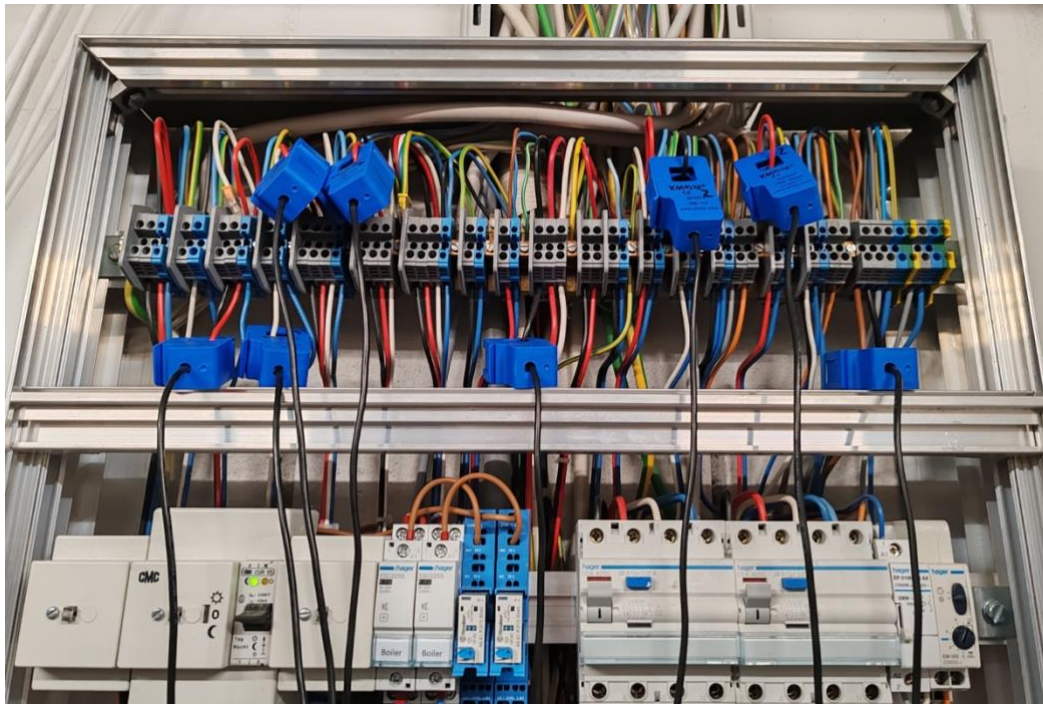


# Energieverbrauchsüberwachung von einem Einfamilienhaus mit CT-Sensoren, Mikrocontroller und Web-Dashboard



Maturaarbeit Alte  
Kantonsschule Aarau

Leumann Nils, G20L  
Thut Benjamin, G20L

Eingereicht bei  
Vázquez Martina,  
Informatik

Oktober 2023

# Inhaltsverzeichnis

<b>Abstract .....</b>	<b>.....</b>
<b>Vorwort .....</b>	<b>.....</b>
<i>Motivation .....</i>	<i>.....</i>
<i>Dank.....</i>	<i>.....</i>
<b>1    Einleitung.....</b>	<b>1</b>
1.1 <i>Ausgangslage und Relevanz.....</i>	<i>1</i>
1.2 <i>Ziele .....</i>	<i>2</i>
1.3 <i>Vorgang.....</i>	<i>2</i>
<b>2    Strommessung .....</b>	<b>3</b>
2.1 <i>Grundlage elektrischer Strom.....</i>	<i>3</i>
2.2 <i>Messung Stromverbrauch .....</i>	<i>3</i>
2.3 <i>Theorie Wechselstrom .....</i>	<i>3</i>
2.3.1    3-Phasen Strom .....	4
2.4 <i>Leistungsfaktor .....</i>	<i>4</i>
2.5 <i>Messung Stromstärke.....</i>	<i>5</i>
2.5.1    CT-Sensor.....	5
2.6 <i>Messung Spannung .....</i>	<i>6</i>
2.7 <i>Messung Leistungsfaktor.....</i>	<i>7</i>
2.8 <i>Genauigkeit der Messungen.....</i>	<i>8</i>
2.9 <i>Messung Photovoltaikanlage .....</i>	<i>8</i>
2.10 <i>Übertragung der Daten .....</i>	<i>8</i>
<b>3    Software .....</b>	<b>9</b>
3.1 <i>Datenbanken .....</i>	<i>9</i>
3.2 <i>HTTP-Datenübertragung .....</i>	<i>10</i>
<b>4    Planung .....</b>	<b>11</b>

4.1	<i>Hardware</i> .....	11
4.1.1	Sensoren.....	11
4.1.2	Messung Photovoltaikanlage .....	11
4.1.3	Mikrocontroller .....	11
4.1.4	Server .....	12
4.2	<i>Datenübertragung</i> .....	12
4.3	<i>Software</i> .....	12
4.3.1	Betriebssystem.....	12
4.3.2	Datenbank.....	12
4.3.3	Programmiersprache.....	12
4.3.4	Datenvisualisierung.....	13
<b>5</b>	<b>Laborversuche</b> .....	<b>14</b>
5.1	<i>Datenübertragung</i> .....	14
5.2	<i>Sensoren</i> .....	15
5.3	<i>Anschluss Sensoren</i> .....	16
5.4	<i>Mikrocontroller</i> .....	19
<b>6</b>	<b>Testaufbau</b> .....	<b>20</b>
6.1	<i>Sensoren installieren</i> .....	20
6.2	<i>Sensoren kalibrieren</i> .....	21
6.3	<i>Software Mikrocontroller</i> .....	22
6.4	<i>Datenübertragung</i> .....	24
<b>7</b>	<b>Implementation</b> .....	<b>25</b>
7.1	<i>Messung Stromverbrauch</i> .....	26
7.2	<i>Messung Spannung</i> .....	27
7.3	<i>Messung Leistungsfaktor</i> .....	27
7.4	<i>Berechnungen</i> .....	28
7.5	<i>Datensammlung</i> .....	29
7.5.1	Photovoltaik-Messungen .....	30
7.6	<i>Fehlervorbeugung</i> .....	31

7.7	<i>Datenvisualisierung</i> .....	31
7.7.1	Flux-Query .....	31
7.7.2	Transformationen .....	32
7.8	<i>Dashboard-Übersicht</i> .....	33
<b>8</b>	<b>Langzeitmessungen</b> .....	<b>35</b>
<b>9</b>	<b>Diskussion</b> .....	<b>36</b>
9.1	<i>Problemlösung</i> .....	36
9.1.1	Lieferzeiten .....	36
9.1.2	Leistung der Mikrocontroller .....	36
9.2	<i>Resultate</i> .....	37
9.3	<i>Datenschutz</i> .....	38
9.4	<i>Zukunft des Projekts</i> .....	38
<b>10</b>	<b>Fazit</b> .....	<b>39</b>
	<b>Abkürzungsverzeichnis</b> .....	<b>40</b>
	<b>Literaturverzeichnis</b> .....	<b>41</b>
	<b>Abbildungsverzeichnis</b> .....	<b>43</b>
<b>11</b>	<b>Anhang</b> .....	<b>i</b>
11.1	<i>Verwendete Open-Source Software</i> .....	<i>i</i>
11.2	<i>Eigenständigkeitserklärung</i> .....	<i>i</i>

## Abstract

Diese Arbeit beschäftigt sich mit der Messung und der Analyse des Stromverbrauchs eines Einfamilienhauses. Es geht um die Konstruktion einer Web-App, welche einem erlaubt, den Stromverbrauch auf einzelne Stromkreise genau über längere Zeiträume (Wochen oder gar Monate) zu analysieren und auszuwerten. Die gemessenen Daten werden mit einem System von sogenannten Current-Transformer (CT)-Sensoren und Mikrocontrollern erhoben. Die Hardware und Software wurden selbst entwickelt und zusammengebaut. Die CT-Sensoren messen den Stromfluss durch einen bestimmten Leiter und geben diese Daten an den Mikrocontroller weiter. Der Mikrocontroller verrechnet diese Daten in einen nützlichen Wert in der SI-Einheit Ampere und gibt diesen dann an einen Linux-Server weiter, welcher die Daten über eine Website darstellt und auf einer Datenbank sammelt. Diese Messungen erlauben dann, herauszufinden, welche Geräte im Haushalt wie viel Strom benötigen und wie man den Einsatz dieser Geräte so anpassen kann, um Energie zu sparen. Ziel der Arbeit ist es, dem Nutzer ermöglichen den Stromverbrauch jeglicher Haushaltsgeräte zu überwachen und zu erkennen, was wie viel Strom braucht. Damit wird eine bessere Planung des Energieverbrauchs ermöglicht, was ein wichtiger Schritt auf dem Weg ist, eine Energieknappheit zu verhindern.

## Vorwort

### Motivation

Die Frage nach einer nachhaltigen und umweltbewussten Lebensweise hat in den letzten Jahren enorm an Bedeutung gewonnen. Eine der zentralen Herausforderungen, der wir uns weltweit gegenüberstehen, ist der drastische Anstieg des Energieverbrauchs und die damit einhergehenden negativen Auswirkungen auf unsere Umwelt. Der Stromverbrauch spielt dabei eine entscheidende Rolle, da er einen Grossteil unseres Energieverbrauchs ausmacht. Mit unserem Projekt wollen wir weitere Möglichkeiten schaffen, Strom einzusparen. Da wir beide sehr an Informatik und Elektrotechnik interessiert sind, haben wir uns für dieses Projekt entschieden.

### Dank

Wir möchten uns herzlich bedanken bei allen Personen und Organisationen, die uns in jeglicher Form bei diesem Projekt unterstützt haben. Besonders möchten wir uns bei unserer Betreuerin Martina Vázquez bedanken, welche uns stets für Fragen zur Verfügung stand. Weiter möchten wir uns beim OpenEnergyMonitor Projekt bedanken, welches viel Recherche zum Thema Strommessung mit CT-Sensoren betrieben hat und alle Dokumentationen frei auf ihrer Website zur Verfügung stellt.

Auch möchten wir allen Entwicklern von Open Source Software, welche in diesem Projekt verwendet wurde, für ihre grossartige Arbeit und Unterstützung der Open Source Community danken.

# 1 Einleitung

## 1.1 Ausgangslage und Relevanz

Im Jahr 2023 sind die Strompreise in der ganzen Schweiz durchschnittlich um 27% angestiegen (Statista, 2023). Dieser Trend wird durch die Fortsetzung des Ukraine-Kriegs und der generell höheren Brennstoffpreise (VSE, 2022) angekurbelt. Dieser Fakt zusammen mit allgemein steigenden Lebenshaltungskosten führt dazu, dass das Stromsparen immer mehr an Relevanz gewinnt. Sei es den Deckel auf die Pfanne beim Kochen oder die «Eco-Taste» beim Geschirrspüler, alle Massnahmen haben dasselbe Problem: Man kann nicht überprüfen, ob sie wirklich etwas bewirken.

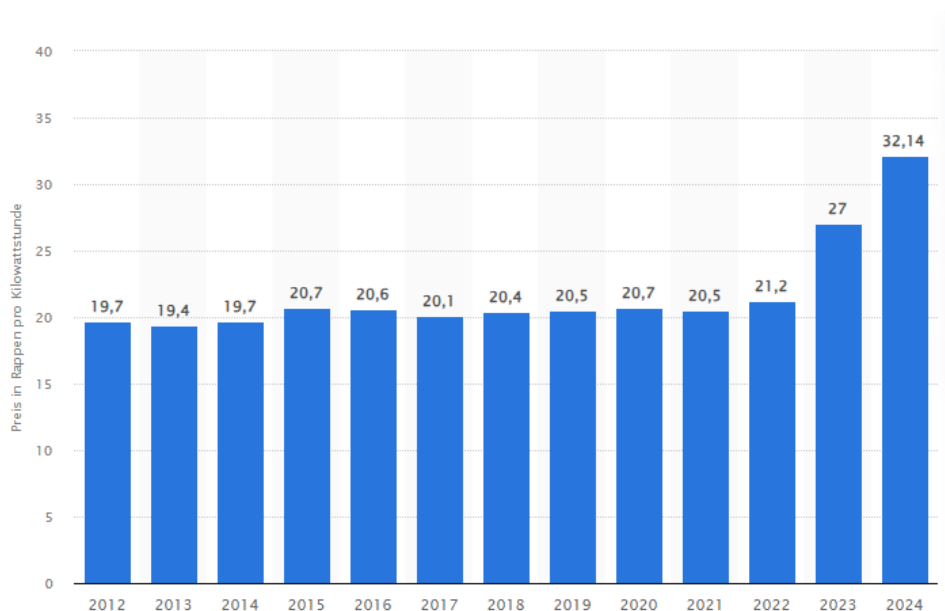


Abbildung 1 Stromkosten pro kWh Schweiz (Statista, 2023)

Dafür gibt es diverse Lösungsansätze wie Stromzähler oder neuer Smart Meter (Entega, kein Datum), die jedoch nur bedingt auf einzelne Geräte zurückschliessen lassen und somit getroffene Massnahmen kaum messen können. Andere Ansätze wie WiFi-Steckdosen sind teuer und nicht bei fest eingebauten Geräten ohne Stecker anwendbar. Deswegen existiert eine echte Nachfrage nach einer kostengünstigen Lösung, die diese Probleme nicht hat.

## 1.2 Ziele

Dieses Projekt zielt darauf ab, eine Lösung für Do-It-Yourself-Stromverbrauchsmessungen im eigenen Haus zu kreieren, welche die Sammlung, Speicherung sowie die visuelle Darstellung der Verbrauchsdaten einschliessen. Erstellt wird das Projekt für das Einfamilienhaus von Nils Leumann in Beinwil am See, einem der Autoren der Arbeit. Es soll jedoch mit geringem Aufwand für andere Häuser anwendbar sein. Das fertige Produkt soll Strom auf allen Stromkreisen einzeln messen sowie zusätzlich die Photovoltaikproduktion miteinbeziehen. Die Installation selbst sollte so wenig Strom wie möglich benötigen und alle verwendeten Fremdlösungen sollen Open-Source sein.

