Programmieren leicht gemacht

Einstieg in den ESP8266-Mikrocontroller mit ESPEasy

Die kompakten Wi-Fi-Module mit dem WLAN-Chip ESP8266/ESP32 erfreuen sich zunehmender Beliebtheit, eröffnen sie doch eine unendliche Anwendungsvielfalt für den einfachen Aufbau drahtloser Datennetze. Mit ihnen lassen sich schnell und unkompliziert Standalone-WLAN-Anwendungen installieren und beispielsweise entfernte Sensoren auswerten. Insbesondere die Möglichkeiten, die alternative Firmwarelösungen für diese kleinen Platinen bieten, entlasten den Anwender von langwieriger Programmierarbeit und ermöglichen das Realisieren schneller Lösungen. Eine dieser Alternativen ist die Firmware ESPEasy, die wir hier in der Hauptsache betrachten wollen.

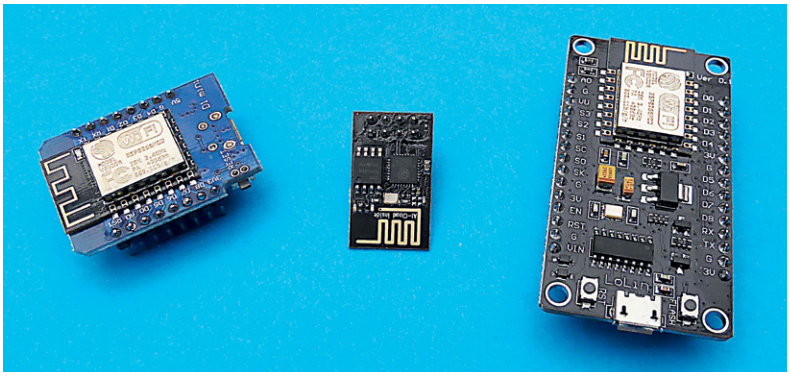

Bild 1: Einige gängige ESP8266-Module, links der Wemos D1 Mini, in der Mitte das weitverbreitete ESP-01-Modul ohne USB-Schnittstelle, rechts ein NodeMCU-Modul

ESP8266 – 32 Bit mit Wi-Fi

Mit der Entwicklung der ESP-Mikrocontroller hat der chinesische Hersteller Espressif Systems [1] einen großen Schritt gemacht, um drahtlose Datenkommunikation unkompliziert, massentauglich und preiswert zu gestalten. Rechtzeitig hat man erkannt, welche Anforderungen und Möglichkeiten im Internet der Dinge (IoT) liegen, und für tatsächlich jeden handhabbare Lösungen geschaffen. Mit einer enormen Produktvielfalt, basierend auf den Espressif-Mikrocontrollern, stellen diverse Hersteller kompakte Module für die Kommunikation per WLAN (802.11b/g/n, 2,4 GHz) sowie per WLAN und Bluetooth zur Verfügung.

Hauptsächlich unterscheidet sich die Programmpalette in zwei Kategorien. Das sind einmal die Single-Core-Mikrocontroller mit Wi-Fi des Typs ESP8266xx. Sie basieren in der aktuellen Version auf einem Tensilica-L106-32-Bit-RISC-Prozessor, der mit bis zu 160 MHz getaktet werden kann. Das integrierte Echtzeit-Betriebssystem (RTOS) und der ebenfalls integrierte Wi-Fi-Stack erlauben die Nutzung von mehr als 80 % der Prozessor-Ressourcen für Anwendungen. Der Prozessor zeichnet sich durch extrem geringe Stromaufnahme im Ruhe- und Bereitschaftszustand aus, was den Chip auch für lange, autarke Einsätze prädestiniert. Das SoC (System-on-a-Chip) integriert neben einem Speichercontroller auch SRAM und ROM, für Anwenderprogramme ist dem SoC auf den verfügbaren Modulen ein externer Flash-Speicher zugeordnet. Auf beliebten Boards wie dem NodeMCU oder den Wemos D1 sind beispielsweise die ESP-12-Module mit 4 Megabyte Flash verbaut. Bild 1 zeigt eine Auswahl verschiedener ESP8266-Module.

Eine besondere Variante ist der ESP8285, der einen internen Flash-Speicher von 1 Megabyte besitzt.

Betrachtet man das Blockschaltbild des SoC (Bild 2), erkennt man die Komplexität und die Schnittstellenvielfalt. Neben SDIO, PWM, IR-Fernsteuerport, UART, SPI und I²C stehen zahlreiche konfigurierbare GPIOs zur Verfügung, ebenso ein 10-Bit-ADC. Schließlich beherbergt der Chip die gesamte Wi-Fi-Hardware, also 2,4-GHz-Receiver/-Transmitter, die zugehörige Takterzeugung und eine gesonderte Spannungserzeugung. Der Chip beherrscht TCP/IP und das gesamte WLAN-MAC-Protokoll 802.11b/g/n (mit WPA/WPA2 PSK, IPv4/IPv6), es unterstützt STA- und AP-Betrieb. Die Firmware ab Werk ist für die Verarbeitung von AT-Kommandos [2] eingerichtet, die in der seriellen Ansteuerung via UART üblich sind.

Die zweite SoC-Linie ist die ESP32-Reihe, die mit Single- und Dual-Core-SoCs ausgestattet ist und neben Wi-Fi auch Bluetooth (unterstützt alle Classic-Bluetooth-Profile sowie BLE) an Bord hat. Prozessorkern ist eine Xtensa-Single-/Dual-Core LX6 CPU mit 32 Bit, bis 240 MHz Takt und 520 kB SRAM, 448 kB ROM, 16 kB SRAM sowie externer Flash-Verwaltung. Dazu kommen 34 GPIOs, ein 12-Bit-ADC mit 18 Kanälen, 2x 8-Bit-DAC, 10 Touch-Sensorkanäle, 4x SPI, 2x I²S, 2x I²C, 3x UART, 1x SD/eMMC/SDIO-Host, ETHERNET-Interface, CAN 2.0, IR, PWM und ein integrierter Hall-Sensor. Daran kann man erkennen, welche enormen Möglichkeiten dieser Chip bietet. In Bild 3 ist ein NodeMCU-Board mit diesem Chip zu sehen, das mit 4 MB Flash-Speicher bestückt ist.

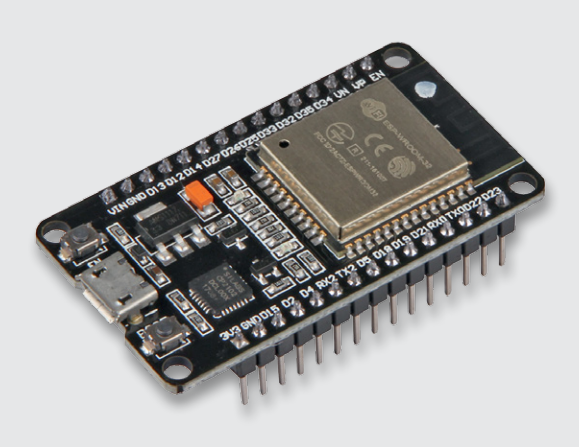

Bild 3: Noch vielseitiger – der ESP32, hier die JOY-iT-Entwicklungsplatine NodeMCU mit ESP32, verfügt neben zusätzlichem Bluetooth über eine Vielzahl an Schnittstellen und eine umfangreiche Speicherausstattung.

