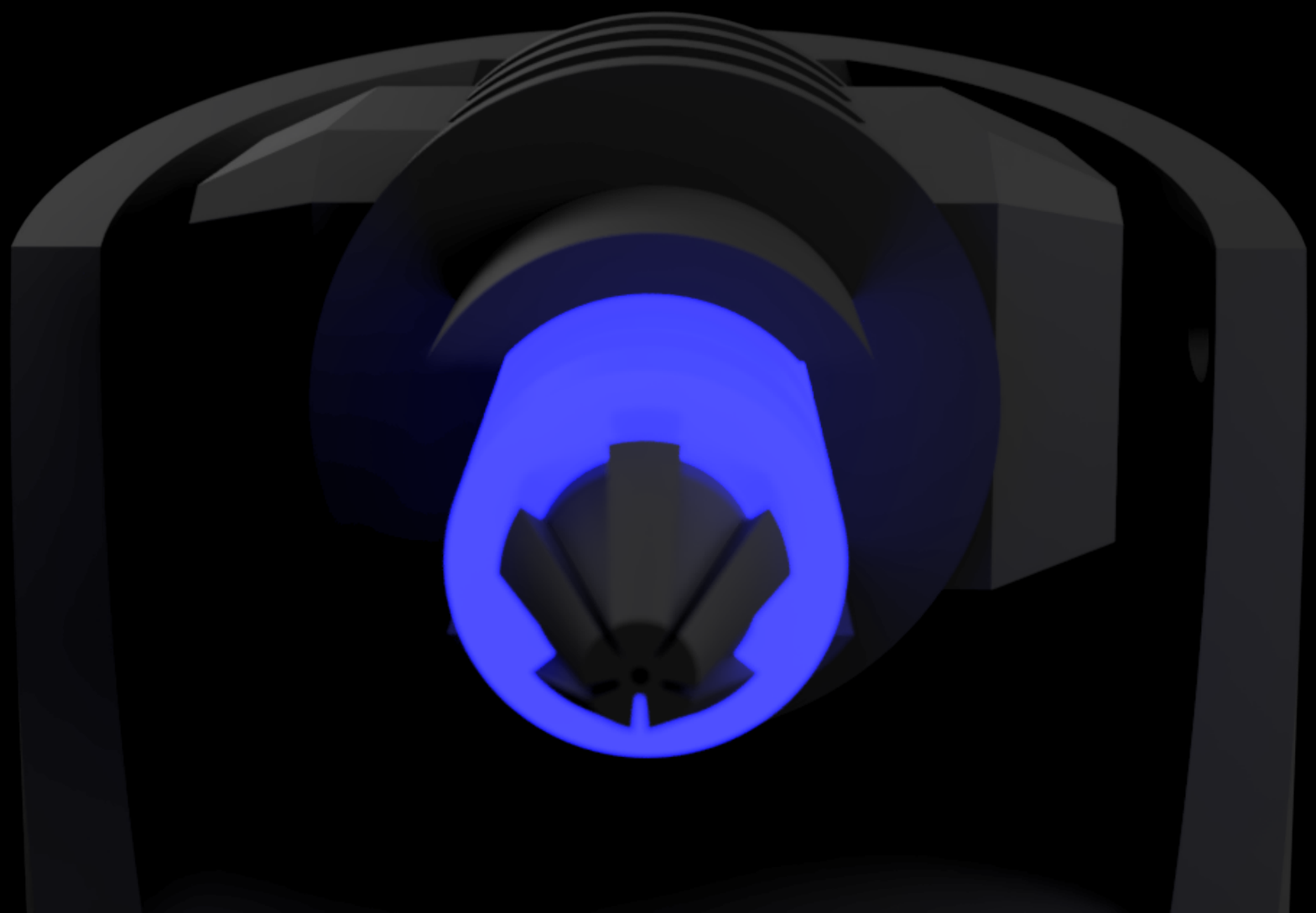
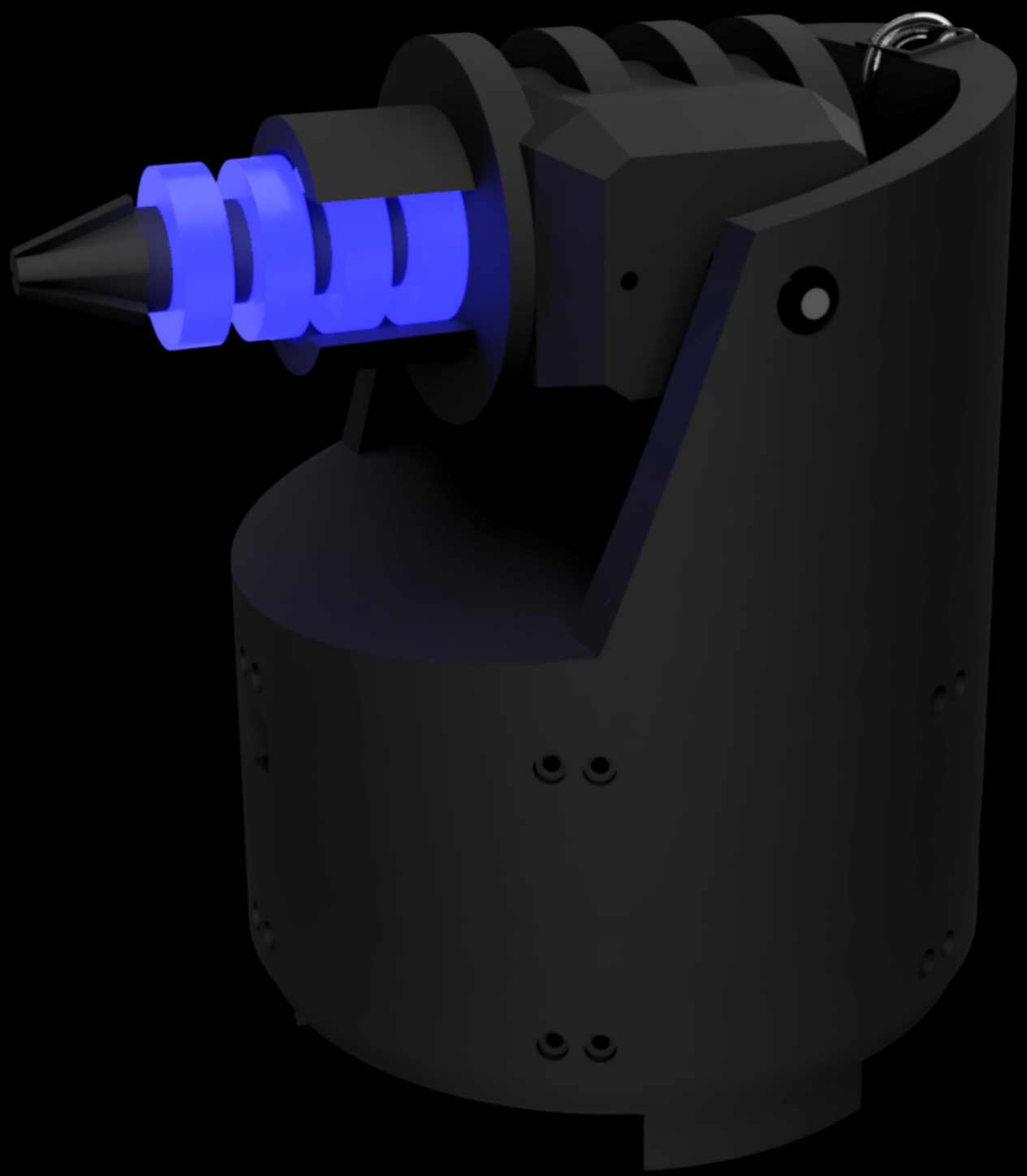
Embedded Systems TW

© 2017 Michael Gafert & Fabian Steiner

Waterblaster

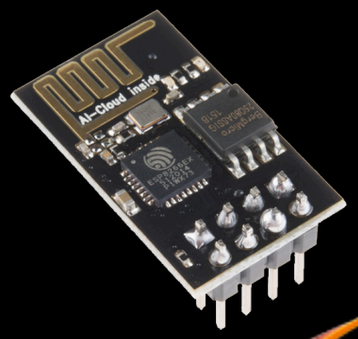




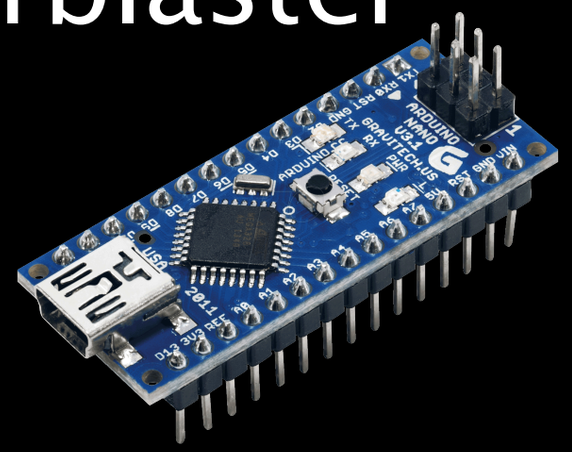
Features

- 5 Meter Reichweite
- 360° Drehung
- All-in-one
- LEDs
- WiFi Steuerung
- IR Steuerung