## 1.3 Vorgang

Nachdem wir auf Grund der immer steigenden Strompreise und natürlich unserem Interesse an Technologie und Energie die Idee hatten, dieses Projekt umzusetzen haben wir im ersten Schritt eine Auswahl an benötigten Komponenten bestellt und ein erster Prototyp zusammengesetzt. Nach erfolgreichen Tests mit diesem, haben wir die restlichen Komponenten bestellt und uns für die definitive Software-Umgebung entschieden. Als wir alle Komponenten hatten, haben wir das finale, in der folgenden Arbeit beschriebene Produkt zusammengestellt und die benötigte Software und Scripts dazu programmiert und angepasst.

Während diesem Prozess haben wir uns laufend Notizen zu unserem Vorgehen gemacht.

Erst nachdem das eigentliche Produkt fertig war und korrekt funktionierte, haben wir mit der Schreibarbeit begonnen. Die im Voraus angefertigten Notizen erleichterten uns diese Arbeit deutlich.



## 2 Strommessung

In diesem Kapitel wird untersucht, wie man die Messwerte des Stromverbrauchs eines Einfamilienhauses bestimmt.

### 2.1 Grundlage elektrischer Strom

Unter elektrischem Strom versteht man die Bewegung von elektrisch geladenen Teilchen (Elektronen) in einem Stromkreis. Dabei kann diese elektrische Energie durch sogenannte elektrische Verbraucher in eine andere Form von Energie umgewandelt werden (Rudolph, 2017).

### 2.2 Messung Stromverbrauch

Unter «Stromverbrauch» versteht man umgangssprachlich die Umsetzung von elektrischer Energie in eine andere Form von Energie, z.B. Wärme oder Licht. Strom, also elektrische Energie, wird im eigentlichen Sinn also nicht «verbraucht». Bei der Messung von elektrischem Strom gibt es zwei wichtige Einheiten: Volt (V) und Ampere (A). Unter Ampere versteht man die Menge elektrischer Ladung (in Coulomb (C)), die pro Sekunde durch ein Kabel fließen. Unter Volt versteht man die Differenz an Ladung zwischen zwei Punkten. Bei Gleichstrom (DC) ergeben die zwei Einheiten Volt und Ampere miteinander multipliziert die reale Energie, die durch das Kabel fließt. Diese Energie wird in Watt (W) gemessen. Um den Energiebedarf eines Verbrauchers herauszufinden, muss somit zuerst die Stromstärke und Spannung, welche zum Verbraucher fließt, gemessen werden sowie die Zeitspanne. (Roos, 2022) Hinzu kommt bei Wechselstrom, wie ihn das Stromnetz liefert, noch der sogenannte Leistungsfaktor. (Evans, 2018)

### 2.3 Theorie Wechselstrom

Im Schweizer Stromnetz wird, wie in vielen anderen europäischen Ländern, Wechselstrom mit einer Frequenz von 50 Hz verwendet. In anderen Ländern wie z.B. den USA wird ebenfalls Wechselstrom verwendet aber mit einer Frequenz von 60 Hz. Gleichstromnetze gibt es mittlerweile nicht mehr, da Wechselstrom diverse Vorteile gegenüber Gleichstrom bietet.

Der Hauptvorteil von Wechselstrom (AC), ist, dass Transformation möglich ist. Mit einem Transformator kann sehr einfach die Spannung erhöht und dadurch Strom über lange Strecken mit minimalen Verlusten übertragen werden. Das ist entscheidend für moderne Stromnetze, welche ganze Kontinente abdecken.

Die übliche Wellenform eines Wechselstromkreises (AC) ist eine Sinuswelle. Dies macht einige Berechnungen zum Thema Stromverbrauch etwas komplexer, weil durch die Verschiebung der Sinuswelle Übertragungsverluste entstehen können, welche bei der Berechnung des Energiebedarfs berücksichtigt werden müssen. Dies wird mit dem sogenannten Leistungsfaktor gemacht (Scientific Comittees Europa, kein Datum).

### 2.3.1 3-Phasen Strom

Eine Phase (auch Aussenleiter) bezeichnet einen stromführenden Leiter. In der Schweiz hat fast jeder Haushalt nicht nur eine Phase, sondern gleich drei. Die Sinus-Wellen der drei Phasen ist um 120 Grad verschoben, somit ergibt sich eine Spannung von 230V zwischen einer beliebigen Phase und dem Neutralleiter und eine Spannung von 400V zwischen zwei beliebigen Phasen. Für unser Projekt muss jede Phase individuell gemessen werden. Die drei Phasen können als drei getrennte Stromanschlüsse behandelt werden. Die Phasenverschiebung muss dabei nicht beachtet werden. Die drei Phasen werden mit L1, L2 und L3 bezeichnet, siehe Abbildung unten (emf-info, kein Datum).

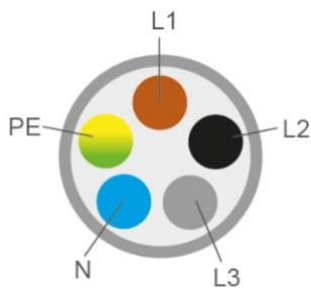


Abbildung 3: 3-phasiges Stromkabel  
(sanier.de, 2022)

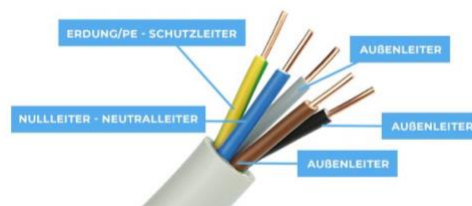


Abbildung 2: Beschriftungen 3-phasiges Stromkabel  
(blogg.de, 2020)

## 2.4 Leistungsfaktor

Der Leistungsfaktor ist ein Begriff aus der Elektrotechnik und beschreibt das Verhältnis der Wirkleistung zur Scheinleistung in einem Wechselstromkreis. Er wird in der Regel als Dezimalzahl zwischen 0 und 1 ausgedrückt und gibt an, wie effizient elektrische Energie in einem System genutzt wird. Ein Leistungsfaktor von 1 bedeutet, dass die Wirkleistung der Scheinleistung entspricht, also ideal ist. Ein Leistungsfaktor von 0,5 bedeutet, dass die Hälfte des Energieverbrauchs Scheinleistung ist. Der Leistungsfaktor ist wichtig, um die Effizienz von Stromversorgungseinrichtungen zu bewerten. Ein niedriger Leistungsfaktor kann zu Übertragungsverlusten führen, während ein hoher Leistungsfaktor die Effizienz erhöht und die Energiekosten senkt. Der Leistungsfaktor kann durch die Verwendung von Kondensatoren oder anderen Geräten zur Blindleistungskompensation verbessert werden. Berechnet

wird dieser, indem man Wirkleistung, also Leistung, die in andere Leistungen umgewandelt werden kann, durch Scheinleistung, also durch die Leistung, die eigentlich zu erwarten ist, teilt. In der Praxis ist das die folgende Formel (ElectronicsTutorials, kein Datum):

$$PF = \frac{P}{I * U}$$

## 2.5 Messung Stromstärke

Bei Geräten, die direkt am Stromnetz hängen, ist der wohl wichtigste messbare Wert die Stromstärke. Es gibt mehrere Wege diesen Wert zu messen. Der konventionelle Weg ist, den Strom durch einen bekannten Widerstand zu leiten. Anhand des Spannungsverlusts über den Widerstand lässt sich die Stromstärke berechnen. Dafür muss jedoch der Stromfluss unterbrochen werden, was die Installation deutlich erschwert und bei Netzspannung auch sehr gefährlich sein kann. Auch geht durch den Widerstand ein wenig Energie in Wärme verloren.

Die bessere Variante, um die Stromstärke zu messen, ist mit einem sogenannten CT-Sensor. Wenn Strom durch einen Draht fließt, entsteht ein Magnetfeld. Je mehr Strom fließt, desto stärker ist das Magnetfeld. Ein CT-Sensor nutzt genau dieses Prinzip und misst das Magnetfeld. Dadurch kann die Stromstärke, die durch das Kabel fließt, berechnet werden.

Der grosse Vorteil von CT-Sensoren ist die leichte Montage. Der Sensor kann einfach um das isolierte Kabel geklemmt werden und es muss nichts getrennt oder an der elektrischen Installation angepasst werden. Wichtig ist, dass man den CT-Sensor nur entweder um die Phase oder um den Neutralleiter installiert, weil die Magnetfelder der zwei Leiter in entgegengesetzter Richtung liegen und sich, wenn man beide Drähte durch den CT führen würde, ausgleichen würden (Gupta, 2019).

### 2.5.1 CT-Sensor

Ein CT-Sensor funktioniert ähnlich wie ein Transformator und basiert auf dem Prinzip der elektromagnetischen Induktion. Es hat eine primäre und eine sekundäre Seite. Die primäre Seite wird durch den stromführenden Draht, um welcher der CT-Sensor montiert wird, dargestellt. Die sekundäre Seite ist der Messausgang des CT-Sensors. Die Anzahl Umdrehungen der beiden Seiten ergibt den Faktor, mit welchem der CT-Sensor die Stromstärke zur Messung verkleinert. Die primäre Seite hat immer nur eine Umdrehung, da der stromführende Draht einmal durch den CT-Sensor geht, die sekundäre Seite hat eine beliebig grössere Anzahl Umdrehungen. Diese Anzahl ergibt den Faktor, mit welchem die Stromstärke multipliziert wird. Je mehr Umdrehungen auf der Sekundärseite, desto kleiner die darauf.

Auf folgender Darstellung steht «HV Line» für die Primärseite, also der Draht, durch den der Strom zum Verbraucher fließt und «Current Sense» steht für die Sekundärseite, also den Messausgang vom CT-Sensor, welcher über eine weitere Schaltung mit dem Mikrocontroller verbunden wird (Gupta, 2019).

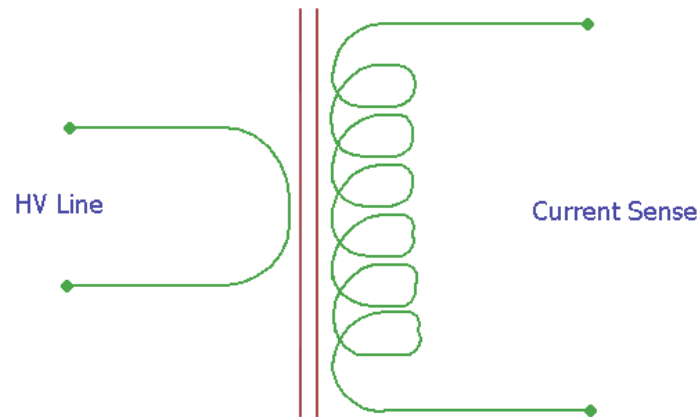


Abbildung 4: Schema CT-Sensor (Gupta, 2019)

## 2.6 Messung Spannung

Die Messung der Spannung ist wichtig, jedoch nicht so wichtig wie die Messung der Stromstärke. Ein Gerät, welches am Stromnetz angeschlossen ist, bekommt immer in etwa dieselbe Spannung. In der Schweiz und diversen anderen europäischen Ländern beträgt diese immer um die 230V, +/- ein paar Prozent. Somit muss die Spannung nicht zwingend gemessen werden, sondern es ist auch möglich, eine Annahme zu tätigen und mit diesem Wert die Berechnungen durchzuführen. Dies verschlechtert jedoch die Genauigkeit der Werte teilweise deutlich, da die Spannung besonders in der Nacht, wenn wenig Last auf dem Netz ist, bedeutsam ansteigen kann.

Die Spannung lässt sich beispielsweise mit einem Mikrocontroller messen. Grundsätzlich kann jeder Mikrocontroller mit einem analogen Eingang direkt Spannung messen. Jedoch ist der Messbereich sehr stark begrenzt, je nach Mikrocontroller-Typ bis maximal 3.3V oder 5V (Jiblom, kein Datum). Die Netzspannung von um die 230V würde den Mikrocontroller sofort durch die Überspannung beschädigen. Somit muss die Spannung zuerst in einen durch den Mikrocontroller messbaren Bereich gebracht werden. Dies ist mit einem Transformator möglich. Ein solcher kann die Spannung mit immer dem gleichen Faktor reduzieren. Wenn dieser Faktor bekannt ist oder durch Versuche errechnet wird, ist es somit möglich, so die Spannung zu berechnen. Der Transformator wird mit der Primärseite an das Netz gehängt und die Sekundärseite wird an den Mikrocontroller angeschlossen. Dieser führt kontinuierlich Messungen auf dem analogen Eingang durch und kann somit die Spannung feststellen.

Solch eine Form der Spannungsmessung ist jedoch recht fehleranfällig und auch heikel. Oftmals ist der Faktor, mit welchem der Transformator die Spannung reduziert, nicht immer gleich, sondern kann je nach Eingangsspannung etwas variieren. Dies bringt erhebliche Ungenauigkeiten in die Messung.

Ein weiterer Weg, die Spannung festzustellen, ist mit käuflichen Spannungssensoren. Diese können an das Stromnetz angeschlossen werden und geben die Spannung dann als digitales Signal aus, welches direkt von einem Mikrocontroller interpretiert werden kann. Diese Sensoren basieren auf einem ähnlichen Prinzip wie ein Transformator zusammen mit einem analogen Eingang eines Mikrocontrollers. Sie sind jedoch wesentlich genauer, weil sie exklusiv für diesen Zweck gebaut wurden. Auch die Umwandlung eines analogen Signals, also einer variablen Spannung zu einem digitalen Signal, geschieht in diesem Sensor, welche diese Umwandlung wesentlich genauer machen kann als der Mikrocontroller.

Beide Wege die Spannung zu messen haben den Nachteil, dass das Gerät, welches die Messung macht, direkt mit dem Stromnetz verbunden sein muss. Dies kann gefährlich sein. Stromschläge bei Netzspannung können zu schweren Verletzungen oder sogar zum Tod führen.

Aus diesem Grund ist es wünschenswert, ein fertiges Produkt zu haben, welches mit einer Steckdose verbunden werden und so die Spannung der ganzen elektrischen Anlage messen kann.

