Gefördert durch:

Bundesministerium
für Wirtschaft
und Klimaschutz

aufgrund eines Beschlusses
des Deutschen Bundestages

Technische
Hochschule
Rosenheim

MonSec

Datenaufnahme – Monitoringschauwand

IoT-Messtechnik – Raspberry Pi

Anleitung

Technische Hochschule Rosenheim

Bearbeiter: Markus Hartmann

27.06.2022

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

In dieser Dokumentation wird der Aufbau der Sensorabfrage, Datenspeicherung und Visualisierung der Monitoringschauwand am Rosenheimer Technologiezentrum für Energie und Gebäude (roteg) beschrieben.

Bei der Monitoringschauwand handelt es sich um einen simulierten Heizkreis mit verschiedenen, in der Gebäudetechnik üblichen, Sensoren und Messgeräten.

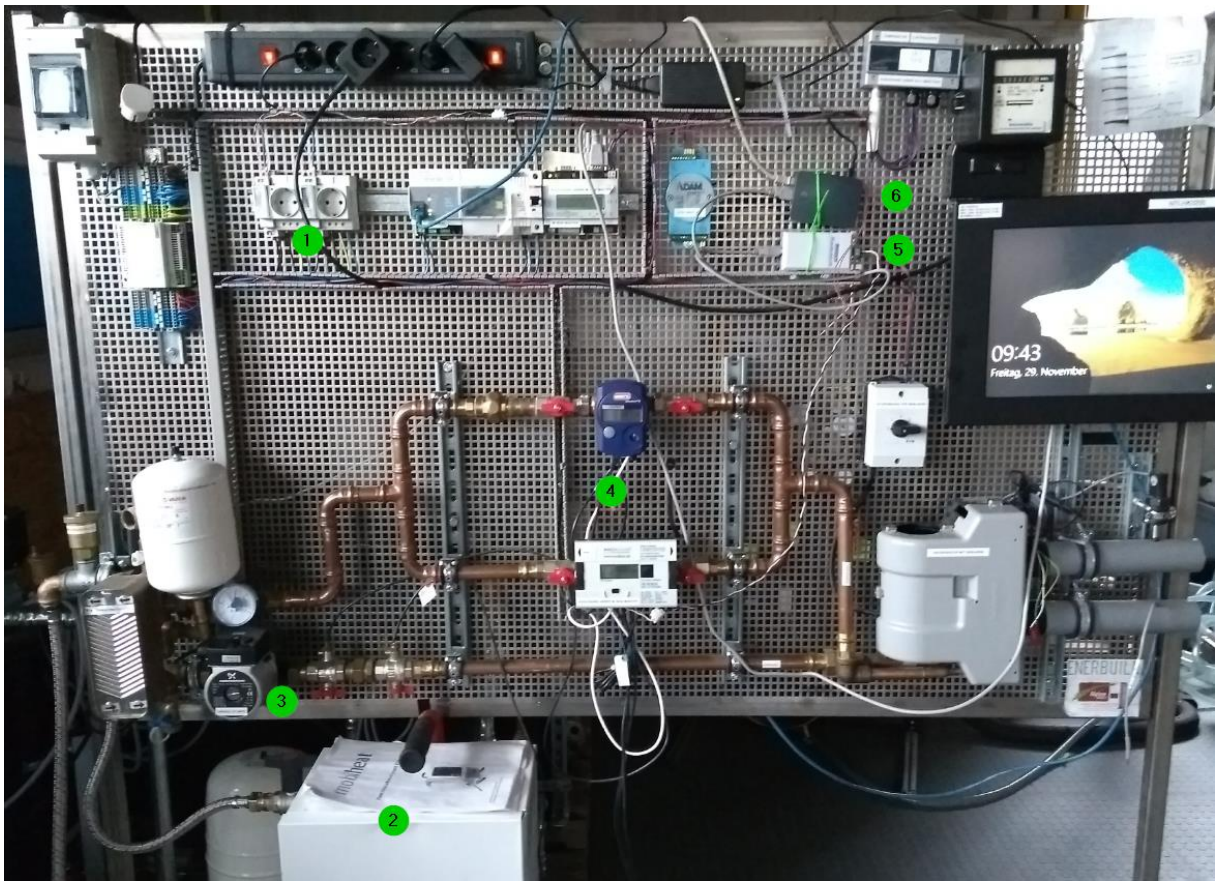

Abbildung 1: Darstellung der Monitoringschauwand

In dieser Doku wird vor allem auf die Datenabfrage der MBus Zähler eingegangen. Diese werden mit dem IoT Device Raspberry Pi (6) (per NodeRed) ausgelesen und in eine lokale Datenbank (InfluxDB) geschrieben. Eine Visualisierungsumgebung für die Daten ist in Form des Programms Grafana ebenfalls auf dem RaPi installiert.

