PixelFan

by enny

Vorüberlegung

Geschwindigkeit des Lüfters:

Stufe 1: 17 U/s (60 ms pro Umdrehungen)
Stufe 2: 19 U/s (52 ms pro Umdrehungen)
Stufe 3: 21 U/s (47 ms pro Umdrehungen)
→ 1 Flügel alleine reicht nicht → 3 Flügel verwenden

Dimension:

Umfang an Flügelspitze: 1068 mm

Bildauflösung:

Ziel: Alle 6 mm neuer Pixel auf äußerster Bahn
26 LEDs pro Flügel = 78 LEDs

→ mindestens 178 refreshs pro Umdrehung (47ms)
→ alle 264 µs LED Refresh (Bei Stufe 3)

Vorüberlegung – LED Anforderungen

Alle 264 µs Refresh bei 78 LEDs bedeutet:

→3,38 µs Zeit zum Beschreiben einer LED

PWM Frequenz der LEDs?

Eine PWM-Periode sollte < 1mm sein!
47 ms/U mit 1068 mm Umfang = 44 µs/mm → 23 kHz

LED Auswahl

3,4 µs zum Beschreiben einer LED
>23 kHz PWM Frequenz

	PWM Frequenz	Zeit zum Beschreiben
WS2812	430 Hz	30 µs
SK9822*	1,2 kHz / 4,7 kHz	1,07 µs
APA102*	3,9 kHz	26,7 µs
APA107*	26 kHz	1,1 µs?
bestellt:	4,8 kHz ☹	
HD107S	26+ kHz	1,6 µs
bestellt:	33 kHz gemessen ☺	

*Angaben ohne Gewähr, teilweise sehr widersprüchlich

Schaltung

Controller: STM32G071 + ESP8266
(Trennung WLAN/LED-Ansteuerung wegen hoher Timing-Anforderungen)

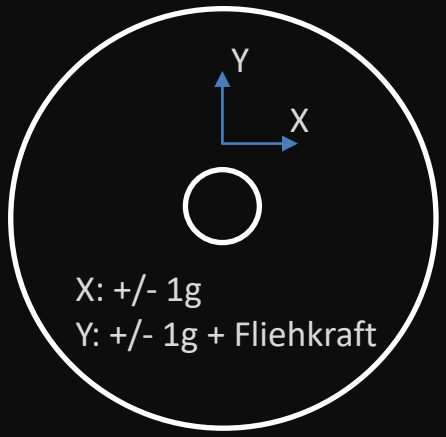
Spannungsversorgung: 5V über Schleifring (LEDs)
3,3 V vom Linearregler (Controller + ESP)

