Embedded Systems

Prof. Dr.-Ing. Jürgen Wübbelmann
Dr.-Ing. Ralf Hagemann

Laborbereich Technische Informatik
Hochschule Osnabrück

Benutzerhinweise

STMNucleo32 L476RG
Embedded Systems Board rev.005 / STMNucleo32-Baseboard

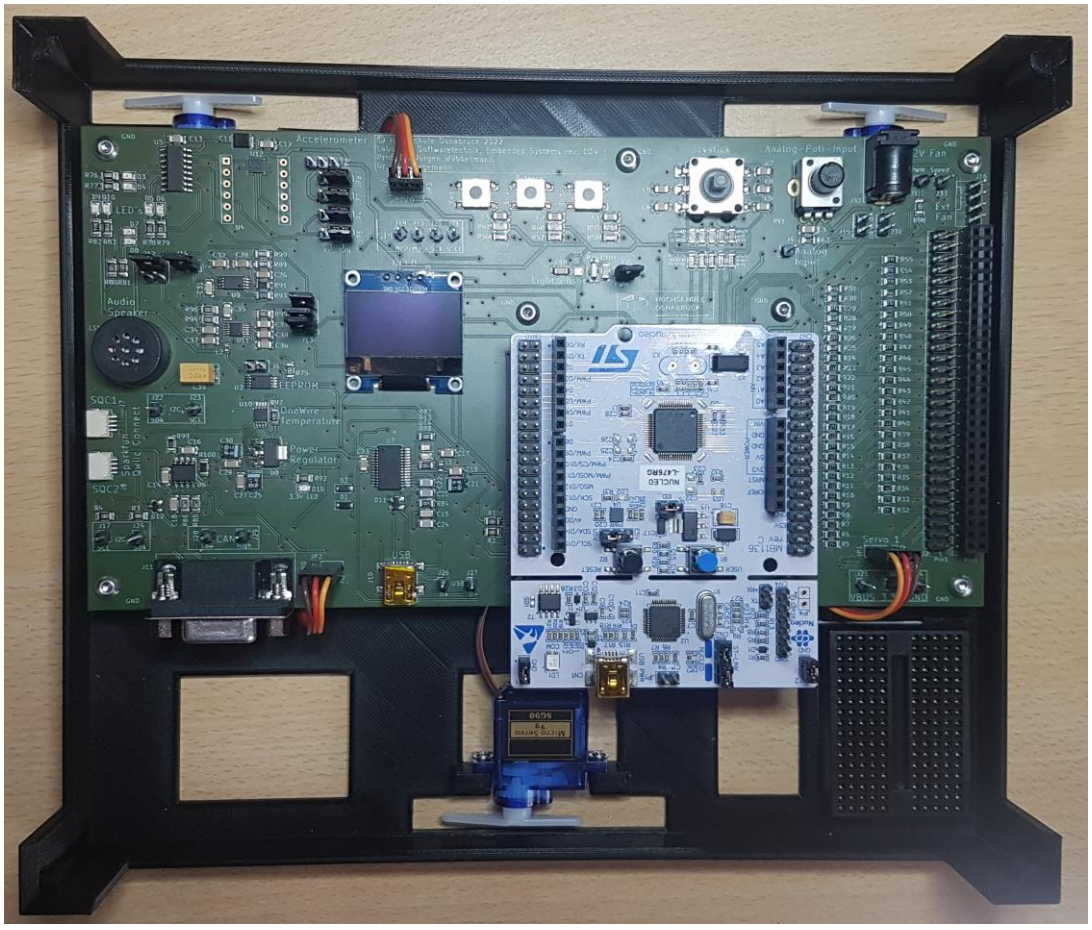

Abbildung 1: STMNucleo32-Baseboard - STMNucleo32 L476RG

Hochschule Osnabrück

Fakultät I & I
Laborbereich Technische Informatik
Albrechtstr. 30
49076 Osnabrück
www.hs-osnabrueck.de

Prof. Dr.-Ing. J. Wübbelmann, j.wuebbelmann@hs-osnabrueck.de
Dr.-Ing. R. Hagemann, r.hagemann@hs-osnabrueck.de

Feedback
Wir freuen uns über ein Feedback zu Verbesserungen an diesem Dokument. Bitte senden Sie Ihre Kommentare an j.wuebbelmann@hs-osnabrueck.de oder r.hagemann@hs-osnabrueck.de.

Versionsgeschichte des Dokuments

Version	Datum	Änderung	Autor(en)
0.1	20.06.2020	initiale Version	R. Hagemann
0.2			

Inhalt

1. Einleitung

Das Embedded Systems Board Rev.005 (kurz STMNucleo32-Baseboard) wurde an der Hochschule Osnabrück für die Veranstaltungen Embedded Systems, Modellbasierte Softwareentwicklung technischer Systeme, studentische Projekte und allgemein für das Labor für Softwaretechnik entwickelt.
Dieses Dokument gibt Hinweise für die Benutzung des STMNucleo32 Boards zusammen mit dem Embedded Systems Board Rev.005.
Der allgemeine Fokus dieses Dokumentes obliegt dem STMNucleo32-Baseboard.
Das STMNucleo32-Baseboard bietet Ihnen viele Aktoren, Sensoren und sonstiger Peripherie.
So haben Sie die Möglichkeit für einen sofortigen Einstieg in Experimente mit dem STMNucleo32 L476RG.
Die folgenden Punkte zeigen die integrierte Peripherie auf, welche mit STMNucleo32-Baseboard zur Verfügung stehen.

1.1. Integrierte Peripherie allgemein

- Steckplatz für den STMNucleo32 (U1)
- 50 Pin-Erweiterungsstecker
  mit zweireihiger Stiftleiste männlich (J2) bzw. Buchsenleiste weiblich (J3),
  zur einfachen Verbindung von zusätzlicher externer Hardware
  mit dem STMNucleo32-Baseboard / STMNucleo32.
  Hier sind nahezu alle Pins des STMNucleo32 vom Steckplatz des
  STMNucleo32 ausgeführt
-

1.2. Integrierte Peripherie Digital I/O

- Joystick rechts, links, hoch, runter, drücken (S1)
- Taster (SW1, SW2, SW3)
- Externer Lüfter (J16)
- LED liegend (D16) / korrespondiert mit dem Lichtsensor
- Onewire / Temperatursensor (U10)
- Audioverstärker (U11)

1.3. Integrierte Peripherie Analog I/O

- Lautsprecher (LS)
- Trimmer/Poti (RV1)
- Audioausgang vom STMNucleo32 zum Lautsprecher

1.4. Integrierte Peripherie Serieller Bus UART

- USB UART Schnittstelle / Mini-USB Anschluss (J10)

1.5. Integrierte Peripherie Serieller Bus SPI

- Beschleunigungssensor (alternativ über I^2C1) (U4)

1.6. Integrierte Peripherie Serieller Bus I²C1

- 2 LED's rot (D3, D4)
  2 LED's gelb (D5, D6)
  2 LED's grün (D7, D8)
  2 LED's blau (D5, D6)
- Lichtsensor (U3)
- OLED (J9)
- EEPROM (U2)
- Sparkfun Qwiic Connect (J7)
- Beschleunigungssensor (alternativ über SPI) (U4)
- Lichtsensor (U3)

1.7. Integrierte Peripherie Serieller Bus I²C2

- Sparkfun Qwiic Connect (J8)

1.8. Integrierte Peripherie Sparkfun Qwiic Connect

- 2 mal Schnittstelle Sparcfun Qwiic Connect über I²C (J7/J8)

1.9. Integrierte Peripherie CAN-Bus-Interface

- 9-polig SUB-D männlich (J11)

1.10. Integrierte Peripherie Spannungsversorgung

- 5V über Mini-USB des STMNucleo32 L476RG
- 12V über separates Netzteil (wird nur für den Betrieb des Lüftermotorversuches benutzt)

1.11. Integrierte Peripherie Servo-Regler

- Modellbau-Servo 1/PWM (JP1)
  Modellbau-Servo 2/PWM (JP2)
  Modellbau-Servo 3/PWM (JP3)

1.12. Integrierte Peripherie Messpunkte

- 5V (J25)
- 3.3V (J30)
- GND (J34)
- Analog Input von Trimmer/Poti (J4)
- I²C1 (J22/J23)
- I²C2 (J17/J24)
- CAN H / CAN L (J29/J28)
- USB-Mini P / USB-Mini N (J26/J27)
- UART_RXD / UART_TXD (J39/J38)
- LED-D3 (J37)
- Joystick Rechts (J36)

2. STMNucleo32 L476RG

Das STMNucleo32-Baseboard (Abbildung 1) wurde speziell für den STMNucleo32-L476RG (Abbildung 2) entwickelt.
Zur Programmierung des STMNucleo32 benutzen wir primär die IDE STM32CubeIDE.
Dieses Kapitel gibt einen kleinen Überblick über den Hardwareaufbau des STMNucleo32-L476RG.

Der 32-Bit-Flash-Mikrocontroller (MCU) der Familie STM32L4 basiert auf dem ARM Cortex™ M4 Core, einem Kern, der speziell für ultra-low-power Anwendungen entwickelt wurde. Der Prozessor STM32 ARM Cortex™ M4 von STMicroelectronics erreicht mit seiner Gleitkommaeinheit 100 DMIPS bei 80 MHz.

https://www.st.com/resource/en/user_manual/um1724-stm32-nucleo64-boards-mb1136-stmicroelectronics.pdf

Ein paar Eckdaten:

- On-Board-STM32L476RGT6 -Mikrocontroller
- 32.768 kHz Kristalloszillator
- On-Board-ST-LINK/V2-1-Debugger/Programmiergerät mit SWD-Steckverbinder
- Netzteile: USB-VBUS oder externe Quelle (3,3 V; 5 V; 7 ... 12 V); Messpunkt für die Stromüberwachungseinheit
- Umfassende STM32-Software-HAL-Bibliothek
- Externe Anschlüsse: SMPS, Micro-USB, MIPI

Unter anderem die dynamische Spannungsanpassung und besonders sparsame Peripherie (LP UART, LP Timer), die auch im Stop-Modus ansprechbar ist, sorgen für die geringe Leistungsaufnahme.

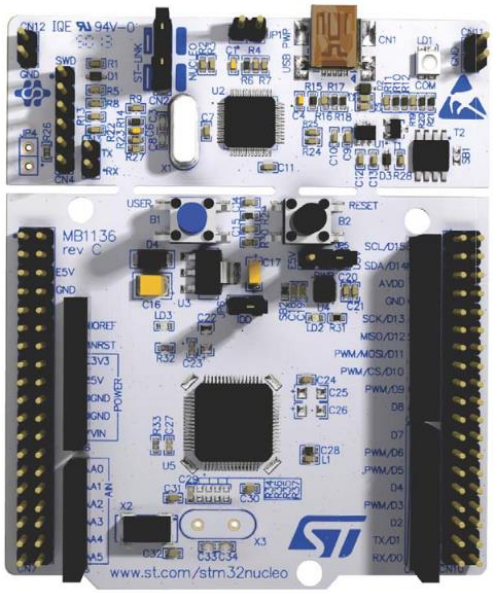

Abbildung 2: STMNucleo32 L476RG

Das STMNucleo32-L476RG Board besteht im Prinzip aus zwei Bereichen (Abbildung 3). Unterhalb der roten Linie befindet sich der eigentliche Controller. Oberhalb der roten Line befindet sich das Debugging- und Programmier-Interface.

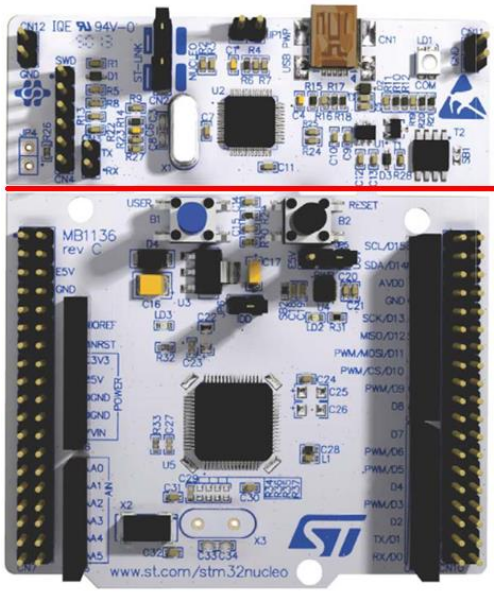

Abbildung 3: STMNucleo32 L476RG - Debugger-Abtrennung

2.1. Spannungsversorgung und USB-Anschluss

Die Spannungsversorgung für das STMNucleo32-L476RG Board erfolgt (bei unserer Anwendung) über den Mini-USB-Stecker (siehe Abbildung 4).

