1 SMA Sunny Island 5048

Wie bereits in der Einleitung genannt, soll die Kommunikation mit dem Inselwechselrichter Sunny Island 5048 der Firma SMA und dem Raspberry Pi über die Schnittstelle RS 485 eingerichtet werden, um aktuelle Daten des Wechselrichters auszulesen.
Um nicht einzelne Datenframes neu anfertigen zu müssen, zur Ansteuerung der RS 485 Schnittstelle, wird die von SMA kostenlos zur Verfügung gestellte Software mit dem Namen YASDI verwendet. YASDI eignet sich für die Entwicklung eigener Applikationen zur Kommunikation mit SMA Produkten. YASDI liegt vollständig im Sourcecode in der Programmiersprache C vor und kann nach jedem beliebigen Wunsch angepasst werden. Bei dieser Projektarbeit wird ein ebenfalls von SMA bereitgestelltes Testprogramm verwendet, das in der Programmiersprache C geschrieben ist, mit dem Namen YASDISHELL.
An Ende dieses Kapitels soll noch auf aufgetretene Probleme bei der Kommunikation eingegangen werden, die leider sehr viel Zeit in Anspruch genommen haben.

Abbildung 1: Inselwechselrichter Sunny Island 5048 der Firma SMA (3)

1.1 RS 485

Die RS 485 Schnittstelle ist eine von SMA eingesetzte Standardschnittstelle zur sicheren Übertragung von Daten zwischen einzelner SMA Komponenten. Dabei ist es möglich mit bis zu 50 Teilnehmern eine störsichere Übertragung auf bis zu 1200 Meter zu realisieren. Es handelt sich um eine differentielle Übertragung zwischen zwei Datenleitungen, DATA+ und DATA-. Es existiert ein gemeinsames Massepotential durch eine dritte Leitung GND. Jedoch muss diese nicht zwingend angeschlossen sein, hat sich bei Versuchen während dieser Projektarbeit ergeben. Die Übertragung ist seriell und bidirektional, wobei immer nur ein Busteilnehmer auf dem Bus senden darf. Aber alle Busteilnehmer können die Daten empfangen. (Halbduplex-Verfahren). (4)
Zwischen den beiden DATA Leitungen liegt mindestens eine Spannung von +/- 200mV, also 400mV. Meist sind es jedoch bis zu +/- 5V, also 10V. In Abbildung 2 ist der Spannungsverlauf von DATA+ und von DATA- gezeigt. Solange der Ruhepegel unter den genannten +/- 200mV liegt, wird weder eine Eins noch eine Null erkannt. Erst nachdem die Spannungsdifferenz den genannten Wert überschreitet wird eine Eins oder Null registriert, je nachdem welche Leitung eine positive und welche Leitung eine negative Spannung aufweist.

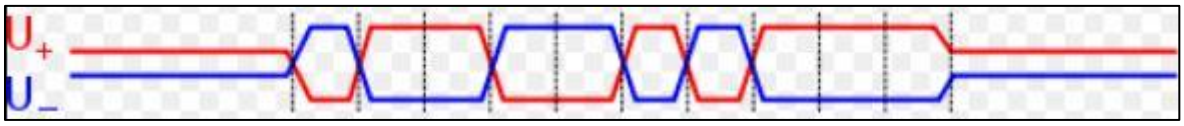

Abbildung 2: Spannungsverlauf DATA+ und DATA- (5)

Die Topologie bei RS 485 ist Daisy Chain, auf Deutsch Gänseblümchenkette. Dabei sind alle Komponenten in Serie miteinander verbunden. Abbildung 3 veranschaulicht dies.

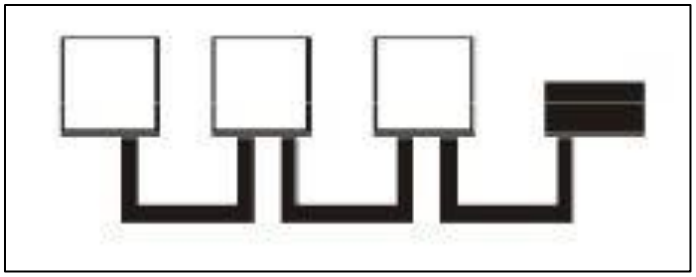

Abbildung 3: Daisy Chain Topologie (4)

1.2 USB-RS 485 Wandler

Dass der Raspberry Pi mit dem Sunny Island kommunizieren kann, wird ein USB-RS 485 Wandler eingesetzt, da der Raspberry Pi selbst keine eigene RS 485 Schnittstelle besitzt. Eine zuerst gewählte Alternative zu dem USB-RS 485 Wandler ist eine Kommunikationsplatine, die direkt auf den Pi aufgesteckt wird. Doch dazu später mehr in Kapitel 1.6.

Hier wird der Wandler DA-70157 der Firma Digitus eingesetzt. Dieser kann direkt an den Pi angesteckt werden. Allerdings wird in diesem Fall ein zusätzlicher USB Hub verwendet, um weiterhin genügend USB Anschlüsse am Pi zu haben. Doch es würde auch ohne zusätzlichen Hub funktionieren, die Stromversorgung des Pi´s ist ausreichend um den Wandler zu versorgen.

Abbildung 4: USB-RS 485 Wandler der Firma Digitus (6)

1.3 Anschlussschema

Beim Sunny Island müssen die Kommunikationsleitungen wie folgt angeschlossen werden:

Pin Kommunikations-gerät (Sub-D 9-polig)	RS232	RS485	RJ45-Buchse
2	RXD	A (Data-)	1
3	TXD	B (Data+)	3
5	GND	GND	2

Tabelle 1: Anschlussbelegung Datenleitungen Sunny Island (7)

In Abbildung 5 ist ein Foto des Bereichs im Sunny Island, um zu sehen wo die Kommunikationsanschlüss verbaut sind. Bei der Inbetriebnahme ist darauf zu achten, dass die beiden Terminierungsstecker eingesteckt sind, wenn der Sunny Island der letzte Teilnehmer am Bus ist. Außerdem ist darauf zu achten, dass der Schiebeschalter, mit der Bezeichnung RS 232, nach oben geschoben ist. Das bedeutet, dass der Betrieb mit RS 485 möglich ist.

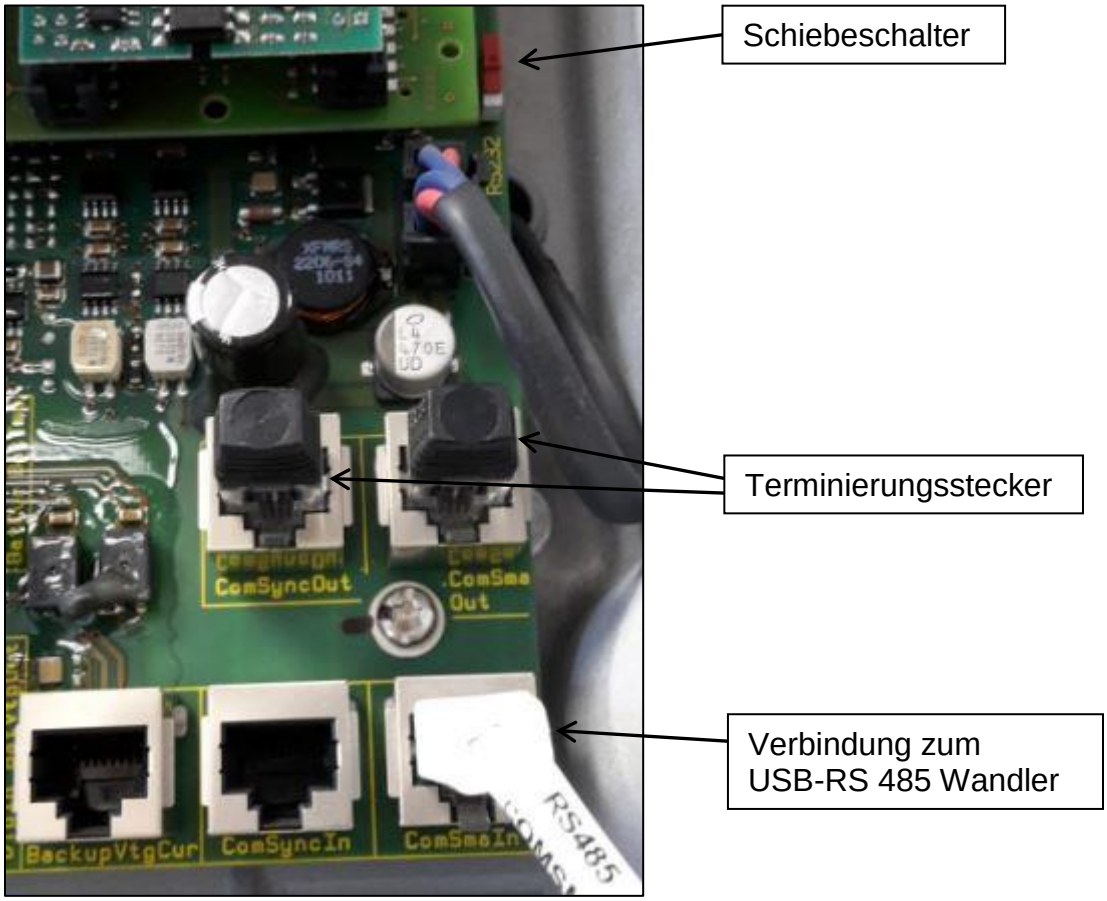

Abbildung 5: Sunny Island Kommunikationsanschlüsse

Das weiße Kabel aus Abbildung 5, Verbindung zum USB-RS 485 Wandler, muss an der Anschlussplatine des Wandlers den Bezeichnungen entsprechend angeschlossen werden.