Bei den hier relevanten Sensoren handelt es sich um zwei Energiemessgeräten (1) an welchen die elektrische Heizung (2) und die Heizkreispumpe (3) separat gemessen werden. In dem heizkreis sind parallel zwei Wärmemengenzähler (4) verbaut.

2 Konfiguration der Datenabfrage

Die Datenabfrage erfolgt über einen USB- Pegelwandler, der die MBus Zähler direkt mit dem Raspberry Pi verbindet.

Die Datenabfrage erfolgt über NodeRed. NodeRed wird unter Angabe der IP des Raspberry Pimit dem Port 1880 im lokalen Netzwerk über den Browser erreicht.

Hierfür werden die folgenden „Paletten“ benötigt. Die Paletten sind die Bibliotheken in NodeRed, welche den Standardumfang um zusätzliche Programmierblöcke erweitern.

- Mbus Abfrage
  https://flows.nodered.org/node/node-red-contrib-m-bus
- Speichern in der InfluxDB
  https://flows.nodered.org/node/node-red-contrib-influxdb

Der grundsätzliche Flow für die Datenabfrage sieht wie folgt aus:

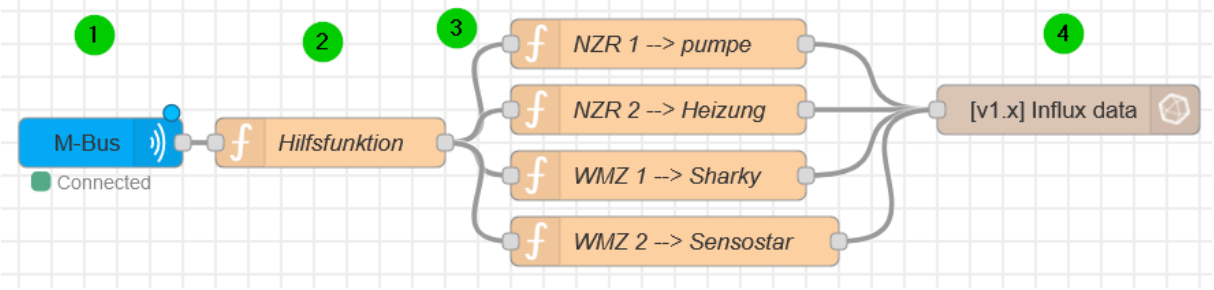

Abbildung 2: Übersicht des NodeRed Flows

2.1 Konfiguration des Mbus Connectors

Für das erstmalige einrichten wird der Beispiel Flow der contrib-m-bus Palette benötigt, um das Busnetz nach verfügbaren Geräten zu scannen und die Geräte-Ids herauszulesen. Danach werden die Geräte Zyklisch durch den Connector abgefragt. Zur Konfiguration des Connectors werden folgende Einstellungen gewählt.