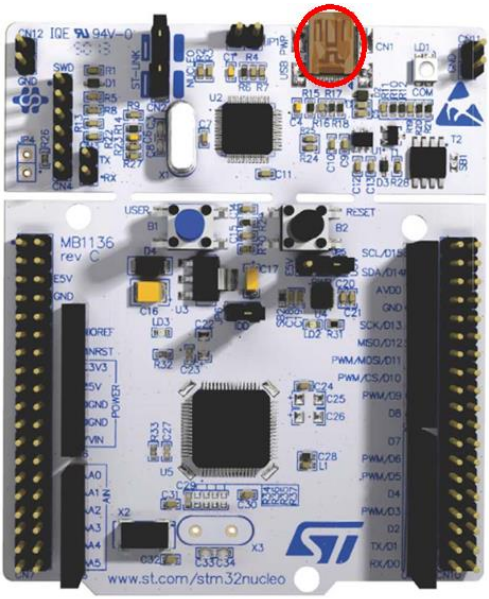

Abbildung 4: STMNucleo32 L476RG - USB

Am angeschlossenen Computer kann der USB-Anschluss als virtueller COM-Port (USART2 des Mikrocontrollers) verwendet werden. Ebenso dient dieser als Programmier- und Debugging-Anschluss (On-board ST-LINK/V2-1). Die STM32CubeIDE verbindet sich mit dem Board über diese Schnittstelle.

2.2. Anschlussleisten 2-reihig

Die 2-reihigen Leisten (Abbildung 5) sind Standard STMicroelectronics Morpho Erweiterungs-Pins und bieten vollen Zugriff auf die meisten I/Os des STMNucleo32 L476RG. Der STMNucleo32 L476RG ist über diese 2 2-reihigen Leisten mit dem STMNucleo32-Baseboard verbunden. So hat der Controller vollen Zugriff auf Hardware des STMNucleo32-Baseboards.

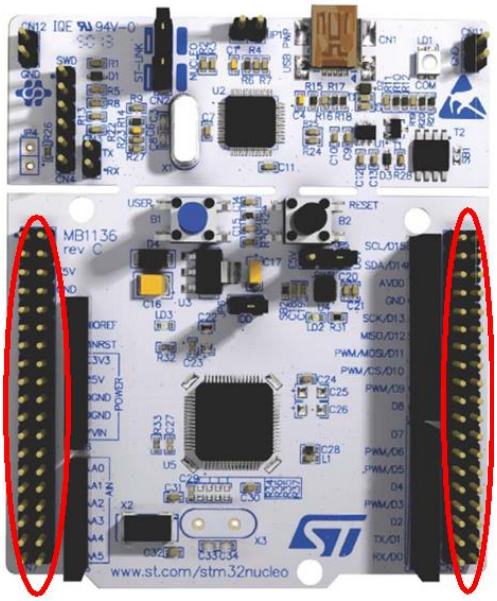

Abbildung 5: STMNucleo32 L476RG - STMicroelectronics Morpho Erweiterungs-Pins

2.3. Anschlussleisten 1-reihig

Die 1-reihigen Leisten (Abbildung 6) sind Standard erweiterungs-Pins wie beim Arduino Uno Revision 3 und bieten Zugriff auf die entsprechenden Erweiterungs-Boards / Shields.
Achtung hier muss vor einem Einsatz solcher Shields, zusammen mit dem STMNucleo32-Baseboard genau geprüft werden, ob die angestrebte Verwendung der I/Os, wegen einer ggf. Verwendung auf dem STMNucleo32-Baseboard, möglich ist.

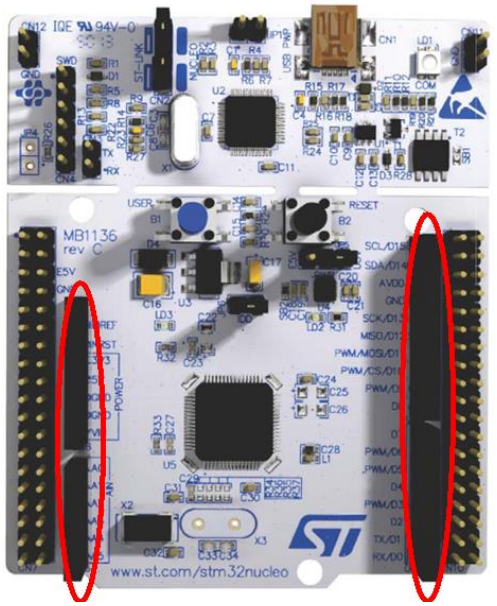

Abbildung 6: STMNucleo32 L476RG -Anschluss Arduino Uno Revision 3

2.4. Taster

Das STMNucleo32-L476RG Board besitzt 2 Taster.

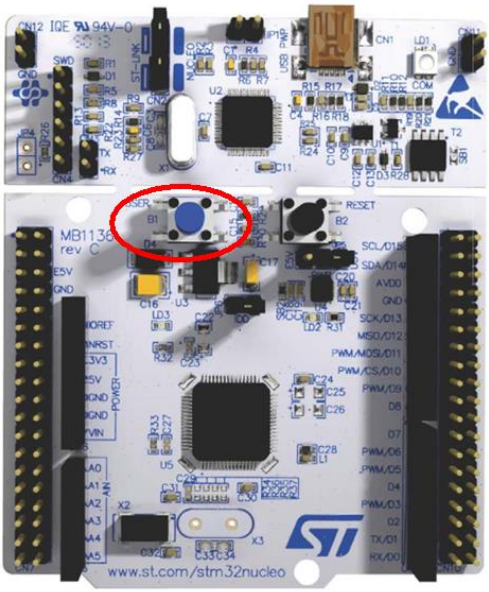

Abbildung 7: STMNucleo32-L476RG Board - Taster blau

Der blaue Taster B1 (Abbildung 7) kann für die benutzerspezifisch genutzt werden. Eine nähere Beschreibung siehe Kapitel 3.4.

Abbildung 8: STMNucleo32-L476RG Board - Taster schwarz

Der schwarze Taster B2 (Abbildung 8) ist der Reset-Taster für das STMNucleo32-L476RG Board. D. h. beim Betätigen des Taster B2 startet der STMNucleo32 neu (bei angeschlossener Spannungsversorgung).

2.5. LEDs

Das STMNucleo32-L476RG Board besitzt 3 LEDs,
USB-Kommunikation LD1 (rot/grün), LED LD2 (grün) und eine Power-LED LD3 (rot).
Eine nähere Beschreibung zur Verwendung der LED LD2 (Abbildung 9) siehe Kapitel 3.10.

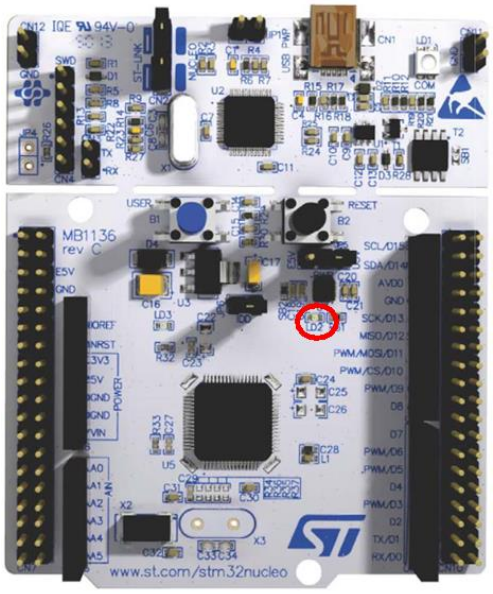

Abbildung 9: STMNucleo32-L476RG Board - LED

2.6. Jumper

Um das STMNucleo32-L476RG Board zusammen mit dem STMNucleo32-Baseboard betreiben zu können sind nachfolgende Jumpereinstellungen notwendig.
Alle in ROT (Abbildung 10) markierten Positionen sind geschlossen, d. h. der Jumper ist gesteckt!
Alle in BLAU markierten Positionen sind offen, d. h. der Jumper ist nicht gesteckt!

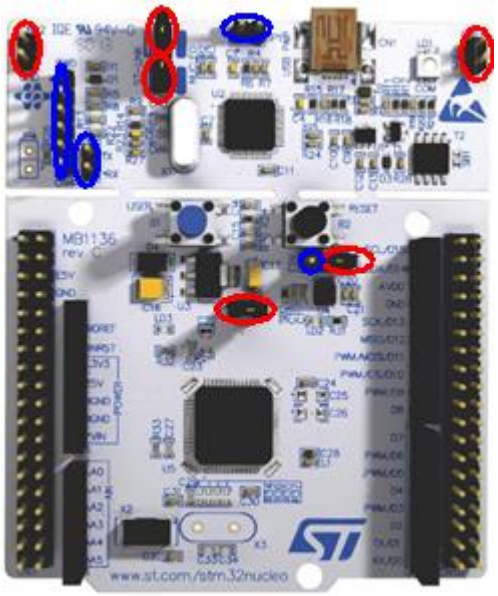

Abbildung 10: STMNucleo32-L476RG Board - Standard Jumpereinstellung

3. STMNucleo32-Baseboard – Funktionen

Dieses Kapitel erläutert die einzelnen Funktionen der Peripherie des STMNucleo32-Baseboard. Spannungsversorgungen, Jumper, Peripherie, Messpunkte und sonstiges werden beschrieben.

3.1. Spannungsversorgung

Das STMNucleo32-Baseboard und der STMNucleo32 können über unterschiedliche Spannungsquellen versorgt werden (Abbildung 11, Abbildung 12, Abbildung 13).

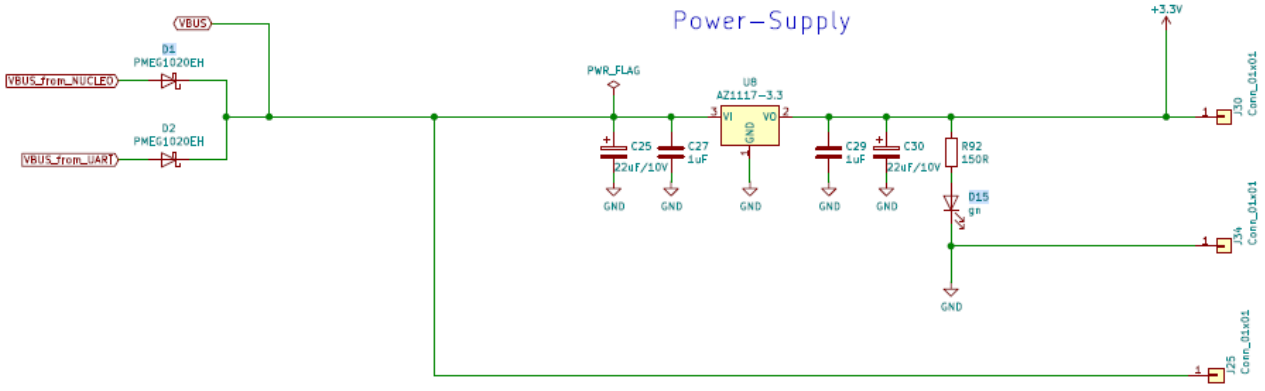

Abbildung 11: Schaltplan Spannungsversorgung 1

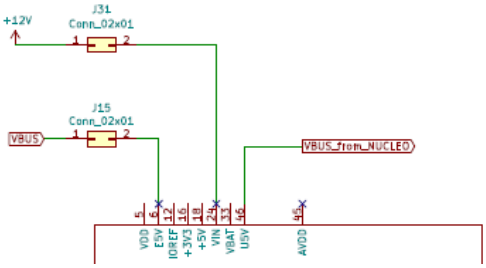

Abbildung 12: Schaltplan Spannungsversorgung 2

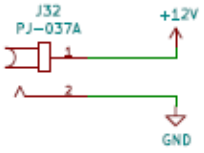

Abbildung 13: Schaltplan Spannungsversorgung 3

A. Über den Mini-USB-Anschluss des STMNucleo32 L476RG

Siehe B.

B. Über den Mini-USB-Anschluss J10 auf dem STMNucleo32-Baseboard

Die Versorgungsspannung des STMNucleo32-Baseboard mit 5V (VBUS) erfolgt über die Mini-USB-Anschlüsse A. („VBUS_from_NUCLEO“, Abbildung 11) oder B. („VBUS_from_UART“, Abbildung 11).

Je nach Spannungspegel an „VBUS_from_Nucleo“ oder „VBUS_from_UART“ wird durch die Zehnerdioden D1 und D2 entschieden, welche der beiden Mini-USB-Anschlüsse das STMNucleo32-Baseboard mit Betriebsspannung (VBUS) versorgt. Der Spannungsregler U8 erzeugt dann aus dem VBUS die 3.3V Versorgungsspannung für das STMNucleo32-Baseboard. Die LED D15 zeigt in GRÜN an, dass die 3.3V Spannung i. O. ist. Alle 3 Potentiale „VBUS“, „3-3V“ und „GND“ sind an den Messpunkten J25, J30 und J34 ausgeführt.

ACHTUNG, folgendes nur der Vollständigkeit halber, nicht zum Anwenden.
Das STMNucleo32-L476RG Board als auch das STMNucleo32-Baseboard können auch über den USB-Anschluss J10 mit Spannung versorgt werden. Dazu muss vorab der Jumper JP5 auf dem STMNucleo32-L476RG Board auf „E5V“, sowie auch der Jumper J15 auf dem STMNucleo32-Baseboard gesteckt werden.

C. Über den 12 V Anschluss (J32) auf dem STMNucleo32-Baseboard

Hier kann mit einem 5,5mm Klinkenstecker ein externes 12 V Netzteil angeschlossen werden. Die 12 V dient primär als Betriebsspannung für den Externen Lüfter.