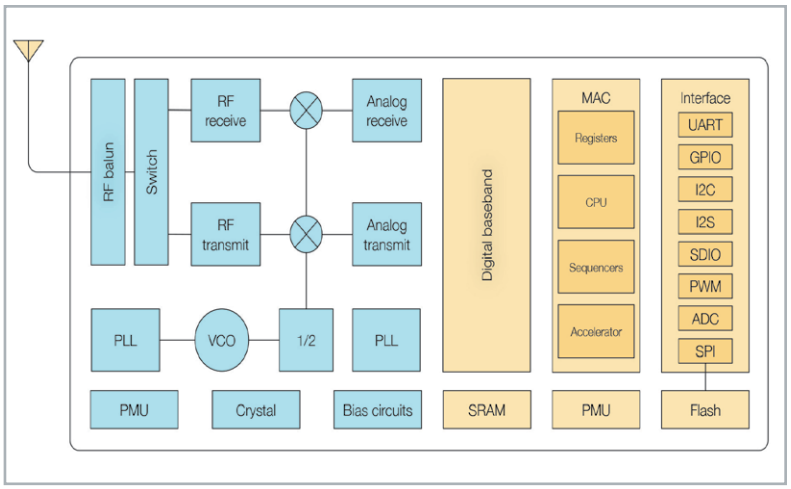

Bild 2: Das Blockschaltbild des ESP8266 zeigt das komplexe Innenleben des SoC.

Die Daten und die umfangreiche Ausstattung machen die ESP-SoCs enorm beliebt, schlagen sie doch in der Performance die allermeisten Arduino-Typen und sind genauso einfach einsetzbar.

Wie gesagt, ab Werk sind die Chips mit einem Bootloader und mit der ESP-AT-Firmware ausgestattet, zusätzlich werden einsatzspezifische Firmware-Versionen bereitgestellt, ebenfalls spezielle ESP-Entwicklungsumgebungen (SDK), u. a. eine auf Apple Home Kit spezialisierte SDK. In der User-Community wurden schnell weitere (und weit komfortablere) Firmware-Versionen kreiert, so die ArduinoCore-Version, die auf der Lua-Scriptsprache [3] basierende NodeMCU-Firmware oder eben ESPEasy [4], das wir hier im weiteren Verlauf näher betrachten wollen. Der fest programmierte Bootloader erlaubt das unkomplizierte Laden anderer Firmwareversionen.

Erste Bekanntschaft via Arduino IDE

Viele von uns nutzen seit Jahren die Arduino IDE, die sich entweder in einigen Abwandlungen selbst für STM32- oder ARM-Prozessoren findet oder zahlreiche Prozessorplattformen per Boardverwaltung integriert. So auch die ESP-Reihe mit ihren zahlreichen Variationen. Für die allererste Bekanntschaft mit dem ESP8266 eignet sich die Arduino IDE gut, denn man kann hier mit fertigen Beispielprogrammen Erfahrungen im Umgang mit dem Board sammeln.

Dazu sollte man die aktuellste Arduino-IDE-Version installieren, dann über „Datei"->„Voreinstellungen" die URL:

http://arduino.esp8266.com/stable/package_esp8266com_index.json

in die Zeile „Zusätzliche Boardverwalter-URLs" eintragen (Bild 4), mit OK abschließen und dann unter „Werkzeuge"->„Board"->„Boardverwalter" das Board-Paket für „esp8266 by ESP8266 Community" installieren (Bild 5).

Als Ergebnis findet sich bei der Board-Auswahl eine neue Abteilung namens „ESP8266 Boards". Unser Screenshot (Bild 6) zeigt links dazu die Auswahl des Wemos D1 Mini Boards, man sieht hier auch, dass man zahlreiche Möglichkeiten hat, unterschiedliche Board-Konfigurationen einzustellen. Je nach eingesetztem Board kann man sich auch die speziellen Beispielsketches beim Hersteller herunterladen, etwa via GitHub->„wemos". Viele der mit der IDE installierten Basics-Beispiele reichen aber für das erste Kennenlernen aus. Auch für den ESP32 gibt es die Möglichkeit, über die Arduino IDE programmiert zu werden, hier fügt man über den Boardverwalter die URL:

https://dl.espressif.com/dl/package_esp32_index.json

ein und erhält nach der Installation wieder eine neue Board-Auswahl.

Sinn des kurzen Ausflugs via Arduino IDE ist es auch zu lernen, wie man die ESP-Boards anschließt. Bei denen, die einen USB-Port an Board haben, gibt es keine Frage – einfach je nach Board den benötigten Treiber (CH340 oder CP2102) installieren, falls noch nicht vorhanden, USB-Kabel anschließen, Port suchen und in der IDE einstellen.

Anders verhält es sich bei den kleinen ESP-01-Versionen, die nur eine Stiftleiste haben. Hier setzt man spezielle TTL-USB-Adapter ein, die an die UART des ESP8266 andocken. Das können entweder eigens dafür entwickelte Programmer-Adapter sein oder vielfach aus der Mikrocontroller-Praxis bereits vorhandene TTL-USB-Adapter. Bild 7 zeigt eine Auswahl solcher Adapter.

In Bild 8 ist der Einsatz eines solchen USB-Adapters mit dem ESP-01-Modul nebst Anschlussbeschaltung des Moduls zu sehen. Wichtig ist hier, dass, um den ESP8266 beim Start in den Programmiermodus zu versetzen, die Pins „GND" und „GPIO0" auf Masse zu legen sind. Zusätzlich ist „EN" bzw. „CH_PD" (unterschiedlich bezeichnet) immer an die Betriebsspannung Vcc (3,3 V) zu legen, um den Prozessor nach Anlegen der Betriebsspannung zu starten. In die TxD-(GPIO1)/RxD-(GPIO3)-Leitungen kann man zur

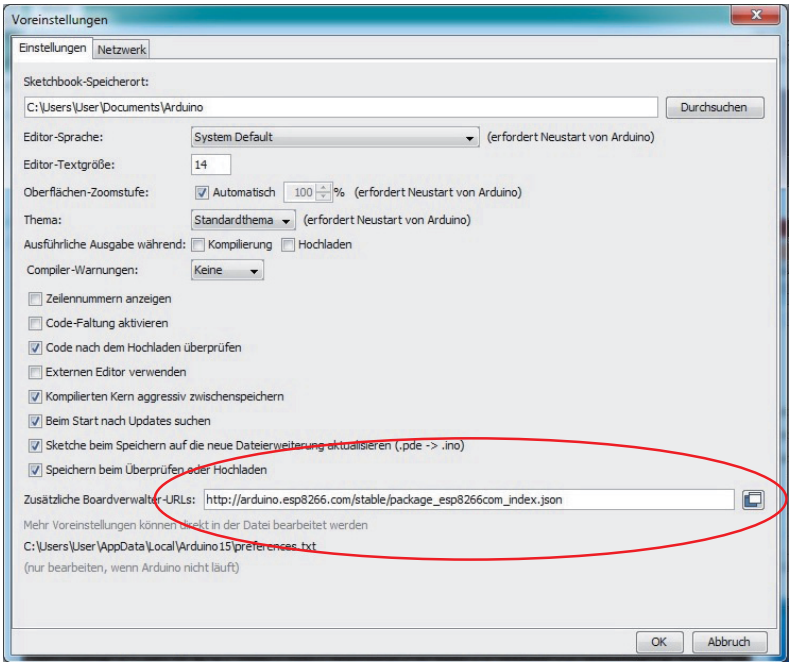

Bild 4: Hier wird per Boardverwaltung die Bezugsquelle für die ESP8266-Treiber eingetragen.

Bild 5: Die Boards der ESP8266-Community werden geladen ...

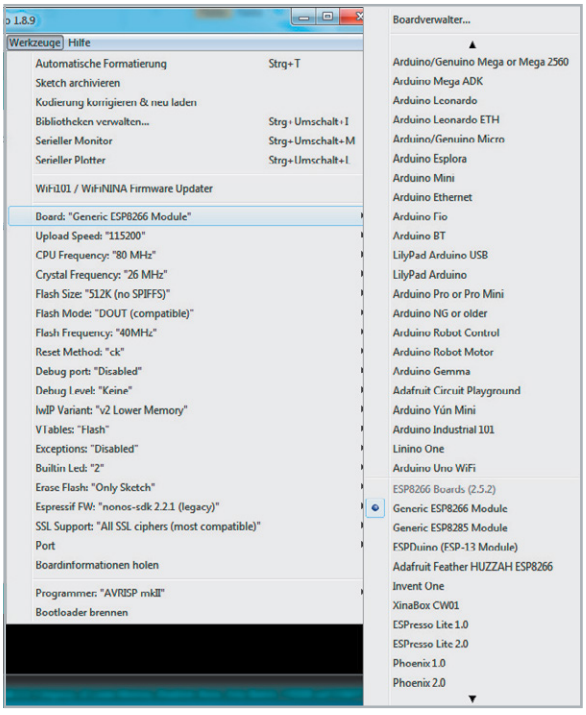

Bild 6: ... und stehen in einer eigenen Board-Abteilung der Arduino IDE zur Verfügung.