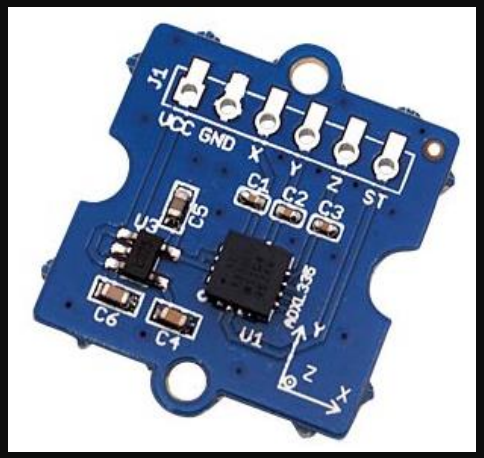
Synchronisierung: Beschleunigungssensor

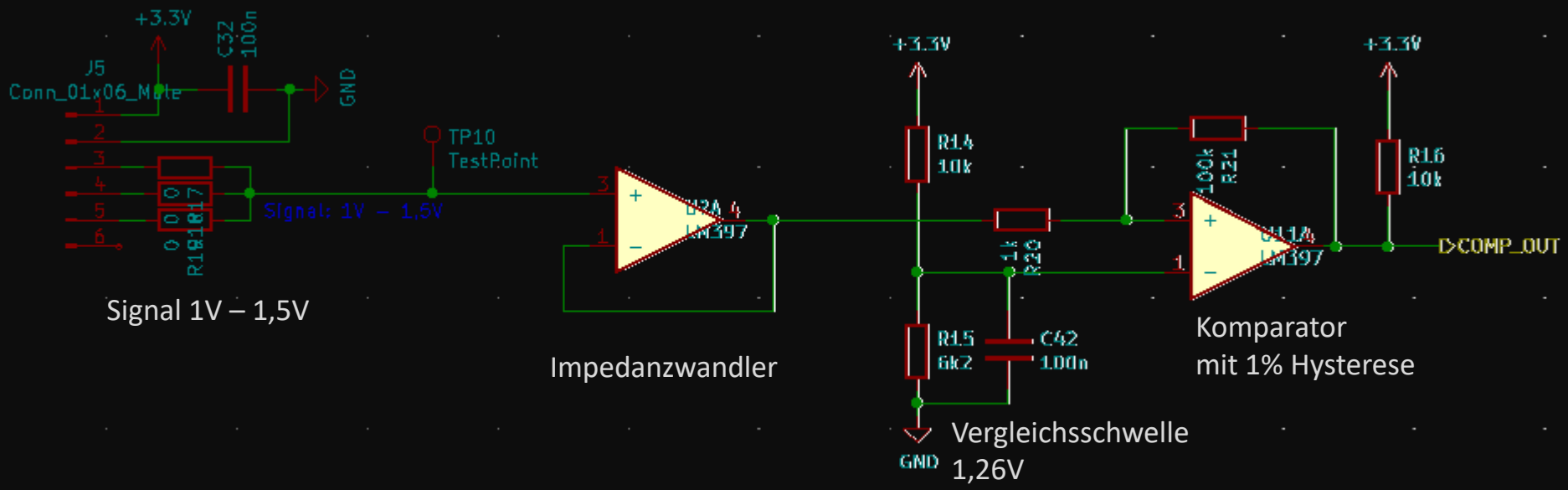
Beschleunigungssensor

ADXL335

- Läuft mit 1,8 – 3,6 V
- ± 3,6 g
- 1,6 kHz Bandbreite







Beschleunigungssensor

Problem:
Bei hohen Drehzahlen kein Signal mehr

Zentrifugalbeschleunigung:

$a = \omega^2 r$

$a = (2 \pi f)^2 r$

$a = (2 \pi 17\ \mathrm{Hz})^2\ 45\ \mathrm{mm}$

$a = 513\ \mathrm{m/s^2}$

$a = 52\ g$

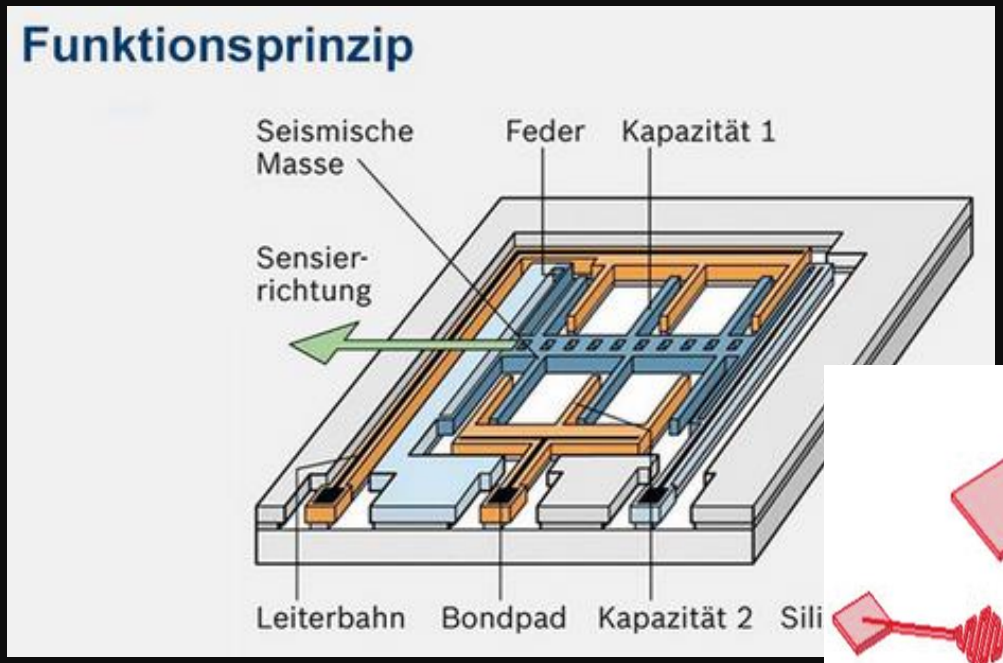
Datenblattangabe:

Absolute Maximum: 10000 g

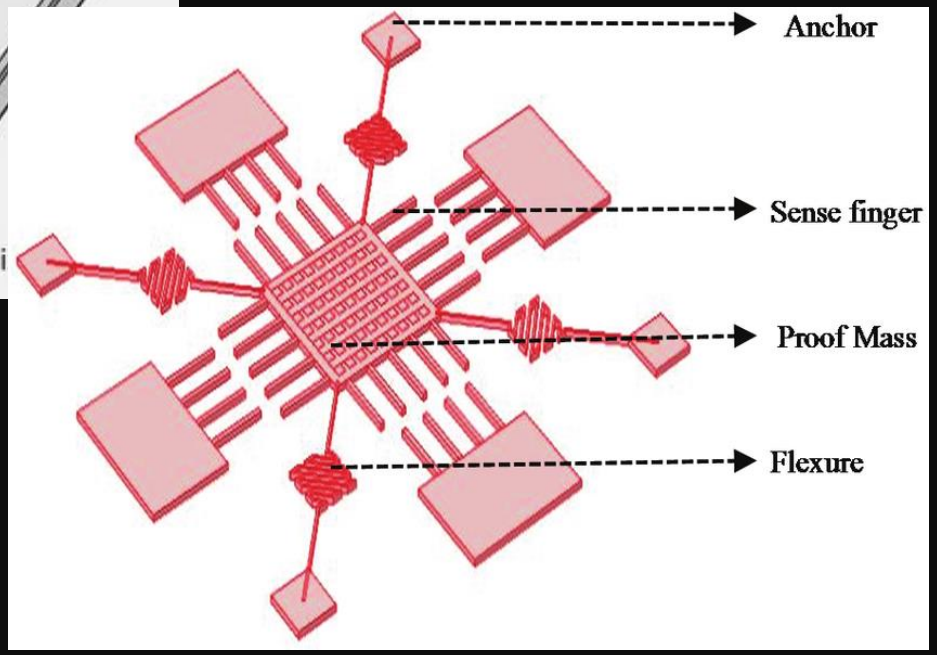
Messbereich: ± 3,6 g

Passt doch, oder?