ACHTUNG, folgendes nur der Vollständigkeit halber, nicht zum Anwenden.
Das STMNucleo32-L476RG Board als auch das STMNucleo32-Baseboard können auch über den 12V Anschluss J32 mit Spannung versorgt werden. Dazu muss vorab der Jumper JP5 auf dem STMNucleo32-L476RG Board auf „E5V“, sowie auch die Jumper J15 und J31 auf dem STMNucleo32-Baseboard gesteckt werden.

3.2. Externe Anschlüsse

Das STMNucleo32-Baseboard besitzt zum Anschließen externer Peripherie sowie Spannungsversorgung einige Anschlüsse (A-J, Abbildung 14), diesen werden im Folgenden aufgelistet und kurz beschrieben.

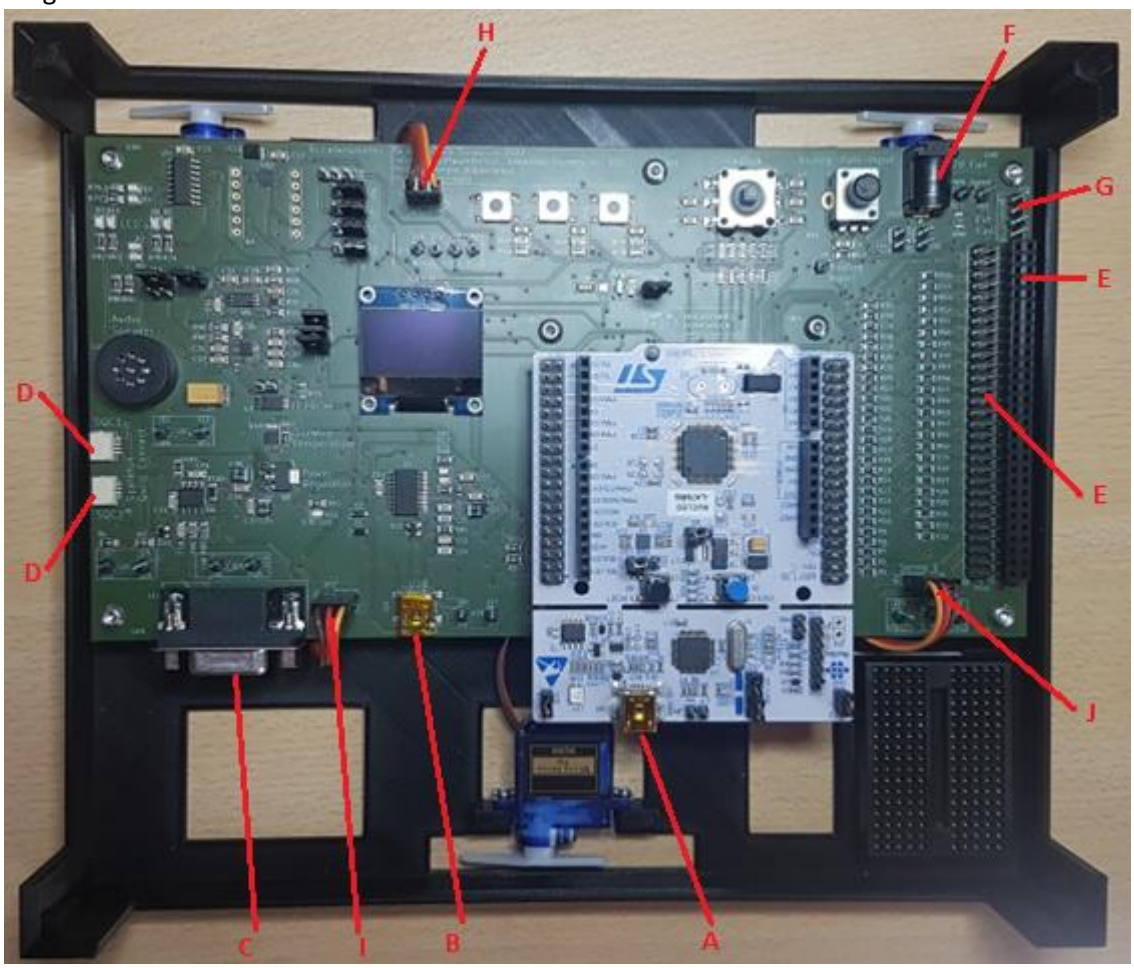

Abbildung 14: STMNucleo32-Baseboard Anschlüsse

A. Mini-USB am STMNucleo32

Dieses ist der zentrale Zugang zum STMNucleo32. Die Entwicklungsumgebung STM23CubeIDE benutzt diese Schnittstelle. Compilierte und gelinkte Programme werden über diese Schnittstelle auf das STMNucleo32 ins Flash geladen. Außerdem findet über diese Schnittstelle das Debugging mit der STM23CubeIDE statt.
Über USART2 des Microcontrollers kann diese Schnittstelle als virtueller COM Port genutzt werden.

B. Mini-USB am STMNucleo32-Baseboard

Diese Mini-USB-Schnittstelle ist als UART USB-Schnittstelle ausgeführt. Ausgabendes Mikrocontrollers über USART3 (Standardeinstellung für stdout / stdin) werden auf diese Schnittstelle geleitet und können z. B. über das Tool TeraTerm/Konsole angezeigt werden. Die Standard-Übertragungsrate liegt bei 115200 Baud (8 DataBits, No Parity, 1 StopBit).

C. CAN

Mit der CAN-Schnittstelle kann der STMNucleo32 mit anderen CAN Teilnehmer kommunizieren. Der Stecker J11 ist als Male ausgelegt.
Auf dem STMNucleo32-Baseboard ist mit R85/R86/C18 bereits eine Abschlusswiderstand verbaut (Abbildung 51).

D. Sparkfun Qwiic Connect

Das STMNucleo32-Baseboard stellt 2 getrennte Sparkfun Qwiic Connect Schnittstellen (Abbildung 15) zur Verfügung. Sparkfun Qwiic Connect ist eine Steckverbindung für I²C Protokoll.

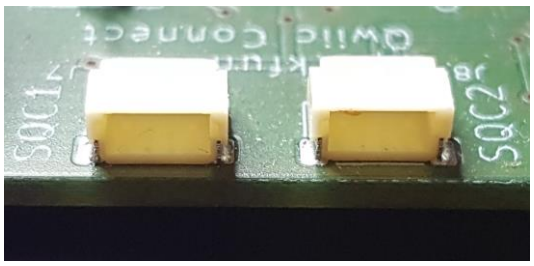

Abbildung 15: Sparkfun Qwiic Connect

E. 50 Pin-Erweiterungsstecker zweireihiger Stift-/Buchsenleiste männlich/weiblich (J2/J3)

Beide Leisten sind parallel ausgeführt. D. h. dass so ein einfacher Anschluss zusätzlicher Peripherie oder Messpunkte an bestimmte STMNucleo32-Pins einfach per Kabelanschluss möglich ist. Jeder Pin ist mit einem Schutzwiederstand (R5-R55) abgesichert (Abbildung 16, Abbildung 17). Trotzdem ist hier höchste Vorsicht einzuhalten, um das STMNucleo32 als auch das STMNucleo32-Baseboard nicht zu beschädigen!
Auf dem STMNucleo32-Baseboard ist ein Steckbrett verbaut, so kann mittels Drahtbrücken eine Verbindung vom Steckbrett (Messpunkte, zusätzliche Sensoren/Aktoren, …) eine sichere Verbindung zu J2/J3 hergestellt werden.

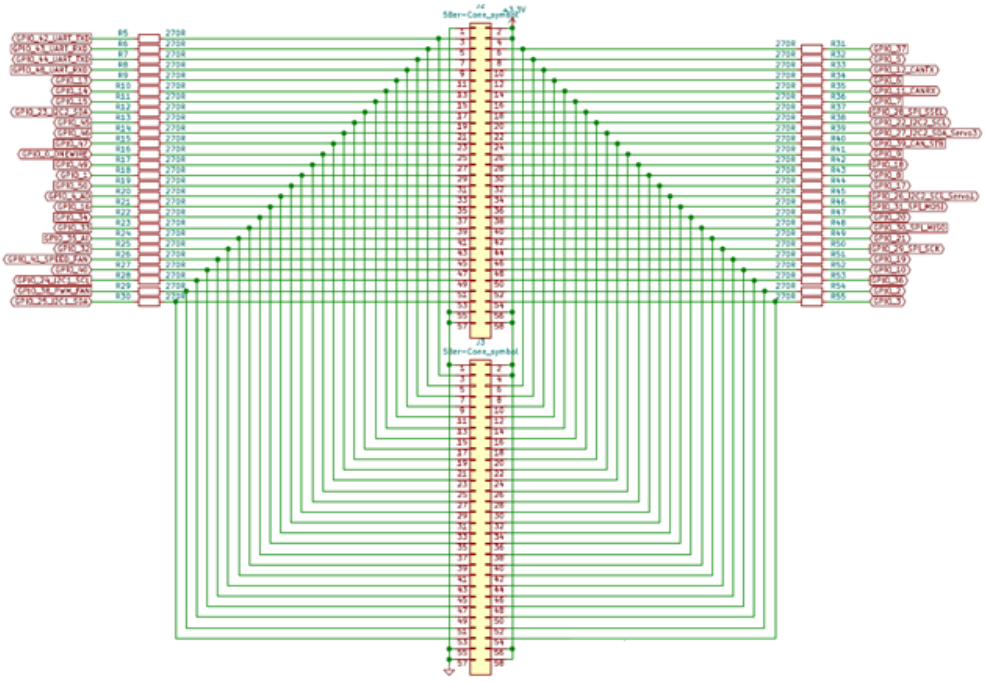

Abbildung 16: Schaltplan 50 Pol. Leisten

Nachfolgend eine Auflistung der einzelnen Pins an J2/J3:

Benutzt J/N	Bezeichnung	Pin-Nr.	Pin-Nr.	Bezeichnung	Benutzt J/N
	GND	1	2	3.3V	
J	GPIO_42_UART_TXD	3	4	3.3V	
J	GPIO_43_UART_RXD	5	6	GPIO_37	N
N	GPIO_44_UART_TXD	7	8	GPIO_5	J
N	GPIO_48_UART_RXD	9	10	GPIO_12_CANTX	J
N	GPIO_13	11	12	GPIO_6	J
N	GPIO_14	13	14	GPIO_11_CANRX	J
N	GPIO_15	15	16	GPIO_7	J
J	GPIO_23_I2C2_SDA	17	18	GPIO_28_SPI_SSEL	J
J	GPIO_45	19	20	GPIO_22_I2C2_SCL	J
N	GPIO_46*	21	22	GPIO_27_I2C2_SDY_Servo3	J
N	GPIO_47*	23	24	GPIO_39_CAN_STB	J
J	GPIO_0_ONEWIRE	25	26	GPIO_9	J
J	GPIO_49	27	28	GPIO_18	J
J	GPIO_1	29	30	GPIO_8	J
J	GPIO_50	31	32	GPIO_17	N
J	GPIO_4_AO	33	34	GPIO_26_I2C2_SCL_Servo1	J
J	GPIO_16	35	36	GPIO_31_SPI_MOSI	J
N	GPIO_34	37	38	GPIO_20	N
J	GPIO_33	39	40	GPIO_30_SPI_MISO	J

J	GPIO_35_AI	41	42	GPIO_21	N
J	GPIO_32	43	44	GPIO_29_SPI_SCK	J
J	GPIO_41_SPEED_FAN	45	46	GPIO_19	N
N	GPIO_40	47	48	GPIO_10	J
J	GPIO_24_I2C1_SCL	49	50	GPIO_36	J
J	GPIO_38_PWM_FAN	51	52	GPIO_2*	N
J	GPIO_25_I2C1_SDA	53	54	GPIO_3*	N
	GND	55	56	3.3V	
	GND	57	58	3.3V	

Abbildung 17: Beschreibung - 50 Pol. Leisten

*Dieser Pin wird auf dem STM32Nucleo-L476RG intern verwendet und nicht herausgeführt. Er kann somit nicht verwendet werden.

F. 12V Spannungsversorgung

Hier kann mit einem 5,5mm Klinkenstecker ein externes 12 V Netzteil angeschlossen werden. Die 12 V dient primär als Betriebsspannung für den Externen Lüfter.

G. Externer Lüfter

Abbildung 18: Externer Lüfter

Der externe Lüfter (Abbildung 18) bietet die Möglichkeit für regelungstechnische Experimente. Über ein PWM Signal (Abbildung 58) kann die Soll-Drehzahl in einem Bereich gesteuert werden. Über einen weiteren Pin (Abbildung 58) kann der STMNucleo32 die aktuelle Ist-Drehzahl ermitteln und somit eine Regelung realisieren.

H. Modellbau-Servo 1

Abbildung 19: Modellbau Servo

Siehe J.

I. Modellbau-Servo 2

Siehe J.