Sicherheit zusätzlich 1-kΩ-Widerstände einfügen. Nach dem Programmieren kann die Leitung von „GPIO0" entfernt werden, und nach einem Neustart arbeitet der ESP8266 im Normalbetrieb.

Hat man das Modul so per USB an den PC angeschlossen und hat die IDE das Modul am eingestellten Port erkannt, kann man einen ersten Test mit dem Beispielsketch „01Basics"->„Blink" ausführen. Nach dem Hochladen sollte nun (sofern vorhanden) die bordeigene LED im Sekundentakt blinken. Das soll uns genügen, Hauptzweck war die erfolgreiche Hardware-Anbindung zum Programmieren.

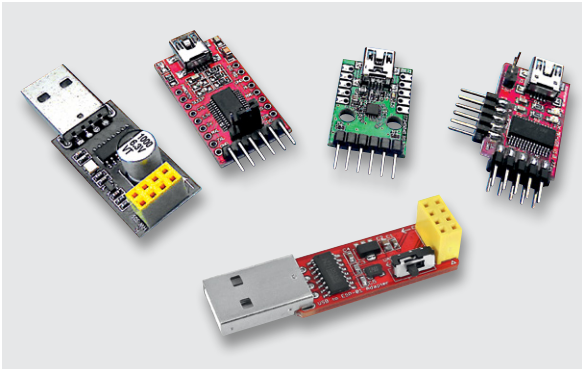

Bild 7: Für das Programmieren der ESP-Module stehen Adapter mit CH340 ebenso zur Verfügung wie solche mit CP2102. Links und unten zwei Adapter zum direkten Aufstecken des ESP-01.

ESPEasy – Konfigurieren ohne Ballast

Der Name sagt es, mit ESPEasy steht eine sehr einfach handhabbare Firmware zur Verfügung, die nicht einmal das Schreiben eigener Software (wie in der Arduino IDE, die mit C++ zu programmieren ist) erfordert. Sie enthält ein sofort nutzbares Web-Interface, über das man den ESP-Controller blitzschnell in das eigene Netzwerk einbinden, aber auch ein eigenes Netzwerk mit mehreren Nodes aufbauen kann. Zusätzlich sind zahlreiche Bibliotheken für Sensoren enthalten, so muss man diese nicht extra einbinden und programmieren.

Im Web-Interface sind eine ganze Reihe von Anwendungen bzw. Protokolle bereits betriebsfertig hinterlegt, so einige Smart Home Interfaces/Protokolle wie z. B. OpenHAB, MQTT, FHEM HTTP, PiDome HTTP oder die beliebte ThingSpeak-Anbindung zur Sensorauswertung. Die meisten Protokolle arbeiten mit HTTP-Request bzw. dem Machine-to-Machine-Protokoll MQTT.

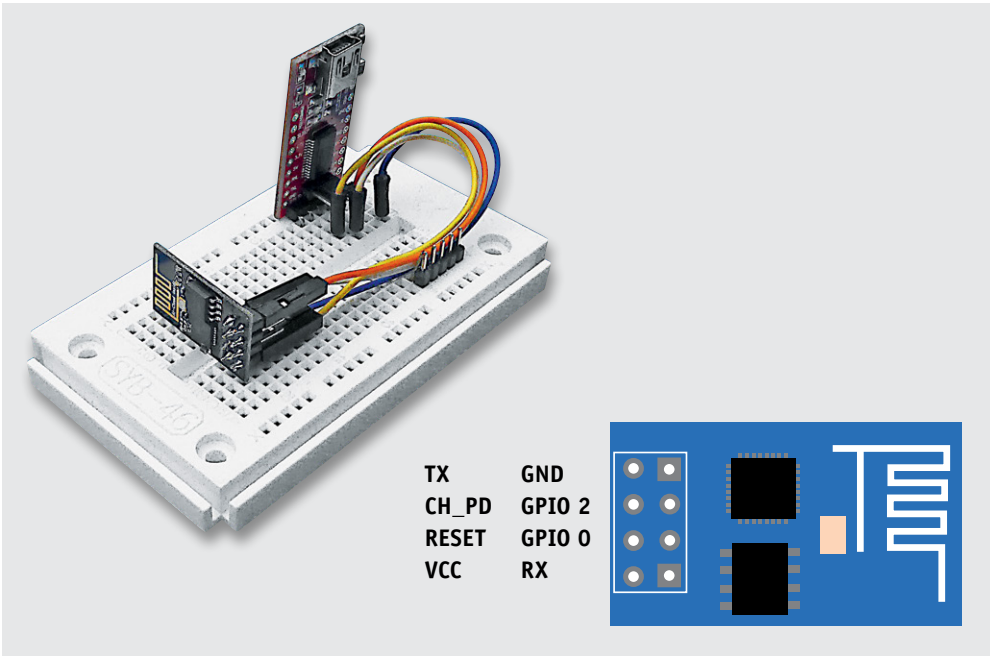

Bild 8: Der ESP-01 wird über einen USB-Adapter mit dem PC verbunden – die notwendigen Verbindungen für das Programmieren und Flashen von Firmware sind im Text erläutert.

Das Laden von ESPEasy als neue Firmware wollen wir im Beispiel unter dem Betriebssystem MS Windows betrachten, hier macht das im Programmpaket für ESPEasy enthaltene Tool „ESP-Flasher" das Flashen der Firmware sehr einfach. Für Linux- und macOS-Computer nutzt man die in deren System vorhandene Programmiersprache Python und das per Paketverwaltung nachladbare Tool „esptool.py". Das Linux-Script ist unter [4] verfügbar und beschrieben.

Zurück zur Windows-Installation. Wir verwenden einen Wemos D1 Mini mit 4 Megabyte Flash, an den wir später einen Klimasensor anschließen und auswerten wollen. Die Kenntnis der Flash-Speichergröße ist wichtig, sie wird beim Flashen benötigt.

Unter [5] laden wir zunächst die aktuelle Version von ESPEasy 2.0 („ESPEasy Mega") als zip-File herunter (zur Zeit der Artikelerstellung war dies die Version „ESPEasy_mega-20190607.zip").

Nach dem Entpacken sieht man zwei Tools, einmal das Flash-Tool „FlashESP8266.exe", das sehr übersichtlich ist, es erfordert nur die Eingabe des vom angeschlossenen ESP8266 belegten Ports und die Auswahl der zu ladenden Firmware. Für unseren Wemos Mini wählen wir die passende Datei mit der Bezeichnung:

„ESP_Easy-mega-20190607_normal_ESP8266_4M.bin"

Deutlich umfangreicher fällt die alternative Flash-Anwendung „ESP Easy Flasher" (Bild 9) aus. Um diese zu nutzen, muss man zunächst die im Ordner mitgelieferten Schriftfonts in den Windows-Fonts-Ordner kopieren.

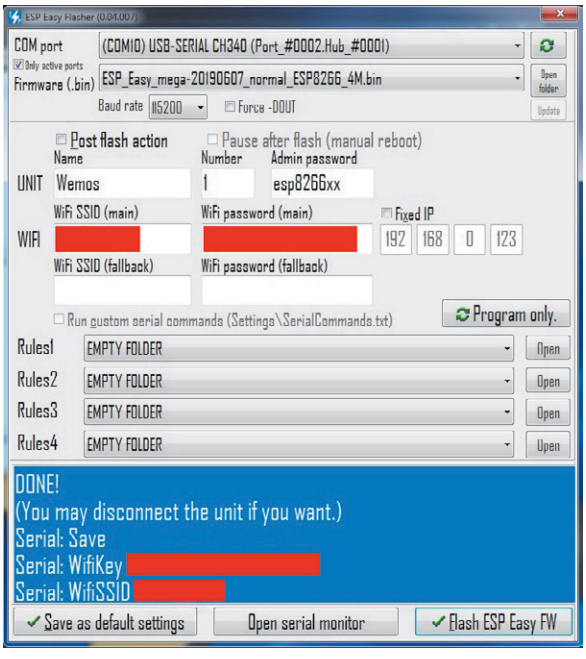

Bild 9: Der ESP Easy Flasher macht nicht nur das Flashen der Firmware einfach, er erledigt u. a. auch das Integrieren des ESP8266 in das eigene Netzwerk auf elegante Weise.

