



Video Meeting vom 5.5.2021  
Makerspace Reinach

# System Setup

## Hardware:

**1) Raspberry PI 3B+** (<https://www.raspberrypi.org>)

## Software Komponenten:

**2) Node-Red** (<https://nodered.org>)

**3) InfluxDB** (<https://www.influxdata.com>)

**4) Grafana** (<https://grafana.com>)

# Beispiel Szenarien

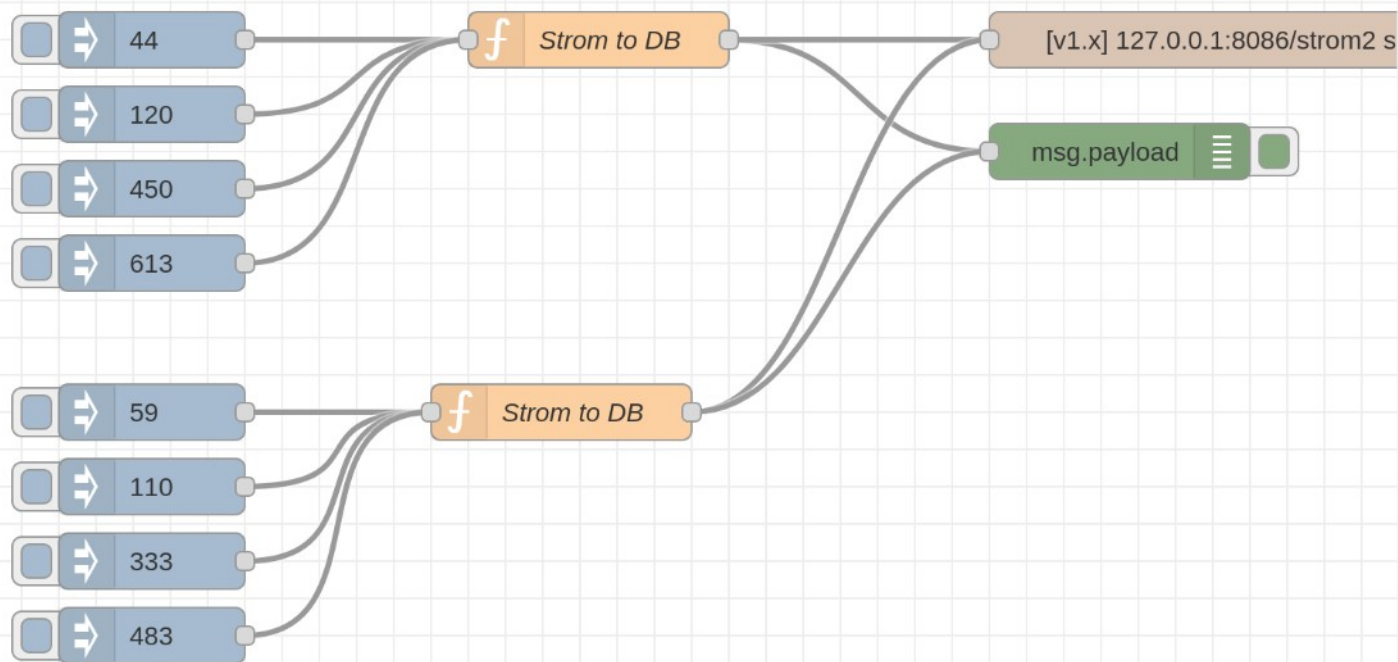
- 1) Neue Datenbank "strom3" mit Node-Red
- 2) Aufbewahrungszeit einrichten Beispiel: Stromproduktion und Verbrauch "strom2"
- 3) Daten verdichten Beispiel: RPI CPU
- 4) Dashboard und Alerts in Grafana

# 1) Neue Datenbank "strom3" und Node-Red / InfluxBD anlegen

```
pi@pi3w:~ $ influx
Connected to http://localhost:8086 version 1.6.4
InfluxDB shell version: 1.6.4
> create database strom3
```

```
> use strom3
Using database strom3
>
```

