Weekly Reports

Luftqualität in Innenräumen - Gruppe 1

6. Dezember 2021

Name	Matrikel Nr.	Arbeitsaufwand (h)
Friedrich Just	1326699	18,00
Stipe Knez	1269206	17,00
Lucas Merkert	1326709	17,00
Achim Glaesmann	1309221	19,00
Max-Rene Konieczka	1211092	16,00
Can Cihan Nazlier	1179244	19,00

Tabelle 1: Arbeitsaufwand dieser Woche

1 Überblick

1.1 Friedrich Just

Der SCD41 Sensor hat die I^2C-Bus Adresse 0x62 [2]Seite 7. An dem Sensor sind 4 Kabel. Das grüne Kabel geht an den SDA-port (Serial Data Line). Das gelbe Kabel geht an den SCL-port (Serial Clock). Das schwarze Kabel ist der GND-port (Ground). Das rote Kabel ist an den VDD-port (Power) angeschlossen [5].

Mit dem hexadezimal (von nun an hex.) Code 0x21b1 wird alle 5 Sekunden eine Messung durchgeführt, mit dem hex. Code 0xec05 wird die periodische Messung beendet. Wenn der Sensor den hex. Code 0xec05 sendet, bekommt er das Ergebnis der Messung zurück. Das sind die wichtigsten Kommandos für das Auslesen des Sensors die vollständige Liste siehe Abbildung 2.

Für jeden Messwert werden 3 Byte eingelesen, davon sind 2 Byte Messauflösung und 1 Byte CRC (Cyclic Redundancy Check) checksum für die Fehlererkennung. Dies und die Umrechnung der Messwerte in eine lesbare Form werden in der Tabelle Abbildung 1 dargestellt.

Für ein besseres Verständnis der Programmierung des Sensors wurde dieser an einem Arduino Nano angeschlossen. Auf Github gibt es bereits einen funktionierenden Code [8] zum Auslesen dieses Sensors und die Umrechnung der Messwerte. Damit ist verständlicher geworden, wie der Sensor über unseren Microcontroller angesteuert werden muss. Durch den Arduino kann man später die Messwerte mit denen des Microcontrollers vergleichen werden.

Write (hexadecimal)	**Input parameter:** -		**Response parameter:** CO_2, Temperature, Relative Humidity		Max. command duration [ms]
	length [bytes]	signal conversion	length [bytes]	signal conversion	
0xec05	-	-	3	CO_2 [ppm] = word[0]	1
			3	T = - 45 + 175 * word[1] / 2^{16}	
			3	RH = 100 * word[2] / 2^{16}	

Example: read sensor output (500 ppm, 25 °C, 37 % RH)

Write	0xec05				
(hexadecimal)	*Command*				
Wait	1 ms	*command execution time*			
Response	0x01f4	0x7b	0x6667	0xa2	0x5eb9	0x3c
(hexadecimal)	*CO_2 = 500 ppm*	*CRC of 0x01f4*	*Temp. = 25 °C*	*CRC of 0x6667*	*RH = 37 %*	*CRC of 0x5eb9*

Abbildung 1: Auswertung einer Messung [2]

Domain	**Command**	**Hex. Code**	**I²C sequence type (see chapter 3.3)**	**Execution**	
				time [ms]	**During meas.***
Basic Commands Chapter 3.5	start_periodic_measurement	0x21b1	send command	-	no
	read_measurement	0xec05	read	1	yes
	stop_periodic_measurement	0x3f86	send command	500	yes
On-chip output signal compensation Chapter 3.6	set_temperature_offset	0x241d	write	1	no
	get_temperature_offset	0x2318	read	1	no
	set_sensor_altitude	0x2427	write	1	no
	get_sensor_altitude	0x2322	read	1	no
	set_ambient_pressure	0xe000	write	1	yes
Field calibration Chapter 3.7	perform_forced_recalibration	0x362f	send command and fetch result	400	no
	set_automatic_self_calibration_enabled	0x2416	write	1	no
	get_automatic_self_calibration_enabled	0x2313	read	1	no
Low power Chapter 3.8	start_low_power_periodic_measurement	0x21ac	send command	-	no
	get_data_ready_status	0xe4b8	read	1	yes
Advanced features Chapter 3.9	persist_settings	0x3615	send command	800	no
	get_serial_number	0x3682	read	1	no
	perform_self_test	0x3639	read	10000	no
	perform_factory_reset	0x3632	send command	1200	no
	reinit	0x3646	send command	20	no
Low power single shot (SCD41 only) Chapter 3.10	measure_single_shot	0x219d	send command	5000	no
	measure_single_shot_rht_only	0x2196	send command	50	no

