CO_2 goes Homematic

Wie sich ein CO_2-Messgerät in Homematic einbinden lässt

Das bei ELV angebotene CO_2-Messgerät technoline WL1030 ist Testsieger eines Vergleichstests der Stiftung Warentest. Das Messgerät bietet neben einem großen, gut ablesbaren und dimmbaren Display viele Zusatzfunktionen wie einen optischen und akustischen CO_2-Alarm und eine Anzeige von Temperatur und Luftfeuchtigkeit. Lediglich eine Kommunikation mit der Außenwelt ist nicht vorhanden. Mit der in diesem Beitrag beschriebenen Modifikation liefert das Messgerät die CO_2-Werte an das Homematic System. Ein zusätzlich eingebauter Luftdrucksensor erweitert das Gerät zu einer fast kompletten Wetterstation.

*Siehe Seite 112

Peter Tschulik
hat für seinen Beitrag zum Leserwettbewerb einen Gutscheincode* über 200,– Euro erhalten.

Hinweis:
Alle hier vorgestellten Modifikationen geschehen auf eigenes Risiko und eigene Gefahr.
Außerdem erlöschen sämtliche Garantieansprüche.

CO_2-Werte für das Smart Home

Ziel des Projekts war es, das CO_2-Messgerät technoline WL1030 [1] mit möglichst wenigen Änderungen am Gerät so umzurüsten, dass es die CO_2-Werte an ein Homematic System übermittelt. Von Homematic IP gibt es zwar einen eigenen CO_2-Sensor [2], der allerdings nur für den Einbau in eine Unterputzsteckdose verfügbar ist.

Zum Einsatz kommt neben einem Arduino Pro Micro (s. Materialliste [5]) das hoch interessante ELV Modul HM-MOD-EM-8Bit [3], das komfortabel 8-Bit-Werte im Homematic System übertragen kann. Da der Arduino Pro Micro mit der Übertragung des CO_2-Wertes bei Weitem nicht ausgelastet ist, wurde außerdem ein Bosch BME280-Modul (s. Materialliste [5]) angeschlossen, das präzise den Luftdruck misst. Bei der Auswahl des Moduls ist darauf zu achten, dass es sich um eine 5-V-Version handelt, die neben einem Spannungsregler auch einen Levelshifter auf dem Breakout-Board enthält.

Jede Modifikation beginnt damit, das Gerät zu öffnen und von innen zu inspizieren. Beim technoline WL1030 ist das sehr leicht möglich. Man löst jeweils drei Kreuzschlitzschrauben rechts und links und eine ein wenig verborgene Schraube in der Mitte unten (unter dem Ständer). Der Ständer ist nur in die Öffnung eingeklickt und kann leicht entfernt werden. Danach kann das Gehäuse nach oben abgehoben werden.

Nur der Buzzer der Rückwand ist mit Kabeln an der Hauptplatine angelötet. Sofort nach dem Öffnen sieht man den Single-Beam-NDIR-CO_2-Sensor (nicht dispersive Infrarottechnologie), der von der Firma Cubic stammt. Anhand der Beschriftung der Pins ist der Sensor leicht auf der Cubic-Webseite zu finden [4], es handelt sich um den Typ CM1106H-NS. Ein leicht verständliches Datenblatt mit der Beschreibung des Protokolls kann von der Webseite heruntergeladen werden und ist auch in der Materialliste verfügbar [5]. Daraus ist ersichtlich, dass Daten nach einer entsprechenden Aufforderung mit 9600 Baud, 8 Bit, No parity und einem Stoppbit gesendet werden.

Mit einem Oszilloskop kann man die Übertragung verifizieren und feststellen, dass der Mikroprozessor des WL1030 ca. alle 30 Sekunden den aktuellen CO_2-Wert vom Sensor abfragt. Deshalb ist es möglich, auf der seriellen Schnittstelle des Sensors mitzulauschen. Mit dieser Information wurde der Schaltplan (Bild 1) entwickelt.

Für den Arduino Pro Micro habe ich mich aus zwei Gründen entschieden. Erstens ist er trotz seiner vielen Anschlüsse sehr kompakt, und zweitens bietet er zwei serielle Schnittstellen an, da die serielle Schnittstelle über den USB-Stecker von der seriellen Schnittstelle (TXO, RXI) – anders als z. B. beim Arduino Uno – getrennt ist. Der Arduino Pro Micro verwendet den gleichen Prozessor wie der Arduino Leonardo und kann mit den entsprechenden Einstellungen programmiert werden (s. Abschnitt Firmware aufspielen).

Das BME280-Modul ist über die I^2C-Schnittstelle über SCL und SDA an den Arduino angeschlossen. Das HM-MOD-EM-8Bit-Modul wird – wie in der dazugehörigen Bedienungsanleitung beschrieben ([3], Downloadbereich) – über den Kanal 3 angesteuert (s. Abschnitt Einbindung in Homematic).

