Steuerung einer Lüftungsanlage für Wohnhäuser

Inhaltsverzeichnis

1 Sicherheitshinweis

ACHTUNG 230V, LEBENSGEFAHR

EIGENES RISIKO BEIM NACHBAU

Die Lüfter, der Bypass und das Vorheizregister werden mit 230V Versorgungsspannung betrieben bzw. mit 230V Spannung (Bypass) gesteuert.

Für den **Betrieb der Lüftungsanlage mit einer Feuerstätte** ohne Fremdluftzufuhr, siehe Abschnitt 6.1

2 Einführung

2.1 Ziel

Ziel dieser Entwicklung ist es, defekte Steuerungen von Lüftungsanlagen für Wohnhäuser durch diese Eigenentwicklung zu ersetzen. Die Entwicklung des Projekt erfolgt in Freizeit als Hobby. Für die Entwicklung werden Standardbauteile verwendet. Dies sind: Arduino Mega, ein LAN-Modul, OneWire Temperatursensoren DS18B20, optional ist die Ansteuerung eines Vorheizregisters, die Ansteuerung für den Sommer-Bypass, Sensoren für CO2, VOC und Luftfeuchtigkeit sowie ein (Touch-) Grafikdisplay für die lokale Anzeige. Die Lüfter können per Optokoppler oder separaten DAC Digital Analog Converter (0-10V) angeschlossen werden. Diese Steuerung implementiert eine MQTT Schnittstelle für die Verbindung zur Haussteuerung.

Die Steuerung wurde ursprünglich für eine Pluggit P300 Anlage entwickelt, funktioniert ebenso für Pluggit P450 und viele weitere Lüftungsanlagen. Die elektrische Anpassung für unterschiedliche Lüftungsanlagen wird im Abschnitt Hardware beschrieben.

Die Steuerung implementiert die wesentlichen Funktionen zum technisch sicheren Betrieb der Lüftungsanlage. Die Frostschutzabschaltung des Zuluftventilators ist immer aktiv, eine Vereisung und damit ggfs Zerstörung des Wärmetauschers sollte damit ausgeschlossen sein. Voraussetzung ist der korrekte Anschluss der Temperatursensoren. Die Autoren dieser Software übernehmen KEINERLEI Garantie. Es werden die folgenden Sensoren/Aktoren geschaltet.

- zwei Lüfter werden angesteuert und deren Tachosignale werden ausgelesen
- vier Temperatursensoren werden abgefragt
- Steuerung des Zuluftventilator zum Frostschutz
- Steuerung des Abluft- und des Zuluftventilators zum Frostschutz bei Kaminbetrieb
- Optional: Vorheizregister als Frostschutz
- Optional: Sommer-Bypass geschaltet
- Optional: Bis zu 2 Temperatur/Luftfeuchtesensoren (DHT22)
- Optional: CO2-Sensor MH-Z14
- Optional: VOC-Sensor

Diese Steuerung stellt eine MQTT Schnittstelle zur Verfügung und kann darüber mit Automatisierungssystemen kommunizieren. Alle Topics der MQTT Schnittstelle sind in https://github.com/svenjust/room-ventilation-system/tree/master/Docs/mqtt%20topics dokumentiert.

Komfortfunktionen, wie Zeitprogramme oder die Regelung der Lüftungsanlage anhand eines VOC Sensors, sind nicht implementiert. Diese Regelungsaufgaben können in übergeordneten Haussteuerung besser, weil spezifischer für den eigenen Anwendungsfall, implementiert werden. Der Autor verwendet fhem für diese Aufgaben. Die Einbindung dieser Steuerung in Fhem wird im Abschnitt 10 beschrieben. Die Messwerte der evtl. verbauten CO2, VOC, oder Feinstaubsensoren in der Lüftungsanlage werden per MQTT gepublishet.

2.2 Systemaufbau

Die Steuerung der Lüftungsanlage kann autark betrieben werden.

Sinnvoll ist ein Betrieb mit einer übergeordneten Haussteuerung. Die Installation von fhem und eines MQTT Brokers werden vorausgesetzt und sind nicht Inhalt dieses Papiers.

2.3 Regelungsstrategie der Steuerung

Der Inbetriebnehmer muss die Normdrehzahl für Zu- und Abluftventilator vorgeben. Differenzdrucksensoren sind nicht verbaut. Hersteller wie Helios verbauen auch keine solchen Sensoren. Pluggit stellt als Normdrehzahl jeweils 1.550 U/min werksseitig ein.

Die Steuerung versucht die vorgegebenen Drehzahlen für Zu- und Abluftventilator zu erreichen. Dem Lüfter wird als Stellgröße ein PWM-Signal oder eine Gleichspannung 0-10V vorgegeben. Über die Tacholeitung teilt der Lüfter über ein PWM Signal seine aktuelle Drehzahl mit. Im Normalbetrieb der Steuerung findet keine Regelung der Drehzahl statt. Die Steuerung kann in den Kalibrierungbetrieb gesetzt werden. Die Lüfter fahren dann die, für die verschiedenen Lüftungsstufen notwendigen Drehzahlen an, und ermitteln per PID-Regler das jeweils notwendige PWM-Signal. Das PWM-Signal für die Drehzahl für jede Lüftungsstufe wird dann getrennt für jeden Lüfter im EEPROM gespeichert. Im Normalbetrieb wird der Lüfter mit dem kalibrierten PWM Signal angesteuert. Die Strategie ist ähnlich dem Betrieb des Originalhersteller, hier findet zu definierten Zeiten (Montagsmorgens) eine „Kalibrierung“ statt. Die Kalibrierung muss per Touch-Display oder per MQTT gestartet werden.

