

Heizungs-Datenerfassung
mit
Raspberry Pi ®
und
Laptop/PC

Inhaltsverzeichnis

Kurzbeschreibung.....	3
1Hardware.....	4
1.1Adapter für Raspberry Pi®.....	4
1.1.1Schaltplan und Stückliste.....	5
1.1.2Funktionsbeschreibung.....	6
1.1.3Inbetriebnahme.....	7
1.1.4Realisierungsbeispiel.....	8
1.2USB-Adapter für PCs und Laptops (HT3-Microadapter).....	10
1.2.1Schaltplan und Stückliste.....	11
2Software.....	12
2.1Verzeichnis-Struktur.....	12
2.2Konfiguration.....	13
2.3Datenbanken.....	18
2.3.1Datenbank 'SQLite'.....	18
2.3.2Datenbank 'rrdtool'.....	21
2.4Applikationen.....	22
2.4.1HT3-Analyser.....	22
2.4.2HT3_Systemstatus.....	24
2.4.3HT3_Logger.....	25
3Installation.....	25
3.1Betriebssystem.....	25
3.2Applikation.....	26
4HT3 Applikation im Betrieb.....	29
5Weiterführende Literatur und URL's.....	31

Kurzbeschreibung

Diese Anleitung beschreibt die Hardware- und Software-Anteile einer Heizungs-Datenerfassung.

Die Adaption beschränkt sich nur auf die Protokolle von Heizungsanlagen mit Heatronic3©-Bus der Firma Junkers.

Es werden die Heizungs- und Solaranlagen Informationen grafisch dargestellt und für die aktuelle und spätere Auswertung gespeichert.

Die Erfassung und die weitere Nachverarbeitung beeinflussen nicht den Heizungsbetrieb. Eine Regelung / Steuerung der Heizung ist nicht realisiert.

Die Hardware ist vorrangig für das Board 'Raspberry Pi ®', kann jedoch ohne größeren Aufwand an andere Hardware adaptiert werden. Es ist nur ein UART erforderlich, der die Datenerfassung durchführt.

Als Alternative ist ein USB-Adapter beschrieben, der die Erfassung der Heizungsdaten durch jeden PC oder Laptop ermöglicht.

Es wird die Programmiersprache 'Python' verwendet. Diese realisiert einen HT3-Telegrammanalysator mit zugehöriger grafischen Anzeige.

Durch Konfigurationsänderungen lässt sich die Software zu einer reinen grafischen Statusanzeige oder zu einem Datenlogger (ohne Grafikausgabe) ändern.

Die Daten werden nach Erfassung eines gültigen Telegramms in eine SQLite – Datenbank geschrieben. Im Rhythmus von 60 Sekunden (default) werden diese SQL-Daten in eine weitere Datenbank (rrdtool) übertragen. Mit diesem Tool wird auch die grafische Darstellung realisiert.

Details sind in den folgenden Kapiteln zu finden.

Gewährleistung, Haftung und Ansprüche durch Fehlfunktionen an Heizung oder Adaption sind hiermit ausdrücklich ausgeschlossen.

Der Nachbau und die Inbetriebnahme der Adaption ist auf eigene Gefahr und die Beschreibung und die Software erheben nicht den Anspruch auf Vollständigkeit.

Eine Änderung an Software-Modulen und Hardware-Beschreibungen ist jederzeit ohne Vorankündigung möglich.

Die hier verwendeten Handels- und Gebrauchsnamen können auch ohne besondere Kennzeichnung Marken sein und somit den gesetzlichen Bestimmungen unterliegen.

Anregungen, Fragen und mehr an:
Norbert Scharf Email: junky-zs@gmx.de

1 Hardware

Die Hardware besteht aus einem Adapter, der die Pegel-Wandlung und Anpassung der Heatronic3-Bussignale in einen seriellen Datenstrom für die UART-Erfassung durchführt. Der Adapter gewährleistet eine galvanische Trennung zwischen der Heizungsanlage und der Datenerfassung.

Die Daten des Heatronic3-Bus werden mit folgenden Parametern übertragen:

Baudrate	: 9600
Datenbits	: 8
Parity	: keine
Stopbit	: 1
Kurzform	: 9600, 8N1

Die Realisierung ist im folgenden für den Computer 'Raspberry Pi®' und für die Schnittstelle USB (Hardware-neutral) beschrieben.

1.1 Adapter für Raspberry Pi®

Der Adapter ist in der Größe und in der Anschlussbelegung für den Einbau in einen Raspberry Pi® ausgelegt.

Für den Bus Anschluss und für die Betriebsstatus-LED's sind Durchführungen im Gehäuse erforderlich. Die mechanische Befestigung des Adapters wird durch die zweireihige 13-polige Buchsenleiste auf dem Adapter realisiert.

Der Aufbau des Adapters kann auf einer Lochraster-Platine durchgeführt werden. SMD-Bauteile sind nicht erforderlich, da genügend Platz auf dem Adapter vorhanden ist.

Die Spannungsversorgung auf der CPU-Seite des Adapters ist durch die Betriebsspannung des Raspberry Pi realisiert. Die Stromaufnahme beschränkt sich dabei auf wenige Milliampere und wird einzig durch den Pullup-Widerstand am UART-Rx Eingang und LED / Vorwiderstand am UART-TX Ausgang bestimmt.

Die Spannungsversorgung des Adapters auf der Heatronic3-Bus Seite wird durch die Heizung realisiert und ist für 15Volt und 3mA ausgelegt.

1.1.1 Schaltplan und Stückliste

Im folgenden sind der Schaltplan und die zugehörige Stückliste aufgeführt:

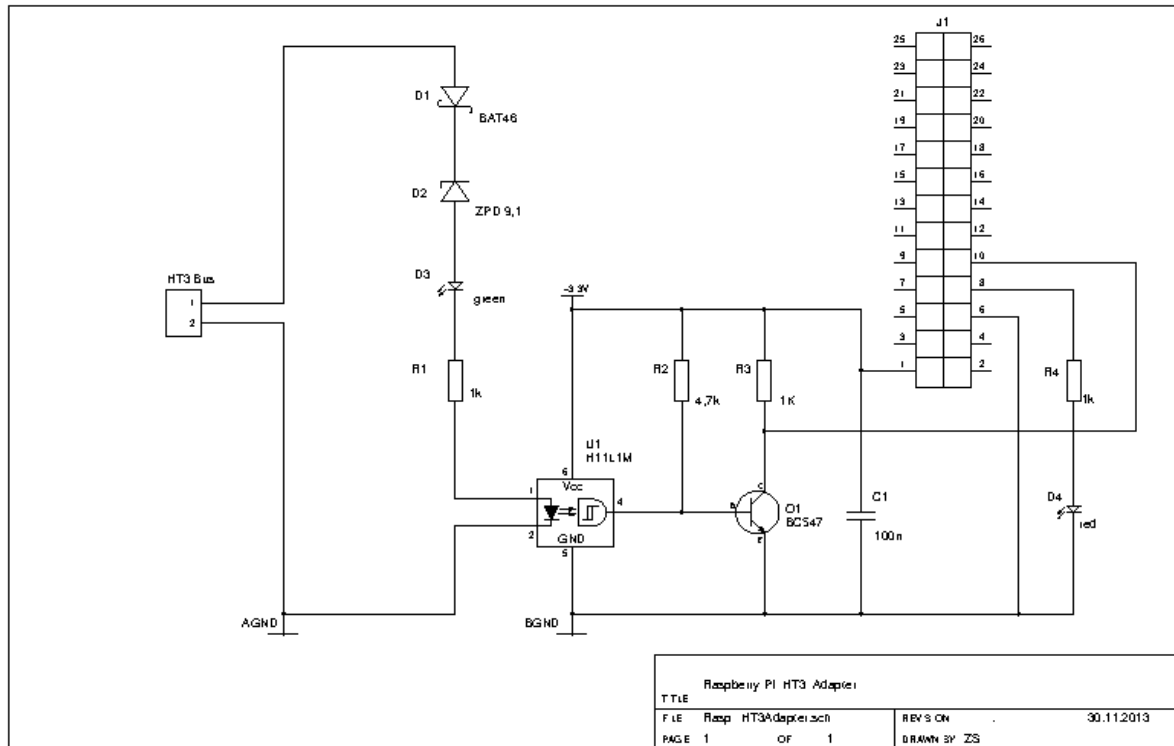


Abbildung 1: Schaltplan HT3 Adapter für Raspberry Pi

Name	Bezeichnung / Type	Anzahl	Bemerkung
U1	H11L1M Optokoppler oder PC900V	1	Versionen H11L2M und H11L3M sind nicht geeignet, da diese erst bei einem höheren LED-Strom schalten aber der Adapter möglichst geringe BUS-Lastung erzeugen soll.
D1	BAT46 oder 1N4148	1	
D2	ZPD9.1 oder BZX55/C9V1	1	Zenerdiode, Spannung 9,1 Volt
D3/4	LED (grün/rot), kleine Ausführung	2	
Q1	BC547 oder ähnlich	1	
R1/R3/R4	1 kOhm Widerstand	3	
R2	4,7 kOhm Widerstand	1	
C1	100 nF Keramik Kondensator	1	
HT3 Port	Printklemmenblock	1	Conrad, Bestell-Nr.: 731986
J1	Buchsenleiste RM2,54	2*13	Conrad, Bestell-Nr.: 733779 oder 733755
-	Lochraster-Platine	1	52*33 mm -> 20*13 Lötungen

Tabelle 1: HT3 Adapterstückliste

1.1.2 Funktionsbeschreibung

Zur galvanische Trennung wird der Optokoppler H11L1M von Fairchild (oder PC900V) zwischen HT3-Bus und Raspberry Pi verwendet.

Dieser hat ein Schmitt-Trigger Ausgangssignal mit Hysterese und ist schnell genug für das Signalprotokoll mit: 9600 Baud.

