Stepupwandler für solares Batterieladen

Von jean-claude.feltes@education.lu

1. Zweck

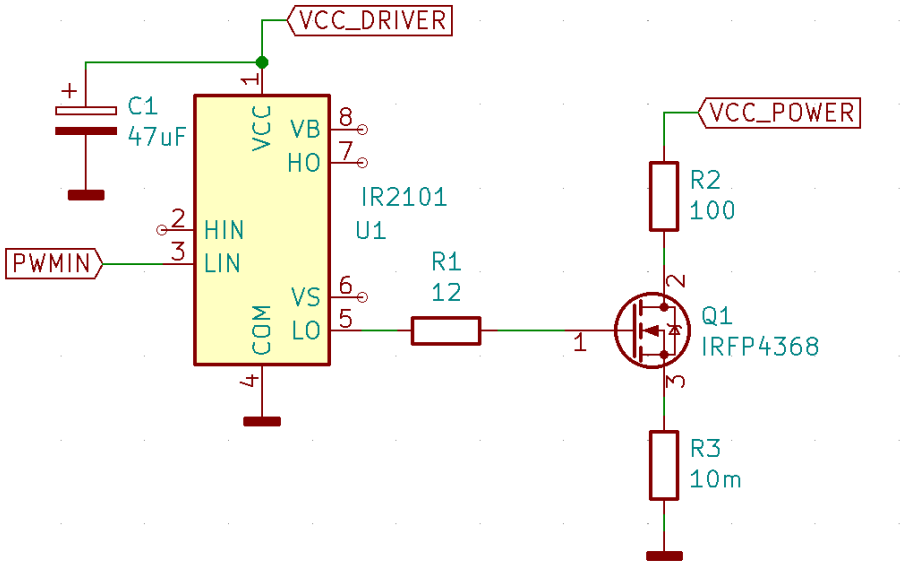
Der Wandler soll zwischen Solarpanel und eine 48V-LiIon-Batterie geschaltet werden, so dass diese mit Solarenergie geladen wird. Wegen der potentiellen Beschattung sind die Solarpanel nur zu jeweils zweit in Reihe geschaltet, diese speisen dann einen Wandler, oder wenn es sinnvoller erscheint, wird für jede Gruppe von 2 Paneln ein eigener Wandler benutzt.

Dieses Projekt ist experimentell, sein Ziel ist ein dreifaches: am Schluss soll eine brauchbare Schaltung herauskommen, daneben vieles über Leistungselektronik gelernt und dabei noch Elektronikschrott recycelt werden.

2. MOSFET + Driver Test

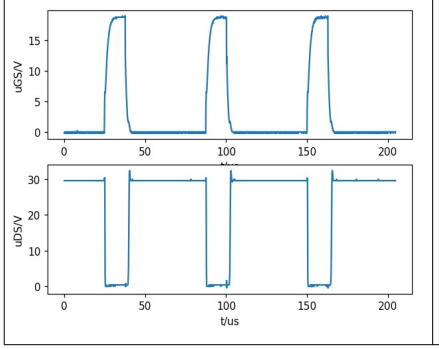
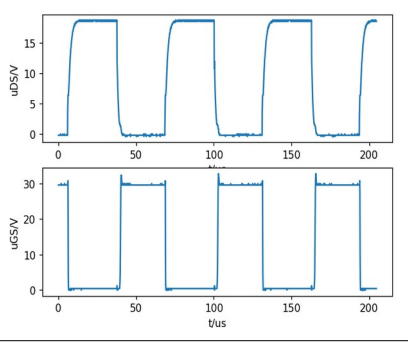
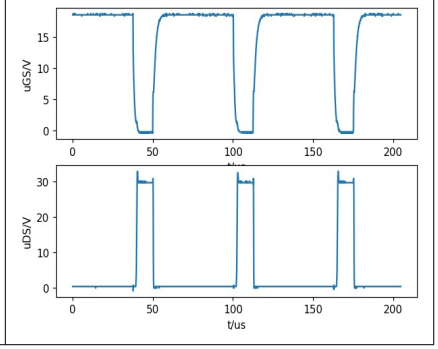
Für einen guten Wirkungsgrad ist es wichtig, Schalttransistoren mit niedrigem R_{DS} zu verwenden. Ausserdem ist ein Gatedriver nötig um die erforderlichen Umladeströme für die Gatekapazität zu liefern.
Als PWM-Signalquelle diente ein eigens für diesen Zweck gebauter Generator:

https://github.com/jean-claudeF/PWM-generator



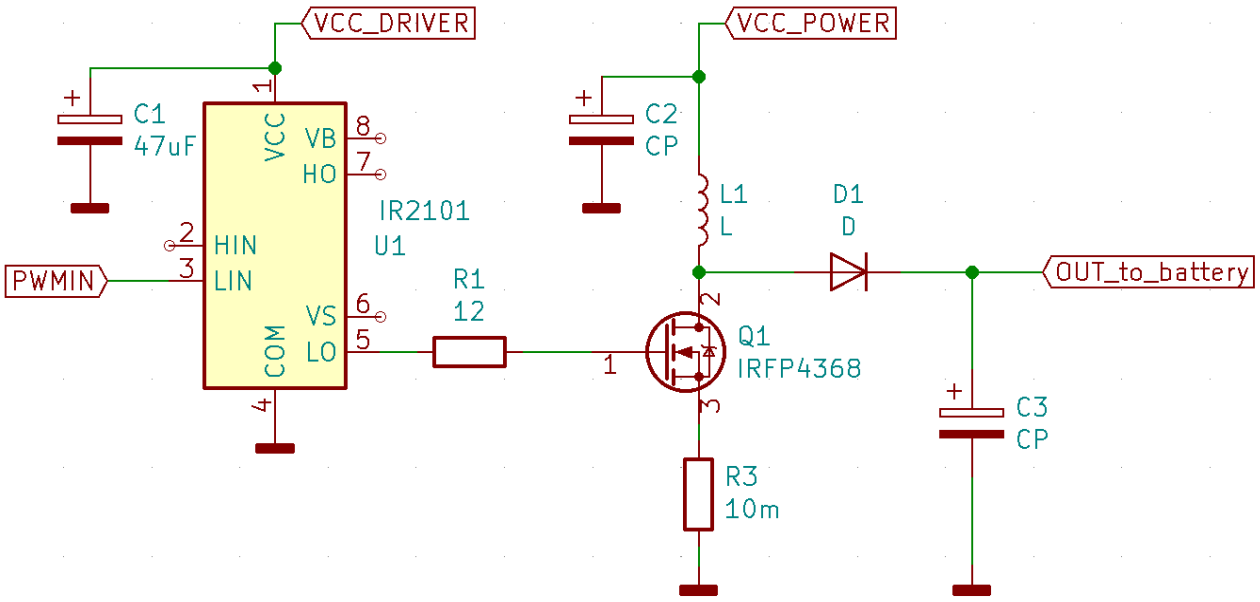
VCCPOWER = 30V
VCC_DRIVER = 18V (max. 20V, min. 10V)

f = 16kHz

20%	50%	80%
		

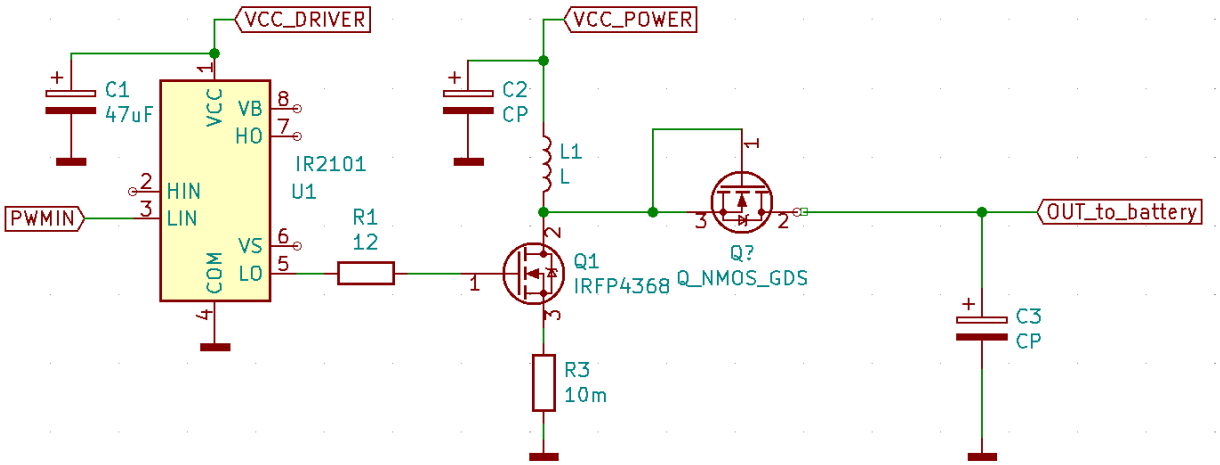
Die Signale am Gate sehen schlechter aus als erwartet, der MOSFET scheint aber (laut u_{DS}) gut zu schalten.

3. Test Stepup converter



L = 300uH
C2 = 330uF
C3 = 470uF
D = STTH3010PI (1000V / 30A, ultra fast) ist eine schlechte Wahl. Sie wird heiss da sie eine hohe Durchlass-Spannung hat.

4. Test Stepup mit MOSFET als Diode an Lastwiderstand



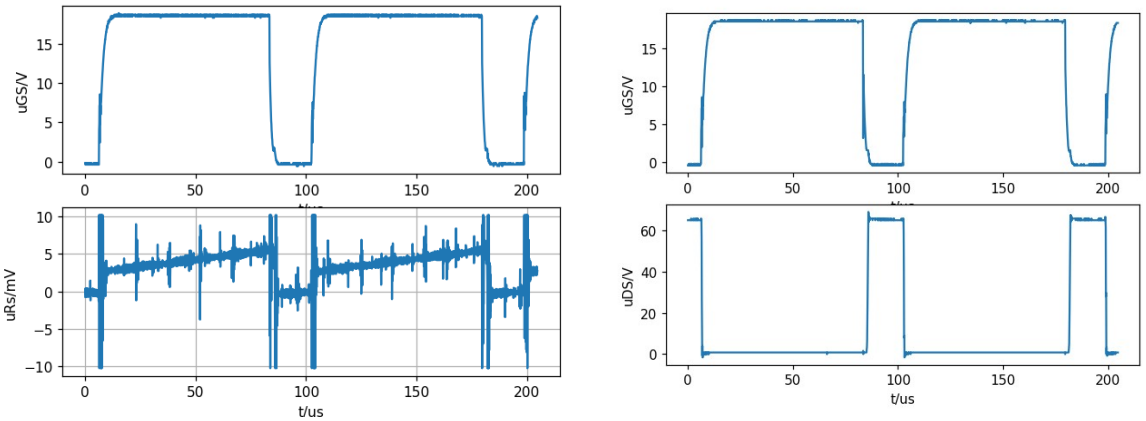
Beispielmessung:

VCC_Power = 12V

f = 10.4kHz, PWM = 80%

ICC = 4.25A, PCC = 51.3W

UOUT = 64.75V an RL = 100 Ohm.

