BSc Computational and Data Science

CDS205 Computer Science

Dozentin: Prof. Dr. Ana Petrus

Sensorbasiertes Leistungsmonitoring

Präzise Energieüberwachung: Integration eines Hall-basierten Stromsensors in ein Raspberry-Pi-System

Fabian Nadler[1,*]

[1]Fachhochschule Graubünden

**E-Mail Adressen: fabian.nadler@stud.fhgr.ch*

10. Januar 2025

Abstract

Die Überwachung des Energieverbrauchs elektrischer Geräte kann teuer und komplex sein. In diesem Projekt wurde ein kosteneffizientes System entwickelt, das einen Hall-basierten Stromsensor (TMCS1100A4) und einen Raspberry Pi kombiniert. Mit einer durchschnittlichen Abtastrate von 650 Werten pro Halbwelle wurde eine präzise Leistungsbestimmung ermöglicht, auch bei verzerrten Kurvenformen. Die Visualisierung der Daten erfolgt benutzerfreundlich und intuitiv über eine lokal gehostete Dash-Anwendung.

Erste Tests zeigten zuverlässige Ergebnisse im Bereich von 0 bis 2,875 kW. Die Beschränkung auf die positive Halbwelle und vorhersehbare Skalierungsprobleme bleiben als Herausforderungen bestehen. Das System bietet eine solide Grundlage für kostengünstige und intuitive Energieüberwachung.

1 Einleitung

In einer Zeit, in der Energiepreise kontinuierlich steigen und das Interesse an nachhaltiger Ressourcennutzung wächst, gewinnt die Überwachung des Energieverbrauchs zunehmend an Bedeutung. Besonders im privaten und industriellen Bereich steigt die Nachfrage nach kostengünstigen und benutzerfreundlichen Lösungen zur Echtzeit-Messung von Leistungsdaten.

Trotz der Verfügbarkeit moderner Technologien sind viele der auf dem Markt erhältlichen Geräte und Systeme oft zu teuer oder technisch zu komplex. Ziel dieses Projekts ist es, mit einfachen Mitteln und geringem Kostenaufwand eine präzise Erfassung der Leistungsdaten sowie deren benutzerfreundliche Visualisierung zu ermöglichen. Die Arbeit zeigt, wie durch die Kombination bewährter Technologien und datenwissenschaftlicher Ansätze eine praktische und erschwingliche Lösung geschaffen werden kann, die sowohl für Privathaushalte als auch für kleinere Unternehmen von Nutzen ist.

2 Methodik

Da es Online nur eine begrenzte Anzahl an vorgefertigen Sensoren gab, wurde eine eigene Leiterplatte entwickelt. Als Stromsensor wurde der aus früheren Projekten bekannte TMCS1100A4 verwendet, der mit einer Sensitivität von 0,4 V/A die Projektanforderungen erfüllt.

Der Stromsensor wurde so konfiguriert, dass nur positive Ströme gemessen werden. Dadurch wird ein Bereich von 0,125A bis 12,5A abgedeckt. Laut Datenblatt ist der Bereich zwischen 0A und 0,125A aufgrund von Rauschen nicht präzise messbar.

Nach der Bestückung der Platine wurde festgestellt, dass der Raspberry Pi keinen integrierten Analog-Digital-Wandler (ADC) besitzt. Daher wurde der MCP3201 hinzugefügt, ein 12-Bit-ADC mit hoher Sampling-Rate, um innerhalb einer Phase (50 Hz / 20 ms) genügend Datenpunkte zu erfassen.

Der ADC hat eine Auflösung von 12 Bit (4096 Werte), wodurch eine Genauigkeit von 0,328 mA pro Schritt erzielt wurde:

$$CURRENT\ RANGE = \frac{5V}{0.4V/A} = 12.5A$$

$$ADC\ RESOLUTION = \frac{4096}{12'500mA} = 0.328mA$$

Die Kommunikation zwischen dem ADC und dem Raspberry Pi erfolgte über die SPI-Schnittstelle. Die SPI-Clock wurde auf 1,5625 MHz festgelegt, basierend auf der Clock-Divider-Berechnung:

$$SPI\ CLOCK\ DIVIDER = \frac{400MHz}{256} = 1.5625MHz$$

Die maximale Sampling-Rate des ADC wurde auf 100 ksps begrenzt, was pro Netzspannungsperiode (50 Hz) maximal 2000 Samples ergibt:

$$MAX\ SAMPLES = 100ksps * \frac{1}{50Hz} = 2000$$

Nach erfolgreicher Kommunikation und Übertragung der ersten Daten vom ADC wurde die Rechenlogik für die Leistungsmessung implementiert. Da die Netzspannung von 230 V / 50 Hz eine symmetrische Sinusform aufweist, reicht es aus, nur eine Halbwelle zu analysieren.

