

Gefördert durch:



Bundesministerium
für Wirtschaft
und Klimaschutz

aufgrund eines Beschlusses
des Deutschen Bundestages

Technische
Hochschule
Rosenheim



MonSec

Datenabfrage Fronius Wechselrichter mit IoT-Technik

Im Projekt MonSEC- Monitoring Secure

TH Rosenheim

Bearbeiter: Markus Hartmann

Stand: 03.08.2022

Inhaltsverzeichnis

Inhaltsverzeichnis	1
Abbildungsverzeichnis	2
1 Einleitung	3
2 Vorbereitung Fronius Wechselrichter	4
3 Installation und Einrichtung des RaspberryPi	5
3.1 Einrichtung von Influx	5
3.2 Einrichten von NodeRed	5
3.3 Einrichtung von Grafana.....	8
4 Abfrage Smartmeter über Wechselrichter	10

Abbildungsverzeichnis

Abbildung 1: Konfiguration der Datenausgabe des Fronius Wechselrichters	4
Abbildung 2: Übersicht der Programmierung in NodeRed	5
Abbildung 3: Eingabe der Registerabfrage.....	6
Abbildung 4: Erstellung der Modbus Abfrage	6
Abbildung 5: Konfiguration des Modbus Flex Getter	6
Abbildung 6: Parsing Funktion 1	7
Abbildung 7: Parsing Funktion 2	7
Abbildung 8: Einstellungen zum Schreiben in Datenbank	7
Abbildung 9: Grafana Einstellungen für Verlaufsgraphen.....	8
Abbildung 10: Einstellung für Zahlenwerte	9

1 Einleitung

In dieser Anleitung wird beschrieben wie man einen Fronius Wechselrichter über Modbus TCP ausliest. Die Daten werden dabei in einer Influx Datenbank gespeichert und über Grafana visualisiert. Zum Programmieren der Abfrage und Speichern der Daten in der Datenbank wird die Programmierumgebung NodeRed verwendet.

2 Vorbereitung Fronius Wechselrichter

Damit der Fronius Wechselrichter die Daten auf der Schnittstelle zur Verfügung stellt, ist diese im Konfigurationsmenü einzustellen.