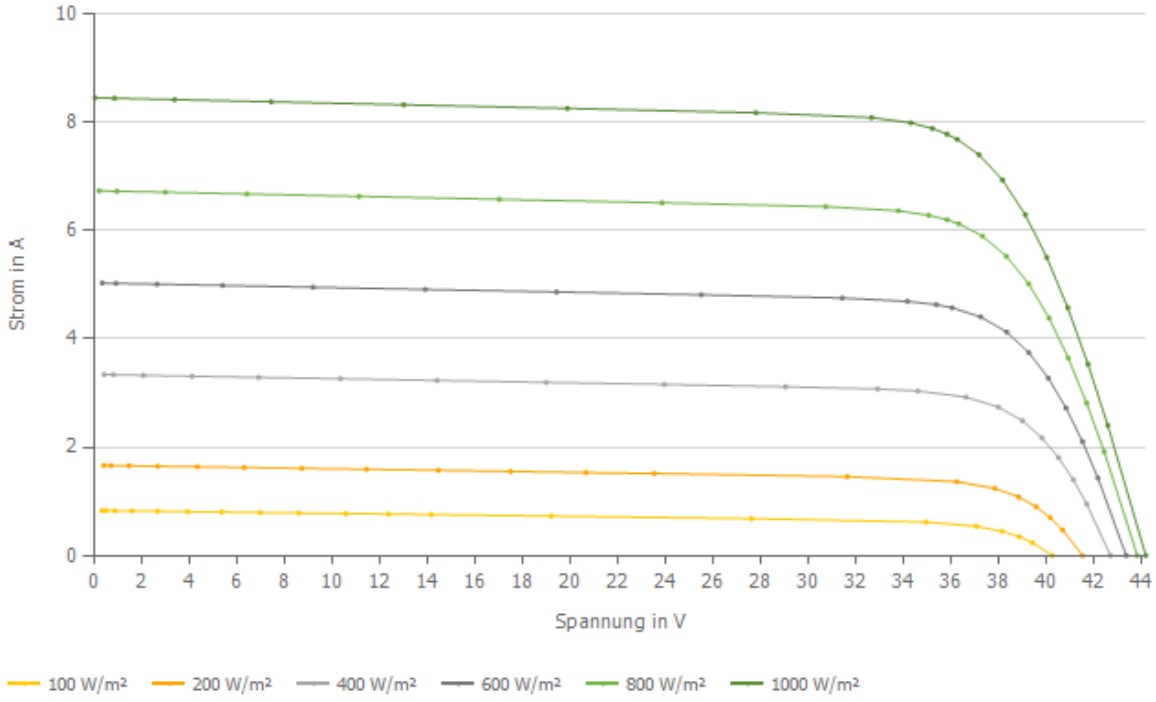


Die beiden Transistoren bleiben ziemlich kühl, die Spule wird leicht warm.

Eingang HIN des Drivers sollte auf Masse gelegt werden!

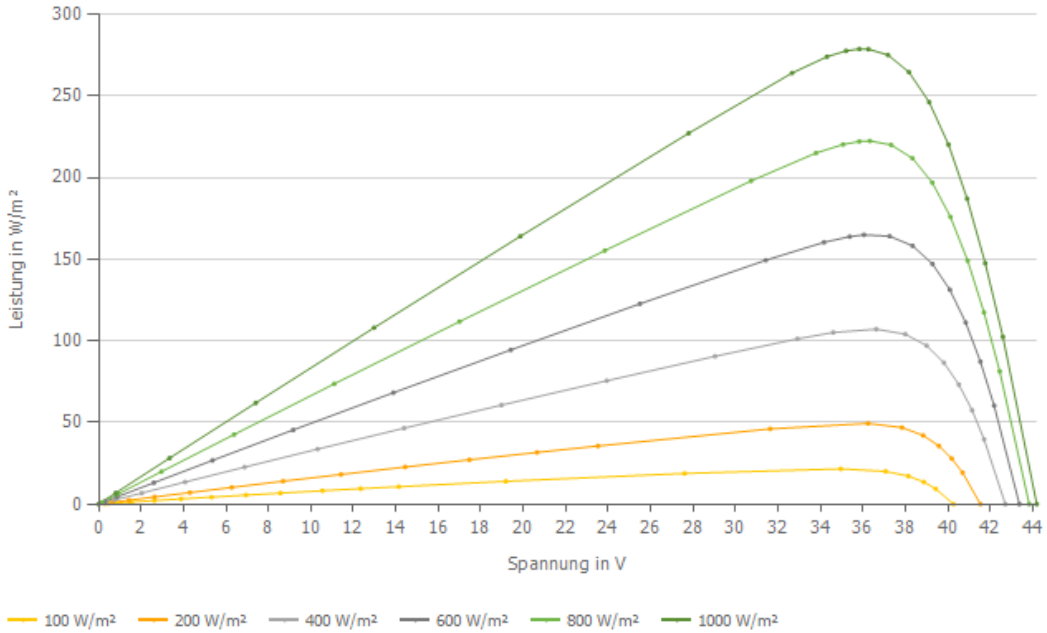
5. Kennlinien von Solarmodulen

Beispiel:



https://help.valentin-software.com/pvsol/de/berechnungsgrundlagen/pv-module/arbeitspunkte-und-kennlinien/

Maximum Power Point:



Für alle Einstrahlungsfälle liegt der MPP bei hoher Spannung 36-37V, nicht weit vom Leerlauffall entfernt. Dies bedeutet dass der PWM-Wert nur wenig variiert werden muss.

Daten unserer Panel:

solartronics 130W
Maximum power voltage = 18.13V
Ishortcircuit = 7.63A
Leerlaufspannung U0 = 22.25V

Davon 2 in Reihe geschaltet.

6. Test als Batterielademodul

Als Entkopplung von der Batterie wird zur Sicherheit eine Diode P600B verwendet.

(Diese wird später weggelassen, da Q2 schon diese Aufgabe übernimmt)

Kleiner Praxistest.

Die Spannung VCC_POWER kommt von 2 in Reihe geschalteten 13W Solarmodulen, der PWM-Wert wird manuell nachgeregelt.

Der PWM-Wert für den MPP liegt um die 13-50%.

Batterie-Ladestrom ca. 0.5 – 2A bei bewölktem Himmel.

Solarspannung ca. 37 – 38V im MPP.

Bei etwas mehr Sonne: MPP bei PWM von 25%

7A, 36V, → 252W (Solarpanel)

4.9A, 44.9V → 220W (Batterie)

→ Wirkungsgrad 87%

7. Erfassung der Sonneneinstrahlung

Ohne diese gibt es keine verlässlichen Vergleiche. Es wird eine kleine Solarzelle von einem Lichtverschmutzungs-LED-Stab (Solar-LED für die Gartenbeleuchtung) benutzt (ca. 25cm^2).

Erster Versuch: Messung des Kurzschluss-Stroms mit mA-Meter. Funktioniert, aber ungenau wegen recht hohem Innenwiderstand des Messgeräts.

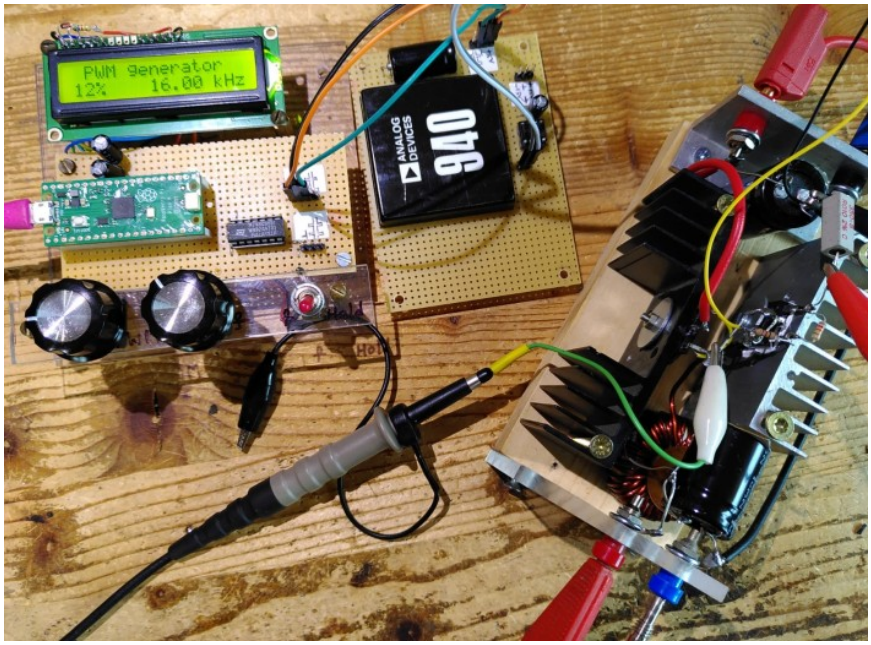
Zweiter Versuch: Nutzung eines vorhandenen I-U-Konverters mit OPV und Messung mit V-Meter. Scheint recht gut zu funktionieren.