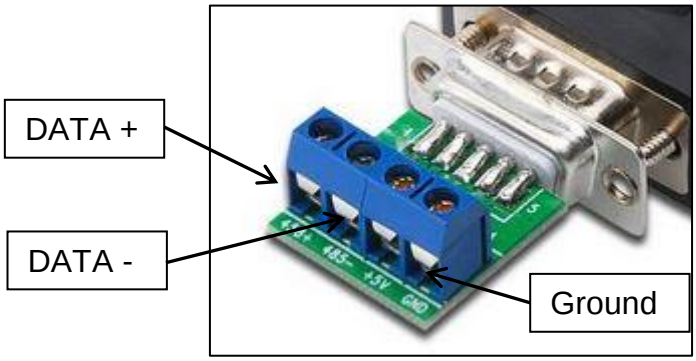

Abbildung 6: Anschlussplatine für USB-RS 485 Wandler (6)

1.4 YASDISHELL

Bei dieser Projektarbeit wird ein von SMA bereitgestelltes Testprogramm verwendet, das in der Programmiersprache C geschrieben ist, mit dem Namen YASDISHELL. Dieses Testprogramm kann, nachdem es kompiliert wurde, direkt aus dem Linuxterminal gestartet werden und über Tastatureingaben eine Auswahl über die gewünschte Aufgabe getroffen werden. Zuerst kann dem Programm die Aufforderung geben werden, nach am Bus angeschlossenen Teilnehmern zu suchen. Anschließend ist es beispielsweise möglich gewisse Parameter zu ändern oder alle Parameter auszulesen.
Um die Aufgabe der Projektarbeit, Daten des Sunny Islands auszulesen, zu erfüllen, werden zwei Funktionen der YADISHELL benötigt. Zum einen den Sunny Island am Bus finden und zum anderen dessen Daten auszulesen. Da jedoch diese Prozedur zyklisch und automatisch geschehen soll, muss der C Code der YASDISHELL entsprechend angepasst werden.

Im Folgenden wird detailliert die Vorgehensweise beschrieben, um das direkt von SMA heruntergeladenen YASDI Softwarepaket für den hier gewünschten Nutzen zu ändern und auf dem Pi lauffähig zu machen.

1.4.1 C Code YASDISHELL anpassen

Von der SMA Website muss der unkompilierte YASDI Source Code heruntergeladen werden. Nach dem Entpacken muss die Datei „CommonShellUIMain.c“ im Ordner „Shell“ mit beispielsweise dem unter Linux zur Verfügung stehenden Programm „geany“ geöffnet werden.

Folgende Punkte werden an der Originaldatei geändert:

- Doppeldeklaration von „*void PrintDevList(void)*“ wird entfernt.

- Nichtbenötigte Funktionen: „*void SetParamValue(void),*
  void DoStartDetectionAsync(int DevCnt),
  void DoCommands(void),
  void printStatTexts(DWORD ChanHandle)“ werden entfernt.

- Funktion „*void DoStartDetection(int DevCnt)“ wird zu „int DoStartDetection(int DevCnt)“* geändert, damit ein Rückgabewert möglich ist, der zur Auswertung von Fehlern nötig ist.

- In der Funktion „*PrintChannelValues“* wird die vorher erwartete Tastatureingabe, über die Nummer des Busteilnehmers dessen Parameter geholt werden sollen, durch die feste Zahl Eins ersetzt.

- In der Funktion „*DoStartDetection“* wird die Variable „*DevCnt“* mit der Zahl Eins beschrieben, da immer nur ein Busteilnehmer gesucht werden soll. Des Weiteren wird eine Eins von der Funktion zurückgegeben wenn kein Teilnehmer gefunden wird, was bedeutet, dass nichts angeschlossen oder etwas falsch angeschlossen ist.

- Die Funktion „*DoChangeAccessLevel*“ wird entfernt

- In „*int main(int argv, char **argc)*“ wird eine zusätzliche Variable „*Error*“ zur Fehlerauswertung benötigt und die Bildschirmausgabe der YASDISHELL wird geändert.
  Im Falle, dass das INI-File nicht korrekt oder nicht vorhanden ist, wird das Programm beendet.
  Im Falle, dass es ein Problem gibt beim Start des USB Treibers, wird das Programm beendet. Ansonsten wird die Detektion des Sunny Islands gestartet und anschließend alle Parameter ausgelesen.

Das komplette, geänderte C Programm ist in Anhang 7.2.1 abgedruckt.

1.4.2 INI-File

Als nächster Schritt muss das INI-File an den richtigen Ort kopiert werden und richtig ausgefüllt werden. Das INI-File wird von der YASDISHELL benötigt. Es enthält für die Kommunikation nötige Informationen.
Ursprünglich liegt das File im Verzeichnis „samples/samples1“ und muss in das Verzeichnis „/projects/generic-cmake“ kopiert werden, da dort später auch das ausführbare Programm YASDISHELL liegt.
Alles was unter „IP1“ und unter „Misc“ steht, spielt in diesem Fall keine Rolle und kann auskommentiert werden. An den zwei Zeilen unter „DriverModules“ muss nichts geändert werden, da dies so korrekt ist. Jedoch unter dem Punkt „COM1“ muss bei „Device“ „S0“ gegen „USB0“ ersetzt werden. Damit weiß die YASDISHELL, dass der Bus an einem USB Anschluss angeschlossen ist. Außerdem muss bei „Media“ „RS 232“ durch „RS 485“ ersetzt werden, da in diesem Fall die Kommunikation über RS 485 stattfinden soll. Die Punkte „Baudrate“ sowie „Protocol“ sind bereits korrekt im originalen INI-File.

Das geänderte INI-File ist in Anhang 7.2.2 abgedruckt.

1.4.3 C Code YASDISHELL kompilieren

Das C Programm muss bevor es gestartet werden kann, im Gegensatz zu Python, zuerst kompiliert werden. Dazu wird hier der Compiler „Make“ verwendet.
Im Verzeichnis „/projects/generic-cmake“ wird zuerst der Befehl „cmake“ ausgeführt. Anschließend der Befehl „make“. Durch das Kompilieren wird im Verzeichnis „/projects/generic-cmake“ neben einiger anderen Dateien die Datei „yasdishell“ erzeugt. Um diese ausführen zu können muss im Terminal zuerst in das genannte Verzeichnis gewechselt werden und anschließend mit dem Befehl „chmod +x yasdishell“ die YASDISHELL ausführbar gemacht werden. Dieser Befehl hat Einfluss auf die Zugriffrechte von „yasdishell“.
Nun kann mit „./yasdishell“ die YASDISHELL gestartet werden und wenn alles korrekt angeschlossen ist, werden nun im Terminal alle Parameter des Sunny Island erscheinen.

1.5 Pythonprogramm

Der Programmiercode von „SunnyIsland_V_0.py“ ist in Anhang 7.2.3 abgebildet. Da dieser bereits ausführlich kommentiert ist, soll hier nur kurz auf die einzelnen Funktionen eingegangen werden.

Das Pythonprogramm „SunnyIsland_V_0.py“ läuft, nachdem es gestartet wurde, in einer Endlosschleife. Es hat die Aufgabe die YASDISHELL im Hintergrund zu starten, ohne, dass es für den Anwender sichtbar ist. Dazu muss jedoch zuerst in das Verzeichnis gewechselt werden, indem die YASDISHELL liegt. Die Ausgabe der YASDISHELL wird jetzt nicht im Terminal ausgegeben, sondern in eine Textdatei mit dem Namen „Rohdaten_SunnyIsland.txt“ umgeleitet. Falls es diese Datei noch nicht gibt, wird sie neu erzeugt, andernfalls überschrieben. Diese Datei liegt im selben Verzeichnis wie auch die YASDISHELL. Solange die YASDISHELL läuft, macht das Pythonprogramm nicht weiter. Somit entsteht keine Gefahr, dass das Pythonprogramm, welches im weiteren Verlauf die Datei „Rohdaten_SunnyIsland.txt“ wieder liest, eine nicht fertig geschriebene Datei vorfindet.
Die Datei „Rohdaten_SunnyIsland.txt“ ist in Anhang 7.2.4 abgedruckt.

Die Funktion „ini“ öffnet die Datei „Rohdaten_SunnyIsland.txt“ und ließt die einzelnen Zeilen in ein Array ein. Wenn die Anzahl der eingelesenen Zeilen kleiner als 159 ist, gibt die Funktion eine Eins zurück. Diese Zahl ergibt sich daraus, dass bei korrekt eingelesenen Daten der letzte Parameter in der Zeile 159 steht. Ist ein Fehler aufgetreten, ist das File wesentlich kürzer.

Es folgt ein Errorhandling, um im Falle der fehlerhaften Kommunikation einen sinnvollen weiteren Verlauf des Pythonprogramms zu gewährleisten. Hierbei wird der Rückgabewert der Funktion „ini“ ausgewertet. Hat dieser den Wert Eins, werden alle Zeilen des Textfiles „Rohdaten_SunnyIsland.txt“ im Terminal ausgeben, worin die von der YASDISHELL festgestellten Fehler enthalten sind. Somit hat der Anwender sofort Informationen über möglich Fehlerursachen und muss nicht zuerst in das File selbst reinschauen.

Um der Aufgabe dieser Projektarbeit, einen bestimmten Parameter auszulesen, näher zu kommen, wird im Programm die Variable „number“ mit einer gewünschten Parameternummer beschrieben.
Anschließend wird die Funktion „value_Programm“ aufgerufen, welche die Variable „value“ zurückgibt. Der Wert von „value“ entspricht der Zeilennummer in „Rohdaten_SunnyIsland.txt“, in welcher der gewünschte Parameter steht. Dabei wird beachtet, dass die Parameter 252 bis 255 nicht existieren, sowie, dass der Parametersatz nicht mit Nummer 1 anfängt, sondern mit Nummer 186. Außerdem wird berücksichtigt, dass in „Rohdaten_SunnyIsland.txt“ nicht in der ersten Zeile bereits Parameter stehen.

Die Funktion „string_Programm“ erhält als Übergabeparameter die vorher bestimmte Zeilennummer und holt die einzelnen Strings aus dieser einen Zeile heraus und speichert diese in ein Array ab. Das Array wird zurückgegeben an das Hauptprogramm.

Im Hauptprogramm werden jetzt die einzelnen Elemente ausgegeben. Zuerst wird die ganze Zeile ausgegeben, danach folgen einzeln: Parameternummer, Parameterbezeichnung, Wert, Einheit. Im Anschluss wird das Array, in dem der komplette Datensatz enthalten ist, wieder gelöscht. Dies ist nötig, da es Probleme gibt, wenn das Array dann beim nächsten Auslesen einfach nur wieder überschrieben wird. Teilweise sind dann noch alte Werte im Array enthalten. Danach startet die Prozedur wieder erneut.

Das Programm „SunnyIsland_V_0.py“ steht bereit, um von weiterführenden Arbeiten für gewünschte Anwendungen verändert zu werden.