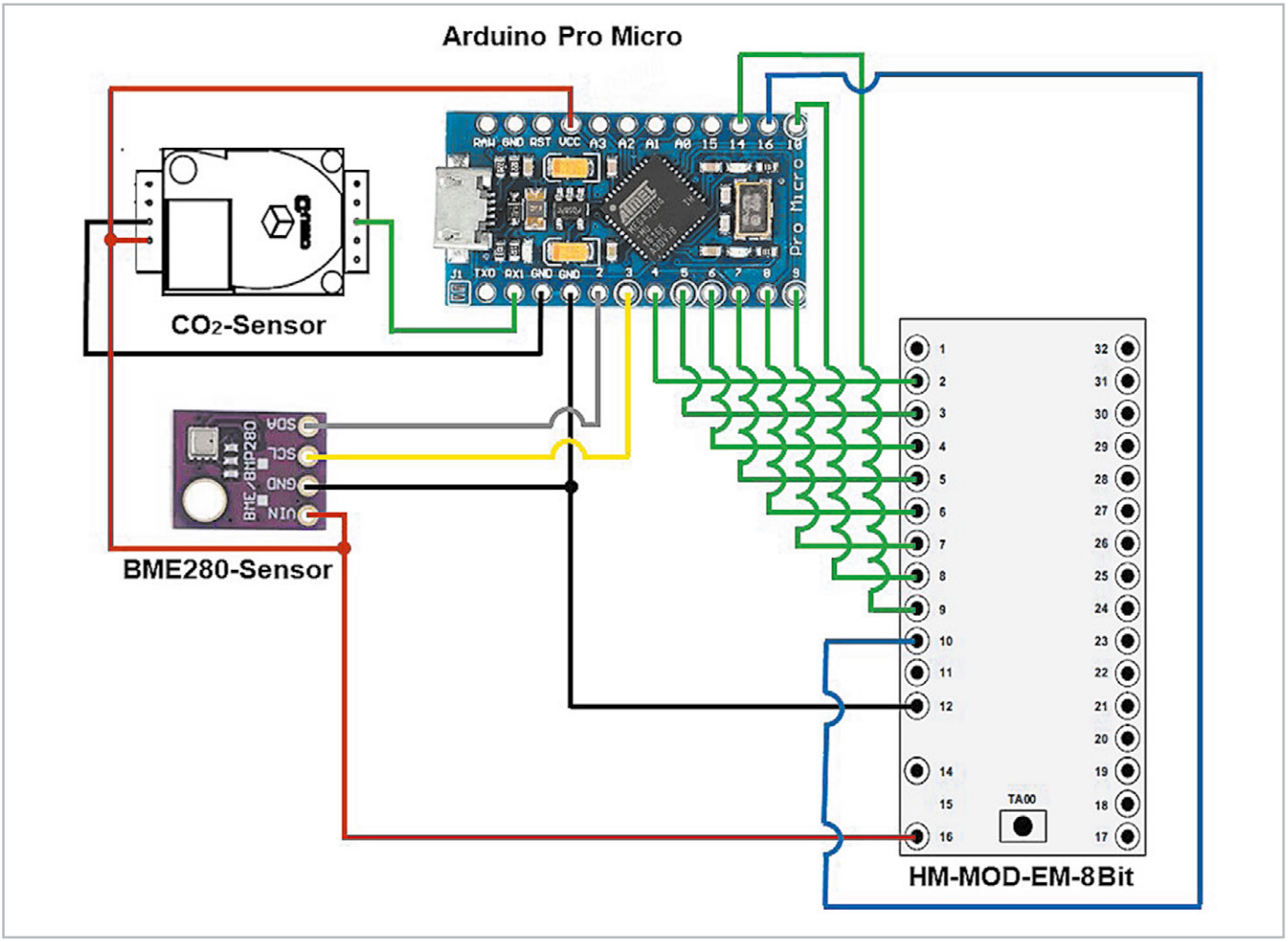

Bild 1: Schaltplan

Nachbau

Nach dem Öffnen des Gehäuses werden zuerst einmal drei Kabel an den CO_2-Sensor angelötet: ein rotes Kabel für die 5-V-Stromversorgung, ein schwarzes Kabel für GND und ein grünes Kabel für die Daten, wie in Bild 2 gezeigt.

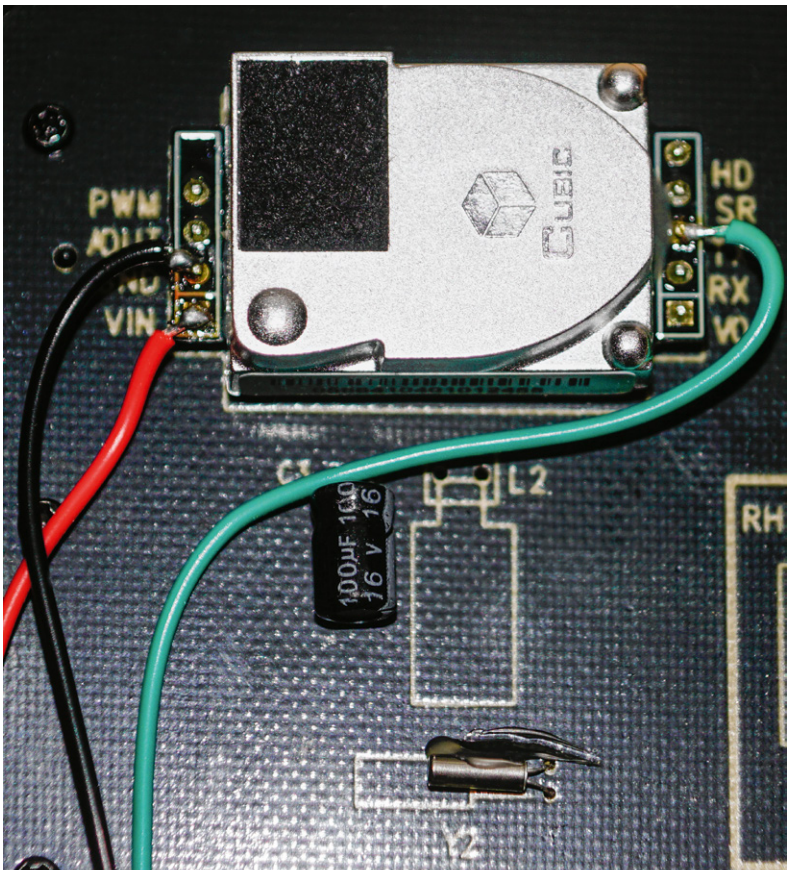

Bild 2: Der CO_2-Sensor wird „angezapft“.