Beschleunigungssensor



Bildquelle: https://www.bosch-mobility-solutions.de/de/produkte-und-services/pkw-und-leichte-nutzfahrzeuge/fahrsicherheitssysteme/fu%C3%9Fg%C3%A4ngerschutz/peripherer-beschleunigungssensor/



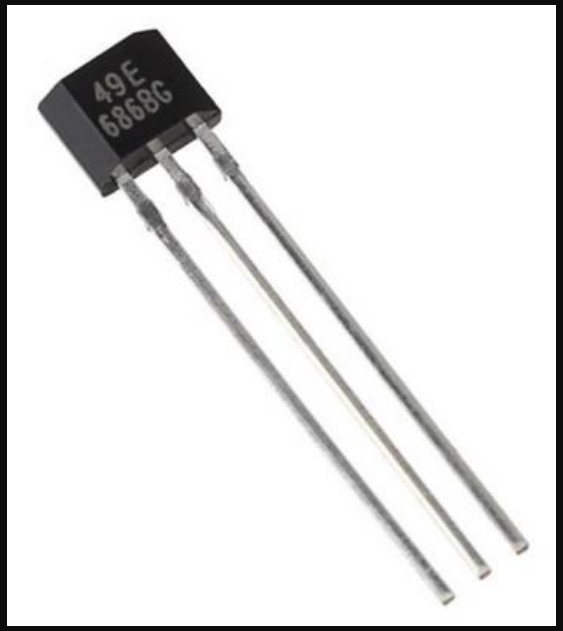
https://www.researchgate.net/figure/3D-Schematic-diagram-of-Tri-axes-accelerometer_fig3_311574442

Beschleunigungssensor

Was tun?

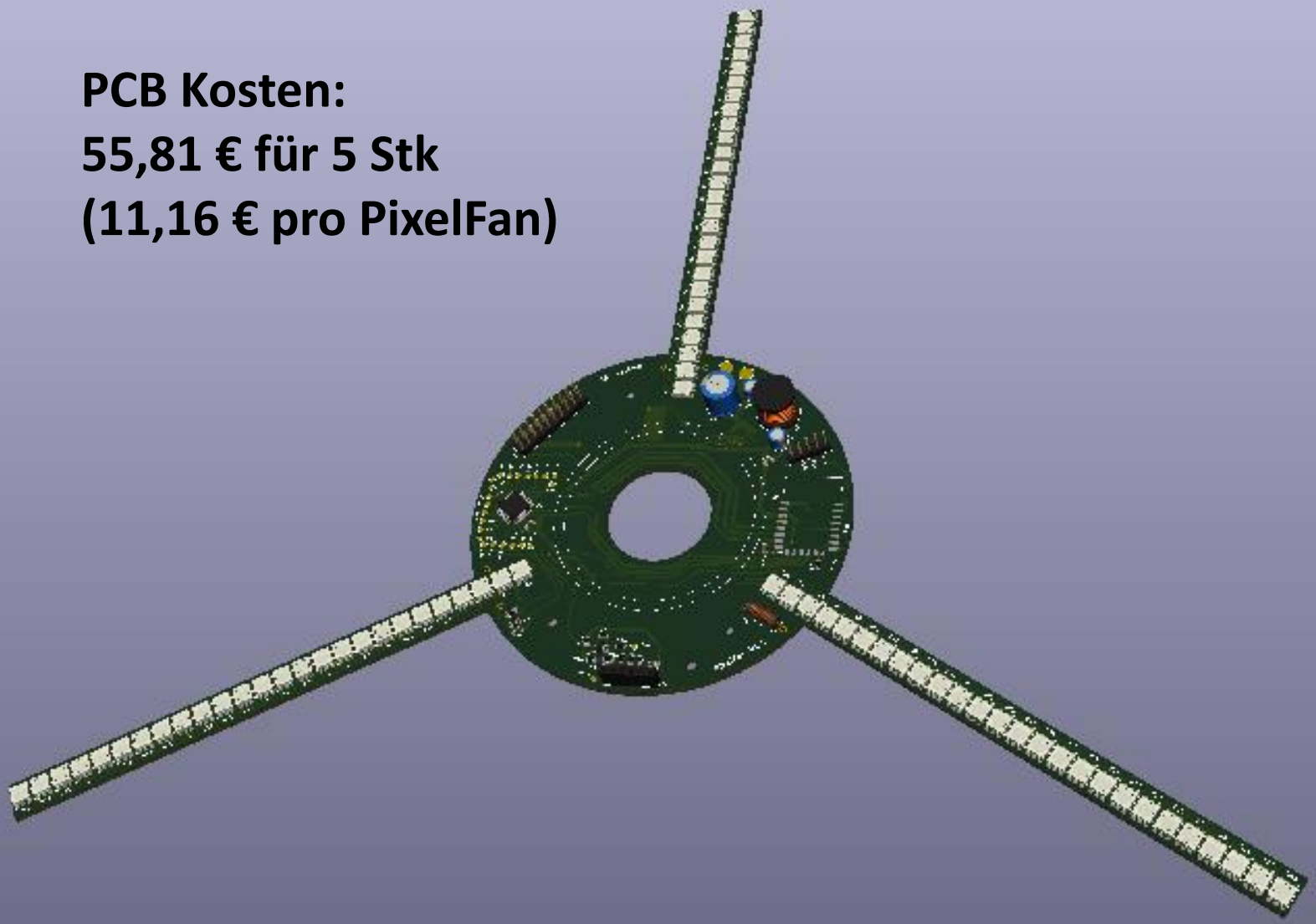
Beschleunigungssensor mit 100g Messbereich nehmen?
→Kostet über 100 € ☹