Mit sogenannten «Smart-Plugs», welche den Stromverbrauch von einem einzelnen Gerät messen, ist dies möglich. Diese Smart-Plugs messen die Stromstärke, die durch sie hindurchfließt und auch die Spannung. Da Spannung in parallel mit den Verbrauchern gemessen werden kann, ist das somit auch die Spannung der ganzen elektrischen Anlage bzw. der Phase, an der der Smart-Plug angeschlossen ist. Da wir in der Schweiz in fast allen Haushalten drei Phasen haben, muss die Spannung individuell pro Phase gemessen werden. Dies kann realisiert werden, in dem man drei Smart-Plugs an verschiedene Steckdosen, welche je auf einer anderen Phase sind, anschliesst.

## 2.7 Messung Leistungsfaktor

Der wohl schwierigste Wert zum Messen ist der Leistungsfaktor. Im Gegensatz zur Spannung kann diese Messung nicht in parallel zum Verbraucher gemacht werden. Sie muss wie die Stromstärke in Serie zum Verbraucher gemessen werden. Dabei ist es zwingend notwendig, die elektrische Leitung zu unterbrechen. Da dies bei einer Zuleitung zu einem ganzen Gebäude, über welche grosse Ströme fließen, gefährlich werden kann, ist es nicht empfehlenswert und auch nicht erlaubt, dass solche Arbeiten von nicht qualifizierten Personen ausgeführt werden. Der einzig realisierbare Weg, den Leistungsfaktor zu messen, ist durch die Verwendung eines bereits installierten Zählers.

## 2.8 Genauigkeit der Messungen

Es gibt immer gewisse Ungenauigkeiten in der Messung. Da die Messungen nicht in einem Labor mit wissenschaftlichen Instrumenten, sondern in einem Einfamilienhaus mit einfachen Komponenten getätigt werden, ist das auch zu erwarten. Es ist jedoch wichtig, sich der Ungenauigkeiten bewusst zu sein und diese mit kalibrierten Messgeräten wie Multimetern festzuhalten.

## 2.9 Messung Photovoltaikanlage

Im zu messenden Haushalt ist eine Photovoltaikanlage vorhanden, welche auch gemessen werden sollte. Das spezielle an einer Photovoltaikanlage ist, dass sie zwar, wie ein normaler Verbraucher am Sicherungskasten angeschlossen ist, jedoch keinen Strom verbraucht, sondern Strom produziert und diesen in den Sicherungskasten einspeist.

Eine Möglichkeit wäre auch mit CT-Sensoren den Strom, welcher durch die Photovoltaikanlage eingespeist wird, zu messen. Da CT-Sensoren die Richtung des Stromflusses nicht feststellen können, muss später in der Software berücksichtigt werden, dass die Photovoltaikanlage nicht Strom verbraucht, sondern Strom produziert, der Stromverbrauch somit negativ ist.

Viele moderne Photovoltaikanlagen haben auch eine direkte Anbindung ans Internet für die Überwachung des produzierten Stromes. Diese Daten können abgegriffen werden und somit kann der Stromertrag der Photovoltaikanlage ohne Sensoren direkt in die Datenbank eingetragen werden.

## 2.10 Übertragung der Daten

Die gemessenen Daten müssen von den Sensoren in die Datenbank übertragen werden. Bei Sensoren welche eine TCP/IP-Schnittstelle haben und somit direkt mit dem Netzwerk verbunden sind, kann das in den meisten Fällen über Application Programming Interface (API)-Calls an den Sensor realisiert werden. So auch bei der Photovoltaikanlage über ihre Netzwerkverbindung.

Bei Sensoren mit analogem oder digitalem Ausgang ist das etwas komplexer. Die Messdaten müssen zuerst «übersetzt» werden, bevor sie in die Datenbank eingetragen werden kann. Dies geschieht mit einem Mikrocontroller, welcher analoge und digitale Eingänge sowie eine TCP/IP-Schnittstelle hat.

Der Mikrocontroller sammelt die Daten der Sensoren und gibt diese in Echtzeit über die TCP/IP-Schnittstelle über ein WLAN-Netzwerk weiter. Diese Daten werden dann durch ein Skript in die Datenbank eingetragen.

## 3 Software

In diesem Kapitel werden die verwendeten Software-Technologien beschrieben und erklärt.

### 3.1 Datenbanken

Eine Datenbank ist ein Tool, das verwendet wird, um grosse Mengen an Daten zu speichern und abzufragen. Datenbanken bestehen aus zwei Bauteilen: Das Datenbankmanagementsystem (DBMS) und die Datenbasis (DB). Das DBMS dient hierbei als Schnittstelle zwischen Anwender und Datenbasis, in der alle gesammelten Daten gespeichert werden (Mullins, 2021). Es gibt verschiedene Arten von Datenbanken. Relationale Datenbanken, die Exceldateien gleichen, bestehen aus einer Sammlung von Tabellen (Spalten und Zeilen). Diese Art von DB wird als der Standard betrachtet und findet viel Verwendung vor allem im geschäftlichen Bereich (Kundendaten etc.) (Mahler, 2022). Eine andere Art von Datenbank, die in dieser Arbeit von Relevanz sind, sind Zeitreihendatenbanken. Diese arbeiten mit einem nichtrelationalen Ansatz, also nicht tabellenmässig, sondern mit Zeitstempel, welche gewisse Werte (beispielsweise Messdaten) sowie zugehörige Metadaten (Datenherkunft, Datentyp) angehängt haben. Dies dient zur einfacheren und ressourceneffizienteren Datenabfrage. (Influxdata, kein Datum).

Key			Value
Metric name	Dimensions (labels)	Timestamp	Sample value
http_requests_total	(status="200", method="GET")	@1112596200	987654
http_requests_total	(status="200", method="GET")	@1112596400	986575
http_requests_total	(status="404", method="GET")	@1112596600	946574
http_requests_total	(status="200", method="GET")	@1112596800	976575

Abbildung 5 Beispiel Time-Series Database (Redis, kein Datum)

## 3.2 HTTP-Datenübertragung

Um Daten zwischen zwei Geräten übertragen zu können, gibt es die Möglichkeit, dass auf dem einen Gerät ein Webserver läuft, der kontinuierlich Daten im JSON-Format hochlädt, beispielsweise Sensordaten. Das zweite Gerät kann dann eine Hypertext-Transfer-Protocol (HTTP)-Anfrage stellen, um diese hochgeladenen Daten nach einer beliebigen Zeitspanne (bspw. alle zwei Sekunden) abzufragen und dann weiterzuverarbeiten. (OPC Router, kein Datum). Der Vorteil des JSON-Formates ist, dass jede moderne Programmiersprache damit einfach durch die standardisierte Darstellung umgehen kann und die Daten auch einfach mit dem menschlichen Auge gelesen werden können.

```
{
  "hausliste": {
    "haus": [
      {
        "typ": "fachwerk",
        "text": "Ein altes Haus im Sauerland"
      },
      {
        "typ": "fachwerk",
        "text": "Ein altes Haus im Ruhrgebiet"
      },
      {
        "typ": "modern",
        "text": "Der Mohneturm in Neheim"
      }
    ]
  }
}
```

Abbildung 6 Beispiel JSON-Darstellung (Nexoma, kein Datum)



## 4 Planung

In folgendem Kapitel geht es um die Planungsarbeit, welche wir vor Beginn der Arbeit gemacht haben. Primär geht es um die Hard- und Softwarewahl.

### 4.1 Hardware

#### 4.1.1 Sensoren

Für die Messung des Stromverbrauches haben wir uns für CT-Sensoren vom Hersteller YHDC entschieden. Diese Sensoren bieten eine hohe Genauigkeit, haben ein gutes Preis-Leistungs-Verhältnis und sind beliebt für diese Art von Anwendungen. Dementsprechend gibt es viele Dokumentationen dazu. Für die Messung von Spannung und Leistungsfaktor, den anderen benötigten Werten, haben wir uns für einen Zähler vom Hersteller Fronius entschieden. Dieser Zähler war dank der Photovoltaikanlage bereits installiert und bietet eine API-Schnittstelle, über welche die Daten über das lokale Netzwerk einfach abgefragt werden können.

#### 4.1.2 Messung Photovoltaikanlage

Der Wechselrichter der Photovoltaikanlage, welcher auch von Fronius hergestellt wird, hat wie der Zähler auch eine Netzwerkverbindung und kann über eine API-Schnittstelle die benötigten Werte (Stromproduktion der Photovoltaikanlage) ausgeben.

#### 4.1.3 Mikrocontroller

Die CT-Sensoren geben ein analoges Signal aus. Aus diesem Grund braucht es ein Gerät, welches dieses analoge Signal aufnehmen und in ein digitales Format übertragen kann.

Wir haben uns für den ESP32 Mikrocontroller von Espressif Systems entschieden. Dieser besitzt mehrere analoge Eingänge und hat diverse digitale Schnittstellen wie USB, serielle Schnittstelle und am wichtigsten WLAN. Auch kann dieser Mikrocontroller einfach durch die Arduino IDE programmiert werden, eine Programmierumgebung mit welcher wir bereits Erfahrung hatten und zu welcher sehr viele Dokumentationen verfügbar sind. Dazu ist er sehr kostengünstig.

Da die Zahl der analogen Eingänge begrenzt ist, mussten wir mehrere dieser Mikrocontroller einsetzen, um alle zu messenden Stromkreise mit einem Sensor auszustatten und alle diese Daten dann auch aufzunehmen und in ein digitales Format umwandeln.

#### 4.1.4 Server

Als Server verwenden wir einen gebrauchten Lenovo ThinkCentre M900 Tiny mit einem i5-6500T, 8 GB RAM, und einer 128 GB SSD. Dieser verbraucht ca. 40W unter Volllast und ca. 10W unter Normallast und ist somit eine Energiegünstige Variante.

## 4.2 Datenübertragung

Da Daten von verschiedenen Quellen (mehrere Mikrocontrollern, Fronius Zähler, Fronius Wechselrichter) übertragen werden, war eine netzwerkbasierte Übertragung am einfachsten zu realisieren. Der Fronius Zähler und der Fronius Wechselrichter haben beide eine direkte LAN-Verbindung. Die Mikrocontroller sind per WLAN mit demselben Netzwerk wie die weiteren Geräte verbunden. Auch mit diesem Netzwerk verbunden ist der Server, auf welchem die Datenbank läuft. So können alle Werte in die Datenbank aufgenommen werden.

## 4.3 Software

### 4.3.1 Betriebssystem

Unser Betriebssystem ist Ubuntu 22.04 Server als Linux-Container (LXC) innerhalb der Virtualisierungssoftware Proxmox. Dafür entschieden haben wir uns aufgrund unseres Vorwissens mit Ubuntu und der einfachen Backuplösung von Proxmox. Wenn etwas schief geht beim Konfigurieren von Ubuntu, kann man entweder ein gemachtes Backup des LXC-Containers wiederherstellen oder innerhalb einer Minute einen frischen Container mit Ubuntu aufsetzen.

### 4.3.2 Datenbank

Für die Datenbank, die wir in diesem Projekt verwenden, gab es diverse Kandidaten. Wichtig war jedoch, dass es eine Zeitreihendatenbank ist, da wir ausschliesslich mit Messdaten arbeiten. Nach Recherche über ähnliche Projekte sind vor allem zwei Datenbanken hervorgehoben: Prometheus und InfluxDB 2.7.1. Da diese beiden Anbieter sehr ähnlich sind, haben wir uns schlussendlich für InfluxDB entschieden, da es etwas anfängerfreundlicher sowie sehr einfach mit Python integrierbar ist.

### 4.3.3 Programmiersprache

Die Wahl der Programmiersprache auf Python 3.10.12 war klar, vor allem da schon Vorkenntnisse vorhanden waren. Auch das unaufwändige Installieren von Python-Modulen via dem Python Module-Installer (PIP) auf Ubuntu war ein grosses Plus.

#### 4.3.4 Datenvisualisierung

Zuerst stand im Raum, eine eigene Website mit HTML/CSS zu erstellen, jedoch gab es kaum Vorteile für eine solche Lösung, da sie viel Zeit in Anspruch nimmt und am Ende nicht besser ist als Grafana v10.0.3. Dies haben wir am Ende auch zur Visualisierung unserer Daten verwendet.

## 5 Laborversuche

In diesem Kapitel geht es um erste Versuche, welche wir im Labor gemacht haben.

### 5.1 Datenübertragung

Um mit Python Sensordaten in eine InfluxDB Datenbank zu schreiben, stellten wir im Voraus diverse Versuche auf. Diese dienten dazu, den Datenfluss Stück für Stück aufzubauen. Der erste Schritt war es, ein Python-Skript zu erstellen, welches eine Verbindung zur Datenbank aufbaut, sich per API-Token authentifiziert und dann testweise Zufallszahlen mit Zeitstempel in die Datenbank schreibt.

```
# Create an InfluxDB point
point = Point('random_measurement') \ #erstellt Datenpunkt
    .tag('source', 'python_script') \ #erstellt tag (metadaten) das es aus dem python script kommt
    .field('value', random_number) \ #erstellt Datenwert
    .time(datetime.utcnow())         #fügt Zeitstempel hinzu
```

Abbildung 7 Datenpunkt für InfluxDB in Python

Nach etwas Recherche und ausprobieren konnten wir ein Skript erstellen, das diese Anforderungen erfüllt (Dieses ist auch komplett auf Github zu finden, mehr Info dazu im Anhang). Dieses Skript diente uns später als die Basis für das finale Skript. Mit dem fertigen Skript konnten wir uns nun daran machen, die Daten zu visualisieren. Dazu erstellten wir eine Query in der InfluxDB-nativen Flux-Sprache.

```
from(bucket: "test")
  |> range(start: -24h)
  |> filter(fn: (r) =>
    r["_measurement"] == "random_measurement" and
    (r["_field"] == "value"))
```

Abbildung 8 Test-Flux Query

Diese Query wählt alle Datenpunkte aus, die aus dem Bucket «test» kommen, in den letzten 24h gesammelt wurden und in der Messung «random\_measurement» ein Feld namens «value» haben, Sprich genau unsere Zufallszahlen. Wenn man diese Query in den Datenexplorer von InfluxDB eingibt, dann kommt der folgende Graph, mit Zeit als X-Achse und Zahlenwert als Y-Achse:

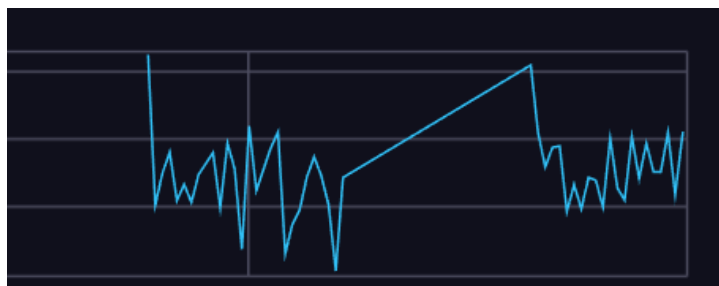


Abbildung 9 Test-Daten im InfluxDB-Dashboard

Darauf konnten wir den ersten Test mit unserer richtigen Datenvisualisierungssoftware Grafana durchführen, bei dem wir diese Zufallszahlen mit Hilfe desselben Flux-Query-Skripts visuell darstellen wollten. Zuerst mussten wir die Datenbank zu Grafana hinzufügen, was trivial mithilfe des eingebauten InfluxDB-Tools. Man muss Datenbankname, IP-Adresse sowie API-Key eingeben, und schon kann man auf die Daten zugreifen. Dazu erstellten wir ein neues Dashboard und einen Graphen bei der wir die Query eingeben konnten. Der daraus resultierende Graph wird unten gezeigt. Die Anzeige wurde auf die letzten fünf Minuten minimiert.