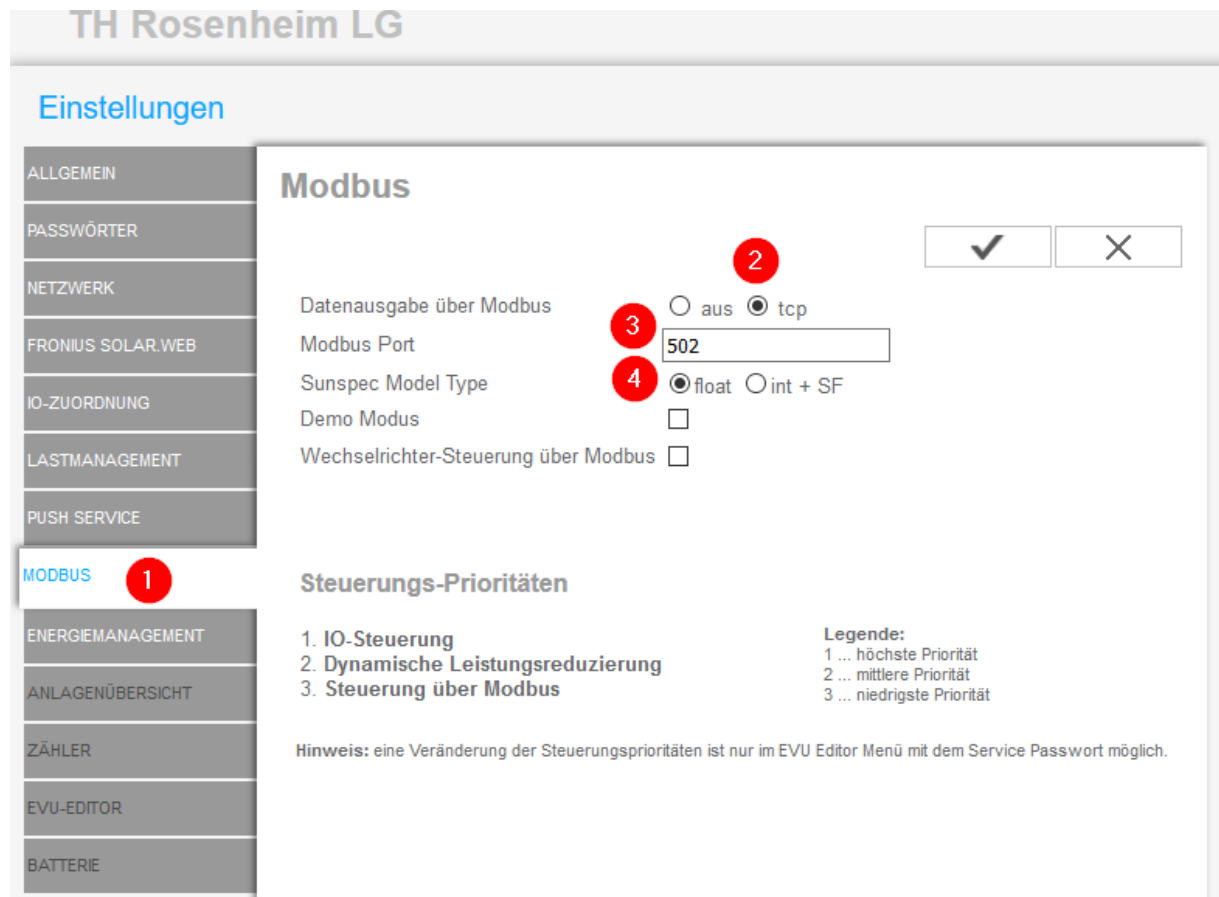


Abbildung 1: Konfiguration der Datenausgabe des Fronius Wechselrichters

Im Einstellungsfenster des Wechselrichters auf den Reiter „MODBUS“ (1) wird die Datenausgabe aktiviert (2). Der Modbus Port (3) ist default auf 502 gesetzt, ebenso die angegebenen Werte als float (4).

3 Installation und Einrichtung des RaspberryPi

Es werden die Programme NodeRed, Influx sowie Grafana benötigt. Die Installation der Programme selbst erfolgt nach bekanntem Vorgang.

3.1 Einrichtung von Influx

Bei Influx muss am wenigsten Aufwand betrieben werden. Neben einem Nutzer wird nur die Datenbank selbst benötigt.

3.2 Einrichten von NodeRed

Für die Programmierung der Datenabfrage und Speicherung der Daten in der InfluxDB werden die folgenden Paletten (Zusatzbibliotheken mit Programmbausteinen) benötigt:

- Node-red-contrib-influxdb
- Node-red-contrib-modbus

Die Paletten können in NodeRed direkt gesucht und heruntergeladen werden.

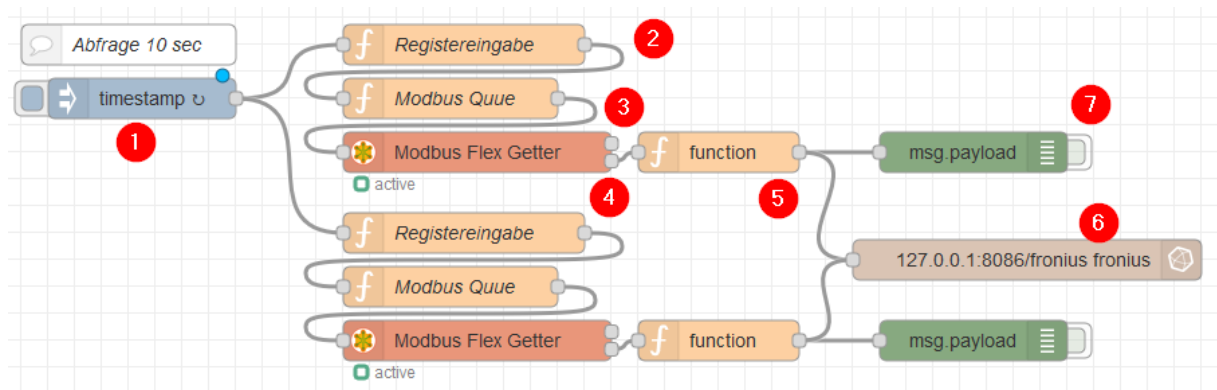


Abbildung 2: Übersicht der Programmierung in NodeRed

In Abbildung 2 ist der Programmcode als Übersicht dargestellt. Da die benötigten Daten auf dem Fronius in weit auseinander liegenden Registern bereit gestellt wird, werden 2 Anfragen parallel gestellt (oben / unten).