Dann doch Plan B:
Hallsensor TLE4905 + Magnet



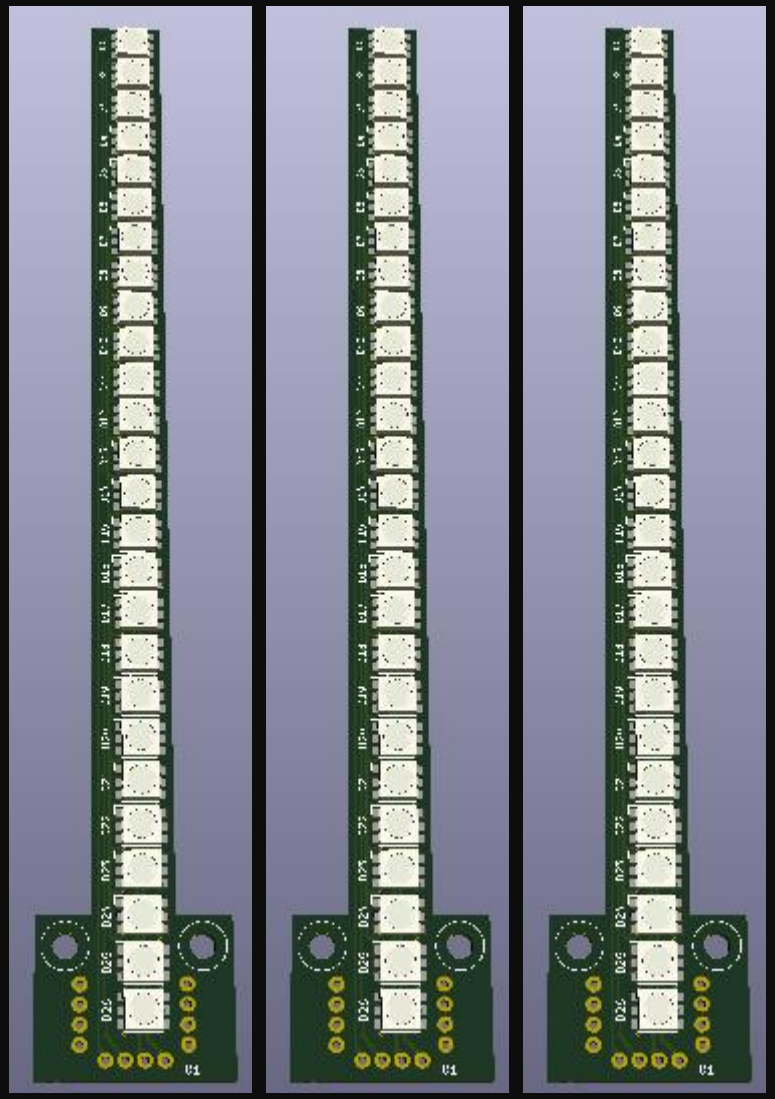
Platine – Option 1

PCB Kosten:
55,81 € für 5 Stk
(11,16 € pro PixelFan)