# 1) Neue Datenbank "strom3" und Node-Red / mit Node-Red Test Daten schreiben



current flow

```
4/25/2021, 10:07:37 PM node: 2853a862.5634d
msg.payload : array[1]
  array[1]
    0: object
      PROD: 0
      USE: 59

4/25/2021, 10:11:05 PM node: 2853a862.5634d
msg.payload : array[1]
  array[1]
    0: object
      PROD: 0
      USE: 333

4/25/2021, 10:11:06 PM node: 2853a862.5634d
msg.payload : array[1]
  array[1]
    0: object
      PROD: 120
      USE: 0
```

# 1) "strom3" / mit Node-Red Live Daten in die DB schreiben

```
1 var mysolarprod = msg.payload;
2 if ( mysolarprod === undefined){
3     mysolarprod = 0;
4 }
5 msg.payload = [{"PROD": mysolarprod, "USE": 0}];
6 return [msg];
```

Server [v1.x] 127.0.0.1:8086/strom3

Measurement solar

4/25/2021, 9:43:38 PM node: 2853a862.5634d

msg.payload : array[1]

▼ array[1]

▼ 0: object

PROD: 44

USE: 0

```
> show measurements
name: measurements
name
----
solar
```

Tabelle

```
> select * from solar
name: solar
time                PROD SolarPVA
----                -
1619379817972484530 44
1619380547298094116 120
1619380549341254812 613
1619381083103792335 613
1619381089710085337 44
```

Datum lesbar

```
> precision rfc3339
> select * from solar
name: solar
time                PROD SolarPVA
----                -
2021-04-25T19:43:37.97248453Z 44
2021-04-25T19:55:47.298094116Z 120
2021-04-25T19:55:49.341254812Z 613
2021-04-25T20:04:43.103792335Z 613
2021-04-25T20:04:43.710085337Z 44
```

## 2) Analyse Historie Daten "strom(2)" / DB Grösse und Einträge

```
pi@pi3w:~ $ sudo du -sh /var/lib/influxdb/data/*
124K   /var/lib/influxdb/data/auto
88K    /var/lib/influxdb/data/gas
88K    /var/lib/influxdb/data/gps
47M    /var/lib/influxdb/data/_internal
3.1M   /var/lib/influxdb/data/strom
96K    /var/lib/influxdb/data/strom2
84K    /var/lib/influxdb/data/strom3
5.0M   /var/lib/influxdb/data/temp
```

Solar Produktion und Stromverbrauch seit April 2020  
=> 3.1MB

```
> select * from solar limit 3
name: solar
time                SolarPVA
----                -
2020-04-01T13:54:56.432181346Z 3537
2020-04-01T13:56:05.422217742Z 3537
```

Solar Produktion und Stromverbrauch seit April 2020  
127'819 Einträge

```
> select count(*) from solar
name: solar
time                count_SolarPVA count_SolarUSE count_stromVERBR
----                -
1970-01-01T00:00:00Z 70507          70683          127819
```

## 2) Analyse Historie Daten "strom2" / Hilfsmittel

```
> select * into solar_test from solar
name: result
time          written
----          -
1970-01-01T00:00:00Z 198506
```

Measurement kopieren

```
> drop measurement solar_test
> select count(*) from solar_test
```

Measurement löschen



## 2) Analyse Historie Daten "strom2" / Retention Policy

```
CREATE RETENTION POLICY "solar_test_oneyear" ON "strom2" DURATION 52w REPLICATION 1
```

```
> show retention policies
```

name	duration	shardGroupDuration	replicaN	default
autogen	0s	168h0m0s	1	true
solar_test_oneyear	8736h0m0s	168h0m0s	1	false

```
> select count(*) from solar_test_oneyear
```

