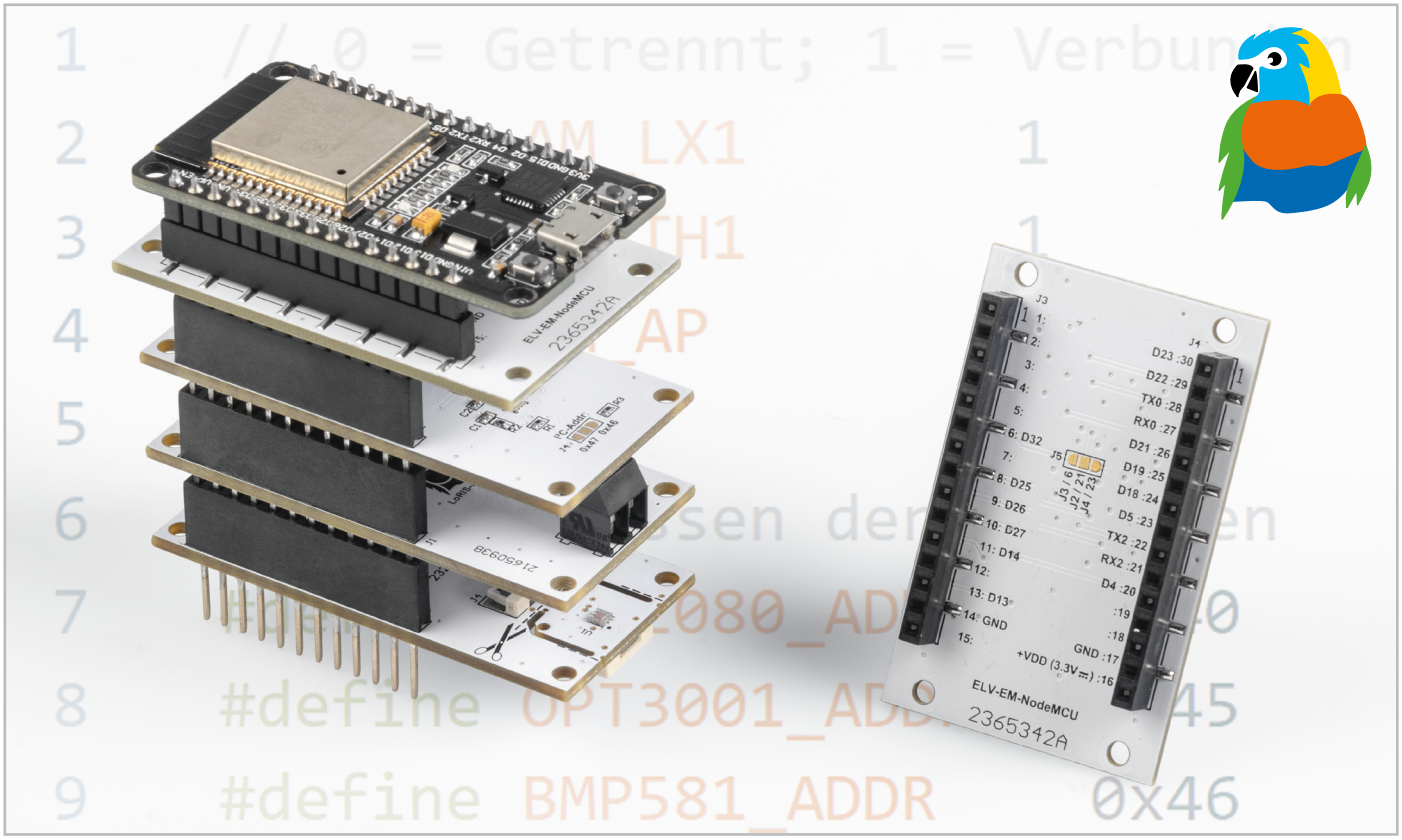

Smartes Prototyping

ELV Erweiterungsmodul NodeMCU - ELV-EM-NodeMCU

Das ELV-Modulsystem bietet eine große Vielfalt an unterschiedlichen Applikationsmodulen zur Erfassung diverser Messgrößen wie z. B. Temperatur, Luftfeuchte, Helligkeit, Position oder Beschleunigung. Mit dem ELV Erweiterungsmodul NodeMCU (ELV-EM-NodeMCU) ist es nun auch möglich, die Applikationsmodule in Kombination mit einem NodeMCU-Entwicklungsboard zu verwenden. In diesem Beitrag werden zunächst die Grundlagen der NodeMCU erläutert und das Schaltungsbild des Erweiterungsmoduls dargestellt. Anschließend wird die Programmierung in der Arduino IDE beschrieben anhand der in Bild 1 zu sehenden ELV Applikationsmodule Temperatur und Luftfeuchte (ELV-AM-TH1) [1], Luxmeter (ELV-AM-LX1) [2] und Luftdruck (ELV-AM-AP) [3]).

Mit einem Klick direkt zum Bausatz

ELV-EM-NodeMCU
Artikel-Nr.
160231
Bausatz-beschreibung und Preis:
www.elv.com

Infos zum Bausatz ELV-EM-NodeMCU

Schwierigkeitsgrad:
leicht

Bau-/Inbetriebnahmezeit:
ca. 0,25 h

Besondere Werkzeuge:
keine

Löterfahrung:
nein

Programmierkenntnisse:
ja (Arduino-Programmierung)

Elektrofachkraft:
nein

Was ist eine NodeMCU?

Der Begriff „NodeMCU“ beschreibt sowohl eine Hardwareplattform als auch eine Softwarebibliothek. Beide Komponenten basieren auf einem ESP8266- oder ESP32-Controller des chinesischen Unternehmens Espressif Systems, der sich vor allem durch einen günstigen Preis und ein integriertes WLAN-Modul auszeichnet [4]. Das Board ist daher ideal für die prototypische Umsetzung neuer Hard- und Firmwarekonzepte.