2.4 Speicherung des Anlagenparameter

Die für einen sicheren Betrieb erforderlichen Anlagenparameter werden im EEPROM des Arduino Mega gespeichert. Der Inhalt des EEPROMs kann im Sourcecode und per MQTT-Befehl gelöscht werden. Die Anlage wird damit auf die sogenannte „Werkseinstellungen“ zurückgesetzt. Die Werkseinstellungen definieren für die P300 einen sicheren Betriebszustand.

3 Hardware

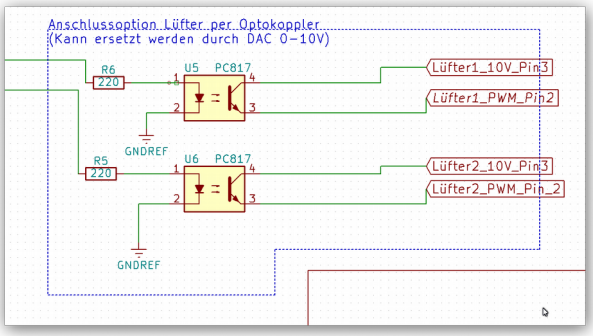
Der Hardwareaufbau kann in unterschiedlichen Ausbaustufen erfolgen, ebenso können verschiedene Sensoren ergänzt werden. Zuerst wird die absolut notwendige Hardware besprochen. Weiter unten folgen optionale Teile.

3.1 Notwendige Hardware

Für die Steuerung ist ein Arduino Mega erforderlich. Für die externe Kommunikation ist ein LAN-Modul erforderlich, aufgebaut und getestet ist das System mit WizNET (W5100) Ethernet Shield, siehe 11.2.

Die Temperaturen werden mit vier Temperatursensoren DS18B20 (siehe 11.5) im Edelstahlgehäuse erfasst und einzeln mit der Steuerung verkabelt. Durch die getrennte Verkabelung können die Sensoren sicher den Luftströmen zugeordnet werden. Für jeden Temperatursensor ist ein 4,7k Widerstand erforderlich.

In den KWL Geräten sind häufig Lüfter mit 230V Anschluss und Steuerung per 10V-PWM-Signal oder per Gleichspannung 0-10V verbaut. Den genauen Lüftertyp muss jeder für seine Anlage bestimmen. In den Pluggit Geräten P300/450 sind Lüfter von Papst verbaut, ähnlich wie http://img.ebmpapst.com/products/manuals/R3G190RC0503-BA-GER.pdf verbaut. In der gerade erwähnten Anleitung, Seite 9 sind die Schaltungsmöglichkeiten des Lüfters aufgeführt. Eine einfache Ansteuerung kann bei dem vorliegenden Lüfter durch Optokoppler erfolgen. Die Optokoppler werden direkt mit dem Lüfter verbunden, siehe folgendes Bild. Das Tachosignal kann direkt mit dem Arduino verbunden werden.



Hardwareliste (Mindestausstattung):

- 1 x Arduino Mega 2560
- 1 x LAN Module WizNET (W5100) Ethernet Shield
- 4 x OneWire DS18B20 in Edelstahlhülse
- 4 x Widerstand 4,7k
- 2 x Optokoppler PC817
- 2 x Widerstand 220 Ohm
- 1 x Netzteil 5V ODER 1 x Netzteil 12V plus LM2596 S für 5V Versorgung wenn DAC verwendet wird.

3.2 Alternative und optionale Hardware

3.2.1 Alternativ: DAC 0-10V für Vorheizregister und Lüfter

Alternativ zu den Optokopplern können die Lüfter mit einem 0-10V Gleichstromsignal angesteuert werden. Ein Arduino kann eine solche Gleichspannung nicht direkt zur Verfügung stellen. Ein Digital-Analog-Converter (DAC) wie z.B. https://www.horter-shop.de/de/home/93-bausatz-i2c-analog-input-modul-5-kanal-10-bit-4260404260752.html stellt die erforderliche Spannung bereit und kann einfach vom Arduino angesteuert werden.

Wenn ein Vorheizregister verwendet werden soll, ist die Ansteuerung per DAC evtl. notwendig.

3.2.2 Optional: TouchDisplay

Für eine Visualisierung am Gerät bzw. eine Bedienung ohne Automatisierungssystem kann ein Touchdisplay (siehe 11.6) verwendet werden. Die Steuerung setzt auf die mcufriend-Library auf und geht von einer Auflösung von 480x320 Pixel aus. Das Touchdisplay sollte wenigstens 3,5“ groß sein.