J. Modellbau-Servo 3

Die 3 anschließbaren Modellbau-Servos (JP1, JP2, JP3, Abbildung 19) bieten die Möglichkeit, das STMNucleo32-Baseboard über die Servos auszurichten. Z. B. ein Experiment mit dem Beschleunigungssensor und den 3 Modellbau-Servos ermöglicht eine Lagereglung des gesamten STMNucleo32-Baseboards (siehe Kapitel 3.18).

3.3. Messpunkte

Auf dem STMNucleo32-Baseboard sind einige Messpunkte (Abbildung 20) ausgeführt.
ACHTUNG bei der Verwendung der Messpunkte ist höchste Sorgfalt wichtig, keine Kurzschlüsse verursachen!
Mit einem Oszilloskop (z. B. PicoScope) oder sonstigem Messgerät, können Spannungspegel Protokolle aufgezeichnet werden (diese sind z. B. Seriell (TXD/RXD), USB, I²C, SPI, CAN).

Messpunkte	Beschreibung	Bemerkung
J34	GND	Spannungsversorgung
J30	3.3V	
J25	VBUS – 5V	
J26	USB D+	USB
J27	USB D-	
J38	USART3 TXD	
J39	USART3 RXD	
J23	I²C1 SCL	I²C1
J22	I²C1 SDA	
J17	I²C2 SCL	I²C2 PWM Servo 1, 3
	PWM Servo 1	
J24	I²C2 SDA	
	PWM Servo 3	
J19	SPI MISO	SPI
J18	SPI MOSI	
J20	SPI SCK	
J21	SPI SSEL	
J29	CAN High	CAN
J28	CAN Low	
J5	Externer Lüfter GND	
J1	Externer Lüfter Ist-Dehzahl	**ACHTUNG**, auf dem STMNucleo32-Baseboard, falsch mit PWM beschriftet!
J33	Externer Lüfter PWM Soll-Drehzahl	**ACHTUNG**, auf dem STMNucleo32-Baseboard, falsch mit Speed beschriftet!
	PWM Servo 2	
J37	D3 LED rot	Kann zum Messen von z. B. Reaktionszeiten vom Programm bei einem Interrupt-Trigger auf J36 verwendet werden.
J36	S1 Joystick rechts	Interrupt Eingang
J14.2	Audio Analog Input für den Verstärker LM4811	Abbildung 45
J12.4		
J12.1	Audio Analog Output vom STMNucleo32	Abbildung 45 (vor Tiefbassfilter)
J12.3		

Abbildung 20: Messpunkte

3.4. Taster

Das STMNucleo32-Baseboard bietet 3 SMD Taster (SW1, SW2, SW3, Abbildung 22) für die Programmsteuerung an.
Diese sind über folgende GPIO's direkt mit dem STMNucleo32 verbunden (Abbildung 21):

SW1 GPIO_1
SW2 GPIO_49
SW3 GPIO_50

Bei der Verwendung ist darauf zu achten, dass die Taster über Pull-Up-Widerstände (R59, R60, R61) mit 3.3V verbunden sind. D. h. bei nicht betätigtem Taster, liegen 3.3V am Eingang des STMNucleo32 und bei betätigtem Taster GND.
In der STM23CubeIDE können die Taster anhand der Klasse „DigitalInOut" im Namespace Platform::BSP benutzt werden.

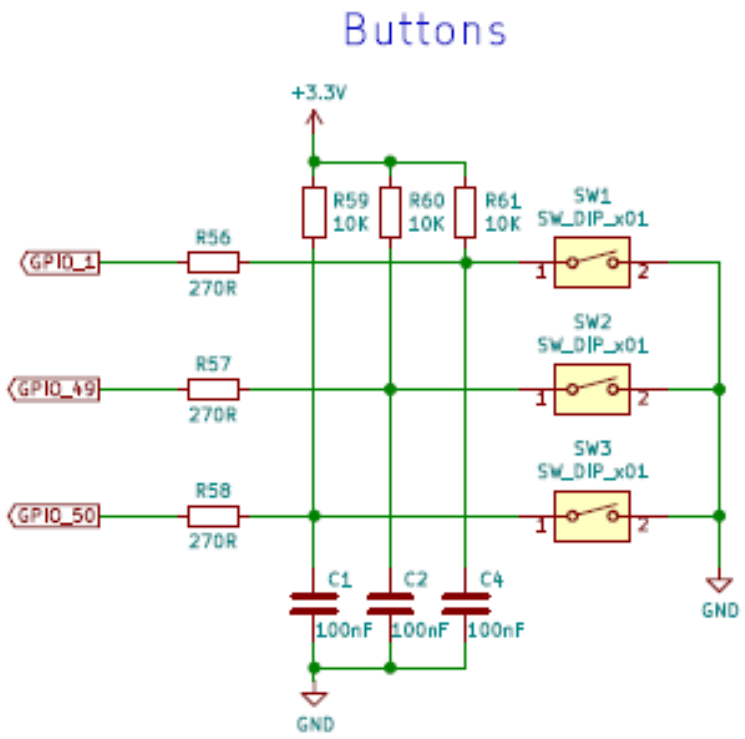

Abbildung 21: Schaltplan - Taster

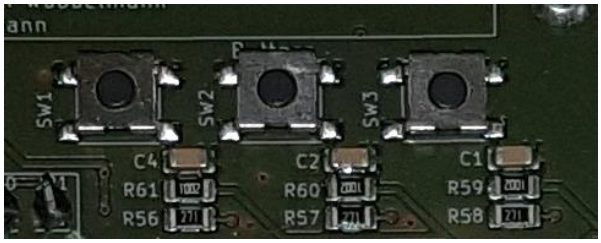

Abbildung 22: Taster

Zusätzlich zu den Tastern auf dem STMNucleo32-Baseboard besteht die Möglichkeit den blauen Taster B1 auf dem STMNucleo32 zu verwenden (Abbildung 23). Dieser ist über einen Pull-Up-Widerstand an den GPIO_45 angeschlossen.
Der STMNucleo32 Eingang GPIO_45 ist bei nicht betätigtem Taster somit 1.

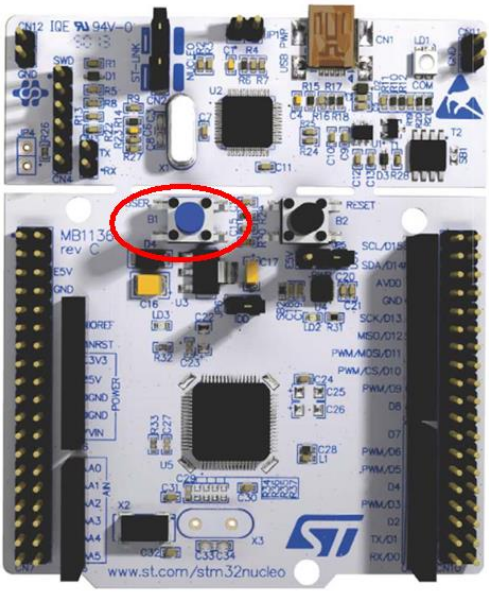

Abbildung 23: Taster - STMNucleo32

3.5. Joystick

Das STMNucleo32-Baseboard bietet einen Joystick (S1) für die Programmsteuerung an (Abbildung 25).

Dieser ist über folgende GPIO's direkt mit dem STMNucleo32 verbunden (Abbildung 24):

S1-links	left	GPIO_6
S1-drücken	push	GPIO_7
S1-hoch	up	GPIO_8
S1-rechts	right	GPIO_9
S1-runter	down	GPIO_10

Bei der Verwendung ist darauf zu achten, dass die Taster über Pull-Up-Widerstände (R68, R69, R70, R71, R72) mit 3.3V verbunden sind. D. h. bei nicht betätigtem Joystick-Taster, liegen 3.3V am Eingang des STMNucleo32 und bei betätigtem Taster GND.

In der STM23CubeIDE können die Taster anhand der Klasse „DigitalInOut" im Namespace Platform::BSP benutzt werden.

Der Joystick-Pin S1-rechts / GPIO_9 ist zusätzlich für Messzwecke als Messpunkt J36 ausgeführt.

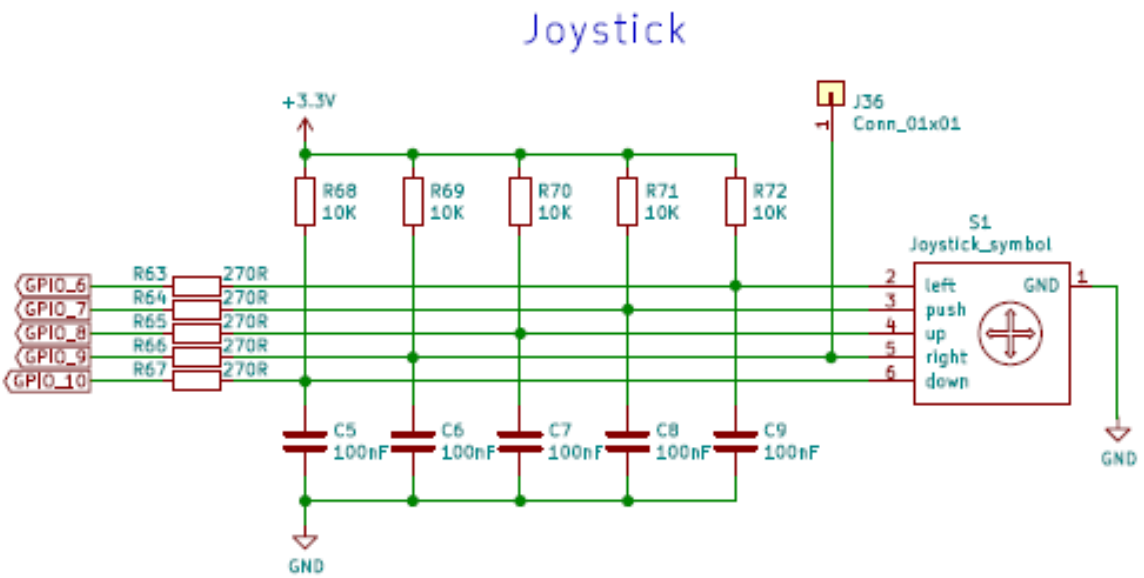

Abbildung 24: Schaltplan - Joystick

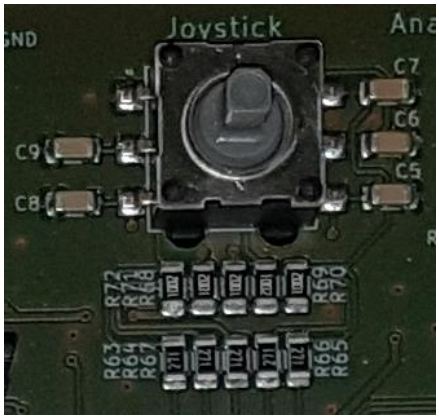

Abbildung 25: Joystick

3.6. Trimmer / Poti

Mit dem Trimmer/Poti (RV1, Abbildung 27) wird dem STMNucleo32-Baseboard an dem analogen Eingang GPIO_35 ein analoger Eingangswert angeboten.
Am Eingang GPIO_35 liegt somit (je nach Stellung des Trimmers) eine Eingangsspannung zwischen 0 und 3.3 V an (Abbildung 26).
Dieser Wert wird für Messzwecke zusätzlich als Messpunkt J4 ausgeführt.
In der STM23CubeIDE kann der Analogeingangswert mit der Klasse „Adc“ im Namespace Platform::BSP ausgelesen werden.

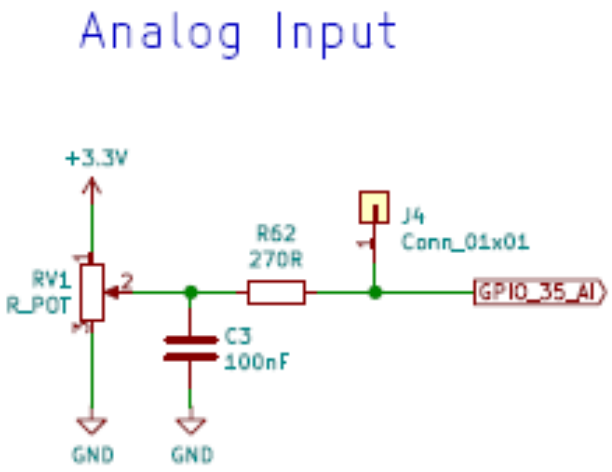

Abbildung 26: Schaltplan - Analog Input / Trimmer / Poti

Abbildung 27: Analog Input / Trimmer / Poti

3.7. Lichtsensor

Der Lichtsensor LTR303ALS (U3, Abbildung 29) ist den Bus I²C1 an den STMNucleo32 angeschlossen (Abbildung 28).
Der Lichtsensor kann die Lichtintensität am Sensor messen. Diese wird durch das Umgebungslicht beeinflusst, kann aber auch ggf. zusätzlich durch die direkt daneben liegende LED D16 gesteuert werden. Die LED D16 wiederum kann durch PWM helligkeitsgesteuert arbeiten.
In der STM23CubeIDE kann dieser Baustein über das I²C-Interface angesprochen werden. Dafür kann die Klasse LTR303ALS01 im Namespace Platform::BSP benutzt werden.
Der LTR303ALS kann mit seinem INT Ausgang einen Interrupt am GPIO_36 auslösen (näheres siehe Datenblatt zum LTR303ALS).

