Krümelbahn%20Info%2011%20-%20JMRI%20-%20Universalwerkzeug%20für%20die%20Modellbahn.pdf)

Dieses „Hochladen“ benötigt kein separates externes Programmiergerät, sondern erfolgt direkt über das LocoNET®. Dazu muss der zu FREDI, auf den die Software übertragen werden soll, einzig allein am LocoNET® angeschlossen sein.

Mit dem Menüpunkt Lade Firmware herunter öffnet sich ein Einstellungs- und Bediendialog:



Die Bedienung ist einfach:

- Über die Schaltfläche Selektiere wird die dmf-Datei[9] mit den zu übertragenden Daten ausgewählt.
- Im Zweifel wählt man anstelle von Nur ältere Softwareversionen überschreiben den Eintrag Software Versionsnummer nicht kontrollieren
- Mit der Schaltfläche Runterladen wird dann die Software auf den AVR übertragen, dabei blinken die LEDs am FREDI. Ist die Übertragung abgeschlossen, steht der Laufbalken bei 100% und die Schaltfläche Runterladen wird wieder freigegeben.

Diese Vorgehensweise bedingt einige nachgelagerte Bedienhandlungen:

- Zum einen wird keine Throttle-Id gesetzt, zum anderen werden keinen Fuses gesetzt. Diese Methode eignet sich also nur für ein Update des FREDI, das erstmalige Flashen erfolgt also immer mit FCalib2.

[9] dmf = **D**igitrax **M**angled **F**irmware, eine Datei mit Einstellungsdaten und der zu übertragenden Software. Im Verzeichnis von FCalib2 sind viele Dateien im dmf-Format hinterlegt.

- Der FREDI geht jetzt nach der Datenübertragung nicht selbständig in den Testmodus, da der EEPROM-Inhalt nicht geändert wurde. Nachfolgende Bedienhandlung bringt den FREDI in den erforderlichen Selbsttest:
  *Man betätigt den Nothalt **vor** dem Anschließen des FREDI an das LocoNET® und hält den Nothalt weiter für mehr als 10 Sekunden fest - der FREDI geht in den Selbsttest.*
  Anschließend wird der Selbsttest wie oben beschrieben durchgeführt.

Weiterführende Links

Zu den verschiedenen Themen gibt es auch Dokumente, die meisten sind aber nur im FREMO erhältlich. Wem eines dieser Dokumente nützlich erscheint, kann mich gerne ansprechen…

Bedeutung der Konfigurationsvariablen innerhalb eines FREDI:
- FrediSvUsage.pdf (https://groups.io/g/fremodcc/files/FREDI/FrediSvUsage.pdf)

Handhabung der Konfigurationsvariablen eines FREDI mit Hilfe von JMRI:
- FREDI-JMRI-LNSV-0.4.pdf (https://groups.io/g/fremodcc/files/FREDI/FREDI-JMRI-LNSV-0.4.pdf)

Auslesen der FREDI-Id mit JMRI:
- Auslesen_FREDI-ID_mit_JMRI.pdf

FREDI Update mit FCALIB2:
- FREDI_Update_WS_2021_03.pdf

JMRI - Java Model Railroad Interface (eine Einführung):
- JMRI_WS_2023_04 forum.pdf