Abbildung 2: Befehle zum Ansteuern des SCD41 Sensors [2]

1.2 Lucas Merkert

Einarbeitung SHT21: Der SHT21 Sensor wird über den I^2C-Bus angesprochen Abbildung 3 um die Temperatur und relative Luftfeuchtigkeit zu messen. Der Sensor gibt die Temperatur in einer 14 Bit Auflösung und die Relative Luftfeuchtigkeit in einer 12 Bit Auflösung zurück. Die Werte können dann mit den Formel

aus Abbildung 4 und Abbildung 5 berechnet werden. Dabei ist das Problem aufgetreten wie genau man den Sensor über HAL_WriteI2cPacket() anspricht. Mit der Lösung aus dem Forumpost sollte sich dies geklärt haben.

Command	Comment	Code
Trigger T measurement	hold master	1110'0011
Trigger RH measurement	hold master	1110'0101
Trigger T measurement	no hold master	1111'0011
Trigger RH measurement	no hold master	1111'0101
Write user register		1110'0110
Read user register		1110'0111
Soft reset		1111'1110

Abbildung 3: Befehle zum Ansteuern des SGT21 Sensors, T für Temperatur, RH für relative Luftfeuchtigkeit [3]

$$T = -46.85 + 175.72 \cdot \frac{S_T}{2^{16}}$$

Abbildung 4: Formel zur Berechnung der Temperatur [3]

$$RH = -6 + 125 \cdot \frac{S_{RH}}{2^{16}}$$

Abbildung 5: Formel zur Berechnung der relativen Luftfeuchtigkeit [3]

Luftqualitäts Faktoren [6]

- Luftfeuchte
  1. $> 23\%$ => Feuchtigkeitsverlust beim Atmen nur noch bedingt kompensierbar, Bauschäden, höhere Chance auf Elektroschocks
  2. $> 40\%$ => austrocknen der Haut, Schleimhaut und Augen
  3. $< 60\%$ => Schimmelbildung => Asthma, Allergien
  4. $< 80\%$ => optimale Feuchtigkeit für Milben, Parasiten, Pilzen
- VOC
  1. Emissionsquellen: Bauprodukte, Möbel, Lack, Lösungsmittel verdunsten, Tabakrauch, Menschen, Tiere, Mikroorganismen höheres Risiko bei Neubau/Renovierung
  2. Schäden. Geruchsbelästigung, Atemwegs-/Augenreiz, Schädigung des Nervensystems, Allergien, Krebs, Erbgutschädigung, Fortpflanzungsschädigen
  3. RW 1: lebenslange Aussetzung führt zu keine Auswirkung (¿0,3mg/m^3)

4. RW 2: Schäden bei anfälligen Leuten sind zu erwarten
5. RW gilt für alle Räume in denen keine Gefahrenstoffe verwendet werden
6. VOC misst die gesamt Anzahl der Partikel in der Luft, einzelne Partikel könne daraus nicht abgelesen werden

- Feinstaub
  1. Arten von Feinstaub: PM10, PM2.5, PM0.1 (Mikrometer)
  2. Je kleiner die Partikel desto weiter können diese in den Körper, vor allem die Lunge eindringen und diese beschädigen
  3. Emissionsquellen: Tabakrauch, Kerzenruß, Kochen, Computer, Drucker, Heizen ohne lüften
  4. Maßnahmen: saugen, wischen(nass!), lüften, nicht mehr als 22C heizen, Dunstabzugshaube in der Küche, Luftreiniger