Der Optokopplers schaltet bei Eingangsströmen größer 1.6 mA den Ausgang auf logisch Null. Unterhalb von 1.0 mA ist das Ausgangssignal auf logisch Eins. Durch diese Hysterese werden Störungen auf dem Eingangssignal unterdrückt.

Der Transistor am Ausgang des Optokopplers invertiert das Signal, sodass der UART einer CPU (z.B. Raspberry Pi) dieses korrekt empfangen kann.

Die Diode D1 dient als Verpolschutz, auf eine Eingangs-Brückenschaltung wurde hier verzichtet.

Diode D2 passt das Signal an den Optokoppler an.

Diode D3 dient als Signalindikator für den richtigen Anschluss (richtige Polarität) des HT3-Bus.

Die Diode D4 ist am TX-UART Ausgang des Raspberry Pi-Ports angeschlossen und wird hier nur als Betriebsindikator genutzt.

1.1.3 Inbetriebnahme

1. Eingangs-Widerstand am HT3-Anschluss (Printklemmenblock) messen.
Es darf kein Kurzschluss vorhanden und der Widerstand muss größer als 1kOhm sein.
2. Galvanische Trennung zwischen HT3-Bus und uC Anschluss messen.
Der Widerstand zwischen HT3-Bus und Ausgangsseite des Optokopplers (U1) muss sehr groß sein ($>> 1 \text{ MOhm}$). Es darf keine Verbindung zwischen HT3-Bus und UART-RX Eingang vorhanden sein.
3. Funktionsprüfung ohne Heizungs-Anschluss mit Prüfspannung.
Eine externe Spannung von ca. 14-15 Volt an den HT3-Bus Printklemmenblock anschliessen. Die LED D3 muss aufleuchten, wenn die Polarität der Eingangsspannung korrekt ist.
Die Stromaufnahme ist dabei größer als 1.6 mA jedoch weniger als 5 mA.

Den gleichen Test mit reduzierter Eingangsspannung von weniger als 11 Volt durchführen.
Die LED D3 darf nicht oder nur wenig leuchten. Die Stromaufnahme muss weniger als 1 mA sein. Falls dies nicht passt, den Einbau der Zenerdiode D2 prüfen.
4. Anschluss an die Heizungsanlage.
Vor Anschluss des Adapters die Heizung ausschalten! Die Hinweise des Herstellers beachten. Dabei Leitungslängen und Kabelquerschnitte berücksichtigen. Die Klemmen für den HT3-Bus sind in der Regel mit 'B' bezeichnet.

1.1.4 Realisierungsbeispiel

In den folgenden Bildern ist die Realisierung des HT3-Adapters mit dem Raspberry Pi gezeigt:

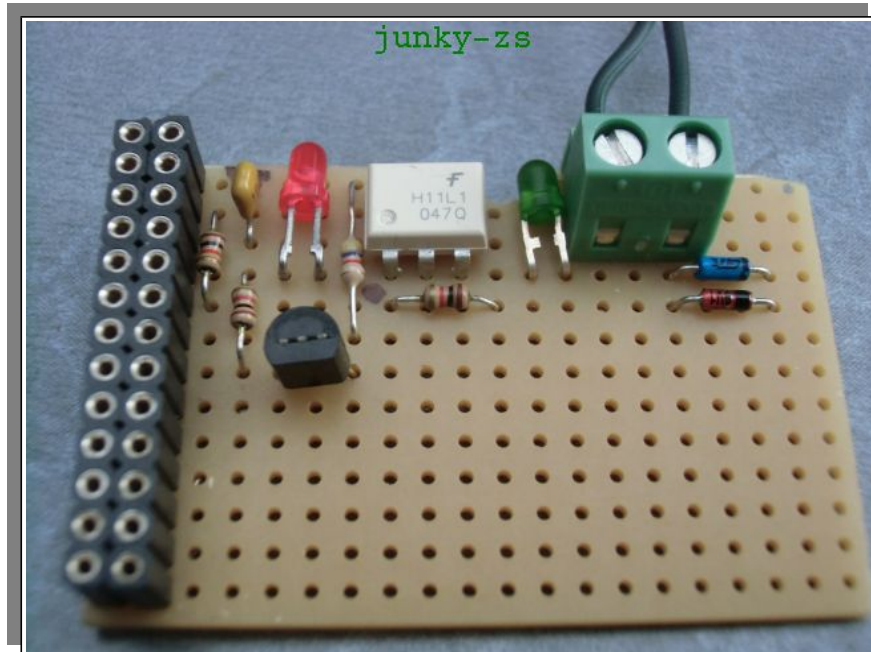


Abbildung 2: HT3 Adapter für Raspberry Pi

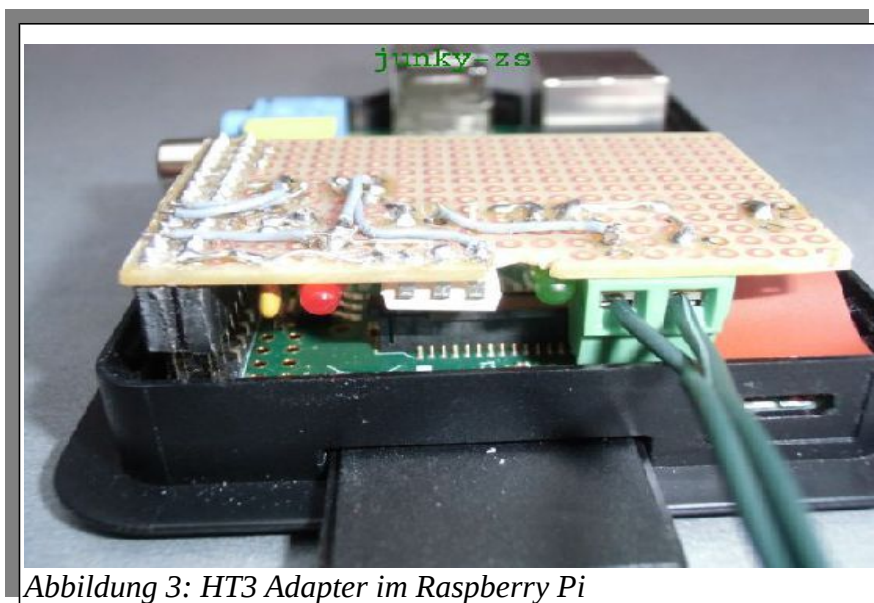


Abbildung 3: HT3 Adapter im Raspberry Pi

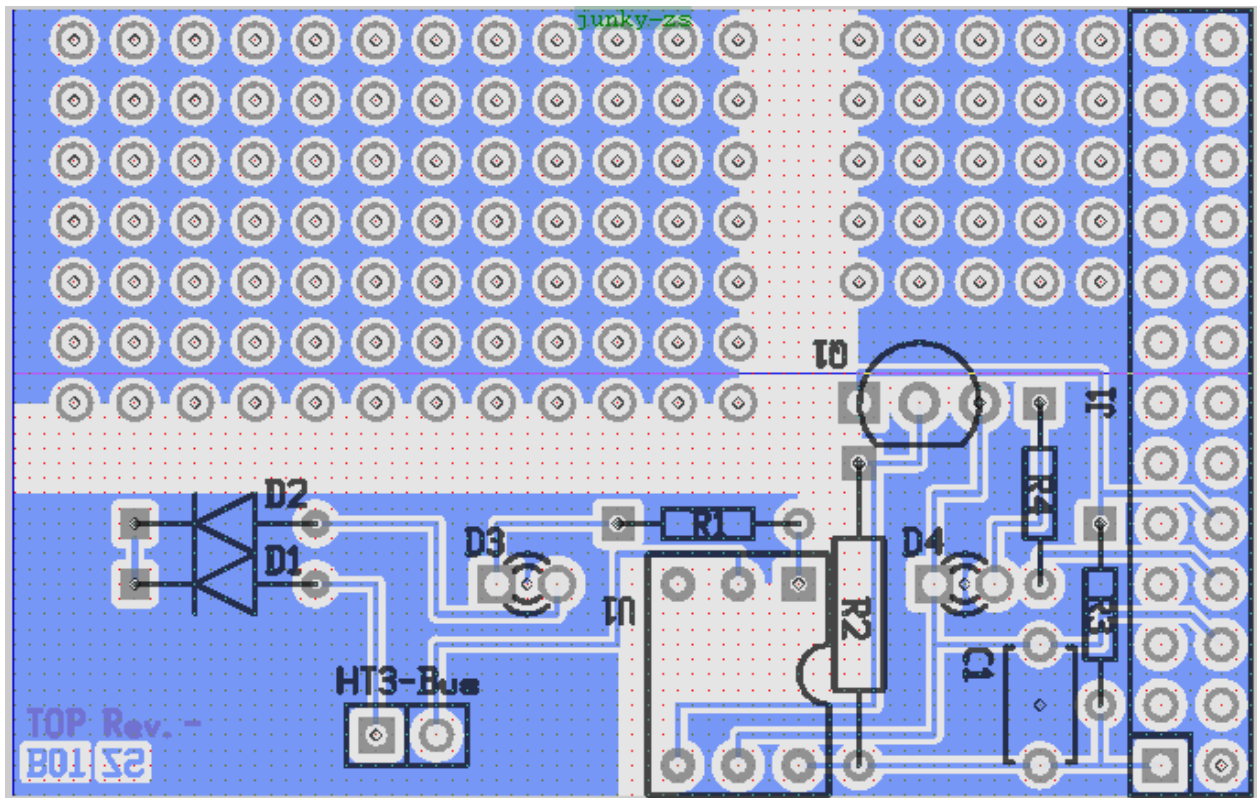


Abbildung 4: Adapter Layout

Bei der Bestückung ist zu beachten, dass die Bauteile so platziert werden wie hier gezeigt. Andernfalls passt der HT3 Adapter nicht auf den Raspberry Pi.

Auch muss der Transistor Q1 tief genug eingelötet werden, damit er keinen Kontakt mit dem darunter liegenden Spannungsregler hat. Tief genug meint hier, nicht höher als die Buchsenleisten-Oberkante.