Hardwareseitig unterstützt der Mikrocontroller des NodeMCU-Entwicklungsboards ein breites Spektrum an Seriellen Schnittstellen wie I²C, SPI oder UART. Über GPIO-Pins (General Purpose Input Output) können Sensoren und Aktoren angeschlossen werden. Zu den weiteren Komponenten gehören eine USB-Schnittstelle zur Spannungsversorgung und Programmierung, ein UART-zu-USB-Wandler für die serielle Ausgabe von Sensordaten, ein Spannungsregler zur Regelung der Eingangsspannung auf 3,3 V sowie zwei Taster für das Aktivieren des Bootloader-Modus und das manuelle Zurücksetzen des Boards.

Firmwareseitig existieren durch eine große Community bereits viele Libraries für die Programmierung der NodeMCU. Am häufigsten erfolgt die Programmierung in Arduino, da die Syntax und Entwicklungsumgebung einen leichten Einstieg ermöglichen. Darüber hinaus werden

jedoch u. a. auch C/C++, Lua-Script oder Python unterstützt [5].

Das ELV Erweiterungsmodul NodeMCU wurde für die Joy-IT Entwicklungsplatine NodeMCU mit einem ESP-WROOM-32-Mikrocontroller entwickelt – generell ist sie jedoch auch mit allen anderen NodeMCU-Boards mit ESP32-Mikrocontroller kompatibel. Dieser Nachfolger des ESP8266 bietet neben dem WLAN-Funkmodul auch eine Bluetooth-LE-Schnittstelle und mehr GPIO-Pins [6].

Bild 2 zeigt exemplarisch die Draufsicht auf die Joy-IT NodeMCU [7]. Auf dem PCB (Printed Circuit Board) sind die zuvor genannten Komponenten farblich gekennzeichnet:

- gelb: ESP-WROOM-32 SoC (System on Chip) mit Tensilica-LX6-Dual-Core-Prozessor
- blau: UART-zu-USB-Wandler (Silicon Labs CP2102)
- grün: Spannungsregler (TI LM1117)
- grau: Boot und Enable Buttons
- orange: Micro-USB-Buchse (Spannungsversorgung und Übertragung der Firmware)

Schaltung

Die Schaltung des ELV Erweiterungsmoduls NodeMCU (Bild 3) verbindet alle relevanten Pins der Applikationsmodule des ELV-Modulsystems mit der NodeMCU. Das Erweiterungsmodul wird dabei über die Stiftleisten J1 und J2 auf die Applikationsmodule gesteckt, während die Buchsenleisten J3 und J4 für die Aufnahme der NodeMCU-Entwicklungsplatine vorgesehen sind.

Für die Spannungsversorgung der Applikationsmodule sind die Pins 14 (GND), 16 (VDD = 3,3 V) und 17 (GND) der Buchsenleisten J3 und J4 mit den gleichnamigen Pins der Stiftleisten J1 und J2 verbunden.

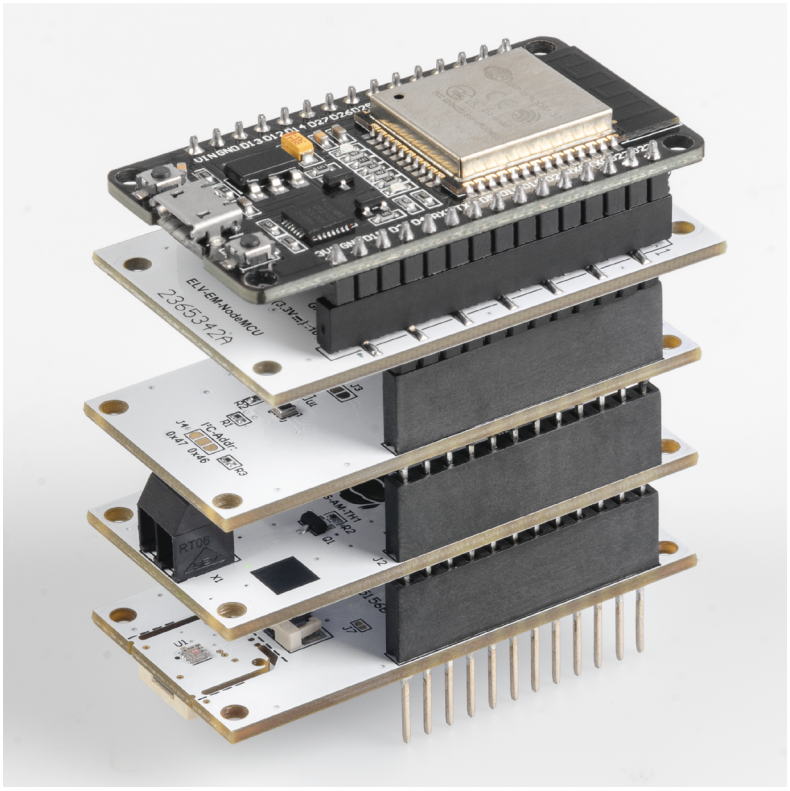

Bild 1: NodeMCU mit ELV Erweiterungsmodul NodeMCU und aufgesteckten Applikationsmodulen (ELV-AM-TH1, ELV-AM-AP, ELV-AM-LX1)