In den allermeisten Fällen ermittelt das Programm den belegten COM-Port selbstständig, ansonsten lässt man es noch einmal scannen. Unter „Firmware" wählt man die o. a. Firmware aus der Liste aus.

Der Komfort des Flashers liegt darin, dass man von hier aus sofort dem ESP8266 unmittelbar alle Netzwerkdaten für den späteren Betrieb im häuslichen Netzwerk (inklusive eines zweiten Netzwerks, falls das Hauptnetzwerk nicht erreichbar ist) vorgeben und über den Button „Program only" zum ESP8266 übermitteln kann, ohne den früher üblichen Umweg über den Aufbau eines ESP8266-eigenen Netzwerks und das folgende Einordnen in das häusliche Netzwerk über das WLAN-Interface des ESP8266 gehen zu müssen. Dieser Vorgang berührt zunächst die vorhandene Firmware nicht. Zusätzlich lassen sich hier serielle Kommandos (AT-Kommandos) an den Controller senden, zahlreiche dieser Kommandos dienen ja auch der Konfiguration des ESP8266 im AT-Betrieb.

Danach startet das Flashen mit „Flash ESP Easy FW". Dieser Vorgang nimmt ca. 90 Sekunden in Anspruch und wird im Erfolgsfall mit „Done!" beendet.

Startet man nun den ESP8266 neu (beim Wemos den Reset-Taster betätigen), kann man ihn unmittelbar im heimischen Netzwerk finden, z. B. über die WLAN-Abfrage der Fritz!Box (Bild 10) oder über Wi-Fi-Scanner. Hier sehen wir die IP-Adresse für unseren Wemos-Controller.

Diese rufen wir nun im Browser auf und gelangen nach Eingabe des allgemeinen Namens „admin" und des zuvor im Flasher festgelegten Passworts, hier „esp8266xx", in das Web-Interface, und zwar auf die Hauptseite (Bild 11). Hier sehen wir einige Systeminformationen, u.a. die Firmwareversion, die IP und die RSSI des Wemos-Controllers. Über „More Info" kann man sich sehr detailliert alle Daten und Systemzustände, Informationen zum ESP-Board, Speicher usw. ansehen. Sind mehrere Nodes in das Netzwerk integriert, werden deren Daten ebenfalls aufgelistet.

Im Config-Menü (Bild 12) hat man die Möglichkeit, alle Netzwerkdaten, Passwörter etc. nachträglich zu kontrollieren und zu editieren.

Unter dem Menüpunkt „Controllers" (Bild 13) trägt man nach Auswahl des Systems, das die erfassten Daten verarbeiten soll (wir haben hier einmal FHEM gewählt), alle Parameter ein, die erforderlich sind, um die Daten im ESP8266 abzuholen. Darauf gehen wir später noch detailliert ein.

Im Menü „Hardware" geht es an die Belegung und Auswertung der Pins des ESP8266-Controllers. Wir können u. a. die bordeigene LED zur Signalisierung des WLAN-Übertragungszustands (D4/GPIO2) heranziehen und für unsere geplante Sensoranwendung das I²C-Interface wie in Bild 14 gezeigt konfigurieren.

Unter „Devices" geht es nun an die Auswahl der angeschlossenen Sensoren bzw. Geräte (Bild 15). Die Liste zeigt eine beeindruckende Anzahl davon an, wir wählen hier den von uns geplanten Luftdruck- und Temperatursensor BMP280 an, den wir danach konfigurieren. Er bekommt einen Namen, z. B. den Standort, wird aktiv geschaltet, die standardmäßige I²C-Adresse wird angezeigt, wobei die Bemerkungen dazu beim Belegen des Sensors zu beachten sind,

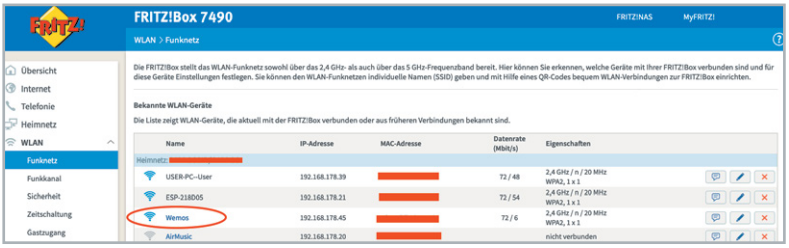

Bild 10: Wenn man dem ESP8266 nicht schon im ESP Easy Flasher eine feste IP zugeordnet hat, findet man die IP z. B. in der WLAN-Verwaltung des Routers.

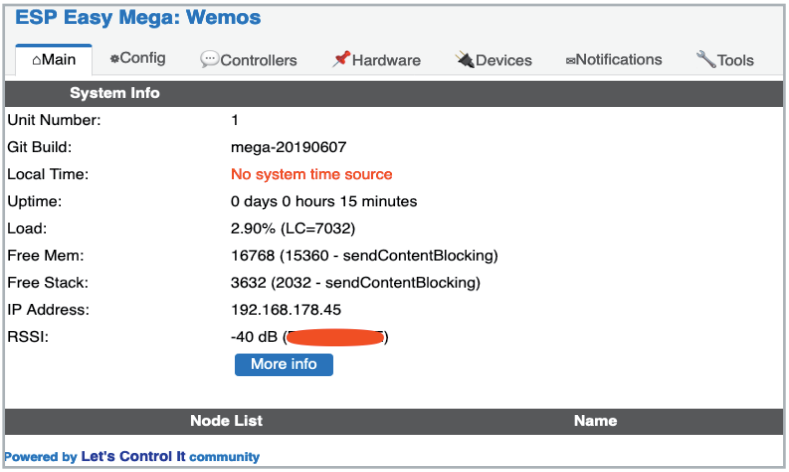

Bild 11: Nach dem Aufruf des ESP-Web-Servers erscheint das Hauptmenü, hier ist noch kein Zeitserver definiert, dieser kann wie im Text beschrieben aktiviert werden.

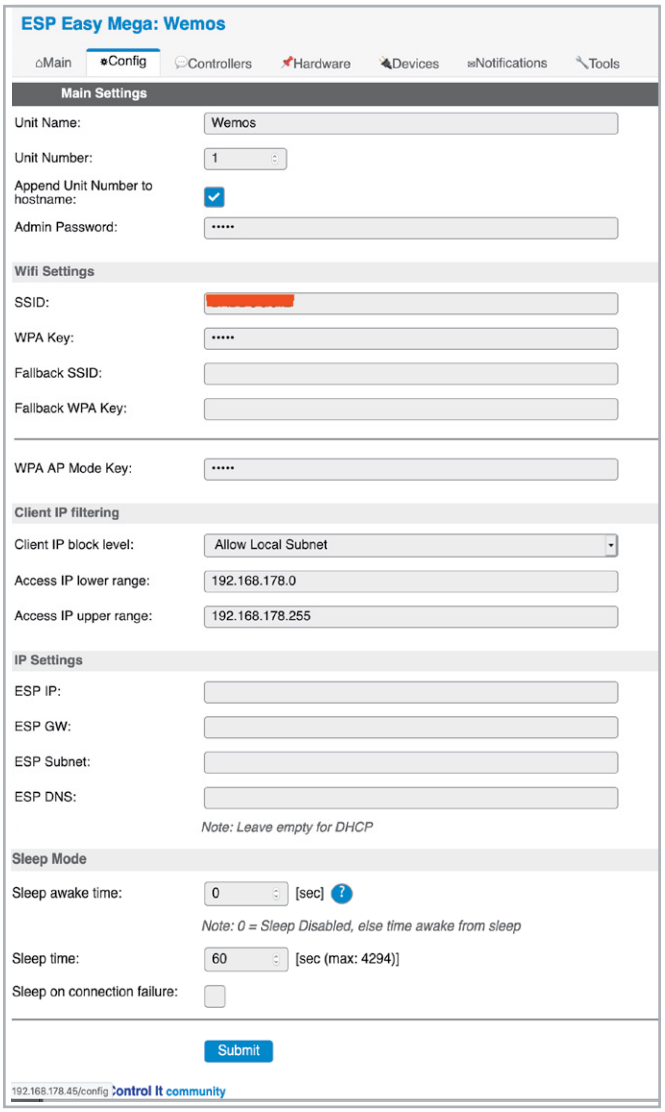

Bild 12: Das Konfigurations-Menü ermöglicht die Eingabe von Grundeinstellungen vor allem zur WLAN-Anbindung.