3.2.3 Optional: Sommer-Bypass per Relais

Wenn das Gerät einen Sommer-Bypass eingebaut hat, kann dieser ansteuert werden. Die Steuerung enthält die dafür notwendigen Funktionen.

GEFAHR: Verschiedene Gerätehersteller bauen bzgl der Ansteuerspannung unterschiedliche Lösungen ein. Hier ist vor Beginn genau zu prüfen, was im eigenen Gerät verbaut.

In Pluggit-Geräten der Serien P300/450 wird der Sommer-Bypass mit 230V~ geschaltet. Die Schaltung entspricht der klassischen Rolladensteuerung, d.h. für die Fahrdauer wird eine Spannung angelegt und nach Ausschalten der Spannung bleibt der Bypass in der aktuellen Stellung stehen.

Helios schaltet den Sommer-Bypass mit 24V=.

Die Schaltung für den Sommer-Bypass entspricht klassischen Rolladensteuerung, es werden zwei Relais verbaut, das erste Relais schaltet die Spannung, das zweite (Wechsel-) Relais bestimmt die Fahrrichtung.

3.2.4 Optional: CO2 Sensor

Mit dem folgenden CO2 Sensor hat der Autor gute Erfahrungen gemacht: MH-Z14, https://www.sensor-test.de//ausstellerbereich/upload/mnpdf/en/MHZ14_14.pdf . Das Signal wird über eine serielle Schnittstelle übertragen.

Die Messwerte des CO2 Sensors werden per MQTT an übergeordnete Geräte übertragen. Eine Regelung der Lüftungsanlage anhand des CO2-Sensors, ist nicht implementiert. Diese Regelungsaufgaben kann in einer übergeordneten Haussteuerung besser, weil spezifischer für den eigenen Anwendungsfall, implementiert werden.

3.2.5 Optional: VOC Sensor

Der Autor verwendet den folgenden VOC Sensor Figaro TGS2600, http://www.figarosensor.com/products/2600pdf.pdf . Das Signal wird durch einen Widerstandswert übertragen. Für das Steuern der Lüftungsanlage ist der VOC Sensor nach Erfahrung des Autors ausreichend, der CO2-Sensor dient eher dem Spiel- und Messtrieb. Beide Werte korrelieren weitgehend.

Die Messwerte des VOC Sensors werden per MQTT an übergeordnete Geräte übertragen. Eine Regelung der Lüftungsanlage anhand des VOC-Sensors, ist nicht implementiert. Diese Regelungsaufgaben kann in einer übergeordneten Haussteuerung besser, weil spezifischer für den eigenen Anwendungsfall, implementiert werden.