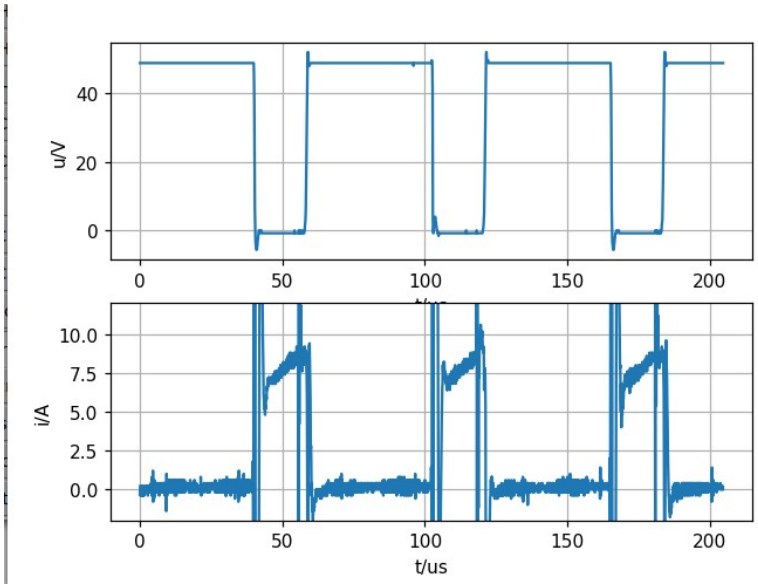
8. Erzeugung der Gatedriver-Betriebsspannung aus den +5V

16.4.2023

Hier wurde ein fertiges (altes) Modul von Analog Devices benutzt : der 940 macht aus 5V eine +15V (VCC_DRIVER) und eine -15V Betriebsspannung (hier nicht benutzt).

Dieses Modul wurde zusammen mit einem Pufferelko und dem Gatedriver IR2101 auf einer separaten Platine untergebracht.





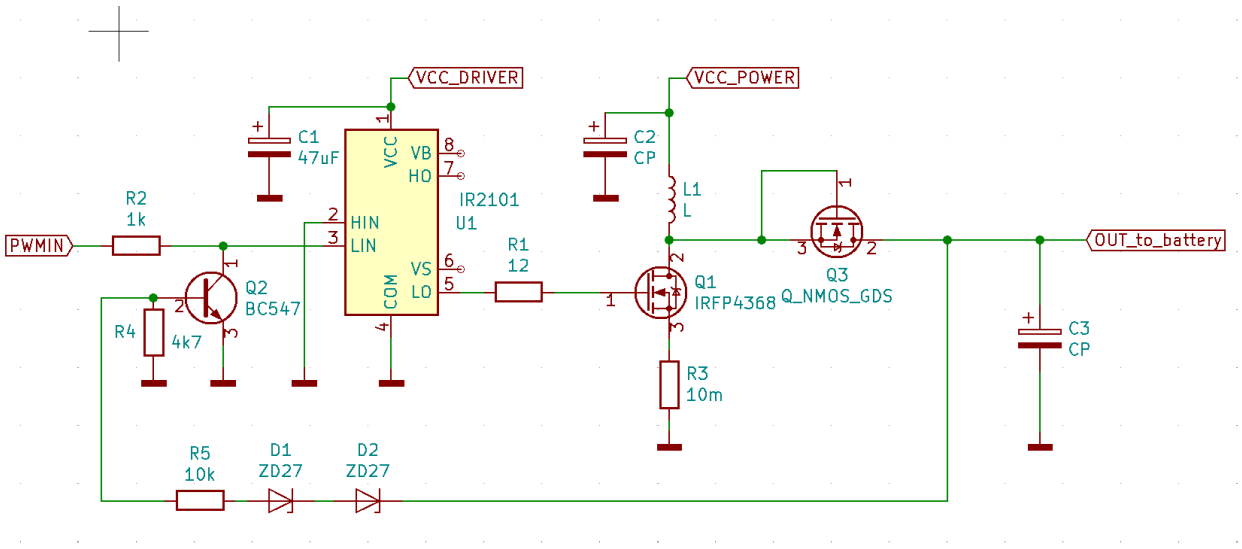
Oben: uDS, unten iDS

9. Problem ohne Belastung am Ausgang

Wenn der Akku keinen Strom aufnimmt (z.B. Abschaltung da voll), wird der Wandler sehr heiss und die Ausgangsspannung steigt stark an.

→ in diesem Fall muss abgeschaltet werden!

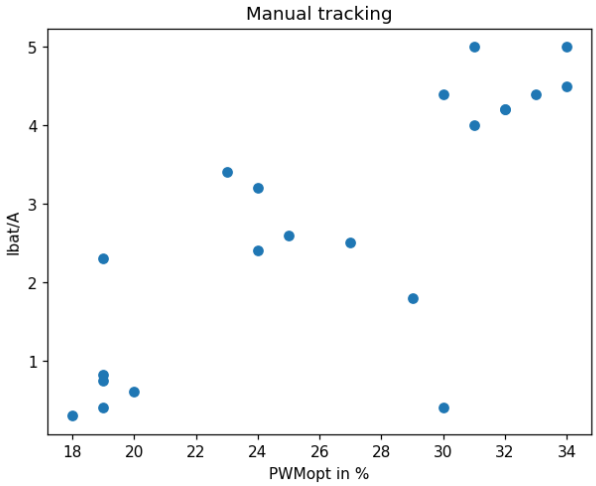
Diese Lösung scheint zu funktionieren:



Im Leerlauf steigt die Spannung am Ausgang auf über 54V, die Z-Dioden werden leitend, Q2 ebenfalls so dass das PWM-Signal kurzgeschlossen wird und der Wandler inaktiv.

Diese Lösung ist bei geringer Sonneneinstrahlung getestet worden. Der Härtetest bei voller Einstrahlung steht noch aus.

10. Manuelles Tracking



Das Diagramm zeigt den Zusammenhang zwischen manuell eingestelltem optimalen PWM-Wert und Batteriestrom (Maximaler Strom entspricht in etwa auch maximaler Leistung da die Batteriespannung relativ konstant ist).

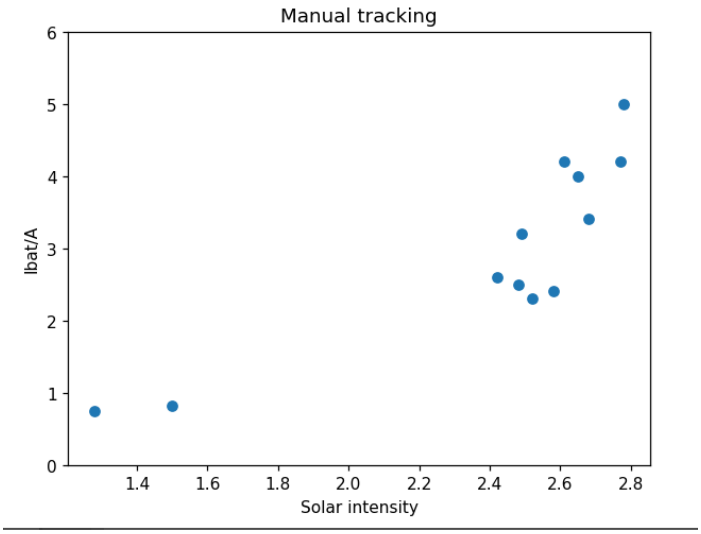
Wie erwartet muss der Tastgrad bei höheren Strömen (durch stärkere Sonneneinstrahlung) für den MPP erhöht werden. Lediglich 2 Punkte entsprechen dem nicht. Messfehler?

Listing:

```
import matplotlib.pyplot as plt

x = [23, 32, 27,25, 24, 31, 19, 19, 19, 24, 34, 29, 18, 30, 19, 20, 33, 30, 31, 32, 34]
```

```
y = [3.4, 4.2, 2.5, 2.6, 3.2, 4, 2.3, 0.75, 0.82, 2.4, 4.5, 1.8, 0.3, 0.4, 0.4, 0.6, 4.4, 4.4, 5,
4.2, 5]

plt.title("Manual tracking")
plt.plot(x, y, 'o')
plt.xlabel("PWMopt in %")
plt.ylabel("Ibat/A")
plt.show()
```

Im nächsten Diagramm ist die Abhängigkeit des MPP-Batteriestroms von der Sonneneinstrahlung dargestellt:



11. Erweiterung der Messtechnik

19.4.2023

Für weitere Experimente mit manuellem MPP-Tracking wäre es wünschenswert folgendes zu erfassen (und die Werte gleich in eine Datei schreiben zu lassen, serielle Übertragung):

- Strom Batterie + Panel
- Spannung Batterie + Panel (neue I2C-Wandler!)
- Sonneneinstrahlung relativ
- Eingestellter PWM-Wert für MPP
- daraus berechnet Pzu, Pab, Wirkungsgrad

Die Steuerung soll weitgehend über den PC erfolgen, PWM-Wert manuell Pot oder Schieber am PC, später automatisch.

18.5.2023

Für die Erfassung von Spannung, Strom und Leistung wurde ein Breakoutboard mit **INA226** ausprobiert. Einen funktionierenden Micropython-Driver gab es, allerdings mit einem Bug den ich nicht sofort bemerkte. Das Datenblatt mit seinem Kapitel über Programmierung war eher hinderlich

als nützlich, die Vorgehensweise erscheint mir auch heute noch extrem unpädagogisch und unlogisch. Nachdem ich dieses Kapitel ignorierte und mich auf das Blockschaltbild fokussierte, gelang die Programmierung des Treibers problemlos. Als Grundgerüst wurde der Treiber von Chris Becker genommen.