Auch sollte der Abstand zwischen HT3-Bus und Raspberry Pi Ports möglichst groß sein um eine gewisse Spannungsfestigkeit zu gewährleisten. Daher auch die getrennten Vollflächen zwischen dem Ein- und Ausgang des Optokopplers U1. Die Vollflächen sollten keinesfalls verbunden werden, dann besser weglassen.

Die zusätzlichen Lötungen lassen Platz für Erweiterungen.

1.2 USB-Adapter für PCs und Laptops (HT3-Microadapter)

Die Bilder zeigen wie der HT3-Microadapter aussehen kann. Die Anpassung zum HT3-Bus ist als eigenständige Platine ausgelegt. Diese wird unter den USB/RS232 Wandler gesteckt.

Als Gehäuse habe ich eine Streichholzschachtel gewählt. Ich dachte mir, das es eine zündende Idee ist es einmal so zu machen. Geräteschutzklasse: IP null

(bitte die Aufschrift 'VON KINDERN FERNHALTEN' nicht beachten, ist alles ist für den Nachbau geeignet)

Hinweis: Die rote und die orangene LED dienen nur zu Testzwecken und sind für den HT3-Datenempfang nicht erforderlich.



1.2.1 Schaltplan und Stückliste

Im folgenden sind der Schaltplan und die zugehörige Stückliste aufgeführt:

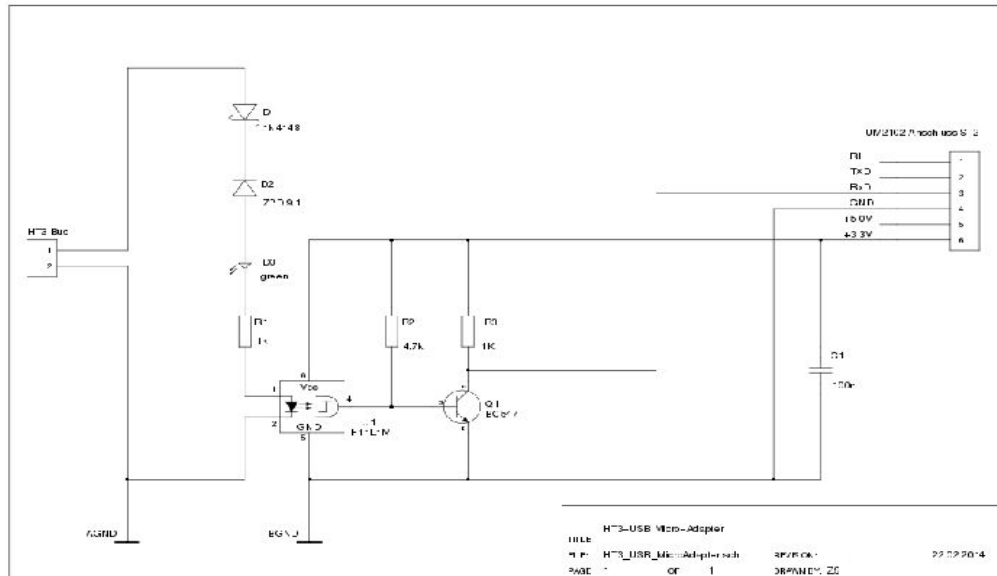


Abbildung 5: Schaltplan HT3 Microadapter (RS232->USB)

Name	Bezeichnung / Type	Anzahl	Bemerkung
--	Bauteile siehe Stückliste für den HT3-Adapter (RaspberryPi)	1 Satz	Siehe Tabelle 1: HT3 Adapterstückliste
UM2102	Mini USB-Modul Bausatz	1	ELV: Bestell-Nr.: 68-91859
ST1-ST3	Stiftleiste RM 2.54 mm 20-polig	1	Conrad: Bestell-Nr.: 741105-62
BU1-BU3	Buchsenleiste 2.54 mm 20-polig (für das Mini USB-Modul)	1	Conrad: Bestell-Nr.: 733755-62

Tabelle 2: Stückliste HT3-Microadapter

2 Software

Die Software ist in Python geschrieben. Der Grund ist die leichtere Portierbarkeit auf verschiedene Betriebssysteme. Es wird die Version 3.x (Python3) verwendet, Versionen darunter werden nicht unterstützt.

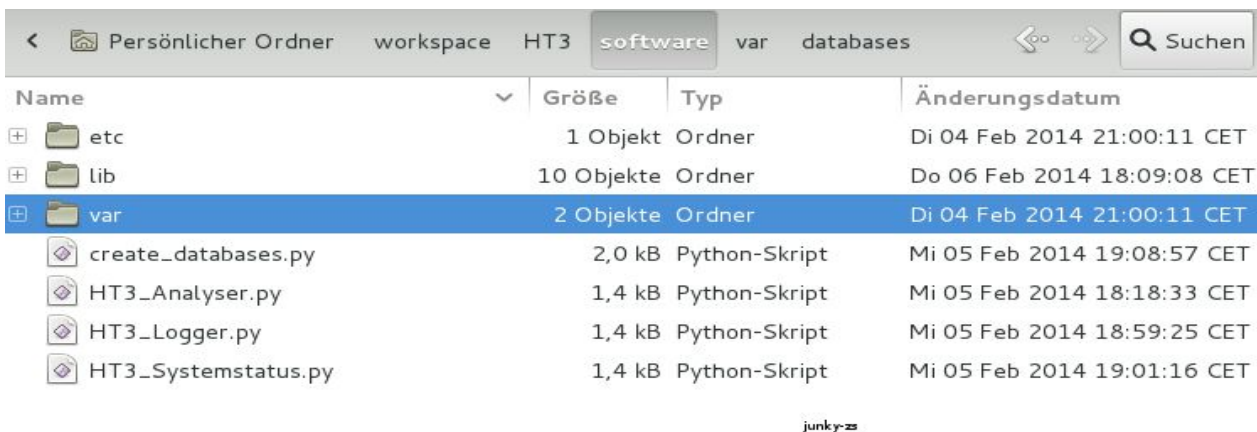
Für die Schnittstelle zur rrdtool-Datenbank wird die Programmiersprache 'Perl' genutzt, da die Python3-Module zur Zeit noch nicht verfügbar sind (Linux-Debian 'Wheezy') .

Durch Konfigurationsänderungen lässt sich die rrdtool-Datenbank abschalten und der Betrieb nur mit der SQL-Datenbank durchführen. Allerdings entfällt dann auch die grafische Ausgabe der Daten.

Die Konfigurationsdaten sind in XML-Dateien abgelegt und dienen u.A. zur Erzeugung der Datenstrukturen der 'SQLite'- und 'rrdtool'-Datenbanken.

2.1 Verzeichnis-Struktur

Die Verzeichnis-Struktur sieht wie folgt aus:



Name	Größe	Typ	Änderungsdatum
etc	1 Objekt	Ordner	Di 04 Feb 2014 21:00:11 CET
lib	10 Objekte	Ordner	Do 06 Feb 2014 18:09:08 CET
var	2 Objekte	Ordner	Di 04 Feb 2014 21:00:11 CET
create_databases.py	2,0 kB	Python-Skript	Mi 05 Feb 2014 19:08:57 CET
HT3_Analyser.py	1,4 kB	Python-Skript	Mi 05 Feb 2014 18:18:33 CET
HT3_Logger.py	1,4 kB	Python-Skript	Mi 05 Feb 2014 18:59:25 CET
HT3_Systemstatus.py	1,4 kB	Python-Skript	Mi 05 Feb 2014 19:01:16 CET

Abbildung 6: Datei-Verzeichnisstruktur

Modulname	Funktion	Bemerkung
create_databases.py	Erzeugt die Datenbank SQLite und rrdtool falls diese noch nicht vorhanden sind. Genutzt wird die Konfiguration unter ./etc/config.	Vorhandene Datenbanken werden nicht überschrieben.
HT3_Analyser.py	HT3 Bus Analyser mit grafischer Datenausgabe in Plain-Text und Hexadezimal mit zugehörigen Protokoll-Beschreibungen.	Daten werden in die vorhandenen Datenbanken geschrieben.
HT3_Logger.py	HT3 Logger schreibt die erfassten Daten in die Datenbanken, es werden diese nicht direkt grafisch angezeigt.	Der Logger dient zur Erfassung der Heizungsdaten ohne grafischer Datenausgabe. Daten werden in die vorhandenen Datenbanken geschrieben.
HT3_Systemstatus.py	HT3 Systemstatus mit grafischer Datenausgabe in Plain-Text und Schreiben der Daten in die Datenbanken.	Daten werden in die vorhandenen Datenbanken geschrieben.

Tabelle 3: Software-Modulinformationen

2.2 Konfiguration

Im Verzeichnis 'etc/config' ist die Konfiguration zur Erzeugung und Nutzung der Datenbanken abgelegt.

Die XML-Datei 'HT3_db_cfg.xml' enthält alle Details für den 'Normalbetrieb'.

Die XML-Dateien unter dem '4test'-Verzeichnis sind für den Testbetrieb der Python-Module vorgesehen.



Abbildung 7: Konfigurations-Verzeichnis

Die Einträge in der HT3_db_cfg.xml Datei werden für die Aktivierung (rrdtool), Namensgebung der Tabellen und Tabellen-Spalten (SQLite) und Startzeiten / Schrittweiten (rrdtool) benutzt.