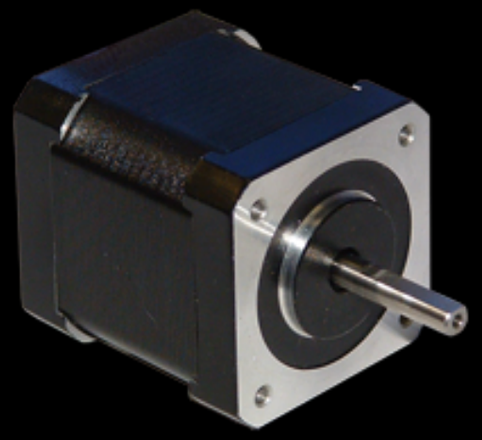
Komponenten Waterblaster



ESP8266 ESP01 Wifi Module

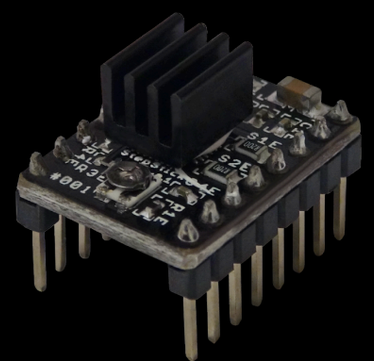


Chinese Arduino Nano

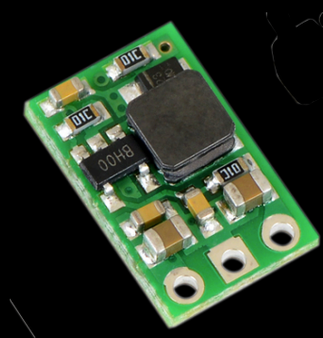


Stepper für Yaw

Servo für Pitch

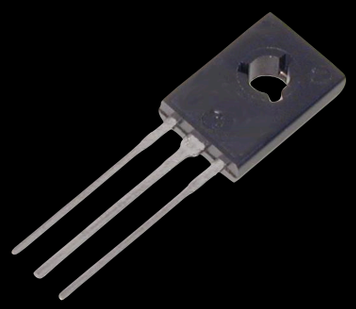


Stepper Driver A4988



3 Step Down Converter
3.3V 5V 12V

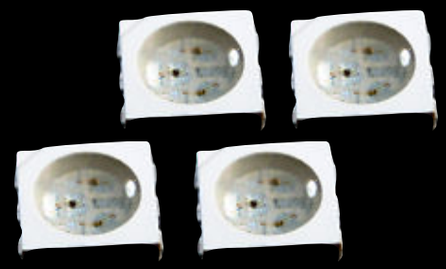
12V Pumpe



BD137 für Pumpe

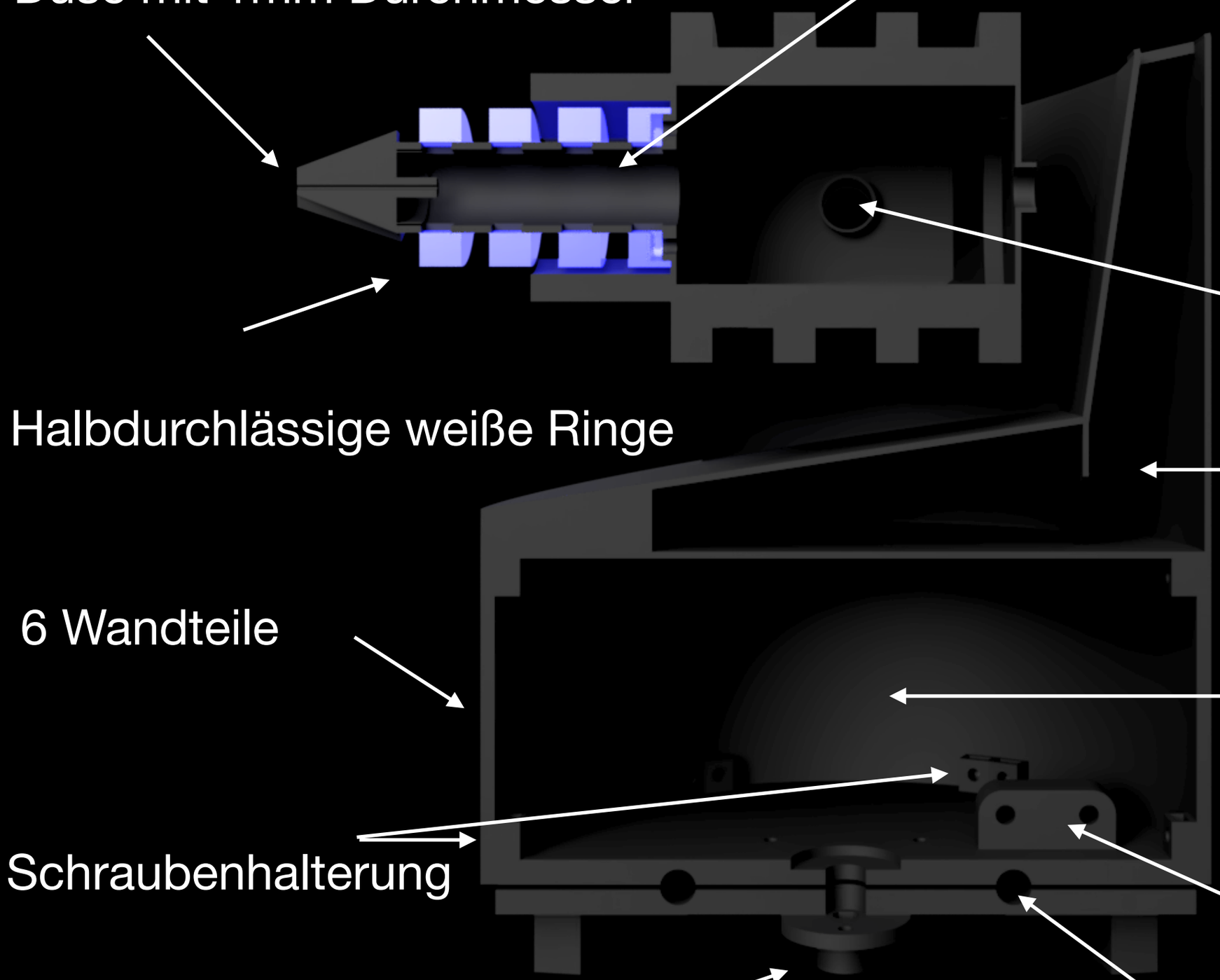
4S Lipo

TSOP4838



WS2812 LEDs

Waterblaster Model



Düse mit 1mm Durchmesser

WS2812s werden hier angebracht

Loch für die Welle

Halbdurchlässige weiße Ringe

Wasser Reservoir (400ml)

6 Wandteile

Platz für Elektronik, Motoren, Batterie und Pumpe

Schraubenhalterung

Halterung für Pumpe

Haltet den Schaft vom Stepper

Kugellager

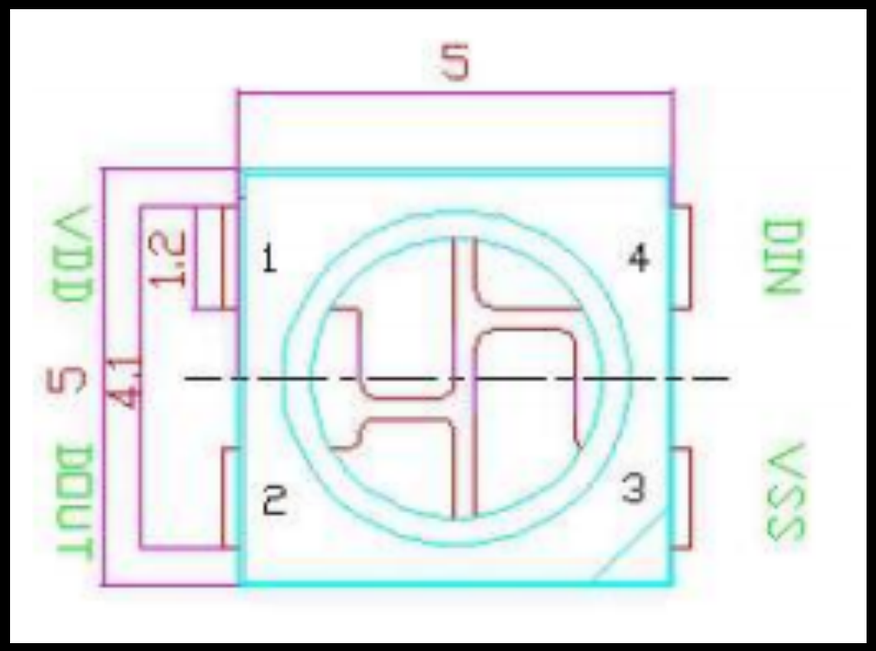
Model Informationen

- Modelierungs Programm: Fusion 360
- Slicer Programm: Cura 3
- 3D Drucker: Renkforce RF500