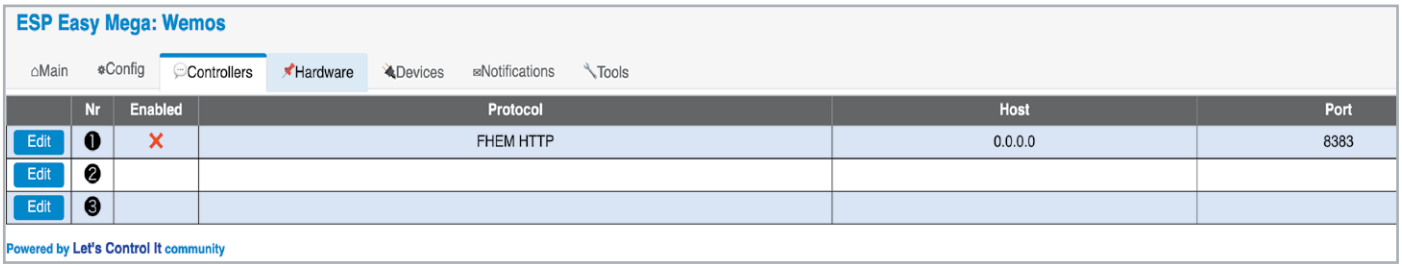

Bild 13: Im Controller-Menü konfigurieren wir den/die gewünschten Partner des WLAN-Controllers, hier FHEM.

man stellt die Einsatzhöhe und ggf. einen Temperaturoffset ein. Unter „Send to Controller" legt man fest, dass die Daten an den zuvor eingestellten Controller zu senden sind. Dem folgt das Sendeintervall, und schließlich erhalten die Anzeigefelder einen Namen und man muss ggf. eine Umrechnungsformel für die Anzeige eingeben, etwa, wenn man Grad-Celsius-Ausgabe in eine Grad-Fahrenheit-Anzeige umwandeln will. Der Link zur Wiki-Seite von ESPEasy führt zu zahlreichen bereits fertig berechneten Formeln. Nach „Submit" und „Close" erscheinen sofort die aktuellen Sensordaten (Bild 16).

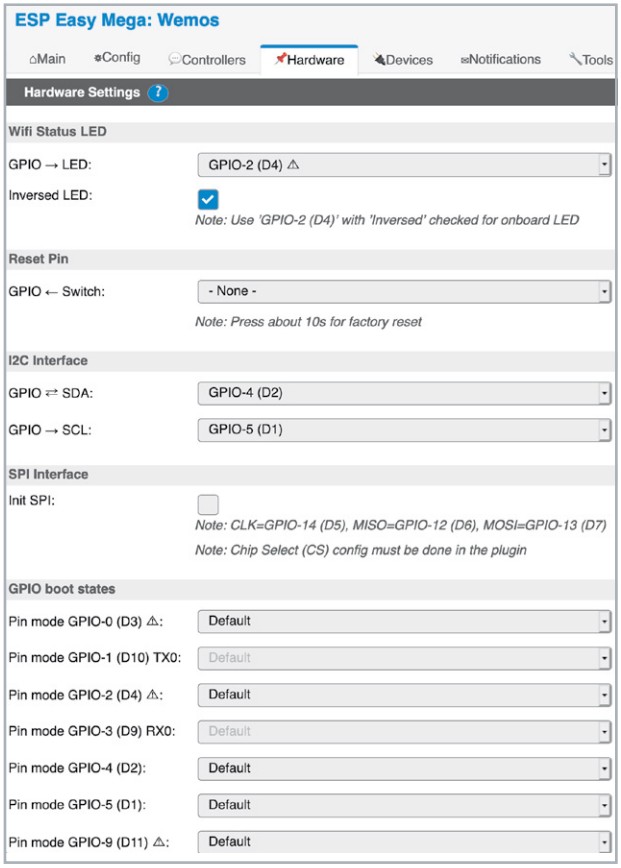

Bild 14: Das Hardwaremenü erlaubt die komfortable Zuordnung und Konfiguration der Pins des Controllers.

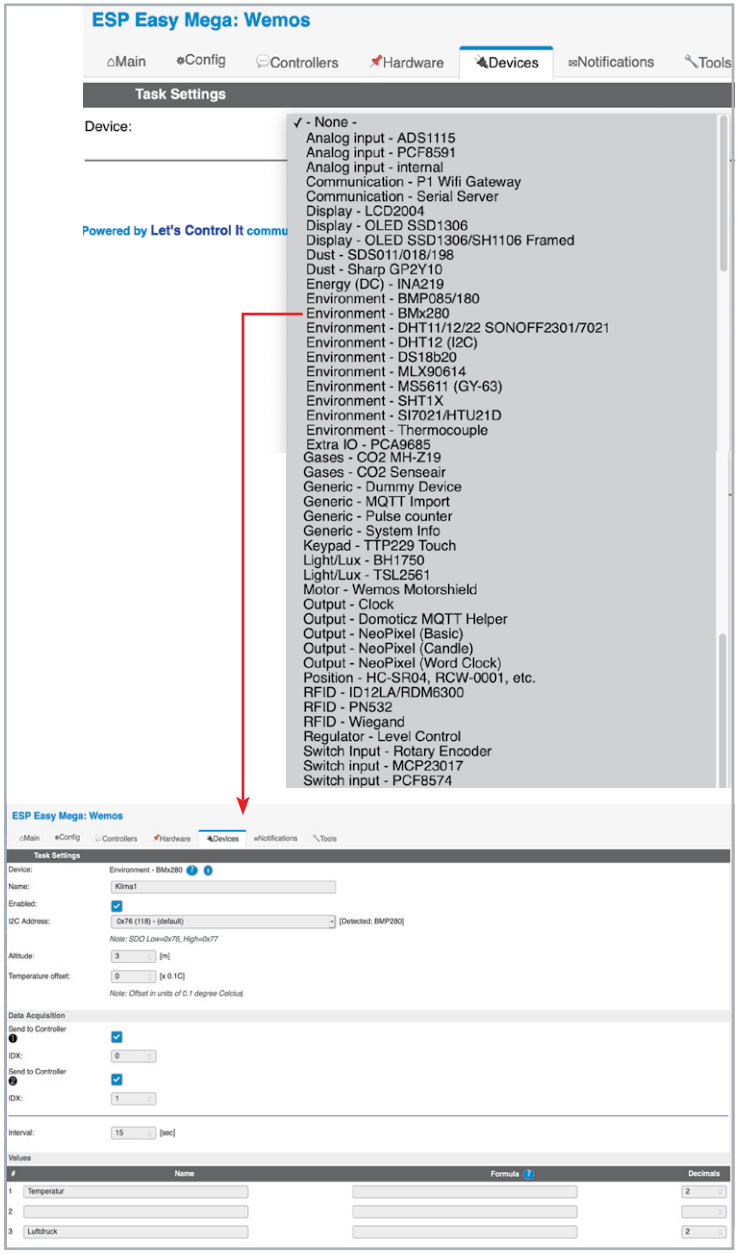

Bild 15: Das Sensor-/Gerätemenü hält eine große Anzahl an Sensoren, Displays und andere Peripherietechnik bereit.