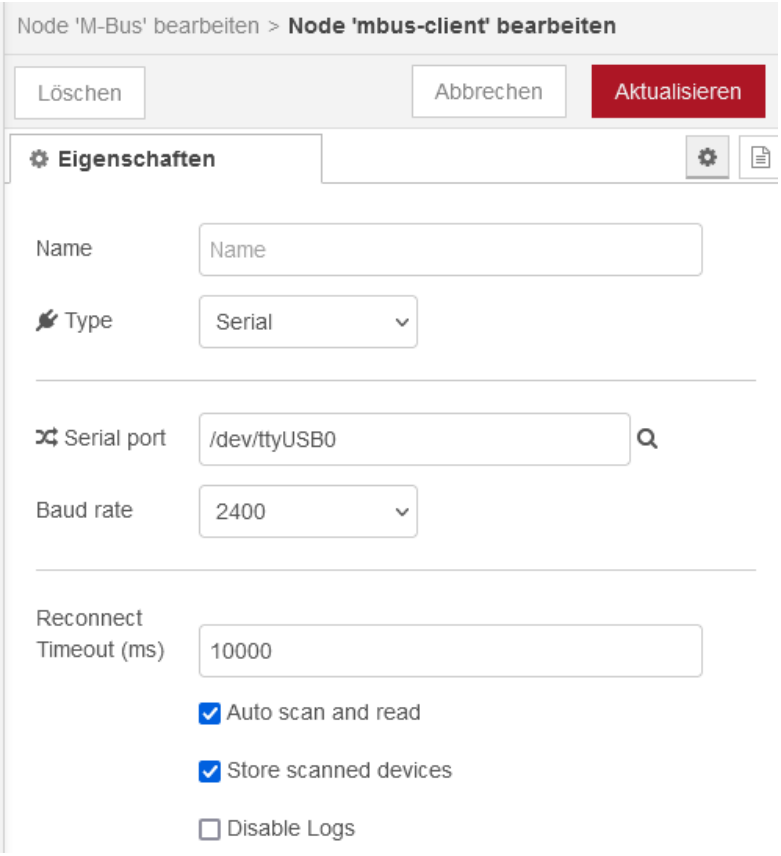

Abbildung 3: Konfiguration Mbus Connector

Für den Serial Port wird der USB-Port des RaPi angegeben.

Die Baud rate ist abhängig von Einstellungen an den Geräten und dem Pegelwandler.

Die Reconnect Timeout Einstellung ist auch gleichzeitig der Zyklus für die Abfrage der Geräte und wurde für die Demonstration der Schauwand auf 10 Sekunden eingestellt. Das Automatisierte Auslesen wird durch die Option „Auto scan and read“ eingeschaltet. Die Option „Store scanned decices“ erlaubt es den Connector ohne scannen des gesamten MBus zu verwenden.

Es gilt zu beachten, dass beim Mbus jedes Gerät separat abgefragt werden muss. Von daher ist auch das gleichzeitige Auslesen nicht möglich. Daher sollte für die Abfrage genügend Zeit (>2 Sekunden) eingeplant werden.

Wird ein Gerät abgefragt, so entsteht eine NodeRed Message im JSON (JavaScipt Object Notation) Format ausgegeben. Ein Beispiel für eine solche Nachricht sieht, in stark gekürzter Form wie folgt aus:

```
  "topic": "mbDeviceUpdated",
  "payload": {
    "SlaveInformation": {
      "Id": 30102556,
    },
    "DataRecord": [
      {
        "Unit": "Energy (Wh)",
        "Value": 3455620,
      },
      {
        "Unit": "1e-1  V",
        "Value": 2334,
      },
      {
        "Unit": "1e-1  A",
        "Value": 0,
      },
      {
        "Unit": "Power (W)",
        "Value": 34,
      }],
  },
}
```


Die Nachricht erhält weitaus mehr Informationen, welche aber für den weiteren Gebrauch nicht relevant sind und aus Gründen der Übersichtlichkeit entfernt wurden.

Die verwendeten Bestandteile der Nachricht sind wie folgt:

- Payload.SlaveInformation.Id → Die Sekundäradresse des Messgerätes
- Payload.DataRecord[ ].Unit → Die Einheit des Messpunktes
- Payload.DataRecord[ ].Value → Der Wert des Messpunktes

2.2 Konfiguration der Hilfsfunktion

In der Hilfsfunktion werden die Daten vorsortiert, um den Code der Parser Funktionen zu reduzieren. Eine IF Abfrage stellt noch sicher, dass nur Daten aus aktualisierten Sensordaten weiterverarbeitet werden. Der Code für die Funktion sieht wie folgt aus:

```
// Filtern der Nachrichten nach "Device Updated"
// Umsortieren der payload für leichteres Erreichen
if (msg.topic == "mbDeviceUpdated"){
msg.Device = msg.payload.SlaveInformation.Id;
msg.payload = msg.payload.DataRecord;
return msg;
};
```

Abbildung 4. Code der Hilfsfunktion

Es werden die ID des Gerätes sowie die Messdaten extrahiert und weitergegeben.

2.3 Konfiguration der Parser Funktionen

Die Parser Funktionen haben die Aufgabe, die Messdaten aus den Speicherregistern zu entnehmen. Für das Einspeichern der Daten in die InfluxDB müssen die Messdaten noch in eine entsprechende Form gebracht werden.

Grundsätzlich gibt es beim Parsen von Bus Daten in NodeRed mehrere Möglichkeiten. In diesem Fall wird für jedes Gerät eine eigenständige Funktion verwendet, welche die Nachrichten nach dem jeweiligen Gerät filtert. Ebenfalls kann die Funktionalität durch einen komplexeren Code vereinfacht und stark automatisiert werden. Im Sinne der Aufgabe einen Einstieg in die IoT-Messdatenaufnahme zu zeigen wurden die Funktionen bewusst einfach und rudimentär gehalten.