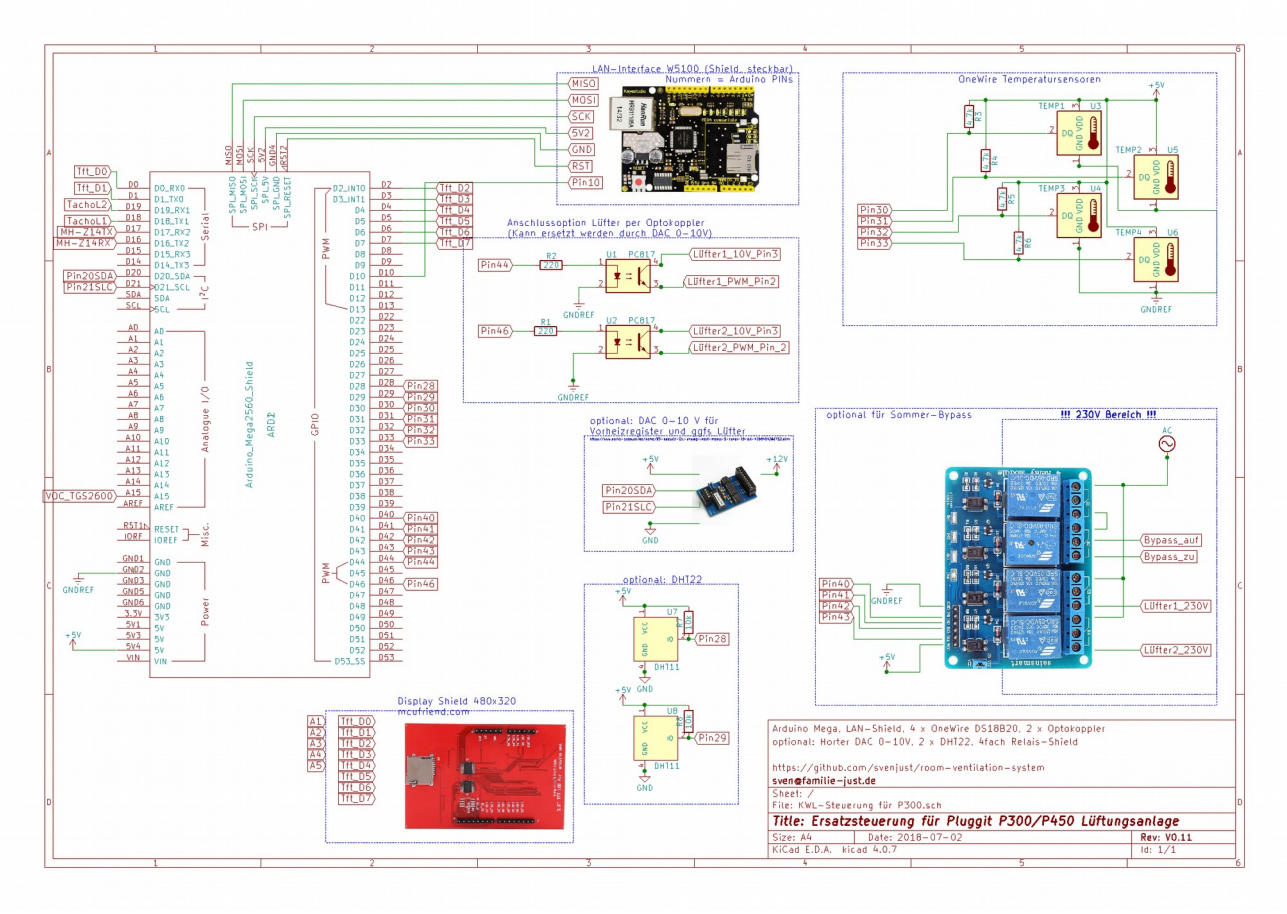
3.2.6 Optional: Feinstaubsensoren

Bisher nicht implementiert.

3.2.7 Optional: Luftfeuchtesensoren DHT22

Für die Messung der Luftfeuchte im Abluft- und Zuluftstrom können bis zu zwei DHT22 Sensoren eingebaut werden. Die Messwerte der Sensoren werden per MQTT an übergeordnete Geräte übertragen. Eine Regelung der Lüftungsanlage anhand der DHT22-Sensoren, ist nicht implementiert.

3.3 Schaltplan



Je Lüfter wird für den elektrischen Anschluss ein 3- und 4-poliger Stecker und Gehäuse (Mate-N-Lok) mit der passenden Anzahl Steckkontakten benötigt. Bezugsquelle: https://www.reichelt.de/Universal-Mate-N-Lok/2/index.html?ACTION=2&LA=2&GROUPID=7501

3.4 Stückliste

Anzahl	Bauplan	Bezeichnung
4	R1 - R4	Widerstand 4,7k Ohm
2	R5, R6	Widerstand 220 Ohm
4		DS18B20 im Edelstahlgehäuse
2		Optokoppler PC817
1		Arduino Mega R3
1		LAN Shield W5100
1		optional: 4 fach Relais Arduino für Sommer-Bypass
1		optional: 4 fach DAC 0-10V für das Vorheizregister, bspw. von https://www.horter-shop.de/de/home/93-bausatz-i2c-analog-input-modul-5-kanal-10-bit-4260404260752.html
1 oder 2		optional: DHT22 Luftfeuchtesensoren, für Abluft- und ggfs Zuluftstrom
		optional: VOC Sensor, z.B. Figaro TGS2600
		optional: CO2-Sensor, z.B. MH-Z14
1		Markenschaltnetzteil, bspw. von der Firma Meanwell für 5 V (Schaltung OHNE DAC) z.B. Meanwell RS-15-5 oder DR-15-5 (DIN Schiene) für 12V (Schaltung mit DAC) z.B. Meanwell RS-15-12 oder DR-15-12 (DIN Schiene) **plus** LM2596 S Abwärtswandler Buck Schaltregler für 5V Versorgung

3.5 Aufbau der Schaltung

Es wird ein fachkundigen Aufbau der Schaltung vorausgesetzt. Dabei ist zwischen den Niederspannungsseite und der 230V-Seite zu trennen. Für die Stromversorgung wird ein gekapseltes externes „Markennetzteil“ empfohlen.

Für die Schaltung existiert momentan keine fertige Platine. Die Schaltung wird dementsprechend auf eine Lochrasterplatine gebaut, Beispielhaft siehe https://forum.fhem.de/index.php/topic,22822.msg733992.html#msg733992

Wenn ein Abwärtswandler eingesetzt wird, so ist auf die korrekte Einstellung der Ausgangsspannung von 5V zu achten.

Die Verbindungen der Bauteile sind zu löten, bzw. zu schrauben. Der Aufbau auf einem Steckbrett ist allenfalls für Tests möglich und wird bei den vorhandenen Vibrationen und Luftbewegungen einer Lüftungsanlage nicht lange stabil sein. Nach dem Löten sind die Lötverbindungen zu kontrollieren.

4 Grundeinstellungen Soucecode

4.1 Definition der Anschlüsse

Abhängig vom tatsächlichen Aufbau der Hardware müssen die Anschlüsse (PIN) für Ein- und Ausgänge kontrolliert bzw. umdefiniert werden, insbesondere werden Displays unterschiedlich angeschlossen. Aktuell verwenden wir Displays für die Anschlüsse A0-A9 und 0-13.

Für die Auswertung des Tachosignals sind Interrupt-fähige Eingänge notwendig, aktuell Pin 18, 19.

Der DAC verwendet die Anschlüsse 20 und 21.

Der CO2-Sensor MH-Z14 verwendet die zweite seriellen Schnittstelle Serial2, Pin 16 und 17.