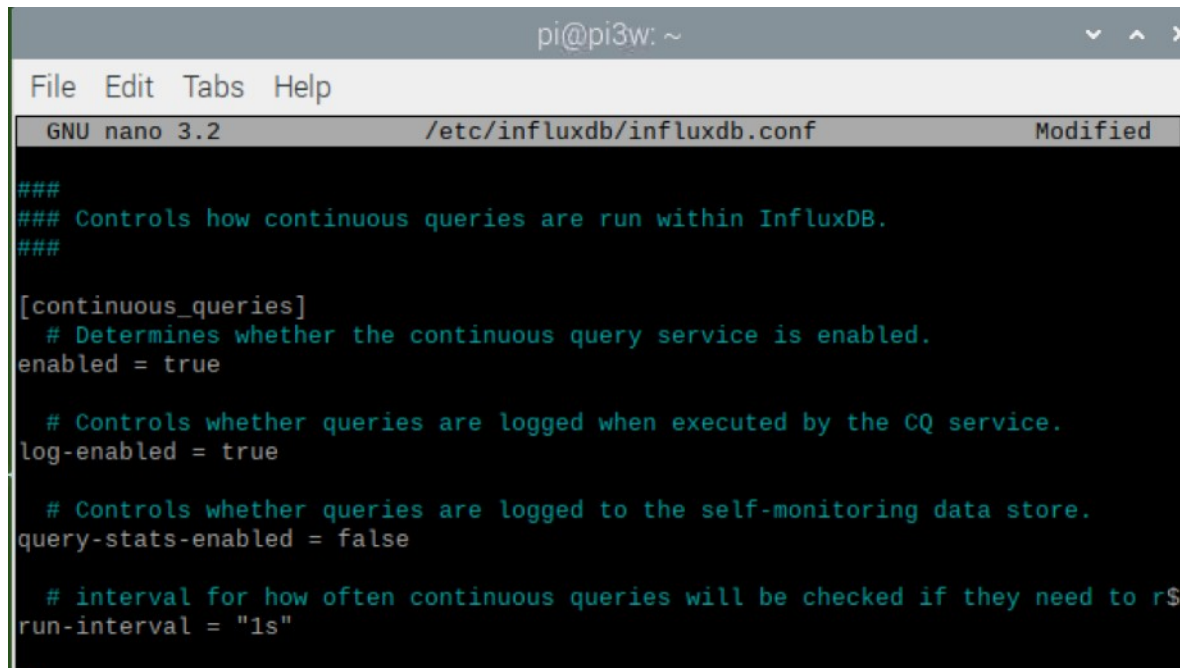
```
name: solar_test_oneyear
```

time	count_PROD	count_SolarPVA	count_SolarUSE	count_StromVERBR	count_TESTTAG	count_USE
1970-01-01T00:00:00Z	16	375	375	757	7	16

Um Daten mit dieser Policy zu speichern entweder die Messwerte bei INTO angeben oder die Regel mit dem Namen "DEFAULT" als Standardregel festlegen.

# 3) Daten verdichten Beispiel: RPI CPU / Continuous Query Config

```
sudo nano /etc/influxdb/influxdb.conf
```



```
pi@pi3w: ~
File Edit Tabs Help
GNU nano 3.2 /etc/influxdb/influxdb.conf Modified
###
### Controls how continuous queries are run within InfluxDB.
###
[continuous_queries]
# Determines whether the continuous query service is enabled.
enabled = true

# Controls whether queries are logged when executed by the CQ service.
log-enabled = true

# Controls whether queries are logged to the self-monitoring data store.
query-stats-enabled = false

# interval for how often continuous queries will be checked if they need to r$
run-interval = "1s"
```

```
$ sudo systemctl restart influxdb
```

# 3) Daten verdichten Beispiel: RPI CPU / Check Field Types

```
> show field keys
name: rpi3w
fieldKey fieldType
-----
CPU      string
```

!

```
> show field keys
name: rpi3w
fieldKey fieldType
-----
CPU      float
```

OK

### 3) Daten verdichten Beispiel: RPI CPU / Continuous Query + Retention Policy

```
CREATE CONTINUOUS QUERY "rpi3w_test_60s" ON
"rpi" BEGIN SELECT mean("CPU") AS "mean_CPU",
max("CPU") AS "max_CPU", min("CPU") AS "min_CPU"
INTO rpi3w_test_onemin FROM "rpi3w" GROUP BY
time(60s) END
```

```
> show continuous queries
```