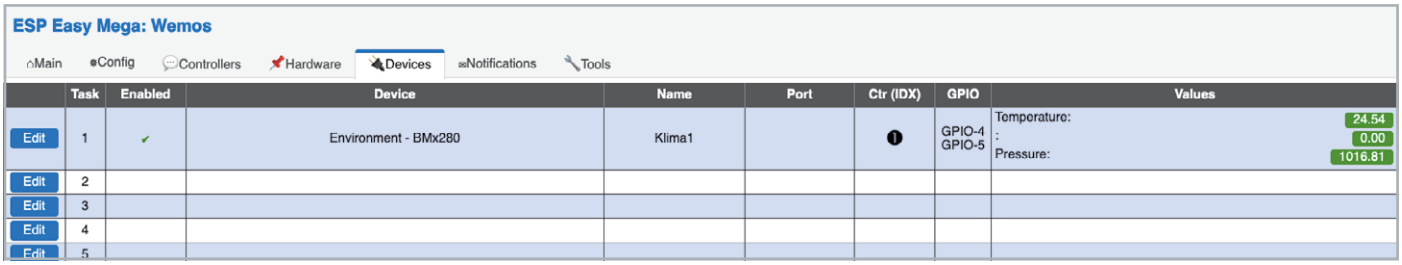

Bild 16: Nach der Konfiguration erscheinen sofort die Sensordaten.

Der Menüpunkt „Notifications" ermöglicht die Benachrichtigung beim Auslösen bestimmter Ereignisse, die durch den angesprochenen Controller auslösbar sind, so ein an den ESP8266 anschließbarer Signalgeber oder eine E-Mail-Benachrichtigung (Bild 17).

Im letzten Menüpunkt „Tools" stehen schließlich zahlreiche Möglichkeiten der Kontrolle von Systemzuständen, der Fernauslösung von Reboots, für das Trennen und Einloggen ins WLAN sowie des Downloads und der Ferninstallation von Software zur Verfügung. Hier kann der WLAN-Controller umfassend kontrolliert und fernkonfiguriert werden. Wir haben hier nur ein kleines Beispiel herausgegriffen, der ESP8266 kann per Firmware die I²C-Adressen und Typen der angeschlossenen Sensoren ermitteln (Bild 18). Unter „Advanced" kann man übrigens dem Controller auch eine Systemzeit via WLAN und Internet zukommen lassen, unter „NTP-Settings" und „DST-Settings" sind die Festlegung eines beliebigen NTP-Servers (z. B. 0.de.pool.ntp.org, siehe [6]) sowie weitere Zeiteinstellungen wie Zeitumstellungen, Zeitzonen etc. möglich. Der Zeitbezug per NTP-Server ist nach „Submit" sofort unter „Main" kontrollierbar.

Als Fazit des ersten Kennenlernens von ESPEasy können wir tatsächlich feststellen, dass man hier ohne Programmierkenntnisse sehr schnell recht komplexe Datenerfassungsaufgaben konfigurieren und die Übertragung per WLAN auf unkomplizierte Weise realisieren kann. Ab und an trifft man auf Beta- oder „not stable"-Hinweise, dazu muss man bedenken, dass das System von einer Community weiterentwickelt und gepflegt wird und Open Source ist. Die „letscontrolit"-Website ist äußerst umfassend gestaltet und hält unendlich viele Anleitungen, Tipps und Tricks, Einsatzbeispiele und jede erdenkliche Detailerläuterung bereit, komplette Anwendungen sind u. a. im zugehörigen GitHub zu finden. Eine der umfangreichsten privaten deutschen Seiten zum ESP8266 findet sich unter [7], hier sind auch unzählige konkrete Anwendungen zusammengefasst.

Noch ein Wort zum ESP32: die letscontrolit-Community arbeitet daran, auch für diese wegen ihrer Kommunikations- und Funktionsvielfalt extrem beliebten ESP-Boards eine ESPEasy-Mega-Plattform zu entwickeln, die sich jedoch derzeit noch im Entwicklungsstadium befindet. Wer sich dafür interessiert, kann unter [8] den Stand der Dinge verfolgen und bereits Experimente ausführen.

Was machen wir nun mit den gesammelten Daten (wobei der Controller natürlich auch etwa für Steuerungsaufgaben via WLAN einsetzbar ist (siehe [10])?

Bei der Auswahl der „Controller" wurde ja schon angedeutet, wie vielfältig die Möglichkeiten sind, u. a. kann man den ESP8266 unmittelbar in eine Smart Home Umgebung integrieren oder aber per HTTP-Abfrage Messdaten auf die eigene Homepage stellen.

Datenvisualisierung per Cloud

Die auf die bisher beschriebene Weise ermittelten und ins Netz gesandten Daten sollen nun praktisch nutzbar gemacht werden. Wir zeigen hier einige Möglichkeiten, die vor allem mit der Web-Plattform „ThingSpeak" zugänglich gemacht werden können. Natürlich sind die Daten per HTTP-Aufruf auch direkt auf die eigene Website oder in eine spezielle Visualisierungs-Applikation zu versenden, wie z. B. in das Homematic System via ioBroker und Simple API (siehe [10]) oder, ebenfalls unter [10] zu finden, direkt an die CCU über CuxD.

Die Visualisierung über „ThingSpeak" ist sehr einfach lösbar, gleichzeitig bietet diese Lösung auch eine grafische Aufbereitung und Übernahme z. B. auf die eigene Website, auf mobile Viewer oder aber als Alarmmeldung an, falls Grenzwerte überschritten werden. Einige Lösungen wollen wir hier betrachten. Die Nutzung des Dienstes ist kostenlos, solange man nicht mehr als 3 Millionen Messwerte im Jahr überträgt. Das sind 8219 je Tag, dies dürfte für die meisten Anwendungen ausreichend sein.

Auf ThingSpeak [9] muss man zunächst ein Benutzerkonto anlegen. Nach dem Bestätigen einer E-Mail von ThingSpeak kann man sich mit seinen Zugangsdaten einloggen und einen eigenen Anzeigekanal für die Abfrage der vom ESP-Web-Server erfassten Daten einrichten. Die Mutterseite „Mathworks" bietet eine sehr umfangreiche Dokumentation [11], die keine Fragen offen lässt.

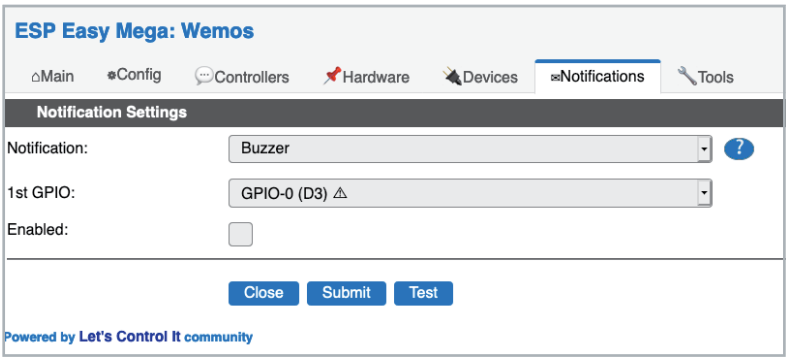

Bild 17: Im Notification-Menü kann man Benachrichtigungen konfigurieren.