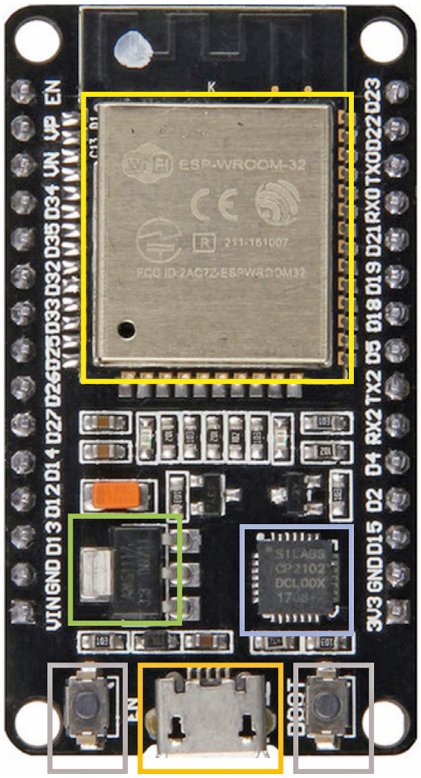

Bild 2: Draufsicht Joy-IT Entwicklungsplatine NodeMCU mit ESP32

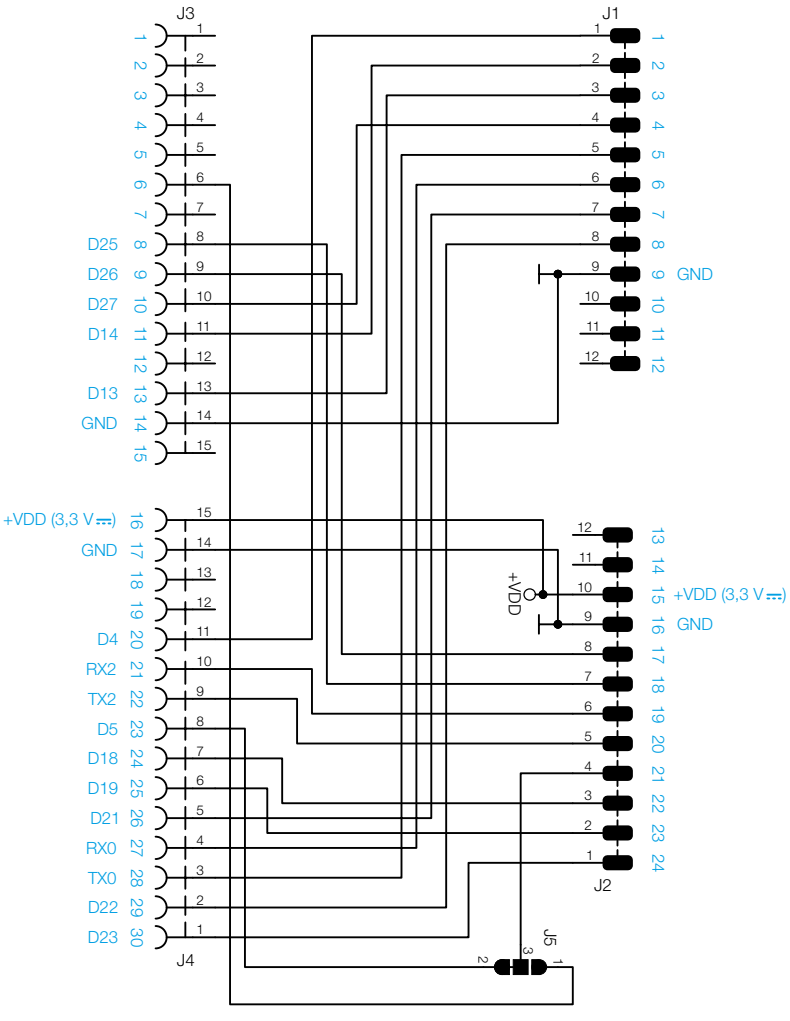

Bild 3: Schaltbild des ELV-EM-NodeMCU

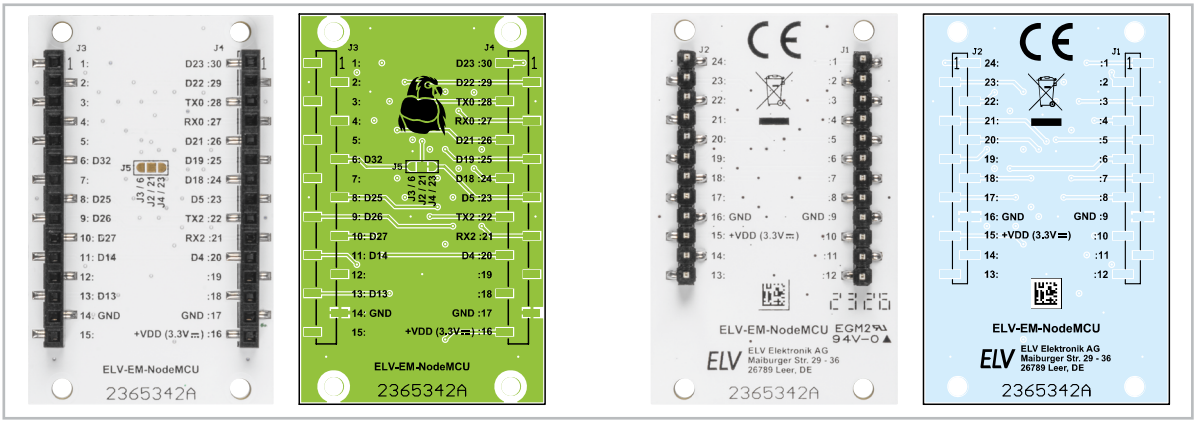

Bild 4: Platinenfotos und Bestückungsdrucke des ELV Erweiterungsmoduls NodeMCU

Hinweis
Eine gleichzeige Versorgung der NodeMCU und der angeschlossenen Applikationsmodule über USB und die VDD-Pins sollte vermieden werden, da es durch unterschiedliche Spannungen und dem daraus resultierenden Strom zu Beschädigungen der Schaltung kommen kann! Um solche Probleme zu vermeiden, ist es wichtig, dass die Stromversorgung eindeutig definiert ist und keine Mehrfachversorgung stattfindet.

Im Bereich der Seriellen Schnittstellen sind zwei UART-Schnittstellen nutzbar. Die Pins RX0 und TX0 dienen zur Übertragung von Firmware und Debug-Informationen. Falls das angeschlossene Applikationsmodul seine Daten über UART bereitstellt, geschieht dies über die Pins RX2 und TX2, die mit Pin 19 und 20 von J1 bzw. J2 verbunden sind.

Ebenfalls verfügbar ist ein SPI-Interface, bei dem die Pins D18 (SPI_SCK), D19 (SPI_MISO) und D23 (SPI_MOSI) auf die Pins 22, 23 und 24 von J2 geführt werden. Am Pin D5 der NodeMCU muss während des Startvorgangs ein High-Pegel anliegen. Sollte eine weitere angeschlossene Peripherie einen Low-Pegel an D5 erfordern, kann über den Jumper J5 alternativ der Pin D32 auf den SPI_CS Pin (Pin 21 an J2) geführt werden. So ist eine korrekte Funktionsweise des SPI-Interfaces sichergestellt. Die I2C-Pins D21 und D22 sind mit den Pins 7 (I2C_SDA) und 8 (I2C_SCL) verbunden.