In depth look:

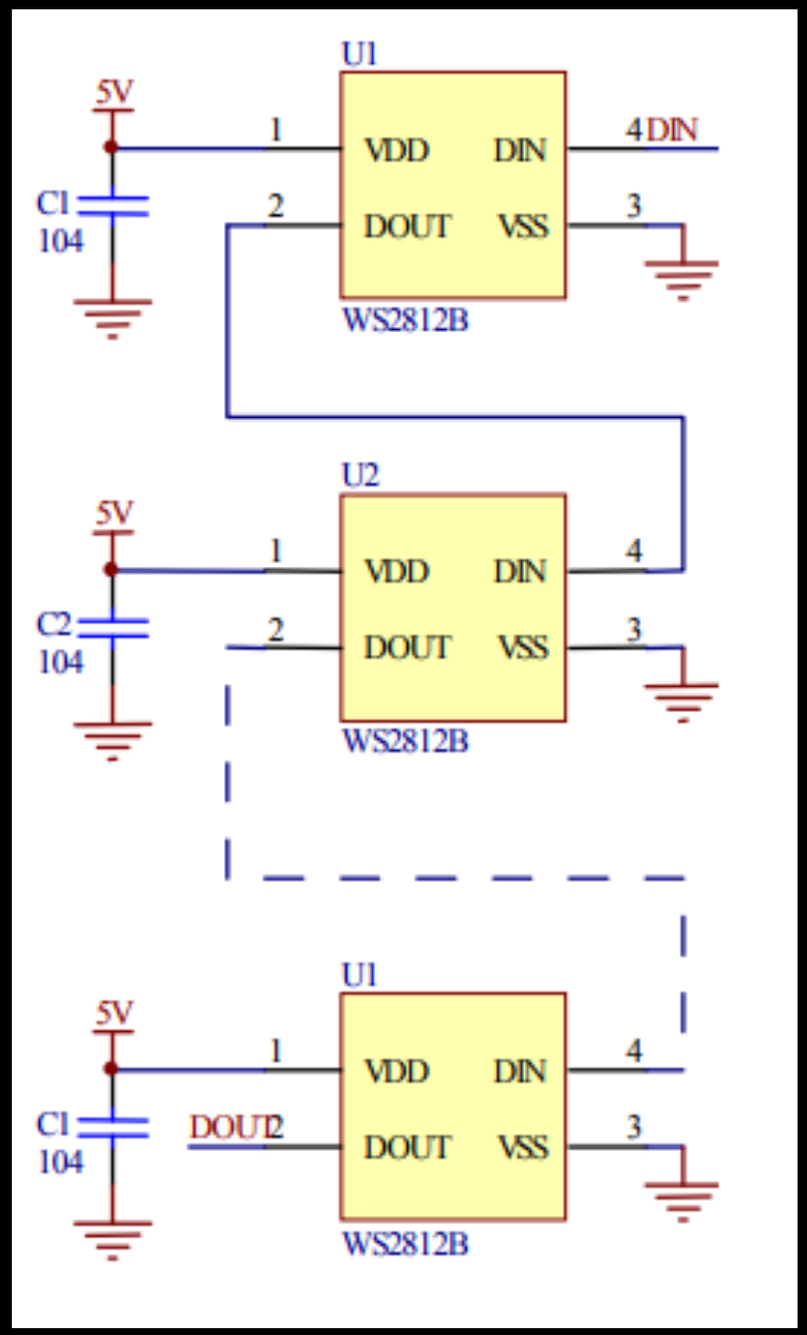
WS2812 LEDS

Pinout



NO.	Symbol	Function description
1	VDD	Power supply LED
2	DOUT	Control data signal output
3	VSS	Ground
4	DIN	Control data signal input

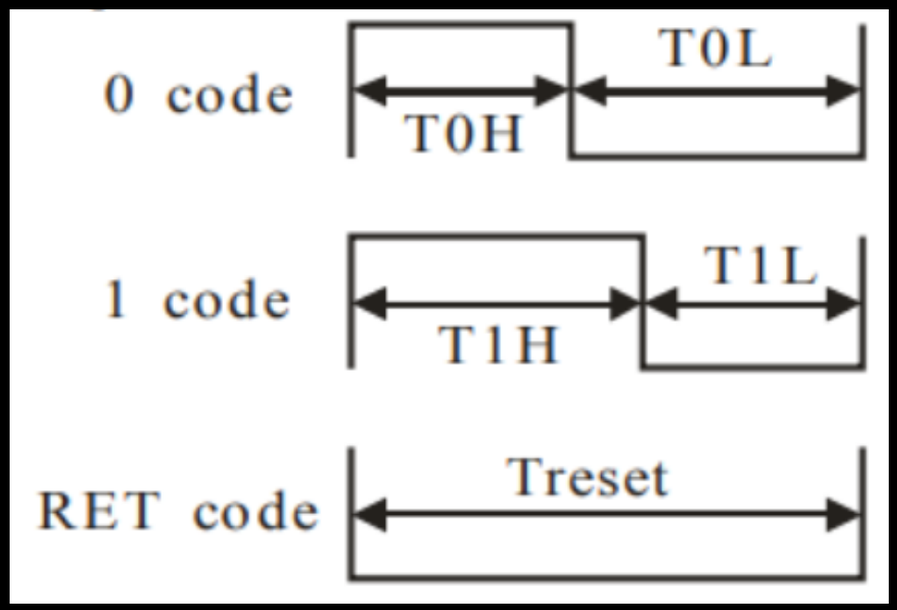
Schaltung



http://www.seeedstudio.com/document/pdf/WS2812B%20Datasheet.pdf

WS2812 LEDS

Timing



Methoden um das Signal zu erzeugen

- SPI Bus
- fastPWM
- Bitbanging (1,0,1,0,0,1)

T0H	0 code ,high voltage time	0.35us	±150ns
T1H	1 code ,high voltage time	0.9us	±150ns
T0L	0 code , low voltage time	0.9us	±150ns
T1L	1 code ,low voltage time	0.35us	±150ns
RES	low voltage time	Above 50μs	