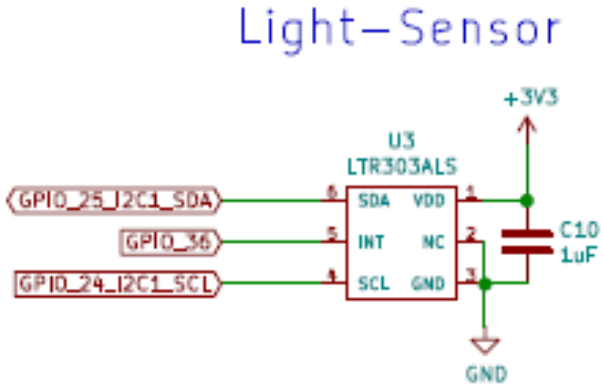

Abbildung 28: Schaltplan - Lichtsensor

Abbildung 29: Lichtsensor

3.8. LED liegend

Die “LED liegend” ist an dem GPIO_38 angeschlossen (Abbildung 30). Mit dem Stecken des Jumpers J35 kann die LED aktiviert werden. Dann ist sie parallel zu der Servo 2, sowie dem externen Lüfter angeschlossen.
D. h. wird der externe Lüfter oder der Servo 2 mit einer PWM betrieben, zeigt die liegende LED hier eine entsprechende Helligkeit an.
Generell ist die “LED liegend” direkt neben dem Lichtsensor (siehe Kapitel 3.7und Abbildung 31) auf dem STMNucleo32-Baseboard verortet. So besteht die Möglichkeit die Helligkeit der LED (über PWM), für die Auswertung des Lichtsensors zu verwenden.
Am Messpunkt J33 kann das Ansteuersignal der liegenden LED gemessen werden.
In der STM23CubeIDE kann die LED anhand der Klasse „DigitalInOut“ im Namespace Platform::BSP benutzt werden. Für eine PWM Ansteuerung ist der GPIO_38 als PWM zu betreiben (mit TIMER, PWM Klasse im Namespace Platform::BSP, …).

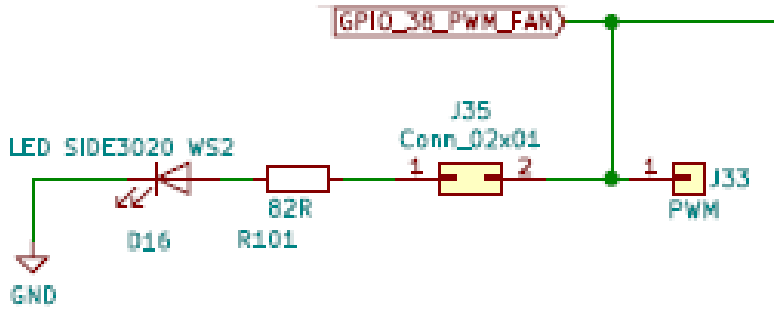

Abbildung 30: Schaltplan - LED liegend

Abbildung 31: LED liegend

3.9. Beschleunigungssensor

Auf dem STMNucleo32-Baseboard ist der Beschleunigungssensor U12 LIS30H verbaut (Abbildung 33)..

Der LIS30H kann mit dem I²C-Bus als auch mit SPI-Bus betrieben werden (Abbildung 32 Für SPI müssen die Jumper JP4, JP5, JP6 und JP7 auf SPI (rechts in Abbildung 32) gesteckt werden. Soll der LIS30H über I²C betrieben werden, findest dieses über den I²C1 -Bus statt. Dafür müssen die Jumper J4, J5, J6 und J7 auf I²C (links in Abbildung 32) gesteckt werden. Dabei muss der Jumper JP8 zum Adressieren auf GND (links) oder 3.3V (rechts) gesteckt werden.
Im Normalfall soll der Beschleunigungssensor mit SPI betrieben werden.
In der STM23CubeIDE kann der Beschleunigungssensor anhand der Klasse „LIS3DH“ (SPI Modus) oder „LIS3DHI2C“ (I2C Modus) im Namespace Platform::BSP angesprochen werden.

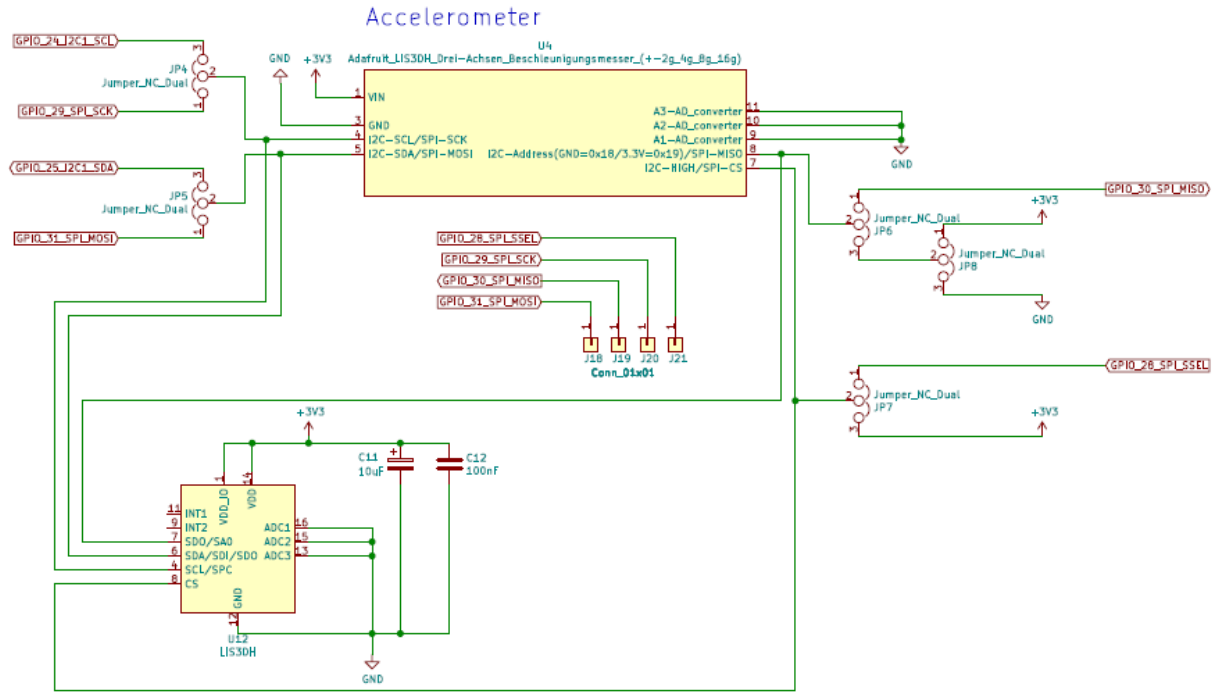

Abbildung 32: Schaltplan - Beschleunigungssensor

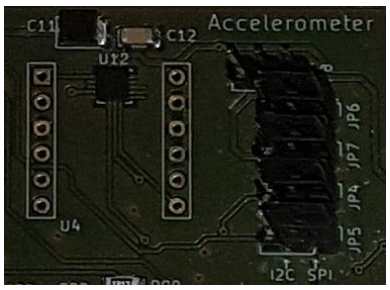

Abbildung 33: Beschleunigungssensor

3.10. LED's

Auf dem STMNucleo32-Baseboard stehen 8 LED's in 4 Farben zur Verfügung (Abbildung 35). Die 8 LED's werden über den I²C-Bus mit dem Baustein PCA9531 als Byte (8 Bits ➔ 8 LED's) angesprochen (Abbildung 34). Die LED's sind als „Kreuz" angeordnet (Abbildung 35). Oben senkrecht 2 LED's D3 und D4 in Rot, rechts waagerecht 2 LED's D5 und D6 in Gelb, unten senkrecht 2 LED's D7 und D8 in Grün und links waagerecht 2 LED's D9 und D10 in Blau.
Am Messpunkt J37 kann der Zustand der LED D3, zum Messen (mit dem Oszilloskop) von z. B. Reaktionszeiten des Programms, benutzt werden.
In der STM23CubeIDE können die 8 LED's (im Block als Byte, wobei das Bit 2^0 der LED D3 entspricht) mit der Klasse „SR74LVC595" im Namespace Platform::BSP angesprochen werden.
Mit dem RESET-Pin des PCA9531 kann über den GPIO_18 eine Reset auf die LED Ausgänge gegeben werden (näheres siehe Datenblatt zum PCA9531).

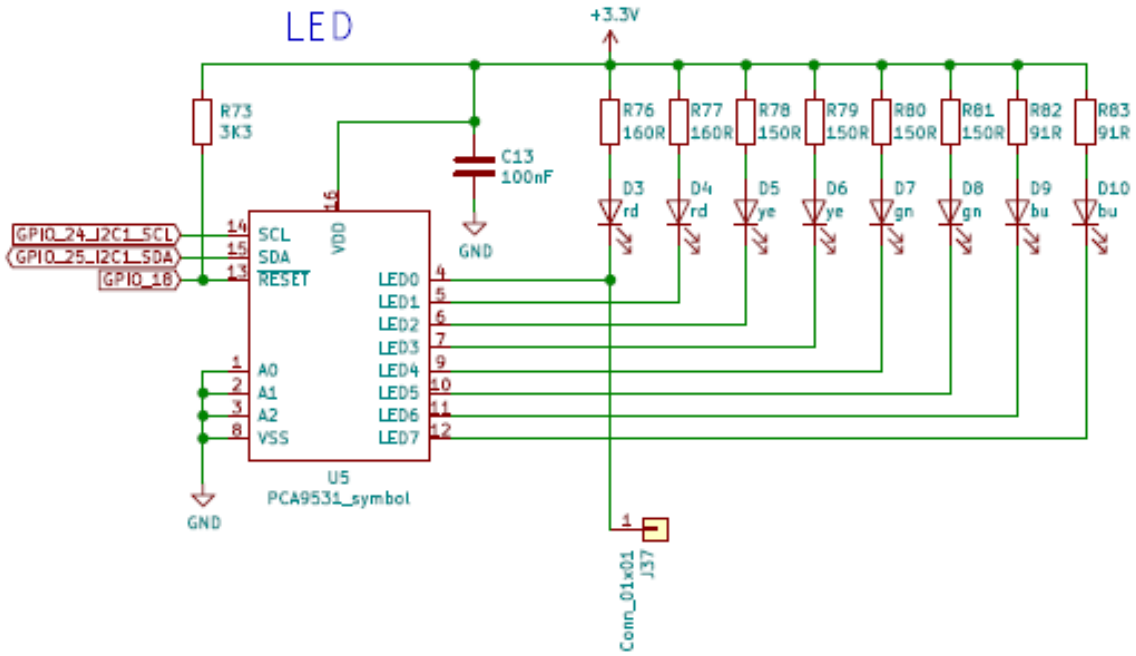

Abbildung 34: Schaltplan - 8-LED's

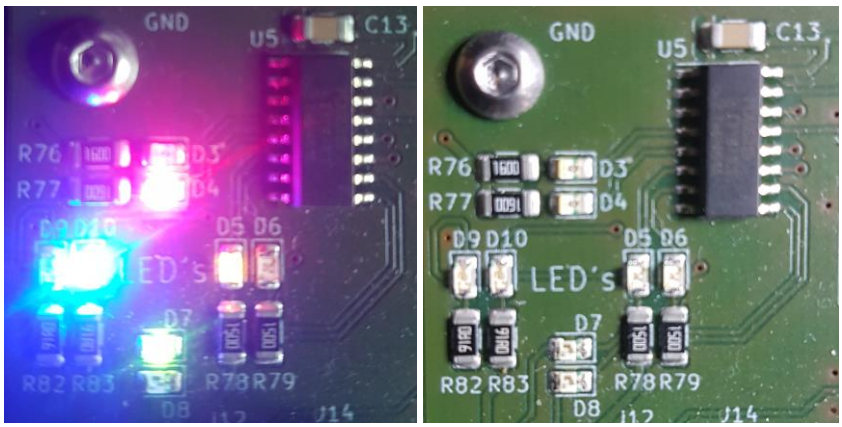

Abbildung 35: 8-LED's

Zusätzlich zu den LED’s auf dem STMNucleo32-Baseboard besteht die Möglichkeit die LED LD2 (grün) auf dem STMNucleo32 zu verwenden (Abbildung 36).
Die LED LD2 kann über PA5 bzw. GPIO_5 angesprochen werden.

