Umwelt Messungen mit Arduino Uno R3, Raspberry Pi Pi Pico und MQTT

Messen - Kommunizieren - Verarbeiten - Steuern

Ma

27. Februar 2024

Inhaltsverzeichnis

1 Einleitung **3**

2 Weg der Information Sensor - Programm - Aktor **4**
2.1 Messung 4
2.2 Messverfahren 4

3 Von der Temperatur in den Controller **5**
3.1 Aufgabe 6
3.2 Aufgabe 6

4 Vom Controller zur Aktion **7**

5 Mehr als lokale Daten: mit MQTT ins Netzwerk **8**

6 Große Lasten schalten: Motoren und andere Verbraucher **9**

7 Von Bussen in der Elektronik: I2C und SPI **10**

1 Einleitung

In Rechenzentren, der Schule und anderen Szenarien sollen Umweltdaten erfasst werden. Abhängig von Messwerten sollen Aktionen ausgelöst werden (Info, Alarm, ...).

Im Kapitel 2 befasse ich mich mit allgemeinen Fragen der Messtechnik: Wie wird eine Temperatur gemessen?

Wie die Information über den Temperatur-Wert in den Controller kommt, damit beschäftige ich mich in Kapitel 3.

Kapitel 4 befasst sich mit der Frage: Wie kann ich kleine Verbraucher, z.B. eine LED, ein- und ausschalten. Zum Beispiel in Abhängigkeit eines Messwerts aus Kapitel 2.

Im Kapitel 5 befasse ich mich mit der Erweiterung der Schaltung und lasse die gemessenen Daten vom Raspberry Pi /Raspberry Pi pico zu einem Server (MQTT-Server) übertragen. Bei dem MQTT-Server abonnieren Clients die Messwerte und lassen sich über den aktuellen Wert und Veränderungen informieren.

Das Kapitel 6 befasst sich mit leistungsstärkeren Verbrauchern, die von dem Controller (Arduino Uno R3, Raspberry Pi, ...) gesteuert werden sollen. Wie man die Signale des Controllers passend verstärkt, und was man bei Relais unbedingt beachten sollte, ist hier der Inhalt des Kapitels.

Final im Kapitel 7 befasse ich mich mit Sensoren, die ihre Messwerte nicht als analoge Spannung liefern, sonder in digitaler Form. I^2C und SPI sind hier typische Transportmittel, sogenannte Busse der Controller. RS232 kann man zusätzlich zur Kommunikation mit dem PC verwenden. Spoiler: ein Arduino Uno R3 ist per USB am PC angeschlossen, eigentlich kommunizieren die beiden nicht über USB.

2 Weg der Information Sensor - Programm - Aktor

Temperaturmessung ist in der Regel einfach . . . Ein Blick auf ein Thermometer verschafft die gewünschte Information. Bei Luftfeuchtigkeit ist es der Hydrometer.

Lediglich die Zugänglichkeit ist eingeschränkt. Ich muss physisch an den Ort des Messgeräts kommen.

Die Automation soll hier helfen. Die Messung soll automatisch erfolgen, bei Bedarf sollen automatisch Aktionen ausgelöst werden.

2.1 Messung

Das Thermometer beruht auf dem physikalischen Prinzip: je wärmer desto mehr dehnt sich ein Körper aus.

ein Pt 100 ist ein solcher Sensor. Hier wird per Widerstandsänderung die Temperatur bestimmt.

Bei Veränderung der Temperatur verändert sich auch der Widerstandswert. Der Sensor hat einen positiven Temperaturkoeffizient (je wärmer er ist, desto größer ist der Widerstand).

2.2 Messverfahren

Die Veränderung des Widerstandswerts kann man mit einem passenden Messgerät (Widerstandsmessgerät alternativ Multimeter) beobachten. Zur automatischen Messung ist dies nur bedingt geeignet. Dass Messgerät muss dann über eine Datenschnittstelle mit dem Controller / Computer verbunden werden.

3 Von der Temperatur in den Controller

Für die Messung mit dem Controller / Computer müssen die Informationen passend aufbereitet werden. Ein Raspberry Pi, Arduino (Uno R3) oder andere Controller können eine Spannung messen und als digitaler Wert abspeichern/verarbeiten.