Bild 5: Abschluss der Arduino-IDE-Installation

Zu den weiteren Verbindungen gehören die zwei ADC-Pins D25 und D26, die mit Pin 17 und Pin 18 verbunden sind. Der ADC-Pin D13 kann funktional zum Auslesen der Batteriespannung genutzt werden.

Nachbau

In Bild 4 sind die Platinenfotos und Bestückungsdrucke des ELV Erweiterungsmoduls NodeMCU zu sehen. Da alle Bauteile bereits bestückt sind, sind keine Lötarbeiten notwendig.

Installation und Vorbereitung der Entwicklungsumgebung

Hinweis: Die nachfolgenden Installationsanleitung bezieht sich auf die Arduino IDE in der Version 2.1.0 und das Betriebssystem Windows 11.

Falls die Arduino IDE noch nicht installiert ist, kann diese von der offiziellen Arduino-Webseite kostenlos heruntergeladen werden [8]. Durch das Befolgen der angezeigten Installationsschritte sollte am Ende der in Bild 5 zu sehende Dialog erscheinen.

Für die Verwendung der NodeMCU wird die offizielle ESP32-Bibliothek benötigt, in der die Boarddefinitionen enthalten sind. In der Arduino IDE kann der zusätzliche „Boards Manager" über das Menü File → Preferences hinzugefügt werden. In das Feld „Additional board manager URLs" wird die folgende URL eingetragen (Bild 6) [9]:

https://raw.githubusercontent.com/
espressif/arduino-esp32/gh-pages/ ⮐
package_esp32_index.json ⮐

Durch Betätigen des Buttons „OK" schließt sich der Dialog. Die IDE enthält nun eine weitere Quelle im Boards Manager. Zur Installation der Boards wird der Boards Manager aus der Seitenleiste ausgewählt. Dort kann nun durch den Suchbegriff „esp32" das entsprechende Package von Espressif Systems heruntergeladen werden (Bild 7).

Für die korrekte Nutzung der UART-zu-USB-Schnittstelle muss der CP210x-Treiber installiert werden. Dieser steht auf der Website von Silicon Labs zum Download bereit [10]. Die Ausführung der Datei „CP210xVCPInstaller_x64.exe" startet den Gerätetreiber-Installations-Assistent (Bild 8). Durch das Befolgen der Installationsanweisungen werden nun automatisch die benötigten Treiber installiert.

Nach dem allgemeinen Vorbereiten der Entwicklungsumgebung können nun die Bibliotheken für die Nutzung der Applikationsmodule eingebunden werden. Zur Nutzung des Applikationsmoduls Luxmeter

Stückliste

Sonstiges:

Stiftleiste, 1x 12-polig, gerade, SMD	J1, J2
Buchsenleiste, 1x 15-polig, gerade, SMD	J3, J4
Platine ELV-EM-NodeMCU	

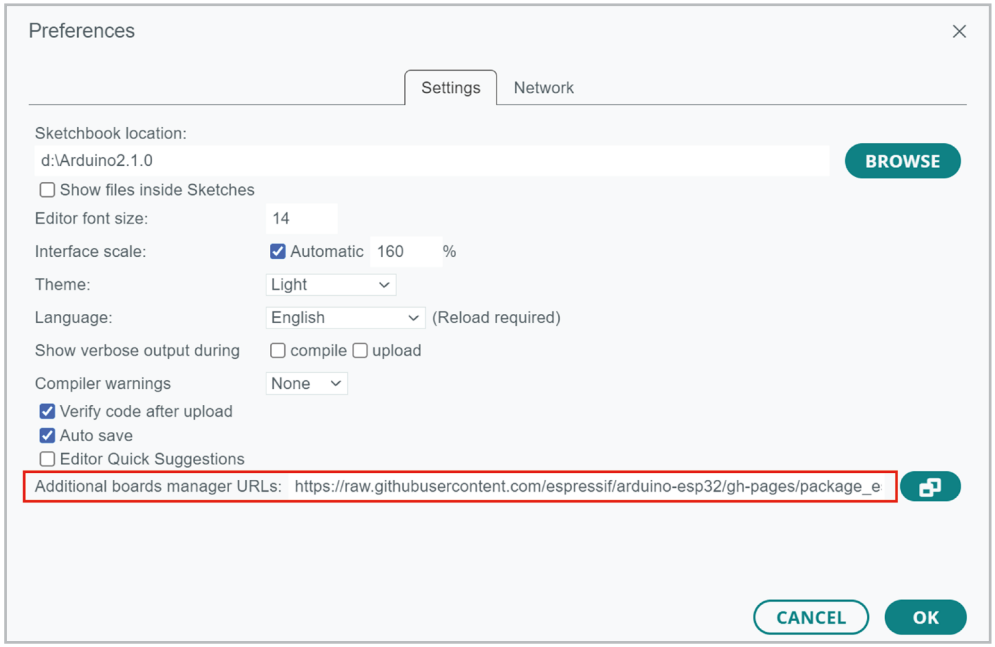

Bild 6: Hinzufügen einer neuen Boards-Manager-URL

steht im Downloadbereich des „ELV Bausatz Lichtsensor OPT3001 mit I²C-Schnittstelle I2C-LS“ [11] der Ordner „Software Arduino Beispielpaket“ zur Verfügung.

Nach dem Herunterladen befindet sich in dem Verzeichnis neben der I2C-LS-Bibliothek auch eine Readme-Datei, die verschoben bzw. gelöscht werden muss, da es sonst zu einer Fehlermeldung beim Importvorgang kommt.