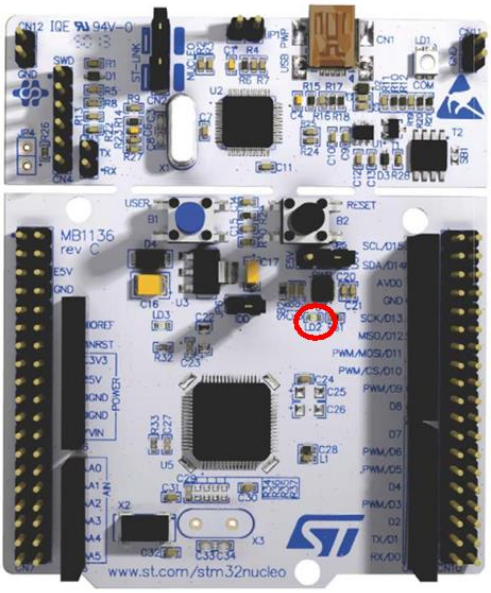

Abbildung 36: LED - STMNucleo32

3.11. EEPROM

Das STMNucleo32-Baseboard bietet die Möglichkeit ein EEPROM vom Typ 24LC08 zu benutzen (Abbildung 38). Dieses wird über den I²C2-Bus angesprochen.
Der Anschluss WP am Baustein 24LS08 steht für „write protected“, d. h. ist der Jumper J6 gesteckt liegt GND am WP Eingang des Bausteins und dieser ist dann auf Schreibschutz eingestellt (Abbildung 37).
In der STM23CubeIDE kann das EEPROM anhand der Klasse „MC24LC08“ im Namespace Platform::BSP angesprochen werden.

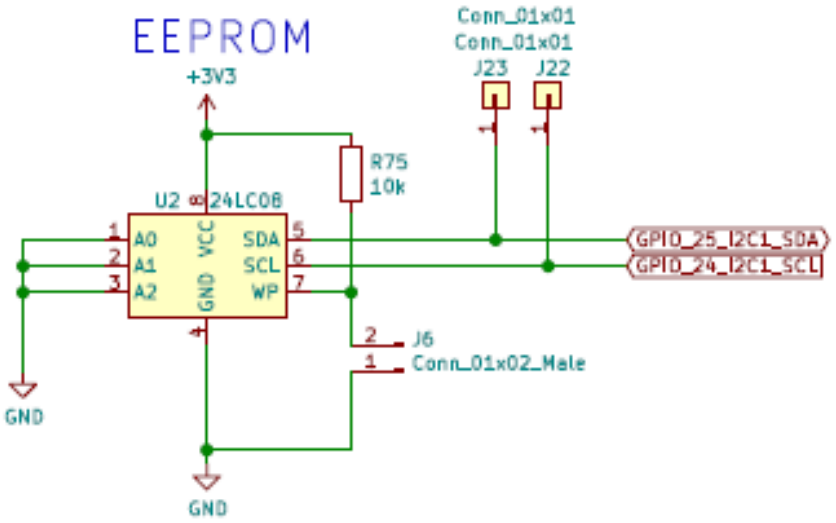

Abbildung 37: Schaltplan - EEPROM

Abbildung 38: EEPROM

3.12. OneWire Temperatursensor

Das STMNucleo32-Baseboard stellt einen OneWire Baustein vom Typ DS18B20U zur Verfügung (Abbildung 40). Dieser Baustein wird anhand des GPIO_0 über das OneWire-Protokoll angesprochen (Abbildung 39). Der Baustein liefert die Sensor-Umgebungstemperatur sowie eine eindeutige 48-Bit Seriennummer.
Weitere Informationen zum OneWire-Protokoll sind ausreichend im www zu finden.

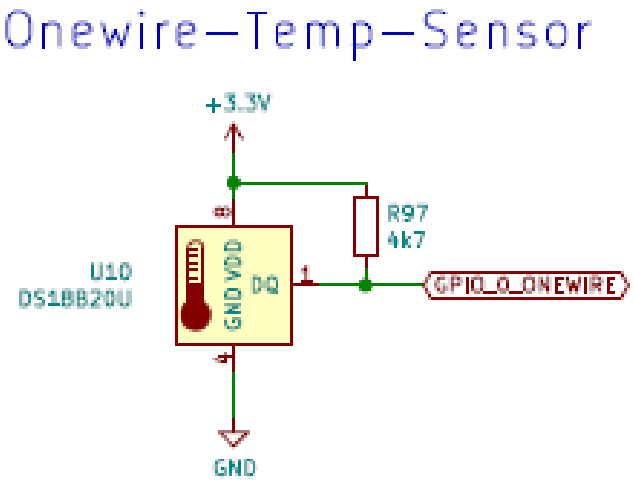

Abbildung 39: Schaltplan - OneWire

Abbildung 40: OneWire

3.13. OLED

Mit dem STMNucleo32-Baseboard wird ein OLED SSD1306 zur Verfügung gestellt. Das monochrome OLED hat eine Auflösung von 128x64 Pixel (Abbildung 42). Angesprochen wird dieses über den I²C1-Bus (Abbildung 41).
In der STM23CubeIDE kann das OLED anhand der Klasse „SSD1306“ im Namespace Platform::BSP angesprochen werden.

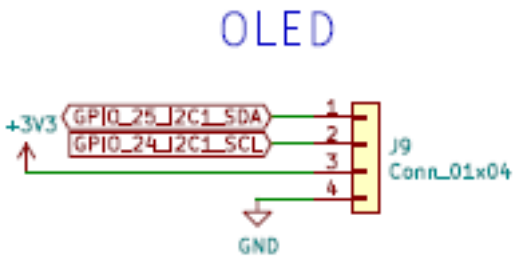

Abbildung 41: Schaltplan - OLED

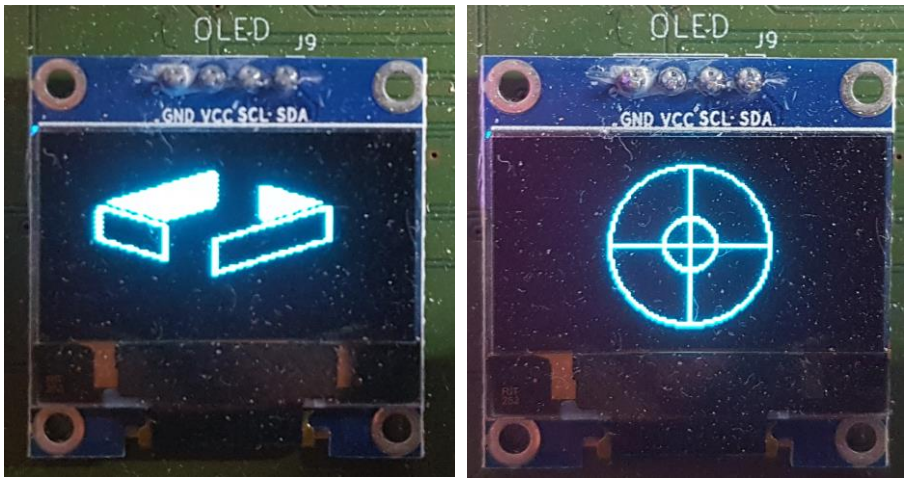

Abbildung 42: OLED

3.14. Lautsprecher / Audio

Zur Sound-Ausgabe ist auf dem STMNucleo32-Baseboard ein Lautsprecher LS1 (CVS1508) verbaut. Dieser wird über den Verstärkerbaustein LM4811 U11 angesteuert (Abbildung 45). Das Eingangssignal am Verstärker liefert der Analoge Ausgang GPIO_AO (Abbildung 45) des STMNucleo32.

Die Schaltung bietet generell zwei Möglichkeiten für den Betrieb.

A) Mit der Jumpereinstellung (Jumperbrücke J12 Pin3-4, rot in Abbildung 43) an J12, wird das Ausgangssignal (GPIO_AO) des STMNucleo32 direkt (ohne weitere Filter) auf den Verstärker-Eingang des LM4811 geleitet.

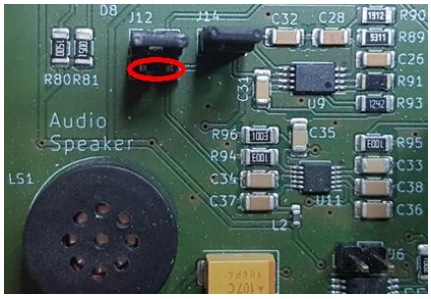

Abbildung 43: Lautsprecher Direkt

B) Es besteht die Möglichkeit das Ausgangssignal (GPIO_AO) des STMNucleo32 mit einem 2-stufigen Tiefpassfilter zu filtern. Der Filter ist so ausgelegt, dass Frequenzen oberhalb der Grenzfrequenz des CVS1508 (ca. 20KHz), nicht zum Verstärker LM4811 und somit zum Lautsprecher LS1 gelangen.

Dafür sind die 2 Jumper J14 sowie die Jumperbrücke J12 Pin1-2 (rot in Abbildung 44) zu stecken. Das ist die Standardeinstellung des STMNucleo32-Baseboards.

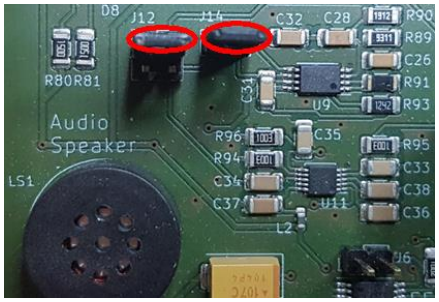

Abbildung 44: Lautsprecher Filter

Der Verstärker LM4811 kann über 3 Steuereingänge verstellt werden.

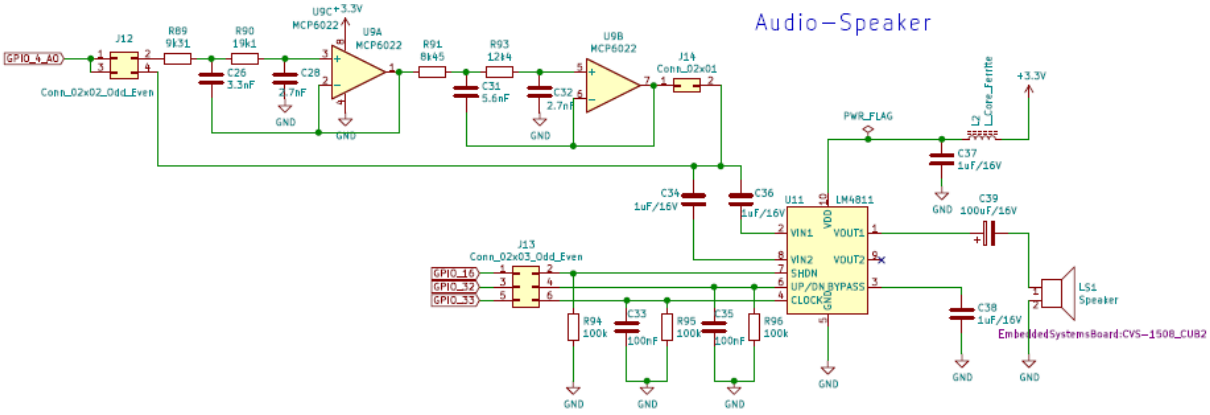

Abbildung 45: Schaltplan - Lautsprecher

Zum Einstellen der Verstärkung des Verstärkers ist das Taktsignal „CLOCK“ erforderlich. Dieses Taktsignal liefert der STMNucleo32 über den GPIO_33. Mit jeder ansteigenden Flanke des Taktsignals, wird die Verstärkung um einen 3-dB-Schritt, in Abhängigkeit vom logischen Spannungspegel am Pin UP/DN des LM4811, erhöht oder verringert.
Der CLOCK-Pin des LM4811 kann mit dem Jumper J13 Pin5-6 (Abbildung 46) aktiviert werden. Ist der Jumper gezogen, wird der CLOCK-Pin über den Pull-Down-Widerstand konstant auf GND (Low-Pegel) gesetzt und ist nicht aktiv.

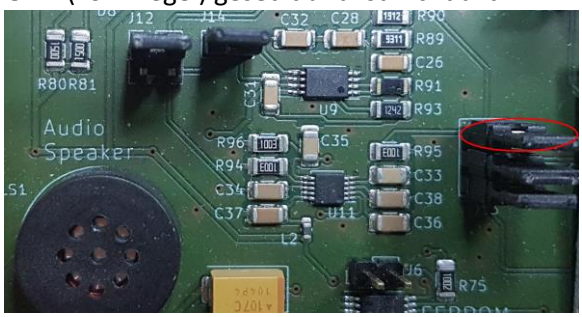

Abbildung 46: LM4811 - CLOCK

Ein hoher Spannungspegel am UP/DN-Pin bewirkt, dass die Verstärkung bei jeder steigenden Flanke des Taktsignals um 3 dB erhöht wird. Ein niedriger Spannungspegel bewirkt, dass die Verstärkung bei jeder steigenden Flanke des Taktsignals um 3 dB verringert wird.
Insgesamt gibt es 16 Verstärkungseinstellungen, von maximal +12 dB bis minimal -33 dB.
Beim Einschalten der Elektronik wird die Verstärkung des Verstärkers auf einen Standardwert von 0 dB eingestellt.
Der UP/DN-Pin des LM4811 kann mit dem Jumper (Abbildung 47) aktiviert werden. Ist der

Jumper gezogen, wird der UP/DN-Pin des LM4811 über den Pull-Down-Widerstand konstant auf GND (Low-Pegel) gesetzt und ist nicht aktiv.