1.6 Probleme Kommunikation RS 485

Um die Kommunikation zwischen Sunny Island und Raspberry Pi herzustellen wird, wie bereits oben erwähnt, ein USB-RS 485 Wandler eingesetzt. Allerdings war zuerst geplant, die RS 485 Kommunikation nicht über den USB Adapter zu realisieren, sondern über die sowieso verwendete Kommunikationsplatine Raspicomm, welche für die RS 232 Kommunikation mit dem Studer Xtender benötigt wird. Leider führte das nicht zum Erfolg.
Um als Anschluss die Platine Raspicomm zu verwenden, muss im INI-File der YASDISHELL unter dem Punkt „Device“ das Wort „USB“ durch „RPC“ ersetzt werden.
Da bei vielen Versuchen, um die Kommunikation zwischen Pi und Sunny Island herzustellen, kein Erfolg zu verbuchen ist, wird die Differenzspannung zwischen den beiden Datenleitungen näher untersucht. Dazu werden mit dem Oszilloskop Singleshotaufnahmen gemacht.
Es wird dazu die originale YASDISHELL verwendet, unter der es möglich ist zu einem selbst gewählten Augenblick die Anweisung zu geben, Busteilnehmer zu suchen. Mit dieser Aufforderung ist es möglich ein definiertes Frame über den Bus zu senden, welches immer bei jedem erneuten Auffordern das gleiche Frame ist. Dadurch kann ein Vergleich der Differenzspannungen der beiden Datenleitungen erfolgen, wobei zweimal die Kommunikationsplatine und zweimal der letztendlich verwendete USB-RS 485 Wandler eingesetzt wird. Bei jeder Komponente wird die zeitliche Auflösung einmal auf 5ms und einmal 25ms am Oszilloskop eingestellt.

Dadurch, dass eindeutig eine Reaktion am RS 485 Anschluss der Raspicomm auf eingehende Befehle ersichtlich ist, ist klargestellt, dass die YASDISHELL die Raspicomm tatsächlich als Hardware verwendet, sofern das INI-File korrekt konfiguriert ist.

Um die Messung nicht durch ein Antworten des Sunny Islands zu verfälschen, wird dieser für die Messungen abgeklemmt.

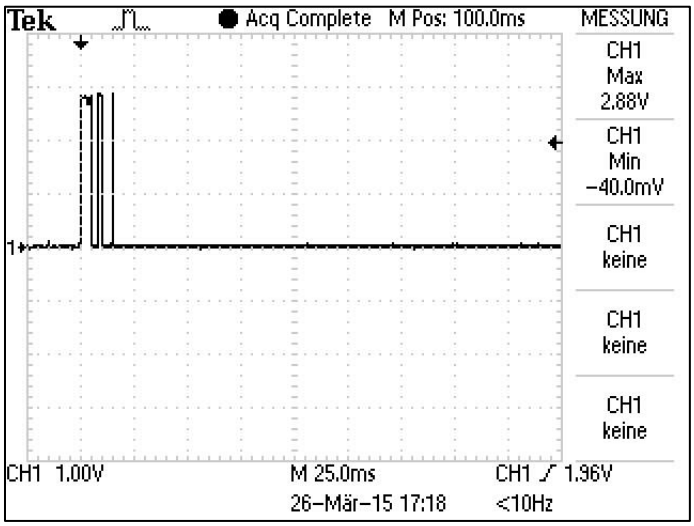

Abbildung 7: Raspicomm 25ms

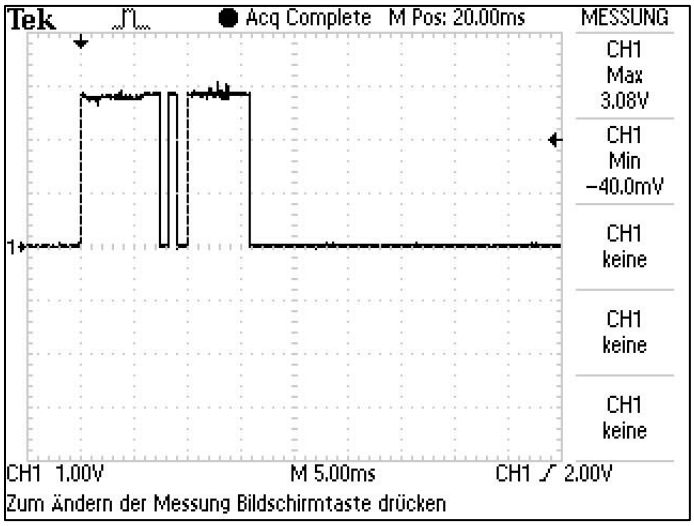

Abbildung 8: Raspicomm 5ms

In Abbildung 7 und Abbildung 8 sind die Singleshotaufnahmen der Kommunikationsplatine Raspicomm dargestellt. Dabei ist ersichtlich, dass die Differenzspannung lediglich dreimal auf HIGH wechselt in diesem Fall ca. 3V. Die Prozedur dauert ungefähr 15ms. Danach folgt nichts mehr. Bei mehrmaligen Versuchen wird festgestellt, dass genau das gleiche Signalbild nach wenigen Sekunden nochmals erscheint, vermutlich weil kein Busteilnehmer antwortet.

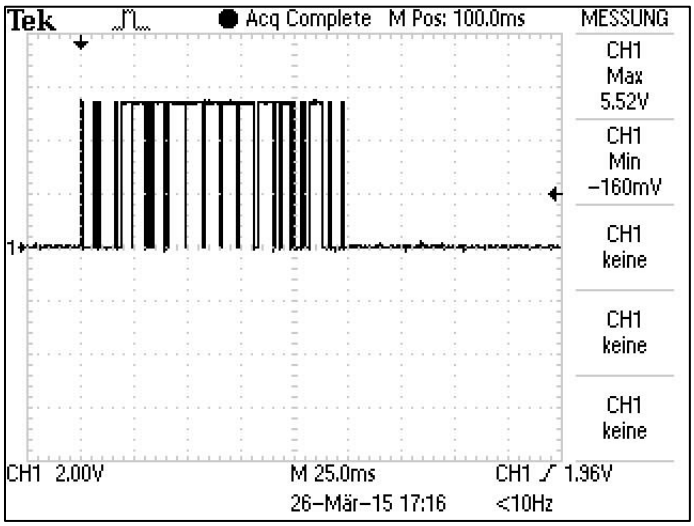

Abbildung 9: USB-RS 485 Wandler 25ms

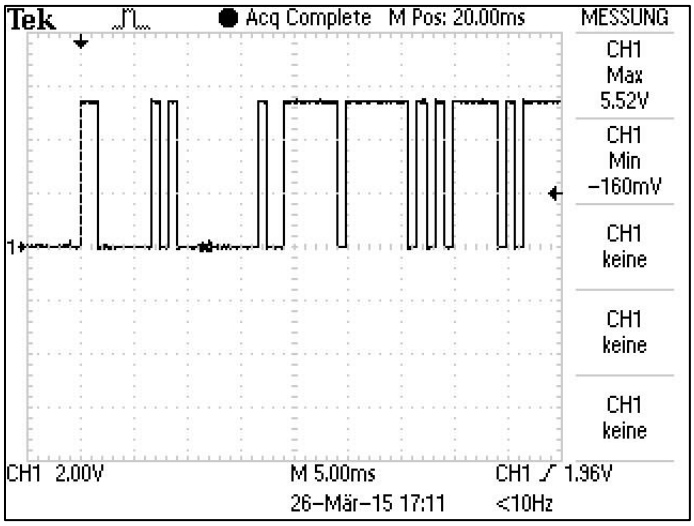

Abbildung 10: USB-RS 485 Wandler 5ms

In Abbildung 10 und Abbildung 9 sind die Singleshotaufnahmen des USB-RS 485 Wandlers dargestellt. Dabei ist zu erkennen, dass die Differenzspannung mehrmals zwischen HIGH und LOW wechselt. HIGH ist in diesem Fall ca. 5,5V. Die Prozedur dauert ungefähr 125ms. Bei mehrmaligen Versuchen wird ebenfalls festgestellt, dass genau das gleiche Signalbild nach wenigen Sekunden nochmals erscheint, vermutlich weil kein Busteilnehmer antwortet.

Vergleicht man die Signalverläufe der beiden Komponenten, so lassen sich deutliche Unterschiede feststellen in Bezug auf die gesamte Länge sowie auf die HIGH und LOW Phasen, was vermutlich dazu führt, dass der Sunny Island nicht auf die Anfrage der Raspicomm antwortet. Die unterschiedlichen Spannungen der HIGH Pegel kann laut der Definition zu RS 485 nicht ausschlaggebend sein, da beides mal die +/- 200mV erreicht werden. Die letztendliche Klärung warum die Raspicomm einen so abweichenden Signalverlauf hat, gibt es im Zuge dieser Projektarbeit leider nicht.

1.7 Balkenanzeige

Zum Abschluss dieses Teils der Projektarbeit wird die Vorlage eines Pythonskripts verwendet, welches die Visualisierung von Messwerten durch ein Balkendiagramm ermöglicht. Dabei werden folgende Werte angezeigt:

- SOC der Batterie in Prozent
- SOH der Batterie in Prozent
- die aktuelle Leistung an der AC Seite des Sunny Islands in Watt
- die Spannung an der DC Seite in Volt
- der Strom an der DC Seite in Ampere

Der Programmiercode von „SunnyIsland_V_1.py“ ist in Anhang 7.2.5 abgebildet, der Screenshot des Diagrammes ist in Abbildung 11 zu sehen.