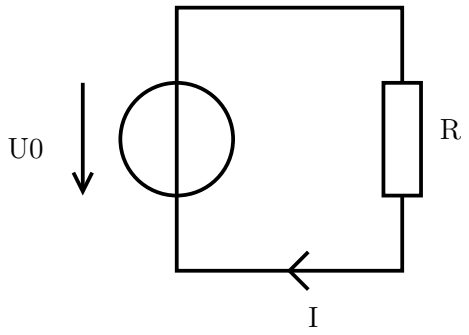

Abbildung 3.1: Schaltung1 Quelle mit einem Widerstand

Die Spannungsquelle (Kreis, links im Bild) steht symbolisch für den Controller (Arduino Uno R3). Am Widerstand R fällt die gesamte Spannung ab[1], die die Quelle zur Verfügung stellt (hier 5 V). Durch den Widerstand fließt ein Strom, er wird wie folgt berechnet $I = \frac{U}{R}$ (I = Stromstärke, U = Spannung R = Widerstandswert).

Wenn ich eine Spannung von $U_0 = 5\,V$ und einen Widerstand von $R = 100\ \Omega$ verwende erhalte ich folgende Werte:
$I = \frac{5V}{100\ \Omega} = 0,05\,A = 50\ mA$ Der Arduino liefert an den Ausgangspins maximal 20 mA, man sollte nicht mehr als 10 mA beziehen. Eine Möglichkeit ist hier den Widerstandswert von 100 Ω auf 500 Ω oder 1000 Ω zu ändern[2].

Leider kann der Arduino nur entweder den Widerstand mit Energie (Spannung und Strom) versorgen oder eine Messung der Spannung am Widerstand durchführen. Die Messung wäre hier im Übrigen 5 V, die Spannung der Versorgung. bei einer Veränderung der Temperatur verändert sich der Wert des Widerstands, jedoch nicht die Spannung, da der Controller (Arduino Uno R3) ggf. mehr Strom liefert.

Eine Strommessung wäre die Lösung, leider unterstützt der Arduino Uno R3 dies nicht direkt.

Daher muss ich eine relative Messung durchführen. Wenn ich mehrere Widerstände in Reihe schalte (der Strom muss dann durch Widerstand R_1, dann durch Widerstand R_2, ...) addieren sich die Spannungen, die an den einzelnen Widerständen abfallen.

$U_0 = U_1 + U_2$ Bei beiden Widerständen gilt weiter $U = I * R$ (die Formel von oben nach I aufgelöst).

Wenn beide Widerstände identische Werte haben, teilt sich U_0 jeweils zu 50 % in U_1 und U_2 auf R_2.

$U_0 = U_1 + U_2$

$U_2 = U_0 - U_1$

[1] genaugenommen wird die Energie in Wärme umgewandelt.

[2] Ω ist das Formelzeichen für Ohm.

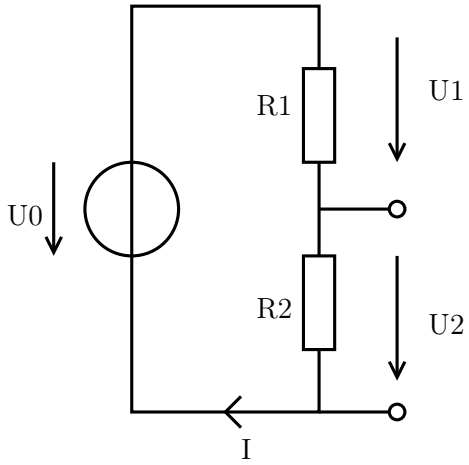

Abbildung 3.2: Schaltung 2 Quelle mit zwei Widerständen in Reihe

Mit $R_1 = R_2$ gilt $U_1 = U_2$ und $U_0 = 2 * U_2$ oder $U_2 = \frac{U_0}{2}$

Der Messwiderstand R_2 hat bei 20° einen Wert von 1000 Ω Ich kann jetzt die Spannung U_2 messen und als Eingang (z.B. A0) am Arduino bestimmen lassen.

3.1 Aufgabe

Recherchiere wie man mit einem Arduino Uno R3 (in C/C++) den analogen Wert am Eingang A0 bestimmt und als Variable r0 abspeichert.

3.2 Aufgabe

Stecke auf einer Steckplatine (breadboard) die Schaltung aus Bild 3.2.

Die Quelle U_0 ist die Spannungsversorgung (PWR, 5V) vom Arduino. der untere Anschluss von R_2 wird an einem Pin GND angeschlossen. Der Pin A0 wird an die Verbindung von R_1 (unterer Pin) und R_2 (oberer Pin) angeschlossen. Beachte die internen Kontakte der Steckplatine. Im mittleren Bereich sind sie quer verbunden, im äußeren längs.

Wenn alles korrekt ist ($R_1 = R_2 = 1000\Omega$) soll ein Wert von ca. 512 gemessen werden. Ein Wert von ca. 0 oder ca. 1024 ist falsch.

4 Vom Controller zur Aktion

5 Mehr als lokale Daten: mit MQTT ins Netzwerk

6 Große Lasten schalten: Motoren und andere Verbraucher

7 Von Bussen in der Elektronik: I2C und SPI