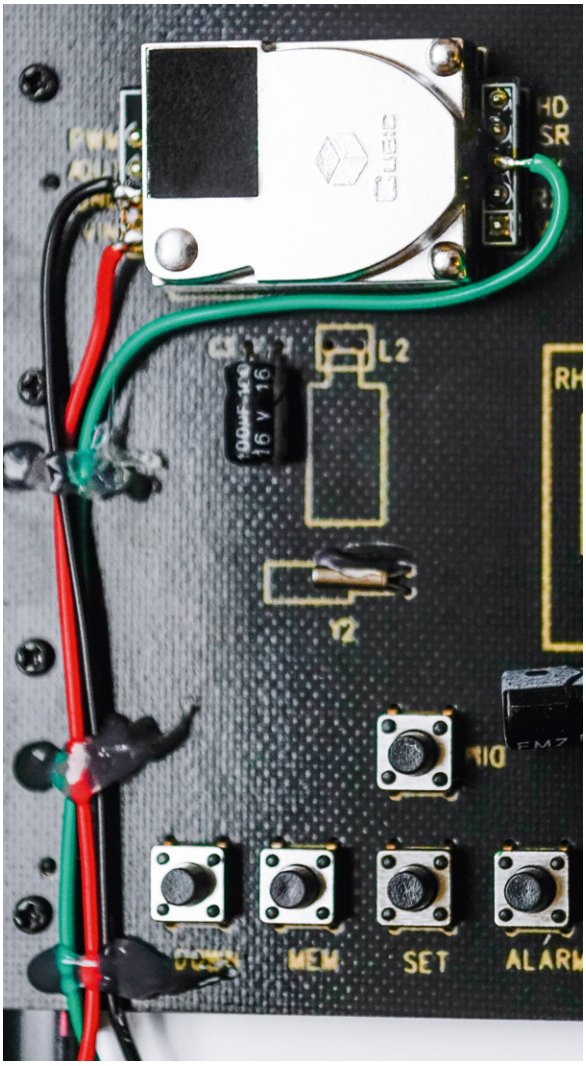

Bild 3: Fixierung der Kabel an der Hauptplatine

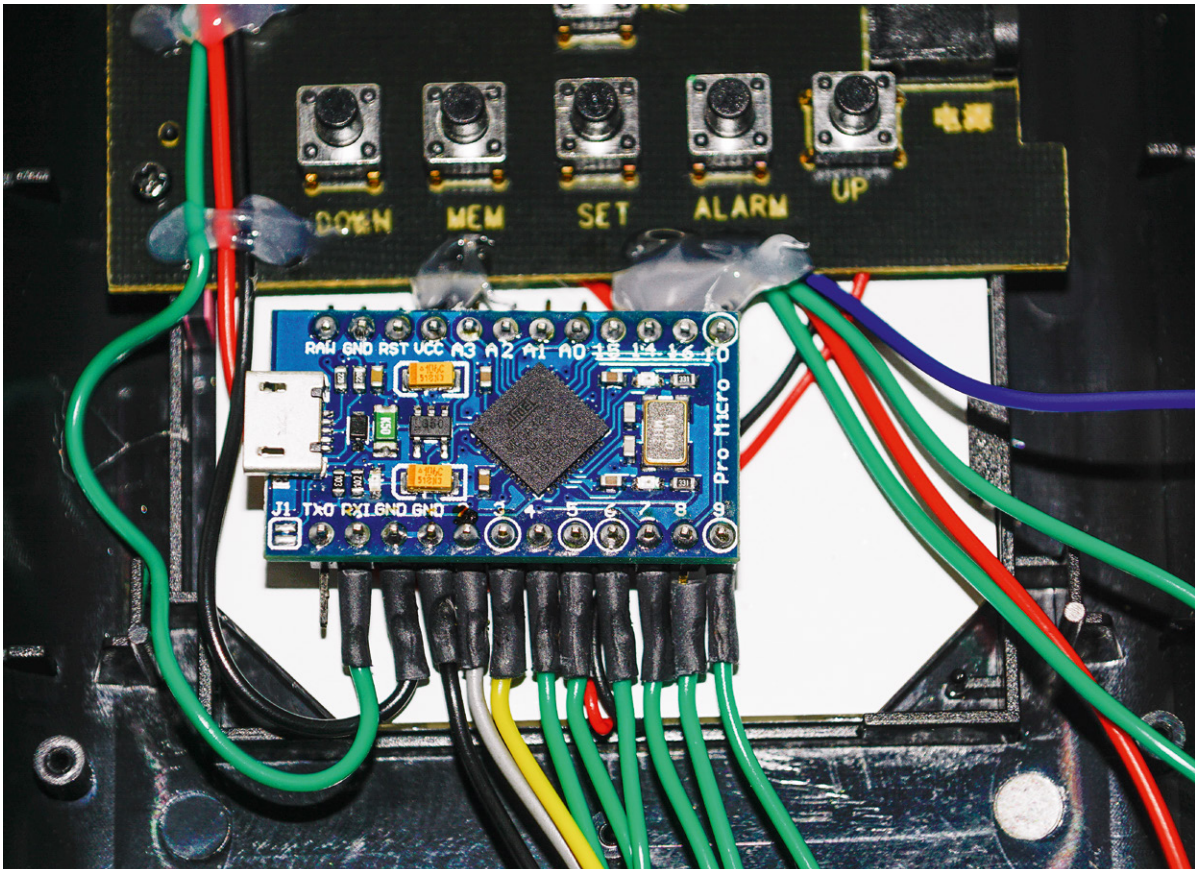

Bild 4: Verkabelung und Fixierung des Arduino

Wie in Bild 1 gezeigt, wird das grüne Kabel mit dem RXI-Pin des Arduino Leonardo Micro verbunden. Über die Verbindung des schwarzen Kabels mit GND und des roten Kabels mit VCC des Arduino Leonardo Micro wird dieser vom technoline WL1030 mit Strom versorgt.

