

Gefördert durch:



Bundesministerium
für Wirtschaft
und Klimaschutz

aufgrund eines Beschlusses
des Deutschen Bundestages

Technische
Hochschule
Rosenheim



MonSec

Datenabfrage eines MBus Wärmemengenzähler mit IoT-Technik

Im Projekt MonSEC- Monitoring Secure

TH Rosenheim

Bearbeiter: Markus Hartmann

Stand: 27.10.2022

Inhaltsverzeichnis

Inhaltsverzeichnis	1
Abbildungsverzeichnis	2
1 Einleitung	3
2 Hardware	4
2.1 Aufbau des Messkasten	4
2.2 Einrichten der Seriellen Schnittstelle.....	4
3 Konfiguration des Raspberry Pi	6
3.1 Konfiguration der Datenabfrage in Node-Red	6
3.1.1 Konfiguration des Mbus Connectors	6
3.1.2 Konfiguration der Hilfsfunktion	8
3.2 Konfiguration der Parser Funktionen.....	9
3.2.1 Konfiguration des InfluxDB Connectors	10
4 Konfigurierung der Visualisierung	11

Abbildungsverzeichnis

Abbildung 1: Aufbau und Beschreibung des Messkastens	4
Abbildung 2: Raspberry Pi mit RS232 Erweiterung	5
Abbildung 3: Übersicht des NodeRed Flows	6
Abbildung 4: Konfiguration des MBus Connectors	7
Abbildung 5: Code der Hilfsfunktion	8
Abbildung 6: Code einer Parsing Funktion	9
Abbildung 7: Konfiguration des InfluxDb Connectors	10
Abbildung 8: Konfiguration der Abfragequery	11

1 Einleitung

Diese Dokumentation beschreibt den Aufbau eines Datenabfrage eines Wärmemengenzählers und Stromzählers über MBus durch einen Raspberry Pi. Es werden hierbei die IoT-Komponenten Nodered zum Programmieren, eine Influx Datenbank zum Speichern der Daten sowie das Visualisierungstool Grafana verwendet.

Die Anwendung für dieses System ist das Monitoring einer Wärmepumpe. Hierzu wird auf der nach der Wärmepumpe ein Wärmemengenzähler (Typ Sharky 775) in den Hydraulikkreislauf eingebaut und vor der Wärmepumpe ein Stromzähler in den Stromkreislauf eingebaut. Beide Geräte verfügen über M-Bus schnittstellen.

2 Hardware

2.1 Aufbau des Messkastens

Die Hardware, umgesetzt in einem Messkasten, besteht aus einem Raspberry 3B+ mit angeschlossener RS232 Platine sowie einem MBus Pegelwandler.