Abbildung 10 Test-Dashboard

Somit war die Testphase für die Datenübertragung von Python zu InfluxDB und zu Grafana abgeschlossen.

## 5.2 Sensoren

In einem ersten Laborversuch ging es darum, die CT-Sensoren zu testen. Dabei kam eine Last mit einem bekannten Stromverbrauch, ein CT-Sensor sowie ein Multimeter zum Einsatz. Die Last wurde an ein spezielles Verlängerungskabel angeschlossen, welches die einzelnen Leiter trennte. Ein CT-Sensor wurde um den Aussenleiter vom umgebauten Verlängerungskabel montiert und ein Multimeter wurde an den Ausgang vom CT-Sensor angeschlossen. Das Multimeter wurde auf die mA-Position eingestellt, um AC-Milliampere zu messen.

Sobald der Verbraucher, ein Wasserkocher mit einer Leistung von 2000W, angeschaltet wurde, zeigte das Multimeter einen Wert von 4.35 mA mit einem CT-Sensoren mit 2000 Umdrehungen (somit einem Faktor von 1:2000) an.

Dieser Wert von 4.35 mA ergibt sich folgendermassen:

Der Wasserkocher benötigt 2000W bei 230V (Netzspannung). Da es eine resistive Last ist, kann der Leistungsfaktor ignoriert werden. 2000W bei 230V entspricht einem Strom von 8.7A, 8.7A mit dem Faktor von 1/2000 multipliziert gibt einen Strom von 0.00435A, also 4.35mA

Die Netzspannung von 230V kann mit demselben Multimeter auf der AC-Volt Einstellung bestätigt werden.

Die Stromstärke, welche der Verbraucher benötigt, kann mit einer speziellen Art von Multimeter, einer sogenannten Stromzange, überprüft werden.

Mit diesen Tests konnte nachgewiesen werden, dass das wohl wichtigste Element, die CT-Sensoren, korrekt funktionieren.

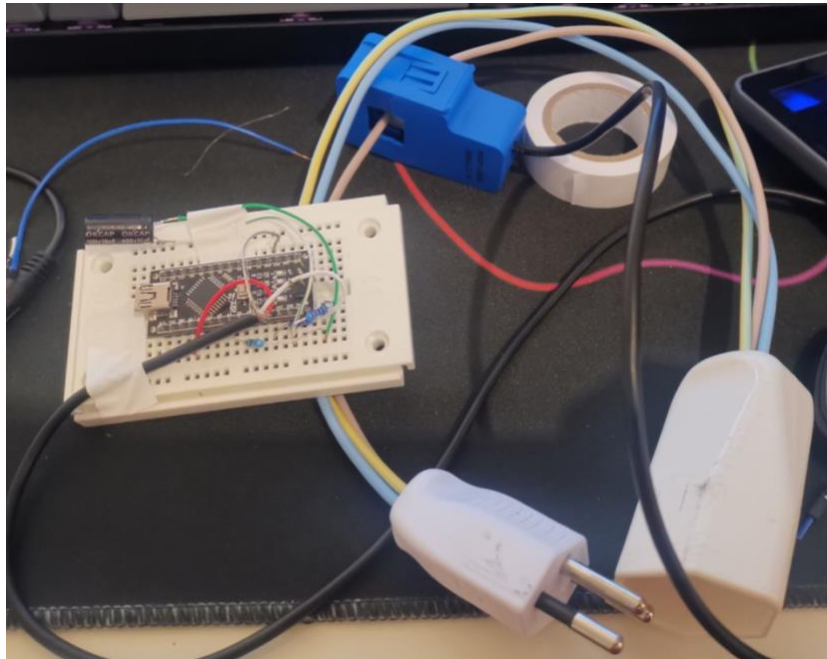


Abbildung 11 Erster Test-Aufbau mit Verlängerungskabel

### 5.3 Anschluss Sensoren

Die durch die CT-Sensoren gemessenen Daten müssen ins Digitale umgewandelt werden. Dies machen wir mit einem Mikrocontroller mit analogen Eingängen. Im ersten Versuch haben wir einen ESP2866 und im finalen Produkt dann mehrere ESP32 verwendet.

Der Grund für diesen Wechsel war die beschränkte Anzahl analoge Eingänge des ESP2866, er hat nämlich nur einen. Der ESP32 hat 12, jedoch sind, wenn WLAN verwendet wird, nur 6 nutzbar. (RANDOM NERD TUTORIALS, kein Datum) Für die Laborversuche reichte der eine Eingang des ESP2866 problemlos.

Die analogen Eingänge vom Mikrocontroller können nicht direkt den Strom, welcher aus dem CT kommt, messen. Sie können nur eine Spannung messen und dies im Bereich von 0V bis 3.3V DC.

Somit muss also zuerst der Strom, der aus dem CT kommt, in eine Spannung umgewandelt werden. Dies geschieht mit einem sogenannten «Burden Resistor». Dieser Widerstand (englisch Resistor) muss

korrekt berechnet werden, um bei maximaler Stromstärke nicht die Eingangsspannung des Mikrocontrollers zu überschreiten.

Im ersten Schritt muss die maximale Stromstärke festgelegt werden, welche der CT messen wird. Da nur einzelne Stromkreise gemessen werden, welche mit 13A oder 16A abgesichert sind, reicht eine maximal zu messende Stromstärke von 16A. Wird diese maximale Stromstärke ein wenig überschritten, werden die Messungen sehr ungenau, bei starker Überschreitung kann der Mikrocontroller sogar beschädigt werden. Da dies jedoch die Absicherung der Stromkreise gar nicht erlaubt, ist das zum Glück keine Gefahr. Bei Strömen, welche die 16A stark überschreiten, würde die Sicherung auslösen und die Last trennen.

Da es sich um Wechselstrom handelt, welcher als Quadratisches Mittel (RMS) repräsentiert wird, muss die Stromstärke im ersten Schritt mit der Quadratwurzel von 2 multipliziert werden, um die Spitzenspannung herauszufinden.

$$16A \times \sqrt{2} = 22.63A$$

Im nächsten Schritt muss die Stromstärke, welcher der CT-Sensor ausgibt, berechnet werden. Die verwendeten CT-Sensoren haben 2000 Umdrehungen, also einen Faktor von 1:2000.

$$22.63A \div 2000 = 0.0113A$$

Da die analogen Eingänge des Mikrocontrollers nur positive Spannungen messen können, bei AC-Strom jedoch die Spannungsspitzen zwischen positiv und negativ variieren, muss die Hälfte der maximalen Spannung für die weiteren Berechnungen verwendet werden.

$$3.3V \div 2 = 1.65V$$

Um den Burden Resistor zu berechnen, muss diese maximale Spannung, also 1.65V, mit dem maximalen Strom, der der CT ausgibt, dividiert werden.

$$1.65V \div 0.0113A = 146.02\Omega$$

Der nächstmögliche verfügbare Widerstandswert ist 150Ω. Wenn ein grösserer Widerstand gewählt wird als berechnet wurde, ist die maximal messbare Stromstärke etwas kleiner. Somit sollte man grundsätzlich den nächstmöglichen kleineren Widerstand wählen. Dieser ist jedoch mit 68 Ω deutlich kleiner. Dazwischen sind keine Widerstände einfach so verfügbar.

Aus diesem Grund haben wir uns dazu entschieden bei Stromkreisen, welche mit 13A abgesichert sind, den etwas grösseren 150Ω Widerstand zu verwenden.

Bei den wenigen Stromkreisen, die mit 16A abgesichert sind, haben wir einen 68Ω Widerstand verwendet.

Je weiter der verwendete Widerstand vom berechneten entfernt ist, desto ungenauer werden die Messungen. Die eingebrachte Ungenauigkeit ist jedoch klein und stellt kein Problem dar.

Wie oben erwähnt kann der Mikrocontroller keine negativen Spannungen messen, der CT-Sensor gibt jedoch beim Messen von Wechselstrom auch negative Spannung aus.

Um dieses Problem zu lösen, muss der Spannungsbereich verschoben werden. Aktuell wird eine Spannung von -1.65 Volt bis +1.65 Volt ausgegeben. Wird nun zu dieser Spannung eine Spannungsquelle mit 1.65 Volt in Serie geschaltet, wird die Spannung zusammenaddiert und der Spannungsreich liegt somit zwischen 0 Volt und 3.3 Volt.

Als 1.65 Volt Spannungsquelle wird die 3.3 Volt Stromversorgung, welche der Mikrocontroller ausgeben kann, mit einem Spannungsteiler halbiert. Ein Spannungsteiler hat zwei gleiche Widerstände in Serie, welche zwischen Masse (GND) (also dem Minus-Pol der Schaltung) und 3.3 Volt verbunden werden. Zwischen den Widerständen kann dann die gewünschte halbierte Spannung, also 1.65 Volt, abgegriffen werden. Je kleiner die Widerstände, desto kleiner der mögliche Ausgangsstrom zwischen den Widerständen, aber auch, desto kleiner die in Wärme verschwendete Energie.

Da diese Spannung nur als Referenz für Messungen benötigt wird und keine Verbraucher damit versorgt werden müssen, können grosse Widerstände verwendet werden. Wir haben uns für 10kΩ Widerstände entschieden. Dabei geht kaum Energie in Wärme verloren.

Somit sieht das Schaltbild folgendermassen aus. Mit «Analog Input» ist ein analoger Eingang vom Mikrocontroller gemeint. Wobei für jeder CT-Sensor ein unterschiedlicher analoger Eingang verwendet wird. Der ganz unten in der Schaltung abgebildete Kondensator ist dazu da Störsignale, welche durch andere Kabel im Sicherungskasten auf die Schaltung übertragen werden können, abzuleiten. (OpenEnergyMonitor, 2023)



Die Schaltung sieht folgendermassen aus:

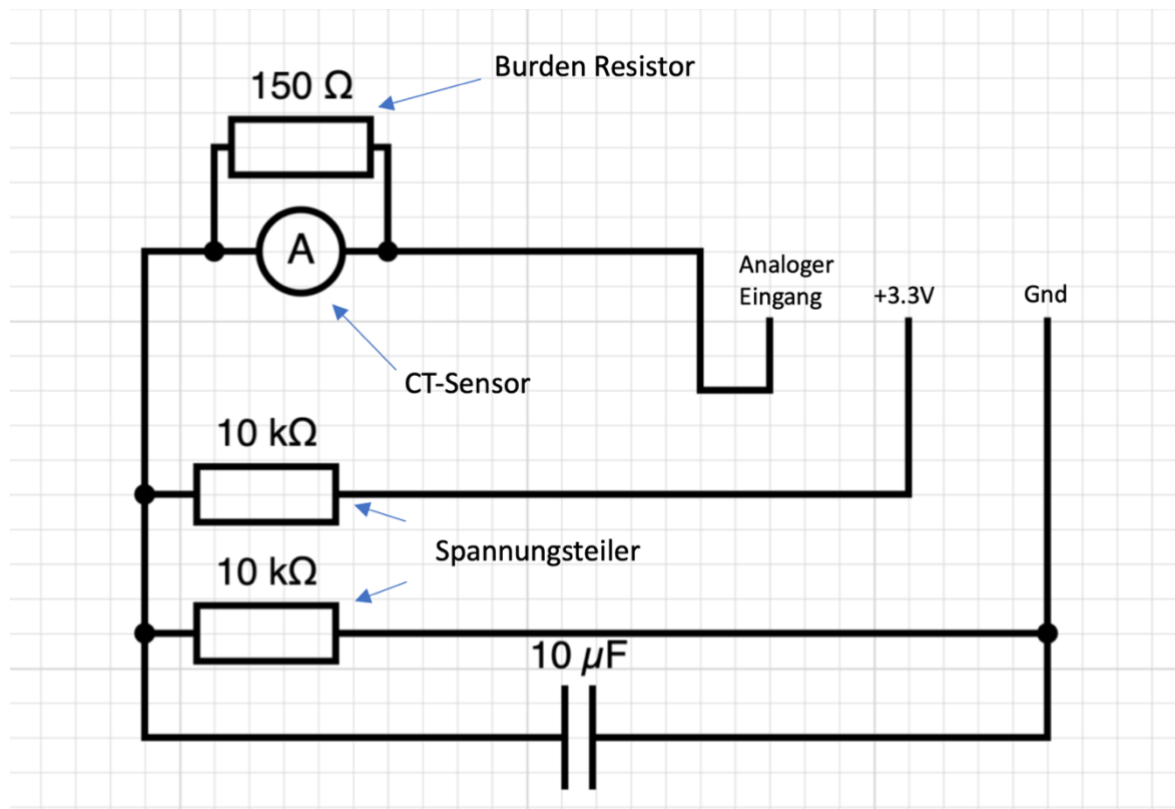


Abbildung 12: Schaltbild Schaltung CT-Sensor

#### 5.4 Mikrocontroller

Nun gibt der Mikrocontroller mit der «analogRead» Funktion einen Wert von 0 bis 1024 aus, entsprechen der eingehenden Spannung, wobei 0 einer Spannung von 0.0V entspricht und 1024 einer Spannung von 3.3V

Eine Spannung von 1.65V, also ein Wert von 511 entspricht dabei 0A und ein Wert von entweder 1024 oder 0 entspricht der durch den Burden Resistor bestimmten Maximalstromstärke.