Danach werden die Drähte mit einem Heißkleber an der Hauptplatine fixiert (Bild 3).

In einem nächsten Schritt werden die Stiftleisten des Arduino Pro Micro vorsichtig nach außen gebogen und ca. 20 cm lange Kabel in verschiedenen Farben – wie in Bild 1 gezeigt – angelötet und mit Schrumpfschläuchen isoliert. Danach wird der Arduino Pro Micro mit Heißkleber an der Hauptplatine fixiert, wie Bild 4 zeigt.

Firmware aufspielen

Als Nächstes wird die Firmware auf den Arduino aufgespielt. Dazu muss vor dem Verbinden des Arduino mit dem PC über ein USB-Kabel das technoline WL1030 mit dem mitgelieferten Netzteil verbunden werden, damit der Arduino durch das technoline WL1030 mit Strom versorgt wird.

Vor dem Aufspielen der Software muss noch die Adafruit Sensor Library [6] installiert werden. Die zweite Library zur Ansteuerung des BME280 findet sich im Verzeichnis des Source Codes und muss nicht getrennt installiert werden.

Im Arduino Hauptprogramm WL1030_Hack.ino müssen nur wenige Anpassungen getätigt werden:

- In Zeile 71 kann die I²C-Adresse des BME280-Moduls geändert werden.
- In Zeile 72 sollte die Höhe des Standortes in Metern über dem Meeresspiegel angegeben werden.
- In Zeile 92 kann der Befehl „#define _DEBUG_“ durch Entfernen der beiden // am Beginn der Zeile aktiviert werden. Dann werden viele zusätzliche Informationen über den Serial Monitor der Arduino IDE mit 115.200 Baud ausgegeben. Das ist sehr hilfreich, wenn etwas nicht funktioniert oder der Code erweitert oder geändert werden soll.

Bild 5: Bearbeitung des Zusatzgehäuses

Unter Tools wird das Board „Arduino Leonardo“ ausgewählt und danach das Programm auf den Arduino Pro Micro übertragen.

War das erfolgreich, trennt man zuerst die USB-Schnittstelle zum Arduino und danach das technoline WL1030 von der Stromversorgung. Da im Gerät selbst nicht genug Platz ist, um neben dem Arduino Pro Micro noch das HM-MOD-EM-8Bit-Modul und das BME280-Modul unterzubringen, muss eine andere Lösung her. In der Materialliste [5] ist ein Gehäuse aufgeführt, das exakt zur Größe der Rückseite des technoline WL1030 passt. Dieses wird – wie im Bild 5 gezeigt – bearbeitet. Natürlich kann auch jedes andere passende Gehäuse verwendet werden.

Der rechte Befestigungsdorn wird mit einem scharfen Seitenschneider entfernt und es werden ein paar Löcher für den Lufteinlass für den BME280 gebohrt. In die Rückwand des technoline-WL1030-Gehäuses wird die Bohrung für die Schraube des verbleibenden Befestigungsdorns des Zusatzgehäuses so ausgemessen, dass das Zusatzgehäuse knapp unterhalb der Tasten positioniert ist. Ein bis zwei weitere 10-mm-Bohrungen zur Durchführung der Kabel werden ebenfalls benötigt (Bild 6).

Im nächsten Schritt werden die Kabel durch die Rückwand gesteckt und die Rückwand wird lose auf das technoline-WL1030-Gehäuse gelegt (Bild 7).

Das HM-MOD-EM-8Bit-Modul wird vor dem Einbau mit einem Netzgerät versorgt und wie in der Bedienungsanleitung beschrieben an die

Bild 6: Bearbeitung des technoline-WL1030-Gehäuses

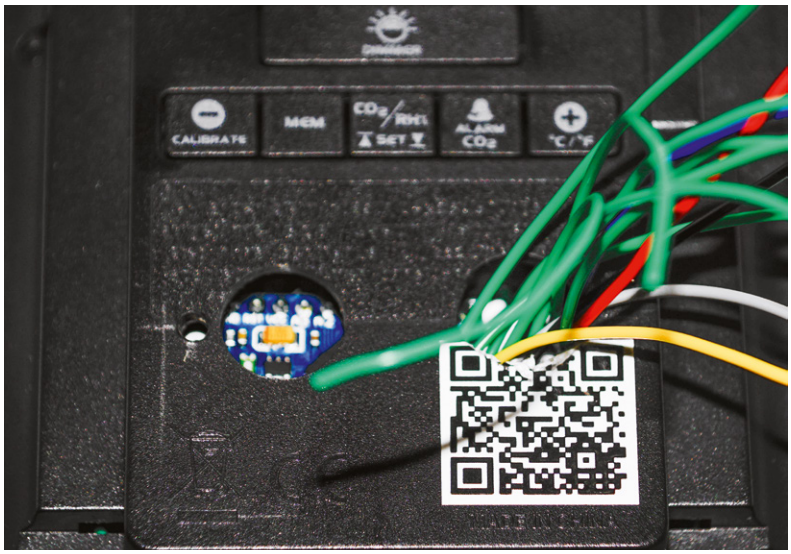

Bild 7: Durchführung der Kabel durch die Rückwand des technoline WL1030

Homematic Zentrale angelernt. Danach werden alle nicht benötigten Pins mit einem Seitenschneider abgeschnitten (sofern Sie bereits montiert wurden) und die verbleibenden Pins nach innen gebogen. Danach wird das Modul mit Heißkleber, wie in Bild 8 gezeigt, in das Gehäuse eingeklebt. Wichtig ist, darauf zu achten, dass der Taster des Moduls nicht gegen das Plastik drückt und damit betätigt wird.

Danach wird auch das BME280-Modul mit Heißkleber eingeklebt und die Verkabelung wird nach dem Schaltplan aus Bild 1 vervollständigt (s. a. Bild 9).