Hier ist das Ergebnis:

https://github.com/jean-claudeF/INA226

Die bequeme Variante nutzt nur das Bus voltage register und das Shunt voltage register, und sollte eigentlich auch noch genauere Ergebnisse liefern:

```
import ina226_jcf as ina226
from machine import Pin, I2C
import time

i2c = I2C(0,scl=Pin(9), sda=Pin(8), freq=100000)
ina = ina226.INA226(i2c, 0x40, Rs = 0.002, voltfactor = 2 )

i=0
while True:


    v, i, p = ina.get_VIP()
    V = '%2.3f' % v
    I = '%2.3f' % i
    P = '%2.3f' % p
    print(i, '\t', V, '\t', I, '\t',  P)
```

12. Intermezzo: Spulen für einen zweiten Leistungsteil

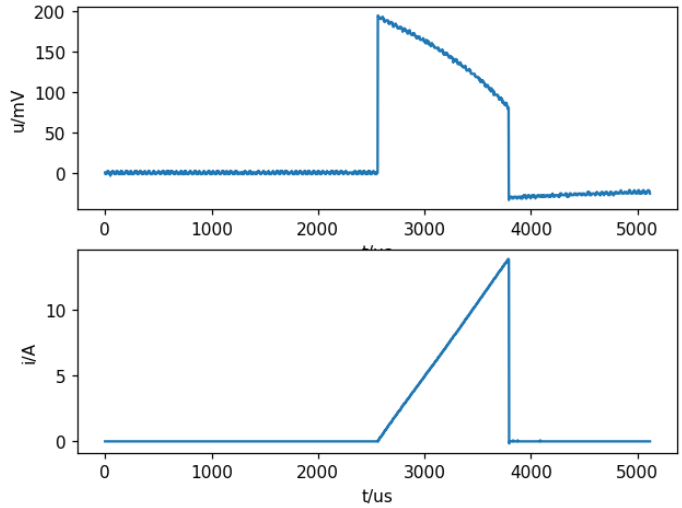
Es wurden aus verschiedenen, meist PC-Netzteilen Spulen ausgeschlachtet, neu gewickelt und vermessen. Dabei leistete das Spulen-Messgerät wertvolle Hilfe:

http://staff.ltam.lu/feljc/electronics/messtechnik/messung_ferritspulen.pdf

Der Impulsgenerator wurde noch leicht umgebaut, um einen grösseren Impulsdauer-Bereich zu ermöglichen.

Nicht alle Spulenkerne erfüllten die Erwartungen. Einige hatten auch bei bescheidenen Windungszahlen recht niedrige Sättigungsströme.

850uH-Spule (unten links im Bild)

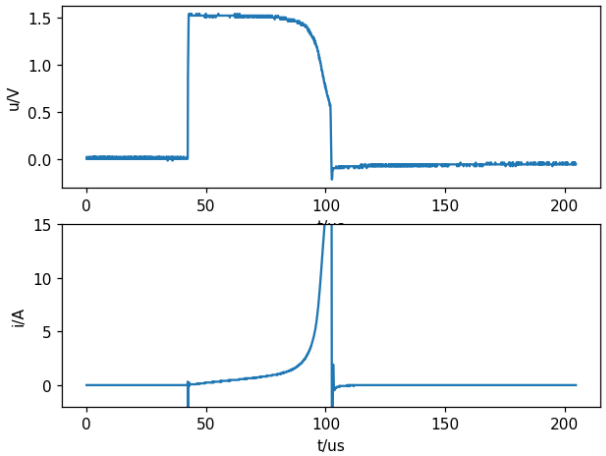


Unten: Strom, oben: induzierte Spannung in 1 Windung

Die zusätzliche Spannungsmessung in 1 Windung wurde vorgenommen, um eine zusätzliche Information über die Sättigung zu bekommen. Das Problem ist nämlich, dass sich eine durch den Widerstand nach unten gekrümmte Kurve und eine durch die Sättigung nach oben gekrümmte Kurve teilweise kompensieren können, so dass die Kurve linear aussieht, obwohl die Spule schon in der Sättigung ist. (Allerdings müsste sich die exponentielle Widerstandskrümmung schon zu Beginn der Kurve zeigen, während die Sättigungskrümmung erst bei hohen Strömen auftritt.)

Im Idealfall sollte die induzierte Spannung impulsförmig sein, im Sättigungsfall müsste sie stark abnehmen.

Ungeeignete 360uH Spule, Is = 1A (nicht im Bild):



Ab dem Sättigungsstrom bricht die induzierte Spannung stark ein.

13. Intermezzo: Schatten-Versuch

Wie wirkt sich eine partielle Abschattung der Panel aus?
Um dies zu untersuchen habe ich folgendes Experiment gemacht:

1. Zunächst wurde bei voller Sonneneinstrahlung der MPP manuell eingestellt, als Ergebnis floss ein Batterie-Ladestrom von ca. 4A bei einem PWM-Wert von 33%.
2. Dann wurde etwa 1/3 der Fläche eines der beiden in Reihe geschalteten Solarpanel abgedeckt (Also etwa 1/6 der Gesamtfläche). Das Ergebnis war dramatisch. Der Strom ging bei unverändertem PWM-Wert praktisch auf null zurück.
3. Nun wurde der MPP wieder manuell nachgestellt, mit dem Ergebnis dass sich der Strom wieder auf 1.8A bringen liess, nun bei einem PWM von 70%.
   Das MPP Tracking lohnt sich also gerade bei Abschattung eines Teils.
4. Beim gleichen PWM-Wert wurde die Abdeckung entfernt. Das Ergebnis war schlechter als zu Beginn, der Strom betrug nur 2A. Mit manuellem Tracking auf einen PWM-Wert von 33% wie zu Beginn ergab sich wieder der Strom von 4A.

Fazit:

Ein MPP-Tracking kann erhebliche Leistungsgewinne bringen, vor allem wenn ein Teil der Zellen im Schatten liegt.

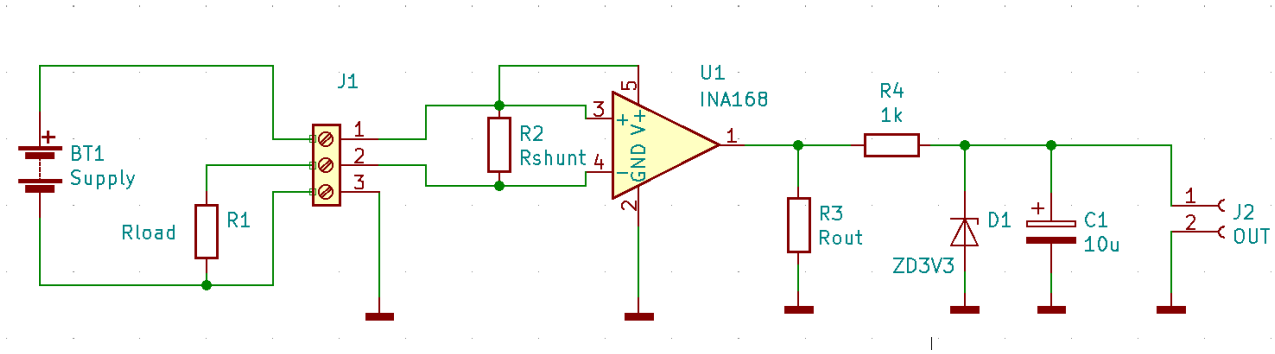
14. “Schönere” Strom-Messung

1.6.2023

Der Wunsch nach einer High Side Messung (Warum? Wegen der gemeinsamen Masse für alle Baugruppen) erinnerte mich an Versuche mit dem INA168. Dieser kann an bis zu 60V betrieben werden, er liefert ein analoges Ausgangssignal:

http://staff.ltam.lu/feljc/electronics/messtechnik/INA168_HighSide_Stromsensor.pdf

Erster Test:

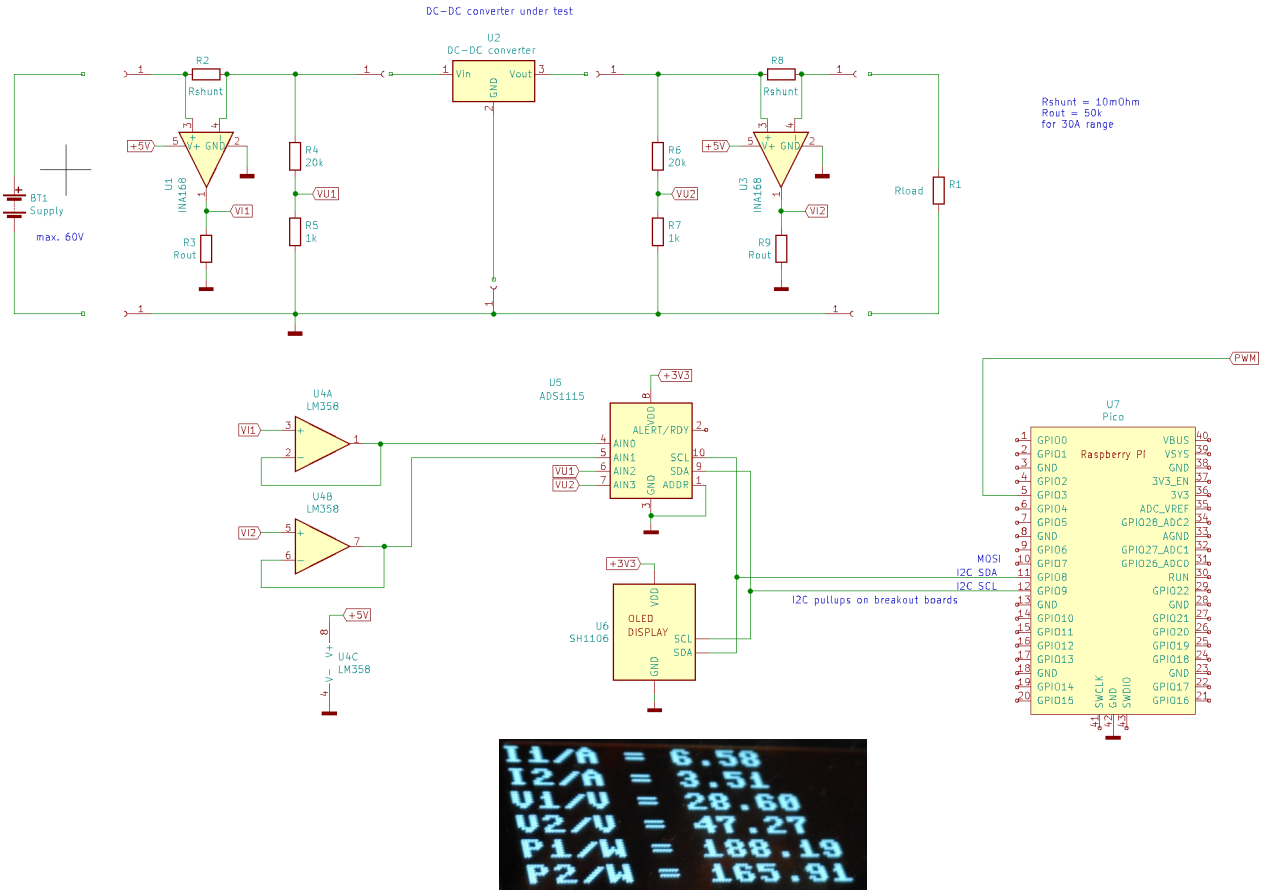


Rshunt = 10mΩ
Rout = 50kΩ

Da das Ausgangssignal auf eine ADS1115 – ADC geführt werden soll, wird es sicherheitshalber von D1 begrenzt (Es können schon mal Störimpulse mit zerstörerischer Spannung am Ausgang erscheinen.) Ausserdem sorgen R4 und C1 für eine Tiefpass-Filterung.