1.3 Stipe Knez

Diese Woche spielte das Einarbeiten in das Projekt und die Vorbereitung auf das Erstellen von Mock-Daten mit Python und eine Recherche was das Express.js-Framework angeht bei mir eine Rolle. Nach dem tieferen Einarbeiten in das Projekt (unter Anderem in die Programmiersprache) ging es an die Recherche bezüglich des Express-Frameworks, da dieses in unserem Projekt eine Rolle spielen soll und dementsprechend Wissen über die Funktionsweise und Nutzungsweise benötigt wird. Das Express.js-Framework ist dabei ein serverseitiges Web-Framework für die JavaScript-basierte Plattform Node.js. Durch dieses Framework werden in Node.js Werkzeuge zur Verfügung gestellt, durch die sich Webanwendungen leichter entwickeln lassen [10]. Node.js ist dabei eine plattformübergreifende JavaScript-Laufzeitumgebung, die in der Lage ist Javascript-Programme außerhalb des Webbrowsers ausführen zu lassen [11]. Wie zuvor erwähnt, war die Vorbereitung auf das Erstellen von Mock-Daten mit Python auch ein Bestandteil der Arbeit in der letzten Woche. Wir möchten mit Hilfe von Python Mock-Daten zum Testen erstellen, die weiterverarbeitet werden sollen. Die damit einhergehende Recherche hat Aufschluss über die Herangehensweise beim Erstellen von Mock-Daten gegeben. Für die kommende Woche ist geplant, dieses Wissen umzusetzen und in Zusammenarbeit mit dem für die ZigBee-Daten verantwortlichen Team, das passende Format für die Daten zu wählen und diese schließlich zu erstellen [7].

1.4 Achim Glaesmann

Es wurde daran gearbeitet die Sensoren mit dem Mikrokontroller auszulesen. Das Auslesen bereitet jedoch unvorhergesehene Probleme. Die Datenblätter wurden ausführlich studiert um eine Kommunikation über I2C zu ermöglichen. [1] [2] [3] Es wurden die Befehle recherchiert und die für die Berechnungen der

Werte nötigen Umrechnungsformeln der Sensordaten. [3] [4] [5] Das I2C Protokoll wurde ausführlich studiert. Die in den Datenblättern vermerkten Adressen der Sensoren führen jedoch nicht zu dem Aufbau einer Verbindung. Zur Fehleranalyse wurde ein Skript für den Arduino entwickelt um die Verbindung mit über I2C zu kontrollieren. Über den Arduino war es möglich eine Verbindung herzustellen. Es wurde ein Skript herangezogen, welches es ermöglicht die 7 bit I2C Adressen zu überprüfen um Abweichungen vom Datenblatt zu erkennen. Beim SHT21 Sensor wurde eine Abweichung festgestellt. Eine Verwendung der so herausgefundenen I2C Adresse führte allerdings ebenfalls zu einem OpenFail. Es wurde ein Übertragungsstandard für die von den Sensoren ermittelten Daten über die UART-Schnittstelle entwickelt. Es ist geplant die Daten durch ein Semikolon getrennt in einer vorher festgelegten Reihenfolge zu übertragen. Dabei erhält jeder Mikrokontroller eine ID, die gemeinsam mit den ermittelten Daten übertragen wird. Der Aufbau wäre vorraussichtlich wie folgt: (ID; DATASENSOR1; DATASENSOR2; DATASENSOR3) Bei Multisensoren erfolgt die Weitergabe der Daten in Reihenfolge der Sensoren durch ein Semikolon getrennt. DATASENSOR1 wäre im Falle des SHT21 dann wie folgt aufgeteilt: TEMPERATURCELSIUS; LUFTFEUCHTIGKEITRH;. Es wurde weiter die Literatur studiert, um einen Überblick über Sinnvolle Gefahrenabschätzung auf Basis der vorliegenden Daten zu erhalten. [6] [1.2] Endgültige Grenzwerte werden weiter recherchiert, eine Entscheidung wird später getroffen, da das Software Team sich bisher hauptsächlich auf das Zeichentool konzentriert, empfiehlt es sich weitere Informationen einzuholen, bis die Entwicklung an dem Punkt ist, an dem die Gefahreneinschätzung etabliert werden muss.