Abbildung 8: Konfigurationsfile-Inhalt

```
....<dbname_sqlite>./var/databases/HT3_db.sqlite</dbname_sqlite>
....<sql-db>
.....<enable>on</enable>
....</sql-db>
....<!-- rrdtool-database -->
....<dbname_rrd>./var/databases/HT3_db_rrd</dbname_rrd>
....<!-- 'dbname_rrd' without suffix, is used only as leading path & name
.....and to create rrdtool db-files for any 'systempart' with there
.....own name and suffix
....-->
....<rrdtool-db>
.....<enable>on</enable>
.....<step_seconds>60</step_seconds>
.....<starttime_utc>1344000000</starttime_utc>
....</rrdtool-db>

....<!-- global configuration-values -->
....<anzahl_heizkreise>1</anzahl_heizkreise>

....<systempart name="heizgeraet">
.....<shortname name="HG"/>
.....<hardwaretype>CSW</hardwaretype>....<!-- Wert wird nur in GUI angezeigt
.....<logitem name="T_vorlauf_soll">
.....<datatype>INT</datatype>
.....<datause>GAUGE</datause>
.....<maxvalue>100</maxvalue>
.....<default>0</default>
.....<unit>Grad</unit>
.....<displayname>T-Soll (Regelung)</displayname>
.....</logitem>
.....<logitem name="T_vorlauf_ist">
.....<datatype>REAL</datatype>
.....<datause>GAUGE</datause>
.....<maxvalue>100.0</maxvalue>
.....<default>0.0</default>
.....<unit>Grad</unit>
.....<displayname>T-Ist (Vorlauf)</displayname>
.....</logitem>
```

Auszug aus dem Konfigurations-File.

Die Bedeutung der einzelnen Parameter ist auf den folgenden Seiten beschrieben.

Parameter	Werte	Funktion
<dbname_sqlite>	Pfad und Name wählbar. Das Verzeichnis muss vorhanden sein.	Pfad und Name der SQL-Datenbank. Die Datenbank wird erzeugt, sofern das Verzeichnis vorhanden ist. Das Verzeichnis wird <u>nicht</u> angelegt.
<sql-db> <enable>	on oder 1 ->> Enable off oder 0 ->> Disable (Gross/Kleinschreibung erlaubt)	Aktiviert die Datenbank 'sqlite'. Es wird die Datenbank erzeugt, falls diese noch nicht vorhanden ist. Jeder andere Wert als 'on' /1 deaktiviert die Datenbank.
<dbname_rrdtool>	Pfad und Name wählbar. Das Verzeichnis muss vorhanden sein.	Pfad und Name der rrdtool-Datenbank. Die Datenbank wird erzeugt, sofern das Verzeichnis vorhanden ist. Das Verzeichnis wird <u>nicht</u> angelegt.
<rrdtool-db> <enable>	on oder 1 ->> Enable off oder 0 ->> Disable (Gross/Kleinschreibung erlaubt)	Aktiviert die Datenbank 'rrdtool'. Es wird die Datenbank erzeugt, falls diese noch nicht vorhanden ist. Jeder andere Wert als 'on' /1 deaktiviert die Datenbank.
<rrdtool-db> <step_seconds>	Default: 60 Sekunden	Der Wert bestimmt das Aktualisierungs-Intervall der rrdtool-Datenbank. Mit diesem Intervall werden die Daten der SQLite-Datenbank in die rrdtool-Datenbank eingetragen. Kleinere Werte als 60 Sekunden sind nicht möglich (und auch nicht sinnvoll).
<rrdtool-db> <starttime_utc>	Default: 1344000000 (entspricht: 13:30:00 03.08.2012)	Frühestmöglicher Zeitstempel für rrdtool-Datenbankeinträge.
<anzahl_heizkreise>	1...4	Gibt die Anzahl der Heizkreise des Systems an. Maximale Wert ist z.Zeit := 4 (0 ist nicht erlaubt).
<systempart name="">	heizgeraet heizkreis1 heizkreis2 heizkreis3 heizkreis4 warmwasser solar sysdatetime	Heizsystemanteile, für die Daten auf dem HT3-Bus gesendet werden. Die Namensgebung entspricht dabei denen der FWxyz – Reglerserie. Mit diesem Namen werden die Tabellen in der SQL-Datenbank erzeugt. Bei der rrdtool-Datenbank wird dieser Name auch als File-Namenserweiterung verwendet. (Hinweis: Die 'sysdatetime'-Daten werden zwar als Systemzeit angezeigt und in die SQL-Datenbank eingetragen, jedoch nicht in die 'rrdtool'-Datenbank übernommen)
<systempart ...> <shortname name="">	HG HK1 HK2 HK3 HK4 WW SO DT	Kurzname des 'systempart'-Namen ->> 'Nickname'. Dieser wird in der Software für das interne Datenhandling verwendet. Es werden nur maximal die ersten drei Charakter benutzt, die somit eindeutig sein müssen. Groß/Kleinschreibung ist erlaubt.
<systempart ...> <hardwaretype>	CSW, KUB, ISM1, ISM2, IPM1, IPM2 etc.	Type der Hardware, welcher diese Systempart-Daten bereitstellt. Dieser Type-Name wird in der GUI im Systempartteil dargestellt und kann auch leer bleiben. Aus diesem Namen werden <u>keine</u> Systemkonfigurationen abgeleitet.

Parameter	Werte	Funktion
<systempart ...> <logitem name="">	T_vorlauf_soll T_vorlauf_ist T_ruecklauf hexdump	Mit diesen Namen wird die SQL-Datenbank (Columns) und die rrdtool Datenitems erzeugt. Innerhalb eines <systempart>-Bereichs muss dieser Name eindeutig sein. Im 'hexdump' wird das zugehörige Datentelegramm in Hexform abgespeichert. Eine nachträgliche Veränderung des Namens nach der Erzeugung und Nutzung der Datenbanken ist zu vermeiden und führt u.U. zu Datenverlust bzw. aufwendigen Anpassungen und Korrekturen.
<systempart name="heizkreis"> <unmixed>	Flag (True/False) wobei: x:= 1...4	Dieses Flag bestimmt, ob der Heizkreis keinen (True) oder einen Mischer (False) hat. Es wird nur die GUI-Anzeige damit gesteuert, die Erfassung wird dadurch nicht beeinflusst.
<systempart name="heizkreis"> <buscodierung>	Wert je nach Heizkreis und Hersteller, Standardbereich 1...8. wobei: x:= 1...4	Dieses Wert wird nur in der GUI-Anzeige verwendet, die Erfassung wird dadurch nicht beeinflusst. Dieser Wert ist vom System abhängig und kann der Anlagenkonfiguration entnommen werden.
<systempart name="warmwasser"> <load_pump>	Flag (True/False)	Dieses Flag bestimmt, ob die Warmwasser-Erzeugung Teil des Heizgerätes (False) oder als externer Wasserspeicher mit separater Ladepumpe (True) vorhanden ist. Es wird nur die GUI-Anzeige damit gesteuert, die Erfassung wird dadurch nicht beeinflusst.
<systempart name="solar"> <second_heater>	Flag (True/False)	Dieses Flag bestimmt, ob es ein zweites Heizsystem (True) gibt oder nicht (False). Dieses zweite Heizsystem (Feststoff-Kessel etc.) wird z.Z. dem Solar-System als „Hybrid-Anteil“ zugeordnet. In der Regel gibt es dann einen separaten Puffer-Speicher, dessen Werte mit Systemmodulen (z.B. ISM2) überwacht werden. Es wird nur die GUI-Anzeige damit gesteuert, die Erfassung wird dadurch nicht beeinflusst.
<systempart name="solar"> <second_buffer>	Flag (True/False)	Dieses Flag bestimmt, ob es einen separaten Pufferspeicher im System gibt (True) oder nicht (False). Dieser wird in der Regel durch Systemmodule (z.B. ISM2) überwacht. Es wird nur die GUI-Anzeige damit gesteuert, die Erfassung wird dadurch nicht beeinflusst.
<logitem name=""> <datatype>	INT REAL TEXT	Wird bei der Erzeugung der SQL-Datenbank als Datentyp genutzt.
<logitem name=""> <datause>	GAUGE COUNTER ... weitere siehe rrdtool	Wird bei der Erzeugung der rrdtool-Datenbank als Logitem-type genutzt. Zur Zeit wird nur der Type: GAUGE verwendet.
<logitem name=""> <maxvalue>	Logitem abhängig	Maximale Wert des 'Logitems'. Kann bei der Erzeugung der rrdtool-Datenbank als Maximalwert gesetzt werden, z.Z. nicht verwendet.
<logitem name=""> <default>	Logitem abhängig	Default Wert des 'Logitems'. Kann bei der Erzeugung der rrdtool-Datenbank als

Parameter	Werte	Funktion
		Defaultwert gesetzt werden, z.Z. nicht verwendet.
<logitem name=""> <unit>	Grad Status % Minuten Zaehler Wh kWh	Dieser Wert wird für die grafische Anzeige (GUI) verwendet und ist frei wählbar.
<logitem name=""> <displayname>	Text ist Logitem abhängig und frei wählbar	Dieser Wert wird für die grafische Anzeige (GUI) verwendet und ist frei wählbar. Ist der Wert leer, wird das Logitem nicht in der GUI angezeigt.

Tabelle 4: Konfigurations-Parameterbeschreibung

rrdtool-Archivwerte	Archiv-Details	Bemerkung
LAST	saved every 5 minutes, kept for 10years back	Letzter Wert wird alle 5 Minuten gespeichert und über 10 Jahre gehalten. Danach werden die ältesten Daten überschrieben.
AVERAGE	saved every 1 minute, kept for 1year back	Der Durchschnittswert wird jede Minute gespeichert und ein Jahr gehalten. Danach werden die ältesten Daten überschrieben.
MAX	saved every 5 minutes, kept for 1year back	Der Maximalwert wird alle 5 Minuten gespeichert und ein Jahr gehalten. Danach werden die ältesten Daten überschrieben.
MIN	saved every 5 minutes, kept for 1year back	Der Minimalwert wird alle 5 Minuten gespeichert und ein Jahr gehalten. Danach werden die ältesten Daten überschrieben.