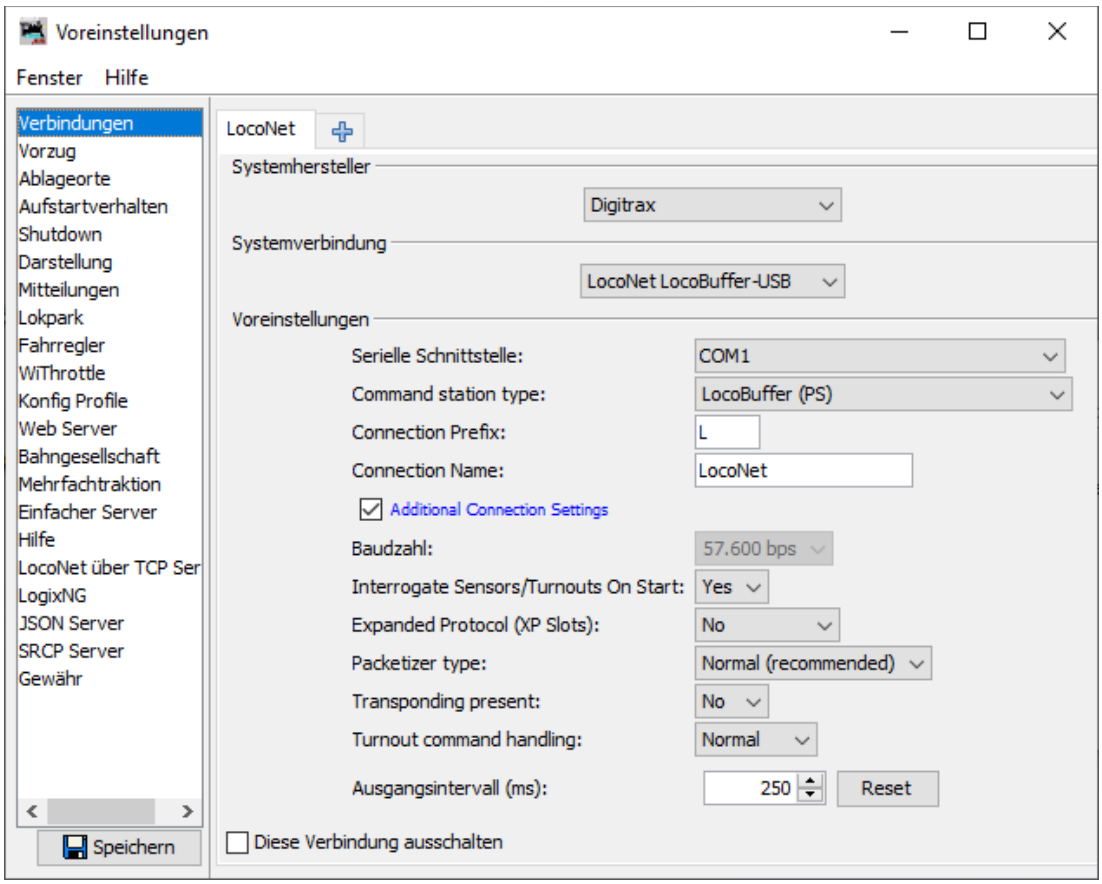
Wie repariere ich einen FREDI:
- Krümelbahn Info 17 - FREDi - Gedanken zu seiner Reparatur (https://github.com/Kruemelbahn/Infoletter/blob/main/Krümelbahn%20Info%2017%20-%20FREDi%20-%20Gedanken%20zu%20seiner%20Reparatur.pdf)

Versionsgeschichte

21.10.2023	Initiale Erstellung
28.08.2024	redaktionelle Änderung
21.09.2024	Links korrigiert

Anhang A: JMRI – LocoNET®-Einstellungen

Im Softwarepaket JMRI sind nachfolgende Einstellungen[10] zur Anbindung an einen LocoBuffer erforderlich, ausgehend vom Menüpunkt Bearbeiten – Voreinstellungen…:



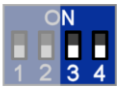
[10] Angaben in diesem Screenshot dienen als Beispiel und sind den tatsächlichen Verhältnissen anzupassen!

Anhang B: AVR-Programmiergerät

Zum Einsatz kommt der DIAMAX-Prog S2 von https://www.diamex.de/dxshop/USB-ISP-Programmer-fuer-Atmel-AVR. Für die Verwendung werden keine zusätzlichen Treiber benötigt.

Der DIAMAX-Prog S2 ist ST500-kompatibel und wird auch von FCalib2 unterstützt.

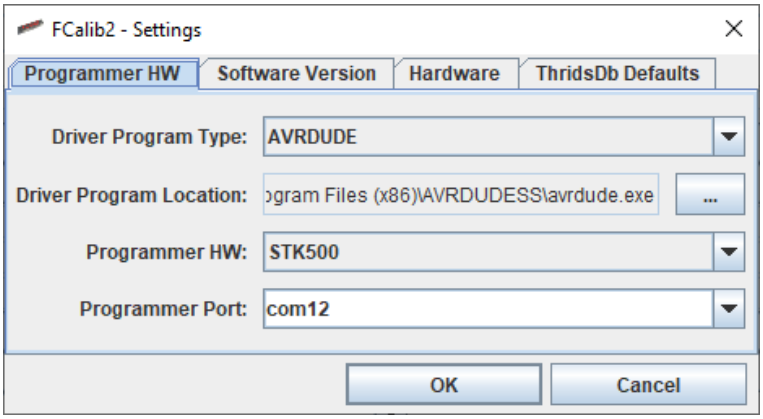
Diamex Prog-S2 *(weiß = Schalterstellung)*