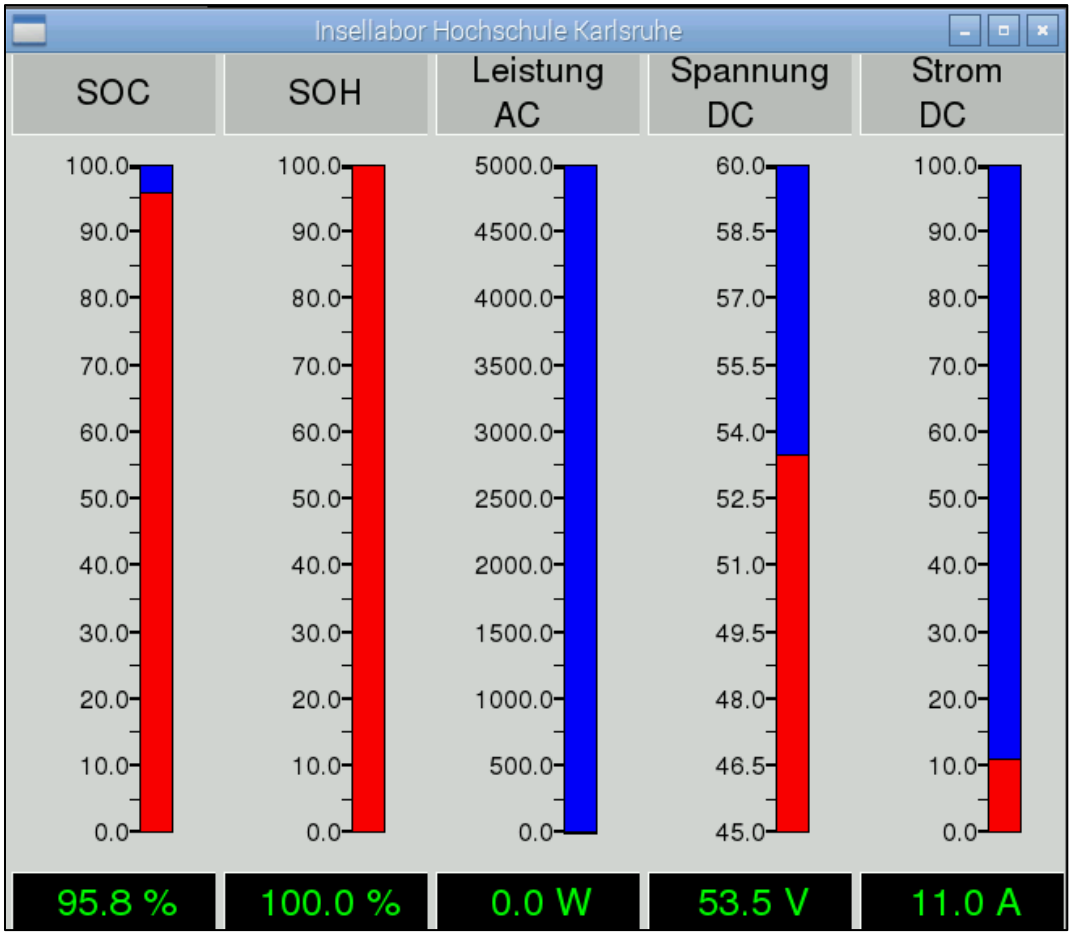

Abbildung 11: Balkenanzeige

2 Zusammenfassung und Ausblick

Während dieser Projektarbeit wurde es ermöglicht, sich als absoluter Neuling in ein Unixsystem einzuarbeiten und die Vor- und Nachteile davon hautnah kennenzulernen. Außerdem wurde das Programmieren in der Programmiersprache Python erlernt, sowie das Verständnis für serielle Schnittstellen erweitert in Bezug auf RS 232 und RS 485.

Beide Teilaufgaben, Sunny Island und Xtender, die bei dieser Projektarbeit gelöst wurden, zielen darauf ab von einer neuen Projektgruppe fortgesetzt zu werden. Hierbei wurden wesentliche Grundsteine gelegt, die ein intelligentes Management des Insellabors an der Hochschule Karlsruhe ermöglichen.
Ein denkbarer Ansatz wäre beispielsweise, beide genannten Teilprojekte auf einem System zu vereinen und eine große Anzeigetafel zu errichten, auf der ersichtlich ist, welche Betriebsdaten die beiden Wechselrichter gerade liefern. Außerdem können Netzdaten des öffentlichen Stromnetzes aus dem Internet abgerufen werden und dem Studer Xtender eine daraus berechnete Einspeiseleistungen vorgegeben werden.

3 Literaturverzeichnis

1. **amescon.** [Online] [Zitat vom: 19. April 2015.] http://www.amescon.com/media/5273/Raspicomm_Benutzerhandbuch.pdf.
2. **Studer.** http://www.studer-innotec.com. [Online] [Zitat vom: 22. 04 2015.] http://www.studer-innotec.com/upload/files/Technical%20specification%20-%20Xtender%20serial%20protocol%20-%20V1.3.1.pdf.
3. SMA. [Online] [Zitat vom: 31. März 2015.] http://www.sma-america.com/products/battery-inverters/sunny-island-4548-us-6048-us.html.
4. **Academy, SMA Solar.** *Drahtgebundene Kommunikation für kleine und mittlere PV-Anlagen.*
5. Wikipedia. [Online] [Zitat vom: 31. März 2015.] http://de.wikipedia.org/w/index.php?title=EIA-485&redirect=no#/media/File:RS-485_waveform.svg.
6. Digitus. [Online] [Zitat vom: 31. März 2015.] http://www.digitus.info/produkte/zubehoer/adapter-und-konverter/r-usb-seriell-adapter-usb-20-da-70157/.
7. **SMA.** *Sunny Island 5048 Installations- und Bedienungsanleitung.*