Ist auch das HM-MOD-EM-8Bit-Modul gemäß Bild 1 verkabelt, wird zuerst ein Funktionstest wie folgt durchgeführt, bevor das Gerät wieder verschraubt wird. Das technoline WL1030 wird mit dem mitgelieferten Netzteil mit Strom versorgt und der Arduino mittels USB-Kabel an den PC angeschlossen. Wurde Debug im Code aktiviert und ist im seriellen Monitor der Arduino IDE eine Baudrate von 115.200 Baud eingestellt, so sollten alle 30 Sekunden die Messwerte für CO_2 und Luftdruck erscheinen. Funktioniert dies, so wird – wie im Abschnitt Einbindung in Homematic beschrieben – ein Programm angelegt.

Werden die Werte auch korrekt in Homematic angezeigt, wird zuerst das Zusatzgehäuse von hinten an die Rückwand des technoline WL1030 angeschraubt, die Kabel im Inneren sorgfältig verstaut und das Gerät mit den sieben Schrauben wieder verschraubt. Die endgültige Modifikation ist sehr unauffällig und von vorne gar nicht sichtbar, wie Bild 10 zeigt.

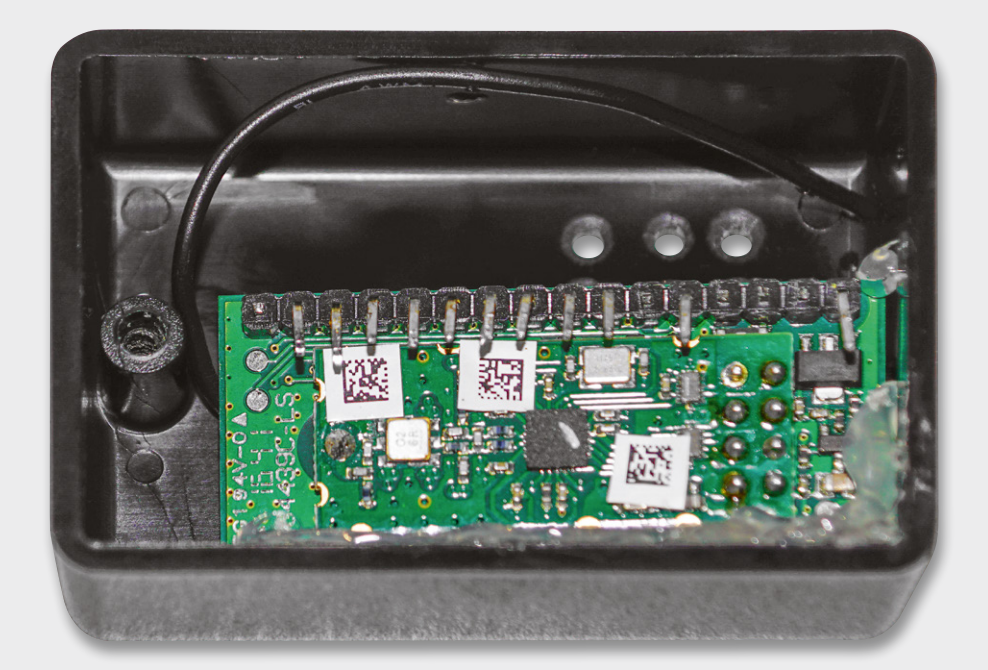

Bild 8: Vorbereitung und Einkleben des Moduls in das Zusatzgehäuse

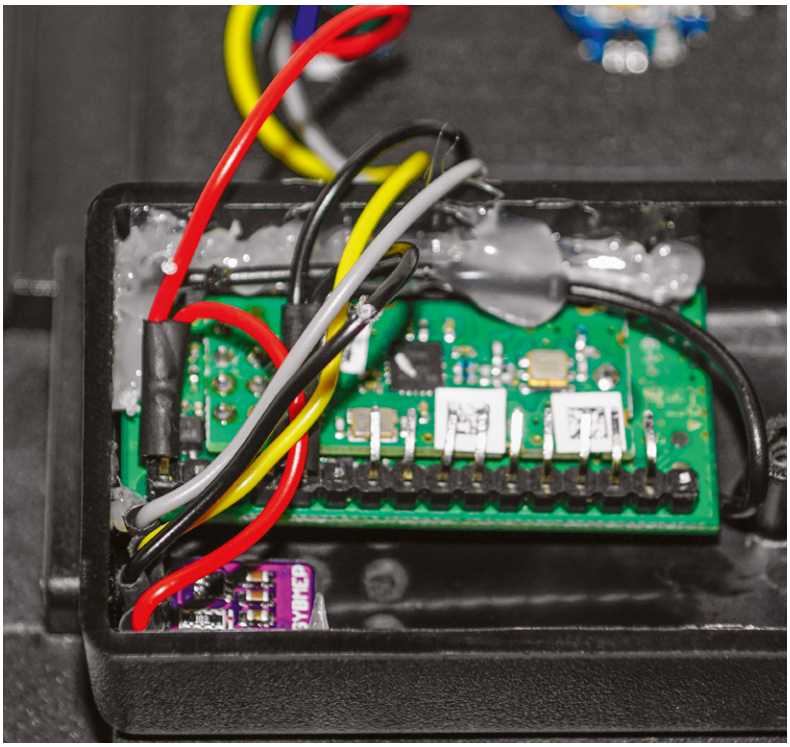

Bild 9: Verkabelung des BME280-Moduls

Bild 10: Vollständiger Zusammenbau des Gehäuses

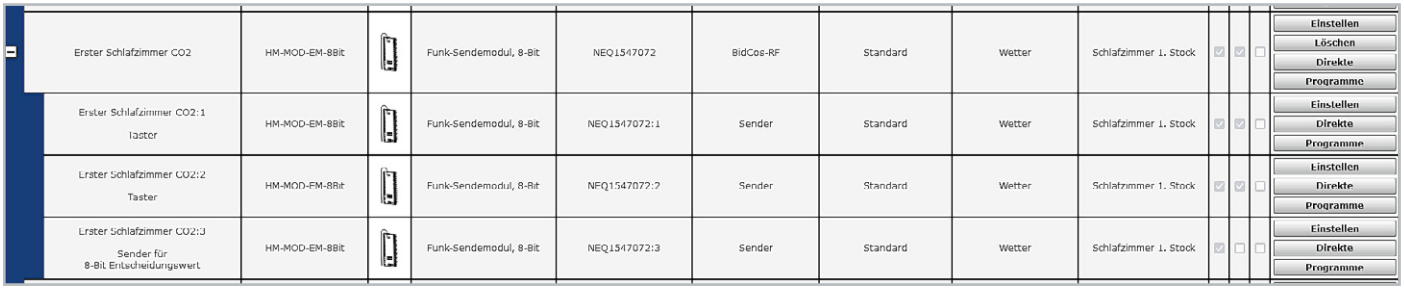

Bild 11: Vergabe des Namens für das Gerät HM-MOD-EM-8Bit in Homematic

Einbindung in Homematic

Bevor die Einbindung in Homematic dargestellt wird, möchte ich noch den Trick beschreiben, mit dem ich zwei Integerwerte über eine 8-Bit-Schnittstelle übertrage.

Die erste Annahme ist, dass sowohl der CO_2- als auch der Luftdruckwert nicht mehr als 12 Bit (0-4095) benötigen. Diese Werte werden in zwei 6-Bit-Werte geteilt. Die verbleibenden 2 Bit nutzt man als Kennung, was gerade übertragen wird. Die Tabelle 1 zeigt die Übertragung der vier Werte.

