Technikerschule der Stadt Ingolstadt
Fachbereich Elektrotechnik

Projektarbeit

Bewerbung zum BVT-Award 2021

Autarkes Ultra-Low-Power IoT-Entwicklungsmodul mit LoRaWAN [1]

Zur Langzeiterfassung von Umweltdaten und deren Visualisierung.

Thomas Köppl

Klasse 2EV

Schuljahr 2019 / 2020

[1] Im Auftrag der **Digital Workbench GmbH**
St.-Gangolf-Straße 2, 85139 Wettstetten

Projektarbeit zum staatlich geprüften Techniker

Einleitung

Der Einzug des Internet der Dinge in Industrie, Forschung und dem alltäglichen Leben ist kaum mehr aufzuhalten. Immer weitere Aufgabengebiete werden für IoT-Module gefunden und sich dadurch Lösungen zu verschiedenen Problemen erhofft.
Hierbei handelt es sich meist um das Sammeln von Langzeitdaten aus Umwelt- und Industrieprozessen, die durch die Verarbeitung in Deep Learning Systemen oder aber der einfachen Visualisierung aus Datenbanken heraus, dazu beitragen sollen Prozesse zu überwachen, zu analysieren und besser verstehen zu können.
Auch in der Landwirtschaft wird das Thema IoT beim Smart Farming, der Kontrolle von Anbauflächen oder der Verminderung von Düngemitteleinsätzen immer wichtiger.
Die IoT-Module kommunizieren bevorzugt über ein Long Range Low Power fähiges Funkprotokoll, dass im freien 868MHz Frequenzbereich arbeitet, dem LoRaWAN.
Das Funknetzwerk ist für jeden kostenlos nutzbar, solange man sich an die Regeln, herausgegeben von der LoRa Alliance, hält. Entwickelt wurde LoRaWAN von der Firma Semtech; seinen Ursprung hat es in der Weltraumkommnikation. Die Reichweite beträgt bis zu 15km pro Gateway. Die Gateways sind mit einem zentralen Server-Netzwerk verbunden, von wo aus die Pakete weiter verteilt werden.

Background Ingolstadt

In Ingolstadt spielt die Digitalisierung eine immer wichtigere kommunale Rolle um den Standort für die Zukunft in den Bereichen der Lebens- und Arbeitsqualität attraktiv zu halten. Die Stadt verfolgt dies im Rahmen der Digitalisierungsstrategie 20|25. http://ingolstadt.digital/.
Enthalten in dieser Strategie ist auch ein digitaler Zwilling der Stadt, der mit Sensordaten aus verteilten IoT-Modulen gespeist werden soll.
Die Module kommunizieren über ein von den Stadtwerken errichtetes flächendeckendes LoRaWAN- Netzwerk.

Parallel hierzu hat sich auch eine Community aus regionalen Firmen, Hochschulen und Interessierten gebildet, bei der auch die Digital Workbench GmbH vertreten ist und die sich intensiv mit IoT und LoRaWAN beschäftigt. Bei regelmäßigen Treffen tauscht man sich zu neuen Entwicklungen und Projekten rund um IoT und LoRaWAN aus. Träger der Community ist das Digitale Gründerzentrum Brigk in Ingolstadt.
Das Brigk organisierte bereits den Hackathon 2019 in Ingolstadt, welcher ganz im Zeichen von IoT und LoRaWAN stand.
Mittlerweile finden sich immer mehr interessierte Firmen, wie Audi AG oder Anylink, sowie Gäste der Stadtwerke Regensburg bei den Treffen ein (Stand März 2020).
Dies zeigt, dass der Einstieg in IoT und LoRaWAN ein Einstieg in eine zukunftsorientierte Technologie ist. Eine Verbreitung in vielen Bereichen unseres Lebens wird immer offensichtlicher.

Digital Workbench GmbH

Die Aufgabe für die Projektarbeit wurde mir von der in Wettstetten (bei Ingolstadt) ansässigen Firma Digital Workbench GmbH gestellt.
Während der Projektarbeit waren meine Hauptansprechpartner Herr Josef Schmidt und Herr Stefan Rupp, die mir sehr viel Vertrauen in die selbstständige Umsetzung entgegen gebracht haben und mir bei Fragen immer zur Seite standen.