Der Reihe nach sind die Blöcke wie folgt:

1. Initialisierung und erzeugen eines zyklischen Startsignals (Inject)

Hier wird das Intervall der Abfrage eingegeben.

2. Funktion zur Eingabe der abzufragenden Modbus Register

```
1 msg.register = 40071;  
2 msg.quantity = 38;  
3 return msg;
```

Abbildung 3: Eingabe der Registerabfrage

In Abbildung 3 ist der Programmcode zur Einstellung der Registerabfrage zu erkennen. Es werden 38 Register ab der Registernummer 40072 abgefragt (Wichtig, Abfrage -1). Die zweite Abfrage enthält 48 Register ab der Nummer 40265.

3. Erstellung der Abfrage für den „Modbus Flex Getter“

```
1 msg.payload = {  
2   value: msg.payload,  
3   'fc': 3,  
4   'unitid': 1,  
5   'address': msg.register ,  
6   'quantity': msg.quantity } ;  
7 return msg
```

Abbildung 4: Erstellung der Modbus Abfrage

Der Code ist für beide Abfragen identisch.

4. Funktionsblock der Modbus Abfrage (Modbus-Flex-Getter)

In diesem Block wird die IP und der Port des Fronius Wechselrichters eingegeben.

Abbildung 5: Konfiguration des Modbus Flex Getter

5. Parsing Funktion

```
1 msg.payload = {  
2   P_PV_AC : msg.payload.buffer.readFloatBE(40,41,42,43),  
3   P_PV_DC : msg.payload.buffer.readFloatBE(72,73,74,75),  
4   E_PV_AC : msg.payload.buffer.readFloatBE(60,61,62,63)/1000,  
5 }  
6 return msg;
```

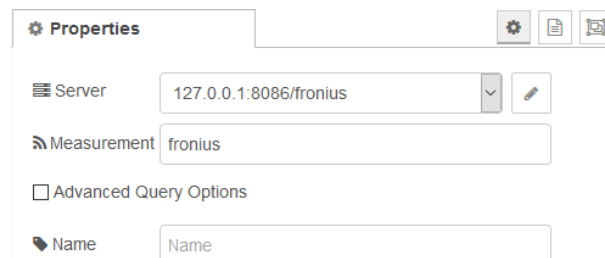
Abbildung 6: Parsing Funktion 1

```
1 var Scale_Factor_I = msg.payload.buffer.readInt16BE(0,1);  
2 var Scale_Factor_U = msg.payload.buffer.readInt16BE(2,3);  
3 var Scale_Factor_P = msg.payload.buffer.readInt16BE(4,5);  
4 var Scale_Factor_E = msg.payload.buffer.readInt16BE(6,7);  
5 msg.payload =  
6 {  
7   E_PV_DC_String_1 : msg.payload.buffer.readUInt32BE(40,41,42,43)*Math.pow(10.0, Scale_Factor_E)/1000,  
8   E_PV_DC_String_2 : msg.payload.buffer.readUInt32BE(80,81,82,83)*Math.pow(10.0, Scale_Factor_E)/1000,  
9   I_PV_DC_String_1 : msg.payload.buffer.readUInt16BE(34,35)*Math.pow(10.0, Scale_Factor_I),  
10  I_PV_DC_String_2 : msg.payload.buffer.readUInt16BE(74,75)*Math.pow(10.0, Scale_Factor_I),  
11 }  
12 return msg;
```

Abbildung 7: Parsing Funktion 2

Hier werden die empfangenen Daten (in Hex) durch Parsingfunktionen in die richtige Form gebracht und den Variablen zugeordnet.

6. Schreiben in Datenbank



The screenshot shows a 'Properties' window with the following fields:

- Server:** 127.0.0.1:8086/fronius
- Measurement:** fronius
- Advanced Query Options:** ☐
- Name:** Name

Abbildung 8: Einstellungen zum Schreiben in Datenbank

Es werden die Datenbank Informationen – IP / Datenbankname sowie eine Bezeichnung der Messreihe benötigt.

7. Debugging

Diente zur Ansicht der Modbus Antworten zur Entwicklung des Codes.