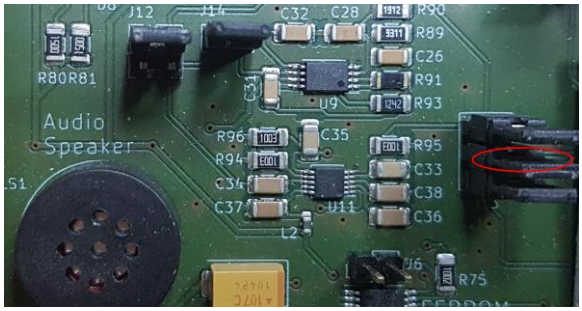

Abbildung 47: LM4811 - UP/DN

Aktivieren und deaktivieren kann man den Verstärker über den SHDN-Pin des Verstärkers. Der SHDN-Pin des LM4811 ist mit dem GPIO_16 des STMNucleo32 verbunden, wodurch die Software die Möglichkeit hat den Verstärker zu steuern.
Der SHDN-Pin des LM4811 kann mit dem Jumper (Abbildung 48) aktiviert werden.

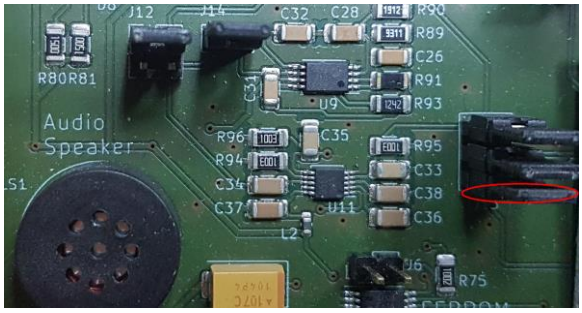

Abbildung 48: LM4811 - SHDN

Ist der Jumper gezogen, wird der SHDN-Pin des LM4811 über den Pull-Down-Widerstand konstant auf GND (Low-Pegel) gesetzt und ist nicht aktiv, der Verstärker ist dann dauerhaft aktiv.
Wird der Shutdown-Modus wieder verlassen, wird beim LM4811 die vorherige Verstärkung wieder eingestellt.
In der STM23CubeIDE kann der Verstärker mit der Klasse „LM4811“ im Namespace Platform::BSP angesprochen werden.

3.15. Sparkfun Qwiic Connect

Das STMNucleo32-Baseboard stellt 2 getrennte Sparkfun Qwiic Connect-Schnittstellen (Abbildung 50) zur Verfügung. Sparkfun Qwiic Connect ist eine Steckverbindung für den I²C Bus.

ACHTUNG, im I²C Bus dürfen Teilnehmer-Adressen nicht doppelt verwendet werden.

Der in Abbildung 50 links dargestellte Sparkfun Qwiic Connect J7, ist am I²C1 Bus (Abbildung 49)des STMNucleo32 angeschlossen. Da hier bereits einige andere Bausteine auch dem STMNucleo32-Basebord angeschlossen sind, ist bei Verwendung im Besonderen auf die Adressierung zu achten.

Der in Abbildung 50 rechts dargestellte Sparkfun Qwiic Connect J8, ist am I²C2 Bus (Abbildung 49) des STMNucleo32 angeschlossen. Soll dieser Anschluss verwendet werden, darf der Servo 3 an JP3 nicht gesteckt sein, da die Leitung „GPIO_27_I2C2_SDA_Servo3“ auch für das PWM-Signal des Servo 3 verwendet wird.

D. h. soll über Qwiic Connect zusätzliche Peripherie angeschlossen werden ist der J8 Anschluss, bei nicht gestecktem Servo 3, zu empfehlen.

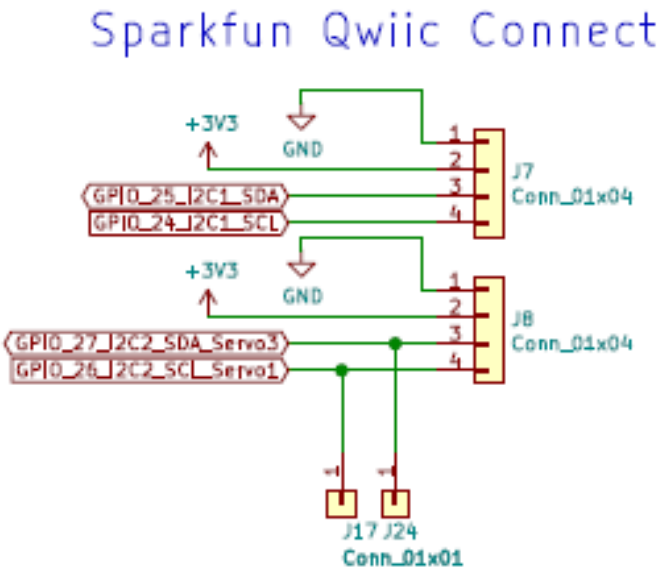

Abbildung 49: Schaltplan - Sparkfun Qwiic Connect

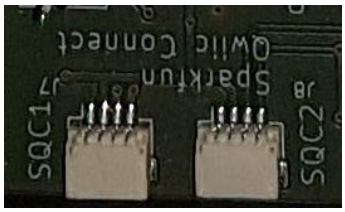

Abbildung 50: Sparkfun Qwiic Connect

3.16. CAN

Mit der CAN-Schnittstelle kann mit anderen CAN Teilnehmer kommuniziert werden (Abbildung 52). Der Sub-D Stecker J11 ist als männlich/male ausgeführt.
In der STM23CubeIDE kann die CAN-Schnittstelle anhand der Klasse „CAN“ im Namespace Platform::BSP angesprochen werden.

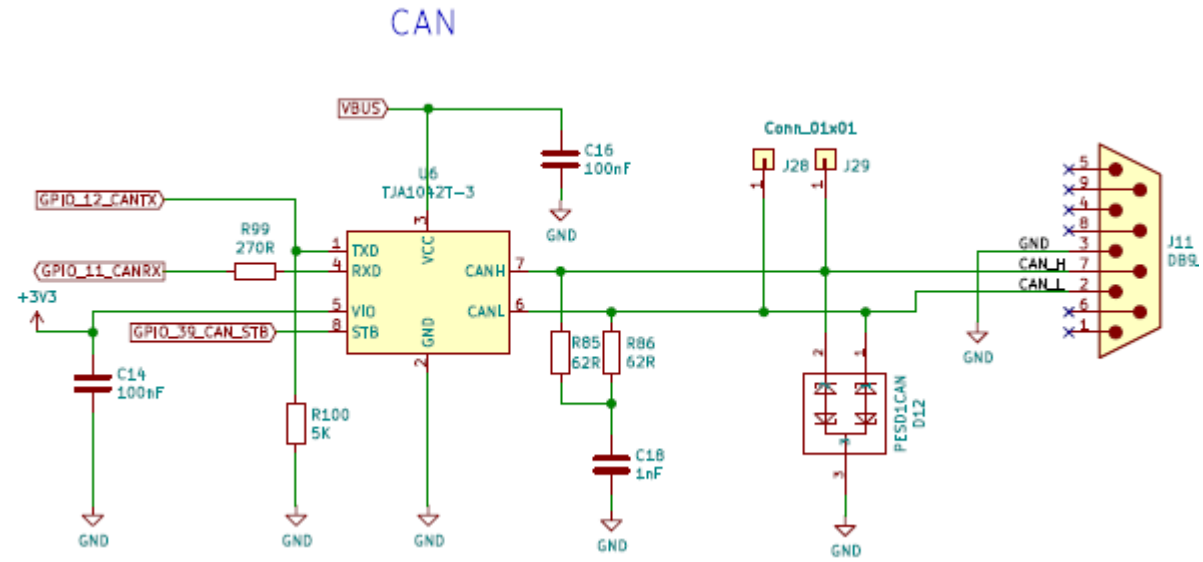

Abbildung 51: Schaltplan - CAN

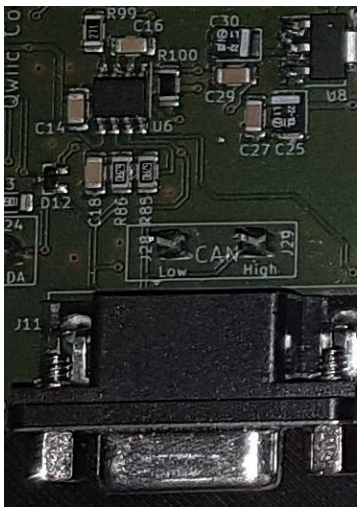

Abbildung 52: CAN

Besondere Jumpereinstellungen werden hier nicht benötigt. An den Messpunkten J29 und J28 (CAN-High / CAN-Low) kann das Protokoll zu Diagnosezwecken mit dem Oszilloskop aufgezeichnet werden (Abbildung 51).
Auf dem STMNucleo32-Baseboard ist mit R85/R86/C18 bereits eine Abschlussimpedanz verbaut (Abbildung 51).

3.17. Mini-USB / UART

Diese Mini-USB-Schnittstelle J10 ist als UART FT232RL USB-Schnittstelle ausgeführt (Abbildung 53). Ausgaben vom STM23CubeIDE mit printf oder cout werden auf diese Schnittstelle (Abbildung 54) geleitet und können z. B. über das Tool TeraTerm/Konsole angezeigt werden. Die Übertragungsrate liegt hier bei 115200 Baud (8 DataBits, No Parity, 1 StopBit). Voraussetzung hierfür ist, dass in der Datei Core/Src/retarget.c USART3 als Schnittstelle für die _write und _read Funktionen ausgewählt ist.
An den Messpunkten J26/J27 (USB_D+/USB_D-) kann mit einem Oszilloskop (z. B. PicoScope) das Protokoll der USB-Schnittstelle aufgezeichnet werden.
An den Messpunkten J38/J39 (UART_TxD/UART_RxD) kann mit einem Oszilloskop (z. B. PicoScope) das UART Protokoll des STMNucleo32 aufgezeichnet werden.
Findet eine Ausgabe über diese Schnittstelle an ein Terminal (z. B. TeraTerm) statt, wird mit den LED's D13 und D14 der Zustand (lesen RX / schreiben TX) der seriellen UART Leitung angezeigt.
Mit dem Anschluss des J10, kann das STMNucleo32-Baseboard ggf. auch mit Spannung versorgt werden, dabei ist unbedingt Kapitel 3.1 zu beachten.

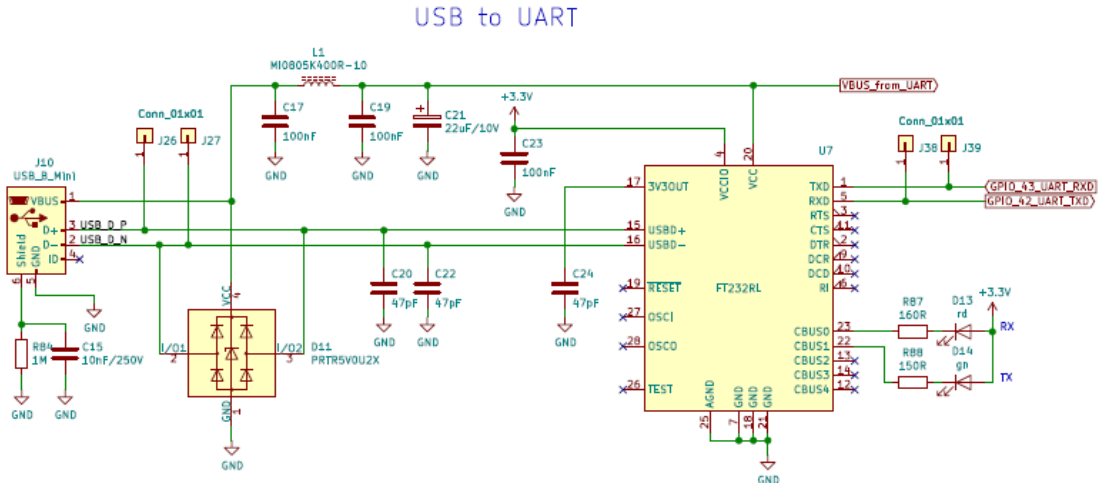

Abbildung 53: Schaltplan - Mini-USB / UART

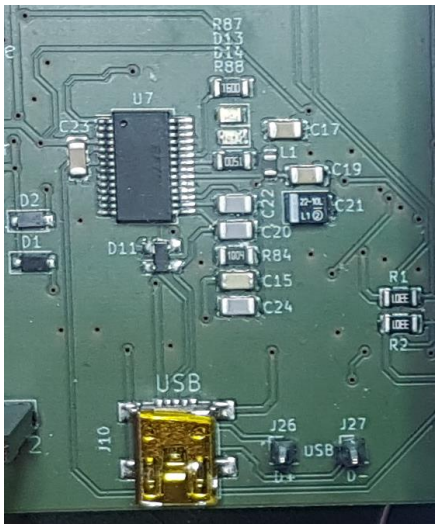

Abbildung 54: Mini-USB / UART

3.18. Modellbau-Servos

Die 3 anschließbaren Modellbau-Servos an JP1, JP2 und JP3 (Abbildung 55, Abbildung 14 B, F und H), bieten z. B. die Möglichkeit das STMNucleo32-Baseboard über die Servos auszurichten. Ein Experiment im Zusammenspiel vom dem Beschleunigungssensor und den 3 Modellbau-Servos, ermöglicht eine Lageregelung des gesamten STMNucleo32-Baseboards. Aber auch für andere Experimente sind die Servos einsetzbar.
Für jeden Servo wird dafür ein PWM Signal benötigt.
Servo 1 ➔ GPIO_26
Servo 2 ➔ GPIO_38
Servo 3 ➔ GPIO_27
Die Servos liefern keine Rückmeldung zu deren Lage. Sie sind über entsprechende PWM-Signale in der Lage auf bestimmte Winkel ca. +- 93° zu positionieren, wobei 0° der Ausrichtung senkrecht nach unten entspricht (siehe Abbildung 56 / Ansicht des STMNucleo32-Baseboards von unten).
Die Grundfrequenz des PWM-Signals soll ca. 50Hz betragen. Über die Pulslänge kann dann der Winkel eingestellt werden. Ein Impulslänge von 1,5ms entspricht ca. einen Winkel von 0°. Ein längerer Impuls (bis ca. 2ms) bewirkt einen Winkel gegen den Uhrzeigersinn (-) und ein kürzerer Impuls (bis ca. 0,7ms) bewirkt einen Winkel im Uhrzeigersinn (+).
Zu beachten ist, dass der Servo 2 parallel zum externen Lüfter angeschlossen ist, also in der Regel nur einer der beiden Anschlüsse J16/JP2 belegt sein sollte.
Zu beachten ist außerdem, dass der Servo 1 parallel eine Leitung des I²C2-Bus (Sparkfun Qwiic Connect J8) verwendet, d. h. das dieser (J8) beim Betrieb vom Servo 1 nicht verwendet werden kann.
Für eine PWM Ansteuerung sind die GPIO_26/GPIO_27/GPIO_38 als PWM zu betreiben, d. h.

mit TIMER, PWM Klasse im Namespace Platform::BSP, usw..