Aufgabenstellung

Ziel der Projektarbeit war es, ein Elektronik-Modul zu entwickeln, dass den Einsatz von Rapid Prototyping im IoT Bereich ermöglicht.
Der Vorteil zu anderen erhältlichen Prototyping-Modulen sollte darin bestehen, dass es bereits vollständig für den Betrieb im Ultra-Low-Power-Modus ausgelegt und komplett modular aufgebaut ist. Dadurch können Anwendungen vor Ort ’feldnah’ getestet und auf Zuverlässigkeit und tatsächlichen Nutzen geprüft werden.
Eine weitere Aufgabe bestand in der Visualisierung der Messdaten, die über eine Kombination von The Things Network, Node-Red, InfluxDB und Grafana abgefangen, aufbereitet, gespeichert und angezeigt werden.
Aus der Projektarbeit entstand das Basiskommunikationsmodul, kurz BaKoMo.

Eigenschaften des Basiskommunikationsmoduls:

- Optimiert auf Ultra-Low-Power Betrieb
- Leistungsfähiger 32-Bit ARM Cortex M0+ Prozessor (mit bis zu 32MHz Systemtakt, 20kByte RAM und 192kByte Flash Speicher auch für Rechenintensive und komplexe Programme, z.B. immer häufiger genutztes Edge Computing)
- 99% Standby-Betrieb mit einem Stromverbrauch von unter 400 nA
- Sequentielle Messdatenerfassung mit einstellbarer Zyklus- / Standbyzeit bis 2 Stunden
- Energieversorgung über Lithium-Thionylchlorid Batterien bis zu 10 Jahre möglich
- Mögliche Erweiterung/Umstellung auf den Betrieb mit Lithium-Ionen-Akkus und Energy-Harvestern
- Herausführung der gängigen Schnittstellen für Sensoren aus dem Embedded-Bereich
- Modem-Schnittstelle für den Einsatz von LoRaWAN oder anderen Funktechnologien
- Neu programmierte Firmware in C++ (Objektorientiert)
- Benötigte Teile der Hard- und Software können einfach und zeitsparend in konkreten End-Anwendungen implementiert werden.

Im Rahmen der Projektarbeit wurden zwei verschiedene Szenarien realisiert.
Ein Show-Case in dem ein ’Bosch BME280 Sensor’ die Temperatur, die Luftfeuchte und den Luftdruck misst, per Funk übermittelt und parallel dazu auf einem eInk-Display ausgibt.
Und als Zweites ein Use-Case, das derzeit auf einer Wiese mit fünf Temperatursensoren vom Typ DS18S20 die Lufttemperatur und an vier Stellen die Bodentemperatur misst und übermittelt.
Beide Module übertragen zusätzlich auch die Batterie-, Prozessorspannung und Sperrschichttemperatur.