Tabelle 5: RRDTool Archiv-Details

(Details sind der rrdtool - Beschreibung zu entnehmen [2])

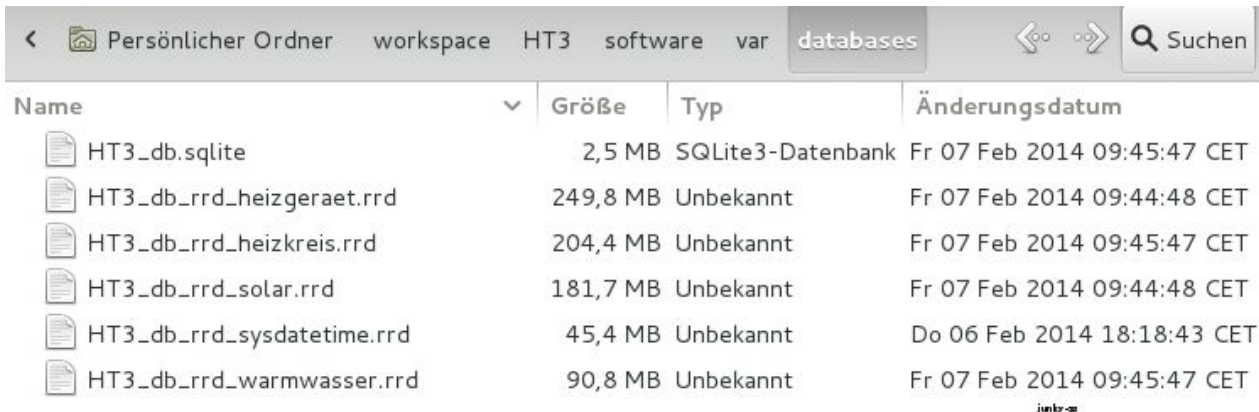
2.3 Datenbanken

Im Verzeichnis 'var/databases' sind die Datenbank-Dateien für 'SQLite' und 'rrdtool' abgelegt.

Diese werden erstmalig beim Aufruf: 'create_databases.py' mit den Informationen der Konfiguration erzeugt

(Hinweis: Die Heizkreise werden jetzt anders als in der Abbildung mit '...rrd_heizkreis1.rrd' bis '...rrd_heizkreis4.rrd' erzeugt).

Die Verzeichnis-Struktur ist wie folgt:



Name	Größe	Typ	Änderungsdatum
HT3_db.sqlite	2,5 MB	SQLite3-Datenbank	Fr 07 Feb 2014 09:45:47 CET
HT3_db_rrd_heizgeraet.rrd	249,8 MB	Unbekannt	Fr 07 Feb 2014 09:44:48 CET
HT3_db_rrd_heizkreis.rrd	204,4 MB	Unbekannt	Fr 07 Feb 2014 09:45:47 CET
HT3_db_rrd_solar.rrd	181,7 MB	Unbekannt	Fr 07 Feb 2014 09:44:48 CET
HT3_db_rrd_sysdatetime.rrd	45,4 MB	Unbekannt	Do 06 Feb 2014 18:18:43 CET
HT3_db_rrd_warmwasser.rrd	90,8 MB	Unbekannt	Fr 07 Feb 2014 09:45:47 CET

Abbildung 9: Datenbanken Verzeichnis

Die Größe der erzeugten Datei wächst bei der SQLite-Datenbank pro Tag um ca. 4 MByte an.

Nach einem Jahr ist mit ca. 1.5 GByte Daten zu rechnen. Eine Löschroutine ist z.Zeit nicht realisiert.

Bei der rrdtool-Datenbank bleibt die Dateigröße fest bestehen und wird durch die gewählte Archiv-Speichertiefe und Anzahl der 'Logitems' bei der Erzeugung der Datenbank bestimmt.

Eine nachträgliche Veränderung der Datenbank-Strukturen bei vorhandenen Datenbanken ist schwierig und kann zu Datenverlusten führen.

2.3.1 Datenbank 'SQLite'

Die Datenbank 'sqlite' wird erzeugt, sobald im Konfigurationsfile der Parameter <sql-db><enable>on aktiviert ist (Details siehe: 2.2 Konfiguration). Eine schon vorhandene Datenbank wird nicht überschrieben.

Die SQLite-Datenbank wird für die 'Zwischen'-Speicherung der dekodierten Datentelegramme benutzt. Intervall gesteuert werden die Daten von der SQL-Datenbank in die rrdtool-Datenbank übertragen.

Eine Auswertung der Daten wird z.Z. in dieser Datenbank nicht gemacht.

Geplant sind: Solarertrag pro Tag und Solarertrag-Gesamtsumme.

Beide Informationen sind nicht in den Datentelegrammen enthalten.

Die folgenden Bilder zeigen die Tabellen und Spalten der SQL-Datenbank.
(SQLite Plugin des Firefox)

Die Tabellen-Namen entsprechen dabei dem 'systempart'-Namen aus der Konfiguration. Eine Ausnahme davon ist die Tabelle 'rrdtool_infos', deren Aufgabe weiter unten beschrieben ist.

Die Spalten: 'Local_date_time' (lokale Zeit) und 'UTC' (UTC-Zeit) enthalten die Zeitstempel, in denen der Datenbank-Eintrag erfolgt ist. Diese Einträge sind in jeder Tabelle enthalten und werden automatisch beim SQL-'insert'-Aufruf erzeugt.

row	Local_date_time	UTC	T_vorlauf_soll	T_vorlauf_ist	T_ruecklauf	T_mischer	V_mocul	V_brenner_motor	V_h_m
25	2014.02.05 18:23:28	1391707406	37	23.7	23.2	39.2	1	0	1
27	2014.02.05 18:23:38	1391707418	37	23.5	23.2	39.2	1	0	1
28	2014.02.05 18:23:48	1391707428	37	23.7	23.2	39.2	1	0	1
29	2014.02.05 18:23:58	1391707438	37	23.5	23.2	39.2	1	0	1
30	2014.02.05 18:24:08	1391707448	37	23.7	23.2	39	1	0	1
31	2014.02.05 18:24:18	1391707458	37	23.7	23.2	39	1	0	1
32	2014.02.05 18:24:28	1391707468	37	23.5	23.2	39	1	0	1
33	2014.02.05 18:24:38	1391707478	37	23.5	23.2	39	1	0	1
34	2014.02.05 18:24:48	1391707488	37	23.7	23.2	39.2	1	0	1
35	2014.02.05 18:24:58	1391707498	37	23.5	23.2	39	1	0	1
36	2014.02.05 18:25:08	1391707508	37	23.5	23.2	39	1	0	1
37	2014.02.05 18:25:18	1391707518	37	23.5	23.2	39	1	0	1
38	2014.02.05 18:25:28	1391707528	37	23.5	23.2	39	1	0	1
39	2014.02.05 18:25:38	1391707538	37	23.5	23.2	39	1	0	1
40	2014.02.05 18:25:48	1391707548	37	23.5	23.2	38.9	1	0	1
41	2014.02.05 18:25:58	1391707558	37	23.5	23.2	38.9	1	0	1
42	2014.02.05 18:26:08	1391707568	37	23.5	23.2	38.9	1	0	1
43	2014.02.05 18:26:18	1391707578	37	23.5	23.2	38.9	1	0	1

Abbildung 10: SQL-Datenbanktabelle 'heizgeraet'

Der UTC-Zeitstempel wird für die interne Bestimmung von Zeitintervallen und als Zeit-Referenz für den intervall gesteuerten Eintrag in die rrdtool-Datenbank genutzt. Dabei ist dieser Wert von der Sommer-/Winterzeit-Umschaltung unabhängig und ist somit immer eine inkrementelle Referenz.

Zu beachten ist die korrekte Zeiteinstellung von CPU / Laptop, auf denen die Applikation läuft. Bei CPU's ohne RTC (RealTimeClock) kann die Zeitsynchronisation durch den Anschluss an ein Gateway/Router (z.B. Fritzbox) erreicht werden.

Die Tabelle 'rrdtool_infos' wird für die Synchronisation der SQLite-Datenbank mit der rrdtool-Datenbank verwendet.

Falls in der Konfiguration die rrdtool-Datenbank deaktiviert ist, wird diese Tabelle nicht erzeugt. Eine Kommunikation mit der rrdtool-Datenbank ist dann nicht möglich.

Das Bild zeigt die Tabelle 'rrdtool_infos' und die zugehörigen Spalten.

rowid	Local_date_time	UTC	rrdtool_timestamp	comment	errors
3386	2014.02.07 08:27:48	1391758068	1391758065	SO:1391758030	None
3387	2014.02.07 08:27:48	1391758068	1391758065	H-G 1391758010	None
3384	2014.02.07 08:27:47	1391758067	1391758065	WW 1391758010	None
3385	2014.02.07 08:27:47	1391758067	1391758065	H-K:1391758061	None
3382	2014.02.07 08:26:48	1391758008	1391758005	SO:1391757968	None
3383	2014.02.07 08:26:48	1391758008	1391758005	H-G 1391757950	None
3380	2014.02.07 08:26:47	1391758007	1391758005	WW 1391757951	None
3381	2014.02.07 08:26:47	1391758007	1391758005	H-K:1391758001	None
3378	2014.02.07 08:25:48	1391757948	1391757945	SO:1391757907	None
3379	2014.02.07 08:25:48	1391757948	1391757945	H-G 1391757901	None
3376	2014.02.07 08:25:47	1391757947	1391757945	WW 1391757901	None
3377	2014.02.07 08:25:47	1391757947	1391757945	H-K:1391757940	None
3374	2014.02.07 08:24:48	1391757888	1391757885	SO:1391757845	None
3375	2014.02.07 08:24:48	1391757888	1391757885	H-G 1391757830	None
3372	2014.02.07 08:24:47	1391757887	1391757885	WW 1391757841	None
3373	2014.02.07 08:24:47	1391757887	1391757885	H-K:1391757879	None
3370	2014.02.07 08:23:48	1391757828	1391757825	SO:1391757784	None
3371	2014.02.07 08:23:48	1391757828	1391757825	H-G 1391757772	None

Abbildung 11: Tabelle 'rrdtool_infos'