Das Programm im Arduino Pro Micro teilt die beiden ursprünglichen Werte in diese vier Werte und überträgt sie zeitversetzt über Homematic. Ein Skript in Homematic muss daher diese vier Werte wieder entsprechend zusammenbauen. Unter Einstellungen → Geräte in der WebUI der CCU3 sollte zuerst einmal ein eindeutiger Name für das Gerät HM-MOD-EM-8Bit und die Kanäle angegeben sein, in meinem Beispiel „Erster Schlafzimmer CO2" (Bild 11). Kanal 3 muss dazu wie in Bild 12 gezeigt konfiguriert sein.

Als Nächstes müssen drei Systemvariablen angelegt werden. Unter Einstellungen → Systemvariable wird zuerst einmal die Variable „Wetter-CO2-Schlaf" wie in Bild 13 gezeigt angelegt, die dann den CO_2-Wert enthalten wird.

Zwei weitere Systemvariablen mit den Namen „Wetter-Luftdruck-Schlaf" für den Luftdruckwert und eine temporäre Variable mit dem Namen „Wetter-CO2-Schlaf-Temp" werden mit den gleichen Einstellungen angelegt.

Danach wird ein neues Programm unter Programme und Verknüpfungen → Programme & Zentralenverknüpfung mit dem Namen „CO2 Sensor Schlafzimmer" wie in Bild 14 gezeigt angelegt.

Zuerst wird der Kanal 3 des Geräts „Erster Schlafzimmer CO2" ausgewählt, danach im Drop-down-Menü „Wert des Dateneingangs" eingestellt, danach im Wertebereich „Größer oder gleich 0" festgelegt und im letzten Drop-down-Menü „bei Aktualisierung auslösen" ausgewählt. Unter Aktivität „Skript" auswählen und das folgende Skript anlegen (siehe auch [5]):

Tabelle 1

Übertragung der CO_2- und Luftdruck-Werte

D7	D6	D5	D4	D3	D2	D1	D0	Beschreibung
0	0	X5	X4	X3	X2	X1	X0	Least Significant Bit (LSB) des CO_2-Werts
0	1	X11	X10	X9	X8	X7	X6	Most Significant Bit (MSB) des CO_2-Werts
1	0	Y5	Y4	Y3	Y2	Y1	Y0	LSB des Luftdruck-Wertes
1	1	Y11	Y10	Y9	Y8	Y7	Y6	MSB des Luftdruck-Wertes