4 Abbildungsverzeichnis

5 Verzeichnis der Tabellen

6 Anhang auf CD

- Programm für Studer Xtender
  Datenblätter/Bedienungsanleitung Xtender

- Programme für SMA Sunny Island
  Datenblätter/Bedienungsanleitung Sunny Island

- Ausarbeitung

7 Anhang

7.1 Studer Xtender

7.1.1 Pythoncode Studer Xtender

```
#!/usr/bin/python
import serial
import struct
import numpy as np
import time
import binascii
import re
import math
import matplotlib.pyplot as plt
import random
#**************************************************************************
********

# Konfiguration serielle Schnittstelle
port = serial.Serial(
port='/dev/ttyAMA0',
baudrate=38400,
parity=serial.PARITY_EVEN,
stopbits=serial.STOPBITS_ONE,
bytesize=serial.EIGHTBITS,
timeout=2,
xonxoff=False,
rtscts=False,
dsrdtr=False
)
#**************************************************************************
********

#Visualisierug
dt=5
T=100
N=int(T/dt)+1
ymax=int(700)
ymin=int(0)
xl='Zeit in [s]'
#yl=[('Temperatur in '+cel+'C')]
yl=['Eingangsaktivleistung in [W]']
tl=[('Eingangsaktivleistung')]

def init_plot ():
    global a1, line
    a1=np.zeros(N)
    x=np.linspace(0,T,N)
    line, = plt.plot(x,a1,'r')
    plt.ylim([ymin,ymax])
    plt.yticks(range(ymin,ymax,100))
    plt.xlabel(xl)
```

```
    plt.ylabel(yl[0])
    plt.title(tl[0])
    plt.grid(b=True, which='both', color='0.65',linestyle=':')

def update_plot (k1):
    global a1, line
    a1=np.append(a1,k1)
    a1= np.delete(a1,0)
    line.set_ydata(a1)
    plt.draw()
    plt.pause(1.e-9)
#*****************************************************************
********

#Frames
#Die ersten 14 Bytes des frames "Schreiben" (konstant!)
frame_header_write =
'\xAA\x00\x01\x00\x00\x00\x65\x00\x00\x00\x0E\x00\x73\x79'
#Die ersten 14 Bytes des frames "Lesen" (konstant!)
frame_header_read =
'\xAA\x00\x01\x00\x00\x00\x65\x00\x00\x00\x0A\x00\x6F\x71'
#*****************************************************************
********

#Funktion frame_data(value)
#
#Aufgabe: Erstellen des mittleren teiles des gesammten Frames Schreiben
#Parameter:Uebergabewert=Stromwert;Rueckgabewert=mittlere Framesequenz
#
#Diese Funktion wandelt den uebergebenen Wert value(float)in einen Hex-
Wert
#Anschliessend wird der Hex-Wert in 4 Byte aufgeteilt. Jedes dieser
Bytes wird in ein
#Integer gewandelt und nach litle Endian in die Variablen var1-4
gespeicher und zurueck gegeben.
#
def frame_data(value):

    if value == 0:
        var1= 0x00
        var2= 0x00
        var3= 0x00
        var4= 0x00
    else:
        Strom_in_hex = hex(struct.unpack('<I',struct.pack('<f',value
))[0])
        #Hilfsstring fuer bytweise Aufteilung von Strom_in_hex
        string_1='0x'
        var1 = int(Strom_in_hex[0:4],16)
        var2 = int(string_1 + Strom_in_hex[4:6],16)
        var3 = int(string_1 + Strom_in_hex[6:8],16)
        var4 = int(string_1 + Strom_in_hex[8:10],16)

    frame_sequence = [0, 2, 2, 0, 243, 5, 0, 0, 5, 0, var4, var3, var2,
var1]
```

```
    return frame_sequence
#*********************************************************************
********

#Funktion check_sum(frame)
#
#Aufgabe:Berechnung der Checksumme der mittleren Framesequenz des
gesammten Frames "Schreiben"
#Parameter:Uebergabewert=mittlere Framesequenz;Rueckgabewert=Checksumme
als Array
#
#Aus dem uebergebenen Framesgement wird die Checksumme berechnet und in
den Parametern
#A und B abgespeichert. Diese Parameter werden in einem Array zurueck
gegeben.
def check_sum(frame):

    laenge = len(frame)

    A=255
    B=0
    i=0
    for i in range(laenge):
        A=(A+frame[i])%256
        B=(B+A)%256

    return [A, B]
#*********************************************************************
********

#Funktion frame_ges(a,b,c)
#
#Aufgabe: Zusammensetzen des gesammten Frames (schreiben des
Einspeisestrom) aus den einzelnen framesequenzen
#Parameter:Uebergabewert:frame_Data als Integer,checksumme als
Array,header_write als String
#          Rueckgabewert: kompletter Frame fuer Einspeisestrom als
String
#
#1.) wandelt die Framesequenz "frame_Data_int" von Integer in Charkter
#2.) Ggewandelte Charakter Elemnte zu einem String bilden
#3.) Gesammten Frame aus den 3 Framesequenzen zu einem String
zusammensetzen
def frame_ges(frame_Data_int,check_char,frame_header_write):
    frame_Data_chr = {}
    #he =Hilfsgroesse fuer Aufsummierung
    he=' '
    for i in range(14):
        frame_Data_chr[i] = chr((frame_Data_int[i]))

    for u in range(14):
        he=he+frame_Data_chr[u]

    he=he[1:]
    #Zusammensetzen des gesammten Frames
    frame_ges = frame_header_write + he + check_char[0]+check_char[1]
```

```
    return frame_ges
#*********************************************************************
********

#Funktion ON_OFF()
#
#Aufgabe: Durch einen Impuls auf den Fernsetuereingang {Nr.1545}, wird
das Relais fuer die Einspeisung aktiviert.
#(Dafuer muss die NR.1576 auf "ja" eingestellt sein)
def ON_OFF():
    port.close()
    port.open()
    wert =
'\xAA\x00\x01\x00\x00\x00\x65\x00\x00\x00\x0E\x00\x73\x79\x00\x02\x02\x
00\x09\x06\x00\x00\x05\x00\x01\x00\x00\x00\x18\xD6'
    port.write(wert)
    time.sleep(1)
    wert =
'\xAA\x00\x01\x00\x00\x00\x65\x00\x00\x00\x0E\x00\x73\x79\x00\x02\x02\x
00\x09\x06\x00\x00\x05\x00\x00\x00\x00\x00\x17\xD2'
    port.write(wert)

#*********************************************************************
********

#Funktion Eingabe(input_I_AC)
#
#Aufgabe:Eingelesenen Stromwert duch aufrufen weiterer Unterprogramme
zu einem Frame zusammenbauen
#        um diesen, am Ende der Funktion dem Wechselrichter zu senden
#Parameter:Uebergabewert=String
#          Rueckgabewert=String
def Eingabe(input_I_AC):

    try:
        #Eingelesene Wert (Einspeisestrom)wird in eine float zahl
gewandelt
        input_I_AC_float = float(input_I_AC)
        #Rueckgabe-Frameteil wird in frame_Data_int gespeichert
        frame_Data_int = frame_data(input_I_AC_float)
        #Berechnete Checksumme(Array) wird in check gespeichert
        check=check_sum(frame_Data_int)
        #Checksumme(Integer)wird in ASCII gewandelt
        check_char=[chr(check[0]), chr(check[1])]
        #Rueckgabewert(fertiger Frame "schreiben")wird in frame
gespeichert
        frame = frame_ges(frame_Data_int,check_char,frame_header_write)
        #fertigr frame mit dem Einspeisetrom wird an Studer gesendet
        port.close()
        port.open()
        port.write(frame)
        time.sleep(2)

        return 'ok'

    except ValueError:
```

```
        return 'False'
#*********************************************************************
********

#Funktion I_AC()
#
#Aufgabe:Aktueller Einspeisestrom I_AC auslesen
#Parameter:Rueckgabewert= Einspeisestrom (AC) als float
#
#Frame fuer Anfrage des aktuellen Einspeisestrom wird an Studer
gesendet.Antwortframe
#von Studer wird empfangen und die Bytes 27-24 (little Endian !)mit der
Information werden in Variablen
#abgespeichert.Die gespeicherten ASCII-Elemente werden anschliessend in
Hex gewandelt.
#In den naechsten Schritten werden  die einzelnen Hexwerte zu einer
Hexzahl zusammengesetzte und in
#eine float Zahl gewandelt und zurueck gegeben.
def I_AC():

    port.close()
    port.open()
    #Frame fuer Anfrage Einspeisestrom I_AC     Object_id=3012
    write =
'\xAA\x00\x01\x00\x00\x00\x65\x00\x00\x00\x0A\x00\x6F\x71\x00\x01\x01\x
00\xC4\x0B\x00\x00\x01\x00\xD1\xD8'
    port.write(write)
    read_string = port.read(30)
    #Zeitstempel der Messung
    t=time.localtime()
    ti=str(t[3])+':'+str(t[4])+':'+str(t[5])+'-
'+str(t[2])+'.'+str(t[1])+'.'+str(t[0])
    #Ueberpruefung korrekter laenge

    if len(read_string)==30:

        #Zurueckgelieferter Strom aus den bytes 24-27
        var_1 = read_string[27]
        var_2= read_string[26]
        var_3 = read_string[25]
        var_4 = read_string[24]

        string=var_1+var_2+var_3+var_4
        help=struct.unpack(">f",string)
        float_I_AC=help[0]

        return [float_I_AC,ti]
    else:
        return ["Fehler lesen Einspeisestrom",ti]
#*********************************************************************
********

#Funktion P_IN()
#Parameter:Rueckgabewert= Einspeiseleistung als float und Zeitstempel
def P_IN():
```

```
    port.close()
    port.open()
    #Frame fuer Anfrage Eingangsaktivleistung   Object_id=3000
    write =
'\xAA\x00\x01\x00\x00\x00\x65\x00\x00\x00\x0A\x00\x6F\x71\x00\x01\x01\x
00\x41\x0C\x00\x00\x01\x00\x4F\xCB'
    port.write(write)
    read_string = port.read(30)
    #Zeitstempel der Messung
    t=time.localtime()
    ti=str(t[3])+':'+str(t[4])+':'+str(t[5])+'-
'+str(t[2])+'.'+str(t[1])+'.'+str(t[0])
    if len(read_string)==30:
        #Zurueckgelieferter Spannung aus den bytes 24-27
        var_1 = read_string[27]
        var_2= read_string[26]
        var_3 = read_string[25]
        var_4 = read_string[24]

        string=var_1+var_2+var_3+var_4
        help=struct.unpack(">f",string)
        P=help[0]
        P_W=abs(P)*1000

        return [P_W,ti]
    else:
        return ["Fehler lesen Leistung",ti]
#*****************************************************************
********

# Funktion V_BATT()
#Parameter:Rueckgabewert= Spannung an Batterie als float und aktuellen
Zeitstempel Array
def V_Batt():

    port.close()
    port.open()
    #Frame fuer Anfrage Einspeisestrom U_Batterie    Object_id=3000
    write =
'\xAA\x00\x01\x00\x00\x00\x65\x00\x00\x00\x0A\x00\x6F\x71\x00\x01\x01\x
00\xB8\x0B\x00\x00\x01\x00\xC5\x90'
    port.write(write)
    read_string = port.read(30)
    #Zeitstempel der Messung
    t=time.localtime()
    ti=str(t[3])+':'+str(t[4])+':'+str(t[5])+'-
'+str(t[2])+'.'+str(t[1])+'.'+str(t[0])

    if len(read_string)==30:
        #Zurueckgelieferter Spannung aus den bytes 24-27
        var_1 = read_string[27]
        var_2= read_string[26]
        var_3 = read_string[25]
        var_4 = read_string[24]

        string=var_1+var_2+var_3+var_4
```

```
        help=struct.unpack(">f",string)
        float_U_Batt=help[0]

        return [float_U_Batt,ti]
    else:
        return ["Fehler lesen Batteriespanung",ti]

# Funktion Stat_relais()
#Parameter:Rueckgabewert= 1 oder 0
#
#Funktion:Abfragen der aktuellen Stellung des des Fernsteuereingangs
{1545}
def Stat_relais():


        port.close()
        port.open()
        wert =
'\xAA\x00\x01\x00\x00\x00\x65\x00\x00\x00\x0A\x00\x6F\x71\x00\x01\x01\x
00\xD6\x0B\x00\x00\x01\x00\xE3\x44'
        port.write(wert)
        string = port.read(30)
        var_4 = string[24]

        return ord(var_4)

#*****************************************************************
********


#*****************************************************************
********
#       main
#
#*****************************************************************
********

while 1:

    start=raw_input("Zum starten 's' eingeben und mit Enter
bestaetigen\n>")
    if start == 's':

        #Abfragen aktueller Stand Fernsteuereingang. muss zu Beginn auf
*Geschl. stehen ==0
        stellung=Stat_relais()

        if stellung==1:

            ON_OFF()

        #Initialisierungen
        plt.close()
        init_plot()
        update_plot (0)
```

```
        #Init:Einspeisesrom auf 0A einstellen und Schalter/Relais fuer
Einspeisung schliessen
        #5 sek. Wartezeit fuer Netzsynchronisation
        Eingabe('0')
        ON_OFF()
        time.sleep(5)

        #Datei mit Einspeisewerte oeffnen und lesen
        Datei = open("dat.txt","r")
        var_1 = Datei.read()
        #Aus der eingelesenen Datei "\n" entfernen und alle Elemente
nach ";" in ein Array aufteilen
        var_2 = var_1.replace("\n", "")
        var_3 = var_2.split(";")

        #Vektor mit der laenge der Elemente des Arrays anlegen -1
        count_1 = range(len(var_3)-1)
        #Vektor
        count_2 = range(4)

        #Datei fuer schreiben der Messwerte oeffnen
        output = open("Output.txt","w")
        #Header Ausgabedatei
        output.write("Projektarbeit - 2015"+"\n\n")
        output.write("Einspeisung [W]"+";"+"Zeitstempel
Leistung"+";"+"Einspeisestrom [A]"+";"+"Zeitstempel
Strom"+";"+"Batteriespannung[V]"+";"+"Zeitstempel
Batteriespannung"+"\n\n")


#*********************************************************************

        #Geschachtelte Schleife fuer Hauptprogramm

        #Aeussere Schleife fuer Messwertaktualisierung
        for c in count_1:
            #Messwert (Einspeisestrom wird eingelesen) wird aus dem
Array eingelesen
            flag=Eingabe(var_3[c])

            if flag =='ok':
            #Innere Schleife fuer Messwerterfassung und Speicherung
                for i in count_2:

                    t0=time.time()
                    Spannung=V_Batt()
                    time.sleep(2)
                    Strom=I_AC()
                    time.sleep(2)
                    Leistung=P_IN()
##                    ZS=time.localtime()

output.write(str(Leistung[0])+";"+Leistung[1]+";"+str(Strom[0])+';'+Str
om[1]+';'+str(Spannung[0])+';'+Spannung[1]+"\n")
```

```
                    if type(Leistung[0]) == float:

                        #Visu
                        update_plot (Leistung[0])
                        t1=time.time()
                        dt_ist=t1-t0
                        tw=dt-dt_ist          # restliche Wartezeit
##                        print(tw)
                        if tw<=0:             # Falls Wartezeit negativ
wird
                            tw=0
                        time.sleep(tw) # tw warten
                    else:
                        time.sleep(dt)
            else:
                output.write('Fehler bei Lesen von Einspeisestrom aus
Datei'+'\n')
                time.sleep(1)


#*********************************************************************

        #Schliessen der zu schreibenden Datei
        output.close()
        #oeffnen des Schalters/Relais
        ON_OFF()
    else:
        break
```