15. Ein Leistungs-Vierpol-Messgerät

Für genauere Untersuchungen des Stepupwandlers (oder eines anderen Leistungs-Vierpols) wäre es wünschenswert, alle Spannungen, Ströme und Leistungen zu messen. Das geht mit dieser Schaltung:

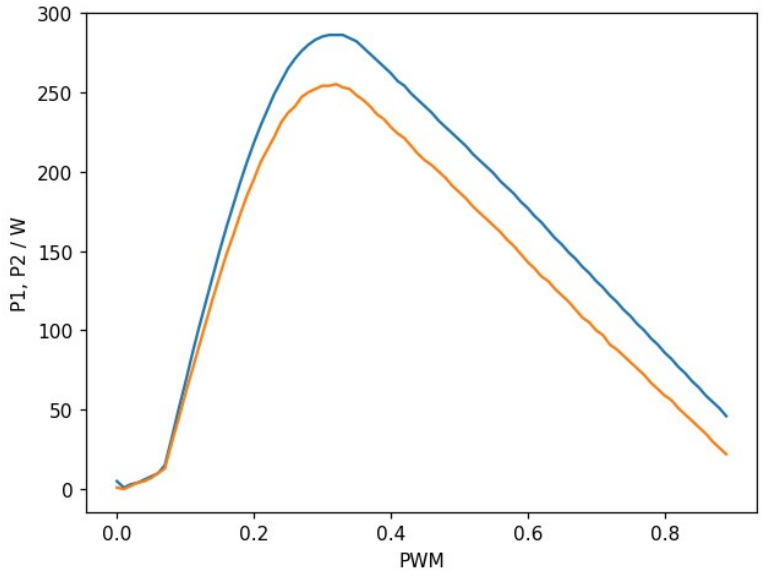


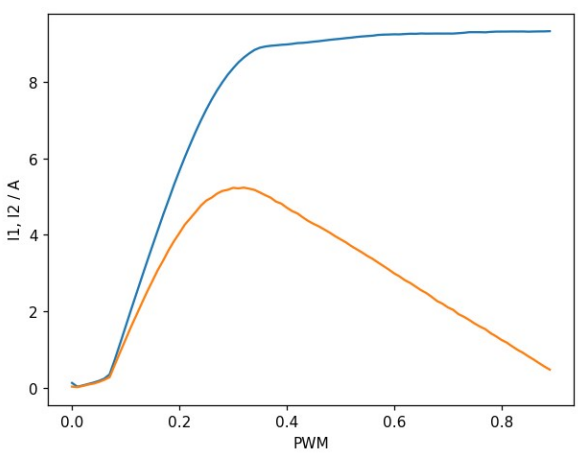
https://github.com/jean-claudeF/MeasurePowerQuadrupole

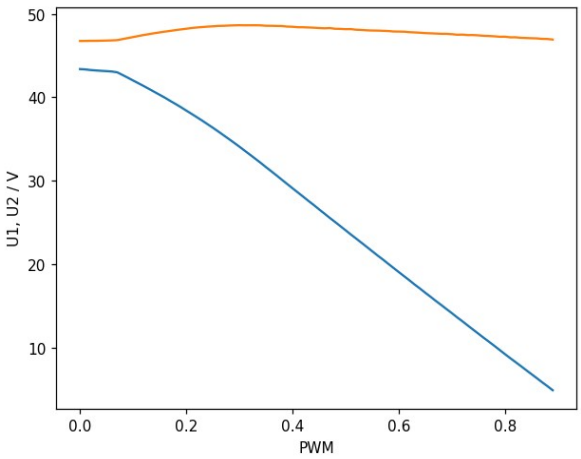
17.7.2023

16. MPP-Kurven

Die folgenden Kurven wurden mit dem INA168-Stromsensor aufgenommen. Dabei wurde eine Parallelschaltung von 2 x 2 in Reihe geschalteten Modulen verwendet.

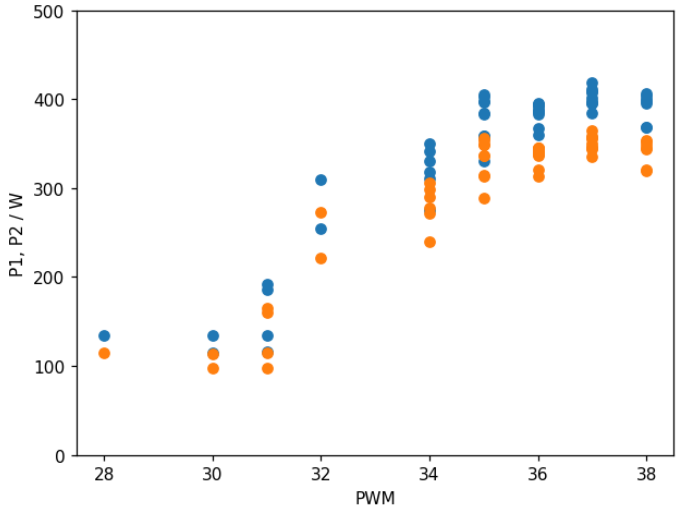






1 = Eingangsgrössen
2 = Ausgangsgrössen des Stepup – Wandlers.

Wie ist der Zusammenhang zwischen optimalem PWM-Wert und Leistung?

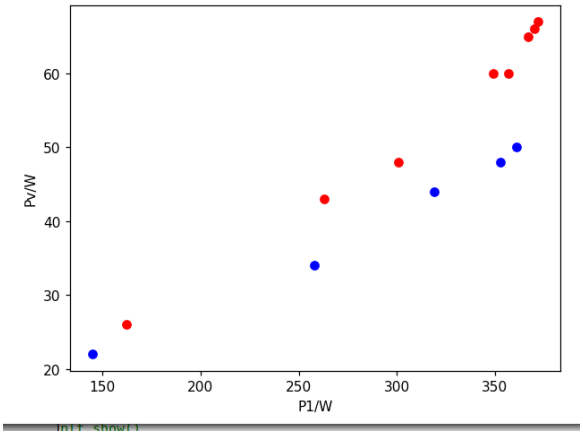


17. Spulenvergleich

Die MPP-Kurven wurden für 2 verschiedene Spulen aufgenommen:

Spule 1: Ringkern 300uH (blau)
Spule 2: Rechteckkern (trafoartig) 700uH, 89 Windungen (rot)

Aus den aufgenommenen Werten wurde Pvmax = f(P1max) ermittelt:



Spule 2 schneidet schlechter ab.

18. MPP Tracking mit dem Leistungs-Vierpol-Messgerät

Für eine einfachere Software-Realisierung wurde das Gerät in einem Objekt abgebildet.

https://github.com/jean-claudeF/MeasurePowerQuadrupole/blob/main/Micropython/mpptrack_01.py

Das Hauptprogramm ist damit recht einfach:

```
# Define object with or without connected OLED:
m4p = Measure4pole(adc, pwmgen, oled = oled)
#m4p = Measure4pole(adc, pwmgen, oled = None)

m4p.set_calibration(k0, k1, k2, k3, offset0, offset1)
m4p.set_pwm(0.3)

# Track MPP, set PWM accordingly in regular intervals
# Display values
i = 0
while True:
    if i % 10 == 0:
        if oled:
            oled.print("MPP tracking")
        m4p.mpp_track()

    ##i1, i2, v1, v2, p1, p2, eta = m4p.measure()
    m4p.measure()
    m4p.print_values()
    m4p.print_oled()
    i += 1
    time.sleep(1)
```

Zunächst wird ein Objekt Measure4pole definiert. Bei der Definition sind adc, pwmgen und oled zu übergeben (die natürlich vorher definiert werden müssen. Dieser Teil des Programms ist hier nicht dargestellt, zu finden aber hier: https://github.com/jean-claudeF/MeasurePowerQuadrupole).

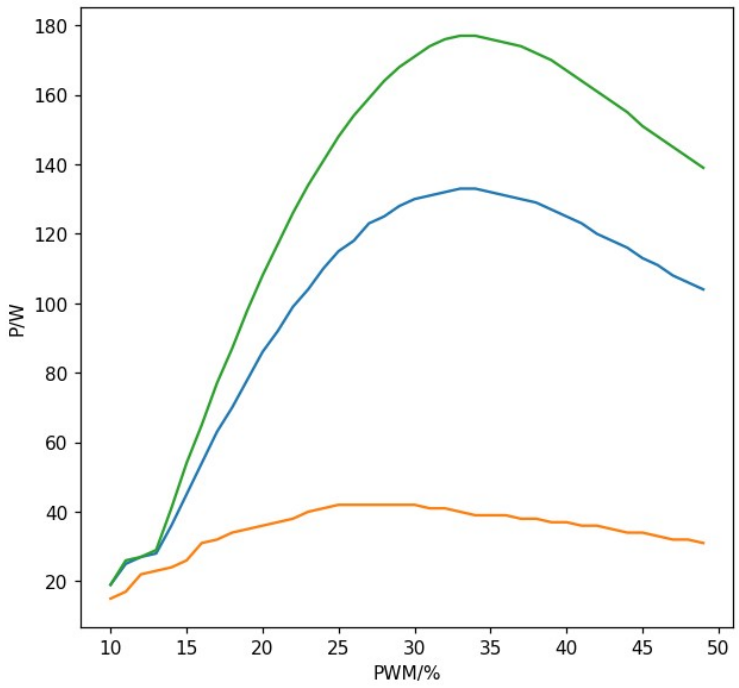
Es ist auch oled = None möglich, wenn kein OLED benutzt wird.