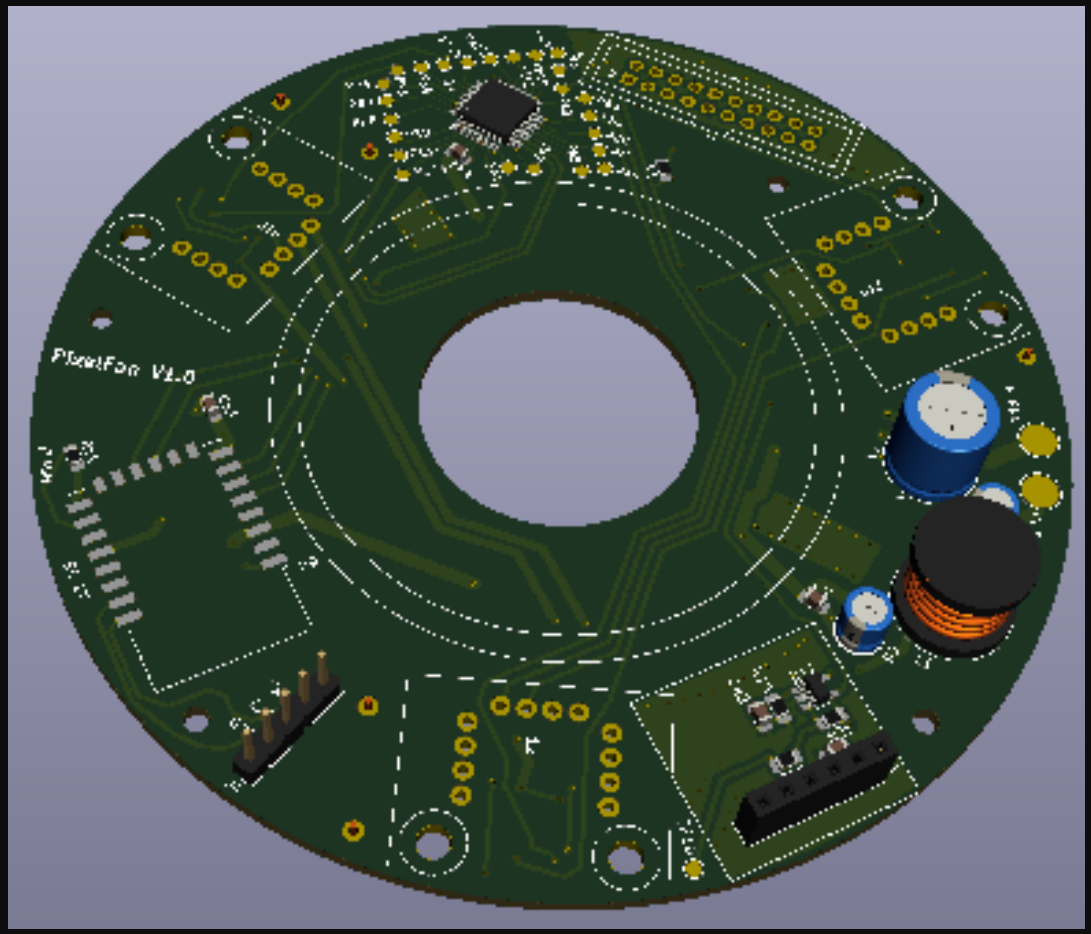


Platine – Option 2

5 € für 5 Stk

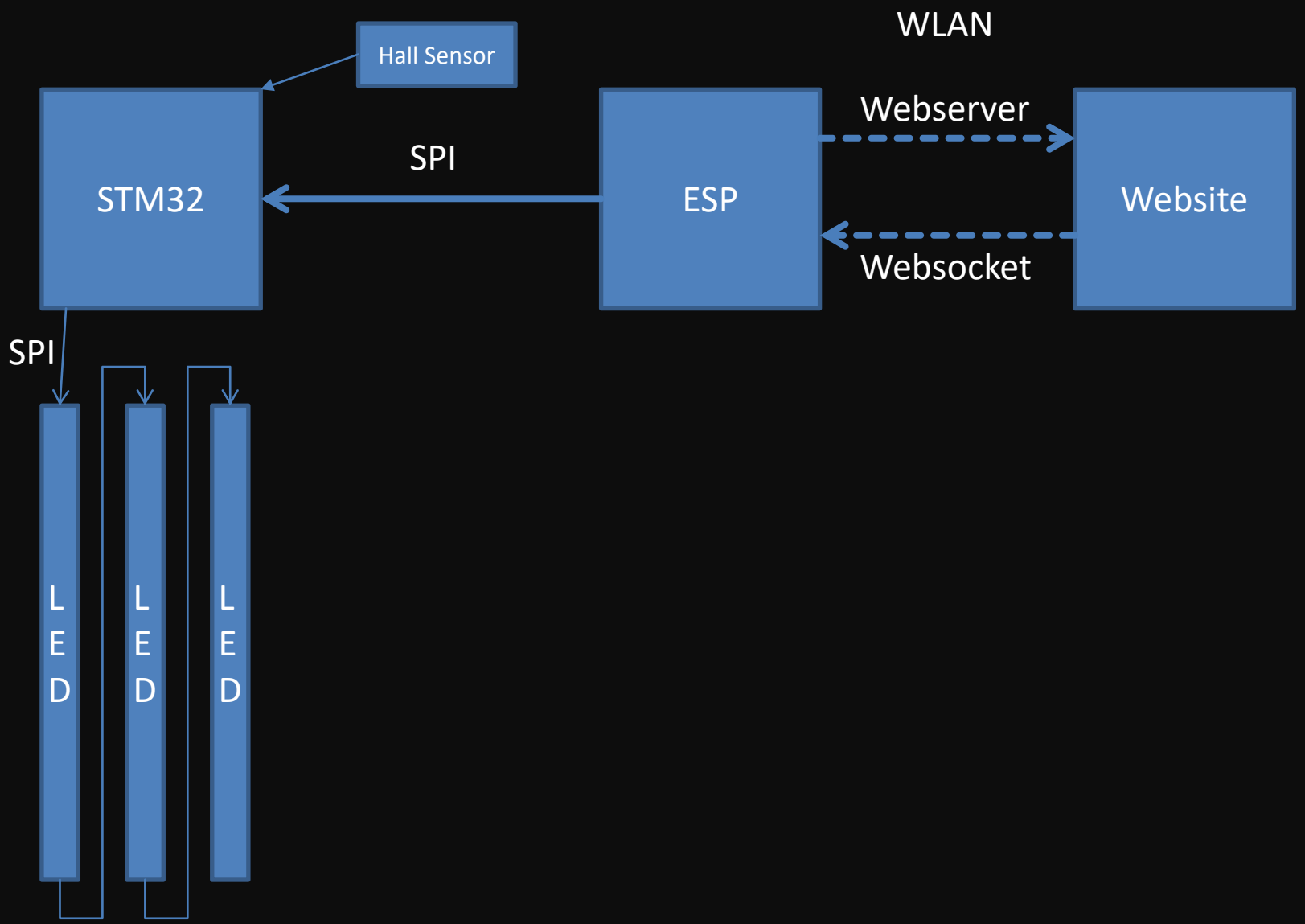


7,36 € für 5 Stk



→ 4,47 € pro PrixelFan

Systemaufbau

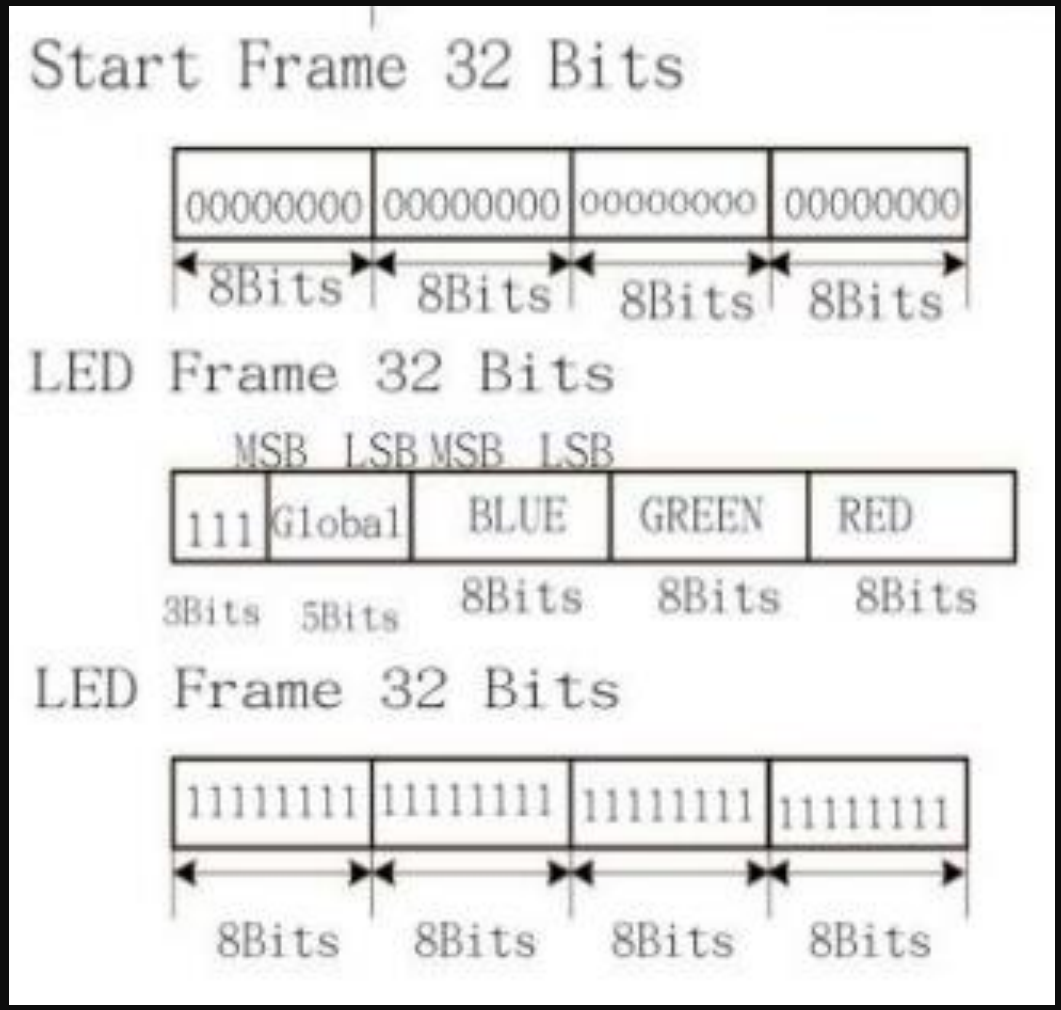


Software – LED Datentransfer

SPI zu den LEDs
3x 26 LEDs = 78 LEDs

Protokoll der LEDs:
4 Byte Startframe
4 Byte LED 1
4 Byte LED 2
4 Byte LED 3
……
4 Byte LED 78
5 Byte Endframe

= 321 Byte in 264 µs
→ 9,7 MHz mindestens
→ 16 MHz verwendet



Software – LED Datentransfer

Wie schnell läuft das?