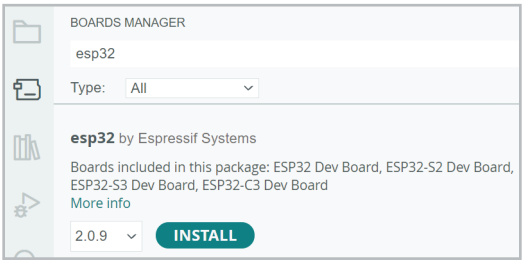

Bild 7: Installation des esp32-Package

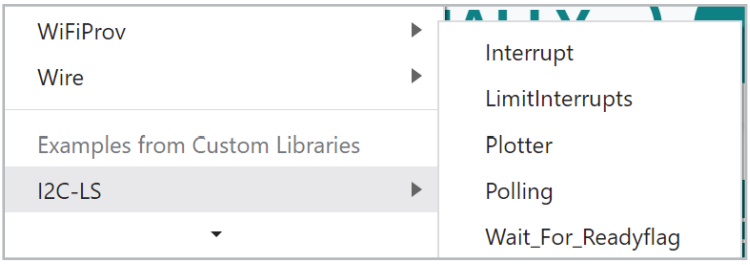

Bild 9: Code-Beispiele der I2C-LS Bibliothek

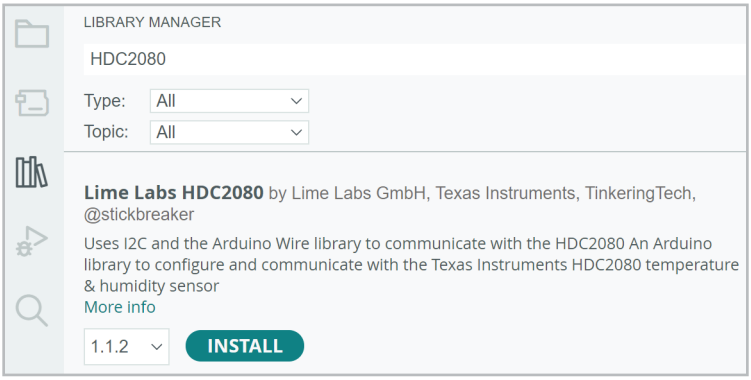

Bild 10: Installation der HDC2080-Bibliothek über den Library-Manager

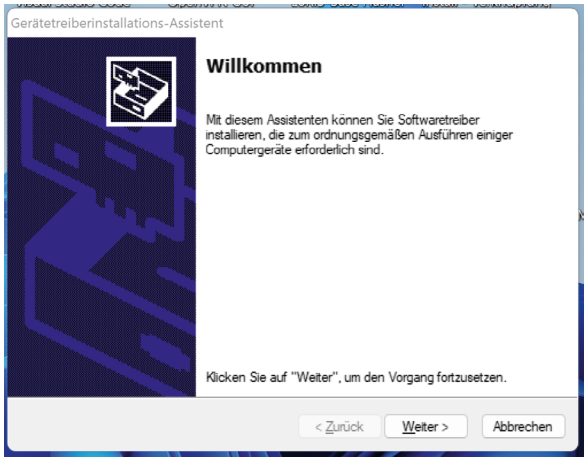

Bild 8: Installation des CP210x-Treibers

In der Arduino IDE wird nun aus der Menüleiste der Punkt Sketch → Include Library → Add .ZIP Library ausgewählt. Im Datei-Explorer wird der heruntergeladene Ordner „i2c-ls_arduino_lib.zip“ ausgewählt. Bei erfolgreicher Installation erscheint im Terminalbereich die Ausgabe „Library installed“. Außerdem stehen jetzt im Bereich File → Examples → I2C-LS mehrere Code-Beispiele zur Verfügung (Bild 9).

Die Bibliothek für den im ELV Applikationsmodul Temperatur und Luftfeuchte (ELV-AM-TH1) verbauten Sensor TI HDC2080 kann direkt aus der IDE heraus installiert werden. Dazu wird der Library-Manager aus der Seitenleiste geöffnet. Durch Eingabe des Sensornamens erscheint die Bibliothek „Lime Labs HDC2080“, die über den Button „Install“ hinzugefügt werden kann (Bild 10).

Auch die Bibliothek für den im ELV Applikationsmodul (ELV-AM-AP) verwendeten Drucksensor Bosch BMP581 kann über den Library-Manager heruntergeladen werden. Durch Eingabe der Sensorbezeichnung erscheint die „SparkFun BMP581 Arduino Library“. Diese wird analog zur HDC2080-Bibliothek installiert.

sketch_jul3a.ino

```
1   void setup() {
2     // put your setup code here, to run once:
3
4   }
5
6   void loop() {
7     // put your main code here, to run repeatedly:
8
9   }
10
```

Bild 11: Ansicht eines neuen Arduino-Sketches

```
1   // 0 = Getrennt; 1 = Verbunden
2   #define AM_LX1        1
3   #define AM_TH1        1
4   #define AM_AP         1
5
6   // I2C Adressen der Sensoren
7   #define HDC2080_ADDR    0x40
8   #define OPT3001_ADDR    0x45
9   #define BMP581_ADDR     0x46
```

Bild 12: Defines für die I2C-Adressen und die Angabe der angeschlossenen Sensoren

```
12  #if defined (AM_TH1) && (AM_TH1 == 1)
13  #include <HDC2080.h>
14  #endif
15  #if defined (AM_LX1) && (AM_LX1 == 1)
16  #include <I2C-LS.h>
17  #endif
18  #if defined (AM_AP) && (AM_AP == 1)
19  #include <SparkFun_BMP581_Arduino_Library.h>
20  #endif
```

Bild 13: Einbindung der benötigten Bibliotheken

```
27  #if defined (AM_TH1) && (AM_TH1 == 1)
28  HDC2080 hdc2080(HDC2080_ADDR);
29  #endif
30  #if defined (AM_AP) && (AM_AP == 1)
31  BMP581 bmp581;
32  #endif
```

Bild 14: Globale Variablen

```
37  #if defined (AM_LX1) && (AM_LX1 == 1)
38  void initOPT3001(){
39
40    // Initialisierung des OPT3001
41    I2C_LS.begin(OPT3001_ADDR);
42
43    // Initialisierung im Continouous Mode
44    I2C_LS.writeConfigReg( I2C_LS_CONFIG_CONT_FULL_800MS );
45  }
46  #endif
```

Bild 15: Initialisierung des Helligkeitssensors OPT3100

Beschreibung der Firmware

Für das Auslesen der Sensorwerte wird in der Arduino IDE unter File → New Sketch ein neuer Sketch angelegt. Dieser beinhaltet bereits die beiden Funktionen „setup" und „loop" (Bild 11). Innerhalb der setup-Funktion befindet sich später der Code zur Initialisierung der Seriellen Schnittstelle und der Sensoren (einmalige Ausführung nach Programmstart). Die loop-Funktion beinhaltet das zyklische Auslesen der Sensoren und die Ausgabe der Werte.