Dann werden die Kalibrierfaktoren gesetzt.

Im Beispiel erfolgt alle 10s ein Tracking, automatisch wird der beste PWM-Wert eingestellt, danach werden die Werte jede Sekunde einmal abgefragt und angezeigt.

Mit diesem Gerät können schnell Kurven bei verschiedener Sonneneinstrahlung, oder bei einer Zusammenschaltung von mehreren Panels aufgenommen werden.

Beispiel:

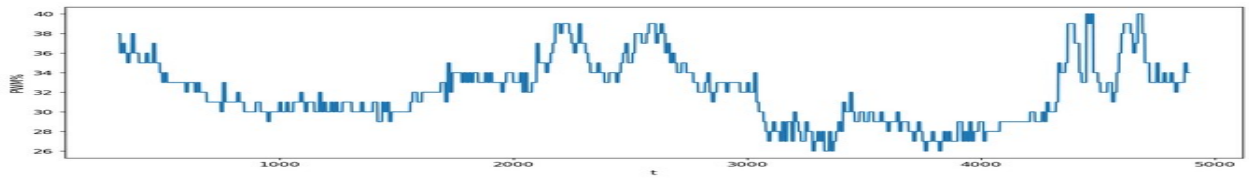
	Aufgenommen um 17:30 orange: Ost-Panel blau: West- Panel grün: Parallelschaltung Wegen des abendlichen Sonnenstandes liefert der West-Panel den grössten Anteil. Es ist auch zu erkennen dass bei schwacher Einstrahlung ein niedrigerer PWM-Wert den grössten Ertrag liefert.

19. Messungen

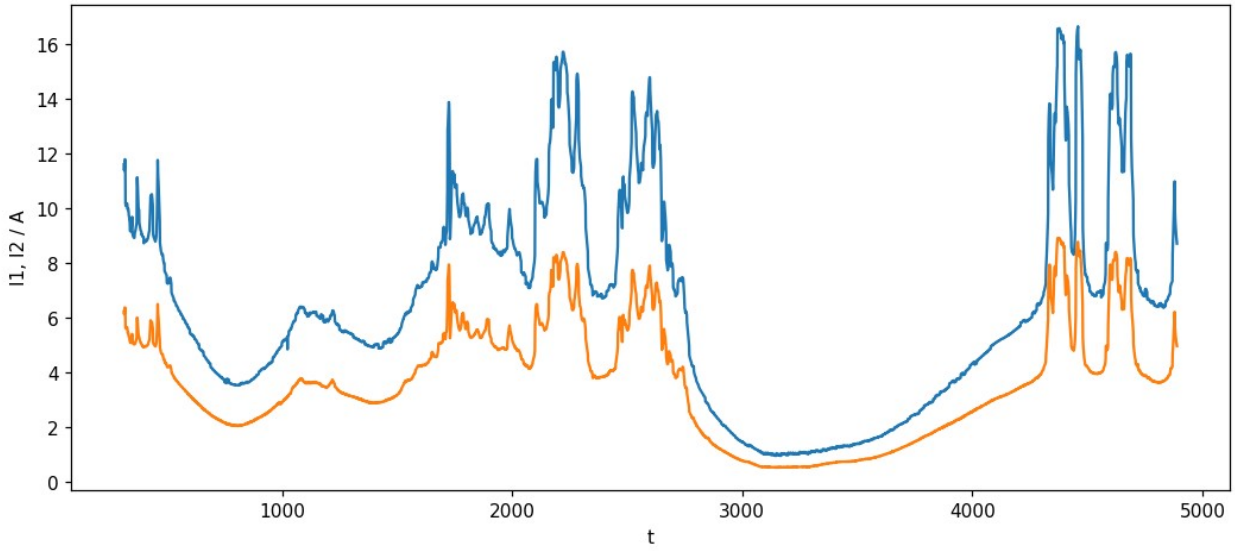
Die folgenden Messungen wurden am 4.8.2023 gegen Mittag gemacht, einem überwiegend bewölkten Tag, mit einer 3x2 – Gruppe von Paneln die unterschiedlich ausgerichtet waren (Nordost, Südwest, Südost, jeweils 2 Panel in Reihe, diese Gruppen dann über eine Diodenentkopplung parallel) .

1. welche PWM-Werte treten auf?

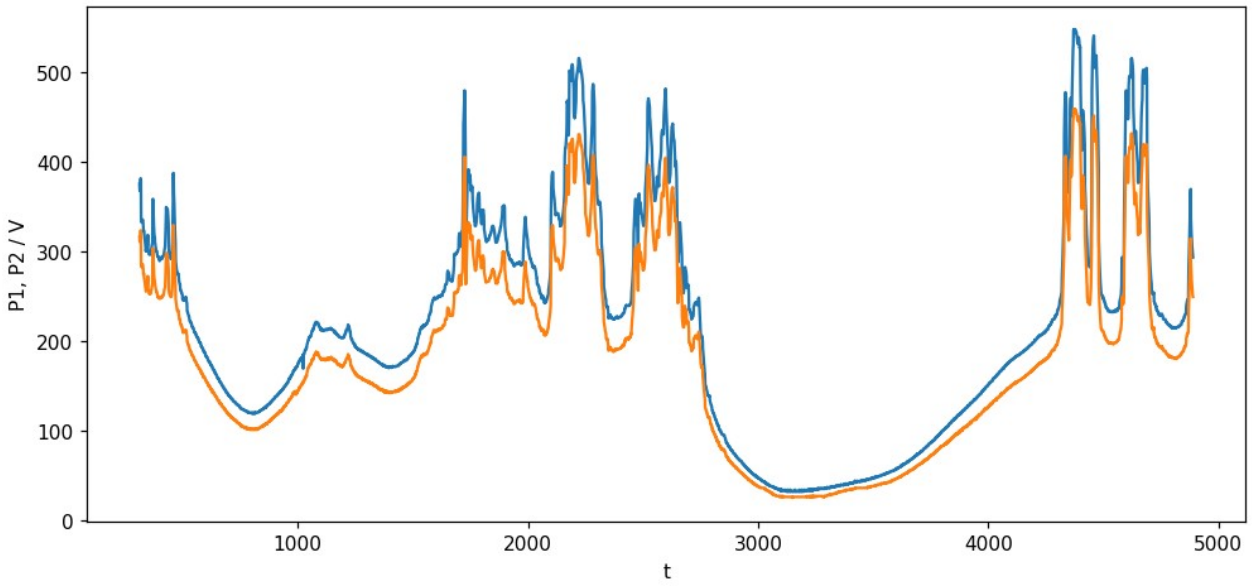
Alle Werte liegen zwischen 26 und 40%



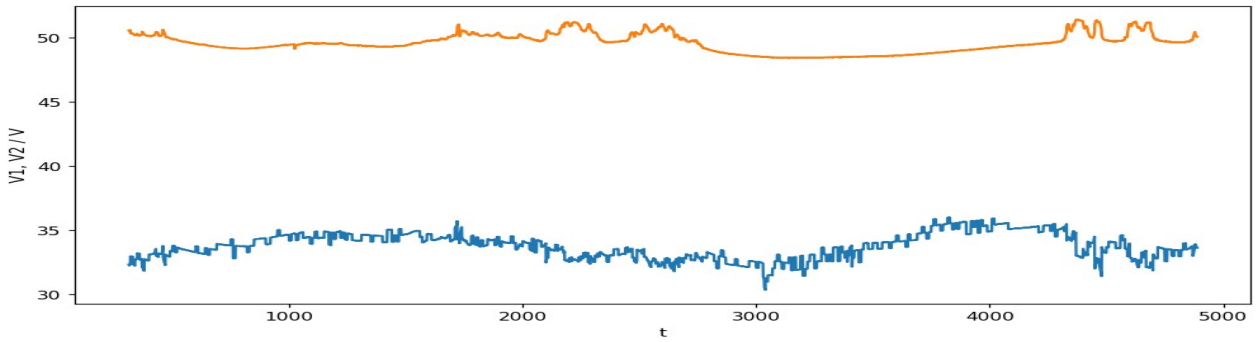
2. In welchem Bereich liegen die Ströme? I1 bis ca. 16A, I2 bis ca. 8A



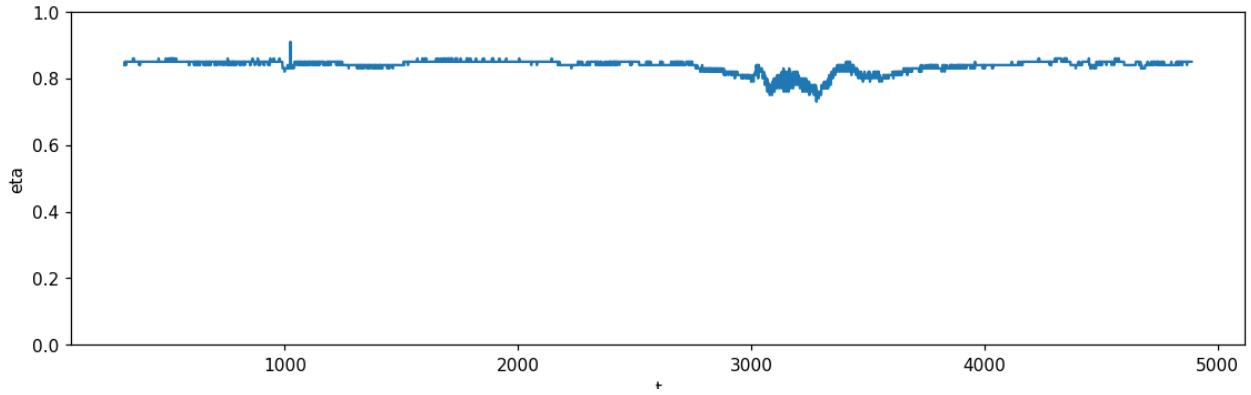
3. Welche Leistungen treten auf? P1max = ca. 570W, P2max = ca.450W



4. In welchem Bereich lag die Panel-Spannung? 32-37V für 2 Panel in Reihe.
Die Batteriespannung lag bei 48-49V, im Diagramm liegt die Kurve höher wegen des Spannungsabfalls am eigentlich zu dünnen Kabel. (Orange = Ausgangsspannung des Wandlers, es folgen einige Meter Kabel bis zur Batterie).