Tabellen-Spalte	Bedeutung	Bemerkung
Local_date_time	Zeitpunkt (Local time) des Daten-'insert'.	Wert wird automatisch beim SQL-insert eingetragen.
UTC	Zeitpunkt (UTC-time) des Daten-'insert'.	Wert wird automatisch beim SQL-insert eingetragen.
rrdtool_timestamp	Oberer Zeitstempel (UTC-time) der Daten für die rrdtool-Datenbank. Diese Spalte sorgt dafür, das auch bei einem Neustart des Programms noch fehlende Daten-Übertragungen nachgeholt werden. Dazu wird der größte 'rrdtool_timestamp'-Wert verwendet um von dort an die Datensatz-Übertragung zu starten. Jede Übertragung zur rrdtool-Datenbank wird hier eingetragen, unabhängig davon ob ein Fehler aufgetreten ist oder nicht.	Es wird hier der obere Zeitwert (UTC) des gerade aktuellen Intervalls verwendet. Die an die rrdtool-db übertragenen Daten haben maximal diesen UTC-Zeitstempel. (Im Bild ist ein 60 Sekunden Intervall zu sehen) Es wird nur der erste Eintrag eines <Systemparts> an die rrdtool-db übertragen, auch wenn mehrere gültige Einträge innerhalb eines Intervalls vorhanden sind.
comment	Text-String mit Informationen, die zu Debug-Zwecken dienen können.	Bedeutung der Informationen im Bild: <Nickname>:<UTC-Timestamp> Beispiel: SO:1391757784 SO := Systempart 'Solar' UTC:= Zeitwert an dem dieses Telegramm in die SQL-db eingetragen wurde.
errors	Fehlereintrag oder None	Falls bei der Datenübertragung in die rrdtool-db Fehler aufgetreten sind, wird dies hier eingetragen. Tritt kein Fehler auf, wird hier 'None' eingetragen. (Wenn ein Wert mit gleichem Zeitstempel erneut in die rrdtool-db eingetragen wird ist dies z.B. ein Fehler)

Tabelle 6: Beschreibung der SQLite Tabelle 'rrdtool_infos'

2.3.2 Datenbank 'rrdtool'

Die Datenbank 'rrdtool' wird erzeugt, sobald im Konfigurationsfile der Parameter `<rrdtool-db><enable>on` aktiviert ist (Details siehe: 2.2 Konfiguration). Eine schon vorhandene Datenbank wird nicht überschrieben.

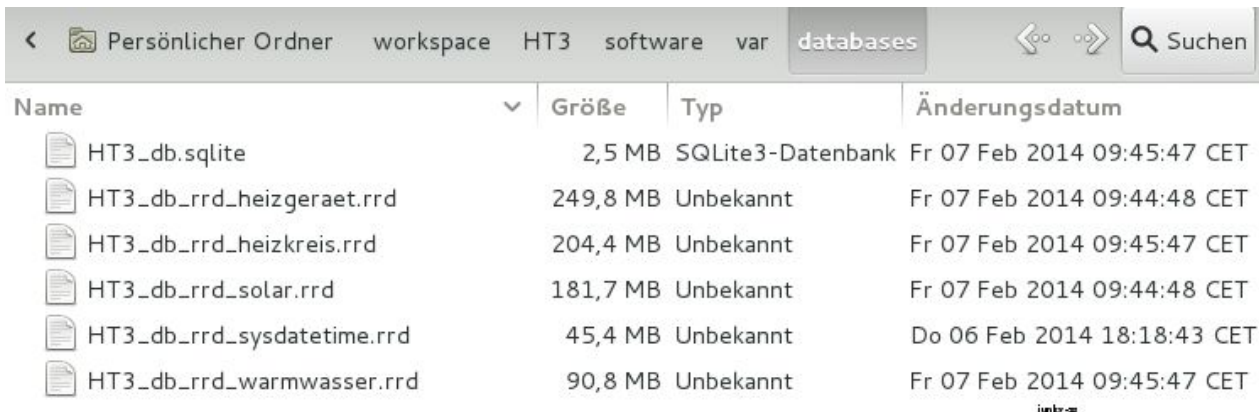
Für die Erzeugung der Datenbank und für die Daten-Aktualisierung werden perl-scripte dynamisch erzeugt und ausgeführt. Ebenso wird die Grafikerzeugung durch 'rrdgraph' mit einem perl-script realisiert.

Die Daten der SQLite-Datenbank werden in 60 Sekunden Intervallen in die rrdtool-Datenbank übertragen.

Es werden alle SQL-Datensätze aus den einzelnen Tabellen übertragen, mit Ausnahme von:

Tabelle : 'sysdatetime' und 'rrdtool_infos'
Tabellen-Spalten: 'Local_date_time', 'UTC' und 'hexdump'

Für jeden 'syspart' des Heizungssystems ('heizgeraet', 'heizkreis(n:=1...4)', 'warmwasser', 'solar' und 'sysdatetime') gibt es ein eigenes 'rrdtool'-Datenbankfile mit dem Datei-Suffix: '.rrd' (siehe Bild).



Name	Größe	Typ	Änderungsdatum
HT3_db.sqlite	2,5 MB	SQLite3-Datenbank	Fr 07 Feb 2014 09:45:47 CET
HT3_db_rrd_heizgeraet.rrd	249,8 MB	Unbekannt	Fr 07 Feb 2014 09:44:48 CET
HT3_db_rrd_heizkreis.rrd	204,4 MB	Unbekannt	Fr 07 Feb 2014 09:45:47 CET
HT3_db_rrd_solar.rrd	181,7 MB	Unbekannt	Fr 07 Feb 2014 09:44:48 CET
HT3_db_rrd_sysdatetime.rrd	45,4 MB	Unbekannt	Do 06 Feb 2014 18:18:43 CET
HT3_db_rrd_warmwasser.rrd	90,8 MB	Unbekannt	Fr 07 Feb 2014 09:45:47 CET

Einträge in die rrdtool-db dürfen für ein 'Logitem' nur einmal für einen bestimmten Zeitstempel gemacht werden. Eine Wiederholung mit gleichem Zeitstempel führt zu einer Fehlermeldung der Datenbank.

Die Einträge in der SQLite-Datentabelle 'rddtool_info' sorgen für die korrekte zeitliche Übertragung in die rrdtool-Datenbank.

Siehe [2]

Wie die HT3-Analyser-GUI mit 3 Heizkreisen und dem Heizgeräte-Type 'KUB' aussehen kann, zeigt das folgende Bild:

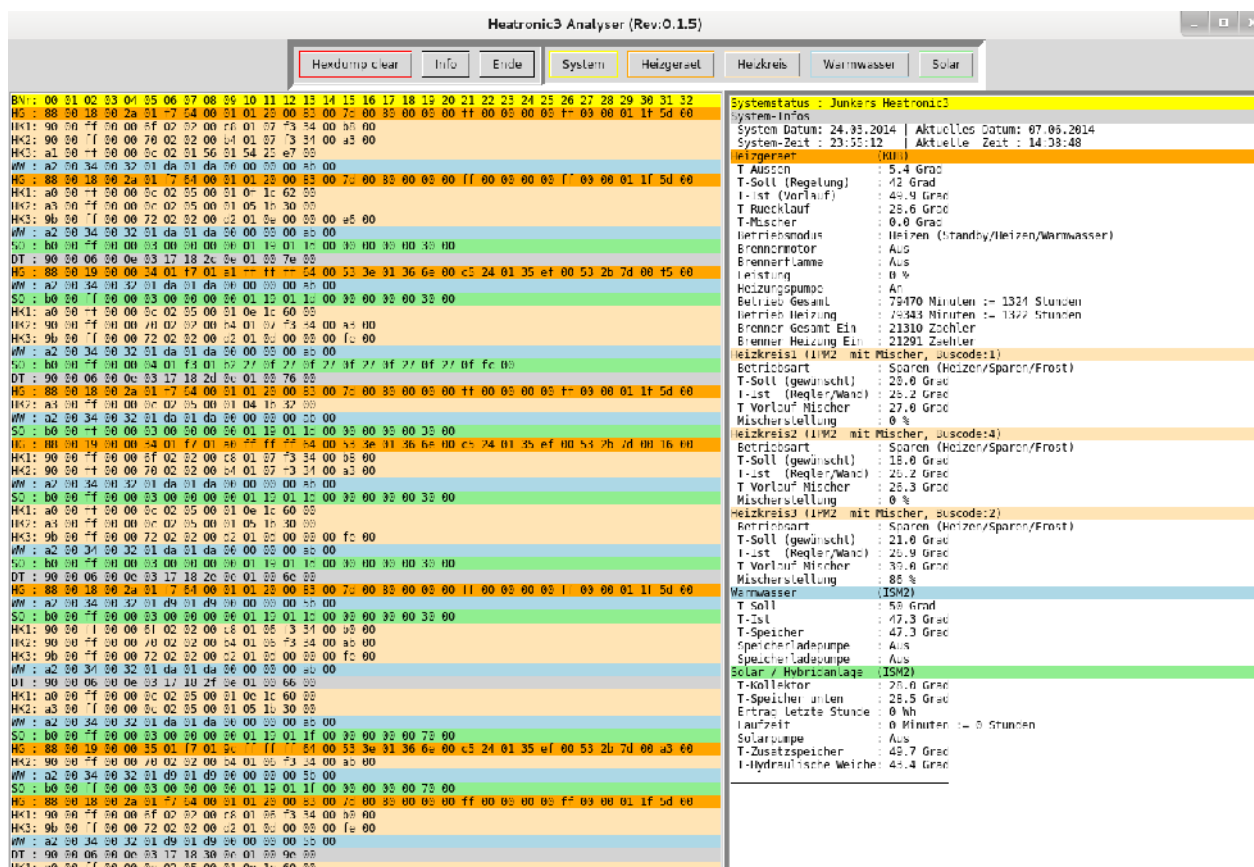


Abbildung 13: HT3 Analyser GUI (System mit 3 Heizkreisen)

2.4.2 HT3_Systemstatus

Der HT3 Systemstatus zeigt als Übersicht den aktuellen Heizungssystemstatus grafisch an. Die einzelnen Systemkomponenten (Heizgeraet, Heizkreis, Warmwasser und Solar) können wie beim HT3_Analyser separat ausgewählt werden.