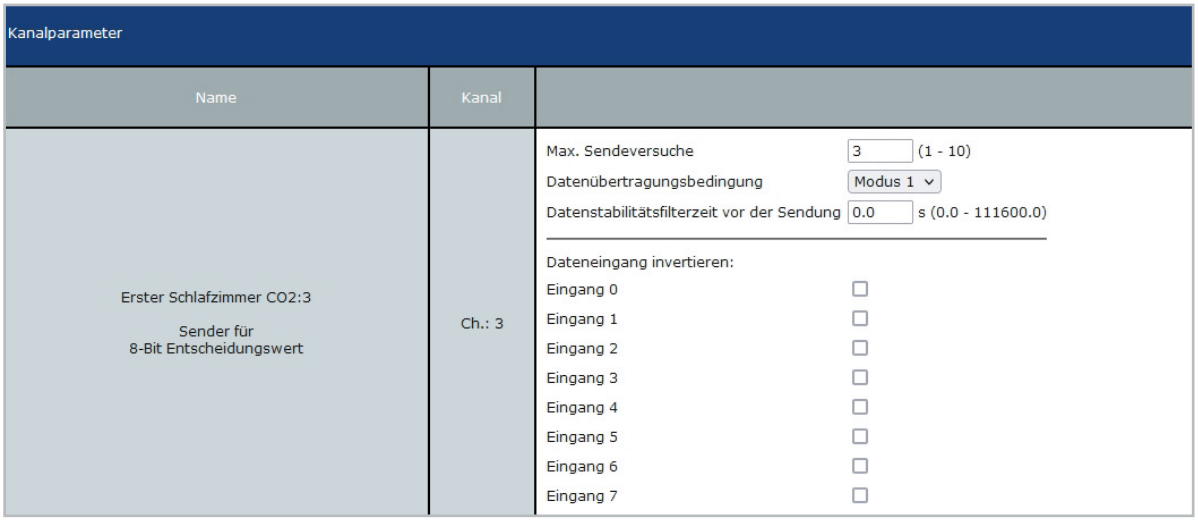

Bild 12: Die Konfiguration von Kanal 3

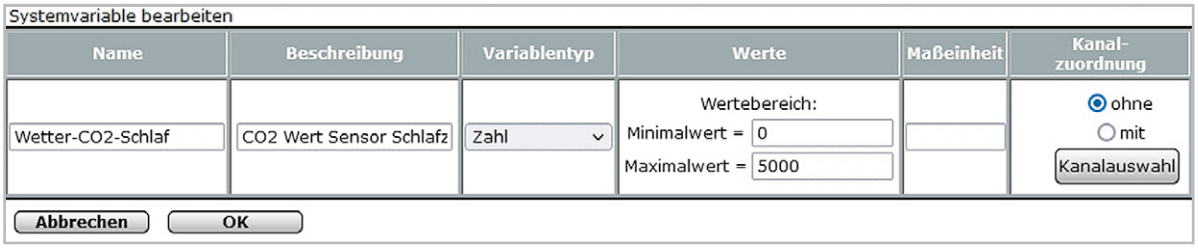

Bild 13: Anlegen einer Systemvariablen für den CO_2-Wert

```
! Skript to receive data from CO2 and air pressure Sensor via HM module
! Protokoll D7 D6 D5 D4 D3 D2 D1 D0
!                 0   0   x5 x4 x3 x2 x1 x0 -->  LSB of CO2 value
!                 x0-x5 = Lower 6 bit of CO2 value
!
!                 0   1   x11 x10 x9 x8 x7 x6 -->  MSB of CO2 value
!                 x6-x11 = Higher 6 bit of CO2 value
!
!                 1   0   x5 x4 x3 x2 x1 x0 -->  LSB of air pressure value
!                 x0-x5 = Lower 6 bit of air pressure value
!
!                 1   1   x11 x10 x9 x8 x7 x6 -->  MSB air pressure value
!                 x6-x11 = Higher 6 bit of air pressure value
!
! Name of System Variable CO2 value
var sysVarName _ 1="Wetter-CO2-Schlaf";
! Name of System Variable air pressure value
var sysVarName _ 2="Wetter-Luftdruck-Schlaf";
! Temporary variable to store LSB value
var sysVarName _ 3="Wetter-CO2-Schlaf-Temp";
!Name of Hm module
var devName = "Erster Schlafzimmer CO2:3";
! If exist Systemvariable 1 then sysExist _ 1 contains the name of the variable else it is NULL var sysExist _ 1= ⮐
dom.GetObject(ID _ SYSTEM _ VARIABLES).Get(sysVarName _ 1);
! If exist Systemvariable 2 then sysExist _ 2 contains the name of the variable else it is NULL var sysExist _ 2= ⮐
dom.GetObject(ID _ SYSTEM _ VARIABLES).Get(sysVarName _ 2);
! If exist Systemvariable 3 then sysExist _ 3 contains the name of the variable else it is NULL var sysExist _ 3= ⮐
dom.GetObject(ID _ SYSTEM _ VARIABLES).Get(sysVarName _ 3);
! If exist Devicename then devExist contains the name of the device else it is NULLvar devExist= ⮐
dom.GetObject(devName);
! Check is all three system variables and device exist?
if (sysExist _ 1 && sysExist _ 2 && sysExist _ 3 && devExist)
{
     ! Read state of channel 3
     var readValue=devExist.State();
     ! Check if D7 = 0 --> CO2 value
     var checkStat = readValue & 128;
     if (checkStat == 0)
    {
         ! Check if D6 = 0 --> LSB of CO2 value
         var checkLSB = readValue & 64;
         if (checkLSB == 0)
        {
             ! Process CO2 LSB value
             var CO2LSB = readValue & 63;
             ! Store LSB in temporary variable
             sysExist _ 3.State(CO2LSB);
        }
        else
        {
             ! Process CO2 MSB value
             var CO2MSB = readValue & 63;
             ! Shift 6 bits left
             CO2MSB = CO2MSB * 64;
             !Read LSB
             var CO2TMP = sysExist _ 3.Value();
             !Generate full CO2 value
             CO2MSB = CO2MSB + CO2TMP;
             !Store CO2 value
             sysExist _ 1.State(CO2MSB);
        }
    }
    else
    {
         ! Check if D6 = 0 --> LSB of air pressure value
         var checkLSB = readValue & 64;
         if (checkLSB == 0)
        {
            ! Process air pressure LSB value
            var AIRLSB = readValue & 63;
            ! Store LSB in temporary variable
            sysExist _ 3.State(AIRLSB);
         }
```

```
            else
            {
               ! Process AIR MSB value
               var AIRMSB = readValue & 63;
               ! Shift 6 bits left
               AIRMSB = AIRMSB * 64;
               !Read LSB
               var AIRTMP = sysExist _ 3.Value();
               !Generate full air pressure value
               AIRMSB = AIRMSB + AIRTMP;
               !Store air pressure value
               sysExist _ 2.State(AIRMSB);
            }
         }
      }
```

Wichtig ist, dass die Definition der Systemvariablen unter sysVarName_1, sysVarName_2 und sysVarName_3 sowie der Name des Gerätes unter devName in Zeile 16–22 mit den angelegten Namen exakt übereinstimmen.