7.2 Sunny Island

7.2.1 C Code YASDISHELL

```
// Projektarbeit Daten aus SMA Sunny Island 5048 auslesen
//
// Hochschule Karlsruhe Sommersemester 2015



/************************************************************************
***
*   I N C L U D E
************************************************************************
**/
#include <stdio.h>

#ifdef __cplusplus
extern "C" {
#endif

#include "os.h"
#include "smadef.h"
#include "chandef.h"
#include "libyasdi.h"
#include "libyasdimaster.h"
#include "smadata_layer.h"
#include "tools.h"

#ifdef __cplusplus
}
#endif


/************************************************************************
****
*   G L O B A L
************************************************************************
***/

#define DEVMAX 30       //For simplicity we use here maximal 30 devices
...
#define MAXDRIVERS 10   //... and 10 YASDI Bus drivers
#define EXPECT_CHAN_CNT 300  //lets say that we expect 300 channels in
max
                             //for one device


/************************************************************************
****
*   L O C A L   F U N C T I O N S
************************************************************************
***/
```

```
void PrintDevList( void );
void PrintChannelValues(TChanType chanType);
int DoStartDetection( int DevCnt );



/*********************************************************************
****
   Description    : print out device list
   Parameter      : (none)
   Return-Value   : (none)
   Changes        : Author, Date, Version, Reason

*****************************************************
                    PRUESSING, 02.07.2001, 1.0, Created
*********************************************************************
***/
void PrintDevList( void )
{
   DWORD i;
   DWORD DevHandles[DEVMAX];
   DWORD DevCount=0;
   char NameBuffer[50];


   /* get all device handles...*/
   DevCount    = GetDeviceHandles(DevHandles, DEVMAX );
   if (DevCount)
   {
      printf("-----------------------------------------\n");
      printf("Device handle | Device Name  \n");
      printf("-----------------------------------------\n");

      for(i=0;i<DevCount;i++)
      {
         /* get the name of this device */
         GetDeviceName(DevHandles[i], NameBuffer, sizeof(NameBuffer)-
1);
         printf("   %3lu        | '%s'\n",
                (unsigned long)DevHandles[i],
                NameBuffer);
      }
      printf("-----------------------------------------\n\n");
   }
   else
   {
      printf("Sorry, no devices currently available...\n\n");
   }
}

/*********************************************************************
****
   Description    : print out all channels of a device
   Parameter      : ChanMask: the filter mask (see SMA-Data-Description)
   Return-Value   : (none)
   Changes        : Author, Date, Version, Reason
```

```
*********************************************************
                    PRUESSING, 02.07.2001, 1.0, Created
                    , 19.03.2015, 1.2, Anpassung fuer Projektarbeit
*********************************************************************
***/
void PrintChannelValues(TChanType chanType)
{
   int res;

   DWORD ChanHandle[EXPECT_CHAN_CNT];
   char ChanName[50];
   char DevName[50];
   char cUnit[17];
   int i;
   int ChanCount;
   int DevHandle;
   double Value;
   char TextValue[30];
   DWORD MaxValueAge = 5; /* maximum age of the channel value in
seconds...*/

   printf("Device handle: ");
   DevHandle = 1;
   ChanCount = GetChannelHandlesEx((DWORD)DevHandle, ChanHandle,
EXPECT_CHAN_CNT, chanType);
   GetDeviceName(DevHandle, DevName, sizeof(DevName)-1);
   printf( "Device '%s' has %d %s%s channels:\n",
           DevName,
           ChanCount,
           (chanType == TESTCHANNELS) ? "(Test)" : "",
           (chanType == PARAMCHANNELS) ? "Parameter" : "Spot" );

   printf("Reading channel values, please wait...\n");

   printf("-------------------------------------------------------------
\n");
   printf("Channel handle |    Channel name    | Channel value (Unit)
|\n");
   printf("-------------------------------------------------------------
\n");

   for(i=0;i<ChanCount;i++)
   {
      GetChannelName(ChanHandle[i],ChanName, sizeof(ChanName)-1);

      //get the channel unit?
      cUnit[0]=0;
      GetChannelUnit(ChanHandle[i], cUnit, sizeof(cUnit)-1);

      /* Get channel value... */
      res = GetChannelValue(ChanHandle[i], DevHandle, &Value,
TextValue,
                            sizeof(TextValue)-1, MaxValueAge);
      if(res==0)
      {
```

```
         /* Status texts?*/
         if (strlen( TextValue )==0)
            sprintf( TextValue,"%.3f", Value);
      }
      else
      {
         printf("Error reading channel value....error code=%d\n",res);
         break;
      }

     printf("     %3lu       | '%16s' | '%s' %c%s%c \n", (unsigned
long)ChanHandle[i], ChanName,
            TextValue,
            strlen(cUnit) > 0 ? '(' : ' ',
            cUnit,
            strlen(cUnit) > 0 ? ')' : ' ');
   }
}



/*********************************************************************
****
   Description   : Start device detection (find all devices)
   Parameter     : (none)
   Return-Value  : (none)
   Changes       : Author, Date, Version, Reason

*****************************************************
                  Pruessing, 02.07.2001, 1.0, Created
                  Pruessing, 01.12.2002, 1.1, Using new API Funktion
                                              "DoMasterCmdEx()"
                  , 19.03.2015, 1.2, Anpassung fuer Projektarbeit
*********************************************************************
***/


//Start device detection synchrone (blocks until device detection is
done)
int DoStartDetection( int DevCnt )
{
   int iErrorCode;

   DevCnt = 1; //Nur ein Wechselrichter soll gesucht werden

   /* Do start searching devices... */
   iErrorCode = DoStartDeviceDetection( DevCnt, TRUE /*blocking*/ );
   switch(iErrorCode)
   {
      case YE_OK:
         break;

      case YE_DEV_DETECT_IN_PROGRESS:
         printf("Sorry, but there is already an running device
detection...\n");
         return 0; //currently not possible...
```

```
      case YE_NOT_ALL_DEVS_FOUND:
         printf("Sorry, but device was not found.\n");
         return 1;

      default:
         printf("mhhh....\n");
   }

   PrintDevList();
   return 0;
}



//! Receive Device Detection Events from YASDI...
void cbDeviceDetectionEvent(TYASDIDetectionSub event, DWORD
DeviceHandle, DWORD param1 )
{
   char NameBuffer[30]={0};

   //resolve the device handle and get the device name if possible...
   GetDeviceName(DeviceHandle, NameBuffer, sizeof(NameBuffer)-1);

   switch(event)
   {
      case YASDI_EVENT_DEVICE_ADDED:
         printf("New device found: '%s'\n", NameBuffer);
         break;

      case YASDI_EVENT_DEVICE_REMOVED:
         printf("Device was removed: '%s'\n", NameBuffer);
         break;

      case YASDI_EVENT_DEVICE_SEARCH_END:
         printf("\nAsync Device Detection finished. %d devices
available.\n", (int)DeviceHandle);
         PrintDevList();
         break;

      case YASDI_EVENT_DOWNLOAD_CHANLIST:
      {
         BYTE percentDone = param1;
         printf(".");
         if (percentDone == 100)
            printf("\n");
         break;
      }

      default:
         printf("unknown yasdi (0x%2x) event...\n", event);
         break;
   }

   return;
```

```
}

void cbLongDataTransportEvent(BYTE percentDone)
{
   printf(".");
}

/*****************************************************************
****
   Description    : This is the Start ("main")
                    I have some trouble with the include files. So the
                    real "main" call this function from a different
place...
   Parameter      : (none)
   Return-Value   : (none)
   Changes        : Author, Date, Version, Reason

*********************************************************
                    PRUESSING, 02.07.2001, 1.0, Created
                    , 19.03.2015, 1.2, Anpassung fuer Projektarbeit
*****************************************************************
***/
int main(int argv, char **argc)
{
   DWORD i;
   DWORD dDriverNum;
   DWORD Driver[MAXDRIVERS];
   int Error=0;
   char DriverName[30];
   BOOL bOnDriverOnline = FALSE; //Is at least one driver online?
   char IniFile[]="yasdi.ini";

   printf("Projektarbeit Patrick Lais und Constantin Reiss\n");
   printf("Hochschule Karlsruhe EITM Sommersemester 2015\n");
   printf("\n");

   if (argv>=2)
   {
      strcpy(IniFile,argc[1]);
   }

   /* init Yasdi- and Yasdi-Master-Library */
   if (0 > yasdiMasterInitialize(IniFile, &dDriverNum))
   {
       printf("ERROR: YASDI ini file was not found or is
unreadable!\n");
       return 0;
   }

   /* I want to be informed if new devices are inserted or removed...
   ** Insert callbac function...*/
   yasdiMasterAddEventListener( cbDeviceDetectionEvent,
YASDI_EVENT_DEVICE_DETECTION );


   /* get List of all supported drivers...*/
```

```
   dDriverNum = yasdiMasterGetDriver( Driver, MAXDRIVERS );

   /* Switch all drivers online (you should only do one of them
online!)...*/
   for(i=0;i<dDriverNum;i++)
   {
      /* The name of the driver */
      yasdiGetDriverName(Driver[i], DriverName, sizeof(DriverName));

      printf("Switching driver '%s' on...", DriverName);

      if (yasdiSetDriverOnline( Driver[i] ))
      {
         printf("success\n");
         bOnDriverOnline = TRUE;
      }
      else
         printf("false\n");
   }
   printf("\n");

   //Check that at least one driver is online...
   if (FALSE == bOnDriverOnline)
   {
      printf("WARNING: No drivers are online! YASDI can't communicate
with devices!\n");
      Error = 1;
   }
   else
   {
       Error = DoStartDetection( 0 );//Automatisch Wechselrichter
suchen
   }

   if (Error != 1)
   {
       PrintChannelValues( SPOTCHANNELS ); /* only all spot channels */
   }

   /* Shutdown all yasdi drivers... */
   for(i=0;i<dDriverNum;i++)
   {
      yasdiGetDriverName(Driver[i], DriverName, sizeof(DriverName));
      printf("Switching driver '%s' off...\n", DriverName);
      yasdiSetDriverOffline( Driver[i] );
   }

   /* Shutdown YASDI..., bye, bye */
   yasdiMasterShutdown();
   return 0;
}
```

7.2.2 INI-File

```
[DriverModules]
Driver0=yasdi_drv_ip
Driver1=libyasdi_drv_serial.so


[COM1]
Device=/dev/ttyUSB0
Media=RS485
Baudrate=1200
Protocol=SMANet

#[IP1]
#Protocol=SMANet
#Device0-192.168.8.100

#[Misc]
#DebugOutput=/dev/stderr
```