Heatronic3 Systemstatus (Rev:0.1.5)

System Heizgeraet Heizkreis Warmwasser Solar Ende

Systemstatus : Junkers Heatronic3

System-Infos

System-Datum: 07.06.2014 | Aktuelles Datum: 07.06.2014
System-Zeit : 12:14:18 | Aktuelle Zeit : 12:13:17

Heizgeraet (CSW)

T-Aussen : 23.5 Grad
T-Soll (Regelung) : 0 Grad
T-Ist (Vorlauf) : 40.6 Grad
T-Ruecklauf : 33.8 Grad
T-Mischer : 40.8 Grad
Betriebsmodus : Standby (Standby/Heizen/Warmwasser)
Brennermotor : Aus
Brennerflamme : Aus
Leistung : 0 %
Heizungspumpe : Aus
Zirkulationspumpe : Aus
Speicherladepumpe : An
Betrieb Gesamt : 193201 Minuten := 3220 Stunden
Betrieb Heizung : 175481 Minuten := 2924 Stunden
Brenner Gesamt Ein : 6447 Zaehler
Brenner Heizung Ein : 3237 Zaehler

Heizkreis1 (CSW mit FB10 ohne Mischer, Buscode:1)

Betriebsart : Heizen (Heizen/Sparen/Frost)
T-Soll (gewünscht) : 21.5 Grad
T-Ist (Regler/Wand) : 28.1 Grad
T-Raum (FB10/FB100) : 22.2 Grad

Warmwasser (CSW)

T-Soll : 50 Grad
T-Ist : 40.8 Grad
T-Speicher : 47.0 Grad
Betriebszeit : 17720 Minuten := 295 Stunden
Brenner WarmW Ein : 3210 Zaehler
Warmwasser-Erzeugung : Aus
Speichertemperatur OK: Ja
Nachladung : Aus

Solar (ISM1)

T-Kollektor : 241.6 Grad
T-Speicher unten : 80.4 Grad
Ertrag letzte Stunde : 916 Wh
Laufzeit : 54928 Minuten := 915 Stunden
Solarpumpe : Aus
Kollektorfeld deaktiv: Ja
Solarspeicher voll : Ja

2.4.3 HT3_Logger

Der HT3 Logger erfasst die Protokoll-Daten und schreibt diese in die SQLite- und rrdtool-Datenbank. Eine grafische Ausgabe ist nicht vorhanden.
Der Betrieb des Loggers ist für Anwendungen ohne Grafikausgabe vorgesehen.

3 Installation

Vor der Installation der Applikation ist das Betriebssystem zu aktualisieren. Je nach Betriebssystem sind unterschiedliche Aktionen erforderlich. Diese Beschreibung beschränkt sich auf das Betriebssystem: 'Linux-Debian Wheezy'.

3.1 Betriebssystem

Aktualisierung des Betriebssystems mit:

```
# apt-get update
```

Den letzten Ausgabestand aktivieren:

```
# apt-get upgrade
```

Python3 installieren (falls noch nicht vorhanden):

```
# apt-get install python3
```

Seriellen Treiber für Python3 laden:

```
# apt-get install python3-serial
```

Perl objekt-orientiertes RRDTool Interface installieren:

```
# apt-get install librrdtool-oo-perl
```

anschliessend

```
# apt-get autoremove
```

RRDTool Datenbank installieren:

```
# apt-get install rrdtool
```

User in Gruppe < dialout > aufnehmen:

```
# adduser 'username' dialout
```

Deaktivieren der default eingeschalteten TTY-Systemausgaben (RaspberryPI):

Datei: /boot/cmdline.txt anpassen

```
# nano /boot/cmdline.txt
```

Zu editierende Zeile finden 'dwc_otg.lpm_enable=...' und anpassen:

von

```
dwc_otg.lpm_enable=0 console=ttyAMA0,115200 kgdboc=ttyAMA0,115200 console=tty1
```

in

```
dwc_otg.lpm_enable=0 console=tty1
```

Danach Datei speichern

Siehe auch [1]

Anpassen der /etc/inittab (RaspberryPi):

Datei: /etc/inittab anpassen

```
# nano /etc/inittab
```

Deaktivierung der Zeile durch Anpassung

von

```
#Spawn a getty on Raspberry Pi serial line  
T0:23:respawn:/sbin/getty -L ttyAMA0 115200 vt100
```

in

```
#Spawn a getty on Raspberry Pi serial line  
#T0:23:respawn:/sbin/getty -L ttyAMA0 115200 vt100
```

Danach Datei speichern

Neustart des Raspberry Pi:

```
# reboot
```

3.2 Applikation

(Vor der Installation der Applikation ist das Betriebssystem zu aktualisieren).

Einrichten eines Datenbank-Verzeichnisses (falls dies separat sein soll, als root):

Beispiel für einen Verzeichnis auf dem USB-Stick

```
# mkdir -p /media/usbstick/HT3/sw/var/databases  
# cd /media  
# chmod -R 775 ./usbstick/HT3/  
# chown -R 'username' ./usbstick/HT3/
```

Bemerkung:

Die **Verzeichnis-Struktur unterhalb** von '/media/usbstick' sollte so angelegt werden, da das Perl-Grafikscript auf diese Struktur zugreift.

Entpacken der Applikation im gewünschten Verzeichnis (als user):

```
$ tar xzvf HT3_application_refA.tar.gz
```

Besser ist es jedoch die aktuelle Software mit Dokumentation von github.com zu holen (als user):

```
$ mkdir testverzeichnis  
$ cd testverzeichnis  
$ git clone https://github.com/norberts1/hometop_HT3.git
```

Anpassen der Konfiguration an neues Datenbank-Verzeichnis (als user):

Datei: ./etc/config/HT3_db_cf.xml anpassen

```
$ nano ./etc/config/HT3_db_cfg.xml
von
$ <dbname_sqlite>./var/databases/HT3_db.sqlite</dbname_sqlite>
in
$ <dbname_sqlite>/media/usbstick/HT3/sw/var/databases/HT3_db.sqlite</dbname_sqlite>
und von
$ <dbname_rrd>./var/databases/HT3_db_rrd</dbname_rrd>
in
$ <dbname_rrd>/media/usbstick/HT3/sw/var/databases/HT3_db_rrd</dbname_rrd>

Danach Datei speichern
```

Erzeugen der Datenbanken (als user):

```
$ cd ./HT3/sw
$ ./create_databases.py
!! Achtung
Das Erzeugen der Datenbanken dauert einige Zeit ( > 5 und < 15 Minuten) auf dem Raspberry Pi.
```

Aktivieren eines zentralen Cronjobs zur Grafikgenerierung (als root):

```
# cp ./HT3/sw/etc/rrdtool_draw.pl /usr/local/bin
# chown 'username' /usr/local/bin/rrdtool_draw.pl
cronjob einrichten
# nano /etc/crontab
Zeile hinzufügen
*/5 * * * * 'username' /usr/bin/perl -S rrdtool_draw.pl /media/usbstick /home/'username'/ 1

                                (Parameter:                1.                2.                3.)

Datei speichern
```

Bedeutung:

Alle 5 Minuten ausführen als 'user' mit Perl das script=' rrdtool_draw.pl' und Parametern:
1.'Datenbank-Verzeichnis'; 2.'Zielverzeichnis'; 3. Anzahl Heizkreise.
Die erzeugten Grafiken werden unter <Zielverzeichnis>/HT3/sw/etc/html abgelegt.

Anpassung und Aktivieren des Startscripts 'ht3_logger' (als root):

Username und Verzeichnisse sind gegebenenfalls anzupassen

```
# nano ./HT3/sw/etc/sysconfig/ht3_logger
USER="zs"                                <!-- auf erforderlichen Wert korrigieren
DAEMON=/home/$USER/HT3/sw/$NAME
PIDFILE=/home/$USER/HT3/sw/var/run/$NAME.pid
APPLICATION_FOLDER=/home/$USER/HT3/sw/

Datei speichern
Danach Datei kopieren:
# cp ./HT3/sw/etc/sysconfig/ht3_logger /etc/init.d
Script aktivieren:
# cd /etc/init.d
# insserv ht3_logger
```

Anpassung und Aktivieren des Startscripts 'httpd' (als root):

(Nur erforderlich, falls man keinen anderen Http-Server installieren will)

Username und Verzeichnisse sind gegebenenfalls anzupassen

```
# nano ./HT3/sw/etc/sysconfig/httpd
USER="zs"                                <-- auf erforderlichen Wert korrigieren
DAEMON=/home/$USER/HT3/sw/$NAME
PIDFILE=/home/$USER/HT3/sw/var/run/$NAME.pid
APPLICATION_FOLDER=/home/$USER/HT3/sw/
```

Datei speichern

Danach Datei kopieren:

```
# cp ./HT3/sw/etc/sysconfig/httpd /etc/init.d
```

Script aktivieren:

```
# cd /etc/init.d
```

```
# insserv httpd
```

Anpassen der Applikationen an die Schnittstelle:

Je nach Schnittstellen-Typ folgende Werte verwenden:

```
# deviceport="/dev/ttyAMA0"    <-- UART-Schnittstelle des Raspberry Pi
# deviceport="/dev/ttyUSB0"    <-- 1. USB -Schnittstelle des Raspberry Pi / Laptop
```

oder

```
# deviceport="/dev/ttyUSB1"    <-- 2. USB -Schnittstelle des Raspberry Pi / Laptop
```

Die Applikationen entsprechend anpassen, folgende Module ändern:

1. /home/\$USER/HT3/sw/HT3_Analyser.py
2. /home/\$USER/HT3/sw/HT3_Systemstatus.py
3. /home/\$USER/HT3/sw/HT3_Logger.py