In der Folgenden Abbildung ist die Parser Funktion für einen der beiden Strommessgeräte (entspricht dem JSON Objekt aus Kapitel 2.1) dargestellt.

```
if (msg.Device == 30102556){
    msg.payload = [{
    Energy: msg.payload[0].Value,
    Power: msg.payload[4].Value,
    Voltage: msg.payload[2].Value/10,
    Amps: msg.payload[3].Value/10,
    },
    {
        Device: "Heizung"
    }];
    return msg;
}
```

Abbildung 5: Code einer Parsing Funktion

Die Nachricht wird durch eine IF Funktion anhand der sekundären Geräte Id gefiltert.

Die Messdaten des Gerätes werden anhand des Datenblattes, aus welchem die Registertabelle sowie die Einheiten und Skalierungsfaktoren entnommen werden, umgeformt.

So steht im Register Nr. Null der Wert für den Energiezähler mit einem Skalierungsfaktor von Eins. Dieser wird in ein Array der Message Payload mit der Variablen „Energy“ verschoben. Die restlichen Werte werden entsprechend umgewandelt, wobei bei den Werten für Volt und Ampere ein Skalierungsfaktor von $1 \cdot 10^{-1}$ verwendet werden muss.

Einschub:
Alle verwendeten Geräte liefern die Messwerte im Datenformat Integer. Somit ist keine Umwandlung des Datentyps nötig. Andere Geräte können Daten in Form von z.B. Float Zahlen liefert. Für diesen Fall muss der Messwert noch umgewandelt werden.

Das Einspeichern in die Influx erlaubt es, dem Messwerten noch Metadaten (Tag) mitzuliefern, welche im zweiten Array der Payload enthalten sein muss. So wird hier noch der Gerätetyp in Form des Tags „Device“ als String mitgeliefert. Dies erlaubt ein einfaches Filtern der Datenpunkte in der Grafana Visualisierungssoftware (siehe Kapitel 3)

2.4 Konfiguration des InfluxDB Connectors

Für das Einspeichern der Daten in die InfluxDB Datenbank wird der Connector wie folgt konfiguriert