Figure 1: BaKoMo Show- and Usecase

Hardware

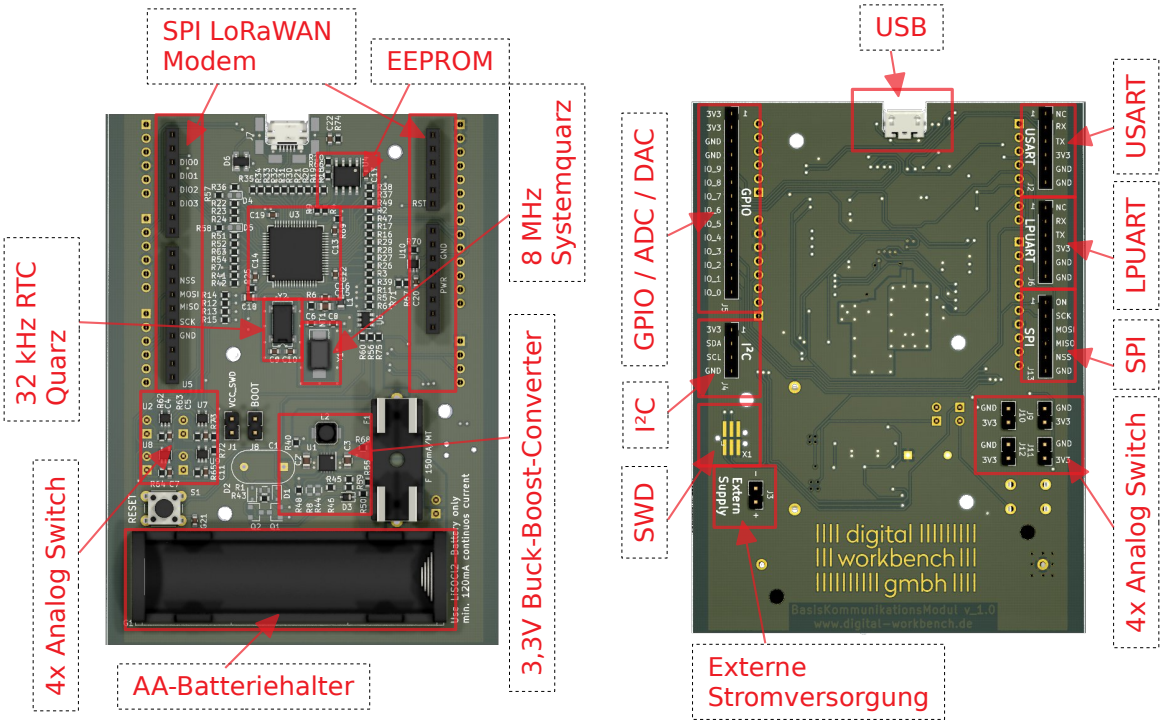


Figure 2: Top- und Bottom-Platinenansicht

Die Grafiken zeigen die Top- und Bottom-Ansicht der entwickelten Platinen mit den wichtigsten Komponentengruppen und Schnittstellen:

-**AA-Batteriehalter:** ausgelegt für Lithium-Thionylchlorid Batterie mit 2200mAh; der Controller wird direkt über die Batteriespannung versorgt.

-**EEPROM:** mit 256 Byte Speicher - zum Zwischenspeichern der Messwerte in der LoRa-Paket-Formatierung, Frame Counter, Zähler für gepeicherte Bytes und Messzyklen.

-**32 kHz Quarz:** zum Takten der Real Time Clock.

-**8 MHz Quarz:** zur Erzeugung des Systemtakts.

-**Buck-Boost-Converter:** zur Erzeugung der 3,3V Spannungsversorgung für Peripheriegeräte.

-**Analog Switch** zur Abschaltung von einzelnen Peripheriegeräten, wie z.B. Sensoren.

-**Externe Stromversorgung:** zum Anschluss von Energy-Harvestern oder anderen Versorgungsquellen.

-**SPI Modem:** Schnittstelle für das Ansteuern von Funkmodulen - Für die Projektarbeit wurde das Semtech MBed Shield SX1272, das ins LoRaWAN sendet, verwendet.

-**SWD:** Schnittstelle zum Debuggen und Programmieren.

-**Ausgeführte Schnittstellen:** GPIO, ADC, DAC, I²C, SPI, LPUART, USART und USB.

Das Platinenlayout und Routing wurde von mir mit der Software KiCAD durchgeführt.
Die Platine im 4-Layer-FR4-Aufbau wurde von einer externen Firma geätzt.
Bei Digital Workbench habe ich die Platinen anschliessend bestückt und gelötet.

Software

Die STM32-Controller-Firmware wurde komplett neu geschrieben und hierbei auf Objektorientiertes C++ umgestellt.
Als Entwicklungsumgebung kam eine Eclipse IDE zum Einsatz, die mit Plug-In's für spezielle STM32 Debugging Funktionen erweitert ist.
Auf den Einsatz eines Real Time Operating Systems, wie FreeRTOS, wurde verzichtet, da das Programm einen vollständig linearen Ablauf aufweist. Das Versetzen des Controllers in den Ultra-Low-Power-Standby Modus musste in Assembler umgesetzt werden.