Neustart des Rechners (als root):

```
# reboot
```

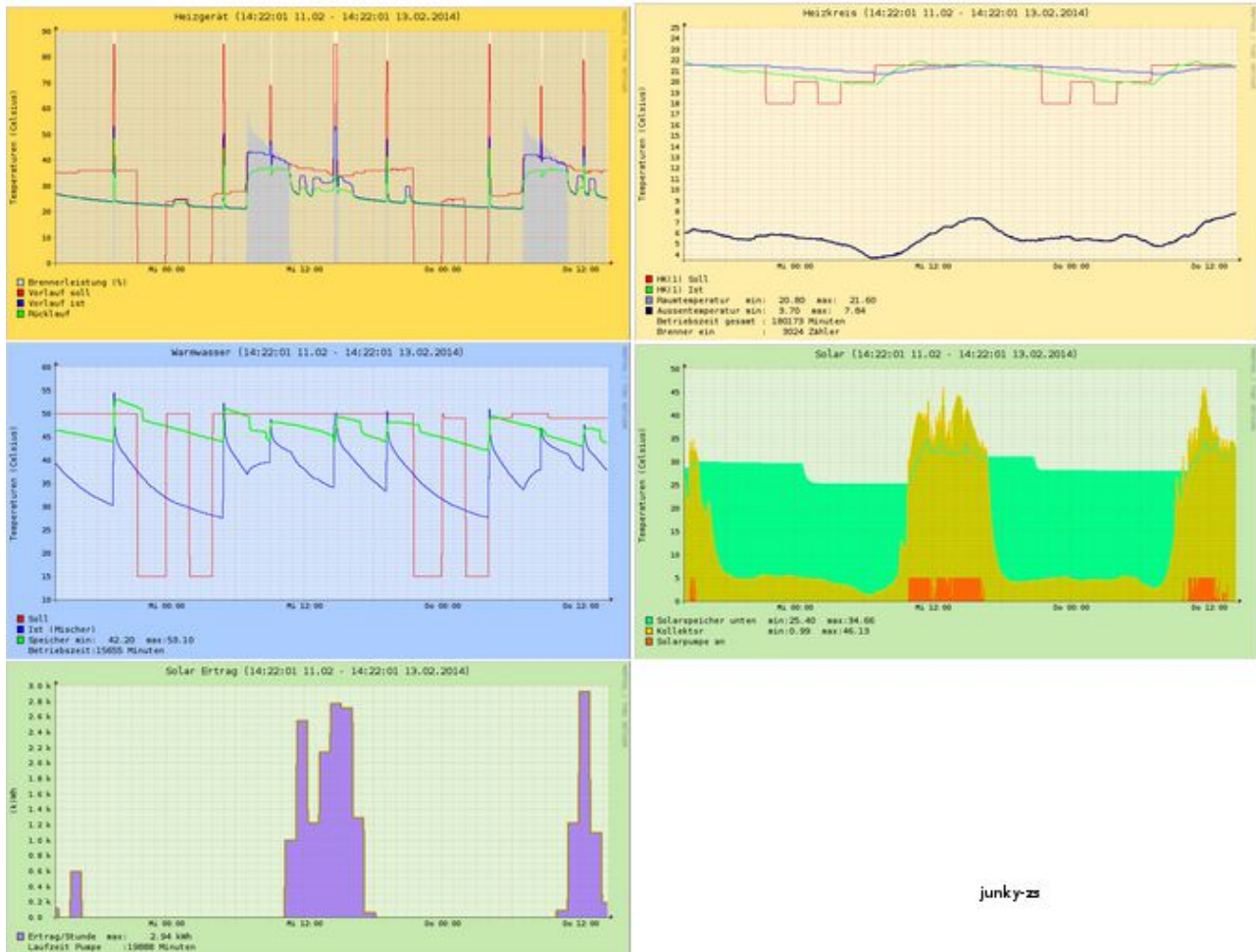
Nach dem Neustart des Rechners muss die Applikation 'HT3_Logger.py' automatisch gestartet worden sein, Daten erfassen und in die Datenbanken schreiben.

Es werden dann alle 5 Minuten die Grafikausgaben der rrdtool-Datenbank im Verzeichnis: <Zielverzeichnis>/HT3/sw/etc/html/ als *.png Files abgelegt. Alte PNG-Files werden überschrieben.

Ebenfalls muss der Http-Server 'httpd.py' automatisch gestartet worden sein. Dieser Server erwartet Anfragen auf dem Port: 8086 und sobald PNG-Dateien erzeugt wurden (alle 5 Minuten intervallbasiert), werden diese vom Browser angezeigt.

4 HT3 Applikation im Betrieb

Folgende Bilder zeigen grafische Ausgaben aus der rrdtool-Datenbank von erfassten Heizungssystemdaten. Die Grafiken werden als PNG-Dateien erzeugt und mit einem Browser dargestellt (das Anzeigeintervall hier ist 2 Tage).



junky-zs

Abbildung 14: HT3 Systemhistorie im Browserfenster

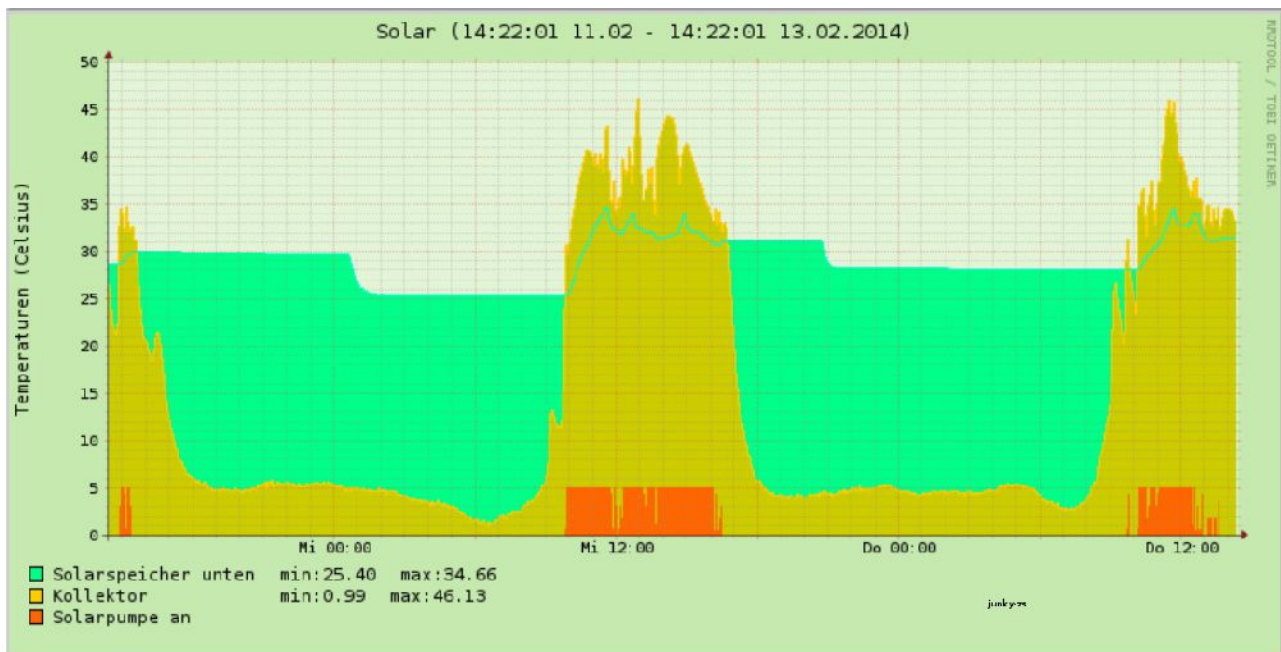


Abbildung 15: HT3 Solarhistorie im Browserfenster

Das HTML-Modul ist zur Zeit sehr einfach gehalten und nur für die Anzeige der Grafiken ausgelegt. Verschiedene Anpassungen sind denkbar, z.B. die Auswahl des Anzeigeintervalls. Dies ist aber noch nicht realisiert.

HTML-MODUL './HT3/sw/etc/index.html' :

```
<HTML>
<HEAD>
<TITLE>Heizungs Historie</TITLE></HEAD>
<BODY>
  <IMG src="./HT3_Heizgeraet.png" alt="Heizgeraet">
  <IMG src="./HT3_Warmwasser.png" alt="Warmwasser">
  <BR>
  <IMG src="./HT3_Heizkreis1.png" alt="Heizkreis1">
  <IMG src="./HT3_Heizkreis2.png">
  <BR>
  <IMG src="./HT3_Heizkreis3.png">
  <IMG src="./HT3_Heizkreis4.png">
  <BR>
  <IMG src="./HT3_Solar.png" alt="Solar">
  <IMG src="./HT3_Solarertrag.png" alt="Solarertrag">
</BODY>
</HTML>
```

Die Grafiken werden mit einem Perl-Modul aus den Daten der rrdtool-Datenbank für das ausgewählte Intervall erzeugt und im '<Zielverzeichnis>/HT3/sw/etc/html'-Verzeichnis abgelegt.

Das 'Zielverzeichnis' ist nach der Installation das Installations-Verzeichnis.

Falls man den 'Http-Daemon Lite' (./HT3/sw/etc/html/httpd.py) nutzen möchte, müssen die erzeugten Grafiken im Verzeichnis des Daemon's liegen.

5 Weiterführende Literatur und URL's

Python 3	Lernen und professionell anwenden Michael Weigend	Verlag: mitp
Python 3	Das umfassende Handbuch Johannes Ernesti und Peter Kaiser	Verlag: Galileo Computing
[1]	http://kampus-elektroecke.de	Elektronik, Code und mehr
[2]	http://oss.oetiker.ch/rrdtool	About RRDtool
[3]	http://www.mrtg.org/rrdtool/gallery/index.en.html	RRDtool Gallery
[4]	http://code.google.com/p/pyrrd	A Pure-Python OO wrapper for RRDTool
[5]	http://www.mikrocontroller.net/forum/Haus & Smart Home	Forum: Haus & Smarthome