5. Was ist mit dem Wirkungsgrad des Wandlers?



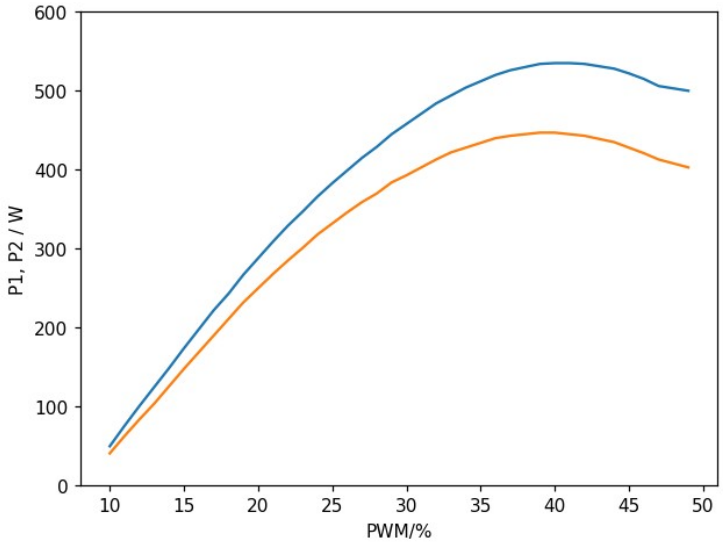
Dieser liegt die meiste Zeit um 0.85.
Erstaunlicherweise liegt das Minimum nicht bei der höchsten Leistung, sondern gerade bei der kleinsten. Dies ist wohl darauf zurückzuführen dass dort die Genauigkeit der Leistungsmessung am kleinsten ist. Der Wirkungsgrad scheint also relativ unabhängig von der Leistung zu sein.
Sicher wäre es interessant, später einen Synchronwandler mit besseren Eigenschaften zu entwickeln. Aber erst soll diese Schaltung richtig fertig werden.

Für die Messungen wurde eine Spule 1140-221K-RC von BOURNS verwendet.
Dies da die Recycling-Spulen bei Leistungen bis ca. 200W gut funktionieren, darüber aber recht heiss werden

Da die MOSFETs und die Spule zeitweise ziemlich heiss wurden, wurde ein 12V Ventilator zur Kühlung benutzt, dies funktionierte recht gut. Eine Temperaturüberwachung sollte den Ventilator schalten.

Immer noch scheint die Spule das heisseste und damit verlustreichste Teil zu sein.
Angestrebt ist eine Leistung des Wandlers von 500-800W (für 6 Panel), dafür müsste eine “dickere” Spule verwendet werden.

Interessant ist noch eine Tracking-Kurve bei guter Sonneneinstrahlung:



20. Entwurf einer praktischen Schaltung

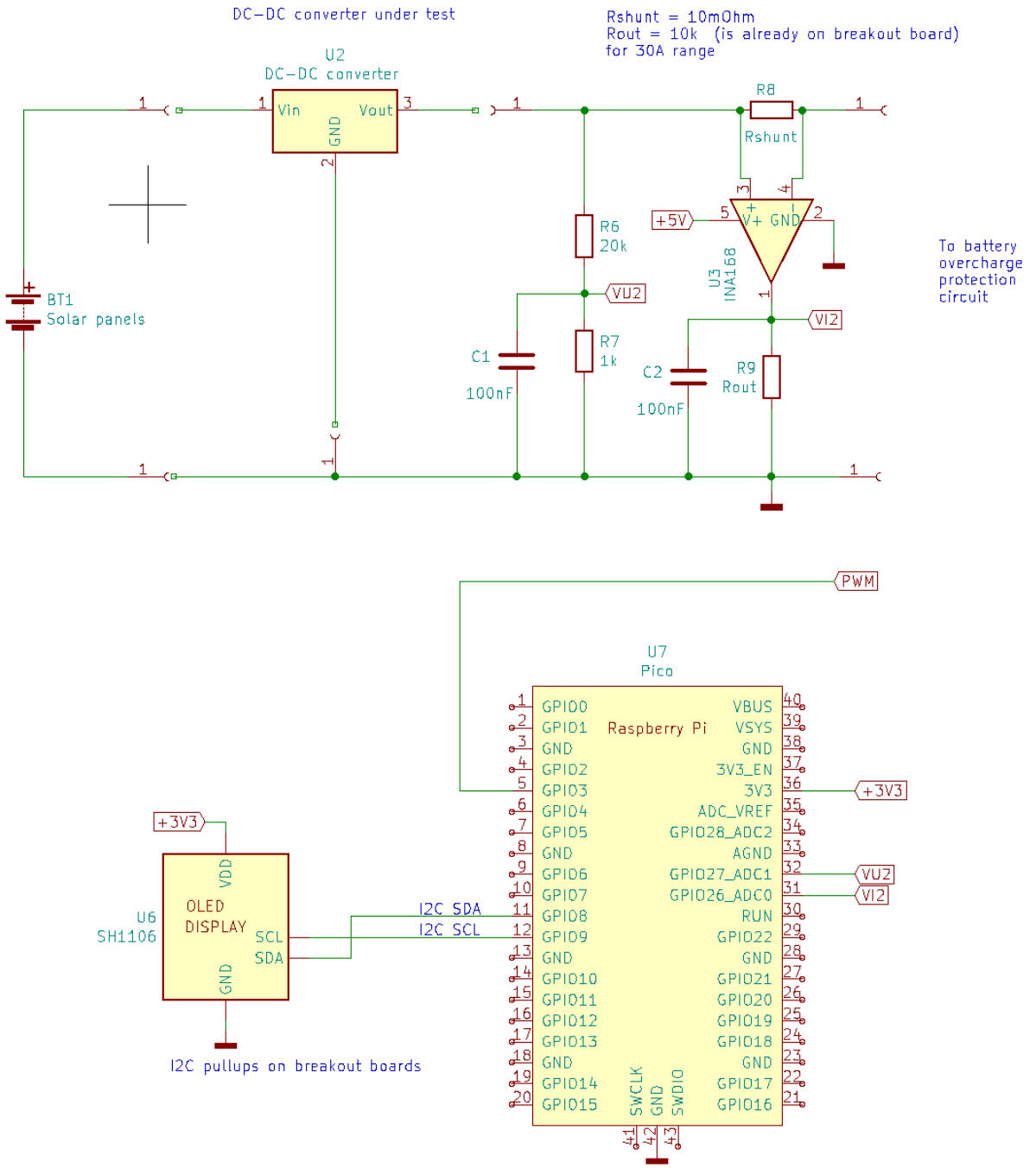
Die Messungen mit dem Measure4pole – Gerät lieferten wertvolle Erkenntnisse.

Für das praktische MPPT ist es aber nicht nötig alle Ein – und Ausgangsgrössen des Wandlers zu erfassen, es reicht, dies auf einer Seite zu tun (wahrscheinlich am besten am Ausgang, denn dies ist ja die Leistung die in die Batterie fliesst). Damit wird der Aufwand etwas kleiner.

Die Schaltung soll:

- U2, I2, P2 messen
- P2 periodisch tracken und den PWM-Wert einstellen so dass P2 maximal wird
- Die Temperatur überwachen und einen Ventilator zuschalten
- Die Leistung reduzieren wenn ein Grenzwert von P2 oder der Temperatur erreicht wird
- Alle Werte anzeigen (OLED) und über eine serielle Schnittstelle (separat, zusätzlich zu USB) ausgeben.

Hardware:



Zunächst wurde die Mess-Schaltung etwas abgespeckt um nur die Ausgangsspannung und den Ausgangsstrom (zur Batterie hin) zu messen. Da für das MPPT die Genauigkeit nicht unbedingt so gross sein muss, wurden die internen AD-Wandler des Pico benutzt.

Ein INA169 – Breakoutboard enthielt einen 100mΩ – Shunt, dieser wurde getauscht gegen einen höher belastbaren 10mΩ – Widerstand (bzw. dieser wurde parallel geschaltet, eine softwaremässige Korrektur des kleinen Fehlers ist ja leicht möglich). Der 10kΩ – Ausgangswiderstand wurde nelassen, damit ergibt sich ein günstiger Messbereich von 30A.

Auf einen Impedanzwandler wurde in dieser Version verzichtet, da Rout niederohmiger ist.

Dagegen wurde ein Filterkondensator eingebaut. Möglicherweise ist dies sogar günstiger als ein Impedanzwandler, bei dem es immer eine Offsetspannung gibt.

Achtung: der INA169 ht eine andere Steilheit S = 1mA/V als der INA168 der in der vorigen Schaltung benutzt wurde.

Für I = 1A ergibt sich mit den angegebenen Werten $U_S = R_s \cdot I = 10\,m\Omega \cdot 1\,A = 10\,mV$

Dies führt zu einer Spannung an Rout von $U_S \cdot S \cdot R_{OUT} = 10\,mV \cdot 1\,mA/V \cdot 10\,k\Omega = 100\,mV$

Da der Eingangsbereich des AD-Wandlers bis 3V reicht, könnten so 0...30A erfasst werden.

Ein praktischer Test zeigte eine Auflösung von ca. 10-20mA.

Die Filter-Grenzfrequenz ist $f_g = \frac{1}{2\pi R_{OUT} C_2} = 159\,Hz$.