Die Temperatursensoren DS18B20 verwendet die Pins 30 bis 33.

Die Luftfeuchtesensoren DHT1 und DHT2 verwenden die Pins 28 und 29.

Das Relais für den Bypass und die Lüfter verwendet die Anschlüsse 40 bis 43.

Der VOC Sensor verwendet den Analogeingang A15.

Im Sourcecode beginnt und endet der konfigurierbare Bereich mit „A N S C H L U S S E I N S T E L L U N G E N“

4.2 Netzwerkeinstellungen

Es muss die MAC- und die IP Adresse, Subnet, Gateway, DNS dieser Steuerung einstellt werden, ebenso die IP-Adresse des mqtt-Brokers.

4.3 „Werkseinstellung“

Die für einen Betrieb erforderlichen Anlagenparameter werden im EEPROM des Arduino Mega gespeichert.

Durch das Setzen von `#define FACTORY_RESET_EEPROM` **`true`** wird bei jedem Neustart der Steuerung der EEPROM-Speicher des Arduinos gelöscht und mit den Werkseinstellungen

überschrieben. Für den regulären Betrieb muss `#define FACTORY_RESET_EEPROM false` definiert sein. Das EEPROM des Arduinos ist für 100.000 Schreibzyklen ausgelegt.

Alternativ kann das EEPROM mit dem

```
mosquitto_pub -t d15/set/kwl/resetAll_IKNOWWHATIMDOING -m YES
```

gelöscht werden. Anschließend wird die Steuerung neu gestartet.

Werkeinstellungen sind in der Software durch den Prefix **defStandard** zu erkennen. Es sind die folgenden „Werkseinstellungen“ definiert.

Name	*Wert*	*Beschreibung*
defStandardModeCnt	4	Anzahl der unterschiedlichen Lüftungsstufen
defStandardKwlModeFactor	{0, 0.7, 1, 1.3}	Definition der Drehzahl je Stufe in Relation zu defStandardSpeedSetpointFan
defStandardSpeedSetpointFan1	1550	Drehzahl in Standardlüftungsstufe für den Zuluftventilator
defStandardSpeedSetpointFan2	1550	Drehzahl in Standardlüftungsstufe für den Abluftventilator
defStandardNenndrehzahlFan	3200	Nenndrehzahl des verbauten Ventilators, siehe Typenschild. Die Nenndrehzahl wird benötigt für die Berechnung der Ansteuerung im unkalibrierten Betrieb
defStandardBypassTempAbluftMin	24	Mindestablufttemperatur für die Öffnung des Bypasses im Automatik Betrieb
defStandardBypassTempAussenluftMin	13	Mindestaussenlufttemperatur für die Öffnung des Bypasses im Automatik Betrieb
defStandardBypassHystereseMinutes	60	Hysteresezeit für eine Umstellung des Bypasses im Automatik Betrieb
defStandardBypassHystereseTemp	3	Hysteresetemperatur für eine Umstellung des Bypasses im Automatik Betrieb
defStandardBypassManualSetpoint	1 (Closed)	Stellung der Bypassklappen im manuellen Betrieb
defStandardBypassMode	0 (Auto)	Automatik oder manueller Betrieb der Bypassklappe. Im Automatikbetrieb steuert diese Steuerung die Bypass-Klappe, im manuellen Betrieb wird die Bypass-Klappe durch mqtt-Kommandos gesteuert.
Speicherplatz 20 bis 60		PWM Signal zur Erreichung der Drehzahlen der beiden Lüfter für die verschiedenen Lüfterstufen nach der Kalibrierungsfahrt. Die Kalibrierungsfahrt wird durch einen mqtt-Befehl gestartet.
defStandardHeatingAppCombUse	0	Einstellung für den Feuerstättenbetrieb. 0 = keine Feuerstätte, 1 = Feuerstätte, Kamin, Ofen ohne Fremdluftzufuhr

4.4 Ansteuerung der Relais

Für die Lüfter und den Sommer-Bypass können bis zu vier Relais verbaut sein. Ohne Sommer-Bypass kann die Steuerung auch ohne Relais betrieben werden. Da verschiedene Relais unterschiedlich geschaltet werden, kann hier die logische Schaltung definiert werden.

```
#define RELAY_ON   LOW
#define RELAY_OFF  HIGH
```

4.5 Debugausgaben

Es können Debugausgaben auf der seriellen Schnittstelle für unterschiedliche Bereiche ein oder ausgeschaltet werden.

```
// ************************************************  D E B U G E I N S T E L L U N G E
N ******************************************************
int serialDebug = 1;             // 1 = Allgemein Debugausgaben auf der seriellen
Schnittstelle aktiviert
int serialDebugFan = 1;          // 1 = Debugausgaben für die Lüfter auf der seriellen
Schnittstelle aktiviert
int serialDebugAntifreeze = 0;   // 1 = Debugausgaben für die Antifreezeschaltung auf der
seriellen Schnittstelle aktiviert
int serialDebugSummerbypass = 0; // 1 = Debugausgaben für die Summerbypassschaltung auf
der seriellen Schnittstelle aktiviert
int serialDebugDisplay = 0;      // 1 = Debugausgaben für die Displayanzeige
int serialDebugSensor = 1;       // 1 = Debugausgaben für die Sensoren
// ****************************************E N D E ***  D E B U G E I N S T E L L U N
G E N ******************************************************
```

5 Build der Software für Arduino

Die Software wird in der Arduino IDE gebaut. Debugausgaben für die serielle Schnittstelle können im Code aktiviert werden.