Die Software ist komponentenweise in Klassen eingeteilt. Die Klassen sind wiederum in 3 Level eingeteilt. Die Methoden eines Levels nutzen ausschliesslich Methoden eines niedrigeren oder gleichwertigen Levels.

Low-Level-Methoden: Sie bilden die Basisfunktionen auf Byte-Ebene ab und sprechen direkt die Register des Controllers an; sie sind 'private' ausgeführt und nicht vom Benutzer aufrufbar (z.B. das Auslesen oder Beschreiben der SPI Register).

Middle-Level-Methoden: Sie sind vom Benutzer zwar aufrufbar, werden aber nur in Ausnahmefällen im Hauptprogramm benötigt. Hierunter fallen z.B. das Senden einzelner Bytes bei SPI ohne dass die Schnittstelle korrekt initialisiert wird.

High-Level-Methoden: Werden vom Benutzer verwendet, um die Komponenten anzusprechen, wie z.B. das Übermitteln einer kompletten Byte-Sequenz mit korrektem Ansprechen des Teilnehmers über SPI und Fehlerkorrektur.

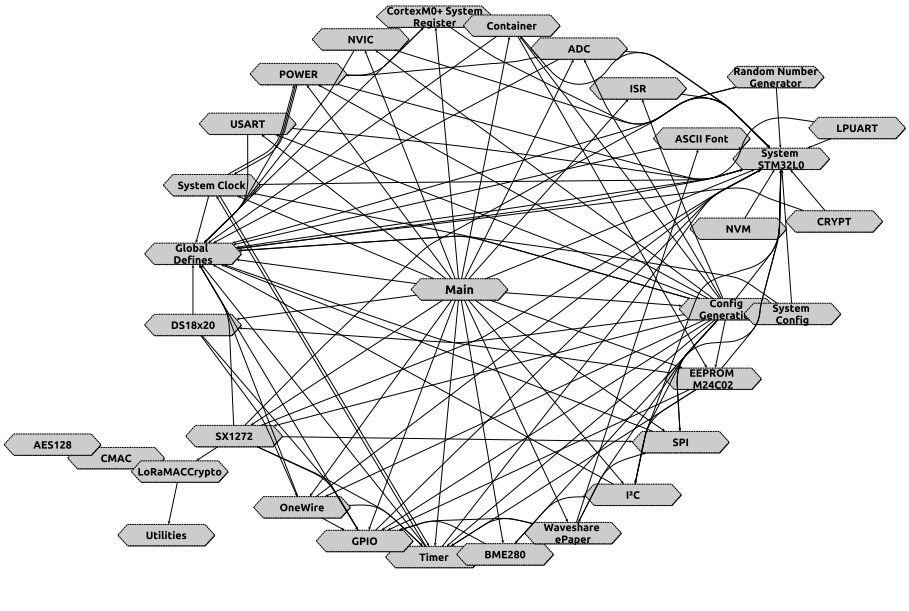
Das erstellte Programm mit den Treiber für die Komponenten umfasst:

- 19.000 Zeilen Programmcode (748kByte)

- 70 kByte Flash- und 2kByte RAM Speicher

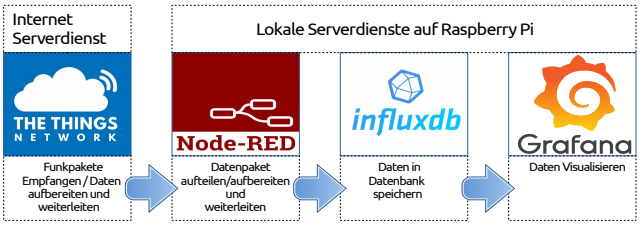
Treiber - z.B. für das Ansprechen des LoRa-Modems, des eInk Displays über SPI, des BME280 über I²C oder Protokolle wie OneWire Bus für die DS18S20 Temperatursensoren u.ä. - wurden von Grund auf neu programmiert und an die entwickelte Firmware angepasst. Dies ermöglicht die höchste Flexibilität und Funktionalität in Bezug auf das Anwenden der Low-Power-Strategien wie z.B. dem Ausschalten der Spannungsversorgung einzelner Peripheriekomponenten, wenn diese nicht oder nur kurz benötigt werden.