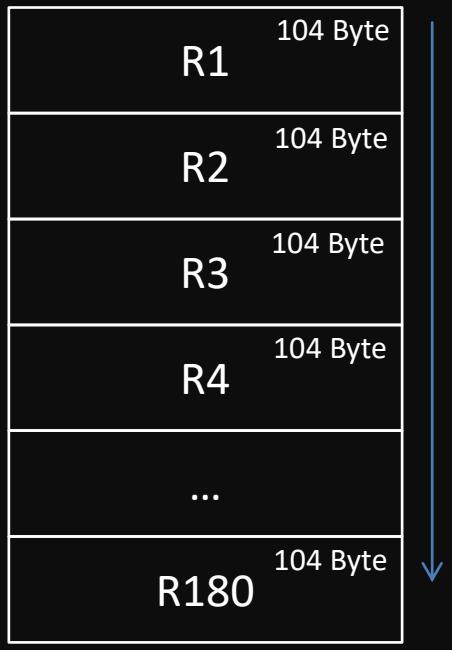
321 Bytes @16 MHz → 1 Refresh aller LEDs: 160,5 µs

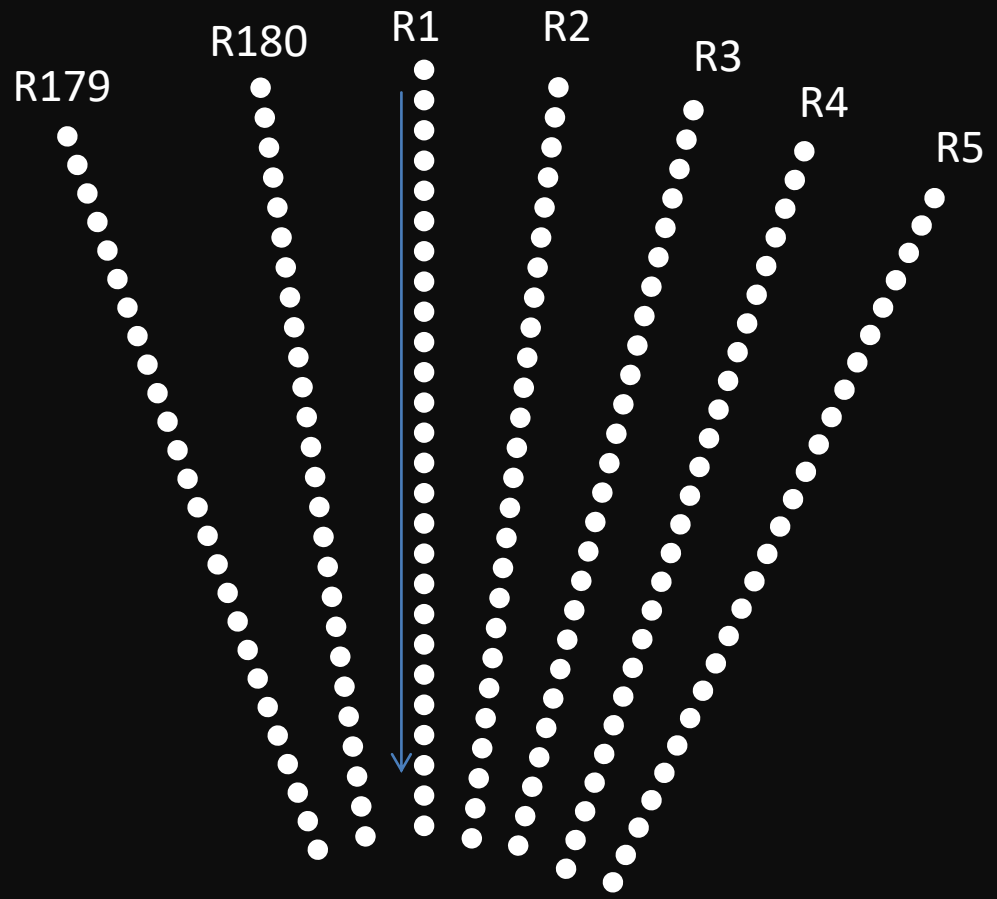
Mit Interrupt Overhead:
208 µs

Zur Erinnerung: Alle 264 µs wollen wir neu zeichnen
→ Passt ☺

Nebenbei: Keine CMSIS verwendet, da sonst noch mehr Overhead.
Es wird alles direkt in die Register geschrieben.
Mit CMSIS: 290 µs