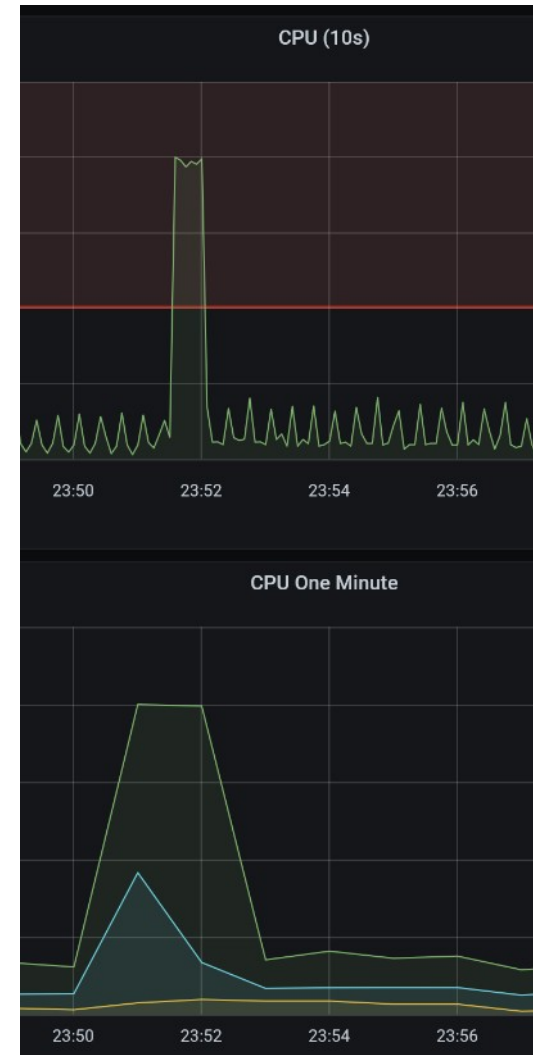
```
name: rpi
name          query
-----
rpi3w_test_60s CREATE CONTINUOUS QUERY rpi3w_test_60s ON rpi BEGIN SELECT
n(CPU) AS min_CPU INTO rpi.autogen.rpi3w_test_onemin FROM rpi.autogen.rpi3
```

```
DROP CONTINUOUS QUERY rpi3w_test ON rpi
```

# 3) Daten verdichten Beispiel: RPI CPU / Daten vergleichen


361 vs. 31

```
> select count(*) from rpi3w
name: rpi3w
time count_CPU
-----
0     361
> select count(*) from rpi3w_test_onemin
name: rpi3w_test_onemin
time count_max_CPU count_mean_CPU count_min_CPU
-----
0     31           31           31
```