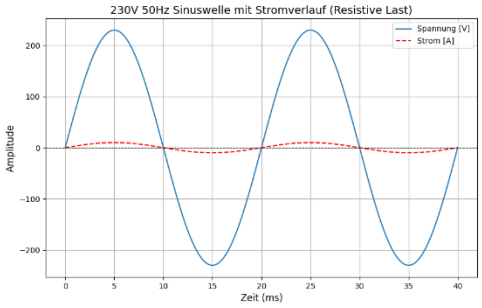

Abbildung 1 Optimaler Strom- / Spannungsverlauf 230V

Die Dauer des Sampling-Prozesses wurde auf 40 ms festgelegt, um zwei komplette Perioden der Netzspannung (50 Hz) aufzuzeichnen:

$$SAMPLING\ TIME = \frac{1}{50Hz} * 2 = 20ms * 2 = 40ms$$

Die positive Halbwelle wurde aus den Messdaten extrahiert. Dabei wurde eine Methode entwickelt, welche Rauschen und Schwankungen an den Übergangspunkten anhand von Grenzwerten filtert.

Der Effektivwert des Stroms I_{RMS} wurde aus den gefilterten Werten berechnet, da der Strom Sinusförmig sein sollte, konnte der Effektivwert anhand des Spitzenwertes berechnet werden.

$$I_{RMS} = \frac{I_{peak}}{\sqrt{2}}$$

Anschließend konnte die effektive Leistung während der Periode berechnet werden:

$$P_{Eff} = I_{RMS} * 230V * \cos(\varphi)$$

Hierbei wurde der Leistungsfaktor cos(φ) mit 0,9 angenommen.

Zur benutzerfreundlichen Darstellung der Daten wurde die Python-Bibliothek **Dash** verwendet. Die Applikation wurde lokal auf dem Raspberry Pi gehostet und bietet interaktive Diagramme, die Leistungsdaten bis auf die minutengenaue Ebene visualisieren.

3 Resultate

3.1 Erster Test

Für den ersten Test wurde eine Mikrowelle als Last verwendet. Dank der technischen Daten des Geräts konnten die gemessenen Werte des Stromsensors abgeglichen und überprüft werden. Dabei zeigte sich jedoch, dass die gemessenen Werte erheblich von den theoretisch erwarteten abwichen.

Grund für die Abweichungen: Anders als angenommen war der Stromfluss nicht rein sinusförmig, wie es idealerweise bei einer rein resistiven Last der Fall wäre. Dies wurde anhand der Messdaten festgestellt und in der folgenden Abbildung visualisiert:

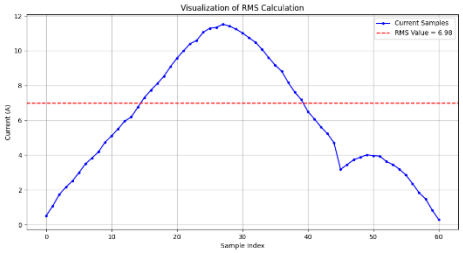

Abbildung 2: Stromverlauf Mikrowelle

Wie in der Abbildung erkennbar, ähnelt der Stromverlauf einer Sinusform, zeigt jedoch deutliche Abweichungen. Besonders auffällig ist der lokale Spitzenwert am Ende der positiven Halbwelle. Die Ursache dafür liegt in den kapazitiven und induktiven Elementen wie Kondensatoren und Spulen, die in der Mikrowelle verbaut sind. Solche Bauteile können den Stromfluss verzerren und zeitweise sogar Strom zurück ins System speisen.

Anpassungen bei der Berechnung: Aufgrund dieser Abweichungen musste die Berechnung des Effektivwerts des Stroms überarbeitet werden. Die bisher verwendete Division durch $\sqrt{2}$ setzt voraus, dass das Signal sinusförmig ist, was hier nicht zutraf.

Ein weiterer kritischer Punkt war die Anzahl der Samples, die während der 40 ms ausgewertet wurden. Diese lag durchschnittlich bei 500 Samples pro 40 ms und damit deutlich unter den Erwartungen. Dies beeinträchtigte die Genauigkeit der Messungen und musste bei der Analyse berücksichtigt werden.

3.2 Anpassungen

Ursprünglich wurde die SPI-Schnittstelle mittels einer Python-Bibliothek umgesetzt. Um die Geschwindigkeit der Kommunikation zu verbessern, wurde mithilfe von Cython eine Methode implementiert, die die SPI-Kommunikation über eine C-Bibliothek steuert. Diese Optimierung führte zu einer erheblichen Zeitersparnis im Sampling-Prozess und ermöglichte eine durchschnittliche Anzahl von 3000 Samples pro 40 ms, was einer deutlichen Steigerung gegenüber den vorherigen 500 Samples entspricht.