Dies geschieht, weil es sich um Wechselstrom handelt.

## 6 Testaufbau

In diesem Kapitel geht es um erste Tests, welche bereits im Sicherungskasten des Einfamilienhauses, von welchem wir den Stromverbrauch messen wollen, gemacht wurden.

### 6.1 Sensoren installieren

In einem ersten Test wurden einige Sensoren im Sicherungskasten installiert. Hierfür musste nichts an der Installation verändert werden.

Nach dem wir den Sicherungskasten geöffnet haben, haben wir identifiziert welche Sicherungen für welche Stromkreise verwendet werden. Danach wird der Sensor oben an der Sicherung um das Kabel befestigt. Diesen Prozess gilt es für alle zu messenden Stromkreise zu wiederholen. Am Ende hatten wir 11 Sensoren, welche 11 verschiedene Stromkreise messen. Es ist wichtig zu notieren welcher Sensor mit welchem analogen Eingang vom ESP32 Mikrocontroller verbunden ist, dass später in der Software die einzelnen Graphen richtig identifiziert, werden können.



Abbildung 13 Geöffneter Sicherungskasten mit CT-Sensoren installiert

## 6.2 Sensoren kalibrieren

Einen Sensor haben wir statt um das Kabel von einem einzelnen Stromkreis, um eine der Phasen der Hauptzuleitung montiert. Bei der Hauptzuleitung hat es bereits einen Stromzähler für die Photovoltaikanlage installiert. Dies haben wir gemacht, weil die Sensoren kalibriert werden müssen. Die analogen Eingänge vom Mikrocontroller messen nicht eine Ampere-Zahl, sondern eine Angabe von 0-1024. Wobei ein Wert von 511 einen Stromfluss von 0A repräsentiert. Je weiter der Wert von 511 entfernt ist umso grösser der Stromfluss. Einen Wert von 0 oder 1024 würde der durch den Burden Resistor bestimmten maximalen Stromstärke entsprechen. Die Zuordnung zwischen dem Wert des analogen Eingangs und vom eigentlichen Stromverbrauch ist linear. Somit kann ein fixer Faktor zwischen den beiden Werten errechnet werden, in dem der Wert vom analogen Eingang mit dem Wert vom vorhandenen Stromzähler verglichen werden. Dies haben wir für mehrere Sensoren und Mikrocontroller wiederholt. Da alle Sensoren vom selben Typ sind haben wir jedoch, wie erwartet, festgestellt, dass der errechnete Faktor überall gleich ist, ausser bei den Sensoren bei denen ein anderer Burden Resistor verwendet wird. Pro Burden Resistor muss der Faktor einzeln bestimmt werden.

Falls kein Stromzähler vorhanden ist, kann diese Kalibrierung auch mit einem Multimeter gemacht werden. Statt Sensor und Zähler vergleicht man dann Sensor und Multimeter. Die Kalibrierung ganz ohne externe Messgeräte zu machen ist kaum möglich. Diese ganzen Berechnungen werden durch den Mikrocontroller durchgeführt. Der Mikrocontroller gibt am Ende über die TCP/IP Schnittstelle eine einfache Ampere-Zahl aus, welche dem Strom der, durch den Leiter um welcher der CT installiert ist, entspricht.

### 6.3 Software Mikrocontroller

Die Daten der CT-Sensoren werden durch einen Mikrocontroller weiterverarbeitet und dann an die Datenbank weitergegeben. Der analoge Eingang vom Mikrocontroller bildet die Sinus-Welle vom Wechselstrom ab. Ein Wert von 511 entspricht einem Stromfluss von 0A.

Weil es sich um Wechselstrom handelt, welcher die Wellenform einer Sinuswelle und eine Frequenz von 50 Hz hat, wechselt der gemessene Wert vom Mikrocontroller 100-mal pro Sekunde zwischen beiden Extremen. In folgendem Beispiel mit einem geringen Stromfluss wechselt der Wert zwischen 510 und 512, und das 50 mal pro Sekunde.

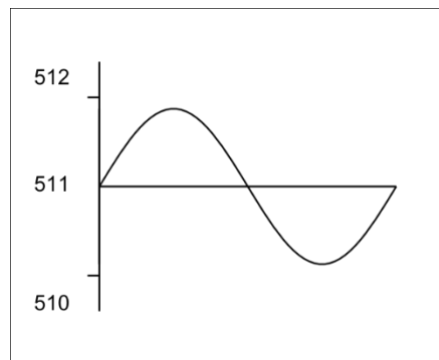


Abbildung 14: Sinus-Welle (OpenEnergyMonitor, 2023)

Um nun aus diesen Werten den Stromfluss zu berechnen, muss die Fläche zwischen Sinus-Welle und der X-Achse, welche bei einem Wert von 511 liegt, berechnet werden. Dies kann mit einer Integralfunktion gemacht werden. Solch eine Berechnung ist für den Mikrocontroller sehr leistungsintensiv. Aus diesem Grund haben wir uns dazu entschieden, statt die Berechnungen von Grund auf zu programmieren, eine fertige Arduino-Library namens EmonLib zu verwenden, welche diese Berechnungen wesentlich effizienter durchführen kann. Der Code dazu sieht folgendermassen aus:

```
void handleRequest() {  
    // Create a JSON object for multiple CTs  
    DynamicJsonDocument jsonDoc(256);  
    for (int i = 0; i < 6; i++) {  
        // Calculate current for each CT using its individual calibration factor  
        float current = emon[i].calcIrms(1480) * emonCalibration[i]; // Number of samples  
        jsonDoc["ct_" + String(i)] = current;  
    }  
}
```

Abbildung 15: Arduino-Code emonLib

Um die ganze Sinus-Welle abzubilden wird 1480 Mal pro Sekunde der Werte des analogen Eingangs gemessen. Danach wird mit der emon[i]-Funktion die Fläche der Sinus-Welle berechnet.

Im letzten Schritt muss die gemessene Fläche der Sinus-Welle mit dem Kalibrierfaktor multipliziert werden. Mehr dazu im Kapitel «6.2 Sensoren Kalibrieren».

Der Code dafür sieht folgendermassen aus:

```
const int ctPins[] = {32, 33, 34, 35, 36, 39}; // Pins for the CT sensors
const float emonCalibration[] = {0.00152, 0.00152, 0.00152, 0.00152, 0.00152, 0.0}; // Calibration factors for each CT, only 5 CTs used
const int ctNumberTurns[] = {2000, 2000, 2000, 2000, 2000, 2000}; // Turns ratios for each CT
```

Abbildung 16: emonLib Kalibrierung

Da mehrere CT-Sensoren an einem Mikrocontroller angeschlossen werden (in diesem Beispiel 5, maximal wären sogar 6 möglich) wiederholt sich dieser Code für jeden analogen Eingang. Da alle Sensoren identisch sind und denselben Burden Resistor haben ist der Kalibrierungsfaktor (emonCalibration) für alle 5 identisch. Die Variable «ctNumberTurns» gibt an was für ein CT-Typ verwendet wird bzw. wie viele Umdrehungen die Sekundärseite hat. Mehr dazu im Kapitel «5.2 Sensoren». Dieser Faktor wird durch die EmonLib Library direkt eingerechnet. Die Variable könnte auch weggelassen werden und in den Kalibrierungsfaktor integriert werden.

Mit der Variable «ctPins» wird angegeben, an welchem analogen Eingang die CT-Sensoren angeschlossen sind. Die Schnittstelle des ESP32 Mikrocontrollers ist durchnummeriert, damit klar ist, welcher Messwert nun zu welchem CT-Sensor gehört. (EmonLib, kein Datum)

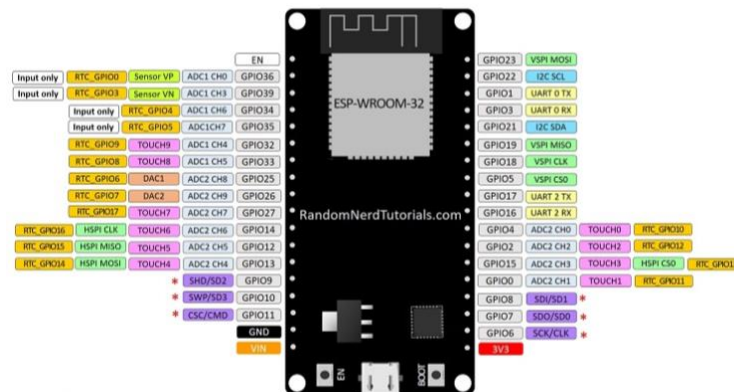


Abbildung 17: Pinout ESP32 (RANDOM NERD TUTORIALS, kein Datum)

## 6.4 Datenübertragung

Die Mikrocontroller sowie der Server, auf welchen die Datenbank läuft, wurden mit demselben Netzwerk wie der Zähler und Wechselrichter für die Photovoltaikanlage verbunden. Der Server konnte somit auf alle gemessenen Daten zugreifen. Für Testzwecke wurden die gemessenen Daten über einen Computer, welcher mit demselben Netzwerk verbunden wurde über den Browser abgerufen, in dem die entsprechende IP-Adresse der Mikrocontroller über den Browser aufgerufen wurde. Der Browser kann genau gleich wie später das Skript auf dem Server einen HTTP-Call an den auf den Mikrocontrollern laufenden Webserver machen. Kommunikation zu den Komponenten der Photovoltaikanlage wurde auch getestet, in dem man die passende Adresse vom Webserver aufgerufen wurde.

Die Daten wurden als JSON direkt im Browser dargestellt und konnten manuell aktualisiert werden, in dem die Seite vom Webserver, welcher sowohl auf den Mikrocontrollern als auch auf den Komponenten der Photovoltaikanlage läuft neu lädt.

Die Daten im JSON-Format sind einfach zu verstehen, jedoch nicht gerade übersichtlich. Auch ist es nur eine momentane Messung, die Daten werden nicht aufbewahrt, vergangene Daten können nicht aufgerufen werden.

```
{"ct_0":3.517603636,"ct_1":0.072611913,"ct_2":0.589103222,"ct_3":0.048810612,"ct_4":0,"ct_5":0}
```

Abbildung 18: Strommessungen ab ESP32 im JSON-Format

## 7 Implementation

In diesem Kapitel geht es um die finale Umsetzung des Projektes. Mit folgendem Blockdiagramm wollen wir diese visualisieren:

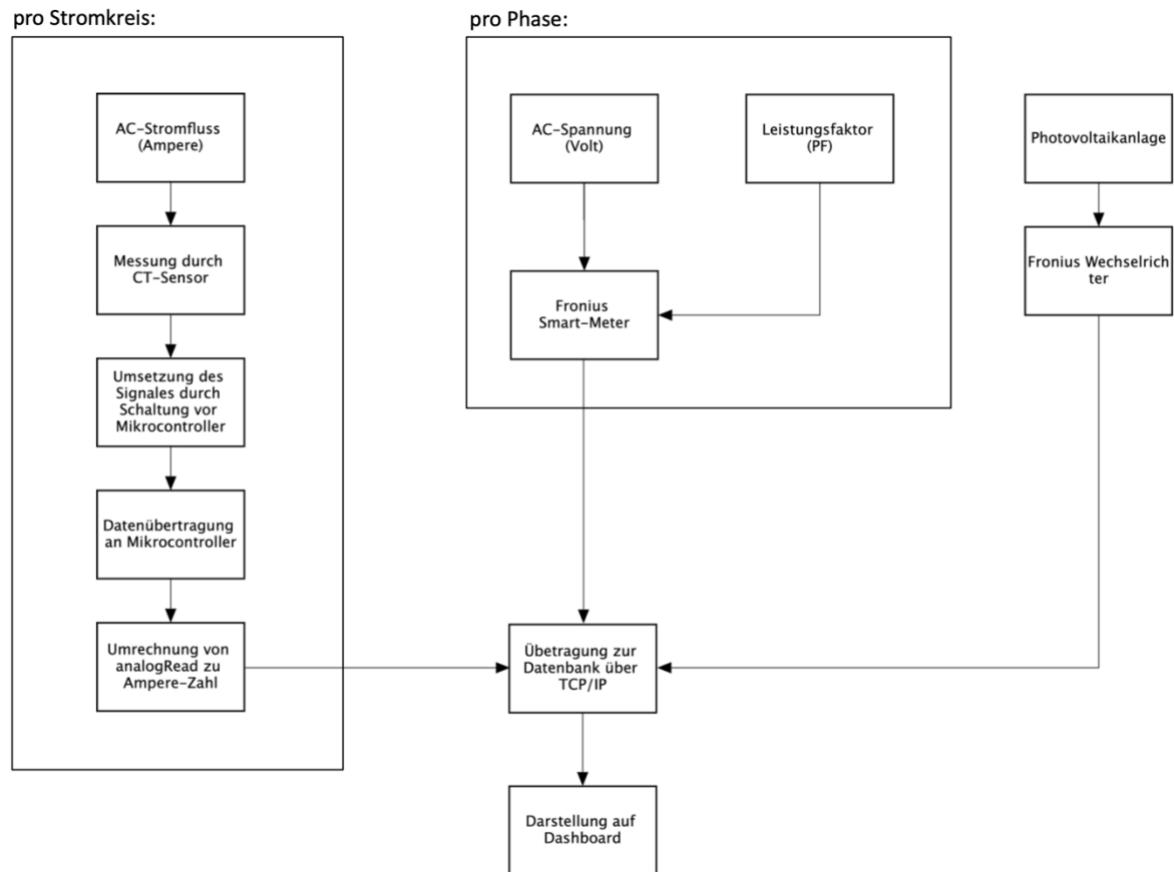


Abbildung 19: Blockdiagramm Gesamtübersicht



## 7.1 Messung Stromverbrauch

Um den Stromverbrauch in Ampere zu messen haben wir uns für sogenannte CT-Sensoren entschieden. Diese Sensoren messen die Stromstärke mit Hilfe des magnetischen Feldes, welches bei Stromfluss durch ein Kabel entsteht. Hierfür wird ein CT-Sensor um den Aussenleiter des jeweiligen Stromkreises im Sicherungskasten montiert.