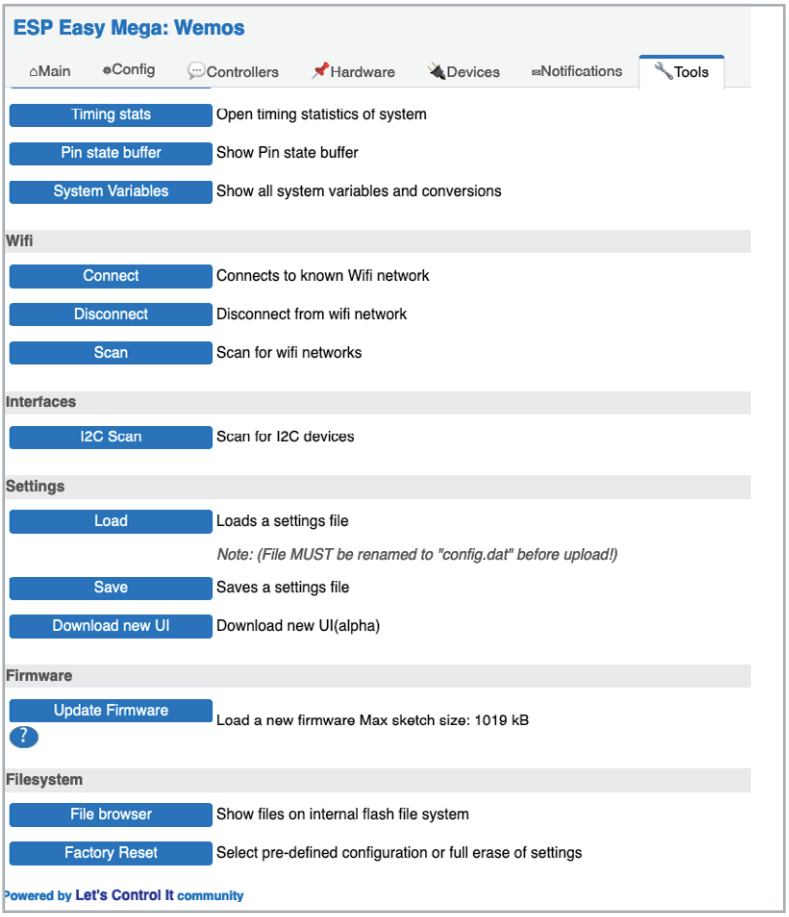

Bild 18: Das Tools-Menü hält zahlreiche und hilfreiche Systemeinstellungen und -kontrollen bereit.

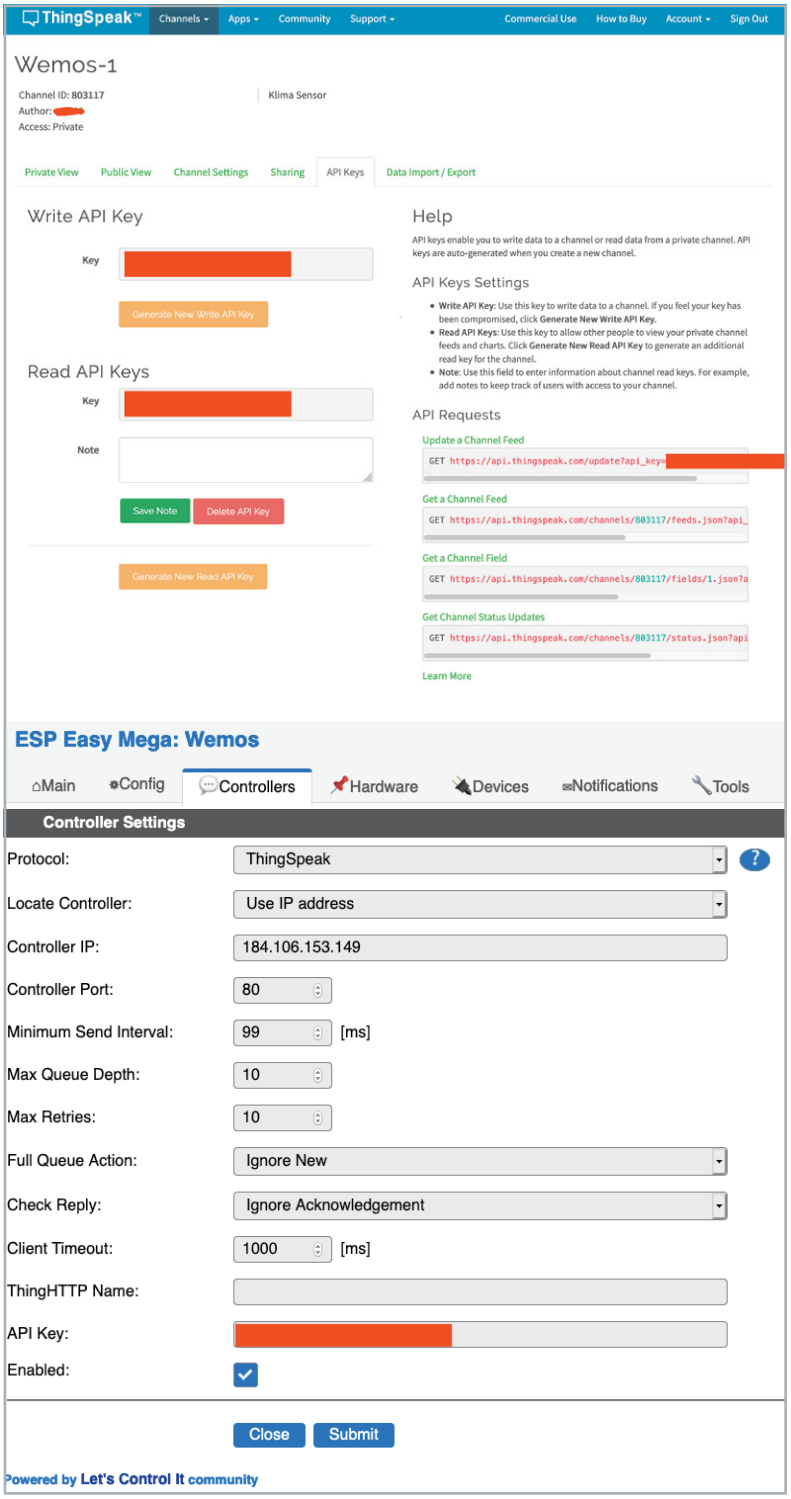

Bild 19: Die Zugangsdaten für ThingSpeak werden im Controllers-Menü hinterlegt, die wichtigste Eintragung ist hier der von ThingSpeak vergebene API-Schlüssel. Statt der IP-Adresse von ThingSpeak kann auch die Web-Adresse eingetragen werden.

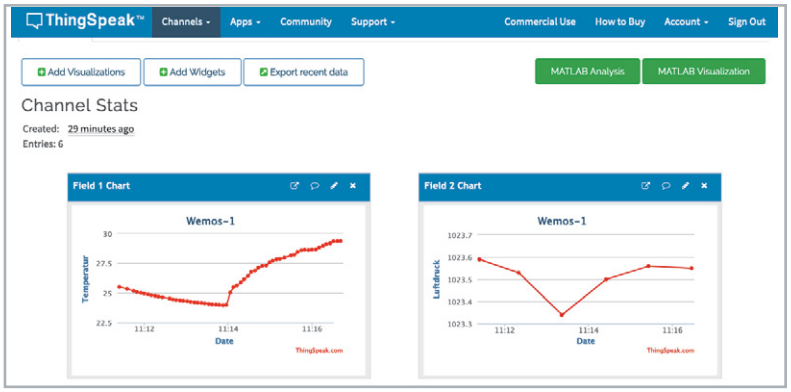

Bild 20: Das Ergebnis nach ordnungsgemäßer Verbindung – die Klimadaten sind im Netz.

Man erhält nach dem Login eine eigene Channel-ID. Diese und den unter „API-Keys" angezeigten Write API-Key benötigt man, um ihn im ESPEasy-Web-Server unter „Controllers"->„Edit" einzutragen, nachdem man „ThingSpeak" ausgewählt hat (Bild 19). Hier wird dann noch die IP bzw. der Name von ThingSpeak eingetragen, als Controller-Port übernehmen wir die Voreinstellung 80. Wichtig ist es, unter „Tools"->„Advanced"->„Message Interval" den Wert 15000 ms einzutragen, um die zugelassene Anzahl der Sendungen an das Portal nicht zu überschreiten. Diese Hinweise finden sich auch, wie die zu anderen Controllern, im letscontrolit-Wiki [12].