Vor dem Beginn der eigentlichen Implementierung werden Präprozessoranweisungen, auch Defines genannt, für eine bedingte Kompilierung des Codes definiert. Durch diesen Mechanismus verringert sich die Größe des späteren Maschinencodes, da nur die Bibliotheken, Funktionen und globalen Variablen der tatsächlich angeschlossenen Applikationsmodule geladen bzw. erstellt werden.

Ein weiterer Anwendungsfall von Präprozessoranweisungen ist das Anlegen von Bezeichnern, die dann im Code an der gewünschten Stelle eingefügt werden. In diesem Kontext werden so die I2C-Adressen der Sensoren definiert. Bild 12 zeigt alle erforderlichen Defines.

Die Einbindung der Bibliotheken ist über den Menüpunkt Sketch → Include Library möglich. Dort befindet sich am Ende der Liste der Bereich „Contributed Libraries", aus dem die Einträge „I2C-LS", „Lime Labs HDC2080" und „SparkFun BMP581 Arduino Library" ausgewählt werden. Unter Einbeziehung der zuvor definierten Präprozessoranweisungen ergibt sich der Code aus Bild 13.

Neben den Präprozessoranweisungen werden zwei globale Variablen für den HDC2080 und den BMP581 benötigt (Bild 14).

Da ein modularer Code leichter zu warten und zu erweitern ist, wird für jeden Sensor eine separate Initialisierungsfunktion implementiert.

Für die Initialisierung des Helligkeitssensors OPT3100 (Bild 15) wird das Objekt „I2C_LS" aus der zugehörigen Library genutzt. Die Funktion „begin" initialisiert die I2C-Peripherie und weist die übergebene Adresse zu. Der zweite Befehl startet die zyklische Messung der aktuellen Helligkeit in einem Intervall von 800 ms.

Die Initialisierung des Temperatur- und Luftfeuchtesensors HDC2080 (Bild 16) setzt sich aus drei Komponenten zusammen: Zunächst wird die Serielle Schnittstelle initialisiert und der Sensor zurückgesetzt (Zeile 52 und 53). Es folgt die Konfiguration des Messmodus, der Messfrequenz und der Auflösung der Messwerte (Zeile 56-59). Abschließend wird der Messvorgang gestartet (Zeile 62).

In der Initialisierungsfunktion des Drucksensor BMP581 (Bild 17) wird überprüft, ob die Verbindung zum Sensor erfolgreich hergestellt werden konnte (Zeile 71). Solange dies nicht der Fall ist, wird eine Fehlermeldung ausgegeben (Zeile 74). Die Ursachen für das Fehlschlagen können in einer falschen I²C-Adresse oder einem fehlerhaft angeschlossenen Applikationsmodul liegen. Erst wenn die Initialisierung erfolgreich verlief, wird die Erfolgsmeldung aus Zeile 80 ausgegeben.

```
48  #if defined (AM_TH1) && (AM_TH1 == 1)
49  void initHDC2080(){
50
51    // Initialisiserung des HDC2080
52    hdc2080.begin();
53    hdc2080.reset();
54
55    //Konfiguration des HDC2080
56    hdc2080.setMeasurementMode(TEMP_AND_HUMID);
57    hdc2080.setRate(TWO_HZ);
58    hdc2080.setTempRes(FOURTEEN_BIT);
59    hdc2080.setHumidRes(FOURTEEN_BIT);
60
61    //Beginn der Messungen
62    hdc2080.triggerMeasurement();
63
64  }
65  #endif
```

Bild 16: Initialisierung des Temperatur- und Luftfeuchtesensors HDC2080

```
67  #if defined (AM_AP) && (AM_AP == 1)
68  void initBMP581(){
69
70    // Initialisierung des Sensors
71    while(bmp581.beginI2C(BMP581_ADDR) != BMP5_OK){
72
73      //Keine Verbindung zum Sensor möglich
74      Serial.println("Error: BMP581 nicht verbunden!");
75
76      //Wartezeit zum nächsten Versuch
77      delay(1000);
78    }
79
80    Serial.println("BMP581 verbunden!");
81
82  }
83  #endif
```

Bild 17: Initialisierung des Drucksensors BMP581

In der „setup"-Funktion (Bild 18) wird zunächst die Serielle Schnittstelle (UART) mit einer Bitrate von 115200 Bit/s geöffnet (Zeile 87).

Anschließend folgen in Abhängigkeit der zuvor genannten Defines die Aufrufe der Initialisierungsfunktionen der angeschlossenen Applikationsmodule bzw. Sensoren (Zeile 90-100).

Die „loop"-Funktion wird automatisch nach der „setup"-Funktion aufgerufen. In ihr befinden sich sämtliche Ausgaben der Firmware, wie die Zeit in Millisekunden nach Programmstart und die formatierte Ausgabe der Sensorwerte. Wie bereits in der „setup"-Funktion werden dabei Präprozessoranweisungen genutzt, um einerseits die Größe des Codes zu reduzieren und andererseits die Ausgabe anzupassen (Bild 19, Bild 20).