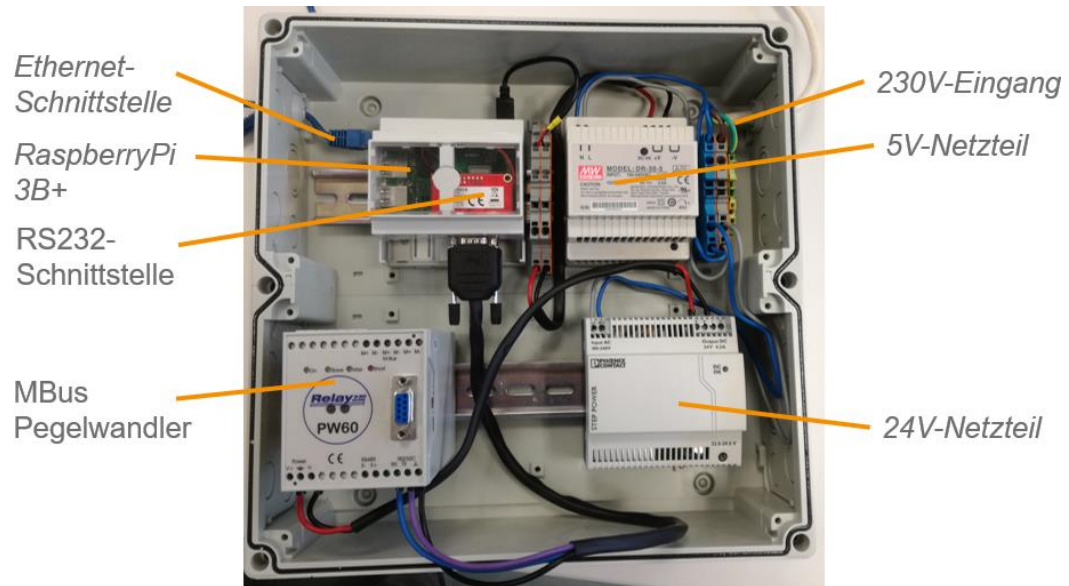


Abbildung 1: Aufbau und Beschreibung des Messkastens

Dabei dient das 24V Netzteil für die Versorgung des Pegelwandlers, das 5V Netzteil für die Versorgung des Raspberry Pi.

2.2 Einrichten der Seriellen Schnittstelle

Für die Serielle Schnittstelle wird die Erweiterungsplatine Renkforce RF-4011279 verwendet. Diese wird über das mitgelieferte Verbindungskabel an die GPIOs 02,06,08,10 angeschlossen

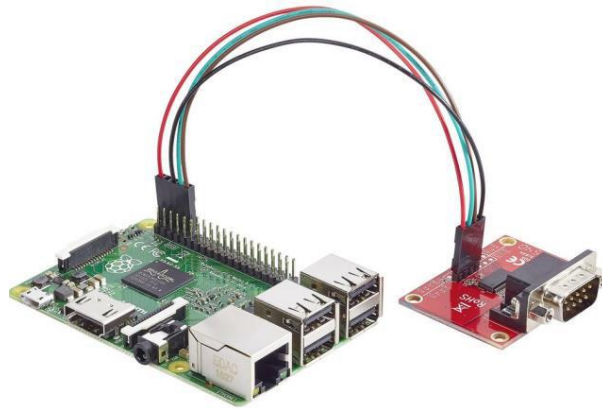


Abbildung 2: Raspberry Pi mit RS232 Erweiterung

Auf dem Raspberry muss die Serielle Schnittstelle erst eingerichtet und konfiguriert werden. Hierzu über den Befehl „sudo raspi-config“ das Konfigurationsmenü aufrufen und in den Interfacing Options → Serial zuerst bei der Frage nach dem „loginShell = Nein“ und bei der Frage nach Verwendung der „serial port hardware = Yes“ einstellen. Anschließend muss noch das vorhandene Bluetooth Modul deaktiviert und die UART Schnittstelle eingeschalten werden. Hierzu über den Befehl „sudo nano /boot/config.txt“ die Konfigurationsdatei bearbeiten. Am Ende der Datei muss die folgende Zeile eingefügt werden.

```
dtoverlay=pi3-miniuart-bt
```

An die RS232 Schnittstelle wird der Pegelwandler angeschlossen.

3 Konfiguration des Raspberry Pi

Als Betriebssystem des Raspberry Pi wird Raspian Buster lite in der neuesten Version verwendet. Es werden die Programme Node-Red, InfluxDB und Grafana benötigt.

3.1 Konfiguration der Datenabfrage in Node-Red

Die Datenabfrage erfolgt über NodeRed. NodeRed wird unter Angabe der IP des Raspberry Pi mit 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:

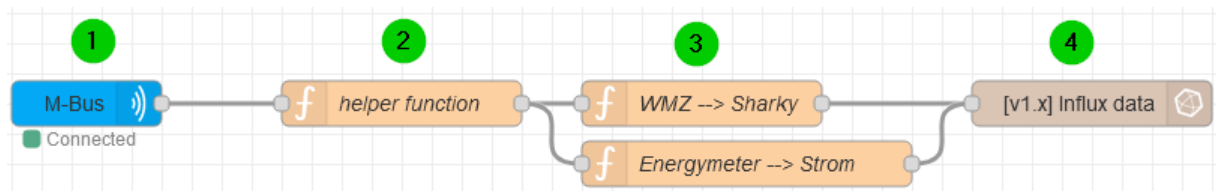


Abbildung 3: Übersicht des NodeRed Flows

3.1.1 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.