Pico software:

Die Messung von Spannung, Strom und Leistung wurde in einer Klasse Measure_VIP gekapselt, die leicht anzuwenden ist, im Prinzip mit

```
#meas = Measure_VIP(adc0, adc1)           # without oled
meas = Measure_VIP(adc0, adc1, oled)      # with oled

while True:
    v, i, p = meas.get_VIP()
    meas.print_VIP()
    meas.print_oled()
    time.sleep(0.5)
```

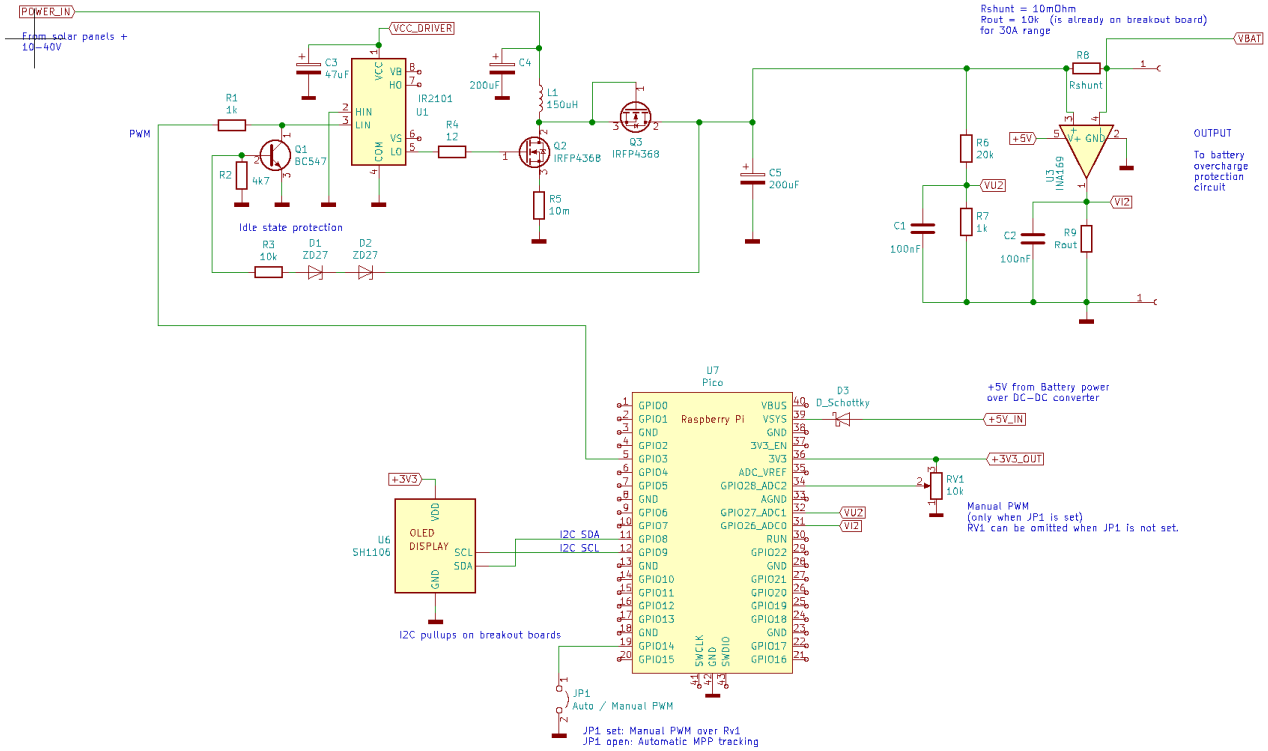
Diese wurde um eine Klasse MPPT erweitert, die von der Measure_VIP Klasse erbt.

Hiermit kann das Hauptprogramm übersichtlich strukturiert werden:

```
...
mppt = MPPT(adc0, adc1, pwmgen, oled)
mppt.nbmean = 10
mppt.pwm_min = pwm_min
...

mppt.set_pwm(0)

i = 0
maxpower = 0              # max power over whole operation
#-----------------------------------------------------------------------
def main_loop():
    global i, maxpower
    while True:
        # MPP track every track_time:
        if ( i % track_time) == 0:
            if pwm_manual.value():
                mppt.mpp_track()

        V, I, P = mppt.get_VIP()

        # Remember max power over whole operation
        if P > maxpower:
            maxpower = P
```

```
        # print and display values:
        print_my_values(i, V, I, P, mppt.pwmval)
        mppt.print_oled(additional = '\tPmax = %2.0fW' % maxpower )

        # PWM manual (potentiometer) or (normally) automatic
        # if manual, must react faster
        if pwm_manual.value() == 0:
            # PWM pot
            vpwm = adc2.read_u16() / 65535
            pwmgen.set_pwm(vpwm)

            time.sleep(0.05)
        else:
            time.sleep(1)

        i+= 1

main_loop()
```

21. Schaltung 20.September 2023

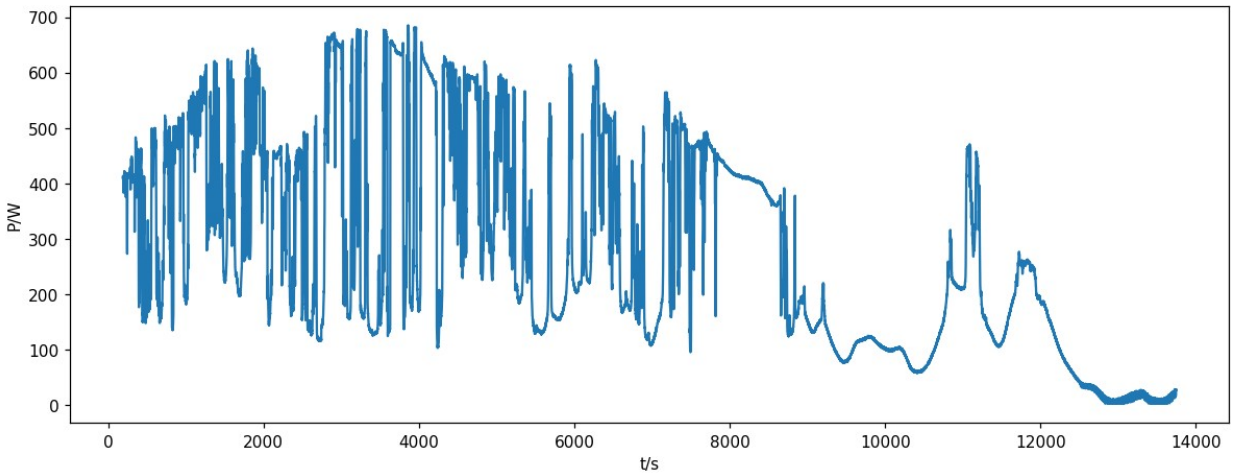


- Links: POWER_IN von 3-4 Gruppen von je 2 in Reihe geschalteten Paneln
- Rechts: Ausgang zur Batterie, über die analoge Schutzschaltung gegen Überladen
- Zu Testzwecken kann der PWM-Wert manuell eingestellt werden über RV1 (JP1 gesetzt)
  Bei offenem JP1 erfolgt automatisches MPPT.
- Nicht eingezeichnet:
  Die Stromversorgung des Pico (+5V_IN) erfolgt über ein kleines Stepdown-Netzteil aus der Batteriespannung.

- Die Stromversorgung des Gatedrivers (VCC_DRIVER) erfolgt nach wie vor über ein kleines Stepup-Netzteil aus der 5V-Versorgung.

22. Eine praktische Leistungsmessung

Eine Messung mit 4 Gruppen von je 2 Paneln zeigte, dass die Schaltung Leistungen bis über 600W abgeben kann.



Das Wetter war sehr wechselhaft, Bewölkung und Sonnenschein wechselten sich ab.

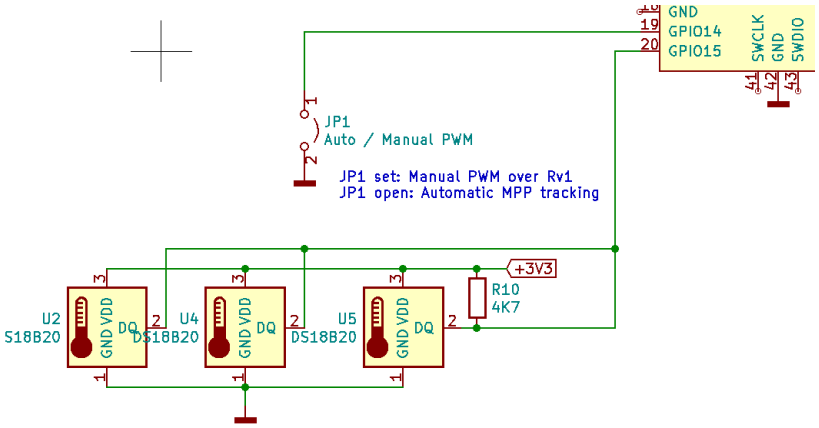
Für die Spule wurde hier eine Parallelschaltung von zwei “geretteten” 330uH – Ringkernspulen benutzt (Foto siehe Kapitel 12, links oben)

23. Temperaturüberwachung

Bei Leistungen über 300W erwärmen sich einige Bauteile ziemlich stark.

Um besser abschätzen zu können ob der Betrieb noch sicher ist, und gegebenenfalls einen Lüfter einzuschalten, wurde die Schaltung mit 3 DS18B20- Sensoren erweitert:

- am MOSFET
- am als Diode geschalteten MOSFET
- an einer der parallel geschalteten Spulen



Zum Auslesen wurde eine Klasse geschaffen:

```
class TemperatureSensors():
    def __init__(self, ds_pin):
        self.ds_sensor = ds18x20.DS18X20(onewire.OneWire(ds_pin))
        self.addresses = self.ds_sensor.scan()

    def convert(self):
        if len(self.addresses):
            self.ds_sensor.convert_temp()

    def get(self):
        self.temps = []
        for a in self.addresses:
            t = self.ds_sensor.read_temp(a)
            self.temps.append(t)
        return self.temps

    def print_temps(self, separator):
        for temp in self.temps:
            print(temp, end = separator)
        print()

    def checktemp(self, alarmtemp):
        alarms = []
        globalalarm = False
        temps = self.get()

        for t in temps:

            if t > alarmtemp:
                alarms.append(1)
                globalalarm = True
            else:
                alarms.append(0)
        return globalalarm, alarms
```

Die Temperaturmessung lässt sich so einfach in die Haptschleife integrieren:

```
ds_pin = Pin(15)
...
sensors = TemperatureSensors(ds_pin)
print(sensors.addresses)
def main_loop():
    global i, maxpower
    while True:

        sensors.convert()
        stemp = str(sensors.get())
        #sensors.print_temps('\t\t')

        # MPP track every track_time:
        if ( i % track_time) == 0:
            if pwm_manual.value():
                mppt.mpp_track()


        V, I, P = mppt.get_VIP()

        # Remember max power over whole operation
        if P > maxpower:
            maxpower = P

        ...

        i+= 1

main_loop()
```