http://www.seeedstudio.com/document/pdf/WS2812B%20Datasheet.pdf

WS2812 LEDS Bitbanging

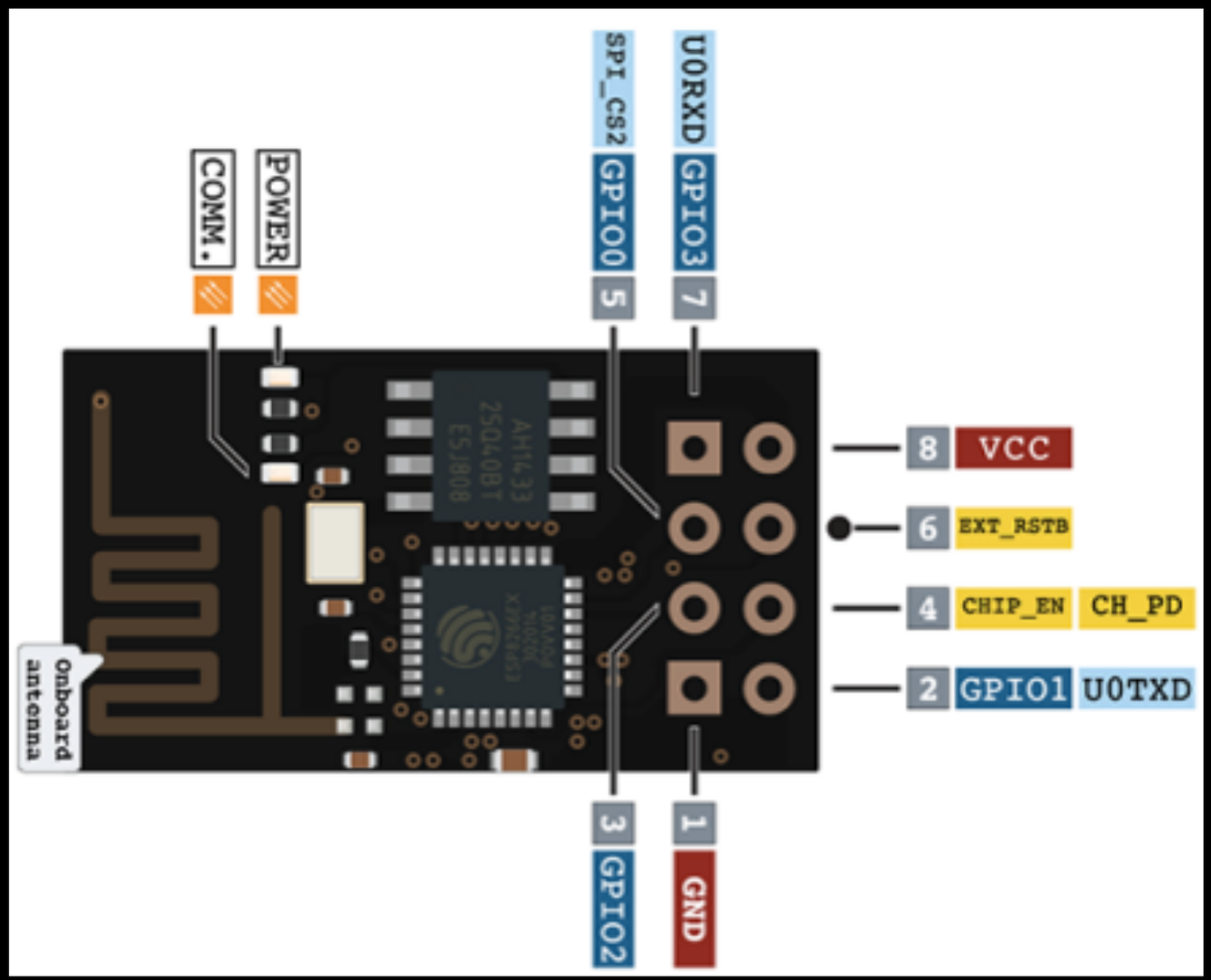
- Pin für eine kurze Zeit ein- und dann wieder ausschalten
- In Assembler geschrieben um der Optimierung aus dem Weg zu gehen —> Assembler ist genauer
- Adaptiert von Instructables http://www.instructables.com/id/Bitbanging-step-by-step-Arduino-control-of-WS2811-/

```
asm volatile(
//      Instruction          CLK     Description                 Phase      Example send 1   Example send 0
"nextbit:\n\t"               // -    label                       (T =  0)   -
        "sbi  %0, %1\n\t"    // 2    signal HIGH                 (T =  2)   signal HIGH   11 signal HIGH      11
        "sbrc %4, 7\n\t"     // 1-2  if MSB set                  (T =  ?)   1 in bit num  1  0 in bit number  11
        "mov  %6, %3\n\t"    // 0-1  tmp'll set signal high      (T =  4)   tmp = HIGH    1  tmp = LOW       -
        "dec  %5\n\t"        // 1    decrease bitcount           (T =  5)                 1                  1
        "nop\n\t"            // 1    nop (idle 1 clock cycle)    (T =  6)                 1                  1
        "st   %a2, %6\n\t"   // 2    set PORT to tmp             (T =  8)   signal tmp    11  signal tmp     00
        "mov  %6, %7\n\t"    // 1    reset tmp to low (default)  (T =  9)                 1                  0
        "breq nextbyte\n\t"  // 1-2  if bitcount==0 -> nextbyte  (T =  ?)                 1                  0
        "rol  %4\n\t"        // 1    shift MSB leftwards         (T = 11)                 1                  0
        "rjmp .+0\n\t"       // 2    nop nop                     (T = 13)                 11                 00
        "cbi   %0, %1\n\t"   // 2    signal LOW                  (T = 15)   signal LOW    00 signal LOW      00
        "rjmp .+0\n\t"       // 2    nop nop                     (T = 17)                 00                 00
        "nop\n\t"            // 1    nop                         (T = 18)                  0                  0
        "rjmp nextbit\n\t"   // 2    bitcount !=0 -> nextbit     (T = 20)                 00                 00
        "nextbyte:\n\t"      // -    label                       -
        "ldi  %5, 8\n\t"     // 1    reset bitcount              (T = 11)
        "ld   %4, %a8+\n\t"  // 2    val = *p++                  (T = 13)
        "cbi   %0, %1\n\t"   // 2    signal LOW                  (T = 15)
        "rjmp .+0\n\t"       // 2    nop nop                     (T = 17)
        "nop\n\t"            // 1    nop                         (T = 18)
        "dec %9\n\t"         // 1    decrease bytecount          (T = 19)
        "brne nextbit\n\t"   // 2    if bytecount!=0 -> nextbit  (T = 20)
        : // No output operands
        : // Input operands   Operand Id (w/ constraint)
        "I" (_SFR_IO_ADDR(WS2821_PORTx)), // %0
        "I" (WS2821_PIN),                 // %1
        "e" (&WS2821_PORTx),              // %a2
        "r" (high),                       // %3
        "r" (val),                        // %4
        "r" (nbits),                      // %5
        "r" (tmp),                        // %6
        "r" (low),                        // %7
        "e" (p),                          // %a8
        "w" (nbytes)                      // %9
        );
```

In depth look:

ESP8266

Pinout



Normaler Modus:

3.3 V —> VCC
3.3 V —> CH_PD
GND —> GND

Flash Modus:

3.3 V —> VCC
3.3 V —> CH_PD
GND —> GND
GND —> GPIO0

https://circuits4you.com/wp-content/uploads/2016/12/ESP-01-Pin-Out.png

ESP8266

- Programmiert mit PlatformIO
- Access Point Modus mit Webserver
- Kommunistiert mittels Websocket
- Sendet Kommandos über I2C

```
WiFi.mode(WIFI_AP);   // Set AP Mode
// Create AP with password and ssid
WiFi.softAP(ap_ssid, ap_password);
```

Enable AP

```
ESP8266WebServer server(80);
server.on("/", handleRoot);
server.begin();
```

```
void handleRoot() {
    server.send_P(200, "text/html", INDEX_HTML);
}
```

Enable server

Website mit Buttons

```
Wire.beginTransmission(8); // Send to device 8
Wire.write(com, 2);        // Writes 2 Bytes
Wire.endTransmission();    // Send stop
```

Send bytes over I2C

```
WebSocketsServer webSocket = WebSocketsServer(81);
webSocket.begin(); // Enables websocket on port 81
webSocket.onEvent(webSocketEvent); // Callback
```

Enable Websocket library

ESP8266

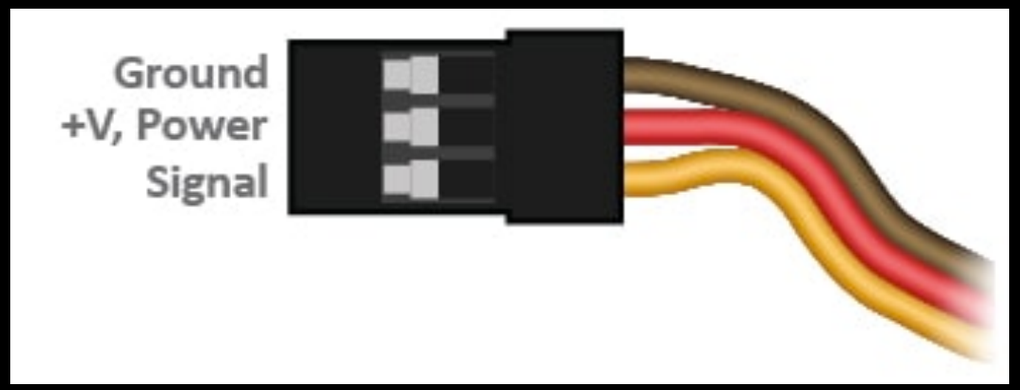
```
try {
    websock = new WebSocket('ws://' + window.location.hostname + ':81/');
    websock.onopen = function (evt) {
        console.log('websock open');
    };
    websock.onclose = function (evt) {
        console.log('websock close');
    };
    websock.onerror = function (evt) {
        console.log(evt);
    };
    websock.onmessage = function (evt) {
        console.log(evt);
        console.log(evt.data[0]);
        console.log(evt.data[1]);
        var id = evt.data[0];
        var state = parseInt(evt.data[1]);
        colorButton(id, state);
    }
} catch (err) {
    console.log("Server not running");
}
```

Websocket in Javascript

In depth look:

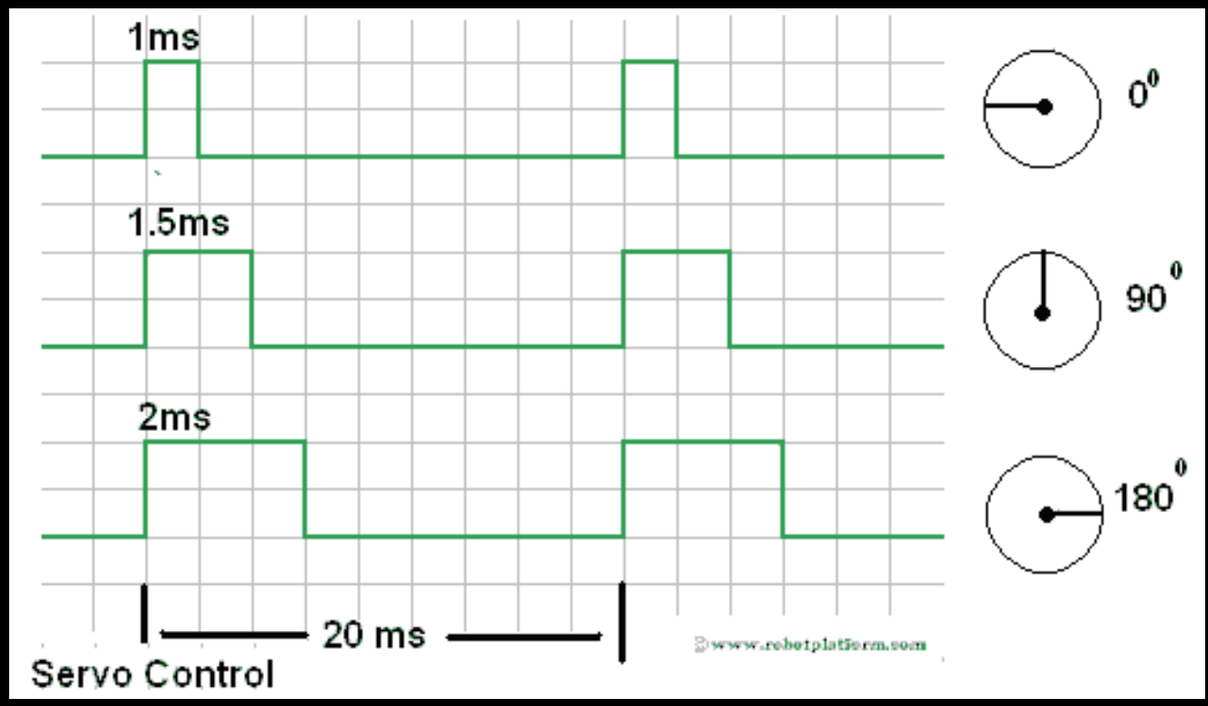
Servo

Pinout



https://www.addicore.com/Addicore-SG90-Mini-Servo-p/113.htm

Servo Control



http://www.robotplatform.com/knowledge/motion_control/servo-control.png

Servo

Init Timer with OCRIA in Compare Match Mode

```
void init_servo(void) {
    DDRB |= _BV(PORTB1);    // Connected to PORTB1

    // Timer wird im CTC Modus betrieben =
    // ISR wird immer dann ausgelöst,
    // wenn der Wert der ins OCR1A steht durch hinaufzählen
    // von dem Startwert 0 erreicht wurde (aka Compare Match)
    // Danach wird der Startwert wieder auf 0 gesetzt (aka Clear Timer on Compare)

    TCCR1A = (1 << COM1A0);                                 // Toggelt bei Compare Match PB1
    TCCR1B = (1 << WGM12) | (1 << CS11) | (1 << CS10); // CTC-Mode; Prescaler 64
    TIMSK1 = (1 << OCIE1A);                                 // Timer-Compare Interrupt an

    OCR1A = 4625;                                           // Neutralposition ((5000-4625)*0.004ms=1,5ms)
    // Höchstwert links = 500*0.004 = 2ms
    // Höchstwert rechts = 250*0.004 = 1ms

    sei();                                                  // Interrupts global an
}
```

Servo

TIMER1 Compare Interrupt

```
ISR(TIMER1_COMPA_vect) {
    OCR1A = 5000 - OCR1A;
    // Das Servosignal wird aus der Differenz von
    // Periodenlänge (5000*0,004ms=20ms) und letztem
    // Vergleichswert (OCR1A) gebildet
    // --> Wenn OCR1A = 375 dann steht der Servo in der Mittelposition (=1,5ms)
}
```

Servo

Zur Position Bewegen

```
// Bewegt den Servo zur richtigen Stelle
void moveServo(uint16_t position) {
    cli();
    // Da OCR1A immer hin und her toggelt (zwischen 0-500 und 4500 - 5000),
    // muss vor dem Setzen der Position überprüft werden, auf welchen Wert man gerade steht.
    if (OCR1A <= 500) {
        OCR1A = position;
    } else if (OCR1A >= 4500) {
        OCR1A = 5000 - position;
    }
    sei();
}
```

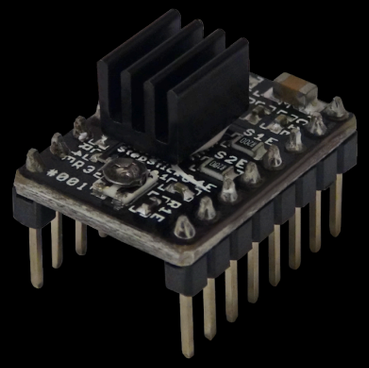
```
void servo_move_down() {
    servo_currentMillis = millis();
    if ((servo_currentMillis - servo_previousMillis) > servo_speed) {
        servo_previousMillis = servo_currentMillis;
        if (moveTo <= 500 - 1) {
            moveTo += 1;
        }
        moveServo(moveTo);
    }
}
```

In depth look:

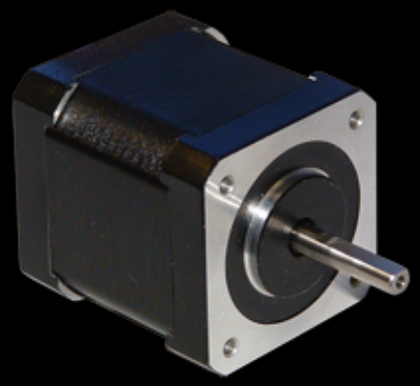
Stepper & Treiber

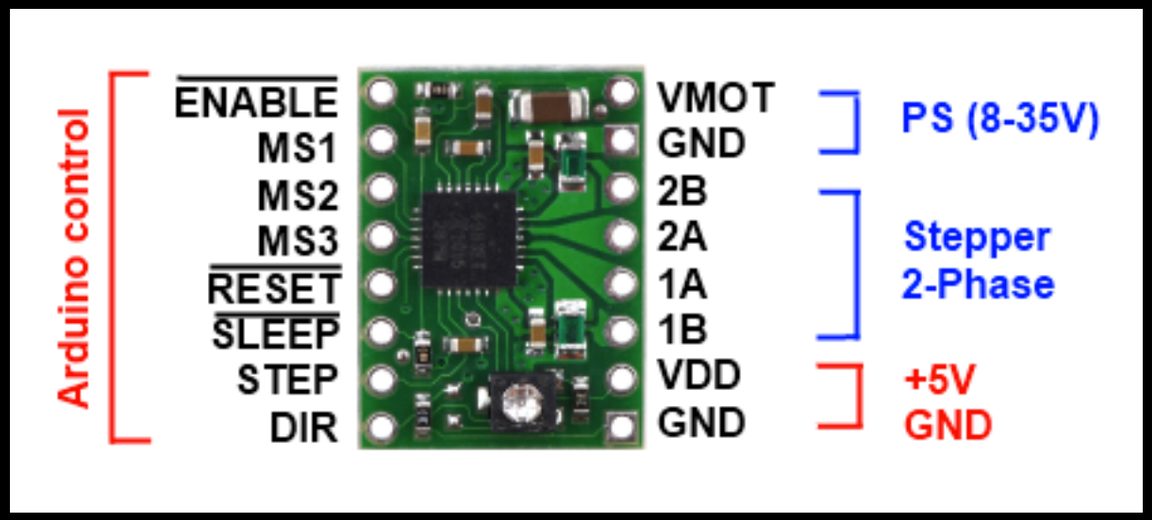
- Sehr präzise
- Bis zu 0.1° pro Schritt
- Verbraucht viel Strom
- Benötigt eigenen Treiber zur Steuerung
- Besteht aus mehreren Spulen

Stepper Driver A4988



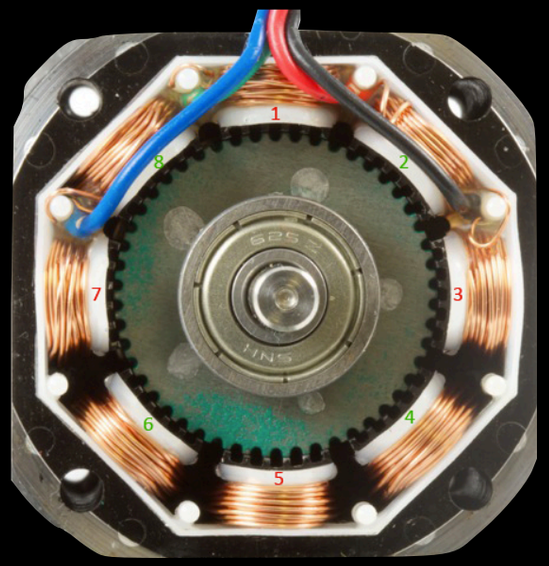
Stepper





http://www.arduino-projekte.de/images/Module/Pololu/Pololu_pins_450px.png

Aufbau eines Steppers



https://www.phidgets.com/docs21/Stepper_Motor_and_Controller_Primer

In depth look:

4S Lipo

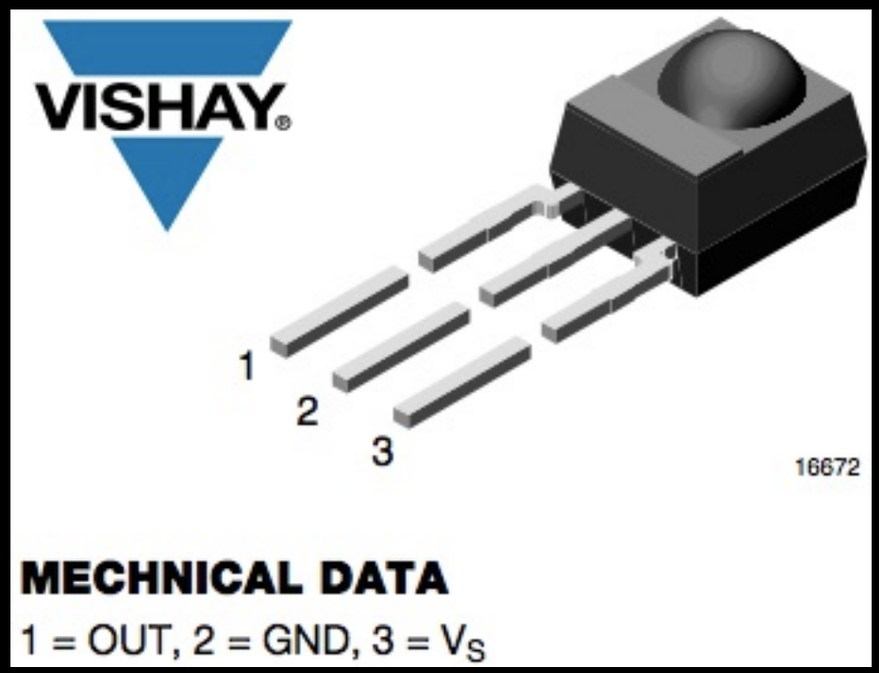
- Für Modelsport
- 4S = 4 Zellen
- Kann viel Strom liefern
- Pro Zelle 3,7 V Nennspannung
- Darf nicht unter 3 V pro Zelle Fallen
- Benötigt eigene Ladegeräte

In depth look:

IR Empfangen

- TSOP4838
- 38 kHz Bandpass
- Output negiert

Pinout

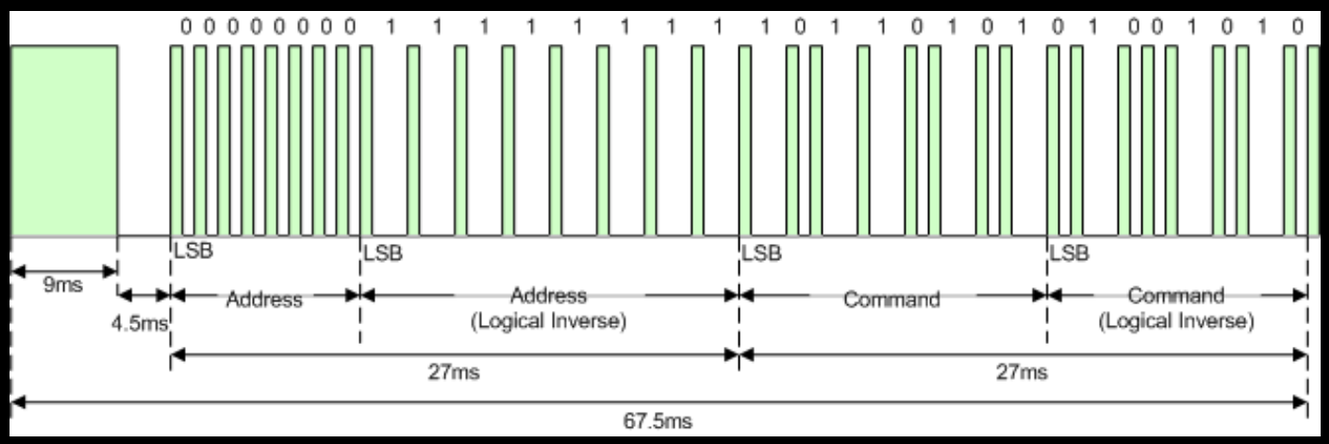


https://maxstrauch.github.io/projects/media-center/tsop4838.jpg

Programmierung

- Interrupt bei fallender Flanke
- Timer1 zählt Zeit mit
- NEC Protokoll verwendet

NEC Protokoll

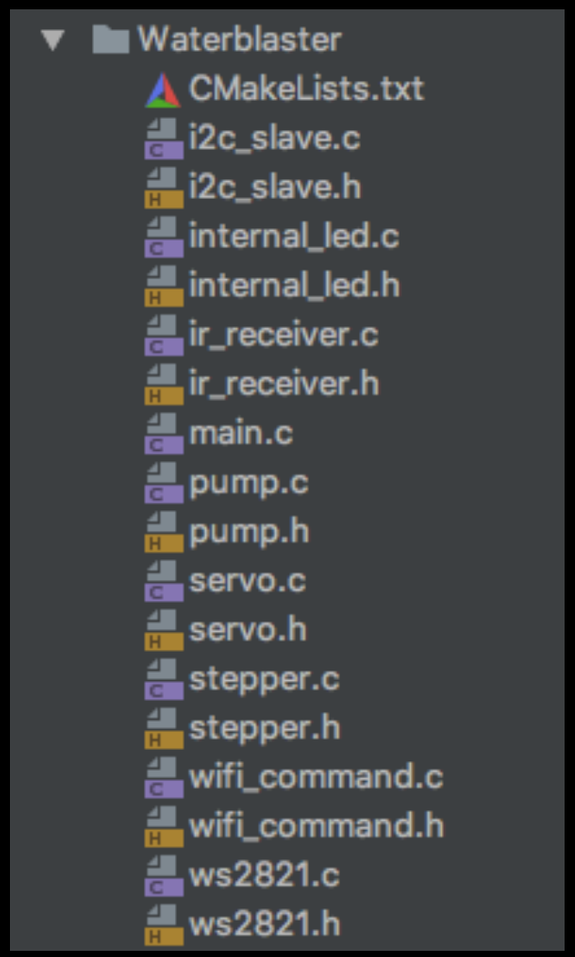


http://techdocs.altium.com/sites/default/files/wiki_attachments/296329/NECMessageFrame.png

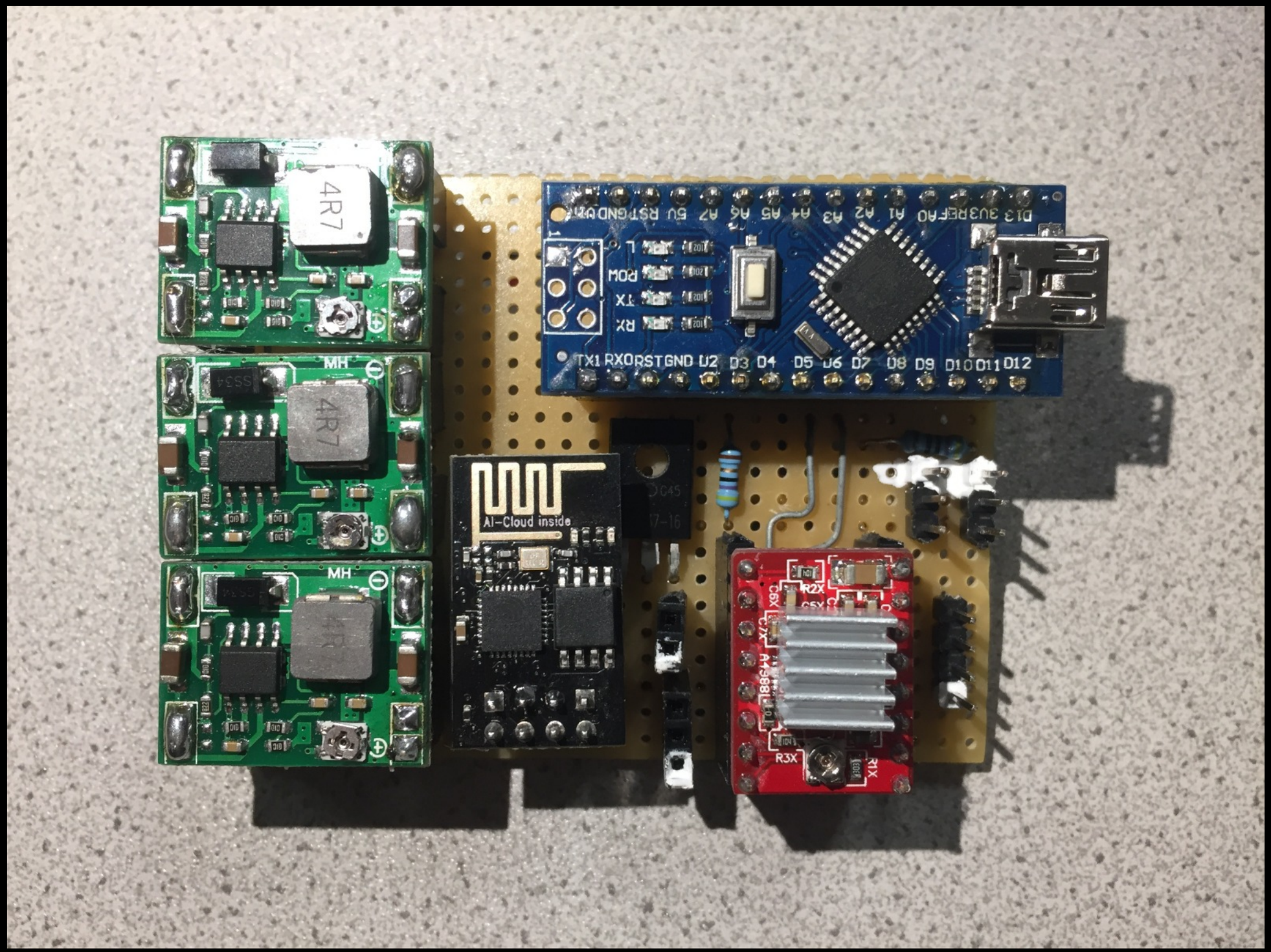
main() {…}

```
int main() {
    init_stepper();
    init_leds(leds, NUM_LEDS);
    init_internal_led();
    init_servo();
    init_pump();
    init_IRReceiver();
    init_nozzle_animation();
    init_wifi_command();
```