The screenshot shows the configuration page for a Node-RED node named 'mbus-client'. The page has a header with the title 'Node 'M-Bus' bearbeiten > Node 'mbus-client' bearbeiten' and three buttons: 'Löschen', 'Abbrechen', and 'Aktualisieren'. Below the header is a tab labeled 'Eigenschaften'. The configuration fields are as follows:

- Name:** A text input field containing the word 'Name'.
- Type:** A dropdown menu with 'Serial' selected.
- Serial port:** A text input field containing '/dev/ttyAMA0' with a search icon to its right.
- Baud rate:** A dropdown menu with '2400' selected.
- Reconnect Timeout (ms):** A text input field containing '10000'.
- Options:** Three checkboxes are listed:
 - ☒ Auto scan and read
 - ☒ Store scanned devices
 - ☐ Disable Logs

Abbildung 4: Konfiguration des MBus Connectors

Für den Serial Port wird die UART Schnittstelle angegeben. An dieser ist die RS232 Erweiterungsplatine softwareseitig angeschlossen

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. Für die Anwendung der Überwachung der Wärmepumpe wurde eine Zykluszeit von 10 Sekunden gewählt. 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 (JavaScript 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

3.1.2 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:

```
1 // Filtern der Nachrichten nach "Device Updated"
2 // Umsortieren der payload für leichteres Erreichen
3 if (msg.topic == "mbDeviceUpdated"){
4   msg.Device = msg.payload.SlaveInformation.Id;
5   msg.payload = msg.payload.DataRecord;
6   return msg;
7 };
```

Abbildung 5: Code der Hilfsfunktion

3.2 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 3.1.1) 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: "Energymeter"  
    }  
    ]];  
    return msg;  
}
```

Abbildung 6: 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.

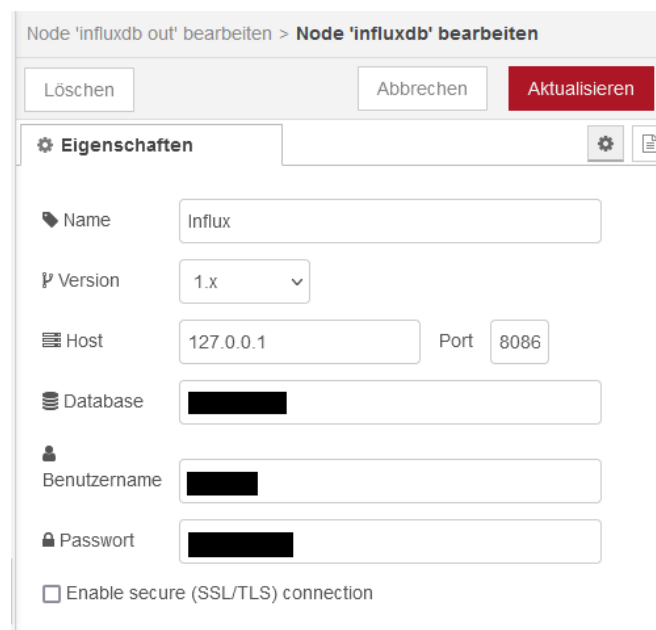
Einschub:

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 4)

3.2.1 Konfiguration des InfluxDB Connectors

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



The screenshot shows the configuration page for the 'influxdb out' node in Grafana. The page title is 'Node 'influxdb out' bearbeiten > Node 'influxdb' bearbeiten'. At the top, there are three buttons: 'Löschen', 'Abbrechen', and 'Aktualisieren'. Below the buttons is a section titled 'Eigenschaften' with a gear icon and a document icon. The configuration fields are as follows:

- Name:** Influx
- Version:** 1.x (dropdown menu)
- Host:** 127.0.0.1
- Port:** 8086
- Database:** [Redacted]
- Benutzername:** [Redacted]
- Passwort:** [Redacted]
- Enable secure (SSL/TLS) connection:** ☐

Abbildung 7: 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 Knoten zur Verfügung gestellt.

4 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 Datenpunkt die Eingaben der Query dargestellt.

FROM	default	data	WHERE	Device	=	Heizung	+
SELECT	field(Power)	mean()	+				
GROUP BY	time(\$__interval)	fill(null)	+				
TIMEZONE	(optional)	ORDER BY TIME	ascending				
LIMIT	(optional)	SLIMIT	(optional)				
FORMAT AS	Time series	ALIAS	Heizung				

Abbildung 8: 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 Aggregation 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.