5.1 Notwendige Bibliotheken

Für das Bauen der Steuerung sind Bibliotheken notwendig.

- DallasTemperature, Quelle: https://github.com/milesburton/Arduino-Temperature-Control-Library

Die weiteren Bibliotheken sind im Library Manager der Arduino IDE verfügbar.

6 Inbetriebnahme

Vor der Inbetriebnahme sind unbedingt die Pinbelegungen für die Motoren und Sensoren, die Netzwerkeinstellungen und die „Werkseinstellungen“ zu kontrollieren und entsprechend den eigenen Bedürfnissen einzustellen. WICHTIG: Wenn Werkseinstellungen im Sourcecode geändert werden, muss das EEPROM gelöscht werden, damit die neuen Werkeinstellungen verwendet werden, siehe 4.3.

Eine schrittweise Inbetriebnahme wird empfohlen. Die Inbetriebnahme kann über das Touchdisplay oder den MQTT Broker erfolgen.

6.1 Feuerstättenmodus (Kaminbetrieb)

Wenn eine Feuerstätte (Kamin/Ofen) im Haus ohne Fremdluftzufuhr betrieben wird, besteht die Gefahr, dass bei negativen Außentemperaturen der Zuluftlüfter abgeschaltet wird und giftiges Kohlenmonoxid (CO) aus der Feuerstätte in das Haus gesaugt werden.

- LEBENSGEFAHR -

Mit dem Einstellen des Feuerstättenmodus werden im Antifrostbetrieb beide Lüfter für vier Stunden ausgeschaltet. Das Wiedereinschalten vor vier Stunden ist nur durch einen Neustart der Steuerung möglich.

Der Feuerstättenmodus kann auf zwei Arten eingeschaltet werden, entweder im Sourcecode oder durch Zusenden einer MQTT Nachricht. Bei beiden Arten wird die Einstellung im EEPROM gespeichert und bleibt bei einem Neustart erhalten. Empfohlen wird die **Einstellung im Sourcecode** anzupassen, da diese Einstellung auch bei einer Rücksetzung auf Werkeinstellungen erhalten bleibt.

Im Sourcecode muss anstatt

```
#define  defStandardHeatingAppCombUse          0
```

eingestellt werden:

```
#define  defStandardHeatingAppCombUse          1
```

Anschließend ist eine Rückstellung auf Werkseinstellungen notwendig, siehe 4.3.

Alternativ kann folgendes per MQTT an die Steuerung geschickt werden:

```
mosquitto_pub -t d15/set/kwl/heatingapp/combinedUse -m YES
```

Diese Einstellung wird gespeichert, aber bei einer Werkeinstellungen zurückgesetzt!

Beim Starten der Steuerung kann die Einstellung überprüft werden, es wird beim Feuerstättenmodus ausgegeben:

```
...System mit Feuerstaettenbetrieb
```

6.2 Optional: MQTT Broker

Für den Betrieb ist ein MQTT Broker sinnvoll. Die Installation eines MQTT Brokers ist an vielen Stellen im Netz beschrieben, u.a. hier: https://wiki.fhem.de/wiki/MQTT_Einf%C3%Bchrung. Wenn der MQTT-Broker installiert ist und die Steuerung gestartet wird, so sendet die Steuerung alle 30 Sekunden ein Heartbeat Signal, diese kann bei einem Mqtt-Client mit

```
mosquitto_sub -v -h localhost -t d15/state/kwl/heartbeat
```

abonniert werden.

6.3 Temperatursensoren

Die vier Temperatursensoren (DS18B20) sind notwendig für einen sicheren Betrieb der Lüftungsanlage. Bei niedrigen Außentemperaturen kann ansonsten der Wärmetauscher durch Frost zerstört werden.

Die Temperatursensoren sind an die Platine anschließen. Im Sourcecode ist die Variable serialDebug = 1 setzten und auf den Arduino flashen. Im seriellen Monitor werden die Temperaturen bei Änderung angezeigt. Für eine Temperaturänderungen können die Sensoren in die Hand genommen werden. Temperaturen von -127.0 zeigen, dass der Sensor nicht gefunden wurde.

Wenn die Temperatursensoren in der Anlage verbaut und verkabelt sind, sollte die korrekte Zuordnung zu den Luftströmen überprüft werden. Bei einer Außentemperatur von 5°C und einer Wohnraumtemperatur von 22 °C hat die Außenluft (T1) die geringste Temperatur, etwa 8 °C, anhängig von den baulichen Gegebenheiten. Die Zuluft (T2) wird bei etwa 18°C liegen. Die Abluft (T3) wird bei etwa 21°C und die Fortluft (T4) bei etwa 10°C liegen (T1 < T4 < T2 < T3). Wird diese Reihenfolge nicht eingehalten, es entweder die Außentemperatur sehr hoch oder die Zuordnung der Temperatursensoren zu den Luftströmen ist nicht korrekt.