3.3 Einrichtung von Grafana

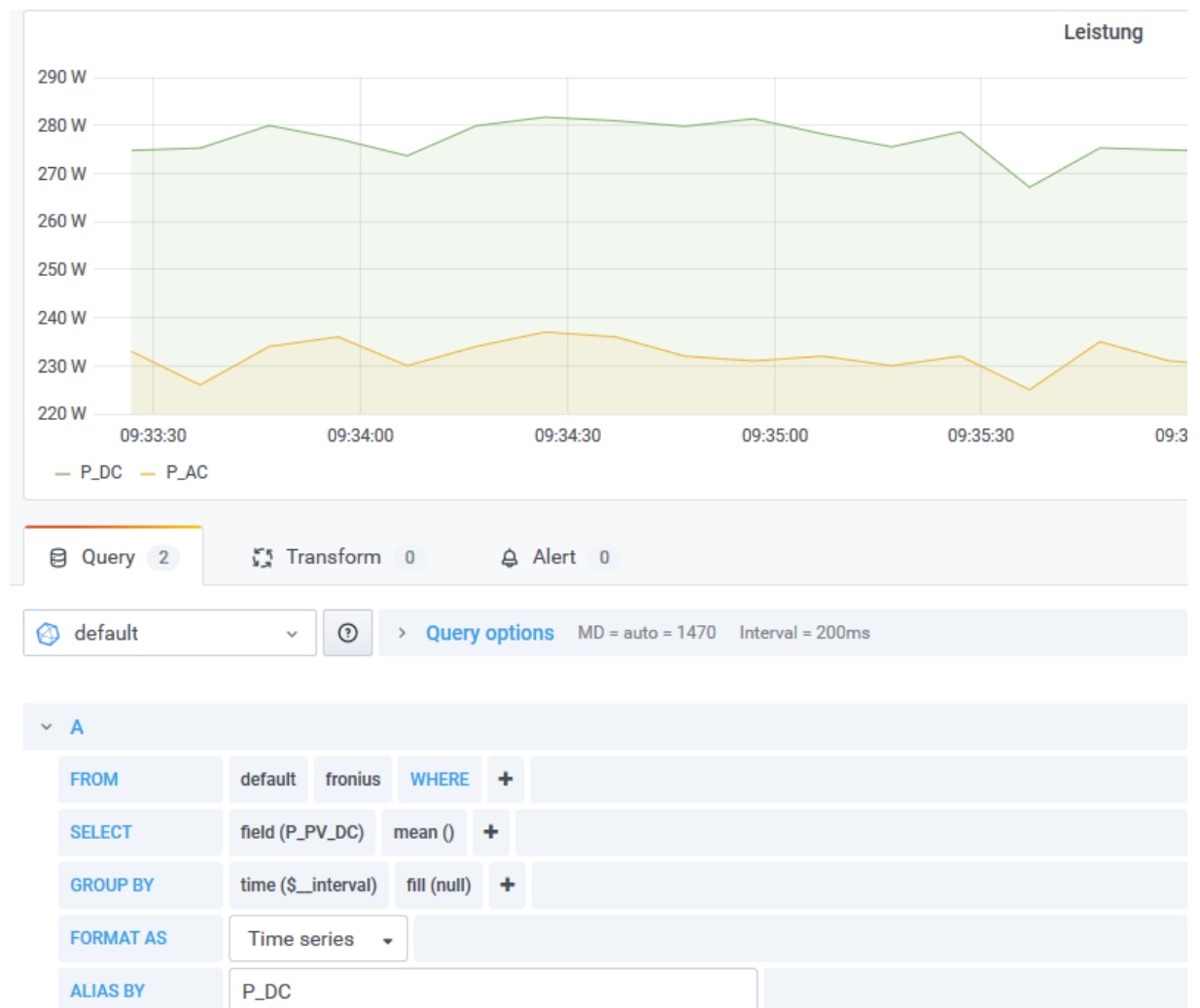


Abbildung 9: Grafana Einstellungen für Verlaufsgraphen

Verlaufsgraphen werden in Grafana wie in Abbildung 9 dargestellt abgefragt.

Die Query ist dabei abhängig, wie die Daten in Influx vorliegen.

Der untere Bereich in der Abbildung ist der s.g. Query Editor mit dem Anfragen zusammengestellt werden können.