7.2.3 Python Code (SunnyIsland_V_0.py)

```
#!/usr/bin/env python

import re
import os
import time

Array = []

def ini():
    #Textfile oeffnen
    Datei = open("Rohdaten_SunnyIsland.txt","r")

    #einzelne Zeilen einlesen
    for line in Datei:
        Array.append(line)

    #Datei schliessen
    Datei.close()

    #Fehlerhandling (wenn nicht der komplette Parametersatz eingelesen
wurde, ist die Datei kuerzer als 159 Zeilen)
    if len(Array)<159:
        return 1


def string_Programm(value):

    #Arrayelement in Variable kopieren
    alleDaten = (Array[value])

    #  " wird ersetzt durch nichts (wird weggemacht)
    alleDaten_optimiert = alleDaten.replace('"', "")

    #Zahlen und Buchstaben werden aus gesamten String rausgeholt
    alleDaten_optimiert = re.sub(r'(\D) (\W+)'," ",alleDaten_optimiert)

    #  n] wird ersetzt durch nichts (wird weggemacht)
    alleDaten_optimiert = alleDaten_optimiert.replace('n]', "")

    #einzelne Elemente werden in Array gespeichert
    result = alleDaten_optimiert.split(" ")

    return result


def value_Programm(number):

    #Wenn der gewuenschte Parameter groesser als 251 ist, wird 4
    #abgezogen, damit die Zeilennummer im Textfile wieder stimmt.
    #Das muss gemacht werden, weil Parameter 252 bis 255 nicht
vorhanden sind.
    if (number>251):
        number = (number-4)
```

```
    # 169 werden abgezogen, damit die Zeilennummer im Textfile stimmt
    # auf diese Zahl kommt man, wenn man beruecksichtigt, dass im
Textfile ganz oben
    # zuerst noch andere Dinge stehen und weil der Parametersatz nicht
bei 1 anfaengt,
    # sondern bei 186 .

    value = number-169

    return value




#main

while 1:

    #Pfad wechseln
    os.chdir("/home/pi/Desktop/FINAL_Sunny/yasdi/projects/generic-
cmake")

    #Daten aus SunnyIsland auslesen und in Textfile schreiben
    a=os.system('./yasdishell>Rohdaten_SunnyIsland.txt')

    #Daten aus Textfile in Array einlesen
    Error=ini()

    if Error:
        value=0
        #Fehlerbeschreibung aus Textfile im Terminal ausgeben
        while value < len(Array):
            result = (Array[value])
            print(result)
            value=value+1
        #Skript beenden
        quit()

    else:
        #Die Parameter 252 bis 255 gibt es nicht!!!
        #HIER muss der gewuenschte Parameter eingetragen werden!
        number = 192
        #Da es 252 bis 255 nicht gibt, muss "number" angepasst werden
        value = value_Programm(number)

        #zerstueckelter String
        result = string_Programm(value)

        #test-prints

        #ganzer String
        print(result)

        #Parameternummer
```

```
        print(result[1])

        #Parameterbezeichnung
        print(result[2])

        #Wert
        print(result[3])

        #Einheit
        if result[4]=="":
            print("Keine Einheit")
        else:
            print(result[4])


    # Array leer machen
    del Array[0:len(Array)]
```

7.2.4 Rohdaten

```
Projektarbeit
Hochschule Karlsruhe EITM Sommersemester 2015

Switching driver 'COM1' on...success

New device found: 'SI5048EI SN:1260008600'
-----------------------------------------
Device handle | Device Name
-----------------------------------------
     1        | 'SI5048EI SN:1260008600'
-----------------------------------------

Device handle: Device 'SI5048EI SN:1260008600' has 142 Spot channels:
Reading channel values, please wait...
-------------------------------------------------------------
Channel handle |    Channel name    | Channel value (Unit) |
-------------------------------------------------------------
     186       | '             Msg' | '0.000'
     187       | '             Soh' | '100.000' (%)
     188       | '     TotInvPwrAt' | '0.000' (kW)
     189       | '       TotInvCur' | '0.900' (A)
     190       | '     TotInvPwrRt' | '-0.200' (kVAr)
     191       | '          BatSoc' | '100.000' (%)
     192       | '          BatVtg' | '52.200' (V)
     193       | '      BatChrgVtg' | '53.797' (V)
     194       | '        AptTmRmg' | '0.000' (hhmmss)
     195       | '       TotBatCur' | '-3.800' (A)
     196       | '          BatTmp' | '22.100' (degC)
     197       | '        RmgTmFul' | '12.000' (d)
     198       | '        RmgTmEqu' | '178.000' (d)
     199       | '       BatSocErr' | '0.000' (%)
     200       | '         GnRmgTm' | '0.000' (hhmmss)
     201       | '        InvPwrAt' | '0.000' (kW)
     202       | '    InvPwrAtSlv1' | '0.000' (kW)
     203       | '    InvPwrAtSlv2' | '0.000' (kW)
     204       | '    InvPwrAtSlv3' | '0.000' (kW)
     205       | '           InvVtg' | '230.000' (V)
     206       | '      InvVtgSlv1' | '0.000' (V)
     207       | '      InvVtgSlv2' | '0.000' (V)
     208       | '      InvVtgSlv3' | '0.000' (V)
     209       | '           InvCur' | '0.900' (A)
     210       | '      InvCurSlv1' | '0.000' (A)
     211       | '      InvCurSlv2' | '0.000' (A)
     212       | '      InvCurSlv3' | '0.000' (A)
     213       | '           InvFrq' | '50.000' (Hz)
     214       | '        InvPwrRt' | '-0.200' (kVAr)
     215       | '    InvPwrRtSlv1' | '0.000' (kVAr)
     216       | '    InvPwrRtSlv2' | '0.000' (kVAr)
     217       | '    InvPwrRtSlv3' | '0.000' (kVAr)
     218       | '        ExtPwrAt' | '0.000' (kW)
     219       | '    ExtPwrAtSlv1' | '0.000' (kW)
     220       | '    ExtPwrAtSlv2' | '0.000' (kW)
     221       | '    ExtPwrAtSlv3' | '0.000' (kW)
     222       | '          ExtVtg' | '2.800' (V)
```

```
223       | '      ExtVtgSlv1' | '0.000' (V)
224       | '      ExtVtgSlv2' | '0.000' (V)
225       | '      ExtVtgSlv3' | '0.000' (V)
226       | '          ExtCur' | '0.300' (A)
227       | '      ExtCurSlv1' | '0.000' (A)
228       | '      ExtCurSlv2' | '0.000' (A)
229       | '      ExtCurSlv3' | '0.000' (A)
230       | '          ExtFrq' | '0.000' (Hz)
231       | '        ExtPwrRt' | '0.000' (kVAr)
232       | '    ExtPwrRtSlv1' | '0.000' (kVAr)
233       | '    ExtPwrRtSlv2' | '0.000' (kVAr)
234       | '    ExtPwrRtSlv3' | '0.000' (kVAr)
235       | '     TotExtPwrAt' | '0.000' (kW)
236       | '       TotExtCur' | '0.300' (A)
237       | '     TotExtPwrRt' | '0.000' (kVAr)
238       | '       TotLodPwr' | '0.000' (kW)
239       | '         GdRmgTm' | '0.000' (hhmmss)
240       | '             Pac' | '0.000' (kW)
241       | '        Firmware' | '7.203'
242       | '             Iac' | '0.900' (A)
243       | '             Vac' | '230.000' (V)
244       | '             Fac' | '50.000' (Hz)
245       | '        ChpPwrAt' | '0.000' (kW)
246       | '        ChpRmgTm' | '0.000' (hhmmss)
247       | '     ChpStrRmgTm' | '0.000' (hhmmss)
248       | '       Sic1PvPwr' | '0.000' (W)
249       | '       Sic2PvPwr' | '0.000' (W)
250       | '       Sic3PvPwr' | '0.000' (W)
251       | '       Sic4PvPwr' | '0.000' (W)
256       | '     TotSicPvPwr' | '0.000' (W)
257       | '    TotSicBatCur' | '0.000' (A)
258       | '    TotMccLodPwr' | '0.000' (kW)
259       | '      TotPvPwrAt' | '0.000' (kW)
260       | '     TotLodPwrAt' | '0.000' (kW)
261       | '    SlfCsmpPwrAt' | '0.000' (kW)
262       | '   SlfCsmpIncPwr' | '0.000' (kW)
263       | '   BatCpyThrpCnt' | '0.000'
264       | '     GdCsmpPwrAt' | '0.000' (kW)
265       | '     GdFeedPwrAt' | '0.000' (kW)
266       | '           PacPV' | '0.000' (kW)
267       | '      PacFeed-In' | '0.000' (kW)
268       | '  PacConsumption' | '0.000' (kW)
269       | '           kWhPV' | '0.000' (kWh)
270       | '           FwVer' | '7.203'
271       | '           OnTmh' | '31063.000' (h)
272       | '          FwVer2' | '7.200'
273       | '        EgyCntIn' | '22.300' (kWh)
274       | '       EgyCntOut' | '221.900' (kWh)
275       | '        EgyCntTm' | '24940.800' (h)
276       | '        GnEgyCnt' | '0.000' (kWh)
277       | '         GnEgyTm' | '0.000' (h)
278       | '         GnOpTmh' | '0.000' (h)
279       | '        GnStrCnt' | '0.000'
280       | '      GdEgyCntIn' | '0.000' (kWh)
281       | '     GdEgyCntOut' | '0.000' (kWh)
282       | '        GdEgyTmh' | '0.000' (h)
```

```
     283        | '          GdOpTmh' | '0.000' (h)
     284        | '        GdCtcCnt' | '0.000'
     285        | '           TotTmh' | '0.000' (h)
     286        | '    Sic1EgyCntIn' | '0.000' (kWh)
     287        | '    Sic2EgyCntIn' | '0.000' (kWh)
     288        | '    Sic3EgyCntIn' | '0.000' (kWh)
     289        | '    Sic4EgyCntIn' | '0.000' (kWh)
     290        | '  TotSicEgyCntIn' | '0.000' (kWh)
     291        | ' Sic1TdyEgyCntIn' | '0.000' (kWh)
     292        | ' Sic2TdyEgyCntIn' | '0.000' (kWh)
     293        | ' Sic3TdyEgyCntIn' | '0.000' (kWh)
     294        | ' Sic4TdyEgyCntIn' | '0.000' (kWh)
     295        | ' TotSicDyEgyCntI' | '0.000' (kWh)
     296        | '   Serial Number' | '1260008600.000'
     297        | '      E-Total-In' | '22.300' (kWh)
     298        | '         E-Total' | '221.900' (kWh)
     299        | '            h-On' | '31063.000' (h)
     300        | '    TotLodEgyCnt' | '0.000' (kWh)
     301        | '   SlfCsmpIncEgy' | '0.000' (kWh)
     302        | '      SlfCsmpEgy' | '0.000' (kWh)
     303        | '   SlfCsmpIncTdy' | '0.000' (kWh)
     304        | '    GdCsmpEgyTdy' | '0.000' (kWh)
     305        | '      kWhFeed-In' | '0.000' (kWh)
     306        | '  kWhConsumption' | '0.000' (kWh)
     307        | '    GdFeedEgyTdy' | '0.000' (kWh)
     308        | '             Adr' | 'Master'
     309        | '          OpStt' | 'Operating'
     310        | '       OpSttSlv1' | '---'
     311        | '       OpSttSlv2' | '---'
     312        | '       OpSttSlv3' | '---'
     313        | '         CardStt' | 'OutOfSpace'
     314        | '            Prio' | 'Ina'
     315        | '       BatChrgOp' | 'Float'
     316        | '          AptPhs' | 'Off'
     317        | '        GnDmdSrc' | 'None'
     318        | '           GnStt' | 'Off'
     319        | '        InvOpStt' | 'Run'
     320        | '    InvOpSttSlv1' | '---'
     321        | '    InvOpSttSlv2' | '---'
     322        | '    InvOpSttSlv3' | '---'
     323        | '         Rly1Stt' | 'Off'
     324        | '         Rly2Stt' | 'On'
     325        | '         GnRnStt' | 'Off'
     326        | '            Mode' | 'Operation'
     327        | '           Error' | '-----'
     328        | '          ChpStt' | 'Idle'
     329        | '      PvGdConStt' | '---'
     330        | '     LodGdConStt' | '---'
     331        | '       BatMntStt' | 'Off'
Switching driver 'COM1' off...
```