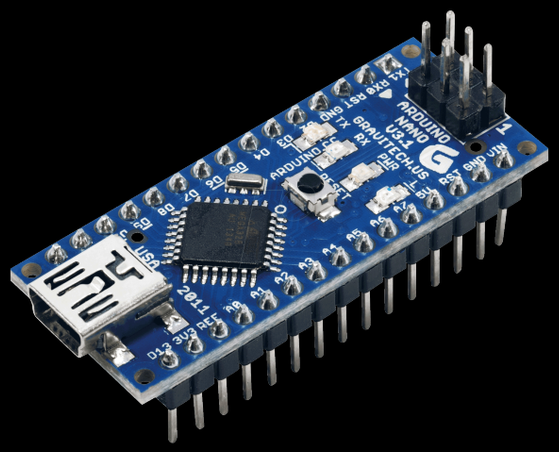
```
while(1){
    if(isIrRightActive || isWifiRightActive){
        stepper_move_clockwise();
    }
    if(isIrLeftActive || isWifiLeftActive){
        stepper_move_counter_clockwise();
    }
    if(isIrDownActive || isWifiDownActive){
        servo_move_down();
        _delay_ms(2);
    }
    if(isIrUpActive || isWifiUpActive){
        servo_move_up();
        _delay_ms(2);
    }
    if(isIrShootActive || isWifiShootActive){
        pump_on();
        color colors_ = {255,0,1};
        set_color(colors_);
    } else {
        pump_off();
        reset_color();
    }
}
```



Perfboard Waterblaster



Komponenten IR Remote

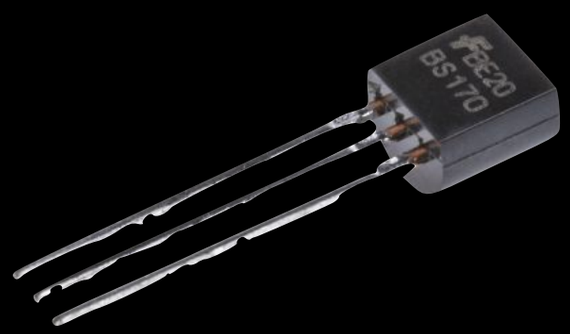


Chinese Arduino Nano

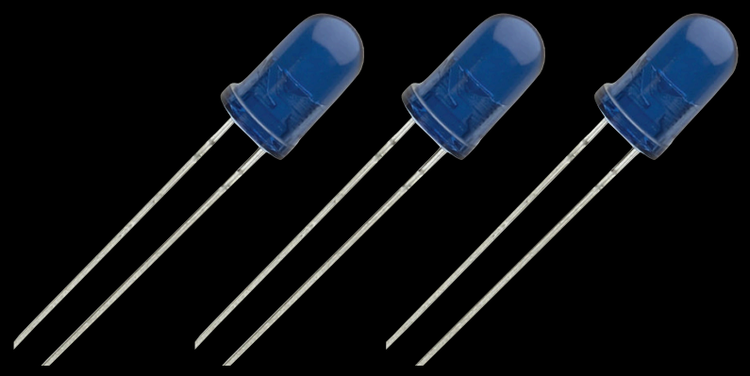
9V Battery

7805 5V Spannungsregler

Joystick

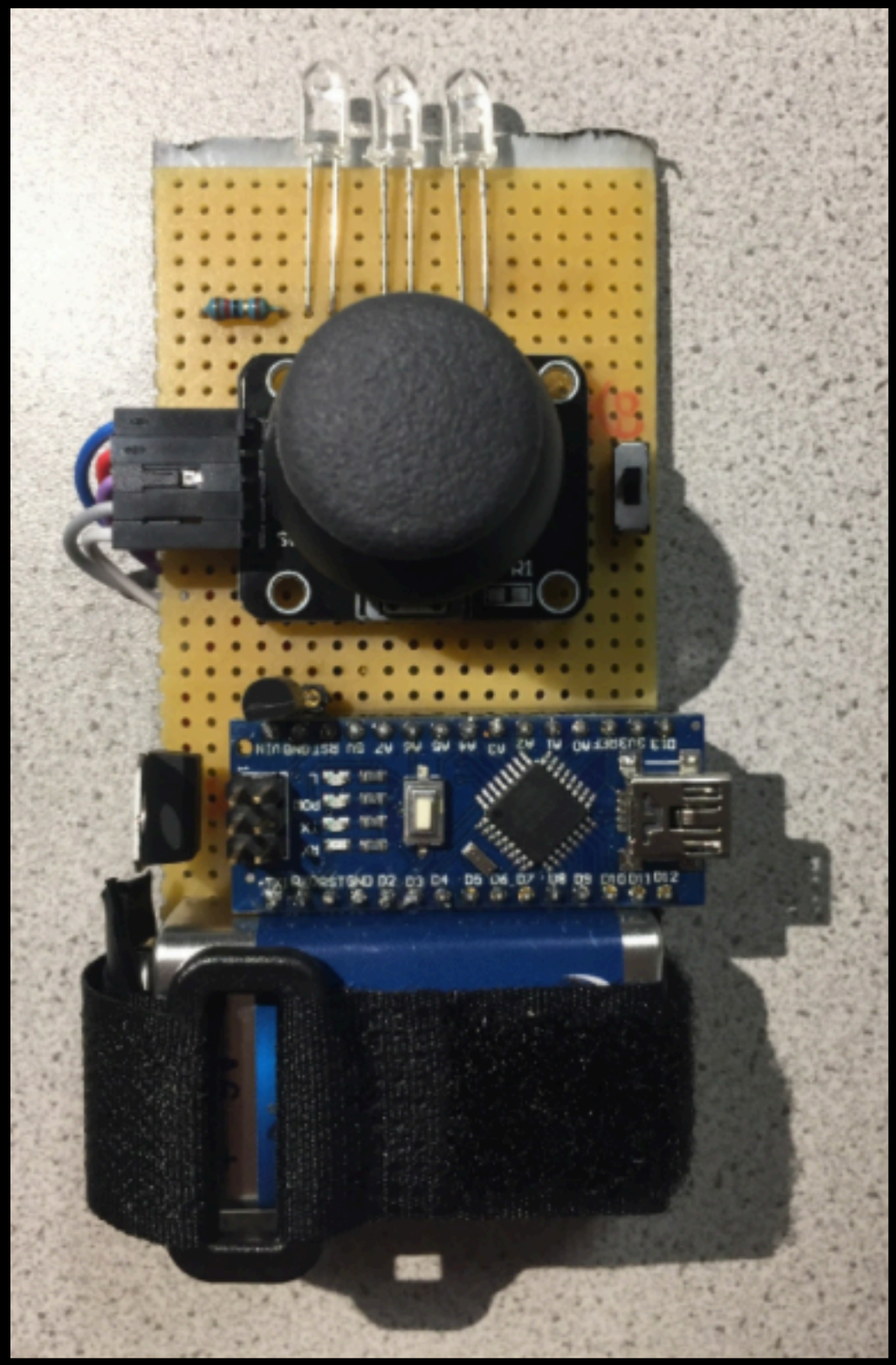


BS170 Mostet



3 IR Transmitter LEDs

Perfboard IR Remote



In depth look:

IR Senden

- 3 Leds mit Mosfet angesteuert
- Verwendeter Mosfet BS170

```
//if new signal, the start signal is transmitted first
//      ________
//     |        |
//     |        |
//     |        |________
//     <--------><-------->
//       4,5ms     4,5ms
//     <------------------>
//         START_SIGNAL

//if Bit = 1 toggle first quarter of signal
//       ________
//      |        |
//      |        |
//      |        |
//      |        |_____________________________
//     <--------><--------><--------><-------->
//       577,5us  577,5us  577,5us  577,5us
//     <------------------------------------->
//                   BIT = 0
```