Nachdem das Skript überprüft hat, ob die Variablen und das Gerät existieren, wird unter Zuhilfenahme der temporären Systemvariablen anhand der oberen 2 Bits der CO_2- und der Luftdruckwert – wie weiter oben beschrieben – zu einem CO_2- und einem Luftdruckwert zusammengesetzt und in der jeweiligen Systemvariable abgespeichert.

Unter Status und Bedienung → Systemvariable sollten die drei Variablen ca. alle 5 Minuten aktualisiert werden.

Arduino-Firmware

Das Arduino-Programm wurde in der Arduino IDE 1.8.12 erstellt und besteht aus dem Hauptprogramm WL1030_Hack.ino und den Reitern „Basis" und „Sensor". WL1030_Hack.ino besteht aus der bei Arduino üblichen Setup- und Loop-Routine.

In der Setup-Routine wird zuerst einmal die serielle Schnittstelle für das Debug-Interface über USB initialisiert. Danach werden die Ein- und Ausgänge für den Anschluss des HM-MOD-EM-8Bit-Moduls initialisiert.

In weiteren Schritten werden die serielle Schnittstelle 1 für den Anschluss des Cubic CO_2-Sensors eingerichtet und die Schnittstelle für den BME280-Sensor initialisiert.

Anschließend wird überprüft, ob ein Sensor unter der voreingestellten I^2C-Adresse vorhanden ist. In der Hauptroutine wird zuerst einmal auf den Empfang eines CO_2-Wertes gewartet. Alle ca. 5 Minuten (änderbar in Variable TRANSMIT_COUNT) werden die Werte über Homematic übertragen. Wenn der BME280-Sensor existiert, erfolgt der gleiche Prozess für den Luftdruckwert.

Unter „Basis" sind Basisroutine für die Übertragung über das HM-MOD-EM-8Bit-Modul zusammengefasst. Die Subroutine SUB_SEND_HOMEMATIC übernimmt die Aufteilung der Integerwerte für CO_2 und Luftdruck in vier 8-Bit-Werte wie weiter oben beschrieben.

Die Subroutine SUB_TRANS_HOMEMATIC wickelt die Übertragung über das HM-MOD-EM-8Bit-Modul ab. Unter „Sensor" sind die Routinen für die Kommunikation mit dem CO_2- und dem BME280-Sensor zusammengefasst.

In der Subroutine SUB_SENSOR_CO2_VALUE wird zuerst einmal 40 Sekunden auf den Empfang des ersten Werts gewartet. Danach wird versucht, 8 Datenbytes zu empfangen, die in einem Array gespeichert werden. Sind die Werte wie erwartet und passt die Checksumme, wird der CO_2-Wert in der Variablen REGISTER_VALUE gespeichert.

In der Subroutine SUB_SENSOR_BME280_VALUE wird der BME280-Luftdruckwert über die Library ausgelesen und in der Variablen REGISTER_VALUE gespeichert. ELV

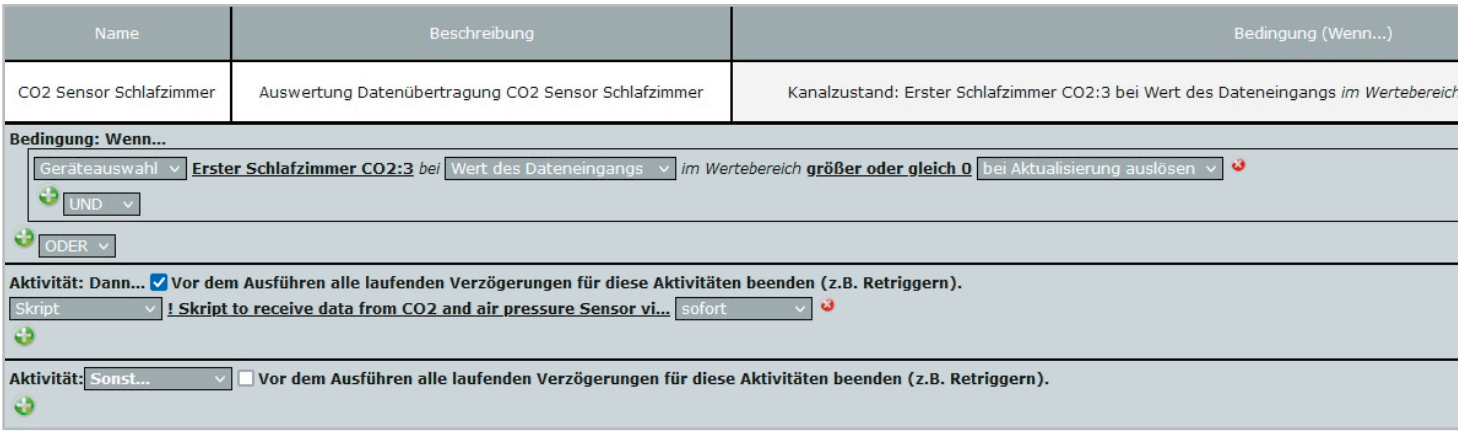

Bild 14: Anlegen eines Programms

Weitere Infos

[1] technoline CO2-Messgerät/CO2-Anzeige WL1030, Kohlendioxid, mit grafischer Ampelanzeige: Artikel-Nr. 251660
[2] Homematic IP CO2-Sensor HmIP-SCTH230: Artikel-Nr. 155645 (Bausatz), 155592 (Fertiggerät)
[3] ELV Modul HM-MOD-EM-8Bit: Artikel-Nr. 150253
[4] Cubic Sensor CM1106H-NS: https://en.gassensor.com.cn/CO2Sensor/info_itemid_88.html
[5] Materialliste und Code-Download: Artikel-Nr. 252596
[6] Adafruit Sensor Library: https://github.com/adafruit/Adafruit_Sensor

Alle Links finden Sie auch online unter: de.elv.com/elvjournal-links