Zusätzlich wurde jedem Sample ein Zeitstempel hinzugefügt, um den genauen Messzeitpunkt des jeweiligen Wertes zu erfassen. Mithilfe dieser neuen Daten konnte die Berechnung des Effektivwerts angepasst werden, um eine höhere Genauigkeit zu gewährleisten. Die neue Berechnung berücksichtigt die gesamte Halbwelle, selbst wenn bestimmte Zeitabschnitte keinen Stromfluss aufweisen konnten.

$$T_{total} = \frac{\frac{1}{50Hz}}{2} = 10ms$$

$$T_{unmeasured} = T_{total} - T_{measured}$$

$$I_{RMS} = \sqrt{\frac{\sum_{i=1}^{n} I_i^2 * \Delta t_i + 0^2 * T_{unmeasured}}{T_{total}}}$$

Mit dieser Methode wird der Effektivwert des Stroms präziser berechnet, da die Fläche unter der Kurve durch die zeitliche Zuordnung der Messwerte exakt bestimmt werden kann. Die Division durch T_{total} stellt sicher, dass auch bei asymmetrischen Stromverläufen oder teilweisen Messungen der Durchschnitt korrekt berechnet wird.

Die finale Leistungsberechnung beruht weiterhin auf der folgenden Formel:

$$P_{Eff} = I_{RMS} * 230V * \cos(\varphi)$$

3.3 Zweiter Test

In dieser Phase wurde das System ausschließlich mit einem Computer und der dazugehörigen Peripherie getestet, da dies die geplante Praxisanwendung darstellt. Die Ergebnisse waren vielversprechend und wirkten korrekt, konnten jedoch nur auf Basis von Annahmen überprüft werden. Da geeignete Messgeräte zur Verifizierung der Messwerte fehlen, wird in diesem Bericht nicht auf die absolute Korrektheit der Messungen eingegangen.

4 Diskussion

Das entwickelte System hat gezeigt, dass es möglich ist, kostengünstig präzise Leistungsdaten zu messen und visuell darzustellen. Dennoch gibt es Optimierungspotenzial. Die Beschränkung auf die positive Halbwelle liefert eine eingeschränkte Perspektive auf den gesamten Stromverlauf, was die Genauigkeit der Messungen bei asymmetrischen Lasten beeinträchtigen kann. Künftige Anpassungen könnten Änderungen der Hardware vorsehen, um beide Halbwellen zu erfassen.

Ein weiteres Problem besteht in der Skalierbarkeit der Datenverarbeitung. Mit zunehmender Laufzeit und wachsender Datenmenge könnte aufgrund von Ladezeiten die Responsivität der Dash-Anwendung leiden. Eine mögliche Lösung wäre die Implementierung einer Datenbank zur effizienteren Verwaltung und Abfrage der Messdaten.

Trotz dieser Herausforderungen bietet das Projekt eine solide Grundlage für kosteneffiziente Leistungsüberwachung und hat praktische Anwendungen in Haushalten und kleinen Unternehmen. Durch die unkomplizierte Anwendung sowie intuitive Visualisierung konnte wie erhofft grossen Wert auf die Benutzerfreundlichkeit gelegt werden.

5 Abbildungsverzeichnis

6 Hilfsmittelverzeichnis

Für die Entwicklung der Software sowie das Verfassen der Dokumentation wurde das ChatGPT Modell GPT-4 zur Unterstützung hinzugezogen.

7 Literatur

Eigene Ausbildungsunterlagen. (2018–2022). **Grundlagen der Elektronik**, Ausbildungsunterlagen der Ausbildung zum Elektroniker EFZ.

Texas Instruments. (n.d.). **TMCS1100: Hall Effect Current Sensor IC Datasheet**. Abgerufen am 02. Januar 2025, von https://www.ti.com

Microchip. (n.d.). **MCP3201: 12-Bit ADC Datasheet**. Abgerufen am 02. Januar 2025, von https://www.microchip.com

Raspberry Pi Foundation. (n.d.). **Raspberry Pi Zero Technical Specifications**. Abgerufen am 02. Januar 2025, von https://www.raspberrypi.org

Dash by Plotly. (n.d.). **Dash Documentation**. Abgerufen am 02. Januar 2025, von https://dash.plotly.com

Cython. (n.d.). **Cython Documentation**. Abgerufen am 02. Januar 2025, von https://cython.org

8 Anhang

GitHub Repository mit der Software/Hardware & Dokumenten:
https://github.com/nadlerfabian/power_monitor