7.2.5 Python Code Balkenanzeige (SunnyIsland_V_1.py)

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-


from Tkinter import *
import time
import re
import os


Array = []

master = Tk()
master.title("Insellabor Hochschule Karlsruhe")
class bmeter:
      ueberschrift  = ''
      hoehe         = 550
      breite        = 0.25*hoehe
      rand          = breite/20
      rhoehe        = hoehe-rand-2
      rbreite       = breite-rand-2
      bw            = rand*1.5
      scalafont     = 'arial'
      wertfont      = '("digital-7",20)'
      fontsize      = 12
      bgcolor       = 'lightgray'
      fgcolor       = 'gray'
      zcolor        = 'white'
      barcolor      = 'red'
      nscale        = 20
      mscale        = 2
      einheit       = ''
      ymin          = 0.0
      ymax          = 100.0
      label=''
      einheit=''

def barmeter_init(container,bm,messwert):
      w = Canvas(container,
           width=bm.rbreite,
           height=bm.rhoehe)
      w.pack()

      x1            = bm.breite*0.75
      y1            = bm.hoehe*0.875
      x0            = bm.breite*0.6
      y0            = bm.hoehe*0.125
      w.create_rectangle(x0,y0,x1,y1,outline='black',fill='blue')
      sl=0.5*bm.fontsize
      yt=(y1-y0)/bm.nscale
      x1=x0-sl

 # Skalierungsstriche
      iscale=0
```

```
        while iscale<=bm.nscale:
            y1=round(y0+iscale*yt)
            w.create_line(x0,y1,x1,y1,fill='black',width=1)
            iscale=iscale+1
        iscale=0
        while iscale<=bm.nscale:
            y1=round(y0+iscale*yt)
            if iscale == 0:
                y1=round(y0+iscale*yt)+2
            w.create_line(x0,y1,x1,y1,fill='black',width=2)
            iscale=iscale+bm.mscale
# Skalierungswerte
        iscale=0
        while iscale<=bm.nscale:
            y1=round(y0+iscale*yt)
            dy=(bm.ymax-bm.ymin)/bm.nscale
            yscale=bm.ymax-iscale*dy
            w.create_text(x1,y1,text=yscale,
justify='center',font=bm.scalafont,anchor='e')
            iscale=iscale+bm.mscale
# Unteres GehÃ¤useteil
        ya=bm.hoehe*0.92

w.create_rectangle(4,ya,bm.breite,bm.hoehe,width=1,outline='white',fill
='black')
        iscale=iscale+bm.mscale
# Oberes GehÃ¤useteil
        ya=bm.hoehe*0.09

w.create_rectangle(4,0,bm.breite,ya,width=1,outline='white',fill=bm.fgc
olor)
# Beschriftung Dimension und Ylabel
        beschriftung='    '+bm.label
        #+bm.einheit
        xa=bm.breite/2-bm.breite*0.1
        ya=bm.hoehe*0.09/2.0
        w.create_text(xa,ya,text=beschriftung,
justify='center',font=('arial',16))
# Digitale Anzeige des Messwertes
        xa=bm.breite*0.5
        ya=bm.hoehe*(1.0+0.92)/2.0-3
        string=str(messwert)+' '+bm.einheit
        w.create_text(xa,ya,text=string,
justify='center',font=('arial',18),fill='green',tag='lcd')
#digital-7
# Balkenanzeige fÃ¼r Messwert
        x0 = bm.breite*0.6
        y0 = bm.hoehe*0.125
        x1 = bm.breite*0.75
        y1 = bm.hoehe*0.875
        y0 = y1-(y1-y0)/(bm.ymax-bm.ymin)*(messwert-bm.ymin)

w.create_rectangle(x0,y0,x1,y1,width=1,outline='black',fill=bm.barcolor
, tag='zeiger')
        return w
```

```
def barmeter_update(w,bm, messwert):
      w.delete('zeiger')
      w.delete('lcd')
      x0 = bm.breite*0.6
      y0 = bm.hoehe*0.125
      x1 = bm.breite*0.75
      y1 = bm.hoehe*0.875
      y0 = y1-(y1-y0)/(bm.ymax-bm.ymin)*(messwert-bm.ymin)

w.create_rectangle(x0,y0,x1,y1,width=1,outline='black',fill=bm.barcolor
, tag='zeiger')
# Digitale Anzeige des Messwertes
      xa=bm.breite*0.5
      ya=bm.hoehe*(1.0+0.92)/2.0-3
      string=str(messwert)+' '+bm.einheit
      w.create_text(xa,ya,text=string,
justify='center',font=('arial',18),fill='green',tag='lcd')
      return w




def ini():
    #Textfile oeffnen
    Datei = open("Rohdaten_SunnyIsland.txt","r")

    #einzelne Zeilen einlesen
    for line in Datei:
        Array.append(line)

    #Datei schliessen
    Datei.close()

    #Fehlerhandling (wenn nicht der komplette Parametersatz eingelesen
wurde, ist die Datei kuerzer als 159 Zeilen)
    if len(Array)<159:
        return 1


def string_Programm(value):

    #Arrayelement in Variable kopieren
    alleDaten = (Array[value])

    #  " wird ersetzt durch nichts (wird weggemacht)
    alleDaten_optimiert = alleDaten.replace('"', "")

    #Zahlen und Buchstaben werden aus gesamten String rausgeholt
    alleDaten_optimiert = re.sub(r'(\D) (\W+)'," ",alleDaten_optimiert)

    #  n] wird ersetzt durch nichts (wird weggemacht)
    alleDaten_optimiert = alleDaten_optimiert.replace('n]', "")

    #einzelne Elemente werden in Array gespeichert
    result = alleDaten_optimiert.split(" ")
```

```
    return result


def value_Programm(number):

    #Wenn der gewuenschte Parameter groesser als 251 ist, wird 4
    #abgezogen, damit die Zeilennummer im Textfile wieder stimmt.
    #Das muss gemacht werden, weil Parameter 252 bis 255 nicht
vorhanden sind.
    if (number>251):
        number = (number-4)

    # 169 werden abgezogen, damit die Zeilennummer im Textfile stimmt
    # auf diese Zahl kommt man, wenn man beruecksichtigt, dass im
Textfile ganz oben
    # zuerst noch andere Dinge stehen und weil der Parametersatz nicht
bei 1 anfaengt,
    # sondern bei 186 .

    value = number-169

    return value




frame1 = Frame(master)
frame2 = Frame(master)
frame3 = Frame(master)
frame4 = Frame(master)
frame5 = Frame(master)

frame1.pack(side=LEFT)
frame2.pack(side=LEFT)
frame3.pack(side=LEFT)
frame4.pack(side=LEFT)
frame5.pack(side=LEFT)



bm1=bmeter()
bm2=bmeter()
bm3=bmeter()
bm4=bmeter()
bm5=bmeter()


bm1.label           = 'SOC'
bm1.einheit         = '%'
bm1.ymin            = 0.0
bm1.ymax            = 100.0
```

```
bm2.label           ='SOH'
bm2.einheit         ='%'
bm2.ymin            = 0.0
bm2.ymax            = 100.0


bm3.label           = 'Leistung\nAC'
bm3.einheit         = 'W'
bm3.ymin            = 0.0
bm3.ymax            = 5000.0

bm4.label           = 'Spannung\nDC'
bm4.einheit         = 'V'
bm4.ymin            = 45.0
bm4.ymax            = 60.0

bm5.label           = 'Strom \nDC'
bm5.einheit         = 'A'
bm5.ymin            = 0.0
bm5.ymax            = 100.0

w1=barmeter_init(frame1,bm1,50)
w2=barmeter_init(frame2,bm2,50)
w3=barmeter_init(frame3,bm3,1000)
w4=barmeter_init(frame4,bm4,62.0)
w5=barmeter_init(frame5,bm5,50)


#main

while 1:
   #Pfad wechseln
    os.chdir("/home/pi/Desktop/FINAL_Sunny/yasdi/projects/generic-
cmake")

    #Daten aus SunnyIsland auslesen und in Textfile schreiben
    a=os.system('./yasdishell>Rohdaten_SunnyIsland.txt')

    #Daten aus Textfile in Array einlesen
    Error=ini()

    if Error:
        value=0
        #Fehlerbeschreibung aus Textfile im Terminal ausgeben
        while value < len(Array):
            result = (Array[value])
            print(result)
            value=value+1
        #Skript beenden
        quit()

    else:

        number = 191
        value = value_Programm(number)
        result = string_Programm(value)
```

```
        val1=float(result[3])
        barmeter_update(w1,bm1,round(val1,3))

        number = 187
        value = value_Programm(number)
        result = string_Programm(value)
        val2=float(result[3])
        barmeter_update(w2,bm2,round(val2,3))

        number = 188
        value = value_Programm(number)
        result = string_Programm(value)
        val3=float(result[3])*1000
        barmeter_update(w3,bm3,round(val3,3))

        number = 192
        value = value_Programm(number)
        result = string_Programm(value)
        val4=float(result[3])
        barmeter_update(w4,bm4,round(val4,3))

        number = 195
        value = value_Programm(number)
        result = string_Programm(value)
        val5=float(result[3])
        barmeter_update(w5,bm5,round(val5,3))

        master.update()


    # Array leer machen
    del Array[0:len(Array)]
```