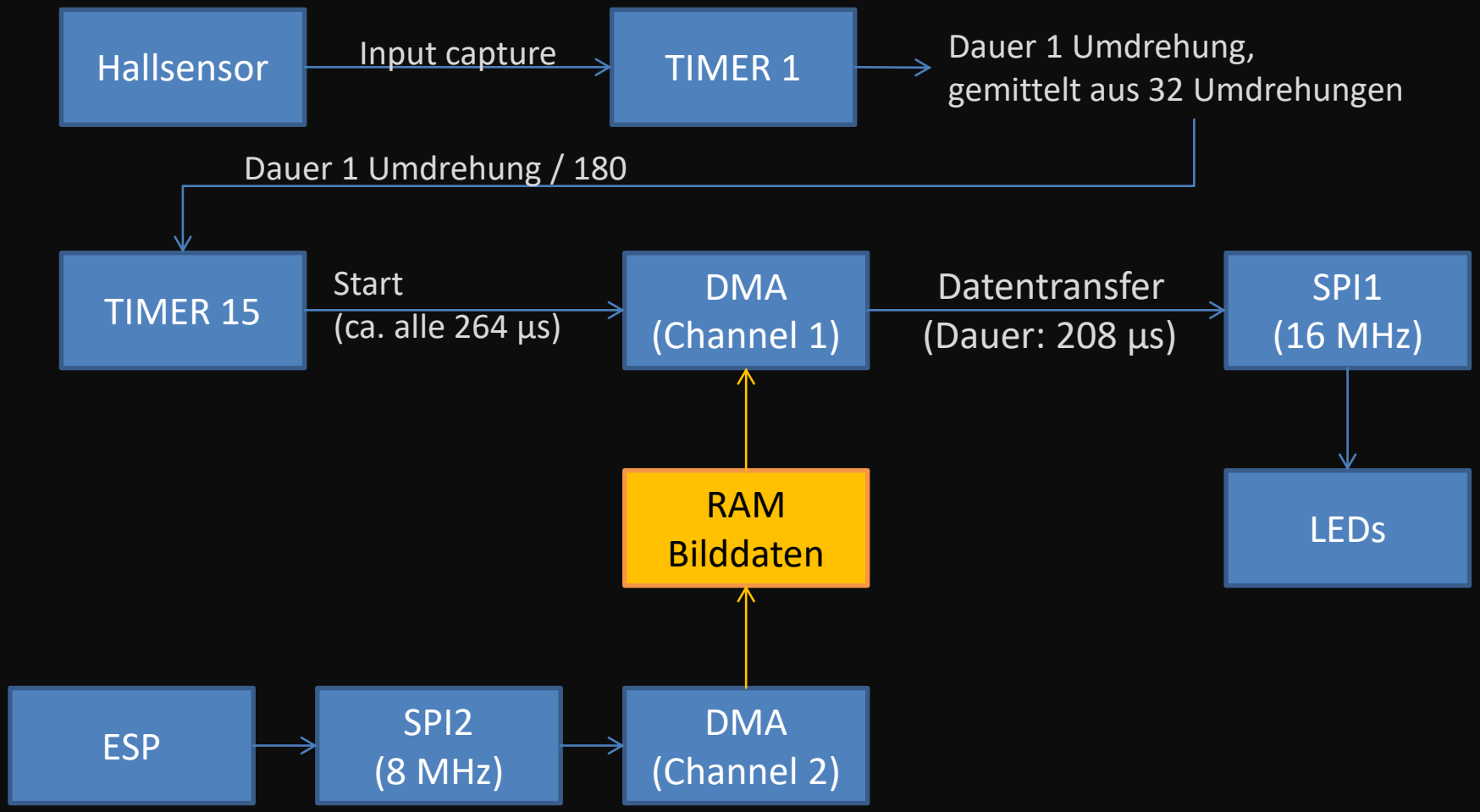
Software – Speicheraufteilung





→ 18720 Byte pro Bild (18,3kB)

Software - Bildaufbau



Software - ESP

Transfer ESP → STM Controller erfolgt ohne Protokoll!
ESP schreibt direkt 1:1 in den RAM
Vorteil: Schreiben der Daten benötigt 0 CPU Resourcen

→Auch der ESP muss jede LED mit 4 Bytes schreiben
(180 Reihen * 26 LEDs * 4 Byte = 18720 Byte bei 8 MHz → 18,7 ms)

Software – Websocket Schnittstelle

Problem:
- ESP Web Libs verwenden bereits viel Speicher vom ESP
- Oft dynamische Speichernutzung
- Websockets buffern mittels dynamischem Speicher

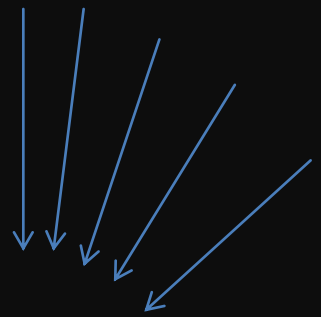
→ Um zuverlässig arbeiten zu können, dürfen maximal 1 kB auf einmal per Websocket übertragen werden.

→ Bild muss in mehrere Pakete aufgeteilt werden

Software – Websocket Schnittstelle

Protokoll:

Schreibe direkt in LED Buffer
Data[0] = Highbyte Startaddr.
Data[1] = Lowbyte Startaddr.
Data[2] = 0xE6
Data[3] = Blau LEDx
Data[4] = Grün LEDx
Data[5] = Rot LEDx
Data[6] = 0xE6
Data[7] = Blau LEDx+1
Data[8] = Grün LEDx+1
Data[9] = Rot LEDx+1
……..



Software – Website

Verbindet sich per Websocket (Port 81)
Foto hochladen
Texte darstellen
Smileys anzeigen
Vorschau

Die Website rechnet die Bilder im Javascript in das rotatorische Speicher-Format des PixelFans um.

Kosten pro PixelFan

STM32G071	2,40 €	
ESP8622	5,50 €	
LEDs (AliExpress)	11 €	(100 Stück)
Reichelt „Kleinkram“	0,50 €	
Schleifring (Amazon)	11 €	
PCBs	4,47 €	
Summe:	ca. 35 €	

*Preise aus 2019

Open Source

Software und Hardware ist auf Github zu finden

https://github.com/C3MA/PixelFan

Noch Fragen?