6.4 Lüfter

Da die Lüfter üblicherweise mit Netzspannung betrieben werden, ist besondere Vorsicht und Fachkenntnis notwendig.

Die Lüfter mit Netzspannung versorgen und an die Platine anschließen. Im Sourcecode die Variable serialDebugFan = 1 setzten und den Arduino flashen. Im Seriellen Monitor werden die Drehzahlen bei Änderung angezeigt. Hier sollte überprüft werden, dass das Tachosignal der Lüfter korrekt erkannt wird.

In der Werkeinstellungen sind vier Lüftungsstufen definiert. Lüftungsstufe 0 ist aus, 1, 2, 3 sind mit jeweils 70,% 100%, bzw 130% Drehzahl der Standardlüftungsstufe definiert. 100% Drehzahl entspricht 1.550 U/min. Die Lüftungsstufen sollten getestet werden:

```
mosquitto_pub -t d15/set/kwl/lueftungsstufe -m 1
mosquitto_pub -t d15/set/kwl/lueftungsstufe -m 2
mosquitto_pub -t d15/set/kwl/lueftungsstufe -m 3
```

Wenn der Test erfolgreich war, kann eine Kalibrierung der Lüfteransteuerung vorgenommen werden. Dabei werden die Lüfter in die Lüftungsstufen geschaltet und es werden die PWM-Signale gespeichert, die zur Erreichung der jeweiligen Drehzahlen erforderlich sind. Diese gespeicherten PWM-Signale werden bis zur nächsten Referenzfahrt verwendet.

```
mosquitto_pub -t d15/set/kwl/calibratefans -m YES
```

Die Kalibrierung dauert mehrere Minuten.

6.5 Optional: Sommer-Bypass

Wenn der Sommer-Bypass mit Netzspannung betrieben wird, ist besondere Vorsicht und Fachkenntnis beim elektrischen Aufbau notwendig.

Der Sommer-Bypass ist in der Grundeinstellung im Automatikbetrieb und die Klappe ist geschlossen. Die Betriebsart (Automatik/Manuell) wird im EEPROM gespeichert, ebenso wie die Klappenstellung im manuellen Betrieb.

Um das korrekte Öffnen und Schließen der Klappe in der manuellen Betriebsart zu überprüfen, muss der Sommer-Bypass in den Manuellen Betrieb gesetzt werden und die Klappe per MQTT-Befehl geöffnet und geschlossen werden.

Die folgenden Kommandos setzen die manuelle Betriebsart und öffnen die Klappe.

```
mosquitto_pub -t d15/set/kwl/summerbypass/mode -m manual
mosquitto_pub -t d15/set/kwl/summerbypass/flap -m open
```

Bis zum Schalten der Bypass-Klappe können bis zu 6 Minuten vergehen, nach dem die erforderlichen Bedingungen vorliegen.

Das folgende Kommando schließt die Klappe, die Verzögerung kann ebenfalls wieder bis zu sechs Minuten betragen:

```
mosquitto_pub -t d15/set/kwl/summerbypass/flap -m close
```

Wenn die Klappe in der manuellen Betriebsart korrekt gefahren wurde, kann der Automatikbetrieb überprüft werden. Der Sommer-Bypass öffnet im Automatikbetrieb, wenn die folgenden Bedingungen erfüllt sind:

1. Die Außenlufttemperatur ist höher als TempAussenluftMin (13°C)
2. Die Ablufttemperatur ist höher als TempAbluftMin (24°C)
3. Die Ablufttemperatur ist 2 Grad höher als die Außenlufttemperatur
4. Der Bypass wurde in den letzten HystereseMinutes (60) Minuten nicht geschlossen.

Die Werkseinstellungen für den Sommer-Bypass entsprechen den am Markt anzutreffenden Anlagen.

Für das Testen des Bypasses sind die Temperatursensoren zu entfernen. Es ist ausreichend, die Datenleitungen der Sensoren vom Arduino zu lösen. Anschließend können die Betriebsart und die Temperaturwerte über den MQTT-Broker in die Steuerung geschrieben werden.

```
mosquitto_pub -t d15/set/kwl/summerbypass/mode -m auto
mosquitto_pub -t d15/debugset/kwl/aussenluft/temperatur -m 14
mosquitto_pub -t d15/debugset/kwl/abluft/temperatur -m 26
```

Der Sommer-Bypass sollte jetzt geöffnet werden.

Die Fahrdauer kann mit der Variablen intervalBypassSummerSetFlaps im Sourcecode angepasst werden.