Abschließend erfolgt die Ausgabe eines Zeilenumbruchs sowie der Aufruf der Funktion „delay" zur Verzögerung der nächsten Ausgabe um 800 ms (Bild 21).

```
84  void setup() {
85
86    // Initialisierung der seriellen Ausgabe
87    Serial.begin(115200);
88
89    // Initialisierung der Sensoren
90    #if defined (AM_LX1) && (AM_LX1 == 1)
91    initOPT3001();
92    #endif
93
94    #if defined (AM_TH1) && (AM_TH1 == 1)
95    initHDC2080();
96    #endif
97
98    #if defined (AM_AP) && (AM_AP == 1)
99    initBMP581();
100   #endif
101
102 }
```

Bild 18: Funktion „setup"

```
104 void loop() {
105
106   //Ausgabe des Zeitstempels in ms seit Beginn der Ausführung
107   Serial.print("Time (ms) :");
108   Serial.print(millis());
109
110   #if defined (AM_LX1) && (AM_LX1 == 1)
111
112   //Ausgabe des Helligkeitswerts des OPT3001
113   Serial.print("\tBrightness (lx): ");
114   Serial.print(I2C_LS.readLuxFloat(), 2);
115
116   #endif
117
118   #if defined (AM_TH1) && (AM_TH1 == 1)
119
120   //Ausgabe der Temperatur und Luftfeuchte des HDC2080
121   Serial.print("\tTemperature (C): ");
```

Bild 19: „loop"-Funktion (Teil 1)

```
122   Serial.print(hdc2080.readTemp());
123   Serial.print("\tHumidity (%): ");
124   Serial.print(hdc2080.readHumidity());
125
126   #endif
127
128   #if defined (AM_AP) and (AM_AP == 1)
129
130   //Auslesen der Sensordaten des BMP581
131   bmp5_sensor_data data = {0,0};
132   bmp581.getSensorData(&data);
133
134   //Ausgabe des Luftdurcks
135   Serial.print("\tPressure (Pa): ");
136   Serial.print(data.pressure);
137
138   #endif
```

Bild 20: „loop"-Funktion (Teil 2)

```
140   //Erzeugung eines Zeilenumbruchs nach Ausgabe der Werte
141   Serial.println();
142
143   //Abstand zwischen zwei Messungen (ms)
144   delay(800);
145 }
```

Bild 21: „loop"-Funktion (Teil 3)

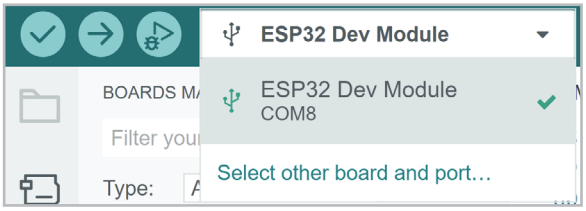

Bild 22: Auswahl des Boards und COM-Ports

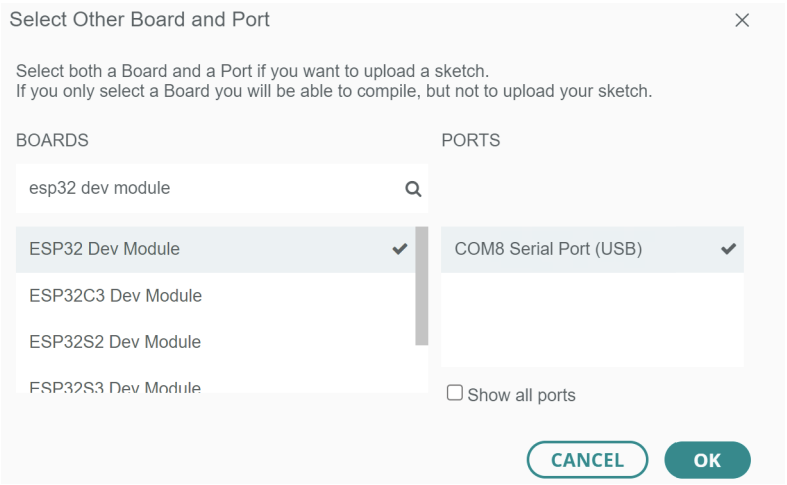

Bild 23: Auswahl des Boards und COM-Ports

Flashen der Firmware und Ansicht der Ausgabe

Für das Aufspielen der Firmware wird die NodeMCU mit den angeschlossenen Applikationsmodulen über ein Micro-USB-Kabel mit dem Computer verbunden. In der IDE muss dann „ESP32 Dev Module" und der Port ausgewählt werden, an dem die NodeMCU angeschlossen ist (Bild 22).

Falls an dieser Stelle noch nicht das richtige Board bzw. der richtige Port hinterlegt ist, kann dies über die Option „Select other board and port" angepasst werden. In dem sich öffnenden Dialog (Bild 23) erscheint das Bord nach Eingabe des entsprechenden Suchbegriffs. Auf der rechten Seite befindet sich zudem eine Übersicht aller verfügbaren COM-Ports.

Die Firmware kann nun auf die NodeMCU übertragen werden. Mithilfe des Buttons „Verify" wird der Code zunächst überprüft und anschließend kompiliert (Bild 24, rot). In der Output-Konsole sollte dabei keine Fehlermeldung erscheinen, sondern lediglich die Auslastung des Speichers (Bild 25). Sofern der Vorgang erfolgreich war, kann die Firmware nun in den Speicher der NodeMCU geladen werden. Dies geschieht über den „Upload"-Button (Bild 24, blau).

Die Ausgaben der NodeMCU können nun im Seriellen Monitor eingesehen werden. Dieser wird über das Menü Tools → Serial Monitor geöffnet und zeigt fortlaufend die gemessenen Sensorwerte an (Bild 26). Neben dem integrierten Seriellen Monitor eignen sich auch weitere Tools wie PuTTY oder HTerm zur Überwachung der Ausgaben [12], [13]. Damit der Zeitstempel bei 0 beginnt, kann der Reset-Button auf der linken Seite neben dem Micro-USB-Port der NodeMCU betätigt werden, um einen Neustart auszulösen.

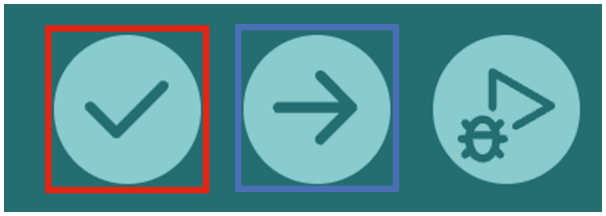

Bild 24: Buttons zum Verifizieren und Hochladen der Firmware auf die NodeMCU

Fazit und Ausblick

In diesem Artikel wurde das Erweiterungsmodul NodeMCU vorgestellt. Nach einer kompakten Definition des Begriffs „NodeMCU" und einer Beschreibung des Schaltbilds wurde auf die Firmware für die Verwendung der Applikationsmodule Temperatur und Luftfeuchte (ELV-AM-TH1), Luxmeter (ELV-AM-LX1) und Luftdruck (ELV-AM-AP) eingegangen. Insgesamt eröffnen sich durch die NodeMCU und das Erweiterungsmodul in Kombination mit den ELV-Applikationsmodulen neue Möglichkeiten zum Messen, Speichern und Weiterverarbeiten von Sensorwerten. ELV