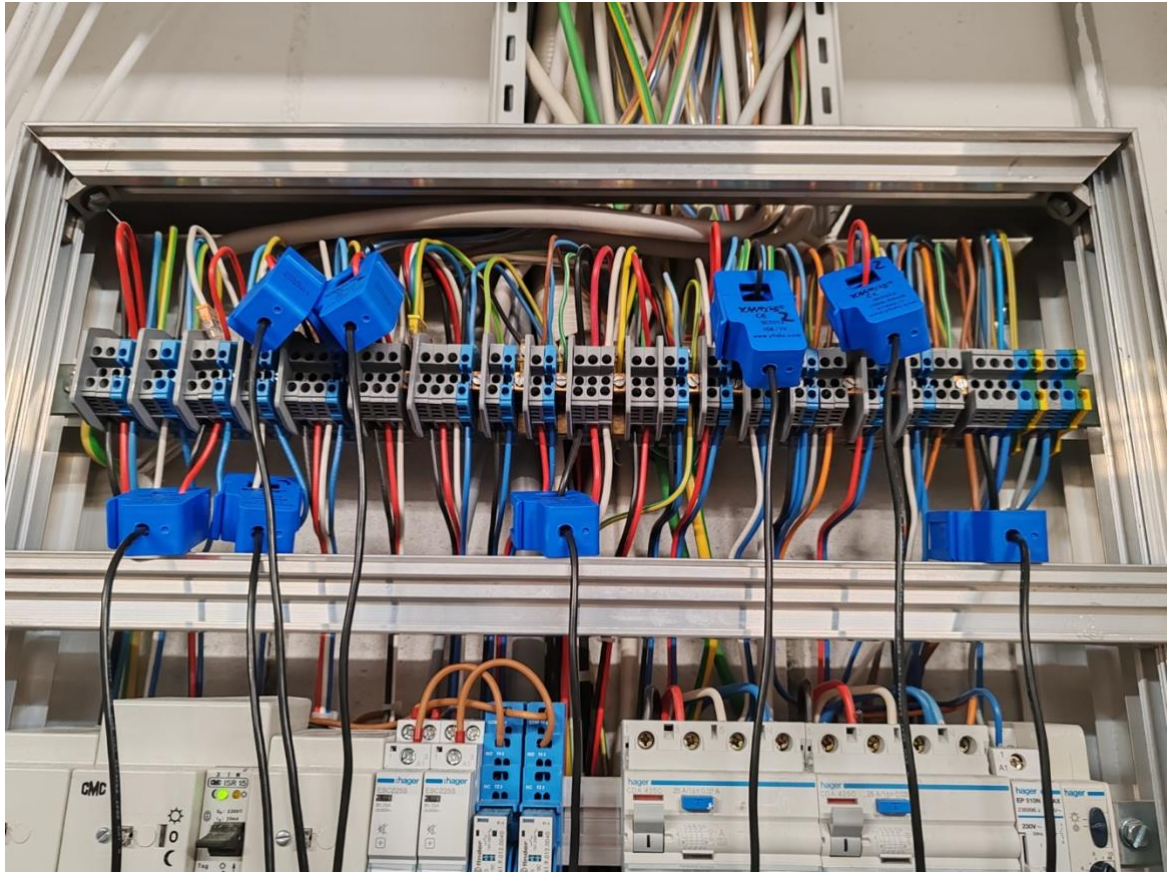


Abbildung 20: CT-Sensoren installiert



## 7.2 Messung Spannung

Da die Spannung nur einmal pro Phase gemessen werden muss und somit in der ganzen Anlage immer gleichbleibt, haben wir uns dazu entschieden, den bereits vorhandenen Zähler für die Photovoltaikanlage für die Spannungsmessung zu verwenden. Dieser Zähler vom Hersteller Fronius hat ein eigenes API und kann damit einfach über das lokale Netzwerk abgefragt werden und die gewünschten Daten so direkt weitergeben.

Durch das in der Schweiz verbreitete 3-Phasen System also 3 Spannungen, für jede Phase eine.

## 7.3 Messung Leistungsfaktor

Der wohl am schwierigsten zu bestimmende Wert ist der Leistungsfaktor. Dieser müsste wie die Stromstärke auch pro Stromkreis individuell bestimmt werden. Da Sensoren, welche den Leistungsfaktor messen, teuer sind und viel Platz brauchen, ist das nicht umsetzbar. Aus diesem Grund haben wir uns dazu entschieden, nur den gesamten Leistungsfaktor pro Phase zu messen.

Der Leistungsfaktor einer Phase entspricht dem Durchschnitt des Leistungsfaktors von allen mit der Phase verbundenen Lasten, nicht der Summe wie beispielsweise bei der Stromstärke.

Aus diesem Grund ist es möglich, anzunehmen, dass jede Last auf der Phase in etwa den durchschnittlichen Leistungsfaktor hat. Der Leistungsfaktor wird wie die Spannung auch über den durch die Photovoltaikanlage vorhanden Stromzähler gemessen. Diese Daten können auch direkt über das API von diesem Zähler abgefragt werden.



Abbildung 21: Fronius Smart Meter

## 7.4 Berechnungen

Für jeder Stromkreis wird einzeln der Strom in Ampere gemessen. Zusätzlich wird die Spannung in Volt und der Leistungsfaktor für die gesamte Anlage bzw. pro Phase gemessen.

Im Dashboard hat es für jeden Stromkreis zwei Graphen, einer der den Strom in Ampere visualisiert und einer der die Leistung in Watt visualisiert. Der Strom-Graph zeigt einfach den jeweiligen gemessenen Stromfluss pro Stromkreis. Der Energie-Graph hingegen ist komplexer. Um die Leistung in Watt  $P$  zu erhalten, muss der Strom  $I$  mit der Spannung  $U$  und dem Leistungsfaktor  $PF$  multipliziert werden.

Für den Strom haben wir pro Stromkreis eine individuelle Messung. Für die Spannung und den Leistungsfaktor haben wir jedoch nur einen Wert für die ganze Phase. Bei der Spannung spielt das keine Rolle, da die Spannung auf der ganzen Phase überall gleich ist. Der Leistungsfaktor hingegen sollte individuell pro Stromkreis bestimmt werden. Da dies aus technischen Gründen nicht realisierbar ist, verwenden wir den Leistungsfaktor-Wert für die ganze Phase, welcher dem Durchschnitt des Leistungsfaktors aller an der Phase hängenden Stromkreise entspricht. Um die korrekte Spannung und der korrekte Leistungsfaktor für die Berechnungen zu nehmen, muss vorher bestimmt werden, an welcher Phase der zu messende Stromkreis angeschlossen ist.

Dadurch ergibt sich die Formel:

$$P = I \times U \times PF$$

## 7.5 Datensammlung

Als Grundlage für dieses Python-Skript wird das erstellte Skript aus den Laborversuchen verwendet. Um die Daten von den zwei Quellen, dem Fronius Smart Meter und dem Mikrocontroller mit den CT-Sensoren, zu sammeln, wird in einem ersten Schritt eine unendliche Schleife geöffnet, in der zuerst eine HTTP-Anfrage an die verschiedenen Webserver mit der get-Funktion des Requests Modul geschickt und als JSON-Datei zwischengespeichert.

```
# Send HTTP requests to the servers
response_1 = requests.get(server_url_1, timeout=5)
data_1 = response_1.json()
response_2 = requests.get(server_url_2, timeout=5)
data_2 = response_2.json()
response_3 = requests.get(server_url_3, timeout=5)
data_3 = response_3.json()
```

Abbildung 22 HTTP-Anfragen (Das eigentliche Skript befindet sich komplett im Anhang)

Danach wird die Datenbank via influxdb\_client Modul aufgerufen und es wird ein Datenpunkt erstellt, dem die aktuelle Zeit sowie die verschiedenen Messwerte angefügt werden.

```
point = Point("power_data") \
    .time(data_1['Head']['Timestamp']) \
    .field("ct_1_value", data_3[ct_1]) \
```

Abbildung 23 Beispiel Datenpunkt

In diesem Beispiel wird ein Datenpunkt namens «power\_data» erstellt, die aktuelle Zeit aus der ersten Datenquelle entnommen und dann ein «field» mit dem Namen «ct\_1\_value» angefügt, was den Wert ct\_1 aus der dritten Datenquelle entnimmt. Natürlich werden im eigentlichen Skript noch viele weitere «fields» angefügt. Danach wird es im Hintergrund an die Datenbank gesendet und das Prozedere fängt von neuem an. Das ganze Python-Skript kommt nur zu Ende, wenn es durch eine Tastatureingabe beendet wird. Für Interessierte: Das komplette Skript ist auf Github hochgeladen. Mehr Informationen dazu im Anhang.

### 7.5.1 Photovoltaik-Messungen

Da bei dem Haus, an dem dieses Projekt umgesetzt wird, eine Photovoltaikanlage verbaut ist, wollten wir auch diese in unsere Messungen miteinbeziehen. In dem Haus sind Produkte von Fronius verbaut, welche von selbst eine Messfunktion eingebaut haben, welche, wie unsere eigenen Sensoren, auf einem Webserver im JSON-Format dargestellt werden (Fronius, kein Datum). Das heisst, dass die Abfrage dieser Daten genau gleich funktioniert wie die Datenabfrage unserer CT-Sensoren. Damit konnten Stromstärke-Messungen der Photovoltaikproduktion pro Phase im selben Skript abgefragt und als «fields» in die DB integriert werden.

```
{
  "Body" : {
    "Data" : {
      "IAC_L1" : {
        "Unit" : "A",
        "Value" : 0.92404073476791382
      },
      "IAC_L2" : {
        "Unit" : "A",
        "Value" : 0.92477774620056152
      },
      "IAC_L3" : {
        "Unit" : "A",
        "Value" : 0.92540097236633301
      },
      "UAC_L1" : {
        "Unit" : "V",
        "Value" : 235.90158081054688
      },
      "UAC_L2" : {
        "Unit" : "V",
        "Value" : 235.94020080566406
      },
      "UAC_L3" : {
        "Unit" : "V",
        "Value" : 235.474609375
      }
    }
  }
}
```

Abbildung 24 JSON Photovoltaik

## 7.6 Fehlervorbeugung

Da der Webserver auf dem Mikrocontroller nicht zu hundert Prozent verlässlich ist, aber die Daten so oft wie möglich mit so wenig Überwachung wie möglich gesammelt werden sollten, gab es die Notwendigkeit für Benachrichtigungen, die ausgesendet werden, falls etwas schief läuft.

```
error_threshold = 4
except requests.RequestException as req_ex:
    error_count += 1
    error_message = f"Error in HTTP request: {req_ex}"
    print(error_message)
    if error_count == error_threshold or error_count%360 == 0:
        send_email(email_subject, error_message)
        time.sleep(10) # Wait before retrying
    else:
        time.sleep(10)
```

Abbildung 25 Request Exceptions

In diesem Beispiel wird, sobald eine `RequestException` (ein Fehler bei einer der HTTP-Anfrage) das vierte Mal auftritt, eine Mail mit dem Error-Status und dessen Beschreibung an unsere Privatmailadressen verschickt. Dies geschieht via dem auf Ubuntu vorinstallierten Postfix, welches die Mails per E-Mail-Relay von einer dedizierten Gmail-Adresse aus verschickt. In das Skript ist das Versenden von Emails mit dem Python Modul «smtplib» eingebunden. Nach jeder `RequestException` wird zehn Sekunden gewartet und dann neu probiert. Nachdem der Fehler viermal nacheinander aufgetreten ist und das erste Mail verschickt ist, werden nur noch stündlich Warnmails versendet.

## 7.7 Datenvisualisierung

### 7.7.1 Flux-Query

Mit den Daten in der Datenbank kann jetzt daran gearbeitet werden, diese zu visualisieren. Dazu erstellten wir wie schon in den Versuchen ein Dashboard in Grafana, um darin die verschiedenen Visualisierungen zu erstellen. Um die richtigen Daten in jedem gewünschten Zeitraum darzustellen, wurde die Flux-Query aus den Versuchen etwas modifiziert.

```
from(bucket: "fronius_data")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) =>
    r["_measurement"] == "power_data" and
    (r["_field"] == "ct_0_value"))
  |> aggregateWindow(every: (if int(v: v.windowPeriod) < int(v: 5s) then 5s else v.windowPeriod), fn: median)
```

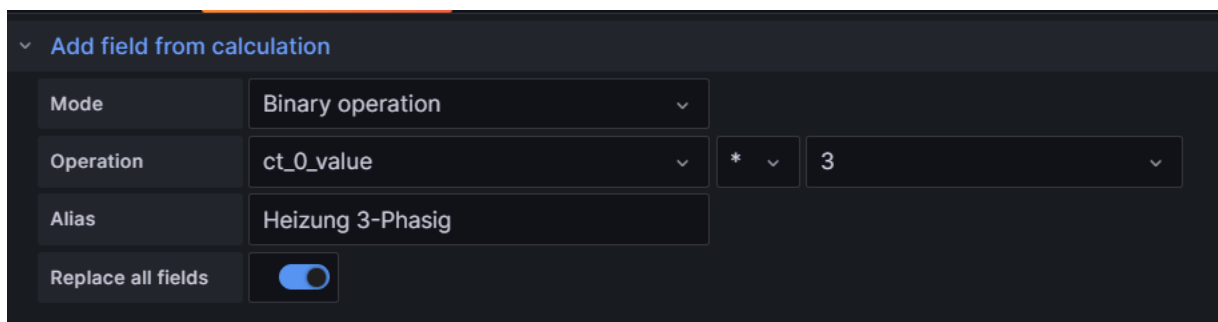
Abbildung 26 Flux Query

Als erstes wird wieder der «Bucket» ausgewählt, was das Äquivalent zum Namen einer Datenbank ist. Danach wird die Zeitränge definiert. «v.timeRangeStart» und «v.timeRangeStop» sind hierbei Variablen

für die Zeitpanne, die man im Grafana-Dashboard anpassen kann und Grafana dann von selbst einfügt. Danach wird eingegrenzt, welche Daten angezeigt werden sollen. In diesem Fall muss es vom «measurement» namens «power\_data» und vom «field» namens «ct\_0\_value» kommen. Heisst, es werden die Daten vom CT-Sensor mit der Nummer 0 angezeigt, welcher in echt an der Leitung zur Heizung angeschlossen ist. Die letzte Zeile Code sorgt dafür, dass nur so viele Daten angezeigt werden, wie auch auf dem Bildschirm Platz haben, ohne zu viel Ressourcen zu verschwenden. Dies im Fall, dass man die anzuzeigende Zeitspanne vergrössert.