6.6 Optional: Vorheizregister

Das Vorheizregister wird automatisch bei drohender Vereisung aktiviert. Für das Testen des Bypasses sind die Temperatursensoren zu entfernen. Es ist ausreichend, die Datenleitungen der Sensoren vom Arduino zu lösen. Anschließend können per mqtt die folgenden Temperaturen vorgegeben werden. Das Vorheizregister sollte jetzt heizen. ACHTUNG: Das Vorheizregister darf nur im Luftstrom betrieben werden.

```
mosquitto_pub -t d15/debugset/kwl/fortluft/temperatur   -m 1
mosquitto_pub -t d15/debugset/kwl/aussenluft/temperatur -m -1.1
```

7 Einstellungen im Betrieb

Alle mqtt-Topics sind in der Datei https://github.com/svenjust/room-ventilation-system/tree/master/Docs/mqtt%20topics/mqtt_topics.ods dokumentiert, ebenso deren Implementierungsstatus. In der Datei sind auch kurze Erläuterung zu den Topics.

Für das Mitlesen aller mqtt-Meldungen der Lüftungsanlage sind zwei Subscribes notwendig:

```
mosquitto_sub -v -h localhost -t d15/state/
mosquitto_sub -v -h localhost -t d15/debugstate/
```

7.1 Setzen der Lüftungsstufe

```
mosquitto_pub -t d15/set/kwl/lueftungsstufe -m <Nr der Lüftungsstufe>
```

8 Steuerung der Bypass-Klappe

Um die Bypass-Klappe manuell zu steuern muss sie in den manuellen Betriebsmodus geschaltet werden.

```
mosquitto_pub -t d15/set/kwl/summerbypass/mode -m manual
```

Anschließend kann die Bypass-Klappe geöffnet bzw. geschlossen werden.

```
mosquitto_pub -t d15/set/kwl/summerbypass/flap -m open
mosquitto_pub -t d15/set/kwl/summerbypass/flap -m close
```

9 Bedienung mit Touch-Display

Doku offen...

10 Einbindung dieser Steuerung in fhem

Für die Einbindung ist ein mqtt Broker erforderlich, siehe Abschnitt 6.2. Im fhem muss ein Device für die Verbindungen zum mqtt Broker erstellt werden.

```
define myBroker MQTT 192.168.20.240:1883 ## bitte EIGENE IP-Adresse eintragen
```

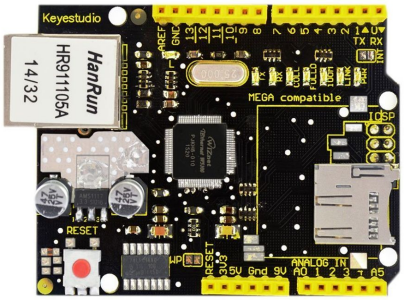
Anschließend muss ein Device für die Lüftungsanlage erstellt werden:

```
define kwl MQTT_DEVICE
attr kwl IODev myBroker
attr kwl publishSet 0 1 2 3 d15/set/kwl/lueftungsstufe
attr kwl room DEV_KWL
attr kwl stateFormat transmission-state
attr kwl subscribeReading_Fan1Speed d15/state/kwl/fan1/speed
attr kwl subscribeReading_Fan2Speed d15/state/kwl/fan2/speed
attr kwl subscribeReading_StateLueftungsstufe d15/state/kwl/lueftungsstufe
attr kwl subscribeReading_Temp01Aussenluft d15/state/kwl/aussenluft/temperatur
attr kwl subscribeReading_Temp02Zuluft d15/state/kwl/zuluft/temperatur
attr kwl subscribeReading_Temp03Abluft d15/state/kwl/abluft/temperatur
attr kwl subscribeReading_Temp04Fortluft d15/state/kwl/fortluft/temperatur
```

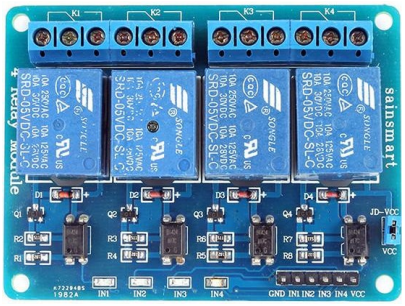
11 Bilder der verwendeten Hardware

11.1 Arduino Mega

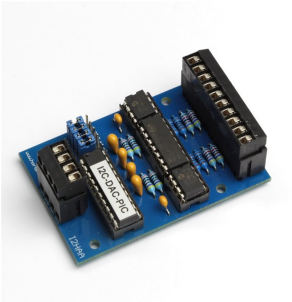
11.2 LAN Shield mit W5100



11.3 Relais Shield

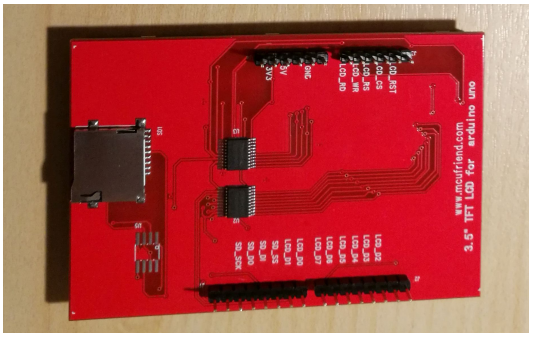


11.4 DAC von Horter



11.5 Temperatursensoren DS18B20

11.6 Touch-Display 3,5“, mcufriend-Library, Auflösung 480x320



Wichtig ist, dass das Display auf das LAN Shield gesteckt werden kann, die Pin also an den beiden Längsseiten und nicht unten an der Querseite hat.