# 4) Dashboard und Alerts in Grafana

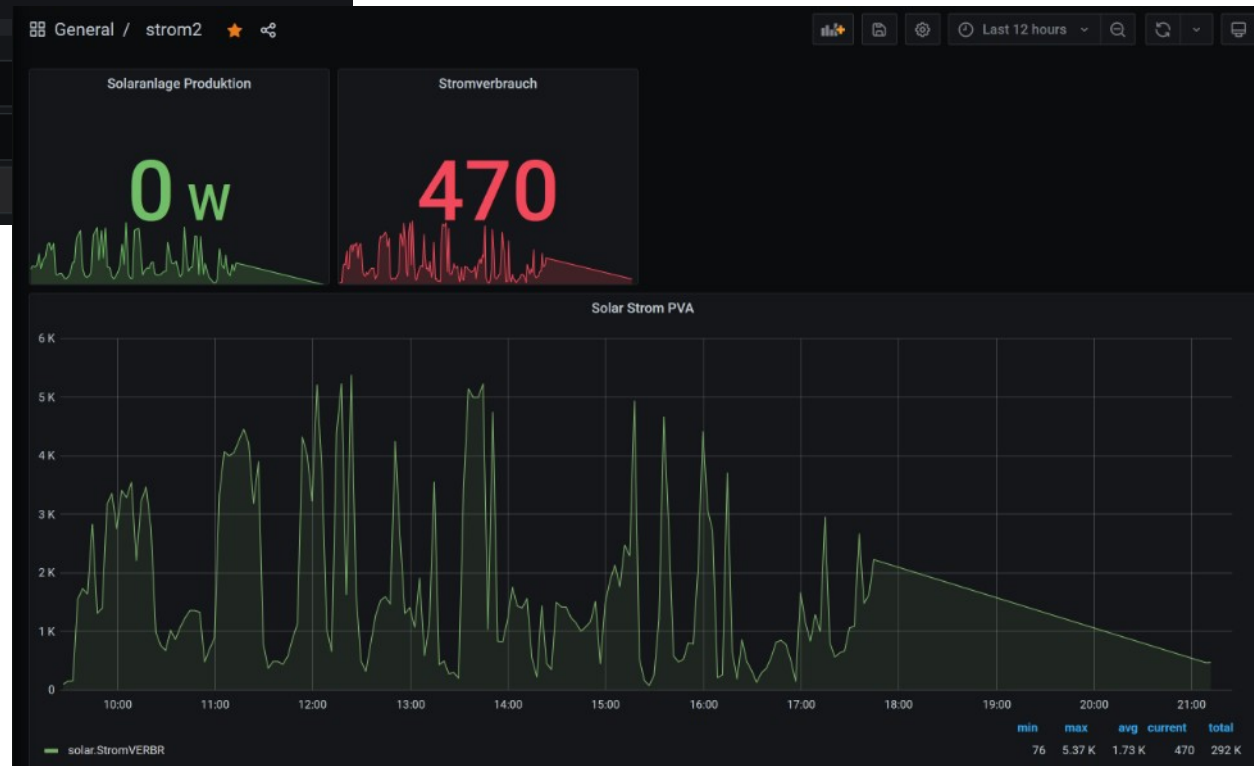
## DB Setup & Dashboard

 **Data Sources / strom2**  
Type: InfluxDB

Settings

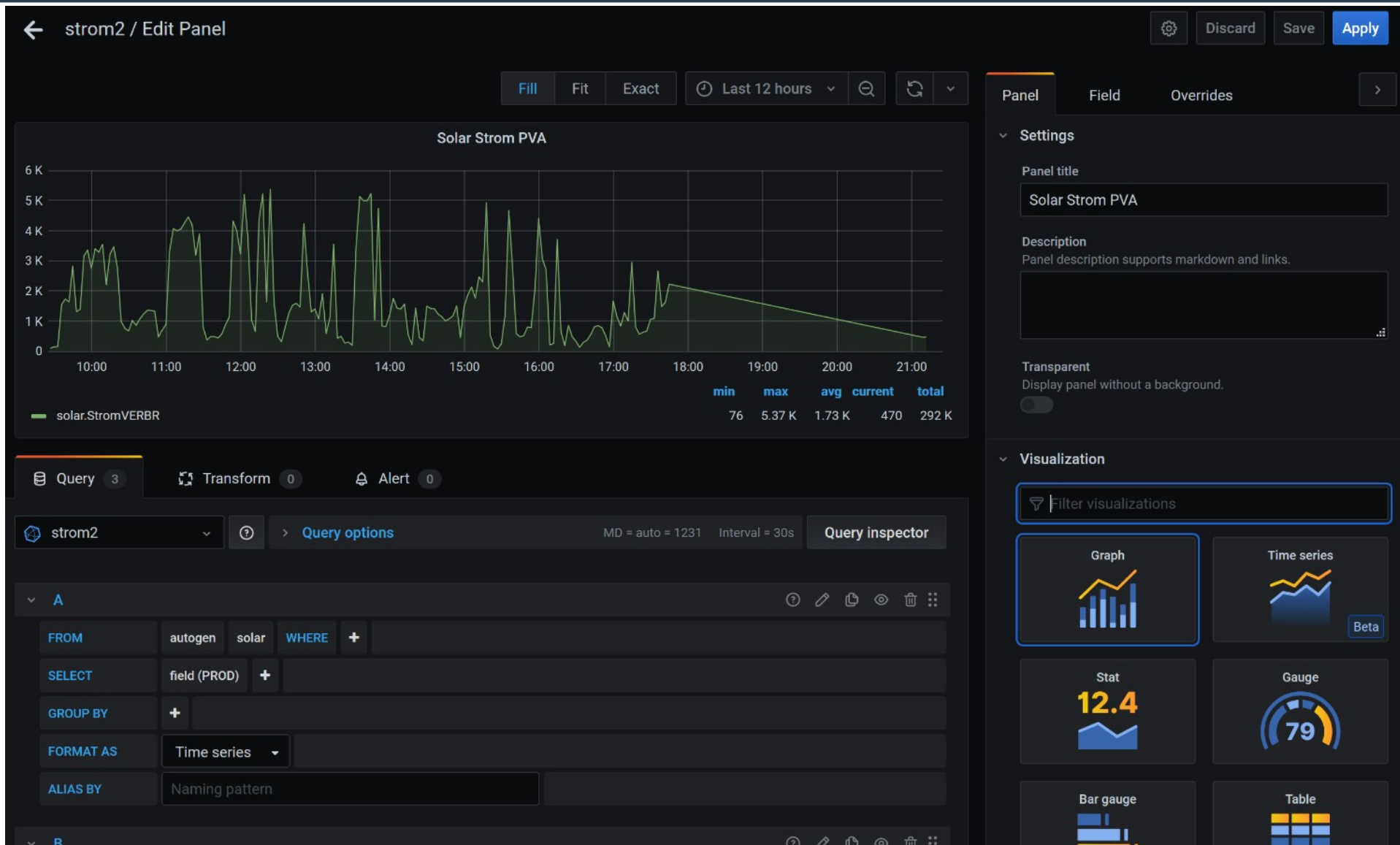
Name  Default

Database	strom2
User	admin
Password	configured



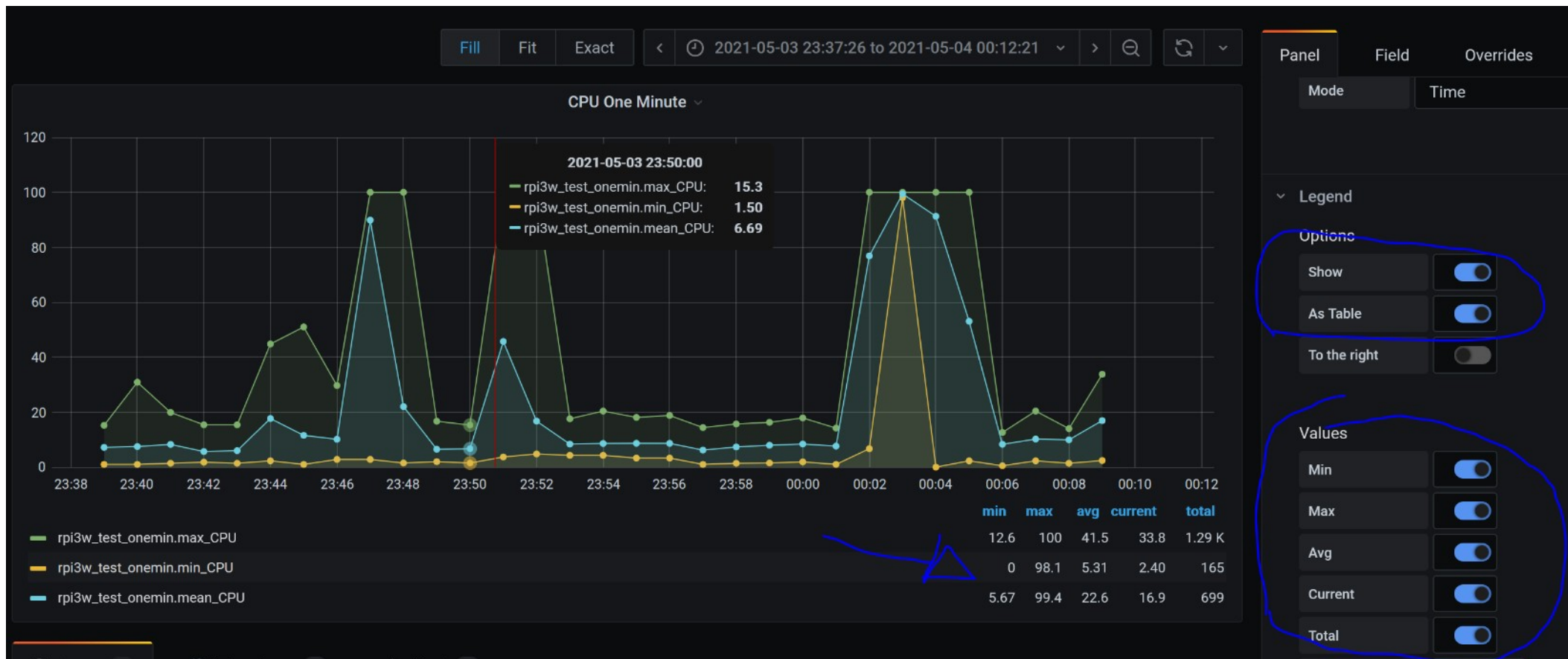
# Dashboard und Alerts in Grafana

## Dashboard Design



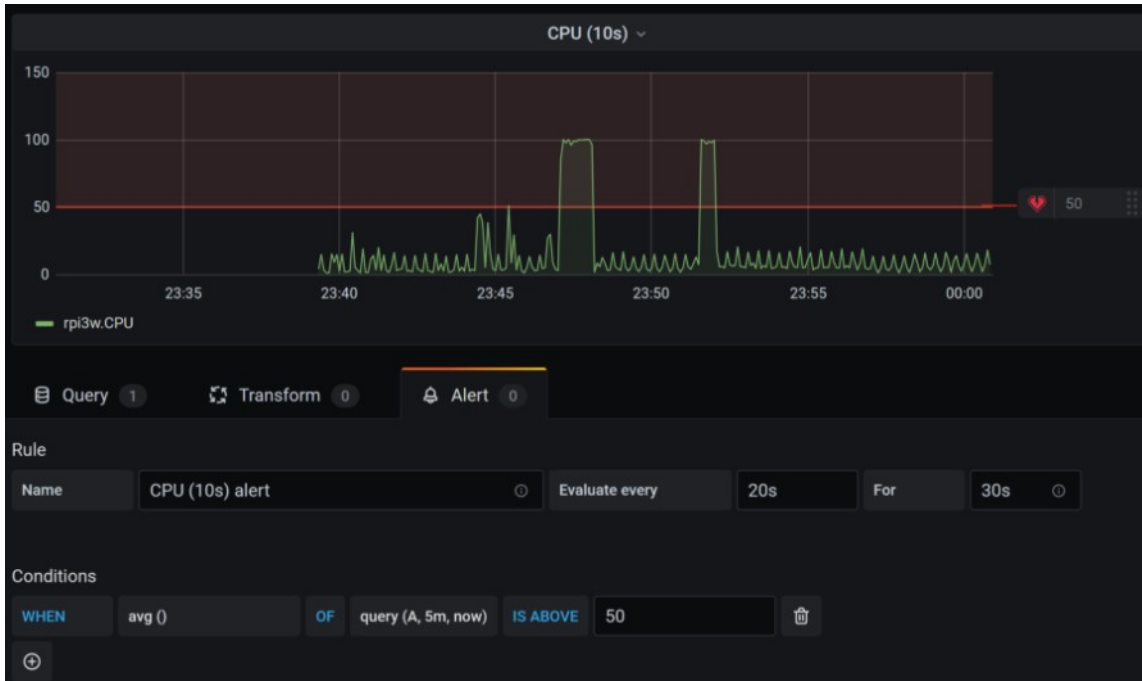
# Dashboard und Alerts in Grafana

## Werte Tabellen





# Dashboard und Alerts in Grafana / Alerts



The 'Notifications' configuration panel shows the alert is configured to be sent to 'Telegram'. The message content is 'RPI3 CPU over 50%'.



The screenshot shows a Telegram chat interface. At the top, it says '4. Mai'. There are two messages from Grafana. The first message is a warning: 'No image renderer available/installed' and '[Alerting] CPU (10s) alert'. The second message is an 'OK' confirmation: '[OK] CPU (10s) alert'. Both messages include the alert details and a URL to view the dashboard.

# Data Links in Grafana

The image shows a Grafana dashboard with two panels: "Solaranlage Produktion" (Solar plant production) and "Stromverbrauch" (Electricity consumption). The production panel shows a green line graph with a large "0 w" indicator. The consumption panel shows a red line graph with a large "363" indicator. A tooltip for the consumption panel shows the field name "solar\_test\_oneyear.StromVERBR".

On the right, the configuration panel for the "Field" is open, showing the following settings:

- Panel:** solar\_test\_oneyear.StromVERBR
- Field:** solar\_test\_oneyear.StromVERBR
- Overrides:** >
- Thresholds mode:** Percentage means thresholds relative to min & max. Options: Absolute (selected), Percentage.
- Value mappings:** + Add value mapping
- Data links:**
  - Details: <http://pi3w:3000/d/PrIakHRRz/strom2?viewPanel=2&orgl...>
  - + Add link



**Fragen?**