Die Grafik zeigt die einzelnen Klassenkomponenten der Firmware, die für den Betrieb programmiert wurden, und deren Abhängigkeit zueinander:

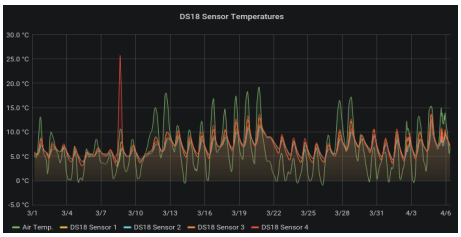


Visualisierung

Die Visualisierung erfolgt über eine Kombination verschiedener Server-Dienste, die die gesendeten Daten empfangen, aufbereiten, speichern und darstellen.



Übersicht der Serverdienste



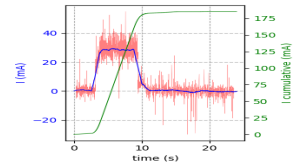
Auszug aus der Visualisierung

Stromverbrauchsmessung

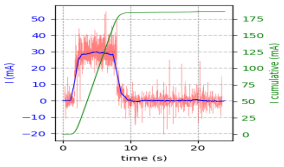
Die Ruhestrommessung wurde mit einem Multimeter durchgeführt; diese liegt im Standby-Modus des Prozessors bei ca. 400nA.
Die zeitabhängige Strommessung des Mess- und Sendezyklus wurde mit einem Oszilloskop über den Spannungsabfall an 1 Ohm durchgeführt und in eine CSV-Datei exportiert.

Die Messungen wurden aufgrund des hohen Rauschens in dem sehr niedrigen Spannungsbereich mit iPython durch einen Savitzky-Golay-Filter geglättet, aufbereitet und in Matplotlib dargestellt.

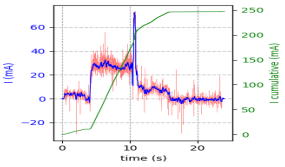
Angezeigt ist die Strommessung des Moduls mit fünf DS18S20-Temperatursensoren. Die Messwerterfassung erfolgte alle 30 Minuten.
Die berechnete Laufzeit der Applikation beträgt **2 Jahre 65 Tage und 4 Stunden**



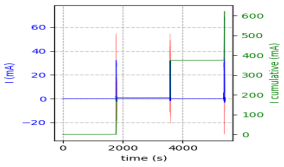
Zyklus 1 - Messung



Zyklus 2 - Messung



Zyklus 3 - Messen und Senden



Gesamtzyklus

Fazit

Im Rahmen der Projektarbeit konnte ich reichlich Erfahrung mit eingebetteten System und den 32-Bit ARM-Cortex-Prozessoren gewinnen. Das Programmieren einer eigenen Firmware war anfangs nicht geplant und auch mit sehr viel Programmierarbeit verbunden. Ich bin mir aber sicher, dass nur dadurch die volle Funktionalität des Controllers gewährleistet ist. Durch den Einsatz von C++ ist es jetzt ohne weiteres möglich, sehr einfach und schnell neue Sensoren oder Funktionen einzuprogrammieren.
Desweiteren war es sehr interessant herauszufinden, wie man Controller so stromsparend aufbauen kann, sodass diese weder Akkus noch Solarzellen benötigen, sondern mit einer Batterie eine sehr lange Laufzeit erreichen.
Mittlerweile wurde die Platine noch ein weiteres Mal mit einem integriertem Modem SX1272 aufgebaut und für den Einsatz in Bienenstockwaagen optimiert.
Die Projektarbeit hat mir den Spass an dem Thema näher gebracht und auch meine weitere berufliche Ausrichtung dorthin maßgeblich beeinflusst.
Ich hoffe, dass wir noch weitere Anwendungen für das Modul finden, welche für das Verstehen von Prozessen hilfreich sein können.
Mit dem Basiskommunikationsmodul sind wir jedenfalls bestens dafür vorbereitet.