### 7.7.2 Transformationen

Wie schon erwähnt, ist es für gewisse Graphen nötig, die gesammelten Daten nachträglich zu modifizieren. Beispiel dafür ist die Heizung. Diese verwendet alle drei Phasen, jedoch wird aus Ressourcengründen nur eine Phase gemessen. Da die Heizung auf allen drei Phasen gleich viel Strom zieht, kann man die Messwerte der gemessenen Phasen mit drei multiplizieren.



The screenshot shows the 'Add field from calculation' configuration panel in Grafana. It contains the following fields and settings:

Field	Value
Mode	Binary operation
Operation	ct_0_value * 3
Alias	Heizung 3-Phasig
Replace all fields	<input checked="" type="checkbox"/>

Abbildung 27 Grafana Datentransformation

## 7.8 Dashboard-Übersicht

Mit dem nun eingerichteten Dashboard kann man nun die Energieversorgung überwachen. Insgesamt wurden acht CT-Sensoren installiert und für jeden davon zwei Graphen erstellt. Einen für Ampere und einen für Watt.



Abbildung 28 Graph Elektroauto Ampere 13-14.10



Abbildung 29 Graph Elektroauto Watt 13-14.10

Im obigen Beispiel kann man den Graphen für den CT sehen, der für das Elektroauto zuständig ist. Die Zeitspanne, die betrachtet werden soll, kann im Dashboard beliebig angepasst werden.

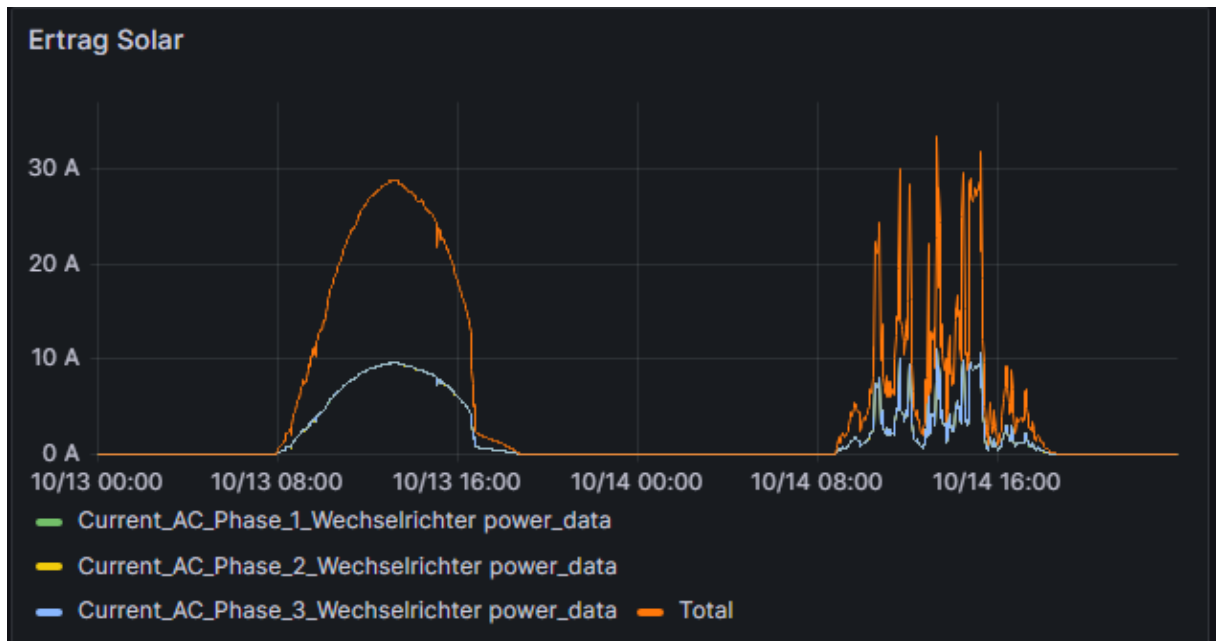


Abbildung 30 Graph Photovoltaikproduktion 13-14.10

Ebenfalls gibt es einen Graphen, welcher die komplette Photovoltaikproduktion pro Phase und Total darstellt.

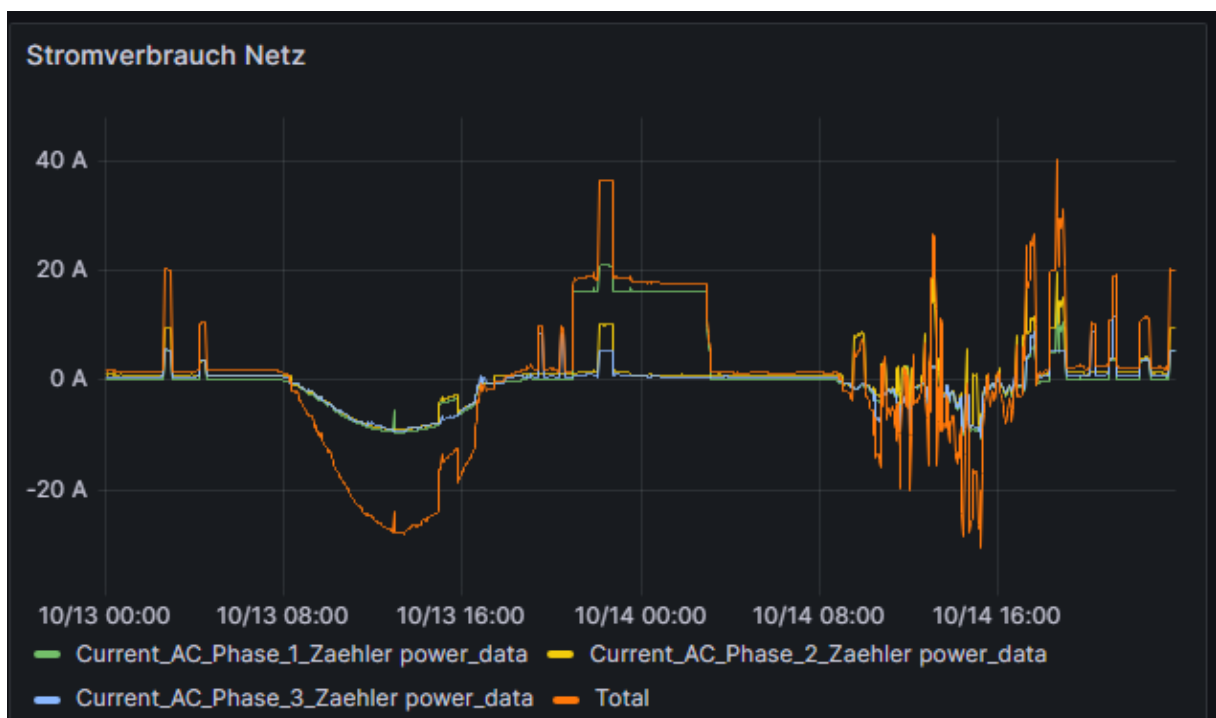


Abbildung 31 Graph Stromverbrauch Netz 13-14.10

Als Letztes gibt es noch einen Graphen, der die Interaktion mit dem Stromnetz darstellt. Immer wenn diese Kurve ins Negative geht, wird Strom von der Photovoltaikanlage in das Netz eingespiesen, wenn sie ins Positive geht, wird Strom vom Netz verbraucht.



## 8 Langzeitmessungen

Die gemessenen Daten werden auf dem Grafana Dashboard als Graphen dargestellt. Es ist möglich eine Zeitspanne der anzuzeigenden Daten festzulegen. Somit können Daten über längere Zeit gemessen und ausgewertet werden.

Mit Hilfe dieser Daten kann der Strombedarf eines Einfamilienhauses visualisiert werden und somit auch besser verstanden werden. Es kann überprüft werden, was für einen Einfluss stromsparende Massnahmen auf den längerfristigen Stromverbrauch haben.

Bei der Bestimmung des Stromverbrauches ist die Zeit ein bedeutsamer Aspekt. Ein Verbraucher, der zwar sehr viel Strom benötigt, aber nur wenige Minuten am Tag läuft (z.B. ein Haarföhn) hat einen wesentlich geringeren Einfluss, als ein Verbraucher, der zwar weniger Strom braucht, aber dafür sehr viel in Betrieb ist (z.B. ein Computer oder ein Fernseher, welcher jeden Tag mehrere Stunden in Betrieb sein kann).

Es kann somit einfacher bestimmt werden, welche Verbraucher einen wie grossen Einfluss auf den gesamten Stromverbrauch haben. Dadurch ist es einfach zu bestimmen welche durch effizientere Modelle ersetzt und/oder auch weniger genutzt werden sollten.

Ein weiterer Aspekt ist der Zeitpunkt an dem Verbraucher verwendet werden. Besonders wenn eine Photovoltaikanlage vorhanden ist, macht es Sinn, möglichst viele Verbraucher mit grossem Energiebedarf am Tag bei Sonnenschein zu verwenden.

Für solche Feststellungen ist eine längerfristige Messung über mehrere Wochen oder Monate unverzichtbar. Mit einem einfachen Stromzähler, welchen man temporär zwischen Verbraucher und Steckdose steckt, kann das nicht erreicht werden.

## 9 Diskussion

### 9.1 Problemlösung

Auch wenn der grösste Teil dieses Projekts schmerzlos ablief, gab es diverse Stellen, an denen wir auf Hindernisse gestossen sind.

#### 9.1.1 Lieferzeiten

Da wir diverse Produkte (Sensoren, Controller) von Aliexpress bestellen mussten, waren die langen Lieferzeiten eine einzigartige Herausforderung. Wenn etwas mit einem Sensor nicht gepasst hat oder einer kaputt angekommen ist, war man aufgeschmissen, bis ein Ersatz ankommt (Dauer 14-40 Tage). Daher haben wir immer diverse Produkte auf Reserve bestellt, damit wir immer genug an Lager haben, auch wenn etwas schief läuft.

#### 9.1.2 Leistung der Mikrocontroller

Die verwendeten Mikrocontroller haben keine grosse CPU-Leistung. Wenn zu oft die Messdaten vom Mikrocontroller abgefragt werden, kann es passieren, dass diese nicht mehr nachkommen und es zu einem Timeout bei der Datenabfrage kommt. Je mehr CT-Sensoren an einem Mikrocontroller angeschlossen sind, desto weniger Anfragen pro Minute können verarbeitet werden. Bei nur einem CT-Sensor ist eine Anfrage alle Sekunden noch möglich. Bei 5 CT-Sensoren, wie wir es in unserem Projekt gemacht haben, ist eine Anfrage alle 5 Sekunden möglich. Bei einer Anfrage pro Sekunde, wie wir es zuerst machen wollten, kommt der Mikrocontroller nicht nach. Aus diesem Grund haben wir uns dazu entschlossen, nur alle 5 Sekunden die Werte von den Mikrocontrollern abzufragen.

## 9.2 Resultate

In der Zeit, in der unser Projekt am Daten sammeln war, konnten wir schon diverse Trends feststellen. Die grössten Verbraucher (vor allem im Oktober, wo es wieder kälter wird) sind die Heizung und der Boiler. Um hier zu sparen, könnte man weniger heisses Wasser verwenden oder vermehrt durch den Tag heizen, da dann die Photovoltaikanlage Strom erzeugt.

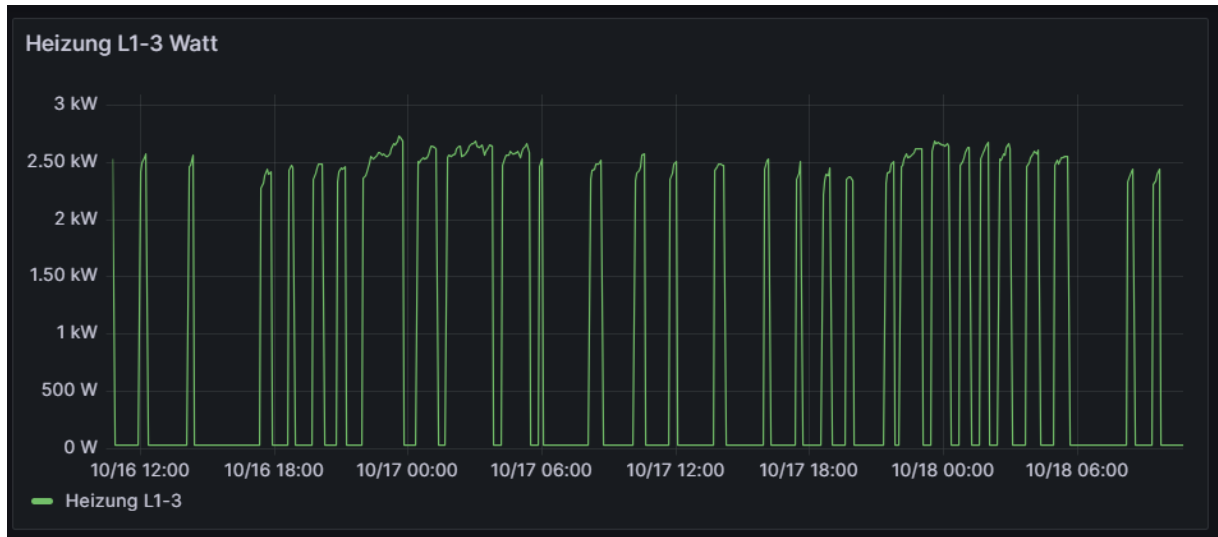


Abbildung 32 Graph Heizung über zwei Tage

Was hingegen weniger Strom verbraucht als gedacht, ist das allgemeine Leben in den Zimmern (Beleuchtung, Ladung Geräte etc.)