Programmierung

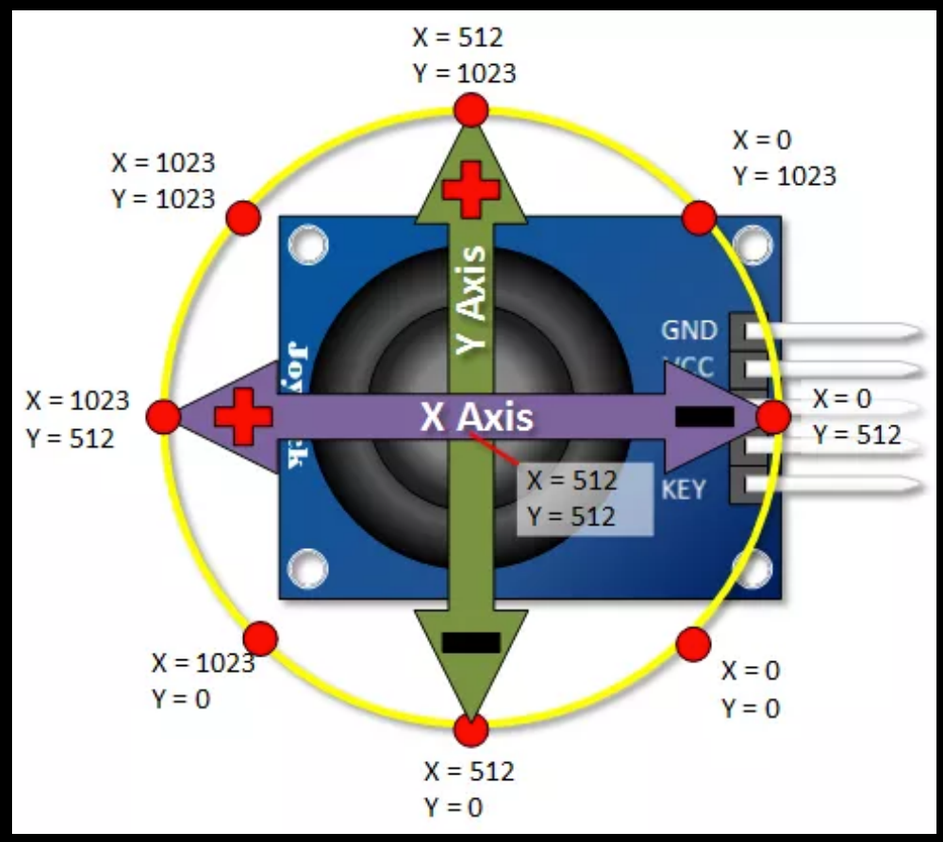
```
//if Bit = 0 --> toggle first half of signal
//       ________
//      |        |
//      |        |
//      |        |________
//     <--------><-------->
//       577,5us  577,5us
//     <------------------>
//           BIT = 0

//if all bits are transmitted, send end-burst
//       ________
//      |        |
//      |        |
//      |        |
//      |        |
//     <-------->
//       577,5us
//     <-------->
//      End-Burst
```

In depth look:

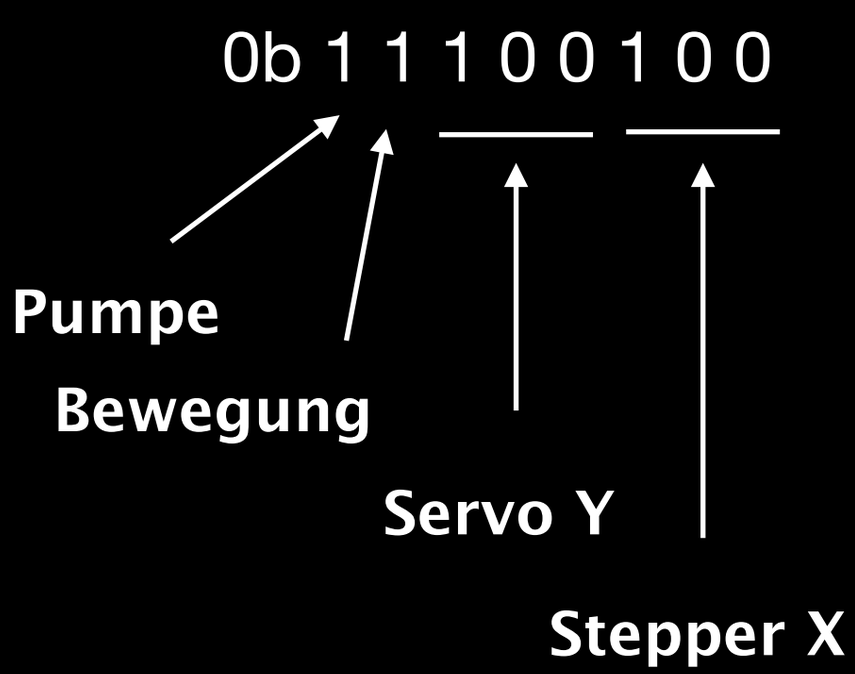
Joystick

- Wir fanden es ungenau
- Original PS2 Joystick (von China)
- Werte wurden über IR übertragen

2 x ADC



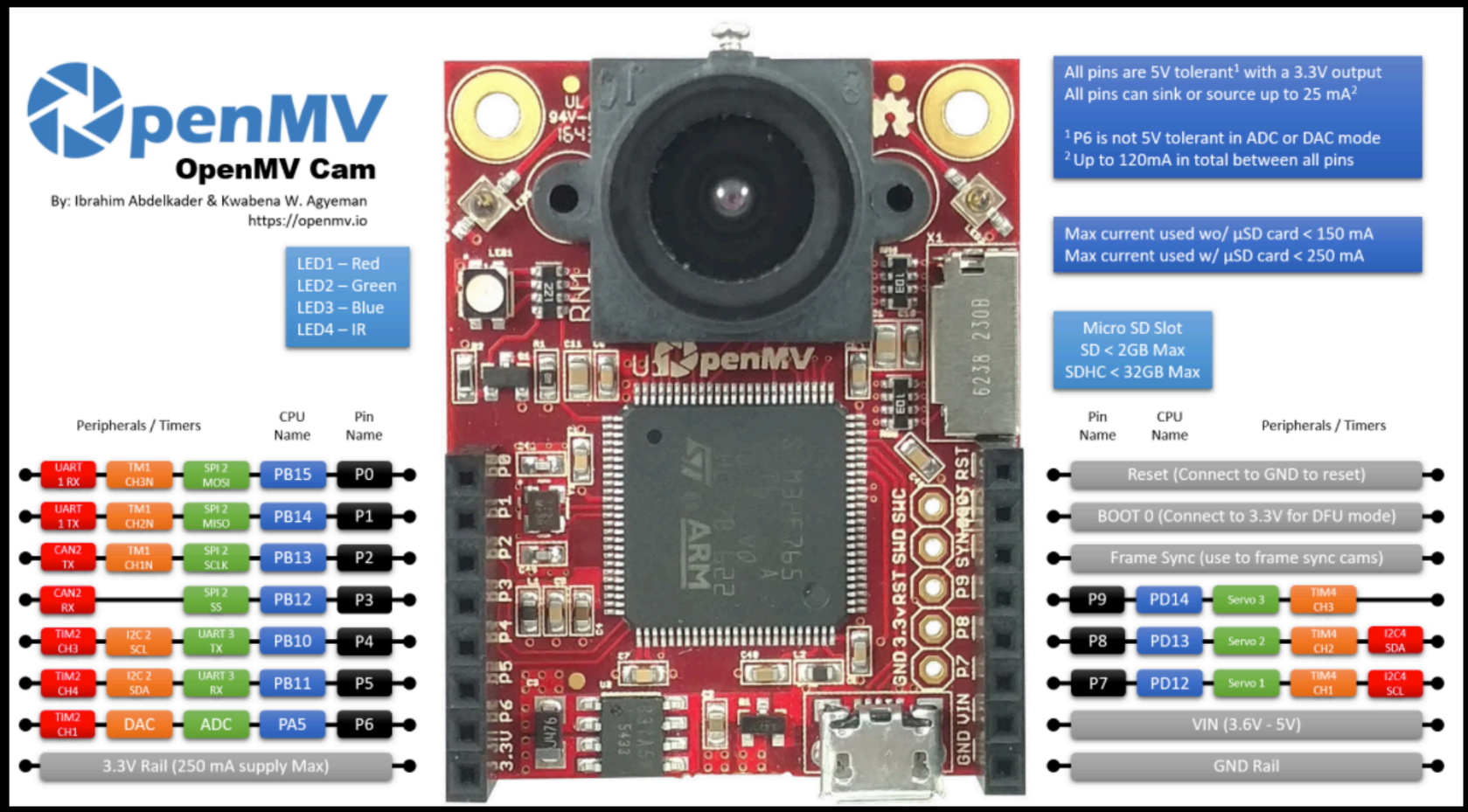
http://henrysbench.capnfatz.com/henrys-bench/arduino-sensors-and-input/arduino-ps2-joystick-tutorial-keyes-ky-023-deek-robot/



In depth problem look:

OpenMV 7 Camera

Für unsere Bedürfnisse eine zu schlechte Gesichtserkennung



https://openmv.io/

Investierte Zeit

- Michael: Zu viel
- Fabian: Viel zu viel