Daten

Geräte-Kurzbezeichnung:	ELV-EM-NodeMCU
Umgebungstemperatur:	-10 bis + 55 °C
Abmessungen (B x H x T):	33 x 52 x 16 mm
Gewicht:	9 g

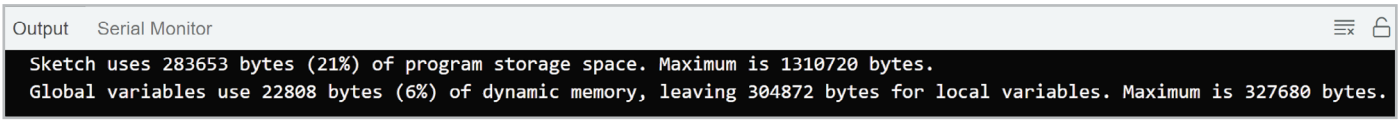

Abb. 25: Ausgabe bei erfolgreicher Überprüfung und Kompilierung des Codes

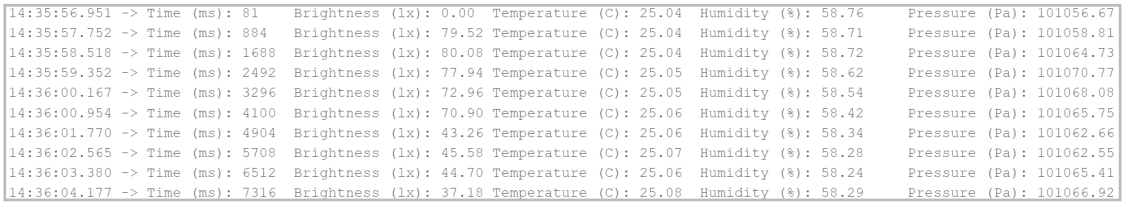

Bild 26: Ausgaben des Seriellen Monitors

Weitere Infos

[1] ELV-Temp-Hum1 Applikationsmodul Temperatur und Luftfeuchte, ELV-AM-TH1: Artikel-Nr. 157134
[2] ELV-LUX1 Applikationsmodul Luxmeter 1, ELV-AM-LX1: Artikel-Nr. 158467
[3] ELV-AirPressure Applikationsmodul Luftdruck, ELV-AM-AP: Artikel-Nr. 156996
[4] Espressif Systems SoCs: https://www.espressif.com/en/products/socs
[5] NodeMCU und ESP8266 – Einstig in die Programmierung: https://www.mikrocontroller-elektronik.de/nodemcu-esp8266-tutorial-wlan-board-arduino-ide/
[6] ESP-WROOM-32: https://www.bastelgarage.ch/esp32-esp-wroom-32
[7] Joy-IT Entwicklungsplatine NodeMCU mit ESP32: Artikel-Nr. 145164
[8] Download Arduino IDE: https://www.arduino.cc/en/software
[9] Installation der ESP-Bords in der Arduino IDE 2: https://www.youtube.com/watch?v=t4Nu5Qn0QkA
[10] CP210x-Treiber: https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers?tab=downloads
[11] ELV Bausatz Lichtsensor OPT3001 mit I²C-Schnittstelle I2C-LS: Artikel-Nr. 152106
[12] Download PuTTY: https://www.putty.org/
[13] Download HTerm: https://www.der-hammer.info/pages/terminal.html

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

ELV Leserwettbewerb

Teilen Sie Ihre Lieblingsprojekt

Das umfangreiche Angebot von ELV Haustechniksystemen, Produkten und Bausätzen bietet für viele Leser den Ausgangspunkt für eigene kreative Ideen. Haben auch Sie ein Projekt entwickelt, das andere Leser interessieren könnte?

Schreiben Sie uns, fotografieren Sie Ihr Projekt, berichten Sie von Ihren Erfahrungen und Lösungen. Teilen Sie Ihre fantasievolle Idee mit den Lesern des ELVjournals! Die interessantesten Anwendungen werden redaktionell bearbeitet und im ELVjournal mit Nennung des Namens vorgestellt.

Per E-Mail
leserwettbewerb@elv.com

Per Post
ELV Elektronik AG
ELVjournal Leserwettbewerb
26787 Leer

Alles, was nicht gegen Gesetze oder z. B. VDE-Vorschriften verstößt, ist für uns interessant. Die Auswahl der Veröffentlichungen wird allein durch die ELV Redaktion ausschließlich nach Originalität, praktischem Nutzen und realisierter bzw. dokumentierter Ausführung vorgenommen. Es besteht kein Anspruch auf Veröffentlichung, auch bei themengleichen Lösungen. Der Rechtsweg ist ausgeschlossen. Für Ansprüche Dritter, Beschädigung und Verlust der Einsendungen wird keine Haftung übernommen. Alle Rechte an Fotos, Unterlagen usw. müssen beim Einsender liegen. Die eingesandten Unterlagen und Aufnahmen verbleiben bei der ELV Elektronik AG und können von dieser für Veröffentlichungen und zu Werbezwecken genutzt werden.

* Der Einsender der veröffentlichten Anwendung erhält einen Gutscheincode zur einmaligen Nutzung im Wert von 200,– €. Der Gutscheincode wird mit einer Bestellung verrechnet – ein etwaiger Restbetrag verfällt. Bei Rückabwicklung des Kaufvertrags oder eines Teils hiervon wird der gewährte Gutscheinbetrag vom zu erstattenden Kaufpreis abgezogen, sofern durch die Ausübung des Widerrufsrechts und der Rückabwicklung der Gesamtwarenwert von 200,– € unterschritten wird. Auszahlung/Verrechnung mit offener Rechnung sowie Gutschrift nach Widerruf sind nicht möglich. Der Gutscheincode ist nicht mit anderen Aktionen kombinierbar.

Machen Sie mit!

Jedes veröffentlichte Projekt belohnen wir mit einem Gutscheincode im Wert von

200,– €*