Abbildung 33 Graph Zimmer Nils über 24h

Es hat nur einen grossen Peak zwischen 22:00-23:40, da dann Nils viele Geräte am Laden und Verwenden war. Das Licht und der Standby-verbrauch waren immer unter 50W, was, wenn man die angeschlossenen Geräte beachtet, relativ niedrig ist.

### 9.3 Datenschutz

Während dieses Projekt viele interessante Erkenntnisse geliefert hat, sind diese Erkenntnisse potenziell auch interessant für Leute, die das nichts angeht. Mit unseren Graphen allein kann man herausfinden, ob jemand zuhause ist und eventuell sogar, was diese Person am Machen ist. Deswegen sind die Datenbank sowie das Dashboard ausschliesslich im internen Netzwerk von Nils hinter einem Passwort erreichbar. Wenn man von aussen darauf zugreifen will, muss man dies über ein verschlüsseltes Virtuelles Privates Netzwerk (VPN) tun.

### 9.4 Zukunft des Projekts

Das Projekt wird im Haushalt von Nils weiterhin bestehen bleiben und in etwas kleinerer Variante auch bei Benjamin installiert werden. Der Code wird öffentlich auf Github gepostet. Erweiterungen sind nicht geplant, wären aber möglich. Man könnte beispielsweise Smart-Plugs für einzelne Geräte mit Stecker installieren und die Messwerte davon in das Dashboard integrieren.

## 10 Fazit

Insgesamt sind wir sehr zufrieden mit unserer Hardware- sowie Softwarelösung. Dank grandioser Open Source Projekten mit solider Dokumentation wurden die Hardware und die Software relativ schnell zusammengestellt. Unser Produkt ist mit kleinem Aufwand auf andere Haushalte übertragbar und hat Platz für weitere Funktionen und Erweiterungen. Der Stromverbrauch ist minimal und die Kosten sind sehr überschaubar.

Bei den mehreren hundert Linien Code in Python, Flux und C/C++ sowie diverser Testaufbauten und Hardwarewechseln (da sich auf Aliexpress ein Fake-CT-Sensor eingeschlichen hat) haben wir zwangsläufig viel gelernt. Es gibt so viele Faktoren zu berücksichtigen und Fragen zu beantworten. Auch wenn dieses Projekt sehr technisch ist, war trotzdem eine gewisse Kreativität gefragt, da es keine richtigen oder falschen Ansätze gibt. Es kommt darauf an, was man daraus macht.

Rückblickend war unsere Zeiteinteilung realistisch geplant. Wir haben früh mit der Planung und Bestellung von Komponenten angefangen und haben uns die Arbeit gut untereinander aufgeteilt. Einzig der schriftliche Aspekt hat etwas mehr Zeit benötigt als geplant.

Was hingegen eine Herausforderung war, war das Löten der Schaltung. Da es sich um eine recht komplexe Schaltung handelt und wir die Platine vollständig von Hand gelötet haben, war die Fehlerquote relativ hoch, weil Lötstellen verbunden wurden, die nicht miteinander hätten verbunden sein sollen. Die Fehler beim Löten liessen sich jedoch einfach mit einem Multimeter finden und dann beheben.

Schritt für Schritt nahm der ganze Datenfluss Gestalt an und begann, einer legitimen Anwendung zu ähneln. Es war sehr befriedigend zu sehen, wie dieses umfangreiche Puzzle aus Elementen und Funktionen zusammenarbeitet, um einen einfachen und kohärenten Datenfluss zu liefern. Als wir zum ersten Mal getestet haben, wie die Anwendung mithilfe der CT-Sensoren Daten erfasst, fühlte es sich fast magisch an, die präzisen Ergebnisse schön dargestellt zu sehen.

## Abkürzungsverzeichnis

CT	Current Transformer
DB	Datenbank/Datenbasis
DBMS	Datenbankmanagementsystem
API	(zu Deutsch) Programmierschnittstelle
LXC	Linux-Container
HTTP	Hyper-Text-Transfer-Protocol
PIP	Preferred-Install-Program (Python Module-Installer)
GND	Masse (eng. Ground, 0V-Referenz)
AC	Wechselstrom (eng. Alternating Current)
DC	Gleichstrom (eng. Direct Current)
RMS	Quadratisches Mittel (eng. Root Mean Square)
V	Volt
A	Ampere
W	Watt
$\Omega$	Ohm
C	Coulomb
U	Spannung in Volt
I	Strom in Ampere
P	Leistung in Watt
PF	Leistungsfaktor (eng. Power Factor)

## Literaturverzeichnis

- blogg.de. (4. 9 2020). Abgerufen am 14. 10 2023 von BLOGG.de: <https://blogg.de/stromkabel-farben-bedeutung/>
- ElectronicsTutorials. (kein Datum). Abgerufen am 2. 10 2023 von ElectronicsTutorials: <https://www.electronics-tutorials.ws/de/actheorie/leistungsdreieck.html>
- Elektrizitätskommission, E. (06. 09 2022). Abgerufen am 4. 10 2023 von Stark steigende Strompreise 2023: <https://www.admin.ch/gov/de/start/dokumentation/medienmitteilungen.msg-id-90237.html>
- emf-info. (kein Datum). Abgerufen am 3. 10 2023 von emf-info: <https://www.emf.ethz.ch/de/emf-info/themen/technik/stromversorgung/dreiphasen-wechselstrom>
- EmonLib. (kein Datum). Abgerufen am 2. 10 2023 von EmobLib Github: <https://github.com/openenergymonitor/EmonLib>
- Entega. (kein Datum). Abgerufen am 2. 10 2023 von STROMVERBRAUCH MESSEN – MIT DEN RICHTIGEN TOOLS.: <https://www.entega.de/blog/stromverbrauch-messen>
- Evans, P. (12. 11 2018). Abgerufen am 3. 10 2023 von THE ENGINEERING MINDSET: <https://theengineeringmindset.com/power-factor-explained/>
- Fronius. (kein Datum). Abgerufen am 30. 09 2023 von Froinus Dokumentation: <https://www.fronius.com/~/downloads/Solar%20Energy/Operating%20Instructions/42,0410,2012.pdf>
- Gupta, S. (6. 5 2019). Abgerufen am 27. 9 2023 von CIRCUIT DIGEST: <https://circuitdigest.com/article/how-to-measure-current-in-a-circuit-with-different-current-sensing-techniques>
- Influxdata. (kein Datum). Abgerufen am 28. 9 2023 von Time series database (TSDB) explained: <https://www.influxdata.com/time-series-database/>
- Mahler, C. (20. 09 2022). Abgerufen am 28. 9 2023 von Relational Databases vs Time Series Databases: <https://www.influxdata.com/blog/relational-databases-vs-time-series-databases/>
- Mullins, C. S. (08 2021). Abgerufen am 1. 10 2023 von Datenbank-Managementsystem (DBMS): <https://www.computerweekly.com/de/definition/Datenbank-Management-System-DBMS>
- Nexoma. (kein Datum). Abgerufen am 18. 10 2023 von JSON: <https://nexoma.de/json/>
- OPC Router. (kein Datum). Abgerufen am 2. 10 2023 von Was ist JSON? Praxisnahes Basiswissen: <https://www.opc-router.de/was-ist-json/>

OpenEnergyMonitor. (2023). Abgerufen am 21. 9 2023 von OpenEnergyMonitor:  
<https://docs.openenergymonitor.org/electricity-monitoring/ct-sensors/measurement-implications-of-adc-resolution-at-low-current-values.html>

OpenEnergyMonitor. (2023). Abgerufen am 19 09 von OpenEnergyMonitor:  
<https://docs.openenergymonitor.org/electricity-monitoring/ct-sensors/interface-with-arduino.html>

RANDOM NERD TUTORIALS. (kein Datum). Abgerufen am 4. 10 2023 von RANDOM NERD TUTORIALS:  
<https://randomnerdtutorials.com/esp32-pinout-reference-gpios/>

Redis. (kein Datum). Abgerufen am 18. 10 2023 von <https://redis.com/nosql/timeseries-databases/>

Roos, D. (3. 10 2022). Abgerufen am 6. 10 2023 von howstuffworks:  
<https://science.howstuffworks.com/environmental/energy/question501.htm>

Rudolph, D. (28. 12 2017). Abgerufen am 11. 10 2023 von Gut-Erklärt.de: <https://www.gut-erklart.de/physik/elektrizitaet-grundlagen-grundwissen.html>

sanier.de. (5. 10 2022). Abgerufen am 6. 10 2023 von sanier.de:  
<https://www.sanier.de/elektroinstallation/herd-anschliessen>

Scientific Comittees Europa. (kein Datum). Abgerufen am 15. 10 2023 von Scientific Comittees Europa:  
[https://ec.europa.eu/health/scientific\\_committees/opinions\\_layman/de/elektromagnetische-felder/glossar/wxyz/wechselstrom.htm#:~:text=Der%20normale%20Netzwechselstrom%20hat%20in,Frequenz%20von%2050%20Hz%20entspricht.](https://ec.europa.eu/health/scientific_committees/opinions_layman/de/elektromagnetische-felder/glossar/wxyz/wechselstrom.htm#:~:text=Der%20normale%20Netzwechselstrom%20hat%20in,Frequenz%20von%2050%20Hz%20entspricht.)

Statista. (09 2023). Abgerufen am 6. 10 2023 von Durchschnittlicher Strompreis für Haushalte in der Schweiz in den Jahren 2012 bis 2024:  
<https://de.statista.com/statistik/daten/studie/329740/umfrage/haushaltstrompreis-in-der-schweiz/>

VSE. (06. 09 2022). Abgerufen am 11. 10 2023 von Darum steigen die Strompreise 2023:  
<https://www.strom.ch/de/nachrichten/darum-steigen-die-strompreise-2023>



## Abbildungsverzeichnis

Abbildung 1 Stromkosten pro kWh Schweiz (Statista, 2023) .....	1
Abbildung 2: Beschriftungen 3-phasiges Stromkabel (blogg.de, 2020) .....	4
Abbildung 3: 3-phasiges Stromkabel (sanier.de, 2022).....	4
Abbildung 4: Schema CT-Sensor (Gupta, 2019).....	6
Abbildung 5 Beispiel Time-Series Database (Redis, kein Datum) .....	9
Abbildung 6 Beispiel JSON-Darstellung (Nexoma, kein Datum) .....	10
Abbildung 7 Datenpunkt für InfluxDB in Python .....	14
Abbildung 8 Test-Flux Query .....	14
Abbildung 9 Test-Daten im InfluxDB-Dashboard .....	14
Abbildung 10 Test-Dashboard .....	15
Abbildung 11 Erster Test-Aufbau mit Verlängerungskabel.....	16
Abbildung 12: Schaltbild Schaltung CT-Sensor .....	19
Abbildung 13 Geöffneter Sicherungskasten mit CT-Sensoren installiert .....	20
Abbildung 14: Sinus-Welle (OpenEnergyMonitor, 2023) .....	22
Abbildung 15: Arduino-Code emonLib .....	22
Abbildung 16: emonLib Kalibrierung .....	23
Abbildung 17: Pinout ESP32 (RANDOM NERD TUTORIALS, kein Datum).....	23
Abbildung 18: Strommessungen ab ESP32 im JSON-Format .....	24
Abbildung 19: Blockdiagramm Gesamtübersicht .....	25
Abbildung 20: CT-Sensoren installiert.....	26
Abbildung 21: Fronius Smart Meter .....	27
Abbildung 22 HTTP-Anfragen (Das eigentliche Skript befindet sich komplett im Anhang) .....	29
Abbildung 23 Beispiel Datenpunkt .....	29
Abbildung 24 JSON Photovoltaik.....	30
Abbildung 25 Request Exceptions .....	31
Abbildung 26 Flux Query.....	31
Abbildung 27 Grafana Datentransformation.....	32
Abbildung 28 Graph Elektroauto Ampere 13-14.10 .....	33
Abbildung 29 Graph Elektroauto Watt 13-14.10 .....	33
Abbildung 30 Graph Photovoltaikproduktion 13-14.10.....	34
Abbildung 31 Graph Stromverbrauch Netz 13-14.10 .....	34
Abbildung 32 Graph Heizung über zwei Tage.....	37
Abbildung 33 Graph Zimmer Nils über 24h .....	37

## 11 Anhang

Der gesamte Code sowie eine kurze Anleitung, wie man vorgehen würde, wenn man dieses Projekt selbst umsetzen möchte, sind auf Github zu finden: <https://github.com/NilsLeumann/FroniusEsp32CT-Dashboard>

Bei Fragen zum Code können Sie uns gerne per E-Mail kontaktieren: [nils.leumann@gmail.com](mailto:nils.leumann@gmail.com) und [thut.benjamin@gmail.com](mailto:thut.benjamin@gmail.com)

### 11.1 Verwendete Open-Source Software

Ubuntu 22.04 und alle in der Apt-Repository vorhandenen Packages

Proxmox v8.0.3

Grafana 10.1.5

InfluxDB v2.7.3

InfluxDB-client 1.38.0 (Python Modul)

Python 3.10.12 und alle standartmässig installierten Module

Arduino IDE 2.2.1 und alle standartmässig installierten Module

emonLib 1.1.0 (Arduino Modul)

### 11.2 Eigenständigkeitserklärung

Wir erklären hiermit, dass

.. diese Arbeit weder ganz noch teilweise abgeschrieben noch kopiert  
oder aus dem Internet oder von einer künstlichen Intelligenz wie  
ChatGPT u.ä. übernommen wurde.

.. der Quellennachweis gemäss den Vorgaben des Handbuches

Projekte korrekt und vollständig ist,

.. die dargestellten Daten und Resultate von den Unterzeichnenden  
selbst und gemäss den Vorgaben des Handbuches Projekte erhoben  
und verarbeitet wurden.

Aarau. 19. Oktober 2023

Nils Leumann

Benjamin Thut