Hat man die Eingaben im ESPEasy-Web-Server abgeschlossen, werden bei ordnungsgemäßer Verbindung die Sensordaten unmittelbar bei ThingSpeak erscheinen (Bild 20). Hier hat man weitere Möglichkeiten, die Momentan-Daten zu visualisieren, etwa über eine Dashboard-Darstellung oder in nummerischen Fenstern (Bild 21). In den Messkurven ist jeder Datenpunkt anwähl- und mit allen zugehörigen Daten sowie der Erfassungszeit als nummerisches Feld anzeigbar.

Über „Sharing" kann man die Daten mit anderen teilen, entweder allgemein oder nur mit bestimmten ThingSpeak-Mitgliedern.

Man kann die Anzeige aber auch, nachdem man die Sharing-Option gewählt hat, in seine eigene Website integrieren. Dazu kopiert man nach Anklicken der Sprechblase im jeweiligen Chart-Fenster des ThingSpeak-Channels die dort erscheinende Tag-Info und kopiert diese in einen neu anzulegenden HTML-Frame auf der eigenen Website, den man dann individuell beschriften und gestalten kann. Wir haben dies einmal auf einer eigens angelegten Demo-Website ausgeführt, das Ergebnis kann man in Bild 22 sehen. Das funktioniert also ähnlich einfach wie die Übernahme eines Wetter-Widgets, das hier ebenfalls abgebildet ist. Auf diese Weise lässt sich ganz leicht eine komplette Website erstellen, die etwa im Smart Home Wand-Display mit mediola AIO NEO als einfach integrierbare Unterseite per Web-Aufruf anleg- und aufrufbar ist.

Will man seine Daten schnell unterwegs über eine Smartphone-App ansehen, empfiehlt sich eine der zahlreichen ThingSpeak-Apps, etwa die freie App „ThingView", die man für Android und iOS beziehen kann. Hier benötigt man die unter ThingSpeak unter „API-Keys" angebotene Read-API, und nach wenigen weiteren Konfigurationsschritten erscheinen die Daten des Kanals auf dem Display (Bild 23).

Eine weitere Visualisierung dieser Art ist das Widget „IoT ThingSpeak Monitor" (Bild 24), das zum einen als Widget mit der aktuellen Datenanzeige agiert und

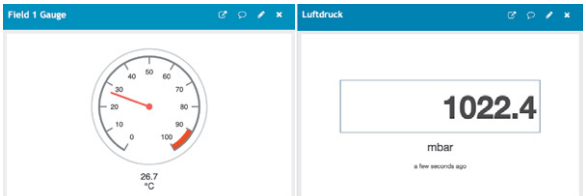

Bild 21: Bei ThingSpeak lassen sich die Daten unterschiedlich visualisieren, hier Beispiele für unseren Sensor.

Bild 22: Die von ThingSpeak aufbereiteten Daten lassen sich auch so nutzen – Beispiel auf einer Demo-Webseite

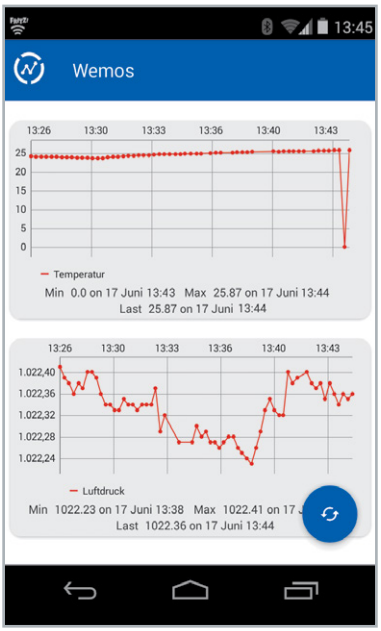

Bild 23: Mit der App „ThingView" lassen sich die Daten auf einem Mobilgerät darstellen.

zum anderen ebenfalls eine grafische Log-Funktion bietet. Derartige Apps gibt es viele, etwa auch, um sich beim Überschreiten von Grenzwerten auf dem Smartphone per Push-Mitteilung alarmieren zu lassen.

All diese Möglichkeiten, die an dieser Stelle nicht erschöpfend beschrieben werden können, zeigen, dass man nicht unbedingt aufwendig programmieren muss, um heute eine bequeme und auch umfangreiche Erfassung und Visualisierung von Messdaten per Funk zu betreiben. ELV

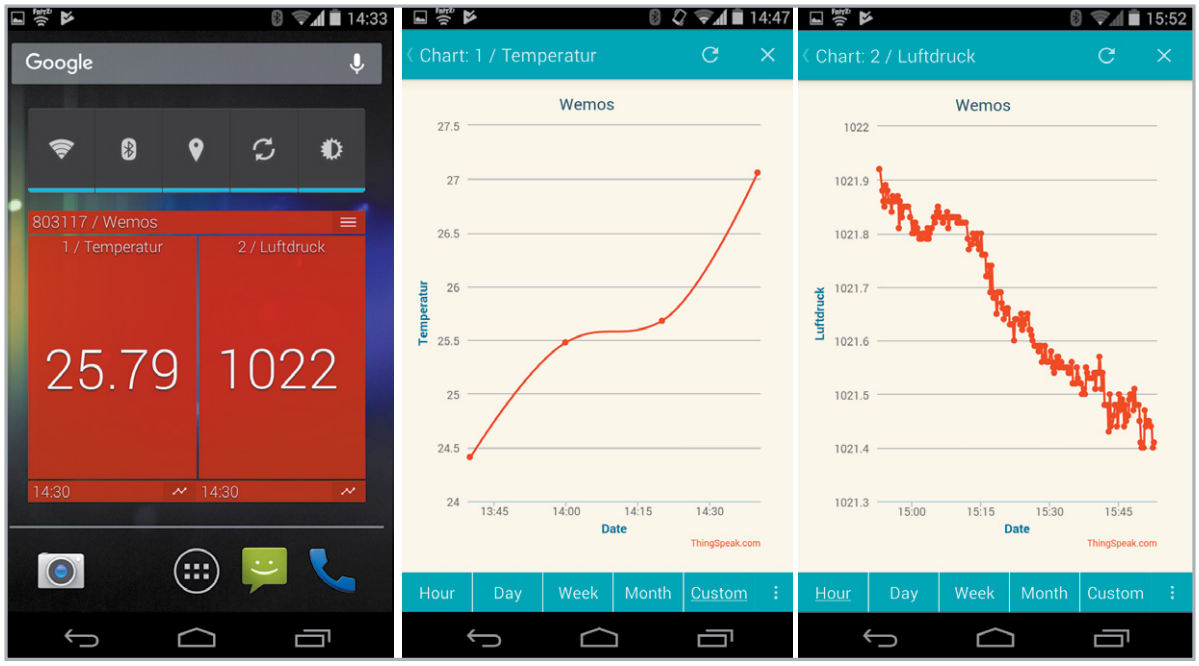

Bild 24: Das Widget „IoT ThingSpeak Monitor" zeigt die Momentanwerte, aber auch geloggte Werte an.

Weitere Infos:

[1] https://www.espressif.com
[2] https://www.espressif.com/en/products/software/esp-at/resource
[3] http://www.lua.org/
[4] https://www.letscontrolit.com
[5] https://github.com/letscontrolit/ESPEasy/releases/tag/mega-20190607
[6] https://www.ntppool.org/de/use.html
[7] https://www.msxfaq.de/sonst/bastelbude/esp8266/index.htm
[8] https://letscontrolit.com/wiki/index.php?title=ESPEasy32
[9] https://thingspeak.com/
[10] https://homematic-forum.de/forum/viewtopic.php?f=31&t=29321&sid=ecd758f11b910bf89a5db5037b8d2ecb
[11] https://de.mathworks.com/help/thingspeak/index.html?s_tid=CRUX_lftnav
[12] https://www.letscontrolit.com/wiki/index.php/EasyProtocols#ThingSpeak