1.5 Max-Rene Konieczka

Aufbauend zur letzten Woche, hat man sich mit der korrekten Einrichtung ein Einarbeitung des Projektes beschäftigt, welches von Can Cihan Nazlier letzte Woche konfiguriert wurde. Darüber hinaus wurde influxDB installiert, was sich gut dafür eignet Zeitreihen-Daten zu verwalten. Da die Applikation eine Reactive Web App sein wird, wurden Recherchen zum Thema Websockets und SocketIO gemacht. SocketIO ist eine JavaScript-Bibliothek, welche für Echtzeit-Webanwendungen verwendet wird. Diese ermöglicht bidirektionale Echtzeit-Kommunikation zwischen dem Browser und einem Server. Dadurch werden Benutzereingaben schneller behandelt und die App läuft flüssiger. [12, 9] Websocket wiederum ist ein Netzwerkprotokoll, was auf TCP basiert und die Kommunikation überhaupt möglich macht. SocketIO macht sich das WebSocket-Protokoll zunutze. [13, 4] Für die kommende Woche wird versucht, die im Zeichentool eingezeichneten Räume persistent zu speichern, sodass das diese nach der Schließung und erneuten Ausführung des Tools wieder verfügbar sind.

1.6 Can Cihan Nazlier

Diese Woche wurde der Prototyp des Zeichentools fertiggestellt, nachdem mit Herrn Krauße über Einzelheiten am Dienstag gesprochen wurde. Man kann nun

Räume erstellen und diese abspeichern(noch nicht persistent). Nach der Abspeicherung, verändert sich die Farbe der Räume auf dem Zeichenboard von Rot auf Grün. Im Verlauf nächster Woche wird daran gearbeitet die Räume persistent abzuspeichern, sei es in eine .json Datei oder in einer Datenbank. Nächste Woche wird dazu noch am Menü gearbeitet, welches gespeicherte Räume anzeigen und der Nutzer diese bearbeiten können soll. Eine geeignete Ticketumgebung um die Übersicht für die Entwickler zu erhalten wurde konfiguriert. Benutzt wird die Applikation 'Trello'. Zwei Boards wurden eingerichtet, jeweils eins für den Backend- und Frontend-Teil.

Literatur

[1] *Datasheet CCS811*. URL: https://learn.adafruit.com/adafruit-ccs811-air-quality-sensor?view=all#documents.

[2] *Datasheet SCD41*. URL: https://www.sensirion.com/en/environmental-sensors/evaluation-kit-sek-environmental-sensing/evaluation-kit-sek-scd41/#c50745.

[3] *Datasheet SHT21*. URL: https://www.sensirion.com/de/umweltsensoren/feuchtesensoren/feuchte-temperatursensor-sht2x-digital-i2c-genauigkeit/.

[4] *Einführung zu Websockets: Sockets im Web*. URL: https://www.html5rocks.com/de/tutorials/websockets/basics/.

[5] *Evaluation Kit SEK-SCD41*. URL: https://www.sensirion.com/en/environmental-sensors/evaluation-kit-sek-environmental-sensing/evaluation-kit-sek-scd41/.

[6] *faktoren der Luftqualität*. URL: https://www.inventer.de/wissen/luftqualitaet-gesundheit/.

[7] *How to generate dummy data in Python*. URL: https://towardsdatascience.com/how-to-generate-dummy-data-in-python-a05bce24a6c6%7D.

[8] *Sensirion / arduino-i2c-scd4x*. URL: https://github.com/Sensirion/arduino-i2c-scd4x.

[9] *Socket IO Documentation*. URL: https://socket.io/docs/v4/.

[10] *Wikipedia - Express.js*. URL: https://de.wikipedia.org/wiki/Express.js%7D.

[11] *Wikipedia - Node.js*. URL: https://de.wikipedia.org/wiki/Node.js%7D.

[12] *Wikipedia - Socket.IO*. URL: https://de.wikipedia.org/wiki/Socket.IO.

[13] *Wikipedia - WebSocket*. URL: https://de.wikipedia.org/wiki/WebSocket.