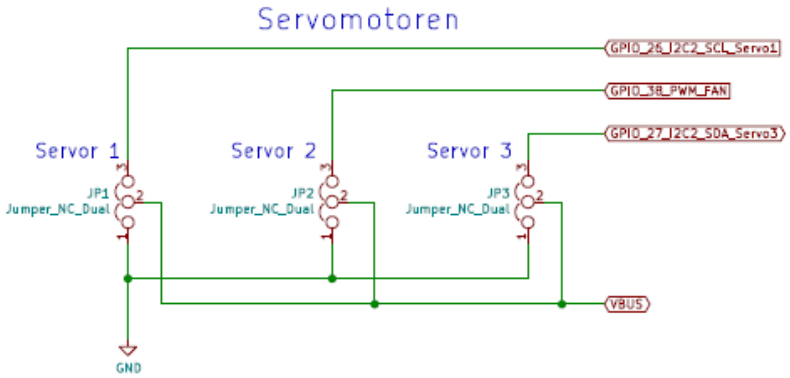

Abbildung 55: Schaltplan - Modellbau-Servos

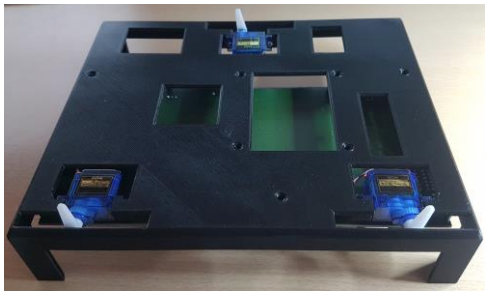

Abbildung 56: Modellbau-Servos

3.19. Externer Lüfter

Für Experimente kann mit dem STMNucleo32-Baseboard eine Regelung aufgebaut werden (Abbildung 59).
Hierzu wird am Anschluss J16 ein externer Lüfter (prinzipiell ein PC CPU-12V-Lüfter) angeschlossen. Mit einem PWM-Signal am GPIO_38 kann in einem Bereich die Drehzahl eingestellt werden. Der externe Lüfter liefert am GPIO_41 zweimal pro Umdrehung einen Impuls. Hiermit kann anhand des Interrupt die Ist-Drehzahl ermittelt werden (Abbildung 58). Beide Signale (Abbildung 57), PWM (Soll-Drehzahl, GRÜN) und Ist-Drehzahl (2 mal pro Umdrehung einen Impuls; ROT) können zu Diagnosezwecken über die Messpunkte J33 und J1 mit einem Oszilloskop gemessen werden.

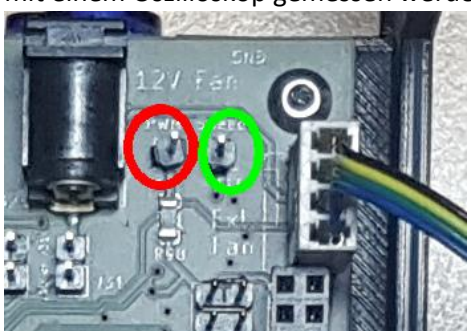

Abbildung 57: Messpunkte externer Lüfter

ACHTUNG, auf dem STMNucleo32-Baseboard, falsch mit PWM beschriftet!
ACHTUNG, auf dem STMNucleo32-Baseboard, falsch mit Speed beschriftet!

Beim Einstecken des Steckers des externen Lüfters an J16 ist auf die Polung zu achten (am Steck schwarz ist am J16 an GND zu stecken, Abbildung 57).
Das PWM Signal von GPIO_38 kann, wenn der Jumper J35 gesteckt ist, optisch über die liegende LED D16 visualisiert werden.
Für eine PWM Ansteuerung ist der GPIO_38 als PWM zu betreiben, d. h. mit TIMER, PWM Klasse im Namespace Platform::BSP, usw.. Zum Ermitteln der Ist-Drehzahl ist die Frequenz an GPIO_41 per Interrupt zu ermitteln.

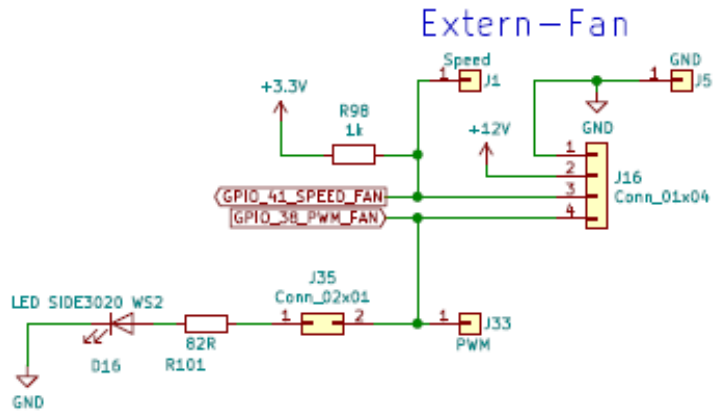

Abbildung 58: Schaltplan - Externer Lüfter

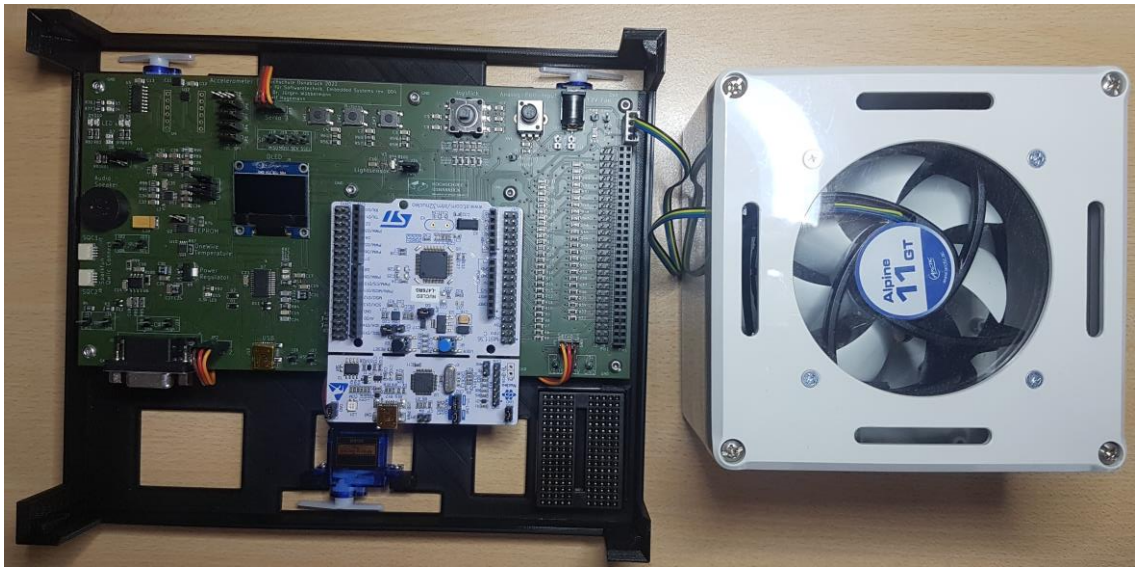

Abbildung 59: Externer Lüfter

4. Standard Jumpereinstellungen

Für den Betrieb des STMNucleo32 L476RG mit STMNucleo32-Baseboard gibt es einige Konfigurationsmöglichkeiten.
In der Standard-Einstellung müssen die Jumper, wie in Abbildung 60 in ROT gekennzeichnet, gesteckt sein.
Dafür sind auf dem STMNucleo32-Baseboard insgesamt 10 Jumper an die gekennzeichneten Stellen zu stecken (Abbildung 60).
Auf dem STMNucleo32 L476RG sind insgesamt 6 Jumper an die gekennzeichneten Stellen zu stecken (Abbildung 60).

Ändern Sie hier ohne Absprache mit den Modulverantwortlichen bitte nichts!

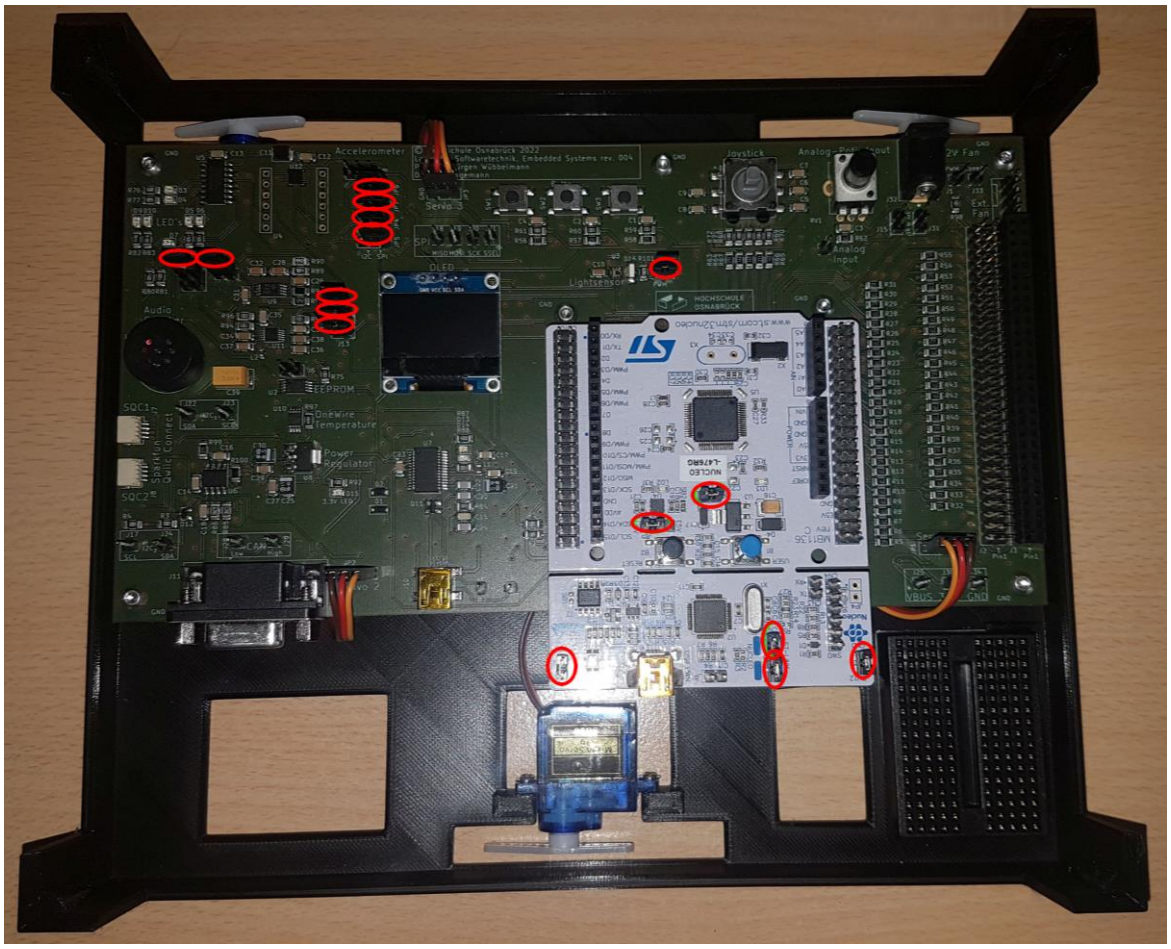

Abbildung 60: STMNucleo32-Baseboard - STMNucleo32 L476RG – Standard Jumpereinstellungen