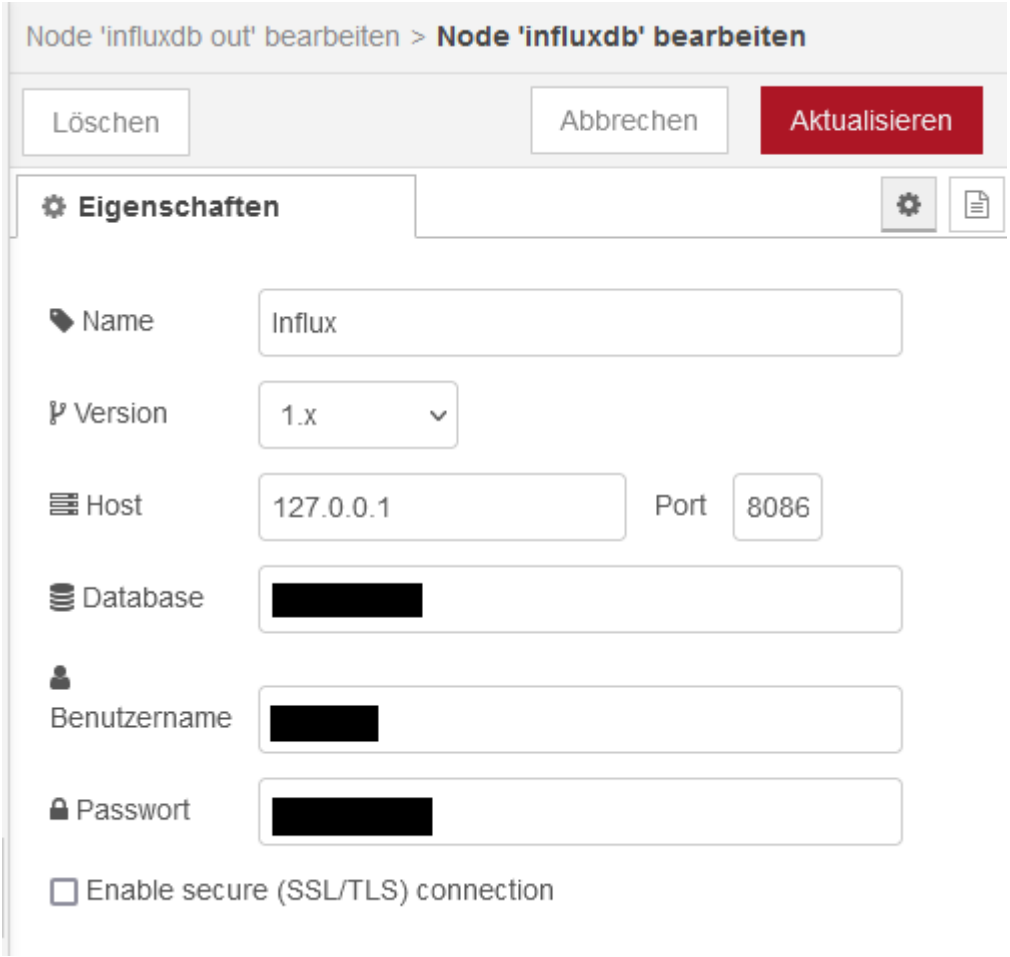

Abbildung 6: Konfiguration des InfluxDb Connectors

Da die Datenbank auf dem RaPi selbst läuft wird für die IP des Hosts die lokal IP 127.0.0.1 angegeben, der Port 8086 ist der Standard Influx Port.

Für die Database wird der Name einer erstellten Datenbank benötigt.

Es empfiehlt sich zur Absicherung der Datenbank einen Benutzer mit Passwort einzurichten.

Die Daten werden nach Vorgaben des Lineprotocols der InfluxDB entsprechend im JSON Format dem Koten zur Verfügung gestellt.

3 Konfigurierung der Visualisierung

Zur Visualisierung wird die Software Grafana verwendet, welche ebenfalls lokal auf dem RaPi läuft. Grafana wird unter Angabe der IP mit dem Port 3000 im lokalen Netzwerk über den Browser erreicht.

Für die Erstinstallation wird die Einstellung der Datenbank benötigt. Hierfür wird die IP der Datenbank (localhost) der Datenbankname sowie Benutzer und Passwort benötigt.

Zur Konfiguration der Abfragen aus der Datenbank (Querys) enthält Grafana einen einfachen Editor über welchen die Querys einfach konfiguriert werden können. In der folgenden Abbildung ist beispielhaft für einen Datenpunk die Eingaben der Query dargestellt.

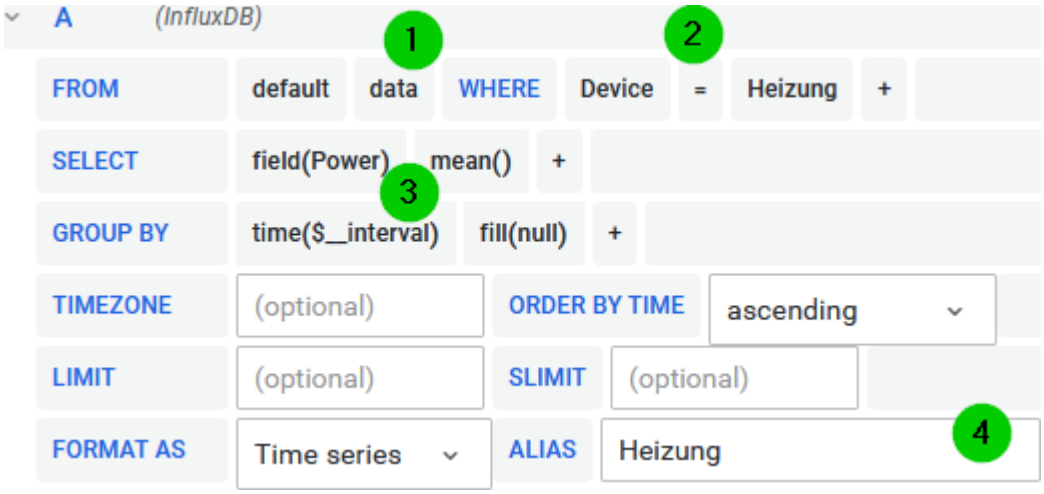

Abbildung 7: Konfiguration der Abfragequery

Es wird der Messpunkt „Leistung“ des Stromzählers der Heizung abgefragt. Hierzu sind folgende Eingaben nötig:

FROM:
Eingabe der Datenbank (default) und des Measurements (1) der Datenpunkte Durch den zusätzlichen Tag (2)„Device“ werden die Datenpunkte nach dem Messgerät gefiltert.

SELECT:
Als field (3) wird der Name des Messpunktes eingegeben. Es wird eine Aggregierung beziehungsweise Selektor Funktion benötigt.

GROUP By:

Enthält zusätzlich Funktionen zur Auswahl des Zeitraums. Durch den Standard Parameter erfolgt die Zeiteinstellung über das Dashboard in welchem die Visualisierung enthalten ist.

ALIAS:
Durch Eingabe des Alias wird der Legendenname eingegeben.