	**1 = OFF (3,3V), 2 = ON (externe Spannung ein)** Höhe der Spannung auf den Datenleitungen und extern = 3,3 Volt Eine externe Schaltung bzw. ein angeschlossener Controller kann vom PROG-S2 mit Strom versorgt werden.
	**3 = OFF, 4 = OFF** ***Programmer für alle Atmel-AVR-Controller mit ISP-Schnittstelle*** Die Programmierung dieser Controller geschieht über die ISP-Schnittstelle, diese ist bei den meisten AVR-Controllern gleichbedeutend mit den Pins für den SPI-Bus (Achtung! Es gibt einige Controller, die separate PDI/PDO-Pins besitzen). Zusätzlich zu SCK, MISO, MOSI wird noch die Reset-Leitung benötigt. PROG-S2 emuliert einen STK500-Programmer und ist hierdurch kompatibel zu AVR/ATMEL-Studio und AVRDUDE. Die Programmierspannung kann je nach angeschlossenem AVR-Controller auf 3,3V oder 5V eingestellt werden.

Die Verbindung zu einem Handregler erfolgt über ein 6-poliges Flachbandkabel, Verdrahtung 1:1 (Standardkabel für ICSP) – wenn denn die ICSP-Buchse (AVR ISP6) auf der Handreglerplatine aufgelötet ist…

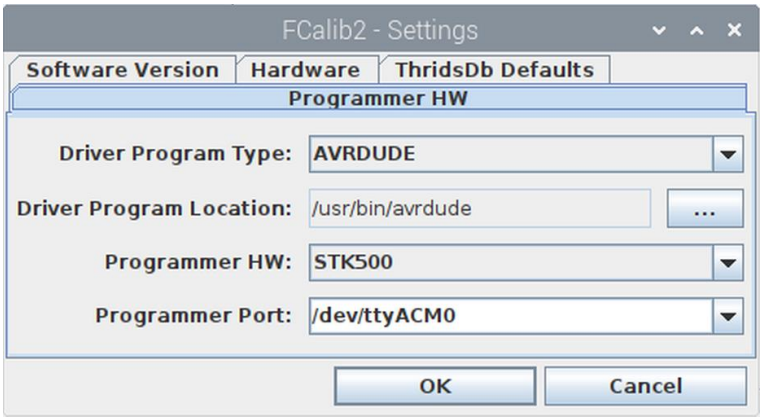
Programmer-Einstellungen für FCalib2 unter Windows

Unter Windows ist das Gerät über einen **COM**-Port erreichbar:



Programmer-Einstellungen für FCalib2 auf dem RaspBerry Pi unter Linux

Unter Linux ist das Gerät über **/dev/ttyACM0** erreichbar:



Anhang C: Software unter Linux installieren

Java

```
sudo apt update
sudo apt install openjdk-11-jdk
java -version                    (=> 11.0.18)
```

FCalib2

FCalib2 ist ein FREMO-Programm:

- Download der aktuellen FCalib2-Version von https://sourceforge.net/projects/fremodcc/files/FCalib2/1.1.1/
- Kopieren aller Dateien in das Verzeichnis /home/pi/FCalib2
- Sofern bereits vorhanden: die aktuelle ID-Datenbank in das Verzeichnis /home/pi/FCalib2/db kopieren
- Ausgehend vom Verzeichnis /home/pi/ erfolgt der Programmstart vom Terminal aus mit

```
cd FCalib2
bash start-fcalib2.sh
```

AvrDude

Zum Flashen der Software auf den AVR benötigt FCalib2 ein entsprechendes Programm. Zum Einsatz kommt hier AvrDude:

```
sudo apt-get install avrdude
```

Nach der Installation befindet sich die Datei `avrdude.conf` im Verzeichnis /etc/.

Für AvrDude ist auch eine GUI verfügbar: http://avr8-burn-o-mat.aaabbb.de/ (wenn man auch einmal ohne FCalib2 arbeiten will…):

- Download des Archives von http://avr8-burn-o-mat.aaabbb.de/AVR8_Burn-O-Mat_2_1_2.zip
- Archiv entpacken nach: /home/pi/burnavr/
- Ausgehend vom Verzeichnis /home/pi/ erfolgt der Programmstart vom Terminal aus mit

```
cd burnavr
bash start.sh
```

JMRI

→ zu Hinweisen, welche JAVA-Version für welche JMRI-Version erforderlich ist: ←
→ siehe https://www.jmri.org/←

- Download der aktuellen JMRI-Linux-Version von https://www.jmri.org/ durch klicken auf die tgz-Datei
- Archiv entpacken nach: /home/pi/