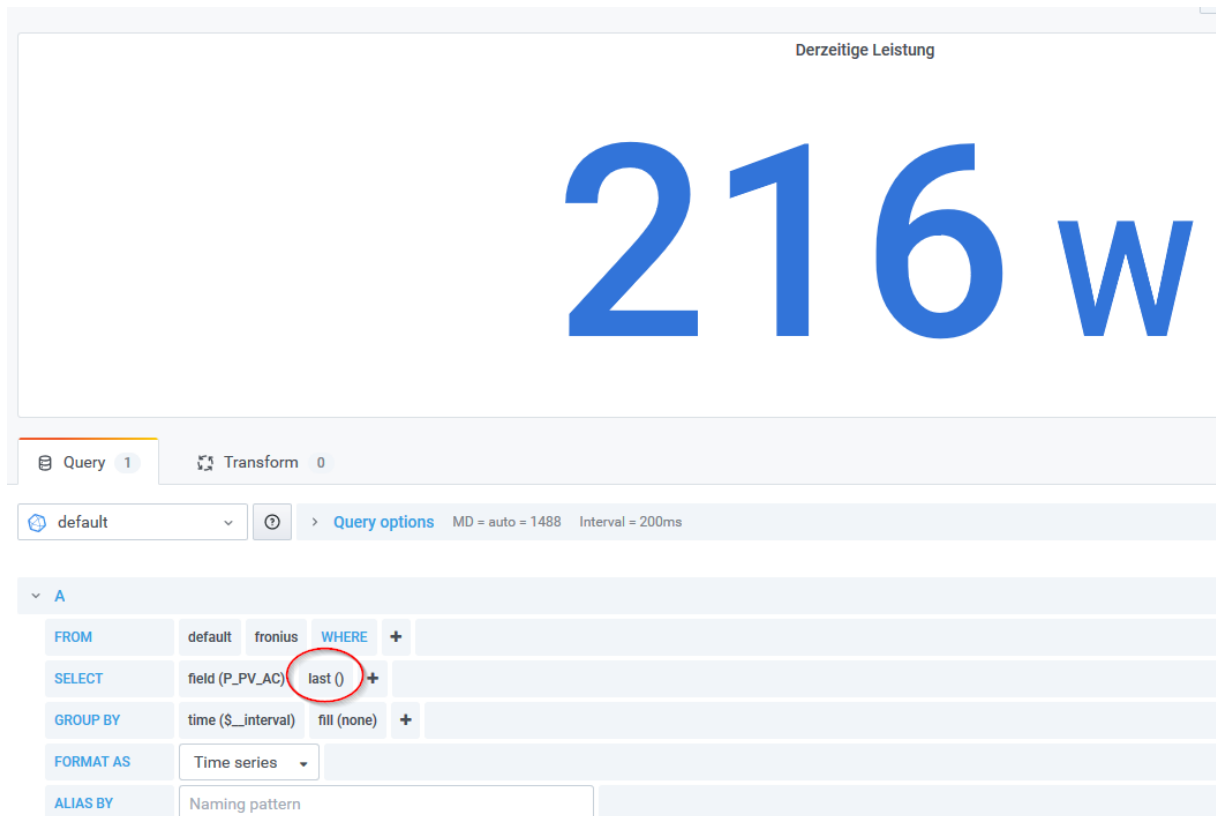


Abbildung 10: Einstellung für Zahlenwerte

Zur Darstellung von Zahlenwerten wird die Visualisierung von Graph auf Stat geändert. Die Abfrage aus der Datenbank zieht dabei den letzten Wert.

Da sich jedoch diese Abfrage auf das angezeigte Intervall bezieht, wird also nur der letzte Wert aus dem Intervall angezeigt. Damit auch tatsächlich der derzeit aktuelle Wert angezeigt wird, muss die Abfrage aus der Datenbank mit einem festen Zeitintervall eingestellt werden. Dazu muss aber die Abfrage in Textform eingegeben werden. Die obige Abfrage sieht in Textform wie folgt aus:

```
SELECT last("P_PV_AC") FROM "fronius" WHERE $timeFilter GROUP BY time($interval)
```

Dies wird wie folgt geändert:

```
SELECT last("P_PV_AC") FROM "fronius" WHERE time > now() -2m and time < now()
```

Grafana fragt also immer den letzten Wert der vergangenen 2 Minuten ab.

4 Abfrage Smartmeter über Wechselrichter

Beim Fronius werden die Smartmeter über Modbus RTU an den Datamanager angebunden.

Es muss die GeräteID des Smartmeters ermittelt werden (sollte 240) sein, konnte bisher noch nicht getestet werden.

Mit der unteren Abfrage kann man die Device IDs herauslesen.

http://141.60.106.94/solar_api/v1/GetActiveDeviceInfo.cgi?DeviceClass=System